Politecnico di Torino

ScuDo
Scuola di Dottorato ~ Doctoral School
WHAT YOU ARE, TAKES YOU FAR

Doctoral Dissertation
Doctoral Program in Industry 4.0 (38.th cycle)

Efficient Adaptation of Large Language Models in Natural Language Processing

Marco Braga

* * * * *

Supervisors
Prof. Gabriella Pasi, Supervisor
Prof. Alessandro Raganato, Co-supervisor

Politecnico di Torino
April 30, 2026

This thesis is licensed under a Creative Commons License, Attribution - Noncommercial- NoDerivative Works 4.0 International: see www.creativecommons.org. The text may be reproduced for non-commercial purposes, provided that credit is given to the original author.

I hereby declare that, the contents and organisation of this dissertation constitute my own original work and does not compromise in any way the rights of third parties, including those relating to the security of personal data.

.....

Marco Braga

Turin, April 30, 2026

Summary

The rapid growth of Large Language Models (LLMs) has significantly improved performance across a wide range of Natural Language Processing (NLP) tasks, including Information Retrieval (IR). Despite their strong generalisation capabilities, LLMs still require domain- and task-specific fine-tuning to achieve competitive performance in highly specialised scenarios. LLMs exhibit some generalisation abilities, though these are not uniformly reliable across tasks and domains. In particular, adapting these models to downstream domains and tasks has become challenging as full fine-tuning is computationally expensive, memory-intensive and difficult to scale across multiple domains and tasks. Over the past few years, substantial research has been made into efficient fine-tuning of LLMs, resulting in various proposed approaches. Nevertheless, it remains an ongoing research area with several unresolved issues and challenges.

Motivated by this, in this dissertation we introduce novel efficient strategies for domain and task adaptation of LLMs, focusing on parameter efficient methods that reduce both training time and costs while preserving effectiveness. First, we provide a systematic review of Parameter-Efficient Fine-Tuning (PEFT) approaches, analysing how efficiency is defined and evaluated in the literature and identifying key open challenges. Building on these insights, we propose AdaKron, a novel adapter architecture based on Kronecker-product parametrisation, designed to increase expressiveness while updating less than 1% of model parameters. We further extend this approach with MAdaKron, which integrates a Mixture-of-Experts mechanism into AdaKron to improve adaptation quality through expert routing without increasing computational overhead. To address multi-domain information retrieval, we introduce DRAMA, a parameter-efficient retrieval framework based on domain-specific adapters and an adaptive gating mechanism that dynamically allocates model capacity depending on the input query domain.

In addition, to go beyond parameter-efficient approaches, we explore training-free model composition techniques, studying Task Arithmetic and task vectors for zero-shot retrieval adaptation and showing how domain- and language-specific retrieval models can be obtained through model merging without additional fine-tuning. Furthermore, through an analysis of task vectors, this thesis investigates what neural ranking models learn when fine-tuned on query-document pairs. Our

analysis reveals that prompt-token embeddings encode a large portion of the relevance adaptation signal in T5-based rerankers, enabling a lightweight adaptation strategy that achieves effectiveness comparable to full fine-tuning. Finally, we contribute new large-scale resources for reproducible evaluation, including SE-PQA, a dataset for personalized community question answering, and its synthetic extension generated with LLMs.

Overall, this dissertation advances the development of scalable and sustainable domain adaptation methods for LLMs, combining structured parameter-efficient architectures, dynamic retrieval adaptation, training-free model merging, and benchmark resources to support future research in efficient NLP and IR.

Acknowledgements

This thesis marks the conclusion of three years of research and dedicated effort. Its completion would not have been possible without the encouragement, guidance, and support of many individuals. First and foremost, I wish to express my sincere gratitude to my supervisors, Gabriella Pasi and Alessandro Raganato. Their expertise, insightful advice, and constant encouragement have accompanied me throughout this work, providing both intellectual challenges and valuable inspiration. Their strong commitment to research excellence and their extensive technical knowledge have significantly influenced my academic development. I am particularly thankful for their patience, constructive feedback, and continuous support, which have been essential to my growth as a researcher. I would also like to thank Craig Macdonald and Sean MacAvaney for welcoming me during my period abroad in Glasgow and for making me feel part of their research group. Their hospitality and support made that experience both productive and enriching.

I am deeply grateful to my family. In particular, I thank my parents for their unconditional love, sacrifices, and unwavering encouragement, which have always been the foundation of my journey. I am also grateful to my grandparents, who have consistently supported me and believed in my potential. Their trust and presence have been a constant source of strength.

Finally, I would like to extend my heartfelt appreciation to my friends and colleagues, who have accompanied me throughout this path with encouragement, motivation, and companionship. Their support has made this journey not only achievable but also genuinely rewarding. To all those who have contributed in any way to this work, I offer my sincere thanks. This achievement would not have been possible without your collective support and confidence in me.

Publications

Within the completion of this dissertation, several publications pertinent to the research content have been produced, some of which are already published or are currently submitted for peer review.

- Marco Braga, Alessandro Raganato, Gabriella Pasi. AdaKron: An Adapter-based Parameter Efficient Model Tuning with Kronecker Product. In Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024). pp. 350-357. ELRA and ICCL. [47]

  Discussed in Chapter 5.

- Pranav Kasela, Marco Braga, Gabriella Pasi, Raffaele Perego. "SE-PQA: Personalized Community Question Answering." In Companion Proceedings of the ACM on Web Conference 2024. pp. 1095-1098. Association for Computing Machinery. [225]

  This work was conducted in collaboration with Dr. Pranav Kasela. My main contributions include training the models on the proposed datasets and writing the paper. Discussed in Chapter 11.

- Pranav Kasela, Marco Braga, Gabriella Pasi, Raffaele Perego. "SE-PQA: StackExchange Personalized Community Question Answering." In Proceedings of the 14th Italian Information Retrieval Workshop, 2024. pp. 99-102. CEUR-WS. [227]

  This work was conducted in collaboration with Dr. Pranav Kasela. My main contributions include training the models on the proposed datasets and writing the paper. Discussed in Chapter 11.

- Marco Braga, Pranav Kasela, Alessandro Raganato, Gabriella Pasi. "Synthetic Data Generation with Large Language Models for Personalized Community Question Answering." 2024 IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT). pp. 360-366. IEEE. [53]

This work was conducted in collaboration with Dr. Pranav Kasela. My main contributions include defining the new approach, generating and analysing the synthetic dataset, as well as writing the paper. Discussed in Chapter 12.

- Marco Braga, Pranav Kasela, Alessandro Raganato, Gabriella Pasi. "Investigating Task Arithmetic for Zero-Shot Information Retrieval." In Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024. pp. 2738-2743. Association for Computing Machinery. [51]

  This work was conducted in collaboration with Dr. Pranav Kasela. My main contributions included contributing to the design and development of the approach, conducting its experimental evaluation, analysing the results, and writing the paper. Discussed in Chapter 8.

- Marco Braga, Pranav Kasela, Alessandro Raganato, Gabriella Pasi. "Investigating Task Arithmetic for Zero-Shot Information Retrieval." In Proceedings of the 15th Italian Information Retrieval Workshop, 2025. pp. 108-116. CEUR-WS.

  This work was conducted in collaboration with Dr. Pranav Kasela. My main contributions included contributing to the design and development of the approach, conducting its experimental evaluation, analysing the results, and writing the paper. Discussed in Chapter 8.

- Marco Braga, Alessandro Raganato, Gabriella Pasi. "MAdaKron: a Mixture-of-AdaKron Adapters." 2026, Knowledge-Based Systems. Volume 334, 115086. Elsevier. [48]

  Discussed in Chapter 5.

- Marco Braga, Sean MacAvaney, Craig Macdonald, Gabriella Pasi. "Revealing MonoT5's Learning Mechanisms via Prompt-Token Adaptation." In European Conference on Information Retrieval (ECIR), 2026. pp. 450-465. Springer Nature Switzerland. [52]

  This work was carried out during a three-month research period at the University of Glasgow, under the supervision of Prof. Sean MacAvaney and Prof. Craig Macdonald. Discussed in Chapter 9.

- Pranav Kasela, Marco Braga, Gabriella Pasi, Raffaele Perego, Nazli Goharian, Ophir Frieder. "DRAMA: Domain Retrieval using Adaptive Module Allocation.", 2026, Submitted for peer review. [222]

  This work was conducted in collaboration with Dr. Pranav Kasela. My main contributions include extending DRAMA to additional approaches and datasets, as well as writing the paper. Discussed in Chapter 6.

- Marco Braga, Alessandro Raganato, Gabriella Pasi. "Parameter Efficient Fine-Tuning Techniques for Natural Language Processing: a Systematic Review.", 2026, Submitted for peer review.

  Discussed in Chapter 3.

Other research publications related to NLP and IR:

- Marco Braga, Alessandro Raganato, Gabriella Pasi. "Personalization in BERT with Adapter Modules and Topic Modelling." In Proceedings of the 13th Italian Information Retrieval Workshop, 2023. Vol. 3448, pp. 24-29. CEUR-WS. [50]

- Lia Morra, Alberto Azzari, Letizia Bergamasco, Marco Braga, Luigi Capogrosso, Federico Delrio, Giuseppe Di Giacomo, Simone Eiraudo, Giorgia Ghione, Rocco Giudice, Alkis Koudounas, Luca Piano, Daniele Rege Cambrin, Matteo Risso, Marco Rondina, Alessandro Sebastien Russo, Marco Russo, Francesco Taioli, Lorenzo Vaiani and Chiara Vercellino. "Designing Logic Tensor Networks for Visual Sudoku puzzle classification." In Proceedings of the 17th International Workshop on Neural-Symbolic Learning and Reasoning, 2023. CEUR-WS, 2023. pp. 223-232. CEUR-WS. [321]

- Marco Braga. "Personalized large language models through parameter efficient fine-tuning techniques." In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024. pp. 3076-3076. Association for Computing Machinery. [44]

- Pranav Kasela, Marco Braga, Effrosyni Sokli, Gian Carlo Milanese, Georgios Peikos, Sandip Modha, Alessandro Raganato, Marco Viviani, Gabriella Pasi. "Overview of the PIR Track at FIRE 2024: Evaluation of Personalised Information Retrieval." In Proceedings of the 16th annual meeting of the Forum for Information Retrieval Evaluation, 2024. pp. 11-13. Association for Computing Machinery. [224]

- Pranav Kasela, Marco Braga, Effrosyni Sokli, Gian Carlo Milanese, Georgios Peikos, Sandip Modha, Alessandro Raganato, Marco Viviani, Gabriella Pasi. "Overview of the PIR Track at FIRE 2024: Evaluation of Personalised Information Retrieval.", In Proceedings of the 16th annual meeting of the Forum for Information Retrieval Evaluation, 2024. pp. 423-426. CEUR-WS. [223]

- Marco Braga, Gian Carlo Milanese, Gabriella Pasi. "Investigating Large Language Models' Linguistic Abilities for Text Preprocessing." 2025 IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), 2025. [46]

- Pranav Kasela, Marco Braga, Alessandro Ghiotto, Andrea Pilzer, Marco Viviani, Alessandro Raganato. "DIETA: A Decoder-only Transformer-based Model for Italian-English Machine TrAnslation." In the Proceedings of the Eleventh Italian Conference on Computational Linguistics (CLiC-it 2025). pp. 539-549. CEUR WS [221]

- Andrea Tirico, Marco Braga, Emiliana Murgia, Gabriella Pasi. "Modelling Children's Language: Vikidia-based Resources to Support Italian Text Simplification." In Proceedings of IEEE Conference on Artificial Intelligence 2026 (IEEE CAI). [446]

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This Chapter introduces the research context, motivation, and objectives of the research undertaken during the Ph.D.. It outlines the main challenges addressed in our work, formulates the research questions, and summarises the core contributions. Finally, it provides an overview of the dissertation structure. This Chapter sets the stage for a comprehensive understanding of the research presented in this dissertation.

1.1 Research Context

In recent years, Natural Language Processing (NLP), which is defined as the automatic processing and understanding of human language, has undergone a profound transformation with the introduction of the transformer architecture [454]. Since their first appearance in 2017, the transformers have rapidly become the dominant paradigm for modelling textual data, outperforming earlier neural architectures, such as long short-term memory networks [177]. This performance gain is largely attributed to the self-attention mechanism, which enables the efficient modelling of long-range dependencies [454]. Transformer-based language models are typically pre-trained on large-scale unlabelled corpora using self-supervised learning objectives, such as masked language modelling [102]. Through pre-training, models learn linguistic representations capturing lexical semantics, syntactic regularities, and contextual dependencies. Once pre-trained, these models can be adapted to downstream tasks by fine-tuning on small labelled datasets. Examples of downstream tasks include text classification [120, 136], semantic similarity [461], text generation [406, 247], question answering [89], and information retrieval [116, 379]. This paradigm, commonly referred to as transfer learning [185], has reduced the dependence on large task-specific labelled datasets.

Recent work [216] shows that the performance of transformer-based language models scales with model size: models with a larger number of parameters achieve

better performance than smaller counterparts. This observation has driven an exponential growth in the size of Large Language Models (LLMs), with parameter counts increasing from millions to hundreds of billions within a few years, as illustrated in Figure 1.1. However, this rapid growth has introduced substantial computational and memory challenges, particularly when fine-tuning models on downstream tasks. Practical constraints related to GPU memory are already noted in the official BERT documentation[1], which reports that training BERT-large [102] with a 512-token context window on a GPU with 12GB of memory results in an effective batch size of zero. At the same time, while model sizes have reached up to 170 billion parameters [54], GPU memory capacity has not increased proportionally to the number of parameters, due to the high costs associated with hardware development and deployment [256].

In both NLP and IR, a *task* is defined as a specific activity that maps input to output, such as classification, ranking, or question answering, characterised by a target prediction function and an evaluation protocol [74]. In contrast, a *domain* refers to the underlying data distribution associated with a particular application setting, encompassing topics, stylistic characteristics, domain-specific vocabulary, and linguistic conventions [505]. While billion-scale transformer language models show strong generalisation across a wide range of tasks [378], their adaptation to unseen tasks or specific domains, such as legal or medical, remains challenging due to distributional shifts, vocabulary mismatches, and domain-specific language [151]. In practice, task- or domain-specific adaptation is often performed through full fine-tuning, which updates all model parameters [102]. Although effective, this strategy is computationally expensive, memory-intensive, and data-hungry [256]. These limitations become particularly important in scenarios where a single backbone model must support multiple tasks or domains and new tasks or domains are introduced incrementally, as is common in real-world Information Retrieval systems [441]. Repeated full fine-tuning not only increases training costs but also results in duplicated model parameters across tasks.

Despite these limitations, billion-scale language models exhibit strong generalisation capabilities [375], enabling them to transfer knowledge to tasks that were not explicitly observed during training. In particular, such models can be applied to new tasks without any parameter updates, the so-called zero-shot learning [375], or by conditioning on a small number of task-specific examples provided directly in the input context, called few-shot learning [136]. These properties, broadly referred to as 'in-context learning', play an important role in the widespread adoption of large language models across various application domains.

However, the effectiveness of in-context learning is inherently constrained by the limited context window of current language models, which restricts the number

[1] https://github.com/google-research/bert

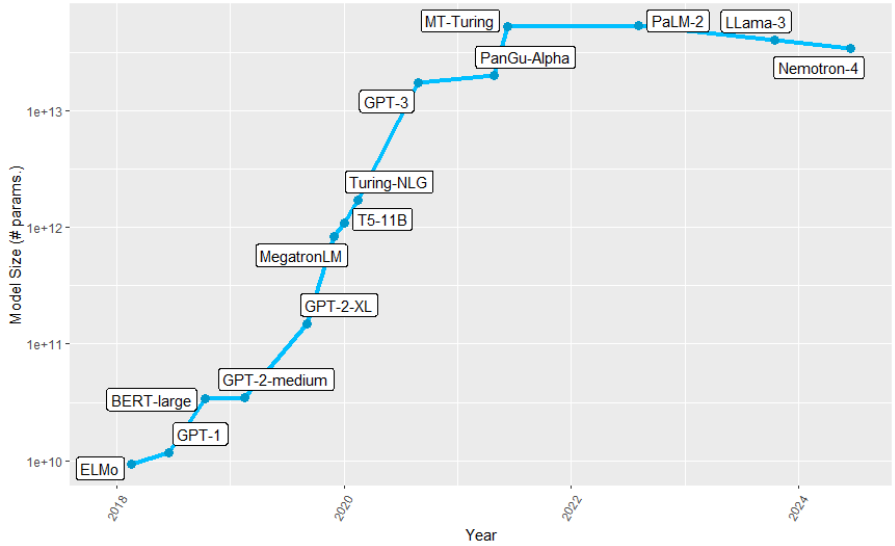

Figure 1.1: Exponential growth in the number of parameters of language models. The y-axis is logarithmic and reports the number of parameters in millions.

of examples that can be included, typically at most one hundred instances [54]. Moreover, increasing the context length leads to quadratic growth in computational and memory costs during inference [256], limiting the scalability of this approach. Beyond efficiency issues, recent empirical studies [115, 269] show that, for tasks such as text classification, fine-tuning small models (e.g. RoBERTa [279], with 355 million parameters) can outperform few-shot billion-parameter models such as LLaMA [449] or GPT-3.5 [54]. These findings highlight that task-specific fine-tuning is still important for achieving strong downstream performance, while simultaneously underscoring the need for efficient adaptation strategies.

In response to these issues, research has increasingly focused on Parameter-Efficient Fine-Tuning (PEFT) methods [256], which recognise that effective adaptation does not require modifying all parameters of a pre-trained model [256]. Rather than updating all model parameters, PEFT techniques adapt pre-trained language models by training only a small subset of parameters [31] or by introducing lightweight and task-specific modules [353, 186]. This design significantly reduces computational and memory costs while preserving or even improving fine-tuning effectiveness. These methods enable efficient reuse of a single backbone model across multiple tasks and domains, substantially reducing training cost and storage requirements while achieving performance comparable to full fine-tuning. Despite their success, existing PEFT approaches still face open challenges. Most methods require task-specific training data, limiting their applicability in zero-shot settings. Moreover, task adaptations are typically learned without explicitly modelling how task-specific knowledge can be reused or composed across tasks.

1.2 Research Questions

Our research aims to address the following research questions:

- **RQ1**: How is the notion of efficiency defined and analysed across various natural language processing and information retrieval tasks?
- **RQ2**: How are parameter efficient fine-tuning techniques defined, categorized and evaluated in the natural language processing literature?
- **RQ3**: How can structured adapter parametrisation, such as Kronecker-based adapters, enhance model expressiveness while preserving parameter-efficiency?
- **RQ4**: How can dynamic module allocation and domain-specific adapters improve multi-domain information retrieval performance while minimising resource consumption?
- **RQ5**: Can training-free merging approaches, such as Task Arithmetic, effectively define domain-specific retrieval models in zero-shot scenarios?
- **RQ6**: While a model is trained on an Information Retrieval task, what is the model learning? Can we train a specific subset of parameters to achieve performance comparable to fine-tuning?
- **RQ7**: How can the development of large-scale, personalised data allow reproducible research and scalable training for efficient IR systems across multiple domains?

The previously mentioned primary research questions are further broken down into sub-questions, each presented and addressed in the subsequent Chapters of this dissertation.

1.3 Research Contributions

The contributions of our research are the following:

- A comprehensive and systematic review of the literature covering 174 studies, which has been conducted to identify the current state of research on efficiency in NLP. This review aims to enhance our understanding of how efficiency has been defined and how parameter efficient approaches are categorized and evaluated across various application domains and tasks. The findings provide practical guidance for the development of novel efficient approaches.

- The definition of a new parameter-efficient adapter architecture that enhances representational expressiveness while maintaining or even improving adapter efficiency and efficacy on downstream tasks. The proposed approach, called AdaKron, incorporates the Kronecker product into an adapter module while updating less than 1% of the original model parameters. Extensive experiments show that AdaKron consistently improves effectiveness and efficiency compared to state-of-the-art PEFT approaches.

- An expansion of AdaKron that incorporates a Mixture-of-Experts architecture into AdaKron enhance the adapter's representational capabilities. The results show that combining structured parametrisation with expert routing further improves adaptation quality without increasing computational overhead.

- We define DRAMA, a parameter-efficient retrieval architecture that leverages domain-specific adapters and adaptive module allocation through a gating mechanism. DRAMA distils knowledge from specialised retrievers while reducing parameter redundancy, storage requirements, and energy consumption. Experimental findings show that dynamic domain adaptation improves effectiveness while maintaining efficiency constraints.

- The application of task arithmetic to define domain- and language-specific retrieval models without requiring any additional training. The proposed approaches enable zero-shot composition of dense, sparse and re-ranking models, improving performance in specialized domains, tasks and languages. This contribution improves the notion of efficiency by reducing the need for further training and optimisation.

- The identification of which parameters contribute most to the adaptation of language models to a retrieval task. Our findings reveal that a subset of embedding layers encodes a significant portion of the learning signal. Based on this insight, we propose an efficient adaptation strategy that updates only a targeted subset of parameters, achieving performance comparable to full fine-tuning while drastically reducing the number of parameters updated.

- We define a large-scale personalised and synthetic datasets designed to support reproducible experimentation and scalable model training across different domains.

1.4 Thesis Organization

The subsequent sections of this dissertation are divided into five main parts, as outlined below.

Part I: Background

This part is organised into three Chapters and establishes the conceptual and methodological foundations of this dissertation. It introduces the fundamental principles of Natural Language Processing and Information Retrieval, which constitute the two core pillars of our research. Particular emphasis is placed on transformer-based architectures, evaluation methodologies, parameter efficient techniques and domain adaptation. In addition, this part discusses recent advances in Large Language Models and frames efficiency as a central research challenge at the intersection of NLP and IR.

Chapter 2 – Foundational Concepts and Research Methods

This Chapter provides the necessary background to contextualise our research. It reviews the main components of modern NLP, IR and search engine architectures. Furthermore, it discusses standard evaluation methodologies, experimental protocols, and training strategies, with a particular focus on balancing effectiveness and efficiency.

Chapter 3 – Parameter Efficient Approaches for training Large Language Models: a Systematic Review

This Chapter presents a systematic literature review of parameter efficient fine-tuning approaches for large language models. The survey analyses current trends in efficient adaptation of LLMs, and identifies open challenges related to their evaluation, scalability, computational cost, and sustainable deployment.

1.4.1 Part II: Adapter-based Models for Efficient Natural Language Processing and Information Retrieval

This part presents the novel parameter efficient approaches developed during the Ph.D. to address efficiency and scalability challenges in NLP and Information Retrieval. It focuses on structured adapter design and domain-adaptive retrieval under strict parameters and energy constraints.

Chapter 5 - MAdaKron: A Mixture-of-AdaKron Adapters

This Chapter introduces AdaKron, a Kronecker-based adapter architecture designed to enhance expressiveness while training less than 1% of model parameters. We evaluate AdaKron across multiple benchmarks and models, showing performance improvements over full fine-tuning and state-of-the-art PEFT approaches.

Building upon this foundation, the Chapter also presents MAdaKron, a Mixture-of-Experts extension that dynamically routes inputs to specialised adapter components. The integration of structured parametrisation and expert routing further improves effectiveness while preserving parameter efficiency.

Chapter 6 - DRAMA: Domain Retrieval using Adaptive Module Allocation

This Chapter introduces DRAMA, a dynamic and parameter-efficient retrieval framework for multi-domain information retrieval. DRAMA combines domain-specific PEFT components with a gating mechanism to allocate model capacity based on the input domain. The Chapter investigates whether domain-specific retrievers outperform multi-domain models and evaluates the impact of adaptive module allocation on effectiveness, storage requirements, and energy consumption.

1.4.2 Part III: Task Arithmetic for Efficient Information Retrieval

This part explores efficiency beyond reducing the number of trained parameters by investigating training-free model composition and analysing learning dynamics in neural retrieval models.

Chapter 8 - Task Arithmetic for Zero-Shot Information Retrieval

This Chapter studies Task Arithmetic and model-merging techniques for zero-shot domain adaptation in Information Retrieval. The proposed approach enables to go beyond parameter-efficient fine-tuning by the definition of domain-specific retrievers without additional training. The evaluation covers dense, sparse, and neural re-ranking retrieval models, showing the effectiveness of model composition approaches in domain-specific scenarios.

Chapter 9 - Revealing MonoT5's Learning Mechanisms via Prompt-Token Adaptation

This Chapter further analyses task vectors to understand how neural IR models learn relevance signals during fine-tuning. The study reveals that prompt token embeddings capture a significant portion of the adaptation signal in T5-based retrieval models. Motivated by this finding, a lightweight adaptation strategy has been defined that updates only prompt-token embeddings, achieving performance comparable to full fine-tuning while reducing training costs. Furthermore, we show how the retrieval performance of T5-based models can be disentangled into the classification of low quality documents, i.e. those that are unlikely to be relevant to any queries, and the learning of the relevance signals.

1.4.3 Part IV: Resources and Benchmarks

This part introduces the datasets designed to support reproducible and scalable research in efficient IR across different domains.

Chapter 11 - SE-PQA: Personalized Community Question Answering

This Chapter presents SE-PQA, a large-scale dataset for personalised community Question Answering. The dataset includes millions of questions and answers enriched with social interaction features. The Chapter describes the construction process, feature representations, and reproducible baselines, showing the impact of personalisation on a Web platform.

Chapter 12 - Synthetic Data Generation with Large Language Models for Personalised cQA

This Chapter introduces Sy-SE-PQA, a synthetic extension of SE-PQA generated using Large Language Models. It describes the prompting strategies, data generation pipeline and baselines, showing that training neural retrieval models on synthetic data outperforms models trained exclusively on human-generated data.

1.4.4 Part V: Conclusions and Insights

Chapter 13 - Conclusions

This final Chapter summarises the main contributions of the research undertaken during the Ph.D. and discusses their broader implications for efficient and domain-adaptive NLP and IR. It highlights open research challenges related to scalability, robustness, zero-shot adaptability, and the sustainable deployment of LLMs, and outlines promising directions for future work.

Part I

Background

Chapter 2

Foundational Concepts and Research Methods

This Chapter introduces the foundational concepts and methodological principles underlying this dissertation, with a particular focus on Natural Language Processing (NLP) and Information Retrieval (IR). It provides a structured overview of language modelling and transformer architectures, and subsequently discusses the main components of modern IR systems, including document representation, ranking functions, and effectiveness evaluation. By outlining the theoretical and practical foundations of both areas, this Chapter establishes the conceptual framework necessary to understand the new models and approaches proposed in the subsequent Chapters. It serves as a reference point for interpreting experimental findings and establishing our contributions within the broader landscape of NLP and IR research.

2.1 Natural Language Processing

The question *"Can a machine think?"* can be traced back to the early foundations of Artificial Intelligence, most notably to the work of Alan Turing, who proposed in 1950 what later became known as the *Turing Test* as an operational criterion for machine intelligence [451]. According to this formulation, a computer may be regarded as exhibiting intelligent behaviour if, during a text-based interaction, a human evaluator is unable to reliably distinguish its responses from those of a human interlocutor. Rather than providing a formal definition of intelligence, the test emphasises observable behaviour, and in particular the ability of machines to engage in meaningful and coherent linguistic interaction. More broadly, Artificial Intelligence (AI) encompasses a wide range of computational techniques aimed at enabling machines to emulate aspects of human cognition and behaviour. These capabilities include Natural Language Processing (NLP) for understanding

and generating human language, automated reasoning for answering complex questions, and machine learning methods that allow systems to learn from data and identify patterns. In addition, AI includes capabilities such as computer vision and speech recognition, as well as robotics, which enables physical interaction with the environment [402]. While AI spans multiple modalities and application domains, this dissertation focuses specifically on methods for processing textual data.

Natural Language Processing is the branch of AI devoted to the automatic processing, analysis, and understanding of human language [30]. The origins of NLP can be traced back to the 1950s, when early research efforts were primarily focused on machine translation and question answering, motivated by the ambition to enable computers to manipulate language [30]. Over time, the field has undergone several paradigm shifts. Early NLP systems were largely rule-based, relying on manually crafted grammars and linguistic rules. These approaches were later superseded by statistical methods, which modelled linguistic phenomena using probabilistic frameworks and large text corpora. More recently, NLP has been profoundly influenced by machine learning approaches, and in particular by deep learning techniques, which allow models to learn complex linguistic representations directly from data [18]. A central concept in NLP is that of a *Language Model* (LM). A statistical language model can be defined as a probability distribution over sequences of tokens, assigning a likelihood to any possible sentence in a language. The term *token* is used rather than *word*, since modern language models typically operate on subword units, which may correspond to whole words, word fragments, symbols, or individual characters [326]. A well-trained statistical language model assigns higher probability to syntactically and semantically plausible sentences, while assigning lower probability to poorly formed or incoherent word sequences. By estimating the probability of the next token given the preceding context, language models can be used to predict likely continuations of a text, thereby enabling applications such as sentence completion, text generation, and speech recognition. When a language model is explicitly designed to generate fluent and coherent text, it is commonly referred to as a *generative model*.

In recent years, the introduction of the transformer architecture [454] has fundamentally reshaped the field of NLP. Transformers rely on self-attention mechanisms to model dependencies between tokens in a sequence, enabling them to capture long-range contextual information more effectively than previous sequential architectures. This innovation has established transformers as the dominant paradigm for building modern Large Language Models (LLMs), which are typically pre-trained on massive text corpora in order to acquire general-purpose linguistic representations and subsequently adapted to a wide range of downstream tasks [102]. These developments constitute the foundation for the methods and models investigated throughout this thesis.

2.2 Transformer-based Language Models

As illustrated in Figure 2.1, a Transformer layer is composed of a self-attention mechanism followed by a fully connected Feed-Forward Network (FFN). Both attention and feed-forward sub-layers are equipped with residual connections [162] and are followed by normalization layers [16], which improve training stability and optimisation efficiency. This section introduces the main components of transformer models and establishes the notation used throughout the remainder of the dissertation.

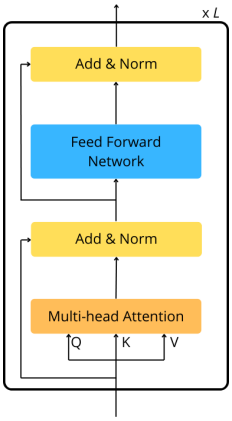

Figure 2.1: A transformer block, as defined by Vaswani et al. [454].

Attention Mechanism The attention mechanism [454] is the core operation that allows transformers to model contextual dependencies within a sequence. It is parametrised by three matrices: query $Q \in \mathbb{R}^{n \times d_k}$, key $K \in \mathbb{R}^{m \times d_k}$, and value $V \in \mathbb{R}^{m \times d_v}$. Given an input representation $X \in \mathbb{R}^{n \times d}$, these matrices are computed as:

$$Q = XW_q, \quad K = XW_k, \quad V = XW_v, \tag{2.1}$$

where $W_q \in \mathbb{R}^{d \times d_k}$, $W_k \in \mathbb{R}^{d \times d_k}$, and $W_v \in \mathbb{R}^{d \times d_v}$ are learnable weight matrices. The attention output is defined as:

$$H = Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{2.2}$$

To increase representational capacity, transformers employ multi-head attention, which applies the attention operation in parallel across multiple subspaces. Each attention head uses distinct projection matrices, and the resulting outputs are concatenated and linearly transformed:

$$H = MultiHeadAttention(Q, K, V) = Concat(H_1, \ldots, H_h)W_o, \tag{2.3}$$

where h denotes the number of heads and W_o is a learnable output projection.

Multi-head attention improves representational capacity by enabling the model to attend to different contextual subspaces in parallel. This property is particularly important in retrieval and semantic matching settings, where capturing fine-grained interactions between query terms and document context is essential. Moreover, several parameter-efficient fine-tuning approaches, such as LoRA [186], modify attention projection matrices directly, suggesting that attention layers constitute a critical adaptation component when transferring large pre-trained models to downstream tasks.

Fully-connected Feed-Forward Network Following the attention sub-layer, each transformer block includes a position-wise feed-forward network. This network consists of two fully connected layers with an intermediate non-linear activation function, typically ReLU [139] or GeLU [173]. For an input $X \in \mathbb{R}^{n \times d_i}$, the FFN output is computed as:

$$F = \sigma(XW_1 + b_1)W_2 + b_2, \tag{2.4}$$

where W_1, W_2, b_1, and b_2 are learnable parameters.

Residual Connections and Normalization Layers Residual connections [162] and layer normalization [16] are applied after each attention and feed-forward sub-layer to mitigate vanishing gradients and stabilise training. Given a function $f(\cdot)$, this operation is defined as:

$$Add\&Norm(X, f) = LayerNorm(X + f(X)). \tag{2.5}$$

Transformer-based language models can be grouped into three major architectural categories, each addressing different NLP purposes:

- **Encoder-only**: these models, such as BERT [102], rely on bidirectional self-attention to capture contextual dependencies from both directions. Pre-training tasks typically include Masked Language Modelling (MLM) and Next Sentence Prediction (NSP), using special tokens like $[MASK]$. The MLM objective seeks to predict masked tokens based on surrounding context:

$$\mathcal{L}_{MLM} = -\sum_{x_m \in M(x)} log(\mathbf{P}(x_m | x_{\setminus M(x)}), \tag{2.6}$$

  where $M(x)$ denotes the set of masked token positions. Variants such as RoBERTa [279] and DeBERTa [165, 163] introduce different pre-training and tokenization strategies or new attention mechanisms.

- **Decoder-only**: they adopt an autoregressive setup, where tokens are generated sequentially, conditioning each prediction on previously generated tokens. The training objective models the log-likelihood of sequences as:

$$\mathcal{L}_{LM} = -log(\mathbf{P}(x)) = -\sum_{i=1}^{T} log(\mathbf{P}(x_i|x_{<i}), \tag{2.7}$$

  where x_0 denotes a special token indicating the start of a sequence. GPT-based models [375, 54] exemplify this class, using causal attention to predict the next token. These models have been shown to be especially effective in generative tasks such as dialogue systems, summarization, and translation.

- **Encoder-decoder**: they integrate both encoder and decoder modules to address tasks requiring both understanding and generation. In this setup, the encoder processes the input into a contextual representation, which the decoder then uses to produce output tokens. Examples include T5 [376], which frames all tasks as text-to-text transformations, and BART [244], which combines bidirectional encoding with autoregressive decoding. These models often use span corruption during pre-training, reconstructing masked spans from the context. The associated training objective is:

$$\mathcal{L}_{Seq2Seq\ MLM} = -\sum_{x_{i:j} \in M(x)} \sum_{t=i}^{j} log(\mathbf{P}(x_t|x_{\setminus M(x)}, x_{i:t-1}), \tag{2.8}$$

  where $M(x)$ is the set of corrupted spans. Different from encoder-only architectures, the decoder additionally incorporates a cross-attention mechanism, which enables it to attend to contextual representations produced by the encoder. Attention is masked to prevent access to future tokens, while cross-attention enables interaction between encoder and decoder representations. Multilingual variants such as mT5 [501] extend this architecture to support cross-lingual tasks.

Following [396], we consider every transformer-based model to be a Large Language Model (LLM).

2.3 Information Retrieval

The expression *Information Retrieval* (IR) was introduced in 1951 [319] by Mooers, who described Information Retrieval as the process through which a user is able to translate an information need into a list of stored documents that may contain relevant content: *"Information Retrieval is the name for the process or method whereby a prospective user of information is able to convert his need for information into an actual list of citations to documents in storage containing information*

useful to him (user). Information Retrieval embraces the intellectual aspects of the description of information and its specification for search, and also whatever systems, techniques, and machines that are employed to carry out the operation." Later work further clarified the key steps involved in retrieval systems. In particular, Salton [407] described IR as a process that requires analysing and representing stored items, organising them to support efficient search, and retrieving them in response to user queries. Salton emphasised four main aspects of IR system design: information analysis, organisation and search, query formulation, and retrieval and dissemination. The aim of an information retrieval system, often referred to as a search engine, is to support users in finding information items that are relevant to a given information need. Such needs are commonly expressed through a query, which may take different forms depending on the application scenario. In many cases, queries are short and consist of keywords or brief phrases; however, they may also include Boolean operators, structured formulations, or complex questions, depending on the model on which the system is based. Although IR research has traditionally focused on text documents, modern retrieval systems often support heterogeneous information items, including multimedia content such as images or videos.

A typical retrieval interaction begins when a user submits a query to the system. The system then searches within a collection to identify candidate documents that may satisfy the request. Since matching requires comparable representations, both the query and the stored documents must be mapped into a shared representation space. Candidate documents are subsequently ordered by a ranking function that estimates their relevance with respect to the query. Early retrieval systems relied primarily on lexical overlap and statistical weighting mechanisms, while more recent approaches have benefited from pre-trained language models, which are able to capture semantic similarity. In addition to textual similarity, ranking pipelines may incorporate further signals, such as metadata or popularity indicators. The final stage of the system presents a ranked list of information items to the user for further investigation. The outlined retrieval scenario corresponds to the so-called *ad-hoc retrieval*. Retrieval systems are also frequently designed for specific application domains, leading to what is commonly referred to as *domain-specific search*. Domain-specific retrieval has been defined [289] as search activity focused on a specific subject area, characterised by specialised vocabularies, multiple information modalities (e.g., text, images, audio), and diverse user groups and tasks. Typical examples include scientific literature search, biomedical retrieval, legal information systems, and financial document analysis. In such contexts, users often formulate information needs using terminology that is highly domain-dependent. Domain experts may rely on technical expressions, abbreviations, or structured queries aligned with professional standards. At the same time, non-expert users may also interact with specialised collections, for instance when consulting scientific publications or technical reports, often expressing similar needs with less precise language. As

a consequence, the design of effective domain-specific retrieval systems involves several challenges. First, the system must accommodate heterogeneous information needs, ranging from highly technical expert queries to exploratory searches by non-specialists. Second, it often requires the integration of domain terminologies, controlled vocabularies, or ontologies within indexing and ranking pipelines, in order to ensure accurate matching. Third, domain-specific systems may need to support retrieval over multiple document types, such as structured records, full-text publications, numerical tables, or metadata archives.

2.3.1 Information Retrieval Systems

Information retrieval systems aim to estimate relevance by automatically assessing how well a document satisfies an expressed information need. In this dissertation, we focus on text-based IR systems. A user provides a query q, which may consist of keywords, phrases, Boolean expressions, or full natural language sentences. Depending on the retrieval scenario, queries may range from short and underspecified to longer and more descriptive formulations. From a practical perspective, a text-based retrieval system can be decomposed into two main components: an *offline* one, responsible for processing the document collection, and an *online* one, responsible for responding to user queries in real time.

Offline Component

The offline component is responsible for maintaining the document collection and building the index structures required for retrieval. It is referred to as offline since it typically involves computationally expensive operations that are not executed at query time. The offline pipeline is updated periodically as new documents are added or existing ones are modified. The update frequency depends on the application domain; for example, web search engines require frequent re-indexing due to the continuous evolution of web content.

The offline component generally includes the following elements:

- *Document Archive*: the repository where information items are stored, potentially including text documents, multimedia content, or mixed formats.
- *Indexing Process*: the procedure that extracts features from documents and stores them into dedicated index structures. In text retrieval, extracted features are often term-based. These features are organised to support efficient retrieval at query time.
- *Document Representation*: the formal representation used to encode document content. The choice of representation is closely tied to the retrieval model and has a direct impact on ranking effectiveness.

Online Component

The online component is responsible for handling user interaction and producing low latency ranked outputs. It includes the following elements:

- *Query*: the user's input expressing an information need.
- *Query Representation*: the transformation of the query into a representation compatible with the indexing structures.
- *Matching Mechanism*: the component responsible for comparing query and document representations and assigning relevance scores.
- *Ranked List of Documents*: the ranked output returned to the user, ordered according to the relevance scores.

Once a query is submitted, the system retrieves candidate documents and ranks them according to a scoring function. This scoring mechanism depends on the retrieval model adopted, which determines how documents and queries are represented and compared. Common retrieval models include the following:

- *Boolean Model*: documents are represented as sets of terms, and queries are formulated as Boolean expressions over these terms. A document is retrieved only if it satisfies the Boolean constraints. Since relevance is treated as a binary condition, Boolean models do not naturally support ranked retrieval.
- *Vector Space Model (VSM)*: documents and queries are represented as weighted vectors in a high-dimensional space. Relevance is estimated using similarity measures, such as cosine similarity, which enables ranking results according to their similarity to the query.
- *Probabilistic Models*: probabilistic retrieval models estimate relevance by assigning a probability that a document is relevant to a given query. A widely used probabilistic ranking function is Okapi BM25 [394], which combines term frequency weighting with inverse document frequency and document length normalisation. Given a query Q composed of terms $q_1, q_2, \ldots, q_n$, the BM25 score for a document D is defined as:

$$\text{Score}(Q, D) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{tf_{q_i,D}(1 + k_1)}{tf_{q_i,D} + k_1 \left(1 - b + b \cdot \frac{|D|}{dl_{\text{avg}}}\right)} \tag{2.9}$$

  where $|D|$ denotes the number of words in D, dl_{avg} is the average length of the document in the collection, and k_1 and b are hyperparameters. The inverse document frequency component is computed as:

$$\mathrm{IDF}(q_i) = \ln\left(1 + \frac{N - N(q_i) + 0.5}{N(q_i) + 0.5}\right), \tag{2.10}$$

where N is the total number of documents in the collection and $N(q_i)$ denotes the number of documents containing term q_i.

2.4 Neural Information Retrieval

Neural Information Retrieval (Neural IR) refers to retrieval approaches that leverage neural networks to model queries and documents and to estimate their relevance through learned representations [316]. Unlike traditional IR models, which typically rely on hand-crafted lexical matching functions and term-based weighting schemes, neural IR models aim to learn semantic representations directly from data, enabling them to capture linguistic variability and semantic similarity beyond exact term overlap. This shift has been enabled by the success of deep learning in natural language processing, and in particular by pre-trained transformer-based language models, which provide contextualised representations that can be adapted to retrieval tasks. A major class of neural IR methods is based on representing both queries and documents as dense vectors in a continuous embedding space. In this setting, retrieval can be formulated as a similarity search problem: relevant documents are expected to lie close to the query representation according to a similarity function, such as cosine similarity or dot product. This approach enables semantic matching and has become a fundamental component of modern retrieval systems. However, learning effective representations typically requires large amounts of training data and carefully designed supervision signals. Neural retrieval models may be trained using supervised relevance judgments, weak supervision derived from click logs, or self-supervised strategies.

Neural IR architectures are commonly categorised into two main families: *bi-encoder* and *cross-encoder* models.

- *Bi-encoders* [388]: bi-encoder architectures encode queries and documents independently using two neural encoders, often sharing the same parameters. Each encoder maps its input into a dense representation space. The similarity between the query and a document is computed using cosine similarity or dot product. Since document representations can be computed offline and stored in an index, bi-encoders are particularly suitable for large-scale retrieval settings.

- *Cross-encoders* [336]: cross-encoder architectures jointly encode the query and the document within the same model. The input is typically formed by concatenating the query and the document, and the model outputs a

relevance score for the pair. Cross-encoders can model fine-grained token-level interactions and often achieve higher ranking effectiveness than bi-encoders, but they are more computationally expensive.

Both approaches present complementary trade-offs between efficiency and effectiveness. Bi-encoders enable scalable retrieval due to the possibility of precomputing document embeddings and performing approximate nearest neighbour search. Cross-encoders, while more costly, generally provide stronger ranking performance and are therefore typically applied only to a limited set of candidate documents. Beyond this architectural distinction, neural retrieval methods can be further analysed from two complementary perspectives: the type of generated representation and the stage within a retrieval pipeline.

From the perspective of representation learning, neural retrieval models are grouped into dense and learned sparse approaches:

- *Dense retrieval* [388]: dense retrieval models map queries and documents into low-dimensional continuous vectors, where relevance is estimated through vector similarity. These models capture semantic similarity and paraphrasing effects, allowing retrieval even when query and document share limited lexical overlap. A typical approach is the bi-encoder architecture, in which separate encoders map queries and documents into embeddings. The query encoder encodes the query q into a single query embedding $\psi_q = Encoder_Q(q)$ and the document encoder encodes the document d into a single vector $\psi_D = Encoder_D(d)$. Finally, a simple dot product is employed to calculate the relevance score between the query representation and each document representation. Based on this approach, several dense retrieval variants have been proposed, including ANCE [496], TCT [264], TAS-B [178], BGE [70] and GTR [330].

- *Learned sparse retrieval* [443]: learned sparse retrieval models such as Doc2Query [335], DeepCT [98] and SPLADE [237], aim to combine the interpretability and indexing efficiency of traditional sparse representations, such as BM25, with the learning capability of neural architectures. Instead of relying on fixed term weights such as TF-IDF or BM25, these methods learn sparse representations in which only a limited number of dimensions are activated, often corresponding to vocabulary terms. Learned sparse models can therefore be indexed using inverted lists, similarly to classical retrieval systems, while capturing semantic information through learned weights and contextual modelling.

Despite their success, dense, learned sparse and cross-encoders retrieval methods often exhibit limited robustness when applied to out-of-domain scenarios such as scientific or biomedical search [442], which differ from the original training data

(typically MS MARCO [328]). Approaches including data augmentation [466], continual learning [64] and task-aware training [82, 453] have been investigated to address this challenge. While effective, these approaches typically rely on domain-specific labelled data for relevance, which are not always available [442], and incur additional computational costs, motivating interest in alternative strategies that promote domain generalisation without specific fine-tuning.

From the perspective of retrieval pipeline, models can be distinguished according to whether they operate as first-stage retrievers or as re-ranking models:

- *First-stage retrievers* [388, 443]: first-stage retrievers effectively rank all the documents in a corpus. Since this step must operate at large scale, first-stage retrieval models are designed to be computationally efficient and are typically implemented using lexical models (e.g., BM25), dense bi-encoders, or learned sparse retrievers. First stage-retrievers can be used alone or their output is commonly a set of top-k candidates, which is then passed to subsequent ranking components.

- *Neural re-rankers* [336, 334]: neural re-ranking models operate as a second-stage component of a retrieval pipeline. In this setting, an initial retriever produces a candidate set of documents, and a neural model re-scores these candidates. Re-ranking models are often implemented using cross-encoders, which provide strong effectiveness due to their ability to jointly model query and document representations, but are computationally expensive for full-corpus retrieval. In some scenarios, bi-encoder models may also be employed as efficient re-rankers.

These categories are not mutually exclusive and are often combined in multi-stage retrieval architectures. In practice, retrieval systems frequently employ hybrid pipelines in which a first-stage retriever generates candidates efficiently, while subsequent neural models refine the ranking by modelling deeper semantic interactions.

2.4.1 Training Neural Retrieval Models

Training neural retrieval models in a supervised setting requires a set of queries Q, a document collection D and relevance judgments for query-document pairs. In many practical scenarios, only positive relevance labels are available, i.e. documents that are known to be relevant for a query. Negative examples are therefore typically constructed by sampling documents that are not labelled as relevant. These negatives may be obtained through random sampling [483], lexical retrieval methods such as BM25 [327], or more challenging strategies such as hard-negative mining [309], which selects negatives that are highly similar to the query according to a baseline retriever. A common training formulation [191] relies on triplets consisting of a query q, a relevant document d^+, and a set of negative documents

D^- randomly sampled from the collection of documents not judged for q. Neural retrieval models can be trained by minimising a cross-entropy objective over the positive and negative candidates:

$$L(q, d^+, D^-) = -\log \frac{\exp\left(\gamma \cdot \cos(q, d^+)\right)}{\sum_{d \in \{d^+\} \cup D^-} \exp\left(\gamma \cdot \cos(q, d)\right)}, \tag{2.11}$$

where q and d denote the vector representations of the query and the document, $\cos(q, d)$ denotes cosine similarity, and γ is a scaling parameter. Alternatively, training may be performed using a triplet margin loss [17]:

$$L(q, d^+, D^-) = \sum_{d^- \in D^-} \max\left(\|q - d^+\|_p - \|q - d^-\|_p + \gamma,\ 0\right), \tag{2.12}$$

where $\gamma \in \mathbb{R}$ denotes the margin and $\|\cdot\|_p$ is the $p-$norm of the difference between the query and document vector representations. This loss encourages relevant documents to be closer to the query representation than non-relevant ones by at least a margin.

In many real-world applications, neural retrieval models are integrated with traditional retrieval methods. Previous work [315] has shown that combining lexical matching signals, such as BM25, with semantic representations learned by neural models often improves robustness and retrieval effectiveness, particularly in scenarios where both exact term matching and semantic generalisation are required.

2.5 Domain-specific Information Retrieval

Information Retrieval systems are often conceptualised under the assumption of general-purpose usage, for example web search engines, where the document collection and user behaviour are diverse and unconstrained. However, many practical retrieval scenarios arise within specialised contexts in which the characteristics of the document collection, the users, and even the definition of relevance may differ significantly from those of general web search. Such scenarios are commonly referred to as *domain-specific information retrieval*, where system design must account for different data distributions and user needs [289]. Domain-specific retrieval settings are typically characterised by specialised terminology, abbreviations and technical content. In domain-specific scenarios, such as biomedical or legal domains, lexical overlap between queries and documents is often insufficient for effective matching, thereby requiring retrieval models that can capture semantic relationships, domain knowledge and contextual relevance [289, 313]. For example, in medical information retrieval, user queries vary in complexity and intent depending on the user's role (e.g., clinician, researcher, patient), and traditional retrieval techniques alone may not meet these diverse needs [156, 140]. For these reasons, applying a model

trained on a general-purpose corpus to a domain-specific scenario in a zero-shot setting often results in performance degradation [441] and thus domain-specific search requires the definition of specialised models or ranking strategies extend beyond standard term frequency–based models. These may include the integration of domain lexicons and query expansion techniques tailored to domain concepts [156]. Furthermore, the need for large-scale labelled relevance data in specialised domains often hinders the direct application of supervised training methods, motivating the use of weak supervision, transfer learning, and synthetic annotation strategies to support effective model learning [140]. A notable example is generative pseudo labelling [467], which leverages query generation models to synthesize training signal from unlabelled target-domain documents, enabling unsupervised fine-tuning of dense retrievers without requiring human annotations. Similarly, InPars [38, 204] employs large language models to generate synthetic training queries, allowing re-rankers to be fine-tuned on the new data with strong out-of-domain generalization. An alternative adaptation approach [159] relies on a brief textual description of the target domain to guide the adaptation of dense retrievers. Other strategies involve working at the representation level, for example Zhan et al. [512] propose a disentangled modelling framework that separates domain-specific features from general relevance signals, facilitating more flexible adaptation across heterogeneous retrieval tasks or Gururangan et al. [151] show that continued pre-training of language models on domain-relevant corpora yields consistent improvements across a diverse range of specialized tasks.

2.6 Evaluation of Information Retrieval Systems

The retrieval models discussed in this section differ substantially in terms of architecture, supervision requirements, and computational cost. Nevertheless, their effectiveness is ultimately assessed through a shared experimental methodology based on standard benchmark collections and evaluation metrics. This section introduces the evaluation paradigms and measures adopted throughout this dissertation.

Retrieval evaluation measures are designed to reward systems that retrieve relevant documents, while penalising systems that return non-relevant results [350]. A common distinction is made between set-based measures and rank-based measures. Set-based metrics, such as precision and recall, treat retrieval as a set selection problem and ignore the ordering of retrieved items. In contrast, rank-based metrics explicitly account for the ranked nature of retrieval outputs, assigning higher importance to documents appearing at the top of the list. Since user attention is typically concentrated on the highest-ranked results, rank-based evaluation measures are particularly relevant in practice. For these reasons, in this dissertation, we primarily focus on rank-based metrics. Evaluation strategies can also be classified

as user-oriented or system-oriented. User-oriented evaluation includes methodologies such as user studies and A/B testing, which provide insights into user satisfaction and real-world performance. However, these approaches are costly and difficult to reproduce, limiting their applicability in research contexts. For this reason, most experimental studies in IR rely on system-oriented evaluation frameworks based on benchmark collections. The dominant methodology in IR research is the Cranfield paradigm, which evaluates retrieval systems using standardised test collections. Such collections typically include three main components: a document corpus, a set of queries, also called topics, and relevance judgments indicating which documents are relevant to each query. Within this paradigm, relevance is usually approximated through topical similarity. This implies that relevant documents are assumed to be equally desirable and independent from one another. Moreover, relevance judgments are often incomplete due to the high cost of manual annotation, especially for large-scale corpora. This incompleteness may introduce noise in evaluation results. Finally, while the paradigm supports graded relevance, many benchmark datasets adopt binary relevance judgments. Since users generally focus on the top-ranked results, evaluation measures are often computed over the first k retrieved documents. In the following, we briefly describe the main effectiveness measures used throughout this dissertation. Let $\mathrm{rel}_q(d)$ denote the relevance label assigned to a document d with respect to query q, and let $R_q^{(k)}$ denote the set of the top-k retrieved documents for query q.

Precision@k and Recall@k Precision@k measures the proportion of relevant documents among the top-k retrieved results. Recall@k measures the proportion of all relevant documents that appear within the top-k retrieved set:

$$\text{Precision@}k = \frac{\sum_{d \in R_q^{(k)}} \mathrm{rel}_q(d)}{k} \tag{2.13}$$

$$\text{Recall@}k = \frac{\sum_{d \in R_q^{(k)}} \mathrm{rel}_q(d)}{\sum_{d \in D} \mathrm{rel}_q(d)} \tag{2.14}$$

Mean Average Precision (MAP) Mean Average Precision assumes binary relevance judgments. The average precision for a query is computed as:

$$\text{AP@}k = \frac{\sum_{i=1}^{k} \text{Precision@}i \cdot \mathrm{rel}_q(d_i)}{\sum_{d \in D} \mathrm{rel}_q(d)} \tag{2.15}$$

where d_i denotes the document retrieved at rank position i. MAP is obtained by averaging AP across all queries.

Normalized Discounted Cumulative Gain (nDCG) Normalized Discounted Cumulative Gain (nDCG) supports graded relevance judgments by applying a logarithmic discount factor to penalise relevant documents retrieved at lower ranks. Discounted Cumulative Gain at rank k is defined as:

$$\text{DCG@}k = \sum_{i=1}^{k} \frac{\text{rel}_q(d_i)}{\log_2(i+1)} \tag{2.16}$$

To ensure comparability across queries, DCG is normalised by the Ideal DCG (IDCG), computed from the ranking obtained by ordering documents by decreasing relevance. The final metric is defined as:

$$\text{nDCG@}k = \frac{\text{DCG@}k}{\text{IDCG@}k}. \tag{2.17}$$

2.6.1 Evaluation of Domain-Specific Systems

To systematically evaluate such retrieval models under domain-specific conditions, the IR research community has established a range of specialised test collections and shared evaluation tasks that go beyond general web search. These benchmarks enable the systematic comparison of ranking models by providing specialised corpora, curated queries, and human relevance judgments that reflect the constraints and task characteristics of real-world professional search settings. Among the shared tasks developed to this end, TREC-COVID serves as a well-known representative example. It extends the Text REtrieval Conference (TREC) framework to the rapidly evolving corpus of scientific literature on COVID-19, leveraging successive releases of the CORD-19 dataset alongside relevance judgments produced by biomedical experts [456]. More broadly, shared-tasks such as CLEF (Conference and Labs of the Evaluation Forum) and NTCIR (Evaluation of Information Access Technologies) have organized shared tasks over specialized collections spanning medical, legal, and patent documents, often incorporating multilingual and multimodal components. These initiatives highlight the challenges of designing retrieval systems capable of handling domain-specific vocabulary, structured metadata, and task-dependent notions of relevance. To support broader evaluation across multiple heterogeneous domains simultaneously, BEIR [441] aggregates eighteen retrieval tasks spanning biomedical literature, scientific fact verification, financial question answering and citation prediction, among others. BEIR provides a unified evaluation framework specifically designed to assess the zero-shot generalization capabilities of retrieval models in out-of-distribution settings. The benchmark encompasses collections targeting specialized domains, including TREC-COVID for biomedical retrieval, SciFact [459] and SciDocs [92] for scientific document retrieval, and FiQA [301] for financial question answering. The BEIR benchmark reveals a consistent performance gap between lexical models and neural dense retrievers in

specialized out-of-domain settings, underscoring the need for domain adaptation approaches.

Chapter 3

Parameter Efficient Approaches for Training Large Language Models: a Systematic Review

One of the key outcomes of the research undertaken during this PhD is the introduction of novel approaches to parameter-efficient fine-tuning in NLP and information retrieval. Therefore, this Chapter presents a systematic literature review that we conducted to enrich the understanding of this research field. Through our systematic review of 174 studies, we have categorized PEFT approaches based on how they update the weights of LLMs. Moreover, we highlight how PEFT approaches are evaluated in the literature.

3.1 Introduction

Our survey systematically examines 174 studies that have proposed and experimentally defined and evaluated PEFT approaches in NLP and IR. We synthesize these studies based on how they update the weights of a pre-trained LLM while fine-tuning it on a downstream task. The specific research questions tailored to our study are presented in Section 3.2. This review aims to aid the development of future PEFT approaches by identifying current necessities and paving the way for future research.

The Chapter is structured as follows. Section 3.2 presents the research questions and the systematic review methodology. Section 3.3 details the results of our systematic review presenting the taxonomy of PEFT methods, describing each category and the models therein and analysing the benchmark datasets and evaluation methods used in the PEFT literature. In detail, Section 3.3.1 introduces the formal definition of PEFT and outlines its foundational principles. Section 3.5 discusses current limitations and open research directions and, finally, Section 3.6 concludes

the study.

3.2 Methodology

This systematic review aims to investigate studies that propose novel Parameter-Efficient Fine-Tuning (PEFT) techniques for Natural Language Processing (NLP) tasks. We categorize the identified studies based on their strategy for selecting which subset of parameters to train during model adaptation. Furthermore, we compile a comprehensive list of benchmark collections that have been used in the reviewed studies, analysing them in terms of the tasks they are designed for. We also examine which datasets and LLMs are most frequently used across PEFT methods, aiming to identify trends, gaps, and opportunities for improving empirical comparisons and reproducibility. Our primary objective is to consolidate the fragmented landscape of PEFT research by identifying recurring methodological patterns, evaluating the consistency of experimental practices, and highlighting emerging trends. To this end, we adopt a systematic review methodology for structured literature analysis. This approach is intended to ensure transparency, reproducibility, and rigor in the selection, annotation, and synthesis of studies.

Following the structured methodology outlined by [93, 351], our systematic review consists of the following key steps: (1) defining the research questions that guide our investigation; (2) establishing clear inclusion and exclusion criteria for the selection of relevant studies; (3) formulating a retrieval strategy, specifying the sources, databases, and keywords employed; and (4) synthesizing the extracted findings to address the research questions.

3.2.1 Research Questions

The overarching aim of this review is to offer a structured understanding of PEFT methods and their evaluation in NLP. To guide this analysis, we formulate the following research questions:

- **RQ1** How is the concept of Parameter-Efficient Fine-Tuning (PEFT) defined in the identified studies?
  - What criteria are used to classify and distinguish different PEFT approaches?
  - What are the key design choices and components that characterize PEFT methods?
- **RQ2** What benchmark datasets and evaluation strategies are employed to assess PEFT methods?

- Which datasets are most frequently used to evaluate PEFT techniques across different NLP tasks?
- What performance metrics are applied to measure the effectiveness of PEFT approaches?
- What evaluation strategies, Language Models, and datasets are applied to measure the effectiveness of PEFT approaches?

3.2.2 Inclusion and exclusion criteria

This phase of the review aimed to formalize the criteria for systematically selecting or excluding articles to be included in our analysis. The development of the criteria was aligned with the specific focus of the review: studies proposing novel PEFT techniques for NLP tasks. Although initial criteria were established in a preliminary phase, they were validated and finalized after processing 10% of the initial study pool, with no further revisions deemed necessary. The final set of criteria is summarized in Table 3.1.

Inclusion/exclusion criteria
Including studies focused on Natural Language Processing and Transformers architecture.
Including scholarly publications subject to peer review.
Including both full-length research articles and short papers.
Sources are confined to top-tier internationally renowned journals, conference proceedings, and workshops.
Specific time frame: from January 1st 2019 to December 31st 2024.
Studies must be written in English.
Studies must define, discuss and evaluate new PEFT methods. We exclude studies in which authors apply existing and well-known PEFT methods to new datasets or domains.

Table 3.1: List of inclusion and exclusion criteria.

In particular, we excluded studies focused solely on non-textual domains (e.g., computer vision or audio processing) in order to maintain a clear emphasis on textual applications of PEFT. To ensure methodological quality and scientific rigor, we included only peer-reviewed studies (both full-length articles and short papers) published in journals, conferences, or workshops. We limited the temporal scope to works published between January 1, 2019, and December 31, 2024, as the earliest known application of PEFT in NLP dates to 2019. Eligible studies were also required to be written in English and to propose new PEFT techniques or significant extensions thereof, as opposed to merely applying existing methods to new tasks or datasets.

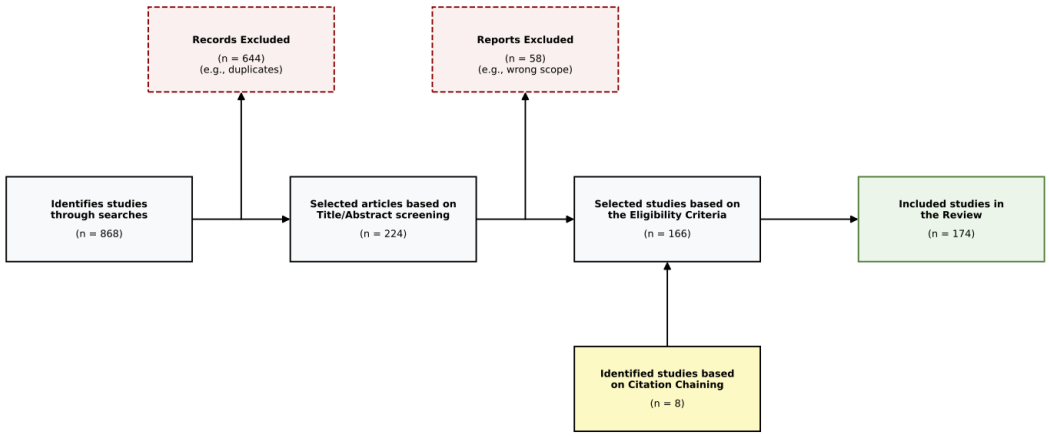

Figure 3.1: Overview of the search approaches followed for the systematic review.

3.2.3 Search strategy and paper selection

The process of identifying relevant literature followed a structured pipeline, as illustrated in Figure 3.1. Guided by the inclusion and exclusion criteria, we conducted systematic searches across major academic repositories and conference proceedings, including the ACL Anthology, ICML, ICLR, NeurIPS, Springer Link, ACM Digital Library, and Elsevier. Search queries were formulated using combinations of keywords such as "parameter efficient" OR "efficient tuning" OR "low rank adaptation", applied typically to "title" and "abstract" fields or, when unavailable, to the full text. We tried different combinations of the aforementioned keywords, aiming to cover most, if not all, of the relevant research for further analysis. The full list of queries is : "parameter efficient", "efficient tuning", "low rank adaptation", "parameter efficient tuning", "efficient low rank adaptation" and "parameter efficient low rank adaptation". Following the aforementioned search process, a total of 868 studies have been identified. Those articles have been manually screened by reviewing their title and abstracts to determine their relevance to this study. At this point, we were interested in reducing the initial document pool to include those focused on NLP and excluding those studies that solely focus on image or audio data. As a result, a total of 224 studies have been selected for further examination. These studies have been evaluated based on the whole set of inclusion/exclusion criteria listed in Table 3.1 and a total of 174 studies have been identified as eligible for this study. The majority of the papers have been excluded because they were merely application of a well-known PEFT method.

3.2.4 Coding scheme and paper synthesis

To support the structured analysis of our review corpus, we develop a detailed coding scheme aligned with the previously outlined research questions. The coding schema comprises three layers:

- General publication metadata, including year, author affiliations, and publication venue, to provide insights into the temporal distribution of research activity.
- PEFT methodology categories, capturing whether the proposed methods fall under additive, selective, reparameterized, or hybrid strategies (addressing RQ1).
- Benchmark-related variables, including task type, dataset name, and LLM used for evaluation (addressing RQ2).

This structured annotation enabled both qualitative synthesis and quantitative aggregation. In addressing RQ1, we analyzed how PEFT methods define and operationalize the selection of trainable parameters, revealing patterns across architectural design choices. In addressing RQ2, we examined the frequency and context in which specific datasets and LLMs are employed, highlighting similarities and divergences in empirical evaluation protocols.

Our synthesis reveals notable commonalities in PEFT implementation across tasks, as well as substantial variation in evaluation setups. By consolidating these findings, we provide a clearer understanding of the methodological landscape and identify opportunities for improving standardization and reproducibility. Ultimately, our aim is to reduce fragmentation in PEFT research by identifying recurring methodological trends, surfacing gaps in current evaluation practices, and proposing a roadmap for future exploration.

3.3 Results

In this section, we analyze and synthesize the publications in our review, following the established coding scheme to answer the posed research questions.

3.3.1 How is the concept of Parameter-Efficient Fine-Tuning (PEFT) defined in the identified studies?

Parameter-Efficient Fine-Tuning (PEFT) represents a paradigm shift in the way Large Language Models (LLMs) are adapted to downstream tasks. As LLMs continue to grow in size, the computational and memory costs associated with

traditional full fine-tuning methods have become a significant barrier to their practical deployment. PEFT techniques address this issue by reducing the number of trainable parameters required for adaptation, while aiming to retain—if not improve upon—the performance typically achieved through full-model fine-tuning. The central intuition behind PEFT is that optimal performance on a target task can often be achieved by updating only a strategically chosen subset of the model's parameters, or by augmenting the model with additional, lightweight components. In doing so, these techniques enable a more scalable and resource-efficient approach to transfer learning, particularly for practitioners working under memory or compute constraints.

The fine-tuning paradigm has long been a fundamental approach for adapting pre-trained models to specific downstream tasks. Conventionally, this process involves updating the entire set of model parameters. However, with the increase of models' number of parameters, often encompassing hundreds of millions or even billions of parameters, the computational costs associated with full fine-tuning have become increasingly high.

To mitigate these challenges, recent research has introduced a family of techniques collectively referred to as Parameter-Efficient Fine-Tuning (PEFT). The aim of PEFT approaches is to adapt large-scale models by modifying only a small subset of their parameters, thus preserving the majority of the pre-trained model while still achieving competitive performance.

Formally, let us consider a pre-trained model characterized by a parameter set $\Theta = \{w_1, w_2, ..., w_N\}$ and a target dataset D. Traditional fine-tuning results in a task-specific model with parameters Θ_D, defined as:

$$\Theta_D = \{(w_1)_D, (w_2)_D, ..., (w_N)_D, ..., (w_{N+M})_D\} = \Theta' \cup \Phi \tag{3.1}$$

where $\Theta' = \{(w_1)_D, (w_2)_D, ..., (w_N)_D\}$ denotes the original model parameters updated on D and $\Phi = \{(w_{N+1})_D, ..., (w_{N+M})_D\}$ denotes any additional parameters introduced during fine-tuning. In cases where no new parameters are added, $\Phi = \emptyset$.

The modification introduced by fine-tuning can be expressed as $\Delta\Theta = \{(w_i)_D - w_i\}_{i=1}^{N}$, which represents the updates on the original model parameters. The subset of modified weights, denoted by $S(\Delta\Theta)$ includes all parameters for which $(w_i)_D - w_i \neq 0$. In standard full fine-tuning, we have $|S(\Delta\Theta)| = |\Theta| = N$ and $\Phi = \emptyset$. In contrast, PEFT approaches aim to minimize the number of trainable parameters such that:

$$S(\Delta\Theta) \cup |\Phi| < |\Theta| \tag{3.2}$$

Notably, the precise threshold for parameter efficiency remains loosely defined across the literature [256]. Several criteria have emerged:

- [105] suggest that PEFT implies an order-of-magnitude reduction in trainable parameters, such that $|S(\Delta\Theta) \cup \Phi| << |\Theta|$,

- [256] describe PEFT methods as those that require only a "small set" of parameters to be updated during adaptation,
- In practice, most PEFT implementations restrict the trainable amount of parameters to less than 2% of the model's original ones [183, 354, 186, 475].

PEFT methods originate in Computer Vision, where convolutional residual modules are introduced to enable fine-tuning [385, 384, 399] on visual models such as ResNet [162] and VGG [420]. These methods incorporate small convolutional layers to reduce the number of trainable parameters during fine-tuning while preserving performance. The first approach in NLP is introduced by Houlsby et al. [183], who propose low-rank residual adapters to fine-tune BERT [102] across multiple text classification tasks. Their findings show that it was possible to fine-tune only a fraction of the total parameters while maintaining strong performance. Building on this insight, further studies show that overparameterized models, such as BERT [367], often work within a lower intrinsic dimensional space [245]. This led to the hypothesis that weight updates during fine-tuning also lie in a low-rank subspace. This concept is further developed and empirically validated by [186].

Since these foundational works, PEFT methods have expanded significantly Recently, PEFT methods have been extended beyond text-based models to fine-tune Large Language Models (LLMs) [519], Vision Transformers (ViTs) [169], and diffusion models [488]. Additionally, research has explored PEFT techniques for Vision-Language Models (VLMs) [207, 84], further showing the versatility and applicability of these approaches across various modalities.

In this survey, we focus on PEFT approaches that have been applied to at least one NLP-related task. While many of the underlying principles are transferable to other domains, our emphasis remains on methods tailored for language models. In what follows, we offer a comprehensive taxonomy of PEFT techniques along with a detailed synthesis of their motivations and empirical evaluations.

3.3.2 What criteria are used to classify and distinguish different PEFT approaches?

Building on formal definitions introduced in Section 3.3.1, PEFT techniques can be broadly categorized into four methodological classes, based on how they constrain or modify the trainable parameter space. This classification draws upon recent efforts to systematize the field [256] and serves as a foundation for the taxonomy adopted in this survey:

1. **Additive methods**: these approaches introduce additional trainable parameters or layers into the model while keeping the pre-trained parameters frozen. Only the newly added parameters are optimized during fine-tuning, i.e. $M \geq 1,\ \Phi \neq \emptyset,\ S(\Delta\Theta) = \emptyset$.

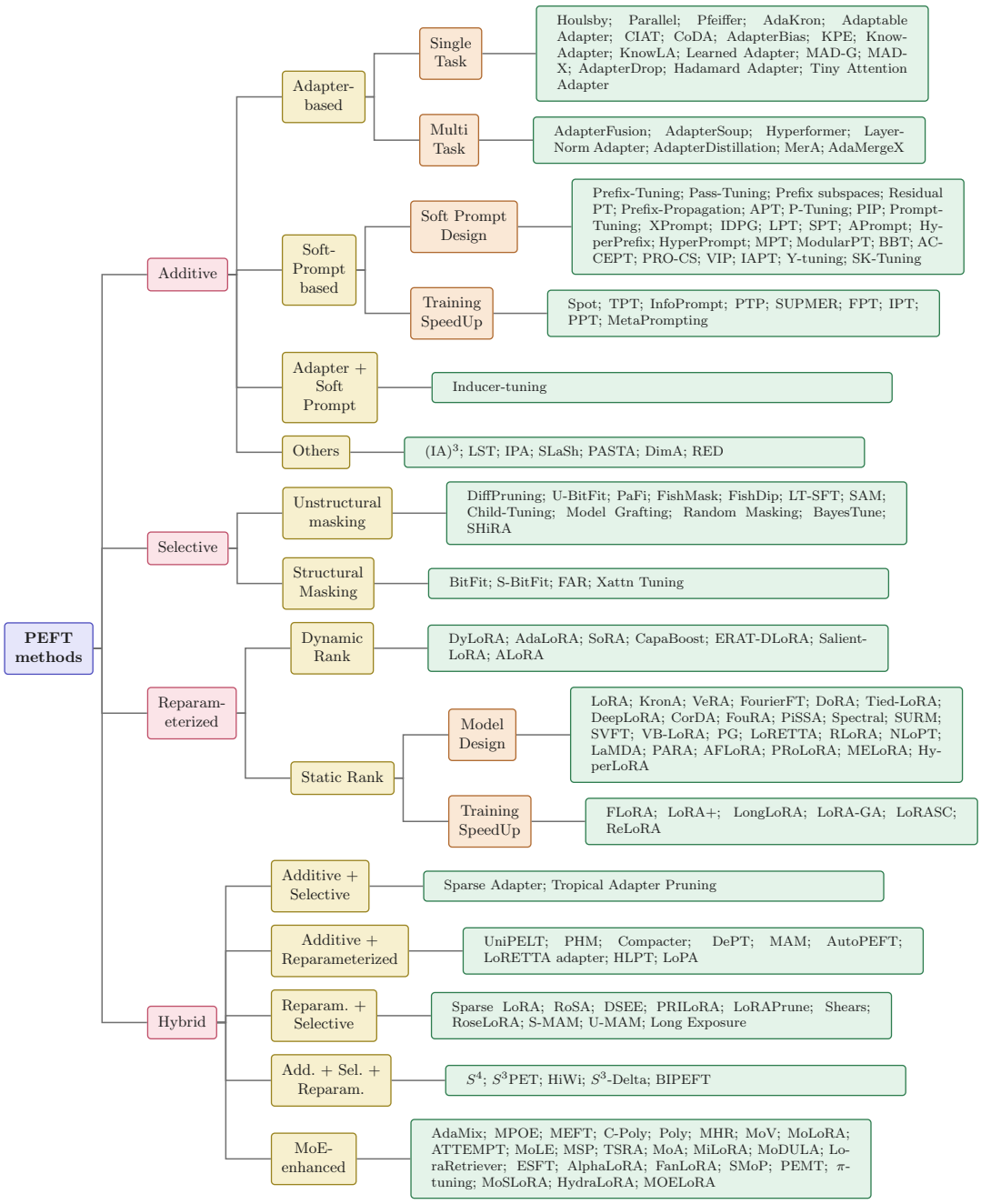

Figure 3.2: Taxonomy of PEFT

2. **Selective methods**: these methods require the training of only a small subset of the original model parameters, i.e. $M = 0,\ \Phi = \emptyset,\ N > |S(\Delta\Theta)| \geq 1$.

3. **Reparametrization-based methods**: these approaches decompose original model parameters into low rank matrices, which are then trained during the fine-tuning process. Let $\mathbb{W} \subseteq \Theta$ denote the set of parameters to be reparametrized. If each $w_i \in \mathbb{W}$ is reparametrized with a new set of parameters $R(w_i) = \{u_1, ..., u_{N_i}\}$, the updated parameter set is expressed as $\Phi = \bigcup_{i=1, w_i \in \mathbb{W}} R(w_i)$. It is worth noting that $\Delta\theta \neq \emptyset$ because a subset of original model parameters is modified by ϕ. However, $\Delta\theta \neq \phi$ but $|\Delta\theta| = |\phi| << N$.

4. **Hybrid methods**: these methods integrate components from different PEFT approaches, allowing flexible combinations of additive, selective, and reparametrized techniques. For example, UniPELT [305] combines additive, selective and reparametrization-based strategies. Alternatively, Compacter [217] applies reparametrization to newly introduced parameters. Thus, their formal definition can vary depending on the way existing approaches are combined. For example, in UNIPELT [305] $\Delta\theta \neq \emptyset, \phi \neq \emptyset$ and

$$\phi = \{(w_{N+1})_D, ..., (w_{N+M})_D\} \cup \bigcup_{i=1, w_i \in \mathbb{W}} R(w_i). \tag{3.3}$$

To facilitate the reader's understanding of the design space, Figure 3.2 presents a visual taxonomy of the PEFT methods included in this review.

These groups reflect the different ways in which PEFT approaches reduce trainable parameter counts, either by appending new modules, selecting a subset of existing parameters, reparametrizing internal structures, or combining multiple strategies.

3.3.3 What are the key design choices and components that characterize PEFT methods?

In this section, we examine each PEFT category in detail, outlining its conceptual foundations, representative implementations, and formal characteristics.

Additive Methods

Additive methods keep the pre-trained model intact while introducing a small set of trainable parameters within the model's architecture. During the fine-tuning process for a specific downstream task, only these additional parameters are updated. By keeping the original network parameters fixed, a high degree of parameter sharing is achieved across different tasks, preserving previously learned knowledge. Within this category, two major subclasses have emerged: adapter-based and soft prompts methods.

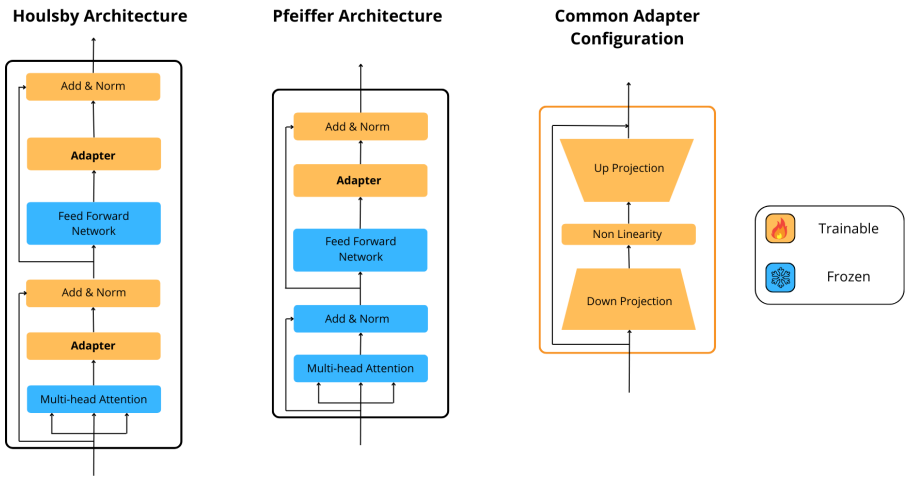

Figure 3.3: Examples of adapter architectures: on the left, Houlsby and Pfeiffer adapters; on the right, the most popular configuration of a bottleneck adapter layer.

Adapter-based Methods Adapters are introduced by [183]. As shown in Figure 3.3, an adapter is composed of two linear layers, the so-called down and up projections, along with an internal residual connection, and is inserted into each transformer layer after both the attention and feed-forward layers. The adapters are applied directly to the output of these layers before the addition of the skip connection, and the adapter's output is then passed to a normalization layer, which is updated during fine-tuning. A key aspect of the design is the near-identity initialization of the linear layers, which ensures that, at the beginning of training, adapter modules approximate an identity function. Together with the internal residual connection, this initialization strategy allows the model to adapt to new tasks while preserving previously learned knowledge.

The architecture of adapters follows a bottleneck structure: the d-dimensional features of the pre-trained model are first projected into a lower-dimensional space of size r by the down-projection weight matrix. A nonlinearity is then applied, followed by an up-projection back to the original dimension d. The total number of parameters added per layer, including biases, is $2rd + d + r$. By setting r much smaller than d, the added parameters per task typically remain below 1%. Formally, given a d-dimensional input x, the output of an adapter with residual connection is:

$$Adapter(x) = LN\Big(x + W_U * (\sigma(W_D * x))\Big) \tag{3.4}$$

where LN is the normalization layer, $W_U \in \mathbb{R}^{d\times r}$ is the up-projection weight matrix, $W_D \in \mathbb{R}^{r\times d}$ is the down-projection weight matrix, and σ is the activation function (usually GeLU [173]). The bottleneck dimension r controls the trade-off between adaptation capacity and parameter efficiency: larger values of r generally improve task performance, whereas smaller values reduce the number of additional

parameters [67].

While adapters are highly effective in introducing task-specific capacity with minimal parameter overhead, they nonetheless introduce a small increase in model size and inference latency. Each Transformer layer receives additional computations due to the down- and up-projection operations, and although these computations are lightweight compared to full fine-tuning, they are repeated for every layer in the network [401]. Consequently, for very deep transformers, this cumulative overhead can become non-negligible, especially under limited computational resources.

Building upon the architecture proposed by [183], [353] introduced a more parameter-efficient adapter design that follows the same bottleneck but inserts adapters only after feed-forward layers, omitting their placement after attention layers. This configuration reduces the number of trainable parameters by 50% compared to the original design while maintaining comparable downstream performance. Motivated by this enhanced efficiency, subsequent studies have proposed various architectural adaptations to further reduce overhead and improve generalization.

Parallel Adapters [161], CIAT [539] and CoDA [240] reframe adapters as parallel side networks to control feature interference or introduce conditional processing pathways. Other methods focus on improving adapter efficiency, for example AdapterDrop [401] removes lower-layer adapters during training and inference to reduce computational cost, while Adaptable Adapters [320] learn to activate only the most beneficial adapter layers. Further, AdaKron [47] optimizes the down-projection layer by decomposing it into two smaller networks multiplied by the Kronecker product, while Hadamard Adapter [80] introduces a variant based on element-wise linear transformation using Hadamard products. AdapterBias [127] proposes an approach in which only a token-dependent vector and a linear transformation are learned to shift the hidden representations of the Transformer, significantly reducing adaptation complexity. Tiny Attention Adapter [530] incorporates an attention mechanism into the adapter, allowing token-level embedding updates to capture contextual information more effectively.

Adapters can be used to integrate structured external knowledge into the model. For example, KPE [531] and Known-Adapter [332] and KnowLA [288] integrate triple embeddings from an external knowledge graph to improve Question Answering [312, 437] and Named Entity Recognition [408, 366] performance.

Adapter-based methods have also been extended to cross-lingual transfer learning. MAD-X [355] introduces a modular framework that separately trains and combines language, task, and adapters to enable adaptation across languages. Language adapters capture typological features, task adapters encode task-specific knowledge, and a new type of adapters, referred to as an invertible adapter, aligns representations between them. In contrast, MAD-G [12] dynamically generates language

adapters using Contextual Parameter Generation [360], allowing zero-shot transfer to previously unseen languages with reduced training and parameter requirements. Subsequent methods further build on these foundations. Inspired by MAD-X, UDAPTER [303] follows a two-step procedure: a domain adapter is first trained to extract domain-invariant features, and a task adapter is then stacked to enhance specialization. Target Language-Ready (TLR) adapters [345] address compatibility issues between independently trained task and language adapters by jointly exposing task adapters to target languages during training, enhancing cross-lingual alignment during inference.

To facilitate multi-task adaptation, a variety of methods extend adapter-based architectures with mechanisms for sharing or merging task-specific knowledge. In the literature, there are two main strategies: two main strategies have emerged: the first trains a different adapter for each task and then merges them at inference time, such as AdapterFusion [353] and AdapterSoup [86]; the second relies on an external hypernetwork that generates adapter weights conditioned on the task description. AdapterFusion [353] introduces an attention-based fusion module that aggregates information from multiple adapters. Similarly, AdapterSoup [86] performs simple averaging of adapters trained on different domains. MerA [167] leverages optimal transport to merge pre-trained adapters based on their weight and activation similarity. In contrast, Hyperformer [300], LayerConnect [418] Hyperdecoders [198] rely on hypernetworks that dynamically generate adapter weights conditioned on task or layer embeddings. While Hyperformer and LayerConnect condition adapter generation globally, Hyperdecoders generate instance-specific weights per input.

Finally, recent works, such as PHA [529], Inspirational Pointer [285], SCALEARN [126] and ADAPTERS MIXUP [329] expand adapter learning to curriculum-based training, prototype matching, pointer mechanisms, and mixup-style regularization.

Soft-prompt Prompting techniques for language models aim to influence model behaviour through structured input text. Traditional prompting methods [375, 272] (called discrete prompting) rely on input sequences that include a task description and a few in-context examples. However, such methods are inherently limited by input length constraints and are challenging to optimize and they are very sensitive to variations in the prompt. To overcome these limitations, researchers introduce Soft Prompts, which are continuous, trainable embeddings prepended to the model input, to shift prompt design from a discrete formulation to a differentiable optimization task. Soft-prompts are typically optimized via gradient descent and integrated into the input embedding layer of pre-trained language models.

Consider a task with input X (context) and output Y (target), modelled by an autoregressive language model $p_\phi(y|x)$. Let $z = [x; y]$ denote the combined input-output sequence. For token index i, the activation h_i is a concatenation of representations across all layers: $h_i = [h_i^1; ...; h_i^n]$, where h_i^j is the activation at the

j-th layer. The model produces:

$$h_i = LM_\phi(z_i, h_{<i}), \tag{3.5}$$

with the final activation used to generate the next token.

Prefix-tuning [253] modifies this structure by prepending a prefix vector, producing $z = [PREFIX; x; y]$. The prefix activations, stored in a trainable matrix $P_\theta \in \mathbb{R}^{|P_{idx}| \times dim(h_i)}$, where P_{idx} denote the sequence of prefix indices and $|P_{idx}|$ is the length of the prefix, serve as input for prefix tokens:

$$h_i = \begin{cases} P_\gamma[i,:] & if\ i \in P_{idx}, \\ LM_\mu(z_i, h_{<i}) & otherwise. \end{cases} \tag{3.6}$$

The training objective is the same as Equation 2.7, but only γ parameters are updated during training; the language model weights μ remain fixed. It is worth noting that even for $i \notin P_{idx}$, prefix activations influence the output due to their presence in the left context.

Inspired by Prefix-Tuning, Prompt Tuning [242] and P-Tuning [277] apply learnable vectors only at the word embedding layer, whereas p-Tuning v2 [278] removes reparametrization and scales across larger models and tasks. Context-sensitive extensions include IDPG [492], LPT [276] and SPT [537], which generate prompts dynamically per input using external prompt generators. APrompt [470] modifies the attention mechanism by learning prompts for the query, key, and value matrices.

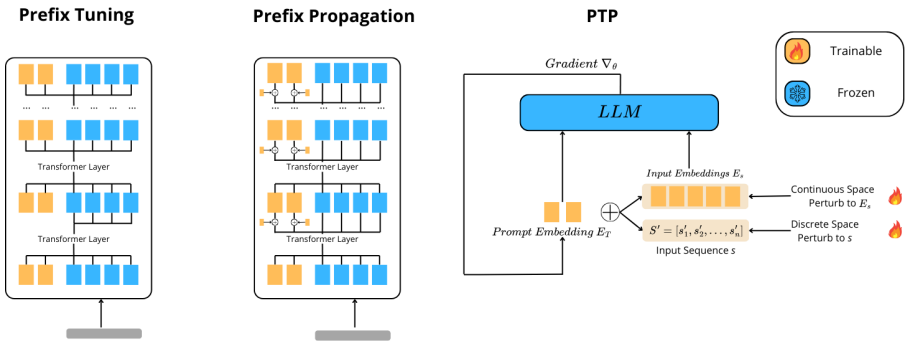

Figure 3.4: Examples of soft-prompts approaches: Prefix Tuning, Prefix Propagation and PTP.

Prefix Propagation [246] conditions prefixes on previous hidden states to enable sequential adaptation. Prefix Subspaces [117] learn a subspace of prefixes instead of a fixed vector, supporting interpolation and improving few-shot generalization. Residual Prompt Tuning [383] adds shallow residual networks over the prompt vector, while Adaptive Prefix-Tuning (APT) [527] employs gating mechanisms to modulate prefix components based on input context. XPrompt [291] identifies the

gap between prompt tuning and full fine-tuning in models under 11B parameters and introduces structured pruning to discard ineffective prompt tokens.

As with adapter-based methods, prompt-based approaches can also be extended to multi-task settings by using external networks or hypernetworks to generate or refine prompts, this category includes Hyper-Prefix [357], HyperPrompt [170], HyperPELT [528], IAPT [538], and BBT/ BBT v2 BBT [432, 431]. Otherwise, multi-task tuning is addressed by MPT [476] and ModularPT [68], while tasks like code understanding or controlled generation motivate task-specific prompt methods such as PIP [460], Pass-Tuning [73], PRO-CS [20], ACCEPT [261] and VIP [421, 214].

Despite their promise, soft prompt methods often suffer from training instability and slow convergence. Empirical studies [429, 193] show that soft prompts require much more training time compared to full fine tuning to converge. To mitigate this, Transferable Prompt Tuning (TPT) [429] initializes prompts from those trained on similar tasks. SPoT [457] follows a similar approach, transferring prompts across tasks. [193] additionally introduce the concept of "partial LLMs", i.e. smaller versions of PLMs created via layer pruning, and show that soft prompts trained on smaller models can be transferred to full-sized models. Their Fast Prompt Tuning (FPT) procedure begins with soft prompt training on a partial model and incrementally restores layers, achieving comparable results with 30% fewer FLOPs.

Recent works have also explored strategies for learning stable and information-rich prompts. InfoPrompt [487] introduces two loss terms, Head and Representation loss, designed to maximize mutual information [153, 362] between prompts and either classifier parameters or PLM representations. To address instability, [71] propose Prompt Tuning with Perturbation-based Regularizer (PTP), which applies perturbations in both discrete (text) and continuous (embedding) spaces (Figure 3.4). Similarly, Intrinsic Prompt Tuning (IPT) [372] learns a low-dimensional subspace for prompt parameters by decomposing the prompt matrix using encoder-decoder projections, inspired by auto-encoding strategies [43]. Finally, Meta-learning strategies such as MetaPrompting [182], SUPMER [341], and PPT [145] combine few-shot learning with meta-optimization [121, 331].

Like adapters, soft prompts are additive PEFT methods; however, unlike discrete prompting, they require gradient-based optimization over continuous embeddings, which can introduce training instability [431]. Because the language model parameters remain frozen, the expressive capacity of the adaptation is constrained to the relatively small prompt embedding space, which may limit effectiveness on tasks requiring deeper modifications to internal representations. Furthermore, prompt tuning often exhibits sensitivity to initialization and prompt length, requiring careful hyperparameter selection to achieve competitive performance [242]. Another practical limitation arises from the position of prompts in the input sequence: since prompt tokens are prepended to the model input, they consume part of the available context window, which can become problematic in tasks involving

long sequences. From a computational perspective, soft prompts introduce minimal additional parameters and negligible runtime overhead, since they operate primarily at the embedding level. However, their limited capacity to influence deeper layers of the network has motivated several hybrid approaches that combine prompt-based conditioning with internal architectural modifications.

Adapter and Soft Prompt Combinations Several approaches have explored the integration of adapter-based and soft prompt-based fine-tuning techniques, aiming to combine their respective advantages. One such method is Inducer-Tuning, proposed by [78], which leverages the residual outputs of adapter modules to enhance initialization in prefix-tuning. Inducer-Tuning introduces adaptive inducers, which adopt the structural characteristics of adapters, enabling a stable and expressive combination of additive fine-tuning mechanisms within a unified framework.

Despite the potential benefits of such hybrid designs, combining prompts and adapters also introduces additional complexity in both optimization and architecture design. When prompts and adapters are both employed, the overall parameter efficiency advantage may diminish slightly, particularly in multi-task scenarios where separate prompts and adapters must be maintained for each task. Consequently, while hybrid approaches offer promising flexibility, careful consideration is required to balance parameter efficiency and training stability. It is worth noting that, at present, we identify only one approach in this class, but we retain the category because it may become more relevant as combinations of adapters and soft prompts continue to develop.

Other Additive Approaches Beyond adapters and soft prompts, several additive PEFT strategies have been proposed, offering distinct mechanisms to efficiently adapt large language models without modifying pre-trained parameters. One notable example is $(IA)^3$ [270], which introduces three learnable vectors $l_k \in \mathbb{R}^{d_k}$, $l_v \in \mathbb{R}^{d_v}$, $l_f \in \mathbb{R}^{d_f}$ to rescale the key, value, and FFN activations, respectively. The $(IA)^3$ method can be summarised with the following equations defined to update the Self Attention (SA) and FFN parameters:

$$SA = Softmax(\frac{Q(l_k \odot K^T)}{\sqrt{d_k}})(l_v \odot V) \tag{3.7}$$

$$FFN(x) = (l_f \odot \sigma(W_1 x))W_2 \tag{3.8}$$

where σ is a non-linearity function, x in the input vector to the FFN layer, $W_{1,2}$ are the weight matrices in a FFN layer and $\odot$ is the Hadamard product.

To reduce memory consumption, LST [433] eliminates the need for backpropagation through the fine-tuned layers by introducing a ladder-side network external to the model backbone. In contrast, Inference-Time Policy Adapter (IPA) [286]

utilizes reinforcement learning to train a lightweight adapter that operates during decoding, guiding generation toward user-defined objectives without modifying model parameters.

Several approaches target token-specific or structure-aware adaptation. PASTA [502] fine-tunes only the hidden representations of special tokens, such as [SEP] and [CLS] in BERT, by adding a trainable vector, resulting in efficient specialization without altering other representations. Dimensionality Augmentation (DimA) [524] proposes expanding the intermediate dimension in attention and feed-forward computations to increase representational capacity without increasing the number of layers.

Representation Editing (RED) [489] offers a minimal approach to modifying FFN outputs by inserting two trainable vectors, $l_{scaling}$ and l_{bias}, at the position typically occupied by adapters. The resulting edited representation is computed as:

$$h_2 = l_{scaling} \odot h_1 + l_{bias}, \tag{3.9}$$

where h_1 is the FFN output and h_2 is the edited hidden state. SLaSh [149] introduces an efficient additive strategy by inserting task-specific parameters at multiple Transformer layers, where each set of parameters is derived from fixed random projections of a single trainable vector.

Selective Methods

Selective PEFT techniques aim to fine-tune only a small, well-defined subset of a model's original parameters. Let $S(\Delta\Theta)$, as defined in Section 3.3.1, represent the set of updated parameters during fine-tuning. A binary mask $M = \{m_1, m_2, ..., m_N\}$ is applied such that $m_i = 1$ if the corresponding parameter w_i is trainable, i.e. $w_i \in S(\Delta\Theta)$, and $m_i = 0$ otherwise, indicating the parameter remains frozen. This setup allows selective methods to control fine-tuning granularity either through predefined structural rules or via data-driven learning of the mask. Following previous works [155], we split the category into two subclasses: structural and unstructural approaches.

Structural Masking Structural selection approaches apply rule-based criteria to determine which parameters to update. A simple and widely used technique is to fine-tune only the top layers of the model (e.g., the last two layers [109]). BitFit [31] updates only the model's bias terms, modifying fewer than 0.1% of total model's parameters while achieving comparable performance to both full fine-tuning and adapters. S-BitFit [238] refines this approach by pruning a subset of biases [317], thereby reducing parameter count. Xattn Tuning [137] restricts training to cross-attention layers for Machine Translation tasks [370]. A particularly notable structural approach is Freeze-And-Reconfigure (FAR) [458], designed for

compact models such as DistilBERT [409]. FAR identifies "learner" and "non-learner" nodes during a priming phase, based on an L_1-norm importance criterion, and then reconfigures the model to separate frozen and trainable nodes for improved memory efficiency. A problem related to this class is that we need to choose a priori a set of trainable parameters, which could depends on the model architecture or the task or the domain we are working on.

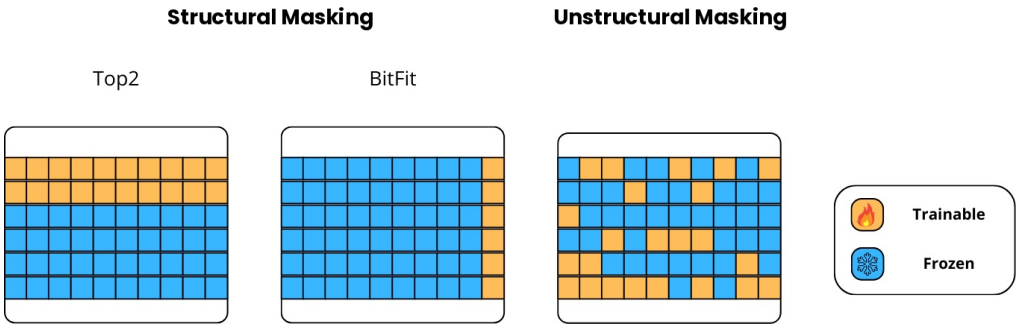

Figure 3.5: Examples of selective methods: on the left Structural (Top2 and BitFit) and on the right an example of Unstructural Masking.

Unstructural Masking In contrast, unstructured selection methods learn binary masks dynamically based on optimization objectives or statistical measures of parameter importance. DiffPruning [148] introduces a differentiable approximation of the L_0-norm to regularize a learnable binary mask applied to model weights. Pruning-and-Finetuning (PaFi) [258] selects trainable parameters task-agnostically, using the same precomputed mask across multiple downstream tasks. Notably, its mask generation is data-less, requiring no task-specific data. Similarly, FISH [434] and its extension FISH-DIP use approximations of the Fisher Information Matrix [123, 10] to rank parameters by importance [233, 3]. Child-Tuning [499] implements selective backpropagation by masking gradients outside a predefined subnetwork. After identifying an effective parameter subset for a specific task, Lottery Ticket Sparse Fine-Tuning (LT-SFT) [11] rewinds the model to its pre-trained initialisation and re-trains only the most influential parameters, inspired by the Lottery Ticket Hypothesis [124, 239, 535, 302, 368]. Different from the Lottery Ticket Hypothesis, Task-aware Grafting [342] does not require re-training. U-BitFit [238] further improves the selectivity of S-BitFit by allowing updates only to selected components within each bias vector.

Several recent works address the complexity of optimal parameter selection. Recognizing it as an NP-hard problem [128], the Second-order Approximation Method (SAM) [128] approximates the original objective with an analytically solvable function. BayesTune [232] applies a sparse Laplace prior over model weights, using posterior inference to determine which parameters are most informative to

update. Similarly, Sparse High-Rank Adapters (SHiRA) [34] explores three masking strategies, including Random Masking, further analysed by [497], to identify subsets that require training only 1–2% of total model parameters.

While selective parameter tuning offers an intuitive and highly parameter-efficient approach to adapting large language models, it also introduces several methodological and practical challenges. By restricting updates to a small subset of the original model parameters, selective methods significantly reduce the number of trainable weights and preserve most of the pre-trained model structure. However, identifying which parameters should be updated is inherently difficult, as the optimal subset of trainable parameters depends strongly on both the model architecture and the downstream task. In practice, many approaches rely either on heuristic structural rules, such as updating only top layers or bias terms, or on statistical importance measures derived from gradients or approximations of parameter sensitivity [31]. While these strategies can achieve strong parameter efficiency, they may also lead to suboptimal adaptation if the selected parameters do not adequately capture task-specific knowledge. Another limitation of selective approaches arises from their dependence on accurate parameter importance estimation. Methods based on gradient statistics, Fisher Information approximations, or differentiable masking require additional computations during the selection phase, which may offset part of the computational savings achieved during fine-tuning [148, 434].

Selective fine-tuning can introduce challenges in multi-task or continual learning settings, where different tasks may require distinct parameter subsets, potentially leading to conflicting updates or fragmentation of trainable parameters across tasks. Despite these limitations, selective methods remain attractive due to their conceptual simplicity and minimal memory overhead. Unlike additive approaches such as adapters or prompts, they do not introduce additional modules or embeddings, allowing the base model architecture to remain unchanged.

Reparametrization-Based Methods

Reparametrization methods are motivated by the hypothesis that parameter updates during fine-tuning reside in a low-dimensional subspace [5]. These methods aim to reduce the number of trainable parameters by constraining weight updates to structured forms, most commonly using low-rank decomposition. These approaches can be split into two subclasses: one that fixes the rank of the low-rank decomposition matrices and one that learns and changes the optimal rank during the training.

Static Rank LoRA [186] formulates the update to a frozen weight matrix $W \in \mathbb{R}^{d\times k}$ as $\Delta W = BA$, where $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times k}$ and $r << min(d, k)$ is the rank of matrix BA. The matrices A and B are trained while W remains frozen. So, the

updated weights are:

$$W_{update} = W + \Delta W = W + BA. \tag{3.10}$$

Variants such as VeRA [234] fix the matrices A and B as random and shared across layers and define the weight updates as:

$$\Delta W = \Lambda_b B \Lambda_d A, \tag{3.11}$$

where the scaling vectors b and d are the only trainable parameters and are applied via diagonal matrices Λ_b and Λ_d. TiedLoRA [392] generalizes VeRA by training both matrices and scaling vectors, and AFLoRA [282] further introduces a score to freeze stable parameters during training. DoRA [275] decouples magnitude and direction in weight updates using normalized low-rank matrices.

To improve efficiency of the approach and reduce the number of trainable parameters, the update of the weight matrices can be parametrized by relying on different matrix decompositions, one of the most popular is Singular Value Decomposition (SVD) [1]. For example, CorDA [503], LaMDa [15], PISSA [310], SVFT [268] and Spectral [514] rely on SVD to structure the decomposition of updates. Otherwise, several methods incorporate spectral transformations: RLoRA [241], FouRA [41] and FourierFT [134] rely on the Fourier Transformation or its generalized versions, such as Walsh-Hadamard Transform [400]. Other approaches focus on decomposing the updates in a chain of matrices or in sub-matrices. For example, PRoLoRA [472] slices the low-rank matrices into chunks along the hidden dimension and uses rotation to increase expressiveness without increasing parameter count. DeepLoRA [506] models the update as a chain of matrix products requiring more trainable parameters compared to LoRA, but being more robust to the choice of rank. MeLoRA [391] ensembles mini-LoRA blocks to achieve higher rank without increasing total parameters. VB-LoRA [254] decomposes the update ΔW as the sum of the outer product of rank-one vectors, which are further represented as a concatenation of sub-vectors.

Alternative decompositions, which aim to reduce the number of trainable parameters, include the Kronecker product [171] in KronA [113] and the Tensor Train (TT) product [339] in LoRETTA [504]. SURM [414] proposes unrestricted structured matrices such as Kronecker and low displacement rank matrices (e.g., Toeplitz, circulant), which allow fewer parameters without requiring low rank.

Several approaches focus on enhancing adaptation with additional semantic information. NLoPT [142] integrates LoRA-based task-adaptive pretraining with task-specific n-gram fusion via mutual information. HyperLoRA [290] and PARA[281] apply hypernetworks [300] to generate LoRA parameters dynamically.

A subset of static rank approaches focus on the optimization aspect and on how to optimize the gradients updated effectively. For example, LongLoRA [79] replaces the attention weights with shifted sparse attention for context extension. It

combines LoRA with trainable embeddings and normalization to save computation in long-context scenarios. LoRAGA [471] improves initialization by aligning low-rank updates with the full gradient direction via eigenvector-based initialization. LoRA+ [160] provides learning rate guidance based on scaling theory. Fast LoRA (FLORA) [480] allows each example in a minibatch to have its unique low-rank adapters, facilitating efficient batching, while FLORA [158] (from LoRA to high-rank updates) relies on a novel optimization technique based on sublinear memory for gradient accumulation and momentum calculation. ReLoRA [257] increases update rank by reinitializing and merging LoRA matrices during training. Optimization stability is maintained via partial reset of optimizer state and learning rate warmup. Similarly, LoRASC [252] employs sequential LoRA expert training, where each expert is merged into the base model after one epoch and learns from the residual of the previous expert's output. WGLora [263] decomposes pre-trained weights into magnitude and direction components, and projects gradients into a low-rank space, reducing memory usage while maintaining gradient statistics. C-LoRA [210] employs derivative-free optimization methods [393, 371], i.e. without relying on the back-propagation, to optimize low-rank modules at each layer alternately.

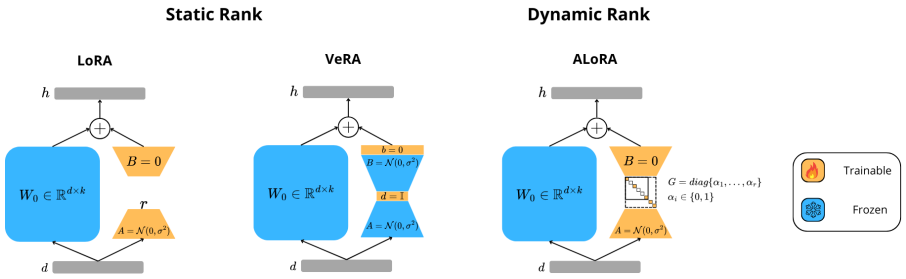

Figure 3.6: Examples of reparametrized methods: on the left Static Rank (LoR and VeRA) and on the right an example of Dynamic Rank (ALoRA).

Dynamic Rank All the previously described LoRA-based methods treat the rank r as a fixed hyperparameter. However, recent studies [518, 452, 106] show that LoRA's is still constrained by its inflexibility in selecting the optimal rank r. Unlike continuous hyperparameters, such as the learning rate, which can be tuned adaptively during the training process, the LoRA rank takes discrete values, and its change will directly alter the model structure. If the rank is set too low, the update may lack sufficient expressive capacity to capture task-specific adaptations; conversely, larger ranks increase memory consumption and computational cost, partially reducing the efficiency gains of the approach [452]. Choosing an excessively large rank can increase training time and computational cost, whereas setting the rank too low may degrade performance. This challenge has motivated the development of dynamic-rank variants that adaptively allocate rank during training,

although these solutions typically introduce additional optimization complexity.

The first proposed approach that introduces a dynamic rank is AdaLoRA [518], which iteratively prune singular values in correspondence to their importance score during training. Based on AdaLoRA findings, SoRA [106] defines a maximum rank, with a trainable sparse gating vector controlling rank during training, while DyLoRA [452] samples the rank at each step from a categorical distribution and truncates A and B accordingly and ERAT-DLoRA [287] enhances this with a dynamic start range. Finally, more recent approaches rely on external architectures to determine the optimal rank, although this comes at the cost of additional computational overhead. For example, ALoRA [280] formulates rank selection as neural architecture search [481], learning gate units that prune or reallocate ranks. Similarly, SalientLoRA [228] constructs a dependency graph of singular values to assign salience scores for rank selection. CapaBoost [427] integrates multiple rank masks with LoRA weights to increase update capacity. AutoLoRA [520] defines the update as a weighted sum of rank-1 matrices, optimizing their weights using meta-learning [121].

Others Several methods extend reparametrized approaches with task-specific architectural elements without relying on a low-rank decomposition as defined in Equation (3.10). GloC [438] introduces latent controller units across layers for bi-directional interaction with frozen representations. QuanTA [81] applies tensor algebra inspired by quantum circuits for efficient high-rank adaptation. RoAD [259] focuses on rotating subspaces of hidden representations, based on the finding that fine-tuning primarily alters angular, not magnitude, components. Kernel-mix [77] adapts kernel structures within attention by re-weighting value components. MuScleLoRA [493] protects against backdoor attacks [75] by integrating frequency-based radial scalings into LoRA, aligning gradients for robust updates. HiFi [147] uses graph analysis to rank attention heads by task relevance, selecting a subset via PageRank for fine-tuning.

Hybrid Methods

Hybrid methods integrate various PEFT techniques to combine the strengths of different approaches in a unified framework. Hybrid methods can be grouped into five different categories based on which approaches are combined. Four categories are previously defined in the literature [256] based on which approaches are combined: Additive + Selective, Additive + Reparametrized, Reparametrized + Selective and Additive + Selective + Reparametrized. We introduce a new category, the so-called Mixture of Experts-enhanced, which groups all the approaches that combine a Mixture of Experts (MoE) architecture with a PEFT approach. We introduce this new category in response to the growing number of PEFT approaches built on Mixture-of-Experts architectures, and we classify it as hybrid because they

integrate a well-known architecture within a PEFT approach.

Additive + Selective The methods in this category propose an Additive approach combined with a selective technique to reduce the number of trainable parameters. For example, Sparse Adapter [168] prunes adapters at the initialization step, deleting the redundant parameters at the early stage and avoiding the time-consuming iterative pruning process. Similarly, [36] propose an approach to pruning adapter layers without any iterative fine-tuning of the model parameters on downstream tasks by using tropical algebra.

Additive + Reparametrized The methods in this group combine additive methods, such as Adapter or Prompt approaches, with reparametrized approaches. These methods are combined in two main ways. In the first, adapter or prompt parameters are themselves reparametrized through low-rank structures, such as Compacter [217], LoRETTA Adapter [504] and MixPHM [208], which compute the adapter's weight as a sum of Kronecker or Tensors products between low-dimensional matrices. These approaches aim to combine the high downstream performances of the adapters while reducing the number of parameters trained compared with the Houlsby adapter [183]. Similarly, LoPA [200] and DePT [419] train low-rank matrices to update prompt vectors.

Otherwise, approaches such as UNIPELT [305], HLPT [143] and MAM [161] combine within a single framework different PEFT models, such as Adapter [183, 354] or Prefix-Tuning [253] and LoRA [186]. While UNIPELT requires the definition of a Gating Function to select iteratively which parameters to train, MAM proposes three different variants of combinations, showing that the best one involves using prefix-tuning in the self-attention layer and the adapter in the FFN layer. A key challenge for this class of methods is how to combine multiple PEFT mechanisms effectively while preserving overall efficiency and thus not introducing too many parameters to be trained. To solve this problem, AutoPEFT [534] defines a search space, which encompasses serial and parallel adapters [183, 354, 168], and an automatic search algorithm, based on Bayesian optimization [125], to find the best combination of PEFT methods for achieving the highest overall performance.

Reparametrized + Selective The approaches in this group combine reparametrized approaches, typically LoRA-based, and then select only a subset of attention parameters to be trained to further reduce the number of trained parameters. For example, RoSA [333] and LongExposure [473] create a task-specific gradient-based sparse mask over the LoRA-updated weights, effectively identifying the most impactful parameters to update. Similarly, PRILoRA [32], LoRAPrune [517], Shears [323], DSEE [76] and RoseLoRA [465] apply different pruning strategies to LoRA low-rank matrices. For example, DSEE [76] derives sparse masks by

selecting the top-magnitude elements in the residual matrix using a thresholding approach. A different strategy is the combination of Selective and Reparametrized methods independently. For example, both S-MAM and U-MAM [238] search for the best combination of BitFit and LoRA, similarly to U-BitFit and S-Bitfit [238] previously cited.

Additive + Selective + Reparametrized The approaches in this group combine all three previously described methods (Additive, Selective and Reparametrized) in a unique architecture. For example, HiWi [258] selects only the FFN of a transformer layer and their weights are the input and output of an external adapter layer. In this way, the FFN parameters are reparametrized with an output of an external adapter, which is the only trainable component during the training phase. S_4 [69] defines and explores design choices across four dimensions, i.e. layer grouping, parameter allocation, tunable groups, and strategy assignment. S^3PET [189], S^3-Delta [188] and BIPEFT [63] search for the best configuration while applying Selective (BitFit [31] and the training of Layer Normalization weights), Adapter-based [183], Reparametrized (LoRA [186]) modules. While S^3PET and S^3-Delta define a unified search space where each module is conditionally activated, BIPEFT disentangles binary module selection from rank dimension search.

Mixture of Experts-enhanced Given the growing interest in combining Mixture-of-Experts (MoE) architectures with PEFT methods, we define a new category: MoE-enhanced PEFT. MoE architectures offer a scalable solution that preserves computational and parameter efficiency by dynamically activating a subset of specialized experts during inference.

A typical MoE layer [417], originally applied between LSTM layers [177], consists of a set of experts $\{E_1, E_2, ..., E_n\}$, which are lightweight neural modules, and a Gating network $\mathbb{G}$, which produces a sparse selection vector. For input x, the MoE output is defined as:

$$\sum_{i=1}^{n} \mathbb{G}_i(x) E_i(x). \tag{3.12}$$

Experts are often lightweight models, ranging from simple linear projections [475] to bottleneck architectures [205]. Gating functions are commonly implemented using Softmax [213], Gumbel-Softmax [299], or task- or domain-specific classifiers [361], depending on the granularity of routing (at token, sentence, or document level).

As shown in Figure 3.7, MoE layers have been integrated across diverse PEFT strategies. For instance, AdaMix [475] introduces MoE in adapter layers, ATTEMPT [13] embeds it into soft-prompt tuning, and MoELoRA [273, 274] applies it to LoRA-based modules. These methods illustrate the adaptability of MoE

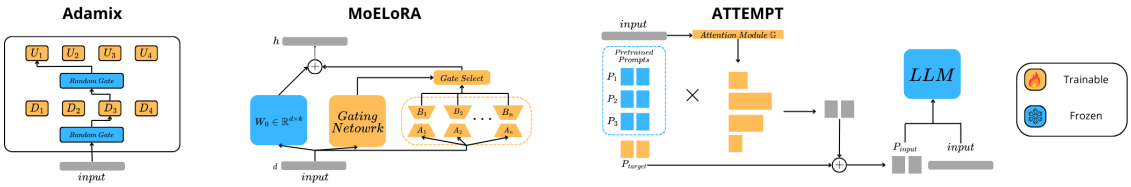

Figure 3.7: Examples of MoE-enhanced PEFT methods: AdaMix, MoELoRA and ATTEMPT.

within all PEFT approaches. MoE based approaches can be classified based on the PEFT methods they rely or on the definition of the gating function and how they train the experts.

Several MoE-enhanced PEFT models extend LoRA by redefining the structure or routing of its components. LoraRetriever [532] combines two strategies: Mixture of LoRAs, which aggregates the outputs of task-specific experts, and Fusion of LoRAs, which merges their parameters. AlphaLoRA [373], inspired by MoLA [129], allocates fewer experts to well-trained layers using Heavy-Tailed Self-Regularization Theory [307, 306]. MoSLoRA [490] partitions LoRA matrices into subspaces, allowing multiplication of sub-matrix experts. HydraLoRA [445] shares a global A matrix while routing among multiple B matrices. TSRA [513] treats each LoRA rank as a separate expert. MiLoRA [515] and FanLoRA [444] take a modular approach, considering entire LoRA blocks (e.g., Q, K, V) as experts, and route using prompt-aware static mechanisms.

MoE integration extends beyond LoRA. ATTEMPT [13] and MSP [66] define expert sets of soft prompts trained on diverse tasks. SMoP [85] optimizes sparse short prompts, each tuned to a specific data region. ESFT [477] selectively tunes only the most relevant experts per task, freezing others to enhance efficiency.

MoE-enhanced methods also differ by gating mechanisms. MoLE [248] utilizes Gumbel-Softmax [299, 202] for adapter routing, while AdaMix [475] employs stochastic gating [541]. MEFT [157] activates expert subsets to reduce inference cost, and MPOE [130] integrates tensor-decomposed experts based on LoRETTA [504]. MoLoRA and MoV [510] define experts as lightweight modules (e.g., LoRA or $(IA)^3$), trained in parallel via a shared router. Another design axis concerns domain- or task-specific routing, such as MixDA [104], which defines domain-specific adapters. Poly [361] and C-Poly [464] treat task adapters as composable skill modules, a concept extended in MHR [55] via low-rank parallelism. MoELoRA [273] builds multiple LoRA experts to enable task-specialization in a unified setup. MoDULA [293] introduces a universal expert alongside domain-specific ones, while MoDULA-Res incorporates residual connections to merge general and specific information paths. MoA [119] assigns experts based on domain metadata using sequence-level routing. PEMT [267] combines pre-trained adapters with a task-correlation-based gating mechanism, encoding tasks via prompt vectors. Similarly,

π-tuning [486] pretrains adapter/prompt experts per task and selects them by measuring similarity to the target task.

3.3.4 Which datasets are most frequently used to evaluate PEFT techniques across different NLP tasks?

To assess the effectiveness of PEFT methods across diverse NLP tasks, researchers have relied on a set of well-established benchmark datasets. Table 3.2 summarizes the main datasets used in the current PEFT literature. We report the datasets that have been used at least three times in the reviewed papers. For each dataset, we report the name, the task it tackles, the number of training, validation and test data and, finally, the metrics used to evaluate.

GLUE The General Language Understanding Evaluation (GLUE) benchmark [461] includes eight NLU tasks spanning linguistic acceptability (CoLA [478]), sentiment analysis (SST-2 [423]), paraphrase detection (MRPC [108], STS-B [61], QQP), and natural language inference (MNLI [482], QNLI [381], RTE [96, 21, 138, 33]). Most studies, [183, 102, 186, 520] to name a few, do not include the WNLI dataset [243] as there are issues with the construction of this dataset such that BERT-like models do not outperform the majority baseline.[1] The main issue with GLUE datasets is that the test sets are not publicly available but are hidden [461]. To evaluate a model on the GLUE test sets, researchers have to submit their predictions on the GLUE form [2], with a limited number of submissions. For these reasons, a widely used strategy [475, 353, 186, 500] is to evaluate the proposed approaches on the validation set, whose labels are publicly available.

SuperGLUE Building on GLUE, SuperGLUE [463] includes harder benchmarks such as BoolQ [89], MultiRC [229], and ReCoRD [522] for QA; RTE for inference; WIC [358] for word sense disambiguation; and commonsense reasoning tasks like CB [100] and WSC [243]. Similarly to GLUE, SuperGLUE test sets are not publicly available and, for this reason, a widely used strategy [353, 504, 81, 63] is to evaluate on the validation set.

Question Answering SQuAD v1.1 and v2.0 [382, 380] are the two most popular datasets for question answering. They consist of questions posed by crowd workers on a set of Wikipedia articles, where the answer to each question is a segment of text from the corresponding reading passage. These tasks are treated as a sequence

[1] See (12) in gluebenchmark.com/faq.

[2] https://gluebenchmark.com/submit

labelling problem, where the aim is to predict the probability that each token is the start and end of the answer span.

Natural Language Generation from Structured Data Common NLG benchmarks include E2E [338] in the restaurant domain, WebNLG [135] using RDF triples from DBpedia, and DART [324], which aggregates data from table-to-text and question generation datasets such as WikiSQL [533], WikiTableQuestions [348].

Commonsense Reasoning In this section, we describe the most popular dataset on Commonsense Reasoning on which PEFT methods are evaluated. ARC-c and ARC-e [91] are the Challenge Set and Easy Set datasets, respectively, of grade-school level, containing multiple-choice science questions. OBQA [312] contains questions requiring multi-step reasoning, which requires commonsense knowledge and rich text comprehension. SIQA's questions [411] are about people's actions and their social implications. WinoGrande [405] is a dataset designed for a fill-in-a-blank task with binary options to choose the right option for a given sentence, which requires commonsense reasoning. PIQA's questions [37] have two solutions requiring physical commonsense. HellaSwag [511] are commonsense NLI questions including a context and several endings which complete the context.

Text Summarization The most popular Text Summarization datasets are XSUM [325] and CNN/Daily Mail [174]. In detail, XSUM consists of BBC articles and single-sentence summaries, which cover a wide variety of domains, for example News, Sports, Weather, Science and Arts, to name a few. CNN/Daily Mail is composed of articles collected starting from April 2007 to April 2015 from both CNN and Daily Mail. XSum and CNN/Daily Mails have a similar number of articles in both validation and test sets. However, documents and summaries in XSum are shorter than CNN and Daily Mail ones, because XSum focuses on the task of Extreme Summarization in which the summary is a short sentence about the topic of the news.

Named-Entity Recognition and Semantic Role Labelling CoNLL-2003 [408] is a Named-Entity Recognition (NER) dataset, which covers two languages, English and German. The English data was taken from the Reuters Corpus. This corpus consists of Reuters news between 1996 and 1997. The text for the German data was taken from the ECI Multilingual Text Corpus, in particular, the data are articles from a German newspaper written in 1992. CoNLL-2004 [60] consists of six sections of the Wall Street Journal part of the Penn Treebank, which annotates the Penn Treebank with verb argument structure. In detail, given a sentence, the task consists of analyzing the propositions expressed by some target verbs of the sentence. In particular, for each target verb, all the constituents in the sentence which

fill a semantic role of the verb have to be extracted. OntoNotes [366] is a large corpus comprising various genres of text, e.g. news, speech, broadcast and talk shows, in three languages, i.e. English, Chinese, and Arabic, with structural information, such as syntax and predicate-argument structure, and shallow semantics, as word sense linked to an ontology and coreference.

Machine Translation WMT16 is a shared task, which included five machine translation (MT) tasks (standard news, IT-domain, biomedical, multimodal, pronoun). Regarding news translation, there are twelve translation sub-tasks, between English and each of Czech, German, Finnish, Russian, Romanian, and Turkish. Since it is one of the most used datasets for evaluating PEFT approaches, we focus on the sub-task English to Romanian.

3.3.5 What performance metrics are applied to measure the effectiveness of PEFT approaches?

The metrics used to evaluate PEFT approaches are task-specific:

1. Text Classification: GLUE tasks such as CoLa and STS-B are evaluated using Matthews's [308] and Pearson's [349] correlation, respectively. MRPC is evaluated using F1 [83]. QQP is evaluated using either F1, as by [183], or accuracy, as by [186], or both, as by [475]. All the other GLUE tasks are evaluated using Accuracy [83], as all SUPERGLUE tasks.

2. Question Answering: Question answering tasks are evaluated using Exact Match (EM) and F1 [83]. For exact match, a prediction receives a score of 1 if it exactly matches the reference answer and 0 otherwise. Commonsense reasoning tasks, such as WinoGrande, ARC, PIQA, SIQA, OBQA and HellaSwag are all evaluated through accuracy.

3. Natural Language Generation: NLG tasks are evaluated using several metrics: BLEU [344], NIST [107], METEOR [19], ROUGE-L [262], CIDer [455], TER [422]. BLEU [344] is a precision-based metric commonly used to evaluate the quality of machine-translated text by comparing n-grams in the generated text to those in a reference translation. The NIST [107] metric is similar to BLEU but gives more importance to the rarer, informative n-grams by weighting them more heavily. METEOR [19] evaluates translations by aligning them to reference texts using synonymy, stemming, and paraphrasing, providing a recall-oriented evaluation. ROUGE-L [262] measures the longest common subsequence between a generated text and a reference text, capturing the similarity of sequence structure between them. CIDEr [455] emphasises precision and also considers term frequency-inverse document frequency (TF-IDF) weighting to downplay frequent but less informative n-grams. TER

[422] measures the number of edits (insertions, deletions, substitutions, and shifts) required to change a system output into one of the references.

All cited metrics must be maximised, except for TER.

Which models are used to evaluate PEFT approaches? And on which datasets?

In this section, we investigate the pre-trained language model and the dataset used to evaluate PEFT approaches. The main differences in the evaluation strategies are the pre-trained language model used as a backbone and how to work when the test set is hidden, as in GLUE. For each dataset described in the previous paragraph, we report, for all the PEFT approaches evaluated on that dataset, the pre-trained language model used and, for GLUE and SuperGLUE datasets, if the evaluation is on the test or validation sets. We report the results of this analysis in Table 3.3, 3.4 and 3.5 for GLUE, 3.6 and 3.7 for SuperGLUE and 3.8, 3.9, 3.10 for the remaining datasets. In Tables 3.3, 3.4, 3.5, 3.6, and 3.7, the values reported in the *Evaluated on* column indicate the dataset split used for evaluation. This can correspond to the test set, the development set (denoted as *Dev.*), or a mixed setting. The latter refers to cases where results are reported on a validation set, but, differently from the standard *Dev.* setting, the authors construct a custom validation split from the original training data for hyperparameter tuning [217, 523]. The most commonly adopted mixed evaluation strategy follows Karimi Mahabadi, Henderson, and Ruder [217]. In this setting, the authors reserve a subset of $1k$ samples from the training set for validation, while the original validation split is used as the test set. For smaller datasets (e.g., RTE, MRPC, STS-B, CoLA, COPA, WiC, CB, BoolQ, MultiRC), where the total number of samples is limited, the original validation set is instead split into two halves, one used for validation and the other for testing.

GLUE

Which GLUE datasets are most frequently used? Across all PEFT approaches, SST-2, RTE, MRPC, and QNLI emerge as the most frequently used benchmarks. In additive approaches, SST-2 and MRPC appear in 85.2% of the analyzed models, followed by RTE (83.3%) and MNLI (75.9%). Selective and reparametrized approaches place even greater emphasis on SST-2 (98%), RTE (90%), QNLI (86%), and MRPC (86%), reflecting a strong focus on core natural language understanding tasks. Hybrid approaches slightly shift the priority to RTE (92.5%) and SST-2 (90%), while also including QNLI (87.5%), MRPC (80%), MNLI (77.5%), and QQP (77.5%). Tasks such as CoLa (ranging from 61.1% in additive to 76% in selective/reparametrized and 65% in hybrid) and STS-B (59.3%

to 78% and 65%, respectively) are less commonly used, highlighting a general underrepresentation of linguistic acceptability and semantic similarity tasks.

Which PLMs are most frequently used as baselines? Regarding pretrained language models, RoBERTa (both base and large) and T5 (especially T5-base) are the most commonly used baselines, appearing in 36%, 28%, and 24% of models respectively. BERT-base and BERT-large remain frequently used at 18% and 11%. More recent large language models, such as LLaMA-2-7B, appear in only 7% of all approaches, while MPT-7B and DeBERTa-v3-base appear in 4–8% of models. In detail, LLaMA usage is particularly muted in additive approaches, appearing in only 1.9% of models, and slightly more in selective/reparametrized (8%) and hybrid approaches (7.5%). Rarely considered models include Electra, Flan-T5-XL, Bloom-7B, Mistral-7B, GPT-2, BabyAI, and Falcon. The limited use of billion-parameter models is likely due to the high memory and computational cost associated with adding parameters, defining adapters, and performing training and inference, especially in additive PEFT approaches.

On which splits are models evaluated? Evaluation splits are highly variable across the literature. Among 144 models analyzed across all PEFT approaches, 50.7% report results on the development set, 17.4% on the test set, 31.9% adopt mixed evaluations. Overall, development sets dominate across all approaches while test sets are underrepresented due to the required limited number of submissions on the GLUE platform.

SuperGLUE

Which SuperGLUE datasets are most frequently used? Across all PEFT approaches, BoolQ clearly emerges as the most frequently used SuperGLUE benchmark, appearing in 70.8% of the analysed models. It is followed by RTE (58.5%), CB (53.8%), and Copa (56.9%), confirming a strong preference for relatively simple inference and classification tasks. WIC is moderately represented (44.6%), while WSC appears in only 35.4% of models, despite its importance for coreference resolution. More complex reasoning benchmarks such as MultiRC (40%) and ReCoRD (33.8%) are comparatively underrepresented, highlighting a general tendency to avoid tasks requiring multi-hop reasoning and long-context understanding. Overall, these trends indicate that, similarly to GLUE, PEFT research on SuperGLUE prioritizes datasets that are easier to optimize and evaluate, while more challenging benchmarks are less consistently explored.

Which PLMs are most frequently used as baselines? Across SuperGLUE-based PEFT methods, encoder-based models such as BERT-base, RoBERTa (base and large), and T5 (base and large) remain the dominant backbones, collectively

accounting for the large majority of approaches. T5-based models are particularly prominent, appearing in more than half of the analyzed methods, especially in additive and hybrid approaches. RoBERTa (base and large) and BERT variants are also widely adopted as standard baselines. More recent large language models, such as LLaMA (including LLaMA-2), appear in approximately 20% of the approaches, primarily within hybrid and reparametrized methods. Despite their growing popularity, their adoption remains limited compared to encoder-based PLMs, likely due to the higher computational and memory costs associated with adapting large-scale generative models.

On which splits are models evaluated? Evaluation protocols on SuperGLUE are heterogeneous across the literature. Among the analyzed models, 53.8% report results on the development set, 15.4% on the test set, and 30.8% adopt mixed evaluation strategies. Development sets clearly dominate across all approaches, while test-set evaluations remain limited due to restricted access and leaderboard-based evaluation. Mixed evaluation protocols are particularly common in hybrid approaches and often involve re-splitting validation sets for smaller datasets (e.g., CB, RTE, WSC) or extracting validation subsets from the training data for larger tasks. While these strategies enable broader evaluation, they introduce inconsistencies that hinder direct comparison across methods and highlight the need for more standardized benchmarking practices in SuperGLUE-based PEFT research.

3.3.6 Other Datasets

Which additional datasets beyond GLUE and SuperGLUE are most frequently used? Across the analyzed PEFT approaches, several benchmarks outside the GLUE and SuperGLUE datasets are commonly used to evaluate performance on a broader set of tasks. Among them, WinoGrande appears most frequently, being used in 27 out of 92 models (about 29%), reflecting the importance of commonsense reasoning benchmarks. Other frequently used datasets include E2E and HellaSwag, both appearing in 20% of the models, while SQuAD v1 and GSM-8K are each used in 20 models (22%). ARC-E/ARC-C, PIQA, and OBQA are also widely adopted, appearing in 16–17 models (17–18%), while WebNLG and XSum appear in 13 and 12 models respectively (about 13–14%). Several datasets appear slightly less often, such as CoNLL03 (11 models, about 12%), DART and WMT16 En-Ro (8 models each, 9%), and SQuAD v2 (7%). Finally, CoNLL04 and OntoNotes appear in 4 models each (4%), while IWSLT is rarely used (2%).

Overall, the distribution shows that, compared with GLUE-style sentence-level classification tasks, PEFT methods are evaluated on a much more heterogeneous set of benchmarks covering question answering (SQuAD), structured data-to-text

generation (E2E, DART, WebNLG), summarization (XSum), reasoning and commonsense inference (WinoGrande, HellaSwag, ARC, PIQA, SIQA, OBQA), information extraction (CoNLL03, CoNLL04, OntoNotes), and machine translation (WMT16 En-Ro, IWSLT). As already observed in the previous paragraph, hybrid approaches are often evaluated with larger language models. Consequently, they appear more frequently in benchmarks that emphasize reasoning or complex generation capabilities, such as commonsense reasoning datasets (e.g., ARC and PIQA), question answering benchmarks (e.g., SQuAD), and translation tasks (e.g., WMT16 and IWSLT), which are typically used to evaluate the capabilities of larger LLMs. In contrast, GSM-8K does not appear in additive approaches, which are more commonly applied to smaller pre-trained models with millions rather than billions of parameters. Similarly, named entity recognition datasets such as CoNLL03, CoNLL04, and OntoNotes are not used in hybrid approaches, reflecting the tendency of hybrid methods, often evaluated with billion-parameter LLMs, to focus on reasoning, generation, and complex language understanding tasks rather than traditional information extraction benchmarks.

3.4 Efficiency of Parameter-Efficient Fine-Tuning

Efficiency constitutes one of the primary motivations underlying Parameter-Efficient Fine-Tuning (PEFT) methods and has been extensively discussed in recent literature [256, 450]. In general terms, efficiency can be interpreted as the relationship between the computational resources required by a method and the quality of the results it produces [413]. A method is therefore considered more efficient if it achieves comparable or superior performance while reducing resource consumption, such as computation, memory, or training time.

Following Schwartz et al. [413], the overall cost associated with producing a result R can be expressed as:

$$Cost(R) \propto E \cdot D \cdot H, \tag{3.13}$$

where E denotes the computational cost of processing a single training example, D is the dataset size, and H represents the number of training runs required for hyperparameter search. This formulation highlights that efficiency is not solely determined by the complexity of individual training steps, but also by the broader experimental process needed to obtain competitive performance. In the context of PEFT, methods are typically evaluated on shared benchmark datasets; consequently, D can be assumed constant across comparisons. Differences in efficiency therefore arise primarily from the execution cost per example (E) and the hyperparameter search cost (H). The execution cost E depends on several factors,

including the number of trainable parameters, the computational complexity of forward and backward passes, memory requirements, and training time per iteration. Among these, the number of trainable parameters represents the only hardware-independent quantity, whereas the remaining factors depend on the specific computational infrastructure. From this perspective, different PEFT families exhibit distinct efficiency profiles. Additive approaches, such as adapter-based methods, introduce additional task-specific parameters through bottleneck architectures. Although these modules are designed to remain lightweight, they typically add between 0.5% and 4% of the original model parameters, depending on the bottleneck dimension and insertion strategy. Moreover, since these components remain active at inference time, they introduce additional computations, which may result in an increase in latency. Selective methods represent the most parameter-efficient class of approaches. Techniques such as BitFit [31] update less than 0.1% of the model parameters, leading to minimal memory requirements and reduced optimization cost. Because these methods operate directly on existing parameters, they do not introduce additional modules and therefore preserve both the original architecture and inference latency. Furthermore, structural selection strategies often do not require additional PEFT-specific hyperparameters, simplifying the optimization process. In contrast, unstructured masking approaches may introduce auxiliary parameters associated with learnable masks. Reparametrization-based methods exhibit similar advantages in terms of inference efficiency. By constraining updates to low-rank structures, these approaches avoid introducing additional operations during deployment. In practice, methods such as LoRA typically train approximately 0.1-0.5% of the total parameters, depending on the chosen rank and the number of modified layers. This yields a favourable trade-off between parameter efficiency and expressive capacity, often outperforming additive approaches in terms of resource utilization while maintaining strong empirical performance. Hybrid approaches based on Mixture-of-Experts (MoE) architectures introduce additional considerations. These methods incorporate multiple expert modules along with a gating mechanism, increasing both the number of parameters and the architectural complexity. The overall training cost and memory requirements grow with the number of experts, as multiple modules must be stored and potentially evaluated. At inference time, the computational overhead depends on the routing strategy: sparse routing activates only a subset of experts per input, limiting computational cost, whereas dense routing may significantly increase latency. Consequently, the efficiency of MoE-based PEFT methods depends critically on the design of the expert configuration and routing mechanism. Beyond computational aspects, efficiency has also been investigated from an environmental perspective. The computational requirements of model training translate directly into energy consumption and associated carbon emissions [256]. The total energy usage depends on the number of floating-point operations (FLOPs), the hardware employed, and the overall training duration. Carbon emissions can then be estimated based on

the energy consumption and the carbon intensity of the energy source. Therefore, reducing computational complexity, training time, and memory usage can significantly decrease both energy consumption and CO_2 emissions. Recent empirical studies [48] provide quantitative comparisons across PEFT methods. For instance, when fine-tuning BERT-base on the RTE dataset for a single epoch on an NVIDIA A100 GPU, LoRA requires approximately 3.09 TFLOPs per epoch, compared to 3.12 TFLOPs for adapter-based configurations. Although the difference is relatively small, it translates into lower training time, memory usage, and environmental impact. In contrast, hybrid approaches such as AdaMix exhibit substantially higher computational costs, as they process each batch multiple times, resulting in increased FLOPs, energy consumption, and emissions.

The second component affecting efficiency, namely the hyperparameter search cost H, depends on the number of method-specific parameters that must be tuned in addition to standard optimization hyperparameters (e.g., learning rate, batch size, number of epochs). Different PEFT families introduce distinct design parameters. For example, adapter-based methods require selecting the bottleneck dimension, which directly controls the trade-off between performance and computational overhead. Common choices in the literature include dimensions such as 64, 128, and 256 [353, 47]. Similarly, reparametrization methods such as LoRA require tuning the rank parameter, which determines the capacity of the low-rank update. MoE-based approaches introduce additional hyperparameters, most notably the number of experts, which influences both model capacity and computational cost.

Overall, these observations highlight the fundamental trade-offs underlying PEFT design. Selective methods achieve the highest degree of parameter efficiency with minimal computational overhead, but may be limited in expressive capacity. Reparametrization-based approaches provide a favourable balance between efficiency and performance, while additive methods offer greater flexibility at the cost of additional parameters and moderate inference overhead. Finally, MoE-based approaches increase modelling capacity but introduce additional complexity, making their efficiency highly dependent on architectural design choices.

3.4.1 Quantization-aware PEFT as an orthogonal efficiency axis

In addition to the efficiency dimensions discussed above, a parallel line of research has explored the integration of PEFT with weight quantization. In this survey, we treat quantization-aware PEFT as an orthogonal efficiency axis rather than a primary dimension of analysis. While PEFT methods reduce the number of trainable parameters, quantization focuses on compressing the storage and computation of the frozen backbone. The combination of these two strategies enables further reductions in memory usage, but falls outside the main taxonomy considered in this review. Therefore, we provide here a brief overview and leave

a systematic analysis to future work. One prominent example is QLoRA [101], which combines a frozen low-bit quantized model with low-rank adaptation. Given a pretrained weight matrix $W \in \mathbb{R}^{d\times k}$, QLoRA first applies a low-bit quantization function $Q(W)$, typically in 4-bit precision, while modelling the task-specific update through a low-rank decomposition:

$$W' = Q(W) + \Delta W, \qquad \Delta W = BA, \tag{3.14}$$

where $W \in \mathbb{R}^{d\times k}$, $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times k}$, $r \ll \min(d, k)$. During training, gradients are propagated through the dequantized representation of $Q(W)$, while only the parameters of A and B are updated. To further reduce memory consumption, QLoRA introduces several key techniques, including NormalFloat4 (NF4) quantization tailored to normally distributed weights and double quantization to compress quantization constants. These design choices enable the fine-tuning of models with tens of billions of parameters on a single GPU, substantially lowering hardware requirements. Quantization-aware PEFT has become increasingly relevant in practice, as it directly addresses the memory bottlenecks associated with large-scale models and is supported by widely used PEFT libraries. More broadly, quantization and parameter-efficient adaptation can be seen as complementary strategies: the former reduces the footprint of the frozen backbone, while the latter limits the number of trainable parameters required for task adaptation. Together, they enable scalable and resource-efficient fine-tuning under constrained computational settings. However, a comprehensive treatment of quantized PEFT would require a dedicated analysis, including specific search criteria, evaluation protocols, and comparison frameworks. For this reason, we limit our discussion to highlighting its role as an important complementary direction, and leave a systematic review of quantization-aware PEFT methods as a promising avenue for future research.

3.5 Discussion and Future Directions

The systematic review reported in this Chapter provides a comprehensive overview of the rapidly evolving landscape of Parameter-Efficient Fine-Tuning (PEFT) techniques for Natural Language Processing. By analysing the selected literature, several recurring trends and methodological patterns emerge, shedding light on both the progress achieved in recent years and the challenges that remain open.

A first notable observation concerns the progressive diversification of PEFT approaches. Early contributions in this field were largely dominated by adapter-based methods [183, 354], which demonstrated that introducing a small number of additional parameters is sufficient to effectively adapt large pre-trained models while preserving their general knowledge. Subsequently, the design space expanded with the introduction of reparameterization-based techniques, most prominently

low-rank adaptation methods such as LoRA [186], which further reduced the number of trainable parameters while maintaining strong empirical performance. More recently, the literature has increasingly explored hybrid approaches that combine multiple PEFT mechanisms within a unified framework, such as Compacter [217] and UniPELT [305]. This evolution reflects a broader shift towards compositional architectures, where different parameter-efficient strategies are integrated to balance flexibility, efficiency, and expressiveness. Despite these advances, our analysis highlights several limitations in current research practices. In particular, the lack of standardized evaluation protocols remains a major obstacle. Existing studies often rely on different model backbones, datasets, and experimental configurations, making direct comparisons between methods difficult. This fragmentation is further compounded by the limited range of evaluation metrics typically considered. While most works report task performance and parameter efficiency, other relevant aspects, such as training time, inference latency, memory consumption, and environmental impact, remain underrepresented in empirical studies. However, these factors are critical for real-world deployment, especially when PEFT methods are applied to large-scale models or resource-constrained environments. Another important limitation concerns the scope of empirical evaluation. Many studies focus on a relatively small set of benchmark datasets and model architectures, such as encoder-based models (e.g., BERT) or autoregressive models (e.g., GPT-style architectures). While this facilitates comparison within specific settings, it also limits the understanding of how PEFT methods generalize across tasks, domains, and model families. As a result, the robustness and transferability of many approaches remain only partially explored. Based on these observations, we identify several directions that could contribute to the maturation of the field. A primary priority is the development of standardized evaluation frameworks that enable fair and reproducible comparisons across methods. In this regard, we propose a unified benchmarking protocol grounded in widely adopted datasets and model architectures. Specifically, we suggest evaluating PEFT methods on the GLUE development set using BERT-base and RoBERTa-large, on SuperGLUE development sets using T5-base, on data-to-text generation tasks such as E2E, DART, and WebNLG using GPT-2 medium, and on commonsense reasoning benchmarks such as ARC-E/C, PIQA, and OpenBookQA using LLaMA-2 or LLaMA-3. These choices reflect the most frequently used configurations identified in our review and therefore facilitate comparison with existing work. In addition to task performance, we argue that future studies should adopt a more comprehensive evaluation protocol that includes efficiency-related metrics. In particular, each method should report the number of trainable parameters, training time, inference latency, GPU memory consumption, and environmental impact, following recent efforts in this direction [48]. Such a multidimensional evaluation would provide a more realistic assessment of the trade-offs between efficiency and performance. Beyond evaluation, the literature also points to several promising research directions. The increasing

interest in hybrid PEFT frameworks suggests a move towards more flexible and modular adaptation strategies, where different techniques can be combined and dynamically configured. Similarly, the scalability of PEFT methods to very large language models remains an active area of research, particularly in conjunction with complementary techniques such as quantization. Finally, some limitations of the present review should be acknowledged. Although the systematic search strategy was designed to capture the most relevant studies, it is possible that some recent contributions or works published outside the selected venues were not included. Moreover, the analysis relies on reported experimental results, which may vary significantly in terms of setup and evaluation criteria. As a consequence, the conclusions drawn in this study should be interpreted as qualitative trends rather than definitive quantitative comparisons. Overall, this survey provides a structured synthesis of current PEFT research and highlights both its rapid progress and its remaining challenges. As the scale and complexity of language models continue to grow, parameter-efficient adaptation techniques are likely to play an increasingly central role in enabling scalable, sustainable, and accessible NLP systems.

3.6 Conclusions

In this Chapter, we systematically reviewed Parameter-Efficient Fine-Tuning (PEFT) techniques for adapting large language models to NLP tasks. By analyzing research published between 2019 and 2024, we introduced a structured taxonomy of PEFT approaches, i.e. including additive methods, selective parameter updates, reparameterization strategies, and hybrid frameworks, and examined their underlying design principles, evaluation practices, and application domains. Our analysis reveals several challenges related to the evaluation protocols. Additionally, practical considerations such as inference latency, memory footprint, robustness, and deployment efficiency remain underreported in empirical studies. Our findings should be interpreted in light of the review covering literature covering up to 2024 and therefore does not include developments in 2025 and early 2026. Finally, we argue that future progress in PEFT should be assessed not only in terms of downstream performance or parameter efficiency, but also with respect to reproducibility, training cost, inference latency and memory usage. Accordingly, we suggest a unified evaluation protocol to evaluate PEFT approaches under the same conditions. By clarifying both the methodological landscape and the gaps in current evaluation practice, the survey reported in this Chapter aims to guide future research toward more rigorous, comparable, and practically meaningful advances in parameter-efficient fine-tuning.

Corpus	Task	#Train	#Dev	#Test	Metrics
		GLUE			
		Single-Sentence Classification			
CoLA	Acceptability	8.5k	1k	1k	Matthews corr.
SST	Sentiment	67k	872	1.8k	Accuracy
		Pairwise Text Classification			
MNLI	NLI	393k	20k	20k	Accuracy
RTE	NLI	2.5k	276	3k	Accuracy
QQP	Paraphrase	364k	40k	391k	Accuracy/F1
MRPC	Paraphrase	3.7k	408	1.7k	Accuracy
QNLI	QA/NLI	108k	5.7k	5.7k	Accuracy
		Text Similarity			
STS-B	Similarity	7k	1.5k	1.4k	Pearson corr.
		Super GLUE			
		Question Answering			
BoolQ	QA	9.4k	3.3k	3.2k	Accuracy
MultiRC	QA	5.1k	953	1.8k	$F1_a$/EM
COPA	QA	400	100	500	Accuracy
ReCoRD	Multiple-Choice QA	101k	10k	10k	F_1/EM
		Natural Language Inference			
CB	NLI	250	57	250	Accuracy/F1
RTE	NLI	2.5k	278	300	Accuracy
		Word Sense Disambiguation			
WiC	WSD	6k	638	1.4k	Accuracy
		Coreference Resolution			
WSC	coeref.	554	104	146	Accuracy
		SQuAD			
SQuAD v1.1	QA	87.6k	10.6k	9.5k	EM/F1
SQuAD v2.0	QA	130.3k	11.9k	8.8k	EM/F1
		Natural Language Generation from Structured Data			
E2E	End-to-end	42k	4.7k	4.7k	BLEU,NIST,MET,ROUGE-L,CIDer
WebNLG	RDF to text	35.4k	4.5k	5.1k	BLEU,TER
DART	Table to text	62.6k	6.9k	12.5k	BLEU,TER
		Commonsense Reasoning			
WinoGrande	Fill-in-a-Blank	63.2K	NA	1,267	Accuracy
PIQA	Physical-based QA	16.1K	NA	1,830	Accuracy
SIQA	Social-based QA	33.4K	NA	1,954	Accuracy
ARC-e	Grade-school level, multiple-choice science QA	1.1K	NA	2,376	Accuracy
ARC-c	Grade-school level, multiple-choice science QA	2.3K	NA	1,172	Accuracy
OBQA	Multi-step reasoning QA	5.0K	NA	500	Accuracy
HellaSwag	NLI	39.9K	NA	10,042	Accuracy
		Text Summarization			
XSum	News Summarization	204K	11.3K	11.3K	Rouge-1, Rouge-2, Rouge-L
CNN/Daily Mail	News Summarization	280K	13K	11K	Rouge-1, Rouge-2, Rouge-L
		Named-entity recognition and semantic role labelling			
CoNLL03	NER	946	216	231	Accuracy
CoNLL04	Semantic Role Labelling	922	231	288	Accuracy
OntoNotes	NER	96K	14K	12K	Accuracy
		Machine Translation			
WMT 2016 en-ro	Translation from English to Romanian				BLEU

Table 3.2: Summary of benchmarks for evaluating PEFT methods. We report the name of the benchmark, the task addressed, dataset sizes and evaluation metrics.

PEFT	MNLI	QNLI	SST-2	QQP	MRPC	CoLa	RTE	STS-B	Evaluated on	PLMs
Additive										
Houlsby	✔	✔	✔	✔	✔	✔	✔	✔	Test	BERT-Large
Pfeiffer	✔	✘	✔	✔	✔	✘	✔	✔	Dev	BERT-base
AdaKron	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base, RoBERTa-Large
Adaptable Adapter	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-Large
CoDA	✔	✘	✘	✘	✘	✘	✔	✘	Dev	T5-base, large and XL
CoDA	✔	✔	✔	✔	✔	✔	✔	✔	Test	BERT-base and large, RoBERTa-base and large
Learned Adapter	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	RoBERTa-Large
Learned Adapter	✘	✘	✔	✘	✘	✘	✔	✘	Mixed	DeBERTa-large, GPT-2-large
Adaptable Adapter	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-base
Hadamard Adapter	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base and Large, RoBERTa-base and Large, BART-base and Large, DeBERTa-base and Large, Electra-base and Large
Tiny Attention Adapter	✔	✔	✔	✔	✔	✔	✔	✔	Dev, Test	RoBERTa-Large
AdapterFusion	✔	✘	✘	✔	✔	✘	✔	✘	Dev	BERT-base
Hyperformer	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-small and base
LayerConnect	✔	✔	✔	✔	✔	✔	✔	✔	Dev	T5-Large
HyperEncoder	✔	✔	✔	✔	✔	✔	✔	✔	Mixed and Dev	BERT-base, RoBERTa-base and Large
PHA	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
Adapters Mixup	✘	✔	✔	✘	✔	✘	✘	✘	Dev	BERT, RoBERTa
Udapter	✔	✘	✘	✘	✘	✘	✘	✘	Dev	BERT-base
Inspirational Pointer	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
ScaLearn	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	RoBERTa-base and Large
HyperPelt	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
Prefix subspaces	✔	✔	✔	✔	✔	✔	✔	✔	(Few-shot) Mixed	BERT-base
PIP	✘	✘	✘	✘	✔	✘	✘	✘	Mixed	BART-base
Prompt-Tuning	✘	✘	✘	✔	✔	✘	✘	✘	Mixed	T5
IDPG	✘	✔	✘	✔	✔	✘	✔	✔	Dev	RoBERTa-Large
LPT	✔	✔	✔	✔	✔	✘	✔	✘	Dev	RoBERTa-Large
LPT	✘	✘	✘	✘	✔	✘	✔	✘	Dev	DeBERTa-large, GPT2-large
SPT	✔	✔	✔	✔	✔	✘	✔	✘	Mixed	RoBERTa-Large
SPT	✔	✘	✔	✘	✘	✘	✔	✘	Mixed	DeBERTa-large, GPT2-large
HyperPrompt	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base and Large
MPT	✔	✔	✔	✔	✔	✔	✔	✔	Dev	T5-base
BBT	✘	✘	✔	✘	✔	✘	✘	✔	Dev	RoBERTa-Large
BBT v2	✘	✘	✔	✘	✔	✘	✘	✔	Dev	RoBERTa-Large
ACCEPT	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
IAPT	✘	✔	✔	✘	✘	✘	✔	✘	Mixed	LLaMA-2-7B
Y-Tuning	✔	✔	✔	✔	✔	✔	✔	✘	Dev and Test	BART-Large
Y-Tuning	✔	✔	✔	✔	✔	✔	✔	✘	Dev	RoBERTa-Large, DeBERTa-XXL
SK-Tuning	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	RoBERTa-Base and Large
SK-Tuning	✘	✘	✔	✘	✔	✔	✔	✘	Mixed	Bloom-7B, LLaMA-2, Mistral-7B, Falcon, phi-2
InfoPrompt	✘	✘	✔	✘	✔	✔	✔	✘	Mixed	RoBERTa-Large
SUPMER	✘	✔	✔	✔	✔	✘	✔	✘	Mixed	T5-base, Flan-T5-XL
FPT	✔	✘	✘	✔	✘	✘	✘	✘	Dev	T5-Large and XL
PPT	✘	✘	✔	✘	✘	✘	✔	✘	Dev	T5-XXL
SPoT	✔	✔	✔	✔	✔	✔	✔	✔	Dev	T5-base
TPT	✔	✔	✔	✔	✔	✘	✘	✘	Dev	RoBERTa-Large, T5-XXL
DePT	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
Inducer-Tuning	✔	✘	✔	✘	✘	✘	✘	✘	Mixed	RoBERTa-base
Ladder-Side Tuning	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base, Large and 3B
SLaSh	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-base and Large
PASTA	✔	✔	✔	✔	✔	✔	✔	✔	Test	BERT-Large
PASTA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-Large
DimA	✔	✔	✔	✔	✔	✔	✔	✘	Test	RoBERTa-base and Large
RED	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	RoBERTa-base and Large, T5-base
Count	40	37	45	39	45	32	44	31		

Table 3.3: Additive PEFT approaches (on row) trained at least on one of GLUE datasets (on columns).

PEFT	MNLI	QNLI	SST-2	QQP	MRPC	CoLa	RTE	STS-B	Evaluated on	PLMs
Selective										
BitFit	✔	✔	✔	✔	✔	✔	✔	✔	Test	BERT-Large
BitFit	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base and large, RoBERTa-base
U-BitFit	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-Large
S-BitFit	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-Large
PaFi	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-Large
FishMask	✔	✔	✔	✔	✔	✔	✔	✔	Test	BERT-Large
Child-tuning	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-Large, RoBERTa-Large, XLNet-large, ELECTRA-large
FAR	✔	✔	✔	✔	✘	✔	✘	✘	Dev	DistilBERT-base
Model Grafting	✔	✔	✔	✔	✘	✘	✔	✘	Mixed	RoBERTa-base
Random Masking	✘	✘	✔	✘	✘	✘	✔	✘	Test	OPT-125M, -1.3B and 13B
BayesTune	✘	✘	✘	✘	✔	✔	✔	✔	Dev	RoBERTa-base
Reparametrization										
LoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-base and large, DeBERTa-XXL
AdaLoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-v3-base
Vera	✘	✔	✔	✘	✔	✔	✔	✔	Dev	RoBERTa-base and large
SAID	✔	✘	✔	✔	✔	✘	✘	✘	Mixed	RoBERTa-base
KronA	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
DyLoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-base
SoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-v3
CapaBoost	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	RoBERTa-base
CapaBoost	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-v3-base
AutoLoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-base
LoRA-Dropout	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-v3-base
Salient-LoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-V3-base
LoRA+	✔	✔	✔	✔	✘	✘	✘	✘	Test	RoBERTa-base, GPT-2
MTL-LoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-base
ERAT-DLoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	LLaMA-2-7b, MPT-7b
FourierFT	✘	✔	✔	✘	✔	✔	✔	✔	Dev	RoBERTa-base and Large
CorDA	✘	✔	✔	✘	✔	✔	✔	✔	Mixed	RoBERTa-base
ALoRA	✘	✔	✔	✘	✘	✘	✘	✘	Mixed	LLaMA-2 7B
Deep LoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base
FouRA	✔	✔	✔	✘	✔	✔	✘	✔	Dev	RoBERTa-base
PISSA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-v3-base
Spectral	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-v3-base
SURM	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base
SVFT	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	DeBERTa-v3-base
RLoRA	✘	✔	✔	✘	✔	✔	✔	✔	Dev	RoBERTa-base and Large
LaMBDA	✘	✔	✔	✘	✔	✔	✔	✔	Mixed	DeBERTa-v3-base
AFLoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-v3-base
MELoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-base
LoRA-GA	✔	✔	✔	✘	✔	✔	✘	✘	Dev	T5-base
LoRASC	✘	✘	✔	✘	✘	✘	✔	✘	Mixed	LLaMA2-7B
RoAD	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-base and Large
Kernel-mix	✔	✘	✔	✘	✘	✘	✘	✘	Dev	RoBERTa-base
HiFi	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-Large
SoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-v3
GloC	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-v3-base
DiffPruning	✔	✔	✔	✔	✔	✔	✔	✔	Test	BERT-base
MuscleLoRA	✘	✘	✔	✘	✘	✘	✘	✘	Mixed	BERT-base and Large, RoBERTa-base and Large, GPT-2-XL and LLaMA-2-7B
Count	43	45	51	41	43	40	47	41		

Table 3.4: Selective and Reparametrized PEFT approaches (on row) trained at least on one of GLUE datasets (on columns).

PEFT	MNLI	QNLI	SST-2	QQP	MRPC	CoLa	RTE	STS-B	Evaluated on	PLMs
Hybrid										
Sparse Adapter	✘	✘	✘	✘	✔	✔	✔	✔	Dev	BERT-base and RoBERTa-base
Sparse Adapter	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-base
PHM-Adapter	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
Compacter(++)	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
AdaMix	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base, RoBERTa-Large
UNIPELT	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base
Compacter	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
MoLE	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base,RoBERTa-base, DeBERTa-base,
S4	✔	✔	✔	✔	✔	✔	✔	✔	Dev	T5-base and 3b, Roberta-base and large
MaM Adapter	✔	✘	✔	✘	✘	✘	✘	✘	Dev	Roberta-base
LoRETTA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-base and RoBERTa-base
AutoPEFT	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base, T5-base
AutoPEFT	✘	✔	✔	✘	✔	✔	✔	✔	Dev	RoBERTa-Large
S^3PET	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-large
Pro-PETL	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	RoBERTa-base, T5-base
PEDRO	✘	✔	✔	✘	✘	✘	✔	✘	Test	LLaMA-2-7b
SAM	✘	✘	✘	✘	✔	✔	✔	✔	Mixed	RoBERTa-base
MOLE	✘	✔	✘	✘	✘	✘	✘	✘	Mixed	Flan-T5
MAdaKron	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base and Large, RoBERTa-base and Large, DeBERTa-v3
ComPEFT	✔	✔	✔	✔	✔	✔	✔	✔	Both	BERT-base and large, RoBERTa-base and large, T5-base and large, T0-3b
Memory-Tuning	✘	✔	✔	✘	✔	✘	✔	✘	Mixed	BERT-base and large
DePT	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
LoPA	✔	✔	✔	✔	✔	✘	✔	✘	Dev	RoBERTa-Large
DSEE	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base
DSEE	✘	✔	✔	✘	✘	✔	✔	✘	Dev	RoBERTa-Large
PRILoRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	DeBERTa-v3-base
RoseLoRA	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	RoBERTa-Large
S-MaM	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-Large
U-MaM	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-Large
HiWi	✔	✘	✘	✘	✘	✘	✔	✘	Dev	RoBERTa-Large
S^3Delta	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-Large
BIPEFT	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-Large
MPOE(++)	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base and Large
Poly	✘	✔	✘	✘	✘	✔	✔	✘	Mixed	BabyAI
ATTEMPT	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base
TSRA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-base and Large
LoraRetriever	✔	✔	✔	✔	✔	✘	✔	✔	Mixed	LLaMA-2-7B
AlphaLoRA	✘	✘	✘	✘	✔	✔	✔	✘	Mixed	LLaMA-7B, LLaMA-2-7B and Mistral-7B
PEMT	✘	✘	✘	✘	✔	✔	✔	✔	Mixed	T5-base
MixDA	✔	✔	✔	✔	✔	✔	✔	✔	Dev	RoBERTa-Large
Count	30	34	35	30	31	25	36	25		

Table 3.5: PEFT approaches (on row) trained at least on one of GLUE datasets (on columns).

PEFT	BoolQ	WIC	WSC	RTE	CB	MultiRC	Copa	ReCoRD	Evaluated on	PLM
Additive										
Pfeiffer Adapter	✔	✘	✘	✘	✔	✘	✘	✘	Dev	BERT-base
CoDA	✔	✘	✘	✘	✘	✘	✘	✔	Dev	T5-base, large and xl
Tiny Attention Adapter	✘	✘	✘	✔	✔	✘	✘	✘	Dev	ALBERT-XXLarge-v2
AdapterFusion	✔	✘	✘	✔	✔	✘	✘	✘	Dev	BERT-base
PHA	✔	✔	✔	✘	✔	✘	✘	✘	Mixed	T5-base
Hyperformer	✔	✘	✘	✘	✔	✘	✘	✘	Mixed	T5-base
Inspirational Pointer	✔	✔	✔	✘	✔	✘	✘	✘	Mixed	T5-base
ScaLearn	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	RoBERTa-base and Large
Residual Prompt Tuning	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-base, T5-base and Large
P-Tuning v2	✔	✔	✔	✔	✔	✔	✔	✔	Dev	BERT-Large, RoBERTa-Large, GLM-xlarge and xxlarge
Hyper-Prefix	✘	✔	✔	✔	✔	✘	✔	✘	Mixed	T5-Large and XL
HyperPrompt	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	T5-base and Large
MPT	✔	✔	✔	✔	✔	✔	✔	✔	Dev	T5-base
ACCEPT	✔	✔	✔	✘	✔	✔	✘	✘	Mixed	T5-base
VIP	✔	✔	✔	✔	✔	✔	✔	✔	Dev	T5-base
IAPT	✔	✘	✘	✔	✘	✘	✔	✔	Mixed	LLaMA-2-7B
Y-Tuning	✔	✘	✘	✔	✔	✘	✘	✘	Dev	RoBERTa-Large
SUPMER	✘	✔	✘	✔	✔	✘	✘	✘	Mixed	T5-base, Flan-T5-XL
FPT	✘	✘	✘	✘	✘	✘	✘	✔	Mixed	T5-base, Flan-T5-XL
PPT	✔	✘	✘	✔	✔	✘	✘	✘	Dev	T5-XXL
Selective										
Random Masking	✔	✔	✔	✔	✔	✔	✔	✔	Test	OPT-125M, -1.3B and -13B
BayesTune	✘	✘	✔	✔	✔	✘	✔	✘	Dev	RoBERTa-base
SHiRA	✔	✘	✘	✘	✘	✘	✘	✘	Mixed	LLaMA-7B and LLaMA-2-7B
PaFi	✘	✔	✔	✔	✘	✘	✔	✘	Dev	RoBERTa-Large
Reparametrized										
SVFT	✔	✘	✘	✘	✘	✘	✘	✘	Mixed	Gemma-7B
LoRETTA (7B)	✔	✘	✔	✘	✔	✘	✔	✔	Mixed	LLaMA2-7B
LoRETTA (13B)	✘	✘	✘	✘	✘	✘	✔	✔	Dev	LLaMA2-13B
DoRA	✔	✘	✘	✘	✘	✘	✘	✘	Test	LLaMA-7b and 13b, LLaMA2-7b, LLaMA3-8b
LaMBDA	✔	✘	✘	✘	✘	✘	✘	✘	Mixed	LLaMA-2-7B
HyperLoRA	✘	✔	✔	✔	✔	✘	✔	✘	Dev	FLAN-T5
HyperLoRA	✘	✔	✔	✔	✔	✘	✔	✘	Mixed	T5-Large and XL
QuanTA	✔	✘	✘	✘	✘	✘	✘	✘	Dev	LLaMA-7B and 13B, LLaMA-2-7B and 13b, LLaMA-3-8B
Count	15	11	9	13	13	6	9	6		

Table 3.6: Additive, Selective and Reparametrized PEFT methods (on rows) trained on at least one SuperGLUE datasets (on columns).

PEFT	BoolQ	WIC	WSC	RTE	CB	MultiRC	Copa	ReCoRD	Evaluated on	PLM
									Hybrid	
Compacter	✔	✔	✘	✔	✔	✔	✔	✔	Mixed	T5-base
Laplace-LoRA	✔	✘	✘	✘	✘	✘	✘	✘	Mixed	LLaMA-2-7b
MoSLoRA	✔	✘	✘	✘	✘	✘	✘	✘	Dev	LLaMA3-8b
LoRASC	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	LLaMA-2-7B
MixLoRA	✔	✘	✘	✘	✘	✘	✘	✘	Dev	LLaMA-2-7b and 13b, LLaMA-3-8b, Gemma-2B
MixDoRA	✔	✘	✘	✘	✘	✘	✘	✘	Dev	LLaMA-2-7b and 13b, LLaMA-3-8b, Gemma-2B
AutoPeft	✔	✔	✘	✔	✘	✘	✔	✘	Mixed	BERT-base
S^3PET	✔	✔	✘	✔	✔	✔	✔	✔	Mixed	T5-large
MTL-LoRA	✔	✘	✘	✘	✘	✘	✘	✘	Test	LLaMA-2-7b
SAM	✘	✘	✔	✔	✔	✘	✔	✘	Mixed	RoBERTa-base
PEDRO	✔	✘	✘	✔	✘	✘	✔	✔	Test	LLaMA2-7b and 13b, Gemma-2b
MultiLoRA	✔	✔	✘	✔	✘	✔	✘	✘	Mixed	LLaMA-7b, 13b, 30b and 65b
Memory-Tuning	✘	✘	✘	✔	✔	✘	✘	✘	Mixed	BERT-base and Large
PHM-Adapter	✔	✔	✘	✔	✔	✔	✔	✔	Mixed	T5-base
Compacter(++)	✔	✔	✘	✔	✔	✔	✔	✔	Mixed	T5-base
Shears	✔	✘	✘	✘	✘	✘	✘	✘	Mixed	LLaMA-7B
RoseLoRA	✔	✘	✘	✘	✘	✘	✘	✘	Mixed	LLaMA-7B
Long Exposure	✘	✘	✘	✔	✘	✘	✔	✘	Mixed	OPT
HiWi	✘	✔	✔	✔	✘	✘	✔	✘	Dev	RoBERTa-Large
S^3Delta	✔	✔	✘	✔	✔	✔	✔	✔	Dev	RoBERTa-Large
BIPEFT	✔	✔	✘	✔	✔	✔	✔	✔	Dev	RoBERTa-Large
C-Poly	✔	✔	✘	✔	✔	✔	✔	✘	Mixed	FLAN-T5-Large and GLM-10B
Poly	✘	✘	✘	✔	✘	✘	✘	✘	Mixed	BabyAI
MoV	✘	✔	✔	✔	✔	✘	✔	✘	Dev	T5-3b
ATTEMPT	✔	✔	✔	✔	✔	✔	✘	✘	Dev	T5-3b
MiLoRA	✔	✘	✘	✘	✘	✘	✘	✘	Dev	LLaMA-2-7B
LoraRetriever	✔	✔	✔	✔	✔	✔	✔	✔	Mixed	LLaMA-2-7B
FanLoRA	✔	✘	✘	✘	✘	✘	✔	✘	Dev	LLM-Assist-7B
PEMT	✘	✔	✔	✔	✔	✔	✔	✘	Mixed	T5-base
MoLoRA	✘	✔	✔	✔	✔	✘	✔	✘	Dev	T5-3b
Count	25	19	15	19	18	12	18	8		

Table 3.7: Hybrid PEFT approaches (on rows) on SuperGLUE on at least one SuperGLUE datasets (on columns).

PEFT	Squad v1	Squad v2	E2E	DART	WebNLG	WinoGrande	HellaSwag	XSum	CoNLL03	CoNLL04	OntoNotes	ARC-e and c	PIQA	SIQA	OBQA	WMT16 en-ro	IWSLT	GSM-8K
Additive																		
Parallel Adapter	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘
CIAT	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔	
CODA	✔	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘
Pfeiffer	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
Known-Adapter	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✔	✘	✘	✘	✘	✘	✘	✘
KnowLA	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘
HyperEncoder	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
Hyper-Prefix	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
PRO-CS	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘
AdapterFusion	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘
Prefix-Tuning	✘	✘	✔	✔	✔	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
P-tuning v2	✔	✔	✘	✘	✘	✘	✘	✘	✔	✔	✔	✘	✘	✘	✘	✘	✘	✘
APT	✘	✘	✘	✘	✘	✘	✘	✘	✔	✔	✔	✘	✘	✘	✘	✘	✘	✘
TPT	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
FPT	✘	✔	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
MPT	✘	✘	✔	✘	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
DEPT	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
Inducer-Tuning	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘
$(IA)^3$	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
SLaSh	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘
IPA	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
VIP	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
IAPT	✔	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
PASTA	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘
SK-Tuning	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘
DimA	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
RED	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
Count	7	1	4	1	4	6	5	6	8	2	3	0	0	2	0	3	1	1

Table 3.8: Additive PEFT approaches (on row) trained at least on one of the most popular datasets (on columns) different from GLUE and SuperGLUE.

PEFT	Squad v1	Squad v2	E2E	DART	WebNLG	WinoGrande	HellaSwag	XSum	CoNLL03	CoNLL04	OntoNotes	ARC-e and c	PIQA	SIQA	OBQA	WMT16 en-ro	IWSLT	GSM-8K
Selective																		
PaFi	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘
FISH-DIP	✘	✘	✘	✘	✘	✘	✘	✘	✔	✔	✔	✘	✘	✘	✘	✘	✘	✘
FAR	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
Xattn Tuning	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘
Random Masking	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
SHiRA	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✔	✔	✔	✔	✘	✘	✘
Reparameterized																		
LoRA	✘	✘	✔	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
VeRA	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
DoRA	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✔	✔	✔	✔	✘	✘	✘
Hyper-LoRA	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
DyLoRA	✘	✘	✔	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
AdaLoRA	✔	✔	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
CapaBoost	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
AutoLoRA	✘	✘	✔	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
Laplace-LoRA	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✔	✘	✘	✔	✘	✘	✘
Lora-Dropout	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
MoSLoRA	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✔	✔	✔	✔	✘	✘	✘
MOLE	✘	✘	✔	✔	✔	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✔	✘	✘
MixLoRA	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✔	✔	✔	✔	✘	✘	✘
ERAT-DLoRA	✘	✘	✔	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
FourierFT	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
Tied-LoRA	✔	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
ALoRA	✔	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
SVFT	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✔	✔	✔	✔	✘	✘	✔
VB-LoRA	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
LaMBDA	✘	✘	✘	✘	✘	✔	✔	✔	✘	✘	✘	✔	✔	✔	✔	✘	✘	✔
LoRETTA	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
RLoRA	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
AFLoRA	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
HyperLoRA	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
FLoRA	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
LoRA-GA	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
LoRA-GA	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
RoAD	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✔	✔	✔	✔	✘	✘	✔
QuanTA	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✔	✔	✔	✔	✘	✘	✔
Kernel-mix	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
GloC	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✘	✔	✔	✔	✘	✘	✘
SalientLoRA	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
Count	8	4	10	4	6	11	10	4	1	1	1	10	9	9	10	3	0	8

Table 3.9: Selective and Reparametrized PEFT approaches (on row) trained at least on one of the most popular datasets (on columns) different from GLUE and SuperGLUE.

PEFT	Squad v1	Squad v2	E2E	DART	WebNLG	WinoGrande	HellaSwag	XSum	CoNLL03	CoNLL04	OntoNotes	ARC-e and c	PIQA	SIQA	OBQA	WMT16 en-ro	IWSLT	GSM-8K
Hybrid																		
S^4	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘
MTL-LoRA	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✔	✔	✔	✔	✘	✘	✘
PEDRO	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
Memory Tuning	✘	✘	✘	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘
Sparse Adapter	✔	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
HLPT	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
RoSA	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
DSEE	✘	✘	✔	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
LoRAPrune	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✔	✔	✘	✔	✘	✘	✘
Shears	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✔	✔	✔	✔	✘	✘	✔
RoseLoRA	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✔	✔	✔	✔	✘	✘	✔
Long Exposure	✘	✘	✔	✘	✘	✔	✔	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘
HiWi	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘
MEFT	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
ATTEMPT	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
MiLoRA	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔	✔	✘	✔	✘	✘	✔
MoDULA	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
LoraRetriever	✔	✔	✔	✔	✔	✔	✘	✘	✘	✘	✘	✔	✔	✘	✔	✔	✘	✘
ESFT	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
AlphaLoRA	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
FanLoRA	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
PEMT	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
HydraLoRA	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✔
AdaMix	✘	✘	✔	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
LT-SFT	✘	✘	✘	✘	✘	✘	✘	✘	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘
MoV	✘	✘	✘	✘	✘	✔	✔	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘	✘
Count	5	1	3	3	3	10	8	2	2	1	0	6	7	3	7	2	1	11

Table 3.10: Hybrid PEFT approaches (on row) trained at least on one of the most popular datasets (on columns) different from GLUE and SuperGLUE.

Method	**Trainable Params (%)**	**Extra Params**	**Training Cost**	**Inference Latency**
Full FT	100%	None	Very High	None
Selective	<0.1%	None	Very Low	None
Reparametrized	0.1-0.5%	Low-rank	Low	None
Adapters	0.5-4%	Bottleneck	Moderate	Moderate
Soft Prompts	<0.1-1%	Embedding dimensions	Moderate	Moderate
Mixture-of-Experts	1-5%	Expert modules	Moderate–High	High

Table 3.11: Typical efficiency characteristics of major PEFT families. Values represent commonly reported ranges in the literature.

Part II

Adapter-based Models for Efficient Natural Language Processing and Information Retrieval

Chapter 4

Introduction

Traditional full fine-tuning of Large Language Models (LLMs) for downstream tasks or domains is computationally expensive, especially when models must be adapted across a wide range of tasks and domains. Adapter-based approaches have emerged as a promising alternative, enabling efficient transfer learning by inserting small trainable modules into a frozen backbone model. This paradigm significantly reduces the number of trainable parameters while maintaining competitive performance, making it particularly suitable for multi-task and multi-domain scenarios. In this part of the thesis, we focus on adapter-based efficient methods for NLP and IR, with an emphasis on improving both parameter efficiency and adaptability. We present two complementary works that address efficiency from different perspectives: task-level adaptation and domain-level specialization.

Chapter 5 (MAdaKron: a Mixture-of-AdaKron Adapters) introduces AdaKron and its extension MAdaKron, designed to improve the efficiency and flexibility of adapter-based fine-tuning for LLMs across a variety of NLP tasks, including text classification, question answering, and natural language generation. While adapter architectures typically rely on a single bottleneck transformation applied uniformly across inputs, AdaKron proposes a refined structure that enhances representational capacity while reducing the number of trainable parameters. MAdaKron further extends this approach by incorporating a Mixture of Experts mechanism, enabling dynamic routing between multiple specialized adapter components. This allows the model to activate different experts depending on the input characteristics, improving task adaptability without introducing further parameters. Experimental results show that both AdaKron and MAdaKron achieve strong performance across diverse NLP benchmarks while significantly reducing training costs compared to full fine-tuning and state-of-the-art parameter efficient approaches.

Chapter 6 (DRAMA: Domain Retrieval using Adaptive Module Allocation) shifts the focus from task-level adaptation to domain-level adaptation in information retrieval. In particular, we investigate multi-domain IR scenarios, where a single system must handle heterogeneous domains such as computer science,

physics, psychology and sci-fi. We leverage lightweight adapters combined with routing mechanisms to select domain-specific model efficiently. A key aspect of this approach is the use of a routing strategy that activates only one expert (i.e., one adapter module) per input, depending on the inferred domain. This hard routing mechanism ensures scalability, as the model does not need to jointly process all domain experts, while still allowing specialization for each domain. As a result, the system can maintain a shared backbone while dynamically adapting its behaviour to different retrieval contexts in a parameter-efficient manner. This approach is particularly well suited for large-scale IR systems, where deploying separate models for each domain would be impractical. By combining adapter-based efficiency with domain-aware routing, the proposed framework achieves a strong trade-off between performance, scalability, and memory usage.

This part is organized as follows: Chapter 5 presents AdaKron and MAdaKron, introducing a novel adapter architecture and a mixture of experts extension for efficient task-level adaptation of LLMs across multiple NLP tasks. Chapter 6 focuses on multi-domain information retrieval, presenting DRAMA, a routing-based adapter framework where domain-specific experts are selectively activated to enable efficient and scalable domain-adaptation across heterogeneous IR domains.

Chapter 5

MAdaKron: a Mixture-of-AdaKron Adapters

In this Chapter, we focus on *Adapters* [364], a widely adopted family of PEFT techniques that has shown strong effectiveness across a broad range of tasks, including Natural Language Understanding and Generation [475], question answering [355], and information retrieval [225]. As outlined in Chapter 3, an adapter consists of a down-projection layer that maps the hidden representation into a lower-dimensional space, followed by a non-linear activation, and an up-projection that restores the representation to the original hidden dimension [183].

The performance of adapter-based methods is highly dependent on the selected bottleneck (intermediate) dimension. In general, increasing this dimension tends to yield better downstream task results, although it comes at the cost of a higher number of trainable parameters [534]. Based on these findings, we propose AdaKron, which aims to reduce the number of trainable parameters in an adapter layer while maintaining a high intermediate dimension and thus avoiding performance degradation. AdaKron incorporates the Kronecker product [118] into an adapter layer. In contrast to prior PEFT methods that rely on Kronecker factorisation [217, 207], which typically decompose weight matrices into sets of smaller low-rank components, AdaKron instead applies the Kronecker product directly to the outputs of two feed-forward networks that together form the adapter's down-projection. The corresponding up-projection is maintained as a standard single feed-forward layer. The Kronecker product expands the representation by scaling one vector with each element of the other, producing a higher-dimensional representation whose dimensionality equals the product of the input vector sizes.

We interpret the Kronecker product as being conceptually related to a Mixture-of-Experts (MoE) mechanism [417]. In particular, one of the two vectors can be interpreted as a gating signal that modulates the other vector, similarly to how an MoE gate assigns weights to expert outputs. However, in contrast to standard MoE architectures, which aggregate experts through a weighted sum, the

Kronecker product generates a *weighted concatenation* of the scaled expert vector, explicitly preserving all individual components. This design remains highly parameter-efficient, since it increases representational capacity without requiring the definition and training of multiple independent expert modules.

To show that AdaKron can be seamlessly incorporated into state-of-the-art PEFT architectures, we propose *MAdaKron*, an extension that integrates AdaKron within an explicit Mixture-of-Experts (MoE) framework, thereby enhancing both flexibility and overall performance. MAdaKron replaces the heuristic interpretation of AdaKron's MoE behaviour with a fully defined architectural formulation, in which the weighting vector used in AdaKron is produced via a Mixture-of-Experts routing mechanism. This design allows the model to adaptively regulate the Kronecker interaction conditioned on the input, enabling the adapter to represent multiple adaptation patterns within a single layer. Importantly, the final output retains the structure of a weighted concatenation as in the original AdaKron formulation; however, the weighting coefficients are now input-dependent. This introduces greater representational capacity while maintaining the parameter efficiency of AdaKron. We assess the performance of AdaKron and MAdaKron across eighteen publicly available datasets spanning Natural Language Understanding and Natural Language Generation tasks, employing eight distinct Pre-Trained Language Models (PLMs) in our experiments. Our evaluation shows that both approaches reduce the number of trainable parameters by approximately one third compared to original adapters [353] and achieve consistently stronger performance while using the same intermediate dimension.

The main contributions of this Chapter can be summarised as follows:

- We propose AdaKron, a novel PEFT method that introduces the Kronecker product within Adapter modules.
- We propose MAdaKron, a new architecture that integrates AdaKron into a Mixture-of-Experts framework, substantially improving its effectiveness;
- We perform an extensive empirical evaluation of AdaKron and MAdaKron on eighteen benchmark datasets covering both natural language understanding and generation tasks. The results show that the proposed methods achieve performance that is competitive with, or surpasses, existing state-of-the-art PEFT techniques, while requiring the training of fewer than 0.55% of the original model parameters.
- We perform an extensive ablation study to analyse the impact of the design choices behind AdaKron and MAdaKron.

The remainder of this Chapter is organized as follows. Section 5.1 introduces the Kronecker product and presents the proposed AdaKron and MAdaKron architectures. Section 5.2 describes the experimental setup, including datasets, baselines,

evaluation metrics, and training configuration. Section 5.3 reports the main experimental results, while Section 5.3.1 presents an ablation study analysing the contribution of each architectural component. Finally, Section 5.4 concludes this Chapter.

5.1 Methodology

This section introduces the Kronecker product and describes in detail the proposed PEFT approaches, namely AdaKron and MAdaKron.

Background Let $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{p \times q}$ be two matrices. The Kronecker product, denoted by $A \otimes B$, is defined as a block matrix in $\mathbb{R}^{(m \cdot p) \times (n \cdot q)}$, obtained by multiplying each entry of A by the entire matrix B:

$$A \otimes B := \begin{bmatrix} a_{11}B & a_{12}B & \dots & a_{1n}B \\ a_{21}B & a_{22}B & \dots & a_{2n}B \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}B & a_{m2}B & \dots & a_{mn}B \end{bmatrix} \in \mathbb{R}^{m \cdot p \times n \cdot q}. \tag{5.1}$$

The resulting matrix can be expressed element-wise as:

$$A \otimes B := \begin{bmatrix} a_{11}b_{11} & \dots & a_{11}b_{1q} & \dots & a_{1n}b_{11} & \dots & a_{1n}b_{1q} \\ a_{11}b_{21} & \dots & a_{11}b_{2q} & \dots & a_{1n}b_{21} & \dots & a_{1n}b_{2q} \\ \vdots & \ddots & \vdots & \dots & \vdots & \vdots & \\ a_{11}b_{p1} & \dots & a_{11}b_{pq} & \dots & a_{1n}b_{p1} & \dots & a_{1n}b_{pq} \\ \vdots & \ddots & \vdots & \dots & \vdots & \vdots & \\ a_{m1}b_{11} & \dots & a_{m1}b_{1q} & \dots & a_{mn}b_{11} & \dots & a_{mn}b_{1q} \\ \vdots & \ddots & \vdots & \dots & \vdots & \vdots & \\ a_{m1}b_{p1} & \dots & a_{m1}b_{pq} & \dots & a_{mn}b_{p1} & \dots & a_{mn}b_{pq} \end{bmatrix}.$$

Given two vectors $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, their Kronecker product yields a resulting vector in $\mathbb{R}^{n \cdot m}$ as follows:

$$y \otimes x = vec(xy^T) = \left[y_1 x, \dots, y_m x\right], \tag{5.2}$$

which can be interpreted as a weighted concatenation of multiple copies of x, where each copy is scaled by the corresponding element of y. For example, given $x = [4,5,6] \in \mathbb{R}^3$ and $y = [1,4] \in \mathbb{R}^2$, we obtain $y \otimes x = [1 \cdot [4,5,6], 4 \cdot [4,5,6]] = [4,5,6,16,20,24] \in \mathbb{R}^6$. The resulting vector has dimension 6, which corresponds to the product of the dimensions of x and y.

Problem formulation We consider the problem of efficiently adapting Large Language Models to downstream tasks. Let $D = \{(x^i, y^i)\}_{i=1}^{P}$ be a labelled training dataset composed of P input–output pairs. Given a Pre-trained Language Model $f_\theta(\cdot)$ with parameters θ, our goal is to adapt f_θ so that it achieves strong performance on the target dataset D, despite the fact that the model has not been explicitly pre-trained for the corresponding task.

5.1.1 AdaKron

AdaKron integrates the Kronecker product within an adapter module. Following previous works Pfeiffer et al. [353], we place the adapter module immediately after the Feed-Forward Network (FFN) sub-layer within each transformer block.

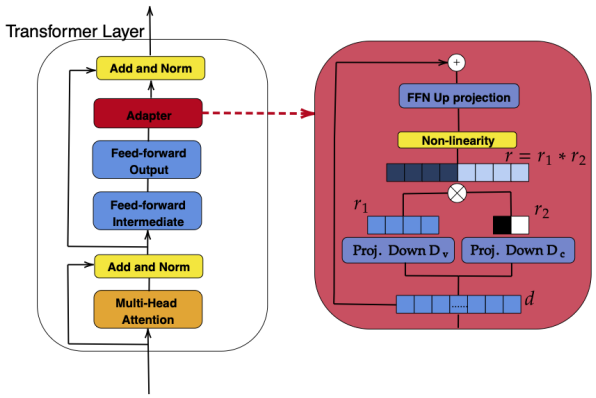

Figure 5.1: Integration of AdaKron in a transformer layer. **Left**: An AdaKron adapter is inserted after the FFN. **Right**: AdaKron consists of two parallel down-projections (D_v, D_c) producing vectors of size r_1 and r_2 from an input of dimension d. Their Kronecker product yields an r-dimensional representation, which is then mapped back via an FFN up-projection.

Figure 5.1 shows the proposed AdaKron architecture: Unlike standard Adapters, which rely on a single down-projection matrix, AdaKron employs two independent down-projection layers, denoted as D_v and D_c, followed by a single up-projection layer. This design is motivated by previous findings [475], suggesting that modifying the up-projection can negatively affect Adapter performance.

Formally, given an input vector $x \in \mathbb{R}^d$, the two down-projections are defined as:

$$y_v = D_v(x) = \mathbf{W}_v x + b_v, \qquad y_c = D_c(x) = \mathbf{W}_c x + b_c,$$

where $\mathbf{W}_v \in \mathbb{R}^{r_1 \times d}$ and $\mathbf{W}_c \in \mathbb{R}^{r_2 \times d}$, with biases $b_v \in \mathbb{R}^{r_1}$ and $b_c \in \mathbb{R}^{r_2}$. The outputs are $y_v \in \mathbb{R}^{r_1}$ and $y_c \in \mathbb{R}^{r_2}$, where $r_1, r_2 \ll d$. The two down-projection outputs are then combined through the Kronecker product:

$$g(y_c, y_v) \coloneqq GELU(y_c \otimes y_v) \in \mathbb{R}^{r_1 \cdot r_2},$$

where $GELU(\cdot)$ denotes the GELU activation function. The resulting representation has dimension $r = r_1 \cdot r_2$ and can be interpreted as a weighted concatenation of y_v, where the weights are provided by y_c. Finally, this r-dimensional vector is passed through a standard up-projection layer to restore the original hidden size.

5.1.2 MAdaKron

As shown in Section 5.1.1, AdaKron can be interpreted as a structured gating mechanism: one vector scales the other before the representation is expanded through the Kronecker product. This behaviour is similar to the Mixture-of-Experts architectures [417], where a gating function controls the contribution of different expert components. Motivated by this observation, we introduce *MAdaKron*, an extension that explicitly incorporates an MoE mechanism into the AdaKron framework. In this design, the weighting vector is produced by an MoE layer, allowing the selection of experts to depend on the input. This enables the model to capture multiple specialised adaptation patterns within a single adapter while preserving the parameter efficiency of AdaKron. Thus, MAdaKron can be interpreted as a more explicit realisation of AdaKron's implicit expert-like behaviour: whereas AdaKron approximates such a mechanism without explicitly defining experts, MAdaKron formalises it through dedicated expert routing while maintaining comparable efficiency. We define two MAdaKron variants, referred to as Partial and Full, which differ in how extensively MoE layers are used within the down-projection stage. As illustrated in Figure 5.2, the Partial MAdaKron variant replaces only the D_c down-projection with an MoE module responsible for generating the weights used in the weighted concatenation, while keeping D_v as a single feed-forward layer. In contrast, as shown in Figure 5.3, the Full MAdaKron variant implements both down-projection components, D_v and D_c, as MoE layers. In this configuration, the model selects one expert from each MoE module for every input, and the resulting outputs are combined through the Kronecker product. To select experts, MAdaKron adopts a stochastic routing strategy to determine which experts are activated for each input. This choice is motivated by recent works [541, 475] showing that a stochastic routing function can achieve on par with or even superior performance than learned gating mechanisms, while avoiding additional trainable parameters. Moreover, since the gating mechanism is not implemented as a separate neural network module, MAdaKron retains the same parameter budget as AdaKron while offering greater representational capacity. In contrast to recent methods such as AdaMix [475], where experts are explicitly parameterised, MAdaKron derives its expert-like behaviour implicitly from the Kronecker-based structure of AdaKron, thereby inducing specialised components in a parameter-efficient way. By coupling these implicitly formed experts with an explicit, input-conditioned routing strategy, MAdaKron maintains the compactness of the original formulation while incorporating the advantages of Mixture-of-Experts models, leading to a more expressive

adaptation mechanism.

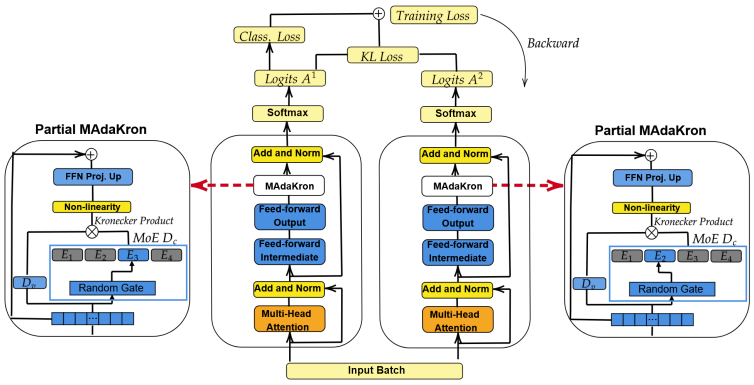

Figure 5.2: Integration of Partial MAdaKron in a transformer layer. Only the D_c down-projection is replaced by a Mixture-of-Experts module (here with four experts). During training, each batch is processed twice with different expert routing, optimising a classification loss and a symmetric KL-based consistency loss.

Figure 5.2 illustrates the training procedure of MAdaKron and how the stochastic routing function works. During each training step, the same batch is processed twice, sampling two different expert configurations. Let n denote the number of Transformer layers and $k \in \{1,2\}$ indicate the two forward passes. The expert configuration for pass k is denoted as $A^k = \{A_1, \ldots, A_n\}$. Given an input x, the model produces logits $z_{(.)}^{A^k}(x)$ for each pass. To improve training stability and ensure consistency between predictions obtained under different expert selections, we incorporate a consistency regularisation term based on the Kullback-Leibler divergence [235]. The resulting optimisation objective can be expressed as follows:

$$\mathcal{L} = -\left(\mathcal{L}_{Class.} + \frac{1}{2}\mathcal{L}_{KL}\right),$$

where $\mathcal{L}_{Class.}$ denotes the classification loss and $\mathcal{L}_{KL}$ is the symmetric KL divergence between the two forward passes. Specifically, the classification loss is defined as:

$$\mathcal{L}_{Class} = \sum_{c=1}^{C} \mathcal{I}(x,c) \log(\text{softmax}(z_c^{A^1}(x))),$$

where $\mathcal{I}(x,c) = 1$ if c is the correct class label for x, and 0 otherwise. The consistency term is:

$$\mathcal{L}_{KL} = KL(z_{(.)}^{A^1}(x) \,||\, z_{(.)}^{A^2}(x)) + KL(z_{(.)}^{A^2}(x) \,||\, z_{(.)}^{A^1}(x)).$$

Finally, during inference, expert weights are merged into a single feed-forward module following the approach proposed in [475], enabling MAdaKron to preserve inference efficiency.

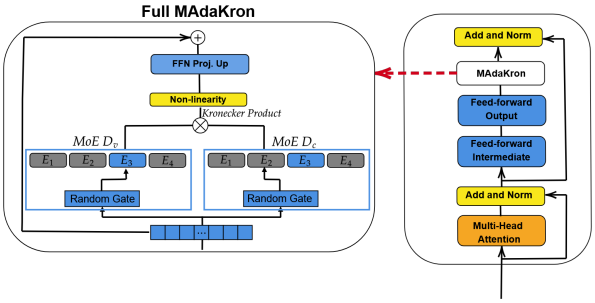

Figure 5.3: Integration of Full MAdaKron within a Transformer layer. In this configuration, both down-projection components of the AdaKron adapter, D_v and D_c, are implemented as MoE modules. For clarity, we illustrate the architecture using four experts per MoE layer.

5.1.3 Parameter Efficiency of AdaKron and MAdaKron

In this section, we provide a comparison of the parameter counts of MAdaKron and AdaKron against the standard adapter architecture [353]. Since MAdaKron and AdaKron share the same number of activated parameters, in this Chapter we refer to MAdaKron, but the same findings are generalizable to AdaKron.

Unlike the original Pfeiffer adapter approach [353], which employs a single down-projection layer with output dimension r, MAdaKron introduces a factorized bottleneck composed of two FFNs with output dimensions r_1 and r_2, respectively. Here, $r_2 \leq r$ acts as a reduction factor corresponding to the number of weighted repetitions applied to the r_1-dimensional representation. Consequently, the parameterization of MAdaKron is dominated by the matrices $W_v \in \mathbb{R}^{r_1 \times d}$ and $W_c \in \mathbb{R}^{r_2 \times d}$. Without loss of generality, assuming $r_1 > r_2$, we obtain $r_1 = r/r_2$, where $r_1 \in \mathbb{N}$, implying that r_2 must be a divisor of r. Given an input dimensionality d and a shared intermediate dimension r, the Pfeiffer Adapter down-projection layer (including bias terms) contains $rd + r$ parameters. In MAdaKron, the number of parameters of the down-projection layer is:

$$rd + r = d\left(\frac{r}{r_2} + r_2\right) + \frac{r}{r_2} + r_2 = (d+1)\left(\frac{r}{r_2} + r_2\right). \tag{5.3}$$

Including the up-projection layer, the total number of parameters for MAdaKron becomes:

$$(d+1)\left(\frac{r}{r_2} + r_2\right) + rd + d, \tag{5.4}$$

whereas the corresponding Pfeiffer Adapter contains $2rd + r + d$ parameters. The resulting parameter ratio between MAdaKron and the Pfeiffer Adapter is therefore given by

$$\mathcal{R} = \frac{(d+1)\left(\frac{r}{r_2} + r_2 + r\right)}{2rd + r + d}. \tag{5.5}$$

Under the assumption that $d \gg r \gg r_2$, the dominant terms in the numerator and denominator of Equation 5.5 are $d\left(r + \frac{r}{r_2} + r_2\right)$ and $2rd + d$, respectively. This setting aligns with standard practice in adapter-based approaches [183, 47] as well as widely used pre-trained language models such as BERT [102] and RoBERTa [279], where the hidden dimension is commonly set to $d = 768$ and adapter bottleneck dimensions are typically selected from the range $r \in \{16, 32, 48, 64\}$. Under this assumption, Equation 5.5 can be approximated as

$$\mathcal{R} \approx \frac{(r + \frac{r}{r_2} + r_2)d}{(2r+1)d} = \frac{r + \frac{r}{r_2} + r_2}{2r+1}. \tag{5.6}$$

As subsequently discussed in Section 5.3.1, the hyperparameter r_2 assumes its optimal value at $r_2 = 4$. Substituting this value into Equation 5.6 yields

$$\mathcal{R} = \frac{r + \frac{r}{4} + 4}{2r+1} = \frac{5r+16}{8r+4}. \tag{5.7}$$

For the commonly adopted bottleneck dimensions $r \in 16, 32, 48, 64$, we observe that $\mathcal{R} < 1$, showing that MAdaKron consistently involves a lower number of trainable parameters compared to the Pfeiffer Adapter. In detail, the ratio $\mathcal{R}$ varies approximately between 0.72 and 0.65, corresponding to a reduction of up to roughly 30% in the number of trainable parameters.

5.1.4 Motivations and Relations with other PEFT methods

Recently, the Kronecker product has emerged as a promising mathematical tool for enhancing parameter efficiency in neural networks, particularly in the context of PEFT methods, as it enables reductions in both the number of trainable parameters and computational cost, including FLOPs [113, 217]. Most existing approaches leverage Kronecker factorisation by decomposing large weight matrices into smaller components that are combined through the Kronecker product. For example, Compacter [217] employs this decomposition within adapter modules, whereas Krona [113] extends the idea to attention mechanisms. Related Kronecker-based compression techniques have also been applied to large-scale transformer models such as BERT [436] and GPT-2 [114]. Formally, given matrices $A \in \mathbb{R}^{m \times n}$ and $B \in \mathbb{R}^{p \times q}$, their Kronecker product $A \otimes B$ yields a matrix in $\mathbb{R}^{mp \times nq}$. This property allows a high-dimensional weight matrix $W \in \mathbb{R}^{mp \times nq}$ to be parametrized via the smaller matrices A and B, which together require substantially fewer parameters to be trained than W. Furthermore, from a theoretical perspective, the Kronecker product can also be interpreted as a matrix representation of the tensor product of linear maps. Specifically, let $S : V \to X$ and $T : W \to Y$ be linear transformations represented by matrices A and B, respectively. Then $A \otimes B$ corresponds to the induced map

$$S \otimes T : V \otimes W \to X \otimes Y$$

, preserving the structure of the underlying tensor spaces. An additional important property is the multiplicative behaviour of rank under the Kronecker product, namely

$$\text{rank}(A \otimes B) = \text{rank}(A) \times \text{rank}(B).$$

This property enables effective learning even under tight parameter constraints, without requiring an explicit reduction in matrix rank. In addition to improving representational efficiency, Kronecker-based structures can also be exploited to reduce computational cost. For instance, Edalati et al. [113] demonstrate that the full Kronecker matrix need not be explicitly formed during inference by leveraging the following identity:

$$(A \otimes B)x = \gamma\left(B\eta_{b_2 \times a_2}(x)A^\top\right),$$

where $x \in \mathbb{R}^d$, $\eta_{b_2 \times a_2}(y)$ denotes the reshaping of a vector $y \in \mathbb{R}^{b_2 \times a_2}$ into a matrix of size $b_2 \times a_2$, and $\gamma(Y)$ reshapes a matrix $Y \in \mathbb{R}^{a_2 \times b_2}$ into a vector by stacking its columns. This reformulation replaces a large-scale matrix multiplication with a combination of structured reshaping operations and smaller matrix multiplications, thereby reducing FLOPs and improving inference efficiency. More recently, Kronecker-based adaptation has been extended within more complex architectures. For example, Yu, Yang, and Yi [509] introduce MoKA (Mixture of Kronecker Product Adaptation), which integrates a MoE architecture within a Kronecker-based LoRA. In contrast, Sadeghi et al. [403] introduce MoKA (Mixture of Kronecker Adapters), which integrates a learnable gating mechanism to combine multiple Kronecker-based adapters through weighted aggregation. As both these approaches build on LoRA updates, they mainly focus on attention weights. Differently from techniques that apply Kronecker factorisation to full weight matrices or attention components, our method operates at the vector level. In particular, we apply the Kronecker product to the outputs of two parallel down-projection layers within the feed-forward network, combining intermediate representations rather than decomposing parameter matrices directly. This choice reduces the number of trainable parameters while enabling structured and expressive representations in the projected space. Moreover, as shown in Equation 5.2, the Kronecker product between vectors $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$ can be written as $x \otimes y = \text{vec}(xy^\top)$, where $xy^\top$ denotes the outer product and $\text{vec}(\cdot)$ transforms a matrix into a vector via column-wise stacking. Since both operands are vectors, the resulting outer product is a rank-one matrix, providing a compact but structured representation. The Kronecker product between vectors can also be interpreted as a form of weighted concatenation. Specifically, for $x = [x_1, x_2, \ldots, x_n]$ and $y = [y_1, y_2, \ldots, y_m]$, we have $y \otimes x = [y_1 x, \ldots, y_m x]$, which corresponds to concatenating scaled copies of x, where each copy is modulated by the corresponding element of y. This perspective highlights a similarity to Mixture-of-Experts gating, where x can be seen as encoding expert activations and y as providing the weighting scheme. However, unlike standard MoE architectures that rely on explicit routing networks and learned

aggregation of expert outputs, the Kronecker-based formulation achieves a comparable effect implicitly through a single parameter-free operation. As a result, it can be interpreted as a lightweight alternative to MoE systems, enabling efficient and effective representation learning without the overhead of explicit routing functions.

5.2 Experimental Setup

The experimental study presented in the following Section is designed to assess the effectiveness and efficiency of our proposed approaches, AdaKron and MAdaKron, across a wide range of downstream tasks and model architectures. In particular, the experimental evaluation is designed to address the following research questions:

- **RQ1**: How does AdaKron compare in terms of effectiveness and efficiency against full fine-tuning and current state-of-the-art PEFT methods?
- **RQ2**: Does incorporating a Mixture-of-Experts component into AdaKron, resulting in MAdaKron, consistently enhance performance?
- **RQ3**: How does MAdaKron compare with state-of-the-art PEFT approaches in terms of downstream task performance?
- **RQ4**: What is the impact of the main architectural and training design choices underlying AdaKron and MAdaKron?

To address the above research questions, we conduct a comparative evaluation covering both natural language understanding and generation tasks. The rest of this section provides an overview of the benchmark datasets employed in our experiments, the comparison baselines, the metrics used to assess both effectiveness and efficiency, as well as the implementation details.

Datasets and Baselines We evaluate AdaKron and MAdaKron on a diverse set of benchmarks. For natural language understanding, we rely on the GLUE benchmark [461], which includes eight tasks covering linguistic acceptability, sentiment classification, semantic similarity, paraphrase detection, and natural language inference. Specifically, GLUE includes CoLA [478], SST-2 [423], MRPC [108], STS-B [61], QQP [461], MNLI [482], QNLI [381], and RTE [138, 33]. Following common practice in the literature [183, 102], we exclude WNLI [243], as its construction issues prevent BERT-like models from outperforming the majority baseline.[1] In addition, we evaluate our approaches on SuperGLUE [462], a more challenging

[1]See (12) in gluebenchmark.com/faq.

benchmark that focuses on high-level reasoning. SuperGLUE includes BoolQ [89], RTE, WiC [358], WSC [243], and CB [100]. Since the official test sets for both GLUE and SuperGLUE are not publicly available, we report results on the development sets, consistently with prior works [186, 31].

To further extend the evaluation beyond sentence-level classification, we also include two Question Answering datasets, namely SQuAD v1.1 [382] and SQuAD v2.0 [380]. Finally, to assess the applicability of our approaches to language generation tasks, we consider three widely adopted benchmarks: E2E [338], WebNLG [135], and DART [324].

Evaluation Metrics Model effectiveness is evaluated using the standard metrics associated with each benchmark, following prior PEFT studies [186, 475, 518] and the results of our systematic analysis in Chapter 3. For GLUE, we report accuracy for most the tasks, excluded CoLA and STS-B that are evaluated using Matthews correlation and Pearson correlation, respectively. MRPC and QQP are evaluated using F1. All SuperGLUE tasks are evaluated using accuracy. For extractive Question Answering, we report both Exact Match (EM) and F_1, as standard in the SQuAD evaluation protocol [382]. For natural language generation tasks, we report multiple metrics, including BLEU, NIST, METEOR, ROUGE-L, CIDEr, and TER. All metrics are maximised, except TER, which is minimised. Furthermore, we conduct a one-sided statistical significance test with a threshold of $p < 0.05$, corresponding to a 95% confidence level, in order to assess whether the mean performance across the four runs of our method is significantly higher than that of the baselines. In addition to model effectiveness, we also evaluate the efficiency of each method. Following recent works [256], we measure the number of trainable parameters, reported in millions as *# Params (M)*. This quantity provides a direct indication of the memory footprint and optimisation cost of each approach, and enables a fair comparison between methods. In Section 5.3.1, we further analyse the efficiency of our approaches by reporting additional measures, including FLOPs, GPU memory usage, and energy consumption.

Baselines We compare AdaKron and MAdaKron against full fine-tuning as well as a broad range of state-of-the-art PEFT approaches. Experiments are conducted on multiple Pre-trained Language Models (PLMs), including encoder-only models and autoregressive decoders. In particular, we evaluate on:

- **BERT-base** [102], which has 12 layers, 768 as hidden dimension, 12 heads and 110M parameters.
- **BERT-Large** [102], which has 24 layers, 1024 as hidden dimension, 16 heads and 355M parameters.

- **RoBERTa-base** [279], which has 12 layers, 768 as hidden dimension, 12 heads and 110M parameters.
- **RoBERTa-Large** [279], which has 24 layers, 1024 as hidden dimension, 16 heads and 355M parameters.
- **DeBERTa-v3-base** [163], which has 12 layers, 768 as hidden dimension, 12 heads and 184M parameters.
- **DeBERTa-v2-XXL** [164], which has 48 layers, 1536 as hidden dimension, 24 heads and 1.5B parameters.
- **GPT-2 Medium** [375], which has 24 layers, 1024 as hidden dimension, 16 heads and 355M parameters.
- **GPT-2 Large** [375], which has 36 layers, 1280 as hidden dimension, 20 heads and 774M parameters.

This selection enables us to test our methods across different model sizes and architectures, ranging from 110M parameters to 1.5B parameters.

Moreover, we evaluate our methods against a range of recently proposed PEFT techniques, grouped into the following categories:

- **Additive methods**, including **Houlsby** [183] and the **Pfeiffer Adapter** [353], as well as **Prefix-Tuning** [253], **Hadamard Adapter** [80], **Diff-Prune** [148], and **(IA)^3** [270].
- **Selective approaches**, exemplified by **BitFit** [31].
- **Reparameterisation-based methods**, such as **LoRA** [186], **AdaLoRA** [518], **DyLoRA** [452], **Vera** [234], and **DiffoRA** [209].
- **Hybrid and MoE-inspired approaches**, including **AdaMix** [475], **UNIPELT** [305], **Compacter(++)** [217], and the **Mixture of Linguistic Experts (MoLE) Adapter** [249].

In cases where published results are not available, we re-implement the corresponding baselines using the *Adapters* library [364]. All reproduced results are explicitly indicated as *repr.* in the tables to distinguish them from values reported in the original works.

Implementation details All experiments are conducted using PyTorch and run on NVIDIA A100 GPUs. We rely on the models that are publicly available on Hugging Face *Transformers* library [484]. Hyperparameter choices follow the configurations proposed in related PEFT literature [475, 518]. We report the detailed

hyperparameter settings in Tables 5.1–5.4. Unless otherwise specified, MAdaKron uses four experts per layer for all natural language understanding tasks, while experiments with GPT-2-based models employ eight experts. All models are trained using AdaFactor [415], which avoids the need for learning rate tuning. The adapter bottleneck sizes are based on the ablation study presented in Section 5.3.1. Specifically, we set the intermediate dimension to 48 for BERT-based architectures, to 16 for RoBERTa and DeBERTa variants, and to 8 for GPT-2 models. Across all experiments, the down-projection dimension r_2 is fixed to 4. Each configuration is evaluated using four random seeds (0, 42, 1234, and 4321), and we report the average performance over these runs. In the result tables, the highest score is shown in **bold**, while the second-best result is underlined. * and † represent significant improvements of our approaches over the best and second best baselines, respectively. *Avg.* is the average performance across the selected datasets. For all reproduced baselines, we adopt the same hyperparameter settings as those used for AdaKron and MAdaKron to ensure a fair comparison under identical experimental conditions. In the case of LoRA and VeRA, we set the low-rank dimension to $r = 8$, following the recommendations of the original work [186]. For the Houlsby and Pfeiffer adapters, we use the same dimensional configurations as in our proposed method. For Compacter and Compacter++, we select the best-performing configuration reported in the original paper [217], corresponding to a value of 24. Regarding parameter counts, we report values taken directly from prior work when available; otherwise, we compute and report the counts based on our own re-implemented setups.

Dataset	Epochs	Batch Size	Warmup ratio	Weight Decay	Adapter Size
BERT-base and Large					
CoLA	100	16	0.06	0.1	48
SST2	40	64	0.06	0.1	48
MNLI	40	64	0.06	0.1	48
RTE	80	64	0.06	0.1	48
QQP	60	64	0.06	0.1	48
MRPC	100	16	0.06	0.1	48
QNLI	20	64	0.06	0.1	48
STS-B	80	32	0.06	0.1	48
RoBERTa-base and Large					
CoLA	80	64	0.6	0.1	16
SST2	20	64	0.6	0.1	16
MNLI	20	64	0.6	0.1	16
RTE	60	64	0.6	0.1	16
QQP	80	64	0.6	0.1	16
MRPC	60	64	0.6	0.1	16
QNLI	20	64	0.6	0.1	16
STS-B	80	64	0.6	0.1	16

Table 5.1: Hyperparameter configurations for BERT-based and RoBERTa-based models on GLUE datasets.

Dataset	Epochs	Batch Size	Warmup steps	Weight Decay	Adapter Size
DeBERTa-v3-base					
CoLA	50	32	800	0.5	16
SST2	25	32	6000	0.1	16
MNLI	20	32	8000	0.1	16
RTE	50	32	600	0.3	16
QQP	50	32	8000	0.1	16
MRPC	40	32	600	0.1	16
QNLI	20	32	2000	0.1	16
STS-B	50	32	800	0.1	16
SQuAD v1.1	10	16	1000	0.1	16
SQuAD v2.0	12	64	1000	0.1	16
DeBERTa-v2-XXL					
CoLA	10	32	800	0.0	8
RTE	11	32	600	0.01	8
MRPC	30	32	600	0.01	8
STS-B	10	32	800	0.1	8

Table 5.2: Hyperparameter configurations used for DeBERTa-based models.

Dataset	Epochs	Batch Size	Warmup ratio	Weight Decay	Adapter Size
BERT-base and Large					
BoolQ	100	16	0.06	0.1	48
RTE	40	64	0.06	0.1	48
WiC	40	64	0.06	0.1	48
WSC	80	64	0.06	0.1	48
CB	60	64	0.06	0.1	48
RoBERTa-base and Large					
BoolQ	100	16	0.06	0.1	48
RTE	40	64	0.06	0.1	48
WiC	40	64	0.06	0.1	48
WSC	80	64	0.06	0.1	48
CB	60	64	0.06	0.1	48

Table 5.3: Hyperparameter configurations for BERT-based, RoBERTa-based models on SuperGLUE.

Dataset	Epochs	Batch Size	Warmup Steps	Adapter Size
GPT-2-Medium and Large				
E2E	20	8	2000	8
WebNLG	25	8	2500	8
DART	20	8	2000	8

Table 5.4: Hyperparameter configurations for GPT-2-based models

5.3 Results and Discussions

This section reports the results of our comparative experimental evaluation. We begin by examining the performance of AdaKron and MAdaKron across a diverse

set of downstream benchmark tasks and PLMs, with the aim of evaluating both their effectiveness and parameter efficiency. The discussion is structured by task category, considering natural language understanding first, followed by question answering, and finally natural language generation. Moreover, Section 5.3.1 provides an extensive ablation study aimed at quantifying the impact of the main design choices introduced in AdaKron and MAdaKron.

Natural Language Understanding Tables 5.5-5.10 report results on the GLUE and SuperGLUE benchmarks. Experiments are conducted using BERT-base and BERT-large, as well as RoBERTa-base and RoBERTa-large.

Model	# Params (M)	MNLI	QNLI	SST2	QQP	MRPC	CoLA	RTE	STS-B	Avg.
Baselines										
Full Fine-Tuning [475]	110	83.2	90.0	91.6	87.4	90.9	62.1	66.4	89.8	82.7
UNIPELT [475]	1.4	83.9	90.5	91.5	85.5	90.2	58.6	73.7	88.9	83.5
Compacter (repr.)	1.3	81.6	89.2	87.9	85.1	88.7	58.7	59.8	86.3	79.7
Compacter++ (repr.)	1.3	80.0	89.8	91.6	83.7	89.5	57.8	60.5	86.2	79.9
$(IA)^3$ (repr.)	1.2	79.4	88.8	92.3	82.7	90.1	59.9	65.4	86.8	80.7
MoLE-Adapter [249]	1.2	84.3	93.0	92.7	87.8	90.4	61.5	70.4	88.7	83.6
UNIPELT (AP) [305]	1.1	83.4	90.8	91.9	86.7	90.3	61.2	71.8	88.9	83.1
AdaMix Adapter [475]	0.9	**84.7**	**91.5**	92.4	**87.6**	**92.4**	**62.9**	74.7	**89.9**	**84.5**
Houlsby Adapter [475]	0.9	83.1	90.6	91.9	86.8	89.9	61.5	71.8	88.6	83.0
Pfeiffer Adapter	0.9	83.3	91.1	92.0	87.5	90.7	60.3	67.6	89.6	82.7
Pfeiffer Adapter	0.6	83.2	90.9	92.1	87.4	90.1	60.5	69.1	89.4	82.8
LoRA [475]	0.3	82.5	89.9	91.5	86.0	90.0	60.5	71.5	85.7	82.2
Prefix-tuning [475]	0.2	81.2	90.4	90.9	83.3	91.3	55.4	76.9	87.2	82.1
BitFit [475]	0.1	81.4	90.2	92.1	84.0	90.4	58.8	72.3	89.2	82.3
Vera (repr.)	0.04	83.1	90.5	92.3	85.9	89.9	59.0	61.0	86.8	81.1
Hadamard-Adapter [80]	0.03	80.4	89.7	92.4	85.9	90.2	58.4	71.9	88.5	82.2
Our approaches										
AdaKron	0.6	$83.5_{0.18}$	$91.1_{0.45}$	$92.0_{0.36}$	$87.1_{0.14}$	$90.8_{1.24}$	$61.1_{0.35}$	$73.8_{1.05}$	$89.4_{0.24}$	83.6
Full MAdaKron	0.6	$83.9_{0.20}$	$91.3_{0.47}$	$\mathbf{92.8}^{*,\dagger}_{0.13}$	$87.4_{0.08}$	$91.5_{0.59}$	$62.3^{\dagger}_{0.35}$	$\mathbf{76.0}^{*,\dagger}_{0.44}$	$89.2_{0.13}$	84.3
Partial MAdaKron	0.6	$83.9_{0.05}$	$91.1_{0.52}$	$92.3_{0.08}$	$\mathbf{87.6}^{\dagger}_{0.03}$	$91.1_{0.41}$	$61.8_{0.90}$	$74.2_{0.75}$	$89.4_{0.42}$	83.9

Table 5.5: Main results on the GLUE validation set with BERT-base.

Model	# Params (M)	MNLI	QNLI	SST2	QQP	MRPC	CoLA	RTE	STS-B	Avg.
Baselines										
Full Fine-Tuning [31]	336	85.5	91.7	93.4	87.5	90.7	62.2	71.9	90.0	84.1
Houlsby Adapter (repr)	4.8	85.1	92.0	93.0	**88.7**	83.1	58.3	70.4	89.7	82.5
Pfeiffer Adapter (repr.)	2.4	85.2	91.7	93.5	87.9	90.4	64.3	74.2	89.7	84.6
UNIPELT (repr.)	3.2	**85.7**	91.6	**94.1**	85.7	91.1	**66.0**	75.2	88.8	84.7
Compacter (repr.)	2.3	84.3	91.4	93.8	85.5	91.1	60.6	68.7	88.0	82.9
$(IA)^3$ (repr.)	2.3	80.7	90.0	93.4	81.7	90.8	64.3	75.0	88.5	83.1
Compacter++ (repr.)	2.2	82.5	91.1	93.7	83.4	90.3	62.9	68.7	88.1	82.6
Diff-Prune [31]	1.7	86.4	93.4	94.2	86.6	91.3	63.5	71.5	89.5	84.5
AdaMix (repr.)	2.4	**85.7**	92.3	94.0	87.8	91.6	59.6	75.4	90.1	84.6
LoRA (repr.)	0.8	85.3	91.5	93.8	85.6	91.2	59.5	66.7	90.0	82.7
BitFit [31]	0.3	84.4	91.4	93.2	85.4	**91.7**	63.6	73.2	**90.3**	84.1
Hadamard Adapter [80]	0.1	83.6	91.1	93.1	87.2	91.3	62.7	73.1	90.0	84.0
Vera (repr.)	0.02	83.9	91.6	93.8	84.5	91.1	58.6	66.0	87.5	82.1
Our approaches										
AdaKron	1.6	$84.8_{0.22}$	$91.7_{0.17}$	$93.3_{0.29}$	$87.5_{0.15}$	$90.1_{0.64}$	$62.9_{0.48}$	$73.4_{0.96}$	$90.0_{0.14}$	84.2
Full MAdaKron	1.6	$84.7_{0.14}$	$\mathbf{92.8}^{*,\dagger}_{0.17}$	$93.6_{0.35}$	$86.4_{0.26}$	$90.9_{0.66}$	$64.8_{1.92}$	$75.3_{2.93}$	$89.7_{0.30}$	84.8
Partial MAdaKron	1.6	$85.4_{0.19}$	$92.4^{\dagger}_{0.06}$	$93.7_{0.33}$	$88.0_{0.13}$	$91.4_{0.30}$	$64.1_{0.79}$	$\mathbf{75.6}_{0.34}$	$90.2_{0.16}$	**85.1**

Table 5.6: Main results on the GLUE validation set with BERT-Large.

On the GLUE benchmark, both AdaKron and MAdaKron provide a strong

trade-off between effectiveness and efficiency. In particular, AdaKron consistently improves upon several adapter-based baselines that require substantially more trainable parameters, including UNIPELT, Compacter and Compacter++, $(IA)^3$, as well as both Houlsby and Pfeiffer adapters. These results show that the proposed Kronecker-based parameterisation enables competitive adaptation while requiring approximately one third fewer trainable parameters than original adapter methods. Furthermore, AdaKron outperforms lightweight approaches such as BitFit, Vera, and LoRA by at least one point on average. Building on these results, both Partial and Full MAdaKron further improve AdaKron by at least 0.7 points on average while maintaining the same parameter budget. The largest improvements are observed on low-resource benchmarks such as RTE and CoLA, suggesting that the proposed approach is particularly effective when training data are limited. In particular, Full MAdaKron achieves state-of-the-art performance on RTE, outperforming AdaMix by more than one point. Finally, although AdaMix marginally outperforms MAdaKron by 0.2 points on average, it requires training approximately one third more parameters than both MAdaKron variants. Table 5.6 shows that these performance trends remain stable when scaling to BERT-large. AdaKron remains competitive with full fine-tuning, while outperforming several baselines that train either more parameters, such as Pfeiffer and Houlsby adapters, or fewer parameters, such as LoRA. The largest improvements achieved by MAdaKron emerge on QNLI, where Full MAdaKron reaches 92.8, improving over AdaKron by more than one point and outperforming the Pfeiffer adapter baseline by 0.3 points. Moreover, Partial MAdaKron provides the strongest overall performance among the proposed variants, reaching 85.1 and improving over AdaKron by nearly one point without increasing the number of trainable parameters. These results suggest that the expert-based mechanism is particularly beneficial at larger model scales, where the additional representational capacity can be more effectively exploited through adaptive routing.

Model	# Params (M)	MNLI	QNLI	SST2	QQP	MRPC	CoLA	RTE	STS-B	Avg.
Baselines										
Full Fine-Tuning [31]	125	86.4	92.3	94.2	88.0	92.5	61.1	77.4	90.6	85.3
UNIPELT [249]	1.4	87.2	92.0	93.9	87.7	91.7	61.9	74.3	90.3	84.9
Compacter [364]	1.3	86.1	92.4	94.1	86.7	90.4	55.5	68.6	90.0	83.0
Compacter++ (repr)	1.3	85.2	92.0	94.1	84.6	91.9	64.1	76.2	90.5	84.8
$(IA)^3$ [364]	1.2	86.2	91.9	94.4	86.4	92.3	63.0	76.9	90.6	85.2
TopLoRA [250]	0.9	85.7	92.1	93.4	88.4	88.7	60.3	78.3	88.9	84.5
AdaMix Adapter (repr.)	0.8	83.2	93.3	94.9	86.7	92.3	64.5	82.3	90.8	86.0
HydraLoRA [250]	0.7	85.5	91.9	93.2	87.6	89.5	57.0	73.7	88.5	83.4
Pfeiffer Adapter (repr.)	0.6	87.1	92.5	94.3	87.9	92.6	63.0	79.0	90.8	85.9
Pfeiffer Adapter_{32} (repr.)	0.6	87.0	92.5	94.6	87.9	92.3	63.1	79.3	90.7	85.9
DoRA [250]	0.6	85.3	91.9	93.4	87.9	87.8	56.5	74.8	88.5	83.3
MELoRA [250]	0.6	85.0	91.5	93.0	87.5	88.7	54.4	75.5	87.3	82.9
LoRA [364]	0.3	**87.6**	**93.1**	**95.0**	88.5	92.6	64.0	80.3	90.7	**86.4**
DyLoRA [452]	0.3	87.0	93.0	94.3	**90.0**	91.7	60.5	76.9	91.0	85.5
AdaLoRA [521]	0.3	87.0	93.0	94.0	89.9	89.4	58.8	75.9	90.6	85.0
AutoLoRA [521]	0.3	87.0	92.9	94.9	90.3	89.4	61.3	77.0	90.8	85.5
BoRA [251]	0.3	85.7	91.8	93.1	87.9	88.2	57.4	76.9	89.3	83.8
Prefix-Tuning [364]	0.2	87.0	92.5	95.2	87.1	91.1	61.6	71.1	90.1	84.5
Bit-Fit [31]	0.1	84.8	91.3	93.7	84.5	92.0	61.8	77.8	90.8	84.6
RoAd [260]	0.09	86.3	91.9	93.9	89.6	89.2	64.4	78.9	90.5	85.6
Vera (repr+ [234])	0.04	87.1	91.8	94.6	87.0	89.5	**65.6**	78.7	90.7	85.6
Hadamard Adapter [80]	0.03	85.5	91.6	93.9	87.7	92.1	64.3	80.7	90.1	85.7
LoReFT [491]	0.02	83.1	91.2	93.4	87.4	89.2	60.4	79.0	90.0	84.2
Our approaches										
AdaKron	0.2	$86.7_{0.14}$	$92.4_{0.29}$	$94.0_{0.17}$	$87.5_{0.05}$	$92.6_{0.49}$	$62.2_{0.91}$	$79.4_{1.74}$	$90.9_{0.12}$	85.7
Full MAdaKron	0.2	$86.4_{0.17}$	$92.6_{0.10}$	$94.8_{0.21}$	$87.4_{0.14}$	$93.1_{0.40}$	$62.6_{0.73}$	$76.5_{0.37}$	$90.0_{0.07}$	85.4
Partial MAdaKron	0.2	$86.6_{0.06}$	$92.8_{0.37}$	$94.7_{0.19}$	$87.5_{0.18}$	$\mathbf{93.3}_{0.45}$	$64.6_{0.84}$	$\mathbf{80.5}^{\dagger}_{0.45}$	$\mathbf{91.1}^{\dagger}_{0.10}$	**86.4**

Table 5.7: Main results on the GLUE validation with RoBERTa-base.

Model	# Params (M)	MNLI	QNLI	SST2	QQP	MRPC	CoLA	RTE	STS-B	Avg.
Baselines										
Full Fine-Tuning [475]	355	90.2	94.7	96.4	92.2	90.9	68.0	86.6	**92.4**	88.9
UNIPELT (repr.)	3.2	90.3	94.0	96.3	88.9	93.2	**72.1**	**89.8**	90.2	89.3
$(IA)^3$ (repr.)	2.3	87.8	92.7	95.2	87.8	92.9	70.1	86.3	92.0	88.1
Compacter (repr.)	2.3	67.6	80.6	82.1	78.8	85.3	68.3	60.4	78.6	75.2
Compacter++ (repr.)	2.2	89.5	94.6	96.0	88.7	93.1	68.3	86.7	91.9	88.6
Houlsby Adapter [475]	0.8	90.3	94.7	96.3	91.5	87.7	66.3	72.9	91.5	86.4
Pfeiffer Adapter	0.8	90.5	94.8	96.6	**91.7**	89.7	67.8	80.1	91.9	87.9
AdaMix [475]	0.8	**90.9**	**95.4**	**97.1**	89.8	**94.1**	70.2	89.2	**92.4**	**89.9**
LoRA [475]	0.8	90.6	94.8	96.2	91.6	90.2	68.2	85.2	92.3	88.6
MoKA [509]	0.2	90.6	94.9	96.3	91.1	91.2	69.3	87.4	92.2	89.1
BitFit [498]	0.1	90.0	94.5	96.1	86.4	93.4	68.1	87.3	91.9	88.5
Hadamard Adapter [80]	0.1	89.5	95.1	96.1	90.3	92.5	67.8	85.9	91.9	88.6
Vera (repr+ [234])	0.02	90.0	94.4	96.1	89.7	90.9	68.0	85.9	91.7	88.3
DoRA [250]	1.6	89.3	93.5	95.7	88.3	88.7	61.7	80.1	90.9	86.0
MELoRA [250]	1.6	88.9	93.2	95.9	87.9	87.8	60.6	79.5	90.2	85.5
HydraLoRA [250]	1.7	89.3	93.4	95.8	88.3	89.5	61.1	79.4	90.6	85.9
TopLoRA [250]	2.4	89.9	94.2	95.6	88.9	90.4	64.6	85.2	91.5	87.6
BoRA [251]	0.9	89.8	93.8	95.3	88.6	89.0	60.6	83.8	91.3	86.5
Our approaches										
AdaKron	0.6	$90.2_{0.19}$	$94.8_{0.25}$	$96.9^{*}_{0.10}$	$91.0_{0.33}$	$90.9_{0.74}$	$69.2_{2.26}$	$87.4_{1.13}$	$92.1_{0.08}$	89.1
Full MAdaKron	0.6	$90.4_{0.14}$	$95.0_{0.26}$	$96.5_{0.11}$	$88.8_{0.18}$	$92.0_{1.48}$	$68.0_{0.96}$	$87.7_{0.17}$	$90.8_{0.20}$	88.6
Partial MAdaKron	0.6	$90.6_{0.17}$	$95.0_{0.21}$	$96.6_{0.06}$	$88.5_{0.23}$	$94.0_{0.41}$	$69.0_{0.63}$	$88.9_{0.23}$	$92.0_{0.20}$	89.3

Table 5.8: Main results on the GLUE validation with RoBERTa-Large.

Table 5.7 confirms that the proposed approaches remain highly competitive when using RoBERTa-base as the backbone. AdaKron provides a consistent improvement over baselines that require substantially more trainable parameters. For instance, it outperforms both Compacter and Compacter++ while training fewer parameters. Similarly, AdaKron improves over UNIPELT and $(IA)^3$, despite these methods introducing additional trainable components. Moreover, AdaKron also surpasses lightweight approaches such as BitFit, Vera, and LoRA, particularly on

low-resource tasks such as CoLA and RTE. These findings suggest that the Kronecker interaction allows AdaKron to preserve a richer intermediate representation compared to Houlsby and Pfeiffer adapters, thereby improving robustness under limited supervision. Building on AdaKron, Partial MAdaKron further improves performance on RoBERTa-base, achieving consistent gains across multiple tasks. In particular, it yields notable improvements on semantic similarity and inference tasks such as STS-B and RTE, indicating that dynamic expert selection is beneficial when the model must generalise beyond shallow lexical overlap. Compared to reparameterisation-based approaches such as DyLoRA and AdaLoRA, MAdaKron achieves competitive performance while training fewer parameters. Overall, these results confirm that MAdaKron enhances AdaKron's effectiveness without increasing the parameter budget. Table 5.8 reports results obtained with RoBERTa-large. In this setting, AdaKron continues to provide strong performance while requiring only a small fraction of trainable parameters, outperforming several baselines that train substantially more parameters, such as Houlsby and Pfeiffer adapters. AdaKron also remains competitive with LoRA and other lightweight baselines, indicating that the proposed Kronecker-based parameterisation scales effectively to larger model sizes. Regarding MAdaKron, the expert-based extension remains beneficial, although improvements are less uniform than in the BERT-large setting. Partial MAdaKron improves AdaKron on tasks such as MRPC and QQP, suggesting that the MoE mechanism is particularly useful for paraphrase detection and pairwise semantic reasoning. However, Full MAdaKron does not consistently outperform AdaKron, indicating that introducing a MoE structure in combination with a large backbone may lead to less stable adaptation. Finally, AdaMix achieves slightly higher performance than our approaches on RoBERTa-large. Nevertheless, these gains remain limited and come at the cost of training approximately one third more parameters. Overall, these results indicate that AdaKron and MAdaKron provide a more favourable efficiency–performance trade-off, especially in settings where memory and training cost constraints are critical.

	Bert-base							Bert-Large						
Model	**# Params (M)**	**BoolQ**	**RTE**	**WiC**	**WSC**	**CB**	**Avg.**	**# Params (M)**	**BoolQ**	**RTE**	**WiC**	**WSC**	**CB**	**Avg.**
Baselines														
Full Fine-Tuning	110	72.9	68.4	71.1	63.5	83.0	71.8	336	77.7	70.4	**74.9**	**68.3**	94.6	77.1
Houlsby Adapter (repr.)	1.8	**76.7**	71.0	**71.8**	66.1	87.5	74.6	4.8	79.2	75.9	73.6	65.9	92.4	**77.4**
Compacter (repr.)	1.3	74.3	68.1	69.4	66.3	87.5	73.1	2.3	76.7	74.1	69.8	65.6	**95.5**	76.3
Compacter++ (repr.)	1.3	73.3	66.5	69.1	65.6	82.2	71.3	2.2	76.6	74.4	69.6	65.1	91.1	75.4
UNIPELT (repr.)	1.4	75.7	74.7	71.6	65.6	88.4	75.2	3.2	77.6	**78.2**	72.0	67.2	91.1	77.1
$(IA)^3$ (repr.)	1.2	74.5	73.0	70.6	65.4	86.6	74.0	2.3	77.2	77.8	71.1	67.0	91.1	76.8
Pfeiffer Adapter_{48} (repr.)	0.9	75.1	**77.4**	71.0	64.7	78.6	72.8	2.4	77.5	77.3	71.7	66.6	87.5	76.1
AdaMix (repr.)	0.9	75.2	65.0	71.1	65.4	80.3	71.4	1.8	77.4	75.2	71.6	65.4	90.4	76.0
Pfeiffer Adapter_{32} (repr.)	0.6	74.6	72.6	70.4	65.2	79.5	72.4	1.6	75.8	76.9	71.7	66.6	87.5	76.1
LoRA (repr.)	0.3	75.4	67.9	71.4	66.1	90.2	74.2	0.8	78.0	74.3	69.6	65.6	91.6	75.8
Vera (repr.)	0.04	73.7	67.0	71.1	66.1	85.7	72.7	0.02	70.2	71.8	68.8	65.4	91.1	73.5
Our approaches														
AdaKron_{48}	0.6	$73.7_{0.50}$	$72.8_{1.97}$	$70.5_{0.28}$	$65.7_{0.51}$	$90.2_{1.27}$	74.6	1.6	$76.4_{0.71}$	$76.5_{0.46}$	$72.9_{0.70}$	$67.0_{0.51}$	$92.5_{0.95}$	77.1
Partial MAdaKron_{48}	0.6	$74.5_{0.91}$	$74.4_{0.86}$	$71.5_{1.27}$	$\mathbf{67.7}^{\dagger}_{0.51}$	$\mathbf{90.8}_{1.20}$	**75.8**	1.6	$77.2_{0.82}$	$74.4_{0.36}$	$73.6_{0.42}$	$67.0_{0.71}$	$93.3_{0.61}$	77.1

Table 5.9: Main results on the SuperGLUE validation with both BERT-base and Large.

Model	# Params (M)	BoolQ	RTE	WiC	WSC	CB	Avg.	# Params (M)	BoolQ	RTE	WiC	WSC	CB	Avg.
	RoBERTa-base							RoBERTa-Large						
Baselines														
Full Fine-Tuning [180]	125	83.5	52.7	69.3	58.7	89.3	70.7	355	**86.9**	86.6	**75.6**	63.5	94.6	77.2
Compacter (repr.)	1.3	79.2	77.4	68.9	64.7	94.2	76.9	2.3	79.2	77.4	68.9	64.7	94.2	76.9
Compacter++ (repr.)	1.3	79.7	77.7	67.2	64.9	90.2	75.9	2.2	85.5	86.8	70.9	64.4	82.2	78.0
UNIPELT (repr.)	1.4	81.6	81.5	69.3	65.1	96.4	**78.7**	3.2	85.8	89.0	71.6	64.7	99.5	82.1
$(IA)^3$ (repr.)	1.2	80.2	79.8	**69.5**	**67.0**	96.4	78.6	2.3	85.3	87.6	71.3	65.9	97.7	81.6
Pfeiffer Adapter (repr.)	0.6	80.8	82.6	61.6	64.7	92.0	76.3	0.8	86.5	**89.4**	70.4	65.6	98.2	82.0
LoRA (repr.)	0.3	80.7	78.2	67.2	64.5	96.0	77.3	0.2	80.7	78.2	67.2	64.5	96.0	77.3
Houlsby Adapter (repr.)	0.2	81.8	80.4	70.6	64.9	93.8	78.3	0.8	81.8	80.7	70.6	64.9	93.8	78.3
AdaMix (repr.)	0.3	**81.2**	82.3	66.9	63.5	96.0	78.0	0.8	85.7	88.1	73.4	64.4	92.3	80.8
$\text{Pfeiffer Adapter}_{16}$ (repr.)	0.3	80.2	81.0	64.4	64.4	92.9	76.8	0.8	86.3	89.0	72.1	65.1	96.0	81.7
Vera (repr.)	0.06	77.2	73.9	65.5	64.2	92.4	74.7	0.02	85.4	85.6	71.5	65.3	93.3	80.2
Our approaches														
AdaKron	0.2	$79.7_{0.32}$	$81.1_{0.77}$	$69.4_{0.81}$	$65.4^{*}_{0.03}$	$96.4_{1.75}$	78.4	0.6	$86.0_{0.17}$	$87.8_{0.61}$	$71.1_{0.36}$	$64.9_{0.70}$	$\mathbf{100}^{**}_{0.10}$	82.0
Partial MAdaKron	0.2	$80.5_{0.28}$	$\mathbf{82.7}_{0.35}$	$68.8_{0.77}$	$64.4_{0.14}$	$\mathbf{97.3}_{1.03}$	**78.7**	0.6	$86.5_{0.17}$	$89.0_{0.40}$	$72.2_{0.87}$	$64.9_{0.70}$	$\mathbf{100}^{**}_{0.11}$	**82.5**

Table 5.10: Main results on the SuperGLUE validation with both RoBERTa-base and Large.

For SuperGLUE, we report only Partial MAdaKron in Tables 5.9 and 5.10, since it achieves higher performance than the Full variant in three out of four models on GLUE. Overall, both AdaKron and MAdaKron achieve strong performance on reasoning-oriented tasks. In particular, MAdaKron consistently attains the best or second-best performance across all backbone models while training substantially fewer parameters than adapter-based baselines. When MAdaKron does not achieve the best performance, the gap remains below 0.3 points, highlighting its stable behaviour across the evaluated architectures. On BERT-base, AdaKron outperforms by at least 0.6 points several strong baselines requiring more trainable parameters, including UNIPELT, Compacter, Compacter++, and both Houlsby and Pfeiffer adapters. This indicates that AdaKron is effective also on tasks requiring deeper semantic reasoning, which cannot be solved through shallow lexical matching. Partial MAdaKron further improves AdaKron across most tasks, confirming the benefit of introducing an explicit expert-based mechanism. Improvements are particularly pronounced on low-resource tasks such as RTE and CB, where models typically struggle to generalise due to limited supervision. This suggests that expert specialisation is beneficial in low-data scenarios. Notably, compared to AdaMix, MAdaKron achieves consistently stronger results across all backbones while requiring fewer trainable parameters. Similar trends are observed when scaling to BERT-large. AdaKron continues to outperform most baselines that train more parameters, including Pfeiffer and Houlsby adapters, while remaining competitive with full fine-tuning. Partial MAdaKron further improves AdaKron, achieving the strongest overall performance on this backbone. These results suggest that the MoE-based extension becomes increasingly beneficial at larger model scales, where the model can better exploit the flexibility introduced by expert selection. On RoBERTa-base and RoBERTa-large, AdaKron again provides strong improvements over adapter baselines and compact PEFT approaches, while MAdaKron achieves the best overall performance across most settings. The largest gains are observed on tasks such as BoolQ and RTE, which require modelling subtle entailment relations and question–answer consistency. Overall, the SuperGLUE results confirm that AdaKron is a robust alternative to standard adapters, and that MAdaKron

further improves adaptation capacity without increasing parameter requirements.

Model	# Params (M)	MNLI	QNLI	SST2	QQP	MRPC	CoLA	RTE	STS-B	Avg.
Baselines										
Full Fine-Tuning [518]	184	90.1	94.0	95.6	**89.8**	89.5	69.2	83.7	91.6	87.0
UNIPELT (APL) [249]	0.9	89.1	93.7	95.6	89.6	92.4	68.0	81.6	**91.7**	87.7
Compacter (repr.)	0.7	89.4	93.9	95.7	86.6	92.5	69.3	85.6	91.2	88.1
Compacter ++ (repr.)	0.6	89.7	94.1	95.7	87.3	93.1	69.4	85.9	91.2	88.1
Houlsby Adapter [518]	0.6	90.2	93.8	95.3	88.9	89.2	67.9	85.5	91.3	88.1
Pfeiffer Adapter [518]	0.6	90.3	94.0	95.5	88.9	89.2	69.5	84.1	91.5	88.2
Houlsby Adapter	0.31	90.02	93.52	95.41	88.81	89.25	67.75	83.39	91.31	87.77
AdaMix (repr.)	0.6	90.1	94.3	95.7	88.6	91.1	**71.0**	87.4	**91.7**	88.7
$(IA)^3$ (repr.)	0.6	87.7	92.2	95.2	86.2	**93.4**	69.6	86.0	91.3	87.7
DiffoRA [209]	0.4	90.5	94.5	96.0	91.8	90.8	70.8	88.0	91.7	**89.2**
LoRA [518]	0.3	90.4	94.0	94.9	88.9	89.7	68.7	85.6	**91.7**	88.3
AdaLoRA [518]	0.3	**90.7**	94.5	**95.8**	89.2	90.4	70.0	87.4	91.6	**89.0**
Pfeiffer Adapter [518]	0.3	90.1	93.9	94.7	88.6	89.7	69.1	84.5	91.4	88.1
Vera [15]	0.2	90.2	93.5	95.1	87.6	90.9	70.7	87.3	91.1	88.3
BitFit [518]	0.1	89.9	92.2	94.8	84.9	87.7	66.9	78.7	91.3	86.3
Our approaches										
AdaKron_8	0.2	$89.9_{0.12}$	$94.0_{0.21}$	$95.7_{0.20}$	$88.4_{0.15}$	$89.9_{1.51}$	$69.8_{1.08}$	$88.1_{0.52}$	$\mathbf{91.7}_{0.29}$	88.5
AdaKron_{16}	0.4	$89.9_{0.16}$	$93.9_{0.12}$	$95.6_{0.06}$	$88.2_{0.10}$	$90.4_{0.71}$	$70.4_{1.84}$	$\mathbf{88.6}^{*,\dagger}_{0.41}$	$91.6_{0.11}$	88.6
Partial MAdaKron_{16}	0.4	$89.8_{0.13}$	$\mathbf{94.3}_{0.18}$	$95.3_{0.11}$	$87.2_{0.12}$	$90.2_{0.93}$	$70.6_{0.96}$	$88.0_{0.43}$	$\mathbf{91.7}_{0.19}$	88.4

Table 5.11: Main results on the GLUE validation with DeBERTa-v3-base.

Model	# Params (M)	SQuAD v1.1 EM	SQuAD v1.1 F1	SQuAD v2.0 EM	SQuAD v2.0 F1
Baselines					
Full Fine-Tuning [518]	184	86.0	92.7	85.4	88.4
AdaMix (repr.)	0.6	**87.7**	**93.6**	85.2	88.1
Pfeiffer Adapter [518]	0.6	86.2	92.8	84.9	87.8
Pfeiffer Adapter [518]	0.3	85.9	92.5	84.5	87.6
Houlsby Adapter [518]	0.15	84.4	91.5	83.4	86.6
Houlsby Adapter [518]	0.3	85.3	92.1	84.3	87.3
Houlsby Adapter [518]	0.6	86.2	92.8	84.9	87.9
Houlsby Adapter [518]	1.2	86.6	93	85.4	88.3
LoRA [518]	0.3	86.6	92.9	83.6	86.7
LoRA [518]	0.6	86.1	92.7	84.5	87.4
LoRA [518]	1.2	86.7	92.9	85	88
LoRA [518]	0.2	86.4	92.8	84.7	87.2
AdaLoRA [518]	0.2	87.2	**93.4**	**85.6**	**88.7**
Adalora [518]	0.30	87.5	93.6	85.8	88.8
Adalora [518]	0.6	87.5	93.7	85.5	88.6
Adalora [518]	1.2	87.6	93.7	86	88.9
DiffoRA [209]	0.1	**87.6**	**93.5**	84.2	87.2
Our approaches					
AdaKron_{16}	0.4	$87.0_{0.17}$	$93.2_{0.17}$	$84.7_{0.14}$	$87.6_{0.28}$
AdaKron_8	0.2	$87.2_{0.20}$	$93.2_{0.10}$	$84.3_{0.26}$	$87.3_{0.26}$
AdaKron_{32}	0.40	$87.0_{0.09}$	$93.2_{0.05}$	$84.1_{0.20}$	$87.2_{0.20}$
AdaKron_{64}	0.77	$87.1_{0.14}$	$93.1_{0.01}$	$82.8_{0.05}$	$85.8_{0.05}$
Partial MAdaKron_8	0.2	$87.5^{*}_{0.08}$	$92.2_{0.07}$	$85.3_{0.11}$	$88.3_{0.13}$

Table 5.12: Main results on the SQuAD with DeBERTa-v3-base.

We further evaluate AdaKron and Partial MAdaKron on DeBERTa-v3-base. Results are reported in Tables 5.11 and 5.12, which present performance on GLUE and both SQuAD datasets, respectively. Overall, AdaKron and MAdaKron remain highly competitive, showing that the proposed approach generalises beyond BERT-like architectures and remains effective under DeBERTa's disentangled attention mechanism. On GLUE, AdaKron consistently matches or improves upon

strong PEFT baselines such as Houlsby and Pfeiffer adapters, while training more than one third fewer parameters. As in previous experiments, the largest gains are observed on low-resource tasks such as RTE and CoLA, confirming that Kronecker-based adaptation is particularly robust when supervision is limited. On question answering, both AdaKron and MAdaKron perform on par with, and in some cases outperform, Houlsby and Pfeiffer adapters on both SQuAD v1.1 and v2.0, despite requiring fewer trainable parameters. These results indicate that AdaKron preserves strong representational capacity even when the task requires modelling question–answer interactions. Finally, AdaLoRA achieves slightly higher performance on SQuAD v2.0, likely because it explicitly optimises rank allocation across layers, which aligns well with DeBERTa's architectural characteristics. Nevertheless, AdaKron and MAdaKron remain competitive, offering a simpler and more parameter-efficient alternative.

As a final robustness evaluation, we scale AdaKron to a 1.5B-parameter configuration based on DeBERTa-v2-XXL and test it on a subset of GLUE tasks, namely MRPC, CoLA, RTE, and STS-B. These benchmarks span both low- and high-resource settings and include single-sentence as well as sentence-pair classification tasks. The results show that AdaKron approaches the performance of full fine-tuning while updating only 0.06% of the total model parameters. In particular, it achieves scores of 90.4 on MRPC (compared to 92.0 with full fine-tuning), 70.6 on CoLA (vs. 72.0), 88.9 on RTE (vs. 89.9), and 91.9 on STS-B (vs. 92.9). Although the average performance gap with respect to full fine-tuning is around three points, the difference on STS-B is reduced to a single point, despite the fact that AdaKron updates only approximately one million parameters. These findings indicate that AdaKron scales effectively to billion-parameter models without the need for full model optimisation, making it well suited for large-scale adaptation settings.

Model	**# Params (M)**	Spanish	Hindi	Russian	Romanian	Marathi	Hausa	Nigerian Pidgin
LoRA (repr.)	0.3	67.3	70.6	**82.6**	**66.8**	86.5	59.5	**55.8**
Pfeiffer Adapter (repr.)	0.9	69.8	70.1	81.1	62.9	84.9	**60.2**	52.0
AdaKron	0.6	**72.1**	**72.4**	82.1	66.4	**87.2**	59.4	54.7

Table 5.13: Main results on the SemEval 2025 "Bridging the Gap in Text-Based Emotion Detection" Task.

Beyond English benchmarks, we further evaluate AdaKron on the validation split of the SemEval-2025 task "Bridging the Gap in Text-Based Emotion Detection" [322], which covers both high- and low-resource languages. For this study, we consider a representative selection of seven languages, including four high-resource languages (Spanish, Hindi, Russian, and Romanian) and three low-resource languages (Hausa, Nigerian Pidgin, and Marathi). Results are reported in Table 5.13. Overall, AdaKron consistently outperforms both LoRA and Pfeiffer adapters across

all selected languages, showing that the proposed approach is robust beyond English. Notably, improvements are particularly large for low-resource languages, where adaptation methods often suffer from instability and limited training data. This suggests that AdaKron is effective at extracting informative representations even when the training distribution is sparse or noisy. Overall, these results confirm that the proposed approach generalises effectively to multilingual scenarios.

Model	**# Params (M)**	**DART** BLEU	**DART** TER ↓	**WebNLG** BLEU	**WebNLG** TER
Baselines					
Full Fine-Tuning [475]	355	46.2	**0.46**	46.5	0.53
Fine-Tuning Top2 [475]	25	41.0	0.56	36.0	0.72
Houlsby Adapter [475]	11	45.2	**0.46**	54.9	**0.39**
AdaMix Adapter [475]	0.4	**47.7**	0.47	54.9	**0.39**
DyLoRA [452]	0.4	46.5	0.48	54.5	**0.39**
Prefix-Tuning [475]	0.3	46.4	**0.46**	55.1	0.40
LoRA [475]	0.3	47.1	**0.46**	**55.3**	**0.39**
Our approaches					
AdaKron	0.3	$45.0_{0.37}$	$0.50_{0.01}$	$51.6_{0.42}$	$0.43_{0.01}$
Partial MAdaKron	0.3	$46.6_{0.22}$	$0.49_{0.01}$	$53.3_{0.54}$	$0.42_{0.01}$

Table 5.14: Main results on WebNLG and DART with GPT-2-medium.

Model	**# Params (M)**	BLEU	NIST	MET	ROUGE-L	CIDEr
Baselines						
Full Fine-Tuning [475]	355	68.2	8.62	46.2	71.0	2.47
Fine-Tuning Top2 [475]	25.0	68.1	8.59	46.0	70.8	2.41
Houlsby Adapter [475]	11.0	68.9	8.71	46.1	71.3	2.47
AdaMix Adapter [475]	0.4	69.8	8.75	46.8	**71.9**	2.52
DyLoRA [452]	0.4	69.2	8.75	46.3	70.8	2.46
Vera [234]	0.4	70.1	8.81	46.6	71.5	2.50
Prefix-Tuning [475]	0.3	69.7	8.81	46.1	71.4	2.49
LoRA [475]	0.3	**70.4**	**8.85**	46.8	71.8	**2.53**
FourierFT [133]	0.048	69.1	8.82	**47.0**	71.8	2.51
Our approaches						
AdaKron	0.3	$69.3_{0.91}$	$8.65_{0.14}$	$45.8_{0.78}$	$71.3_{0.70}$	$2.41_{0.08}$
Partial MAdaKron	0.3	$69.2_{0.07}$	$8.68_{0.04}$	$46.3_{0.05}$	$71.1_{0.22}$	$2.46_{0.01}$

Table 5.15: Main results on E2E with GPT-2-medium.

Tables 5.14 and 5.15 report the performance of AdaKron and MAdaKron on E2E, WebNLG, and DART using GPT-2 medium, while Table 5.16 reports results on E2E using GPT-2 large. Overall, AdaKron and MAdaKron achieve competitive performance across all tasks, showing that the proposed approach generalises to autoregressive models and generation settings. On E2E, both methods outperform full fine-tuning across all evaluated metrics while training less than 1% of the original model parameters. On WebNLG and DART, AdaKron and MAdaKron perform comparably to LoRA and Houlsby adapters despite requiring fewer trainable

parameters. Moreover, MAdaKron consistently improves over AdaKron, achieving higher ROUGE-L and BLEU scores. Finally, results on GPT-2 large further confirm that both methods remain effective when scaling to larger autoregressive backbones (774M parameters), outperforming full fine-tuning while training only 0.6M parameters.

Model	# Params (M)	BLEU	NIST	MET	ROUGE-L	CIDEr
Baselines						
Full Fine-Tuning [186]	774	68.5	8.78	46.0	69.9	2.45
Houlsby Adapter [186]	0.9	69.1	8.68	46.3	71.4	2.49
Prefix-Tuning [186]	0.7	70.3	8.85	46.2	71.7	2.47
LoRA [186]	0.7	**70.4**	8.89	46.8	**72.0**	2.47
Vera [234]	0.2	70.3	8.85	46.9	71.6	**2.54**
FourierFT [133]	0.072	70.2	**8.92**	**47.0**	71.8	2.50
Our approaches						
AdaKron	0.6	$68.8_{0.46}$	$8.70_{0.01}$	$46.4_{0.37}$	$71.5_{0.34}$	$2.45_{0.01}$
Partial MAdaKron	0.6	$69.5_{0.55}$	$8.76_{0.06}$	$46.6_{0.15}$	$71.4_{0.28}$	$2.49_{0.01}$

Table 5.16: Main results on E2E with GPT-2-Large.

5.3.1 Ablation Study and Analysis

This section analyses the main design choices behind AdaKron and MAdaKron. In particular, we investigate: (i) the impact of the intermediate dimension and reduction factor, (ii) alternative formulations to the Kronecker product, and (iii) the contribution of the main Mixture-of-Experts (MoE) components in MAdaKron.

Intermediate Dim. $r = r_1 \cdot r_2$	8	16	32	48	64	256	8	16	32	48	64	256	32	48
Down projections Dim. r_1, r_2	2,4	2,8	2,16	2,24	2,32	2,128	4,2	4,4	4,8	4,12	4,16	4,64	8,4	12,4
# Params (M)	0.1	0.2	0.4	0.6	0.9	3.5	0.1	0.2	0.3	0.6	0.8	3.0	0.3	0.6
Average Performance	86.6	87.4	87.7	87.7	88.1	88.2	86.9	87.5	87.8	**88.3**	88.2	88.2	87.6	88.0

Table 5.17: Ablation study on the four new GLUE development datasets with BERT-base to study the impact of the intermediate dimension.

Intermediate Dimension We conduct an ablation study to evaluate the impact of the AdaKron intermediate dimension r and the reduction factor r_2, which together determine the sizes of the two down-projection layers (i.e., $r_1 = r/r_2$ and r_2), and thus control both the expressive capacity and the number of trainable parameters. For this analysis, we partition the original training sets of SST-2, CoLA, RTE, and STS-B into 85% for training and 15% for evaluation. We select these datasets because they cover a broad spectrum of tasks, including both high- and low-resource scenarios, as well as single-sentence and sentence-pair classification problems. We investigate multiple intermediate dimension sizes, specifically

8, 16, 32, 48, 64, and 256, together with two reduction factors, namely 2 and 4. The results show that small intermediate dimensions (e.g., 8 and 16) yield the lowest average performance, whereas larger dimensions (e.g., 64 and 256) improve effectiveness at the cost of a higher number of trainable parameters. Overall, the configuration $r = 48$ with $r_2 = 4$ provides the best trade-off, achieving the highest average performance across tasks while keeping the number of trainable parameters below 0.6% of the backbone. In contrast, configurations that produce excessively small blocks within the Kronecker structure lead to a degradation in performance. For instance, fixing the same intermediate size but increasing the reduction factor to $r_2 = 12$ produces blocks that are too small, which negatively impacts the final results.

Dim. r	**# Params (M)**	**SST2** Acc	**CoLA** Mcc	**RTE** Acc	**STS-B** Pearson	**Avg.**
8	0.1	95.0	**85.2**	**93.4**	<u>94.5</u>	**92.0**
16	0.2	**96.5**	83.7	<u>92.2</u>	**94.8**	<u>91.8</u>
32	0.4	95.7	<u>85.0</u>	92.1	92.7	91.3
64	0.8	<u>95.8</u>	83.3	89.7	91.1	90.0

Table 5.18: Ablation Study on the four newly introduced GLUE development sets using DeBERTa-v3-base as the pretrained language model, investigating the optimal intermediate dimension for both AdaKron and MAdaKron.

We further analyse the impact of the intermediate dimension using DeBERTa-v3-base on four GLUE tasks. Results reported in Table 5.18 indicate that, unlike BERT, DeBERTa achieves its best performance with a smaller intermediate dimension, namely $r = 16$. Increasing the dimension to 32 or 64 does not consistently improve results, and in some cases slightly decreases performance on SST2 and STS-B. This suggests that DeBERTa can effectively exploit more compact adaptation modules, while larger intermediate representations may introduce overfitting. Consequently, we adopt $r = 16$ as the default setting for DeBERTa-based experiments.

Variations of Kronecker Product In this section, we investigate the role of the Kronecker product within the AdaKron layer. We evaluate AdaKron against several alternative formulations, including variants in which the Kronecker operator is applied to different adapter components, symmetric versions of the operation, Kronecker-sum formulations, as well as other matrix interactions such as Hadamard-like and Khatri–Rao products. The corresponding results are summarised in Table 5.19. In the original AdaKron design, the Kronecker product is applied exclusively to the down-projection modules. In addition to this baseline configuration, we examine variants where the operator is instead applied to the

up-projection layer, as well as configurations where it is used in both up- and down-projection stages. Given the non-commutative nature of the Kronecker product, we further introduce a symmetric variant to evaluate the influence of operand ordering. Specifically, the Symmetric AdaKron variant applies the Kronecker product twice, swapping the input vectors and summing the results:

$$g(y_c, y_v) = g(y_v, y_c) := GELU(y_c \otimes y_v + y_v \otimes y_c) \in \mathbb{R}^{r_1 \cdot r_2},$$

where $y_v \in \mathbb{R}^{r_1}$, $y_c \in \mathbb{R}^{r_2}$, and $g(\cdot)$ denotes the output of the down-projection layers, as defined in Section 5.1.

We further investigate Kronecker-sum variants defined as:

$$y_v \oplus y_c = y_v \odot \mathbf{1}_{r_2} + y_c \odot \mathbf{1}_{r_1},$$

where $\mathbf{1}_n = (1,1,\ldots,1) \in \mathbb{R}^n$ is the all-ones vector. We evaluate both the pure sum formulation and a hybrid variant that combines product and sum outputs. Beyond Kronecker-based formulations, we evaluate alternative matrix products. Since AdaKron produces down-projection outputs $y_c \in \mathbb{R}^{r_1}$ and $y_v \in \mathbb{R}^{r_2}$ with $r_1 \neq r_2$, the standard Hadamard product cannot be directly applied. We therefore adopt the penetrating face product, defined as:

$$y_c[\odot]y_v = [(y_c)_1 \odot y_v \mid \cdots \mid (y_c)_{r_1} \odot y_v] \in \mathbb{R}^{r_1},$$

where $\odot$ denotes the element-wise product. Compared to the Kronecker product, which produces a vector of size $r_1 \cdot r_2$, this operation yields a lower-dimensional output, potentially introducing a representational bottleneck. We also investigate the Khatri–Rao product, defined for matrices A and B as:

$$A \star B = (A_{i,j} \otimes B_{i,j})_{i,j},$$

where each block is the Kronecker product of the corresponding blocks of A and B. In our setting, the Khatri–Rao product between y_c and y_v is defined as:

$$y_c \star y_v = [(y_c)_1 \otimes y_v \mid \cdots \mid (y_c)_{r_1} \otimes y_v] \in \mathbb{R}^{r_1 \times r_2}.$$

The results indicate that the original AdaKron configuration-where the Kronecker product is applied solely to the down-projection layers-yields the highest average performance. It outperforms all other variants by at least 0.4 points and is the only setting that exceeds the Pfeiffer adapter baseline. In contrast, moving the Kronecker operation to the up-projection reduces the number of trainable parameters but results in a notable degradation in performance, with an average drop of more than one point. This behaviour is consistent with the functional role of the up projection: since it maps the intermediate representation back into the transformer hidden space, imposing excessive structural constraints may prevent

Model	**# Params (M)**	**SST2** Acc	**CoLA** Mcc	**RTE** Acc	**STS-B** Pearson	**Avg.**
Pfeiffer Adapter (repr.)	0.9	96.8	78.7	84.1	**92.8**	88.1
AdaKron	0.6	**96.9**	78.8	84.7	**92.8**	**88.3**
Kronecker product in Up projection	0.5	96.7	77.6	**84.9**	89.7	87.2
Kronecker product in both Down and Up projection	0.3	96.4	**81.6**	72.3	90.5	85.2
Symmetric AdaKron	0.6	96.6	77.5	84.8	92.4	87.8
Kronecker Sum	0.6	96.0	78.5	83.7	92.5	87.7
AdaKron + Kronecker Sum	0.6	95.2	78.0	84.7	92.6	87.6
Hadamard product	0.2	96.5	76.8	83.7	91.9	87.2
Khatri-Rao product	0.6	96.7	78.1	84.7	92.4	87.9

Table 5.19: Ablation study with BERT-base to study the impact of the Kronecker product on the new introduced development sets.

the adapter output from properly aligning with the residual connection. Moreover, symmetric Kronecker and Kronecker-sum variants underperform the original formulation. Although these alternatives introduce commutativity or additive combinations, they also reduce the separable structure induced by the Kronecker expansion, effectively blending the two projected representations and limiting expressiveness. Similarly, the penetrating face product yields approximately one point lower performance on average, likely due to its reduced output dimensionality. While the Khatri–Rao product performs closer to the Kronecker formulation, it still underperforms AdaKron, suggesting that its block-wise structure restricts global interactions across projected components. Overall, these results indicate that AdaKron's gains are not simply attributable to increasing intermediate dimensionality, but rather stem from the specific representational structure induced by the Kronecker product, which expands the feature space while maintaining parameter efficiency.

MAdaKron	**# Params (M)**	**SST2** Acc	**CoLA** Mcc	**RTE** Acc	**STS-B** Pearson	**Avg.**
Pfeiffer Adapter (repr.)	0.9	96.9	78.7	84.1	**92.9**	88.1
MAdaKron	0.6	**97.0**	79.1	85.6	92.2	**88.5**
w/o Consistency	0.6	96.5	78.7	84.3	92.1	87.9
w/o Merging, with Consistency	0.6	96.6	78.7	85.2	92.3	88.2
w/o Merging, w/o consistency	0.6	96.1	78.0	84.5	92.1	87.7
biased probability, with Consistency	0.6	96.4	78.8	**86.3**	92.3	88.4
biased probability, w/o consistency	0.6	95.3	**79.6**	82.5	92.3	87.4
Softmax Gating, with balancing	0.7	96.6	78.2	84.7	92.8	88.0
Softmax Gating, w/o balancing	0.7	96.7	78.0	83.4	92.7	87.7

Table 5.20: Ablation results on the four newly introduced GLUE development sets using BERT-base, analyzing the effects of the MAdaKron architecture, the inference-time merging strategy, the consistency loss, and the learnable gating mechanism.

Variations of Mixture of Experts We perform an ablation study to evaluate the contribution of the key components of the Mixture-of-Experts mechanism used in MAdaKron. We first examine the expert activation distribution. Given that experts are sampled uniformly at random, each expert is selected in roughly 25% of the training steps. Importantly, no individual expert is preferentially activated, which ensures a balanced update frequency across all experts during optimisation. To further disentangle the impact of the main design choices in MAdaKron, we investigate three variants:

- removing the consistency regularisation term from the training objective;
- substituting uniform sampling with a skewed selection strategy, where the first expert is chosen with probability 0.7 and the remaining experts with probability 0.1 each;
- replacing random selection with a learnable softmax gating mechanism trained jointly with the model parameters and regularised via a load-balancing objective, while selecting the top-2 experts at inference time.

The corresponding results are presented in Table 5.20. In all experiments, the number of experts is fixed to four and the adapter bottleneck dimension is set to 48. Overall, the complete MAdaKron configuration—including consistency regularisation during training and expert aggregation at inference—achieves the strongest performance among the considered variants. Removing the consistency loss results in an average drop of 0.6 points, indicating that the KL-based regulariser is crucial to stabilise expert training and prevent experts from converging to inconsistent local optima. Intuitively, each expert minimises the task loss while also aligning its predictions with another randomly selected expert, which promotes agreement and improves generalisation. Biased expert selection does not substantially affect performance when the consistency objective is preserved. However, once the regulariser is removed, biased routing yields an additional drop of approximately one point, suggesting that consistency regularisation is essential for ensuring that rarely activated experts still learn meaningful representations. Moreover, removing expert merging at inference time and instead using random expert selection consistently degrades performance by at least 0.3 points. This suggests that expert merging effectively functions as an implicit ensembling strategy, combining complementary information learned by the different experts, in line with findings from prior work [475]. In addition, substituting stochastic routing with a learned softmax gating mechanism results in an average performance drop of at least 0.5 points, even though it introduces additional trainable parameters. The learned gate exhibits relatively balanced usage of the four experts during training on SST2, CoLA, RTE, and STS-B, with selection frequencies of 24.4%, 28.3%, 23.5%, and 23.8% for the first, second, third, and fourth experts, respectively. Thus, expert selection frequencies remain relatively balanced due to the load-balancing objective;

however, this result is consistent with recent findings on sparsely activated architectures [541], which highlight that learned gating can introduce instability and expert collapse. Overall, these findings indicate that stochastic expert selection provides a more effective balance between simplicity, stability, and performance in the MAdaKron setting.

MAdaKron Model	# Experts	SST2	CoLA	RTE	STS-B	Avg.
Full	2	96.9	77.4	85.4	91.2	87.7
Full	6	95.9	73.3	80.6	87.0	84.2
Full	8	95.2	67.4	83.0	80.0	81.4
Partial	2	96.9	76.0	84.8	**92.3**	87.5
Partial	6	**97.0**	**79.2**	84.7	91.6	88.1
Partial	8	**97.0**	77.9	83.0	91.5	87.3
Full	4	96.7	78.7	83.7	**92.3**	87.9
Partial	4	**97.0**	79.1	**85.6**	92.2	**88.5**

Table 5.21: Ablation study on the new introduced development sets with BERT-base to evaluate the impact of the number of experts on MAdaKron.

Number of Experts We further investigate how the number of experts influences MAdaKron's performance using BERT-base as the backbone model. In particular, we evaluate setups with 2, 4, 6, and 8 experts across four GLUE benchmark tasks. Results are reported in Table 5.21. Overall, the results indicate that using four experts provides the most effective and stable configuration for both Partial and Full MAdaKron. Consequently, we adopt four experts as the default setting in all experiments. Increasing the number of experts beyond this value does not yield consistent improvements and often leads to degraded performance on low-resource tasks. A clear pattern emerges when contrasting low- and high-resource datasets. As the number of experts increases, routing becomes sparser and each expert receives fewer effective parameter updates. This negatively impacts datasets with limited supervision, such as RTE and STS-B, which contain fewer than $7k$ training instances. For instance, on RTE and STS-B, the two-expert configuration outperforms the eight-expert setting by 1.8 and 0.8 points, respectively. This behaviour indicates that, in low-resource scenario, distributing supervision across a larger number of experts can lead to insufficient training of individual components, as the available learning signal becomes overly fragmented. By contrast, high-resource tasks such as SST-2 (with approximately 67k training examples) show little sensitivity to the number of experts, with performance remaining relatively stable across configurations. This suggests that, when sufficient supervision is available, the model can effectively leverage a larger set of experts without suffering from sparse updates. A similar pattern is observed in the generative setting with GPT-2 large: on the E2E dataset (approximately 42k training instances), increasing the number of experts yields modest gains, with MAdaKron achieving 73.8 BLEU using

eight experts compared to 72.8 BLEU with four experts. Overall, these findings indicate that larger expert sets are more beneficial in high-resource scenarios, whereas smaller configurations tend to be preferable in low-resource conditions.

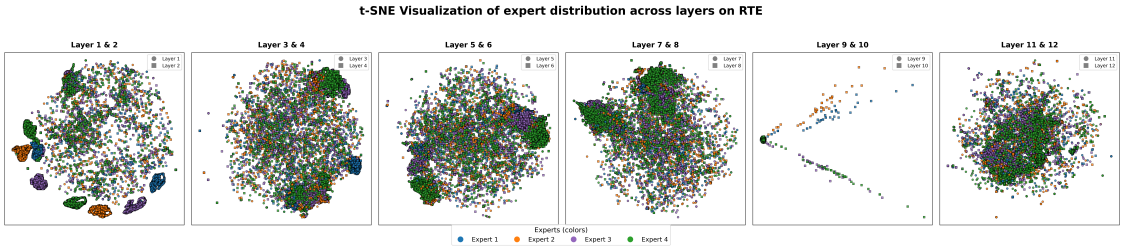

Figure 5.4: t-SNE projection of the 48-dimensional output vectors produced by MAdaKron for a single batch from the RTE dataset. Visualisations are shown for consecutive layer pairs to illustrate how expert representations evolve across the network depth.

Visualisation of MAdaKron Experts To gain further insight into the behaviour of MAdaKron, we analyse the evolution of expert representations across layers when fine-tuning BERT-base on RTE. The goal is to assess whether different experts develop distinct representational roles and how these roles change with depth. We extract the 48-dimensional output vectors produced by MAdaKron for a single batch and project them into a two-dimensional space using t-SNE. For clarity, we restrict the batch size to four and fix the sequence length to 128. The resulting visualisations for successive layer pairs are presented in Figure 5.4, allowing a layer-wise examination of representation dynamics. In the initial layers (layers 1–2), the embeddings form clearly separated clusters, suggesting that each expert specialises in capturing different aspects of the input, primarily related to general lexical and semantic patterns. In intermediate layers (e.g. layers 3–4), these clusters begin to overlap, showing increasing interaction between expert representations. From layers 5–8, a gradual blending of representations is observed, reflecting a transition towards more task-specific features. In deeper layers (layers 9–10), the representations largely converge into a dominant cluster, suggesting that information becomes progressively unified. Finally, in the output layers, expert representations are almost indistinguishable, indicating strong integration prior to prediction. Overall, this analysis reveals a consistent progression in representation dynamics: early layers encourage expert specialisation, intermediate layers facilitate interaction and gradual convergence, and deeper layers consolidate information into a unified representation for final decision-making.

Model	# Params (M)	Training Phase						Inference Stage
		TFLOPs	MACs	Time (s)	Memory Usage (GB)	Energy (J)	CO_2 (mg)	Inference Time (s)
LoRA	0.3	3.09	1.54	33.2	15.0	48.28	5.36	1.31
Pfeiffer Adapter	0.9	3.12	1.56	35.0	19.9	48.67	5.40	1.36
AdaMix	0.9	6.24	3.12	64.8	39.4	97.34	10.82	1.38
AdaKron	0.6	3.11	1.55	36.5	19.8	48.52	5.39	1.37
Partial MAdaKron	0.6	6.22	3.11	65.2	39.3	97.03	10.78	1.39

Table 5.22: Training and Inference Efficiency. Training and inference times are reported in seconds (s), memory usage in GB, energy consumption in Joules, and carbon emissions in milligrams.

5.3.2 Efficiency Analysis

As noted by Lialin et al. [256] and discussed in Chapter 3, parameter efficiency should be understood as a multidimensional notion that goes beyond the mere count of trainable parameters, encompassing factors such as memory usage, training and inference latency, and energy consumption. While the preceding sections primarily focused on model effectiveness, we now turn to an analysis of the computational efficiency of AdaKron and MAdaKron. For this purpose, we fine-tune BERT-base on the RTE dataset for a single epoch, using a batch size of 32 and a maximum sequence length of 512. All experiments are carried out on an NVIDIA A100 GPU. We evaluate a range of efficiency metrics, including peak GPU memory consumption, total training duration, TFLOPs, MACs, estimated energy usage, and the associated CO_2 emissions. The results are reported in Table 5.22. From a parameter perspective, both AdaKron and MAdaKron reduce the number of trainable parameters by roughly one third compared to Pfeiffer adapters and AdaMix, further confirming the parameter efficiency advantages introduced by the Kronecker-based design. From a computational perspective, AdaKron requires 3.11 TFLOPs to complete a full training epoch, which is marginally lower than the 3.12 TFLOPs observed for Pfeiffer adapters. Although the difference is minimal, it suggests that incorporating the Kronecker product does not increase computational overhead and may even slightly reduce it by effectively shrinking the dimensionality of the down-projection operations. In terms of runtime and memory usage, AdaKron records a training time of 36.5 seconds and a peak GPU memory consumption of 19.8 GB, closely matching the figures of Pfeiffer adapters (35.0 seconds and 19.9 GB, respectively). Overall, these results indicate that AdaKron maintains the efficiency characteristics of conventional adapter tuning while providing a more expressive parameterisation. In contrast, MAdaKron requires 6.22 TFLOPs, approximately twice that of AdaKron, which is expected given that each training step involves two forward passes due to the consistency regularisation scheme. Notably, this computational cost is on par with AdaMix, which requires 6.24 TFLOPs. Similarly, MAdaKron exhibits a training time of 65.2 seconds and a memory footprint of 39.3 GB, which are nearly identical to those of AdaMix (64.8 seconds and 39.4

GB). This indicates that MAdaKron achieves its performance improvements without introducing additional overhead beyond what is inherent to consistency-driven MoE training. We further estimate energy usage and carbon emissions. Using FLOP-based approximations for NVIDIA A100, AdaKron consumes 48.5 J and generates 5.39 mg of CO_2, slightly below Pfeiffer adapters (48.67 J and 5.40 mg CO_2). As anticipated, both MAdaKron and AdaMix approximately double the energy consumption of AdaKron due to their two-pass training procedure. Nevertheless, MAdaKron produces a marginally lower carbon footprint than AdaMix (by approximately 0.1 mg CO_2), suggesting that the Kronecker-based formulation introduces a modest reduction in computational cost even within the MoE setting. Finally, we consider inference latency. All adapter-based approaches (Pfeiffer adapters, AdaMix, AdaKron, and MAdaKron) show comparable inference times, ranging between 1.36 and 1.39 seconds, and are slightly slower than LoRA (1.31 seconds). This behaviour is expected, as LoRA operates through modifications of existing weight matrices without introducing additional modules, whereas adapter-based methods add extra computation during the forward pass. Overall, these findings confirm that AdaKron maintains an efficiency profile comparable to standard adapter approaches, while MAdaKron remains computationally competitive with AdaMix.

5.4 Conclusions

In this Chapter, we introduced AdaKron and MAdaKron, two parameter-efficient fine-tuning approaches based on the Kronecker product. AdaKron leverages a Kronecker-based parameterisation to enhance adapter expressiveness while maintaining a compact number of trainable parameters. Building on this formulation, MAdaKron introduces a Mixture-of-Experts architecture within the adapter layer to further increase adaptation capacity, while still requiring the training of less than 1% of the original model parameters. Our experimental evaluation across eight pre-trained language models and eighteen downstream datasets demonstrates both the effectiveness and efficiency of the proposed methods. With respect to RQ1, AdaKron achieves competitive performance across natural language understanding, question answering, and natural language generation benchmarks while updating only a small fraction of model parameters. In multiple settings, AdaKron improves upon both lightweight PEFT methods (e.g., BitFit and LoRA) and heavier adapter-based baselines, confirming that Kronecker-based adaptation provides a strong mechanism for improving expressiveness without increasing parameter cost. Regarding RQ2, we investigated whether incorporating a Mixture-of-Experts component into AdaKron improves performance. The results confirm that MAdaKron consistently strengthens AdaKron on most benchmarks, particularly on challenging

NLU tasks, indicating that dynamic expert selection provides additional representational flexibility. Importantly, MAdaKron achieves these gains while maintaining the same parameter budget as AdaKron, showing that the expert-based extension improves effectiveness without sacrificing parameter efficiency. Finally, in relation to RQ3, MAdaKron achieves state-of-the-art or near state-of-the-art performance across several benchmarks when compared with recently proposed PEFT approaches. MAdaKron integrates a MoE architecture with a random uniform gating function, which outperform the classic learnable gate, as shown to answer RQ4. Overall, these results highlight MAdaKron as a practical and effective fine-tuning strategy, capable of improving adaptation performance while requiring substantially fewer trainable parameters than full fine-tuning.

Chapter 6

DRAMA: Domain Retrieval using Adaptive Module Allocation

Domain adaptation is a central challenge in modern Neural Information Retrieval (NIR). While dense retrieval models achieve strong performance by learning contextual semantic representations, they typically require fine-tuning on in-domain data in order to reach their full effectiveness [146, 219]. As a consequence, retrieval systems deployed in heterogeneous environments often rely on multiple domain-specific models, each specialised to a specific domain or task. Although this strategy yields strong performance, it scales poorly with the number of domains, as it increases storage requirements and computational cost. A popular alternative consists in training a single unified retriever that is expected to generalise across all domains. However, prior work has shown that achieving robust cross-domain generalisation with a single model remains challenging [442], particularly when domains exhibit substantial differences in terminology, writing style, and relevance patterns. In practice, this creates a trade-off between domain specialisation and scalability: highly specialised models improve retrieval effectiveness, but require maintaining multiple copies of large neural architectures, whereas unified models reduce deployment complexity but often suffer from performance degradation. To mitigate these limitations, Parameter-Efficient Fine-Tuning (PEFT) [155, 45] has emerged as an effective strategy to enable efficient adaptation by updating only a small subset of parameters while keeping the backbone model frozen. Among PEFT methods, adapter-based approaches such as Houlsby adapters [184] and low-rank reparameterisation techniques such as LoRA [187] provide modular mechanisms to introduce domain-specific knowledge with limited parameter overhead. In parallel, model compression techniques such as pruning [386] and knowledge distillation [359] aim to reduce the computational requirements of neural models. Nevertheless, reducing model dimensionality often leads to weak robustness in cross-domain evaluation settings, showing that small models may struggle to preserve domain-specific

representational knowledge [56, 111]. These observations highlight the need for retrieval architectures that support scalable multi-domain adaptation while remaining computationally efficient. In particular, an ideal framework should satisfy three key requirements: (i) maintain strong effectiveness comparable to domain-specific fine-tuning, (ii) avoid storing and updating multiple full retrieval models, and (iii) enable the incremental integration of new domains with minimal retraining cost. To address this limitation, in this Chapter we introduce DRAMA (Domain Retrieval using Adaptive Module Allocation), a modular approach aimed at scalable and energy-efficient multi-domain retrieval. DRAMA trains a single shared dense retriever via knowledge distillation and augments it with lightweight domain-specific adapter modules, enabling domain adaptation without updating the frozen backbone model. During inference, a gating mechanism selects the most suitable adapter for each incoming query, allowing the system to activate domain-specific parameters only when required. This routing strategy is inspired by Mixture-of-Experts (MoE) architectures employing top-1 gating [199]. However, unlike conventional MoE training pipelines—which jointly optimise both experts and router in an end-to-end manner—DRAMA trains its components separately. This separation improves scalability and modularity, as additional domains can be incorporated by training new adapters (and, if necessary, updating the gating classifier) without retraining the underlying retriever. We evaluate DRAMA in both dense bi-encoder and cross-encoder retrieval configurations on two domain-sensitive tasks: academic search and community question answering. DRAMA is compared against domain-specific fine-tuned retrievers as well as unified multi-domain baselines. Experimental results show that DRAMA attains retrieval performance comparable to domain-specialised models, while significantly reducing inference-time energy consumption and associated CO_2 emissions. In addition, DRAMA demonstrates strong robustness when deployed on previously unseen domains, suggesting that its modular allocation strategy facilitates effective generalisation beyond the training distribution. The remainder of this Chapter is structured as follows. Section 6.1 reviews relevant literature on knowledge distillation for information retrieval. Section 6.2 introduces the DRAMA framework and details its architectural components. Section 6.3 describes the experimental setup, including datasets, evaluation metrics, and training procedures. Section 6.4 presents and analyses the experimental findings. Finally, Section 6.5 concludes the Chapter by summarising the main contributions.

6.1 Related Work

In this section, we present related works on dense retrieval, knowledge distillation and Mixture of Experts.

Dense retrieval and domain sensitivity Dense retrieval models, introduced following the success of BERT [103], have become a dominant paradigm in neural information retrieval [218]. Unlike sparse methods such as BM25 [395], dense retrieval models learn continuous representations of queries and documents in a shared embedding space, enabling semantic matching beyond lexical overlap [230]. Despite their effectiveness, dense retrievers are highly sensitive to domain shift. Models trained on a given distribution often experience significant performance degradation when applied to new or heterogeneous domains. Recent work has explored parameter-efficient mechanisms to improve adaptability without fully retraining large models. In particular, adapter-based methods [184] have been successfully integrated into dense retrieval pipelines [292, 215], enabling lightweight domain adaptation while preserving the shared backbone.

Knowledge distillation for efficient adaptation Knowledge distillation (KD) [176] has been widely used to transfer knowledge from large, high-capacity models (teachers) to smaller models (students), enabling efficiency improvements without substantial loss in effectiveness. In information retrieval, KD has been applied to compress large ranking models into more efficient architectures suitable for deployment. Representative models such as DistilBERT [410] and MiniLM [474] show that compact models can retain strong performance while significantly reducing computational cost. In neural retrieval, KD is particularly relevant for domain adaptation, as it allows student models to inherit domain-specific ranking behaviour from specialised teachers. Approaches such as Margin-MSE loss [179], TCT-ColBERT [265], and RocketQA [374] further illustrate how distillation can be used to improve efficiency while preserving retrieval quality. However, while KD improves efficiency at the model level, it does not directly address the scalability issue arising from maintaining multiple domain-specific models.

Mixture-of-Experts and modular adaptation Mixture-of-Experts (MoE) architectures [199, 416] provide a complementary direction for scalable modelling by activating only a subset of specialised components (experts) for each input. This conditional computation paradigm has been widely adopted in large-scale neural systems, including recent LLM architectures such as Mixtral [206]. In the context of domain adaptation, MoE models such as DEMix [150] and Branch-Train-Mix [430] show that separating experts by domain can improve performance while maintaining scalability. These approaches rely on learned routing mechanisms that assign inputs to the most appropriate expert. Although effective, MoE systems are often trained end-to-end, which can make incremental extension to new domains expensive and inflexible. In information retrieval, MoE-inspired designs [424, 426, 425] have been explored for tasks such as question answering [97, 220], showing the potential of modular architectures for domain-sensitive retrieval. Overall, existing work highlights a trade-off between adaptability and scalability: dense retrievers

require domain-specific tuning, KD improves efficiency but not modularity, and MoE improves modularity but often at the cost of training complexity. In contrast, DRAMA combines knowledge distillation, parameter-efficient adapters, and MoE-inspired routing into a unified framework for scalable domain adaptation in neural retrieval. This enables efficient domain specialization without requiring full model duplication or joint re-training of all components.

6.2 Domain Retrieval using Adaptive Module Allocation

We introduce DRAMA (Domain Retrieval using Adaptive Module Allocation), a modular framework designed to enable efficient and scalable domain adaptation for neural retrieval. The core idea is to decouple domain specialization from the backbone retrieval model, allowing lightweight components to capture domain-specific behaviour while maintaining a single shared encoder. This design supports efficient adaptation to multiple domains without increasing the complexity of the underlying retrieval architecture. In the following, we describe the main components of DRAMA: (i) domain-specific adapters, (ii) the domain classification (gating) mechanism, (iii) the inference pipeline, and (iv) the overall parameter efficiency of the system.

6.2.1 Domain-Specific Adapters

DRAMA relies on a frozen dense retrieval encoder $Encoder(\cdot)$, which can be either a Bi-Encoder or Cross-Encoder architecture. Instead of training separate full models for each domain, we introduce lightweight adapter modules that enable domain-specific adaptation. Let $\mathcal{D} = \{d_1, \ldots, d_N\}$ denote the set of different domains. For each domain d_n, we associate a dedicated adapter module A_n with parameters ϕ_n, which modulate the behaviour of the frozen backbone according to domain-specific characteristics. This design enables efficient domain adaptation: rather than fine-tuning a whole model per domain, DRAMA only learns a small set of additional parameters. As a result, adding a new domain requires training only a new adapter module, without modifying or retraining the shared encoder. Each adapter is trained independently using knowledge distillation from a domain-specific teacher retriever. For each query–document pair in the training data, the teacher model produces a similarity score reflecting the relevance of the document to the query. The distillation objective is to make the student retriever to match the behaviour of the domain-specific teacher by minimising a Mean Squared Error (MSE) loss [387] between the teacher's score and the student's predicted score (denoted as the DRAMA score in Figure 6.1). In addition, we leverage the available ground-truth relevance labels to directly optimise the student model using a

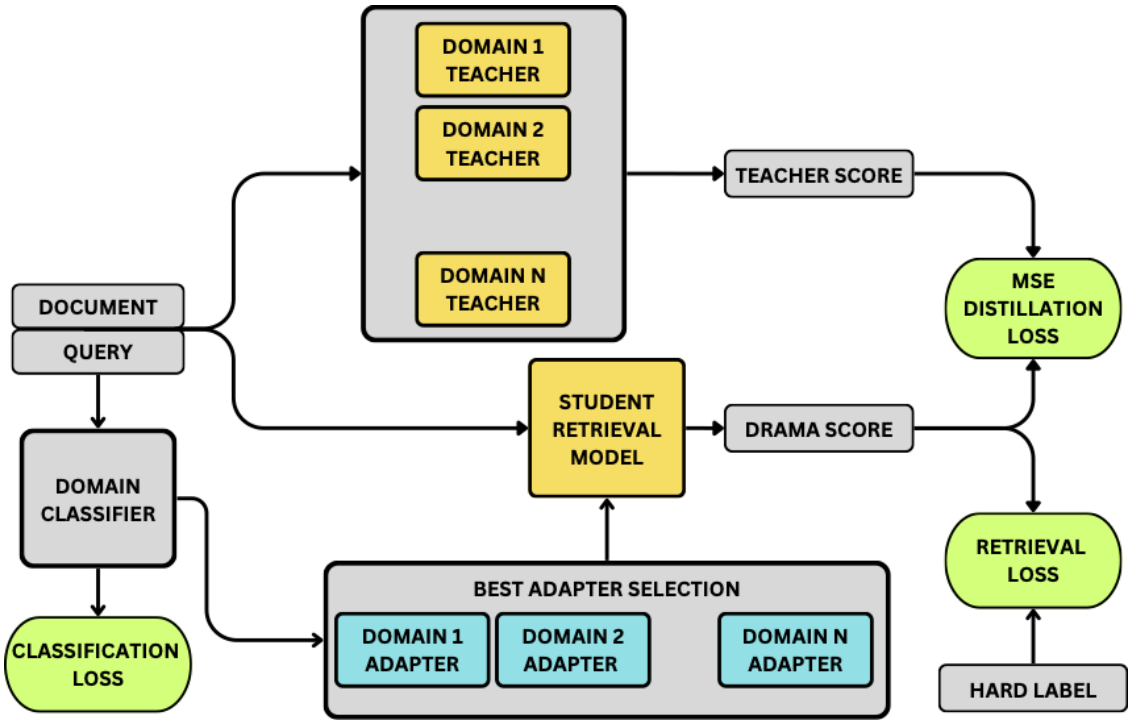

Figure 6.1: Overview of the DRAMA training procedure.

triplet margin loss [17] (Retrieval Loss in Figure 6.1). This training strategy allows DRAMA to benefit jointly from the original supervision signal and from the teacher model's guidance. Leveraging both teacher-generated similarity scores and explicit relevance labels is a common practice in knowledge distillation for information retrieval [374].

6.2.2 Domain Classifier (Gating Function)

To support adaptive behaviour at inference time, DRAMA incorporates a lightweight gating component that predicts the most appropriate domain for a given query. Concretely, this module functions as a domain classifier defined as:

$$G(q) = \arg\max_{n} P(d_n \mid q), \tag{6.1}$$

where $P(d_n \mid q)$ is computed using the shared encoder backbone followed by a compact classification head. This gating mechanism is essential for enabling conditional computation. Rather than evaluating all domain-specific adapters, DRAMA activates only the adapter associated with the predicted domain, thereby reducing inference-time computational cost while retaining the benefits of domain-specific specialisation. Unlike classical Mixture-of-Experts architectures, where experts and

gating networks are jointly optimised, DRAMA decouples the training of the classifier from the adapter training process. The classifier is trained independently using query-domain labels via a cross-entropy objective. This decoupled training strategy enhances modularity and facilitates extensibility, since additional domains can be supported by training a new adapter and updating the domain classifier, without requiring retraining of the entire retrieval model.

6.2.3 Inference Pipeline

During inference, DRAMA operates through a two-stage adaptive procedure aimed at maximising efficiency and scalability. Given an input query q, the first step consists of applying the gating function to identify the most probable domain:

$$d_n = G(q). \tag{6.2}$$

Then, only the corresponding adapter A_n is activated within the frozen encoder $Encoder(\cdot)$. The query and candidate documents are then encoded using this domain-specific configuration, and relevance scores are computed. This conditional activation mechanism ensures that only a small fraction of parameters is used per query, while still enabling domain-specific behaviour. Importantly, the backbone model remains unchanged across all domains, guaranteeing consistent behaviour and simplifying deployment.

6.2.4 Parameter Efficiency

A key objective of DRAMA is to achieve scalable domain adaptation without increasing the cost of deployment. This is achieved through the use of lightweight adapters and a frozen backbone. Let P denote the number of parameters of the shared encoder and A the size of a single adapter, where $A \ll P$. For N domains, DRAMA requires only:

$$P + N \cdot A + A_{gate} \tag{6.3}$$

parameters, where A_{gate} corresponds to the small gating classifier. By contrast, deploying N separate dense retrievers would require $N \cdot P$ parameters, leading to a linear growth in memory and storage requirements with respect to both model size and the number of supported domains. DRAMA avoids this duplication and thus yields substantial reductions in storage footprint, memory usage, and computational cost. This modular structure further enhances scalability. Additional domains can be supported by training a single new adapter module, without modifying or retraining the existing components. Consequently, DRAMA enables continual extension to new domains in multi-domain retrieval scenarios while preserving an efficient and stable deployment profile.

6.3 Experimental Settings

The experimental evaluation is designed to address the following research questions:

1. **RQ0**: Do dense retrievers fine-tuned on individual domains outperform a single unified model trained jointly on data from multiple domains?
2. **RQ1**: Does DRAMA improve retrieval effectiveness over a unified baseline by dynamically selecting domain-specific parameters based on the input query?
    - **(a)** How accurately can the gating mechanism predict the correct domain at inference time?
    - **(b)** To what extent does DRAMA generalise to previously unseen domains?
3. **RQ2**: Does DRAMA provide substantial reductions in energy consumption and storage requirements?

In this section, we describe the datasets, model configurations, baselines, and evaluation setuo adopted to assess the effectiveness and efficiency of DRAMA. Our experimental design aims to evaluate multi-domain retrieval under heterogeneous domain structures, while also analysing scalability and generalisation to unseen domains.

Datasets Our experiments rely on two multi-domain retrieval benchmarks that differ in size and domains. Table 6.1 reports the main characteristics of the datasets, including the number of queries and documents, as well as the distribution of domain annotations. The first benchmark is an academic search collection introduced by [28], which comprises four scientific domains, each containing roughly five million documents. The dataset is designed to approximate realistic academic retrieval scenarios. Queries are constructed from paper titles, and relevance labels are derived from citation links: for each query paper, the documents cited by the paper are judged as relevant. To prevent temporal leakage, the dataset is split into training and test splits based on publication dates, such that test queries are associated with papers published after those included in the training set. The second benchmark is SE-PQA, a community question answering (cQA) dataset constructed from Stack Exchange [226]. The full dataset spans 50 communities, each representing a distinct topical domain. Since our study targets multi-domain retrieval, we restrict the evaluation to communities with at least 50,000 training queries. This selection results in four domains—SciFi, Gaming, English, and Apple—containing approximately 52k, 83k, 100k, and 86k training questions, respectively. In this setting, user questions serve as queries, while relevant documents correspond to answers

	Academic Search		SE-PQA (cQA)	
	Computer Science	***Physics***	***Apple***	***English***
# documents	4 809 684	4 926 753	173 990	265 100
# train queries	552 798	728 171	86 386	100 543
# val queries	5 583	7 355	6 687	6 644
# test queries	6 497	6 366	7 367	6 384
avg. relevants	3.24	4.17	1.27	1.79
	Political Science	***Psychology***	***Gaming***	***Scifi***
# documents	4 814 084	4 215 384	151 362	118 416
# train queries	162 597	544 882	83 281	52 256
# val queries	1 642	5 503	3 672	3 661
# test queries	5 715	12 625	4 071	6 046
avg. relevants	3.88	4.73	1.24	1.36

Table 6.1: Statistics for the benchmark datasets used to train and evaluate DRAMA.

that received positive feedback from the community. These communities exhibit strong topical separation and distinct linguistic patterns, making the dataset particularly suitable for evaluating domain routing and specialisation. Taken together, the two datasets represent complementary multi-domain retrieval scenarios. The academic benchmark contains partially overlapping domains and more ambiguous query-domain boundaries, whereas SE-PQA is characterised by clearly separated domains and highly domain-specific vocabulary. Importantly, both datasets provide query-level domain labels, enabling controlled evaluation of domain-specialised adaptation strategies. To assess generalisation beyond the training domains, we additionally evaluate DRAMA in a zero-shot setting on two BEIR benchmarks [442], i.e. SCIDOCS [92] and Quora [442]. These datasets are selected because they align with the two retrieval scenarios considered during training. In particular, SCIDOCS addresses citation prediction, which is closely related to academic search tasks, whereas Quora focuses on community question answering, making it comparable to SE-PQA.

Model Details and Baselines To evaluate DRAMA under different retrieval paradigms and compute constraints, we consider multiple backbone encoders and training configurations, and compare against both domain-specific and multi-domain baselines.

We conduct experiments using both Bi-Encoder and Cross-Encoder architectures. Evaluating both architectures allows us to assess whether DRAMA remains

effective across different retrieval settings. For the academic search benchmark, we consider two model configurations. First, we use DistilBERT [410] (66.4M parameters) as student and BERT-base [103] (110M parameters) as teacher, and evaluate this setting and Bi-Encoder architecture. Second, we use MiniLM [474] (22.7M parameters) as both student and teacher, and evaluate it under both Bi-Encoder and Cross-Encoder settings. For the SE-PQA benchmark, we adopt MiniLM as both the teacher and student model, and we evaluate both Bi-Encoder and Cross-Encoder retrieval configurations. For each dataset, we consider three primary baseline systems. First, we train a unified multi-domain model (ALL) by fine-tuning a single retriever on the combined training data from all domains, without explicitly distinguishing domain boundaries. This model represents the standard unified approach to multi-domain retrieval. Second, we train specialised domain-specific models (S), obtained by fine-tuning the same backbone independently on each domain. These models provide a reference point for the performance achievable through standard domain-specific adaptation. Third, we train teacher models (T), which are fine-tuned independently on each domain and used as supervision sources for knowledge distillation. Depending on the experimental configuration, the teacher can share the same architecture as the student (e.g. MiniLM) or correspond to a larger backbone (e.g. BERT-base distilling into DistilBERT).

Building on the general multi-domain backbone, DRAMA introduces domain-specific parameter-efficient modules trained via knowledge distillation from the corresponding teachers. In this setting, the encoder remains frozen and only the adapter parameters are updated. To show that DRAMA can be seamlessly integrated with different parameter-efficient fine-tuning approaches, we evaluate two alternative techniques: (i) Houlsby bottleneck adapters [184] with reduction factor 4, and (ii) LoRA modules [187] with rank 8. Adapters are trained with a learning rate of 10^{-4} in the DistilBERT and BERT-base configuration, and with a learning rate of $5 \cdot 10^{-7}$ in all MiniLM-based experiments. All models are trained for 40 epochs with batch size 128 and learning rate $2 \cdot 10^{-5}$. The margin of the ranking loss is fixed to 0.1 across all experiments. All results are reported in a re-ranking scenario. BM25 is used to retrieve an initial set of candidate documents, with its hyperparameters optimised on the validation data. Subsequently, neural ranking models are applied to re-rank the top-k retrieved candidates. We set $k = 1000$ for academic search and $k = 100$ for SE-PQA, following the original benchmark configurations [28, 226]. Finally, the gating classifier responsible for domain routing is trained independently from the adapters. Across all datasets and backbones, it is trained for 10 epochs with learning rate 10^{-4}.

Evaluation Metrics Retrieval performance is measured using three standard ranking metrics: Mean Average Precision at 100 (MAP@100), Mean Reciprocal Rank at 10 (MRR@10), and Normalised Discounted Cumulative Gain at 10

(NDCG@10). Statistical significance is evaluated via a Bonferroni-adjusted two-sided paired Student's t-test with a 99% confidence level.

6.4 Results

This section reports the experimental results obtained across both the academic search and the community question answering benchmarks. Our analysis is structured around the research questions introduced in Section 6.3, and aims to assess both the retrieval effectiveness and the scalability of DRAMA under different retrieval architectures and model sizes.

Results for the academic search benchmark are reported in Tables 6.2, 6.3, and 6.4, while results for SE-PQA are reported in Tables 6.5 and 6.6.

Model	Computer Science			Physics			Political Science			Psychology		
	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10
BM25	.123	.489	.225	.128	.537	.269	.133	.502	.241	.126	.512	.239
Full Model												
BERT_{T}	.203	.609	.314	.207	.657	.364	.190	.589	.303	.231	.660	.358
$\text{DistilBERT}_{\text{S}}$	.193	.600	.304	.200†	.650†	.356†	.186	.588	.300	.219	.644	.344†
$\text{DistilBERT}_{\text{ALL}}$	.190	.595	.300	.190	.637	.345	.191	.589	.306	.211	.638	.337
Knowledge Distillation (Oracle)												
$\text{DistilBERT}_{\text{LoRA}}$	.197	.602	.307	.190	.638	.345	.197	.597	.312	.219	.648	.345
$\text{DistilBERT}_{\text{Houlsby}}$	.204	.609	.315	.196	.646	.351	.201	.603	.316	.224	.651	.350
DRAMA												
$\text{DistilBERT}_{\text{LoRA}}$	.197*†	.602	.307†	.190	.637	.345	.196*†	.596†	.312*†	.219†	.648†	.345†
$\text{DistilBERT}_{\text{Houlsby}}$	.204*†	.609†	.314*†	.196†	.647†	.352†	.202*†	.605*†	.319*†	.223*†	.650†	.349*†

Table 6.2: Results of the DistillBERT BiEncoder model on the academic search dataset. The symbols $*$ and † denote significant improvements over S and ALL, respectively.

Model	Computer Science			Physics			Political Science			Psychology		
	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10
BM25	.123	.489	.225	.128	.537	.269	.133	.502	.241	.126	.512	.239
Full Model												
MiniLM_{S}	.193	.597	.300	.182	.626	.335	.184	.578	.295	.218	.647	.342
$\text{MiniLM}_{\text{ALL}}$	.189	.593	.292	.178	.620	.330	.191	.590	.304	.211	.636	.335
Knowledge Distillation (Oracle)												
$\text{MiniLM}_{\text{LoRA}}$	.190	.595	.294	.181	.624	.332	.193	.592	.306	.215	.642	.341
$\text{MiniLM}_{\text{Houlsby}}$	.193	.597	.300	.182	.626	.335	.194	.592	.307	.216	.644	.341
DRAMA												
$\text{MiniLM}_{\text{LoRA}}$	.190	.595†	.293	.181†	.624	.332	.192	.592†	.305	.214	.642†	.341†
$\text{MiniLM}_{\text{Houlsby}}$	.193†	.595†	.300†	.182†	.627†	.335†	.194*†	.592†	.307*†	.216†	.644†	.341†

Table 6.3: Results of the MiniLM BiEncoder model on the academic search dataset. Notation for * and † as per Table 6.2.

Model	Computer Science			Physics			Political Science			Psychology		
	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10
BM25	.123	.489	.225	.128	.537	.269	.133	.502	.241	.126	.512	.239
Full Model												
MiniLM_{S}	.177†	.589†	.289†	.174†	.624†	.328†	.184†	.584†	.296†	.213†	.644†	.339†
$\text{MiniLM}_{\text{ALL}}$	.163	.564	.273	.155	.600	.307	.177	.579	.292	.186	.615	.312
Knowledge Distillation (Oracle)												
$\text{MiniLM}_{\text{LoRA}}$	.164	.567	.276	.159	.605	.311	.177	.579	.292	.190	.624	.318
$\text{MiniLM}_{\text{Houlsby}}$	.177	.589	.289	.172	.620	.322	.181	.584	.296	.207	.640	.335
DRAMA												
$\text{MiniLM}_{\text{LoRA}}$	.164†	.567†	.276†	.158†	.604†	.311†	.177	.581	.293	.189†	.622†	.316†
$\text{MiniLM}_{\text{Houlsby}}$	.174†	.584†	.283†	.171†	.618†	.320†	.182†	.584†	.296†	.202†	.635†	.332†

Table 6.4: Results of the CrossEncoder model on the academic search dataset. Notation for * and † as per Table 6.2.

Model	Apple			English			Gaming			Scifi		
	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10
BM25	.250	.272	.276	.466	.544	.506	.437	.463	.474	.480	.516	.521
Full Model												
MiniLM_{S}	.378†	.414†	.413†	.606†	.691†	.644†	.592†	.626†	.627†	.637†	.685†	.677†
$\text{MiniLM}_{\text{ALL}}$	.371	.407	.406	.600	.685	.639	.580	.613	.615	.624	.672	.664
Knowledge Distillation (Oracle)												
$\text{MiniLM}_{\text{LoRA}}$	.359	.393	.396	.595	.677	.635	.577	.609	.613	.634	.680	.675
$\text{MiniLM}_{\text{Houlsby}}$	.373	.409	.409	.606	.691	.644	.589	.622	.624	.641	.689	.680
DRAMA												
$\text{MiniLM}_{\text{LoRA}}$	.371	.406	.407	.595	.677	.635	.582	.615	.617	.633†	.681†	.673†
$\text{MiniLM}_{\text{Houlsby}}$	.373	.408	.409	.606†	.690†	.644†	.589†	.622†	.624†	.641†	.689†	.680†

Table 6.5: Results of the BiEncoder model on the cQA dataset. Notation for * and † as per Table 6.2.

6.4.1 RQ0: Do dense retrievers fine-tuned on individual domains outperform a single unified model trained jointly on data from multiple domains?

We investigate whether domain-specific dense retrievers outperform a single model trained jointly on data from multiple domains. To investigate this question, we evaluate domain-specific models (S), each trained independently on a single domain, and compare them against a unified model (ALL) trained on the entire multi-domain dataset without any explicit domain distinction. Across nearly all domains and architectures, the results show a consistent performance gap between specialised and general training. In particular, the general model systematically underperforms compared to the specialised baselines, confirming that a unified retriever struggles to capture domain-specific relevance signals when trained on heterogeneous data distributions. This behaviour is expected, since different domains often exhibit distinct lexical conventions, semantic structures, and relevance patterns, which can introduce conflicting gradients during joint training. The only exception is observed in the Political Science domain of the academic search benchmark, where the unified model slightly outperforms the domain-specific model. This

Model	Apple			English			Gaming			Scifi		
	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10
BM25	.250	.272	.276	.466	.544	.506	.437	.463	.474	.480	.516	.521
Full Model												
MiniLM_{S}	.300†	.328†	.333†	.536†	.621†	.579†	.515†	.545†	.554†	.540†	.583†	.586†
$\text{MiniLM}_{\text{ALL}}$	.290	.318	.323	.523	.608	.567	.503	.532	.542	.524	.566	.570
Knowledge Distillation (Oracle)												
$\text{MiniLM}_{\text{LoRA}}$	.297	.328	.330	.531	.616	.573	.505	.535	.544	.516	.556	.561
$\text{MiniLM}_{\text{Houlsby}}$	.307	.338	.340	.543	.628	.585	.512	.543	.553	.541	.583	.586
DRAMA												
$\text{MiniLM}_{\text{LoRA}}$	.297†	.328†	.330†	.531†	.616†	.573†	.505	.535	.544	.515	.556	.561
$\text{MiniLM}_{\text{Houlsby}}$	.307*†	.338*†	.340*†	.543*†	.627*†	.585*†	.513†	.543†	.553†	.540†	.582†	.586†

Table 6.6: Results of the CrossEncoder model on the cQA dataset. Notation for * and † as per Table 6.2.

behaviour can be attributed to two factors. First, Political Science is the smallest domain in the dataset, and the limited supervision reduces the effectiveness of isolated fine-tuning. Second, queries in this domain exhibit stronger cross-domain overlap, suggesting that the corresponding documents may not be strictly domain-specific. We further analyse this behaviour in the discussion on routing performance. Overall, in response to **RQ0**, the results show that fine-tuning models on individual domains leads to higher retrieval performance compared to training a single model jointly on multiple domains.

6.4.2 RQ1: Does DRAMA improve retrieval effectiveness over a unified baseline by dynamically selecting domain-specific parameters based on the input query?

To answer RQ1, we evaluate two variants of our approach. First, we consider a distilled oracle setting (*KD*), where the correct domain is provided at inference time and the corresponding adapter is activated. Second, we evaluate the full DRAMA configuration, where the domain is predicted automatically using the gating classifier. In both experimental settings, we evaluate two adapter-based parameterisations, i.e. Houlsby adapters and LoRA updates, in order to assess the robustness of DRAMA across different PEFT strategies. We first present the results on the academic search benchmark in Tables 6.2, 6.3, and 6.4. Using DistilBERT as the student and BERT-base as the teacher, the distilled oracle model (*KD*) attains performance comparable to the teacher and, in certain cases, slightly exceeds it. This indicates that the domain-specific ranking behaviour encoded by the teacher can be successfully transferred into lightweight adapter components, even when operating with a smaller backbone model. When the gating mechanism is incorporated, both $\text{DRAMA}_{Houlsby}$ and DRAMA_{LoRA} remain within roughly 2% MAP@100 of the teacher, while consistently outperforming both the domain-specific student baseline (*S*) and the unified multi-domain model (*ALL*). On average, DRAMA improves

MAP@100 by around 3% over the general model, suggesting that allocating parameters in a domain-aware manner helps mitigate the negative effects of heterogeneous training data distributions. The gating classifier achieves an overall accuracy of 89%, performing particularly well on Computer Science, Physics, and Psychology, while showing reduced accuracy on Political Science. This pattern aligns with the partial overlap among academic domains and the relatively smaller size of the Political Science training data. We further evaluate DRAMA using MiniLM as both teacher and student. In this setting, both DRAMA variants consistently outperform the baseline models while remaining close to the specialised per-domain models, with a parameter overhead below 4% per domain. In the Cross-Encoder setting, DRAMA surpasses the unified baseline across all domains, yielding MAP@100 gains between 1.7% and 2.9%. The small performance gap with respect to the oracle KD model is mainly attributed to reduced domain classification accuracy, which drops to approximately 80We then evaluate DRAMA on the SE-PQA benchmark using MiniLM as both teacher and student, with results reported in Tables 6.5 and 6.6. In the Bi-Encoder configuration, the gating module achieves 99% overall accuracy, with per-domain precision ranging between 98% and 99%. This is expected given that the selected Stack Exchange communities correspond to clearly distinct domains with strong linguistic separation. Consequently, DRAMA performs very close to domain-specific fine-tuning. In particular, DRAMA$_{Houlsby}$ remains within 0.3% MAP@100 of the specialised models while improving over the unified baseline by 1.6% on average. DRAMA$_{LoRA}$ also improves over the general model, albeit to a lesser extent, indicating that bottleneck adapters provide stronger capacity for domain adaptation in this scenario. In the Cross-Encoder setting, DRAMA$_{Houlsby}$ matches the performance of specialised models and improves MAP@100 by 3.6% over the unified baseline. DRAMA$_{LoRA}$ yields improvements primarily on the Apple and English domains, while providing weaker gains on SciFi and Gaming. These results show that LoRA may be less effective at capturing fine-grained domain-specific ranking behaviour compared to bottleneck adapters. Overall, to answer RQ1, DRAMA substantially improves retrieval effectiveness compared to a unified multi-domain model, and achieves performance close to domain-specific fine-tuning and the oracle KD configuration, while maintaining a scalable shared backbone.

How accurately can the gating mechanism predict the correct domain at inference time? RQ1(a) analyses whether DRAMA's performance gains depend on the ability of the gating classifier to select the correct domain adapter. To disentangle the specific effect of domain routing, we consider a random routing baseline (RND), in which the adapter is selected uniformly at random for each query during inference. Results in Table 6.7 show that random routing leads to a consistent performance drop. Indeed, DRAMA significantly outperforms the random baseline, confirming that accurate domain classification is essential for effective parameter allocation. This behaviour shows that the improvements achieved by DRAMA do

not arise simply from adding adapters, but depend critically on selecting the appropriate domain-specific module. We further analyse the weaker precision on Political Science in the academic dataset, where precision is 63%. This domain contains the smallest training set and exhibits substantial topical overlap with other scientific areas. We manual inspect misclassified queries, such as "Grazing behavior and associations with obesity, eating disorders, and health-related quality of life in the Australian population", which is classified under Political Science but is categorised as Psychology by our classifier. This analysis confirms that many examples contain interdisciplinary terminology, indicating that routing errors often reflect genuine domain ambiguity rather than classifier failure. Importantly, these misclassifications do not substantially harm retrieval performance, suggesting that adapters retain a degree of robustness to domain overlap. Overall, to answer **RQ1(a)**, the domain classifier is a critical component of DRAMA, and replacing it with random routing yields a substantial decrease in retrieval effectiveness.

To what extent does DRAMA generalise to previously unseen domains? **RQ1(b)** investigates whether DRAMA generalises effectively to domains that were not observed during training. To assess this scenario, we build an unseen-domain test set from SE-PQA by combining queries from communities that are excluded from training, yielding 58,190 validation queries and 76,010 test queries. Since these domains are not observed during training, this setup provides a strict evaluation of out-of-distribution generalisation. In this analysis, we focus on the Bi-Encoder MiniLM configuration with DRAMA$_{Houlsby}$, as it achieved the best performance in the in-domain experiments. We compare it against the unified baseline (MiniLMALL) and the random routing variant (MiniLM$_{RND}$). As shown in Table 6.8, DRAMA consistently surpasses both baselines on unseen-domain queries. Specifically, it improves MAP@100 by 1.7% and 3.3% in the Bi-Encoder and Cross-Encoder settings, respectively, relative to the general model. These results show that DRAMA is able to exploit the modular adapter structure even when the target domain does not correspond directly to one of the trained adapters. This suggests that the adapter pool captures transferable domain features that can still provide meaningful benefits under distribution shift. To further assess generalisation, we evaluate DRAMA in a zero-shot setting on BEIR. We apply the model trained on academic search to SCIDOCS, and the model trained on SE-PQA to Quora and we report the results in Table 6.9. On SCIDOCS, DRAMA achieves performance comparable to the unified baseline, while consistently outperforming random routing. On the Quora benchmark, DRAMA consistently outperforms BM25 and achieves statistically significant improvements over the random baseline, even in the presence of a clear mismatch between the training and target tasks. Overall, with respect to **RQ1(b)**, these results indicate that DRAMA generalises effectively to previously unseen domains, preserving strong retrieval performance without the need for additional retraining or task-specific fine-tuning.

Model Variant	MAP@100	MRR@10	NDCG@10
DRAMA$_{Houlsby}$	**.313**	**.594**	**.414**
Random Gating (RND)	.312	.582	.370

Table 6.7: Ablation study on the routing mechanism in the Bi-Encoder MiniLM model.

Model	Bi-Encoder			Cross-Encoder		
	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10
BM25	.412	.503	.423	.412	.503	.423
MiniLM$_{\text{ALL}}$	.515	.618	.569	.423	.515	.474
MiniLM$_{\text{RND}}$	.515	.618	.569	.423	.515	.474
DRAMA-MiniLM$_{\text{Houlsby}}$	**.524***	**.627***	**.578***	**.437***	**.530***	**.489***

Table 6.8: Results on out-of-domain queries in the cQA dataset. The symbol * denotes significant improvements over RND.

6.4.3 RQ2: Does DRAMA provide substantial reductions in energy consumption and storage requirements?

RQ2 investigates whether DRAMA can lower inference-time computational cost and storage demands in comparison to deploying multiple separate domain-specific models. We report the results of our analysis in Table 6.10. We assume a query length of 128. From a storage perspective, measured in terms of total trainable parameters, DRAMA attains performance comparable to an ensemble of domain-specific models while requiring only about 112–120% of the parameters of the base model. By contrast, maintaining four independent models would increase the parameter cost to roughly 400% of the base model size. This highlights that DRAMA achieves substantial memory savings while preserving the benefits of domain-specific modelling. In terms of energy consumption, a multi-model ensemble scales linearly with the number of domains, since each model must be executed separately at inference time. Conversely, DRAMA performs a single routing step and activates only one adapter per query, leading to inference costs that remain constant regardless of the number of domains. As a result, DRAMA reduces inference-time energy usage and associated carbon emissions by more than 75% compared to a multi-model setup, while maintaining similar retrieval effectiveness. Beyond efficiency, DRAMA addresses a fundamental scalability limitation of multi-domain retrieval. In conventional approaches, supporting additional domains requires storing and deploying additional full retrievers, leading to linear growth in hardware and maintenance requirements. In contrast, DRAMA maintains a single shared backbone and introduces only lightweight domain adapters. These adapters are

Model	SCIDOCS			Quora		
	MAP@100	MRR@10	NDCG@10	MAP@100	MRR@10	NDCG@10
BM25	.108	.277	.158	.729	.760	.770
$\text{MiniLM}_{\text{ALL}}$	.128	.315	.183	.812	.840	.848
$\text{MiniLM}_{\text{RND}}$	.123	.308	.178	.794	.825	.831
$\text{DRAMA-MiniLM}_{\text{Houlsby}}$	.125	.312	.180	.795*	.826*	.832*

Table 6.9: Results on BEIR datasets. Notation as in Table 6.5.

Model	Params (M)	Total GFLOP/query	Energy/query (J)	CO_2/query (mg)
BERT_{S}	440	89.40	1.40	0.16
$\text{DistilBERT}_{\text{S}}$	264	44.72	0.70	0.08
$\text{DRAMA-DistilBERT}_{\text{LoRA}}$	0.73	0.20	0.003	0.0003
$\text{DRAMA-DistilBERT}_{\text{Houlsby}}$	80.6	15.68	0.25	0.028
MiniLM_{S}	90.8	11.48	0.18	0.020
$\text{DRAMA-MiniLM}_{\text{LoRA}}$	0.36	0.15	0.002	0.0002
$\text{DRAMA-MiniLM}_{\text{Houlsby}}$	26.3	4.02	0.06	0.007

Table 6.10: Per query energy and carbon costs in an ensemble setting.

sufficiently compact to be stored jointly with the backbone model, enabling multi-domain deployment without replicating the entire retrieval architecture. Overall, results provide a clear answer to **RQ2**: DRAMA substantially reduces storage and inference cost compared to maintaining multiple domain-specific models, while preserving competitive retrieval performance, making it a scalable and energy-aware framework for multi-domain retrieval.

6.5 Conclusions

In this Chapter we have presented DRAMA, a modular and parameter-efficient framework for domain-adaptive neural information retrieval. The approach integrates lightweight domain-specific adapters with a query-dependent gating mechanism, allowing dynamic selection of specialised components within a single shared retrieval backbone. Across both academic search and community question answering benchmarks, DRAMA achieves retrieval effectiveness comparable to fully specialised domain-specific models, while requiring only a small additional parameter footprint. This substantially reduces storage requirements and inference-time computational cost, resulting in lower energy consumption and improved scalability in multi-domain retrieval settings.

Part III

Task Arithmetic for Zero-Shot Domain Adaptation

Chapter 7

Introduction

Traditional approaches to improving information retrieval systems and large language models often rely on supervised fine-tuning, which requires labelled data, which are not only avaliable in specific domains due to the high cost of human annotation. While efficient adaptation methods such as adapters reduce the cost of training, they still assume the availability of task and domain-specific data. Recently, a new line of research has shown that model behaviour can be manipulated in a training-free manner through linear operations in parameter space. In this part of the thesis, we move beyond efficient fine-tuning and explore lightweight, training-free strategies for information retrieval. We focus on two complementary perspectives: first, the use of task arithmetic to directly manipulate model behaviour without additional training; and second, the analysis of task vectors to better understand what large models learn, leading to a new efficient architecture derived from this interpretation.

Chapter 8 introduces the use of task arithmetic for information retrieval, extending the idea of task vectors to ranking and retrieval models. In this framework, different tasks are represented as weight deltas, called task vectors, obtained from independently fine-tuned models, and these vectors can be combined through simple arithmetic operations such as addition and subtraction. This enables controlled behavioural transfer between tasks without any additional training. In the context of IR, this approach allows us to merge or interpolate retrieval behaviours, for example, combining model trained to learn query-document relevance with model trained as language models on specific domains.

Chapter 9 builds on this idea by analysing task vectors more deeply in the context of MonoT5, a strong neural reranking model based on encoder-decoder architectures. By studying how model weights change after task-specific fine-tuning, we investigate which parameters contribute most significantly to task adaptation. This analysis reveals that only a small subset of parameters, particularly those associated with prompt token embeddings, undergo the highest changes across all the parameters. Based on this observation, we propose an interpretation of what

monoT5 learns during fine-tuning, showing that much of its knowledge about query-document relevance can be attributed to localized modifications in embedding space rather than full fine-tuning. Inspired by this insight, we introduce Light-MonoT5, a new efficient model that required the training of only the parameters that exhibit the largest changes in the task vectors, i.e. prompt token embeddings. This targeted training strategy preserves most of the model's pre-trained knowledge while significantly reducing computational cost. Despite its simplicity, Light-MonoT5 retains strong ranking performance, both on in-domain an out-of-domain benchmarks, showing that carefully selected parameter updates can approximate the behaviour of fully fine-tuned models.

This part is organized as follows: Chapter 8 presents a training-free framework for information retrieval based on task arithmetic, showing how linear operations in parameter space can be used to combine and transfer retrieval behaviours without additional training. Chapter 9 provides an in-depth analysis of task vectors in MonoT5, offering an interpretation of its fine-tuning dynamics and introducing Light-MonoT5, an efficient model that updates only a selected subset of parameters identified through task vector analysis.

Chapter 8

Task Vector for Zero-Shot Information Retrieval

Pre-trained Language Models (PLMs) have achieved state-of-the-art performance across a wide range of natural language processing and Information Retrieval (IR) tasks [540]. These models are typically pre-trained on large-scale unlabelled corpora using masked or causal language modelling, and subsequently fine-tuned on labelled datasets for downstream applications. In the IR field, a common practice is to fine-tune PLMs on MS MARCO [328], a large-scale benchmark where the task is to determine the relevance of passages to given queries. This approach has driven significant advances in both bi-encoder and cross-encoder retrieval paradigms. Recent studies [442, 453, 466] have shown that domain mismatch (characterised by different vocabulary, style, or content distribution) remains a significant challenge for both dense, learned sparse and cross-encoder retrieval models, especially when compared to lexical methods. Although parameter-efficient fine-tuning on target datasets with gold labels [141] or automatically generated labels [99] is widely adopted, the annotation process remains time-consuming and expensive. This limits its scalability and makes it difficult to obtain large-scale labeled datasets for training neural retrieval models. For these reasons, this Chapter focuses on approaches that extend beyond efficient fine-tuning. Recently, *Task Arithmetic* [196] has been proposed as a promising alternative to conventional fine-tuning. By merging the parameters of two or more PLMs, Task Arithmetic transfers domain knowledge into a target model without requiring additional training or labelled data. In detail, this approach relies on the *task vector*, which is defined by subtracting the parameters of a PLM from those of a domain-fine-tuned version of it. In other words, a task vector is the difference between model weights resulting from the fine-tuning process, and represents the knowledge that the model has learned while training on the domain-specific data. By performing simple arithmetic operations, such as addition or subtraction, between pre-trained models and task vectors, Task

Arithmetic allows us to compose, transfer, or even remove domain-specific knowledge into Language Models. Task Arithmetic has shown promising performance across a wide range of Natural Language Processing (NLP) tasks, including Sentiment Classification [196] and Summarization [87].

However, the application of Task Arithmetic to different retrieval pipelines and paradigms remains unexplored. Therefore, in this Chapter, we perform a study of Task Arithmetic for domain adaptation in IR on three different paradigms, i.e. Dense, Learned Sparse and Cross-Encoder models. Each retrieval paradigm relies on a different PLM, which is fine-tuned on a large corpus, usually MS MARCO, for learning the relevance of passages to given queries. We define the domain task vector as the difference by subtracting the parameters of the PLM to its counterpart trained as Masked Language Model (MLM) or Language Model (LM) on a large domain corpus. We rely on PLM, its domain-fine-tuned version and ranking models that are publicly available, in such a way that our approach remains reproducible and does not require additional training, thereby minimising carbon footprint. We evaluate Task Arithmetic on six different PLMs spanning seventeen retrieval paradigms (11x Dense, 3x Learned Sparse models, 3x Cross-Encoders) on four scientific and biomedical benchmarks from BEIR [35]. The results of our evaluation show that the performance of Task Arithmetic consistently improves the baseline models fine-tuned on MS MARCO in three out of four datasets, with gains of up to 24% in $nDCG@10$. Furthermore, we investigate the application of Task Arithmetic in a multilingual scenario, showing that domain-specific models defined through task arithmetic outperform a model trained on multiple languages. Finally, we thoroughly investigate the impact that each model and Task Vector have on the retrieval performance.

The contributions of this work are as follows:

- We provide the first in-depth analysis of Task Vectors for retrieval adaptation across Dense, Learned Sparse and Cross-Encoder paradigms.
- We show, through extensive experiments, that Task Arithmetic improves effectiveness in scientific, biomedical retrieval domains as well as in multilingual scenario and it offers a simple and training-free strategy for domain adaptation.
- We further analyse the behaviour of Task Vectors to better understand how they affect retrieval effectiveness across different models and domains.

The rest of the Chapter is organised as follows: Section 8.1 provides an overview of Task Arithmetic and its application, Section 8.2 describes in detail Task Arithmetic and how we apply it to retrieval models. Next, Section 8.3 outlines the experimental setup used to evaluate our approach, Section 8.4 analyses the results of our evaluation and finally, Section 8.5 contains final remarks.

8.1 Related Work

This section provides a general overview of Task Arithmetic and its application.

A complementary line of research explores model merging as a lightweight alternative to multi-task learning [485], leveraging the observation that models fine-tuned on related tasks may reside in compatible regions of parameter space [8]. While multi-task learning requires access to training data for each target task [526], model merging combines independently fine-tuned models directly at the parameter level, making it attractive in low-resource scenarios where labelled data are scarce [196, 212]. Early approaches [196, 485] focused on averaging the parameters of the models. For example, average merging [485] defines a model with competitive performance across different tasks by simply averaging weights across fine-tuned models. Building on this idea, Task Arithmetic [196] introduces scaling factors that emphasise the relative importance of different task-specific models. We argue that Task Arithmetic emerges as interesting because it allows the transfer of task-related knowledge between models without any further training. In detail, given a pre-trained language model and its fine-tuned version on a specific task, Task Arithmetic is based on the differences in parameters between the fine-tuned version and the pre-trained language model, which are referred to as task vectors. The task vector represents the knowledge learned by the model while fine-tuning it on a specific task and adding or subtracting this vector transfers the knowledge without any additional training.

8.2 Task Vectors for Information Retrieval

In the following, we define Task Arithmetic and task vectors and how they can be applied to Dense, Learned Sparse and Cross-Encoder approaches for domain-specific retrieval.

Task Vector Definition For our purposes, a task t refers to the specific objective a model is trained to achieve, such as document classification or relevance prediction, using a task-specific loss. A domain d denotes the application area in which the task is performed, such as biomedical or scientific text, and a domain-specific model is defined by continually pre-training a PLM with a masked language modelling objective to adapt it to the domain's word distribution. Let $\Theta_0 = \{(\theta_1)_0, \ldots, (\theta_N)_0\}$ denote the parameters of a pre-trained language model and $\Theta_t = \{(\theta_1)_t, \ldots, (\theta_N)_t\}$ and $\Theta_d = \{(\theta_1)_d, \ldots, (\theta_N)_d\}$ the corresponding weights after fine-tuning on task t and continual pre-training on domain d, respectively. The task vector $\tau_t = \{(\tau_1)_t, \ldots, (\tau_N)_t\}$ is the element-wise difference between Θ_t and Θ_0, i.e $\tau_t = \Theta_t - \Theta_0$ and $(\tau_i)_t = (\theta_i)_t - (\theta_i)_0$. Thus, the task vector has the same size as the models Θ_0 and Θ_t, but its values typically have a smaller magnitude,

as it is a delta between two models at the parameter level. Task vectors can be applied to any model parameters Θ from the same architecture, via element-wise addition, with an optional scaling term α, such that the resulting model has weights $\Theta_{new} = \Theta + \alpha\tau_t$. When $\alpha = 0$, Θ_{new} reduces to the Θ model, while adding τ_t to Θ_0 with $\alpha = 1$ results in Θ_t, i.e. the fine-tuned model on that task. In practice, α is a hyperparameter that can be fixed (and be in a fully zero-shot scenario) or tuned on a small development set (if available). The task vector τ_t represents the knowledge learned by the model while fine-tuning it on the task t. Applying the task vector to Θ allows us to transfer the task-related knowledge to a new model. Recently, task vector has been used to transfer domain-specific knowledge by defining the task vector as $\tau_d = \Theta_d - \Theta_0$, which represents the domain-specific knowledge, which can be transferred to a new model $\Theta_{new} = \Theta + \alpha\tau_d$.

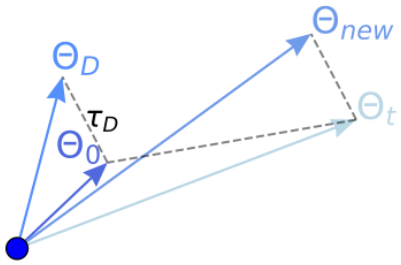

Figure 8.1: Illustration of the application of task vectors to define domain-specific ranking models: given a PLM Θ_0, we define the task vector as the difference between a Θ_d and Θ_0, where Θ_d is the domain-specific fine-tuned version of the PLM. Then, we add the task vector τ_t to model Θ_t, which is a ranking model based on Θ_0, to obtain a new domain-specific ranking model Θ_{new}.

Task Vectors for IR We focus on using task vectors to improve the retrieval performance of dense, learned sparse and cross-encoder models on specific tasks, such as scientific and biomedical retrieval. The application of task vectors allows us to transfer domain knowledge from the multitude of publicly available fine-tuned models, without additional training or access to training labelled data. We start from Θ_0, which is a pre-trained language model. For defining and applying task vectors, we need two fine-tuned versions of Θ_0, i.e. Θ_t and Θ_d. Since in our work we focus on transferring domain-specific knowledge into a ranking model, Θ_d is the version of Θ_0 obtained by continuing pre-training the model Θ_0 on a large corpus of unlabeled domain-specific text by using the masked language model loss [103, 279], while Θ_t is the version of Θ_0 fine-tuned to learn to estimate the relevance score between a given query and document pair. We define the task vector of the specific domain d as the difference between the fine-tuned model on the specific domain d

and the pre-trained model:

$$\tau_d = \{(\tau_1)_d, \ldots, (\tau_N)_d\}, \quad \text{with} \quad (\tau_i)_d = (\theta_i)_d - (\theta_i)_0. \tag{8.1}$$

Then we transfer the domain-specific knowledge into a ranking model, i.e. Θ_t, to define a domain-aware ranking model as follows:

$$\Theta_{new} = \{(\theta_i)_{new} = (\theta_i)_t + \alpha(\tau_i)_d\}_{i=1}^{N}. \tag{8.2}$$

While Θ_t encodes knowledge for assessing relevance, Θ_d captures domain-specific knowledge, including shifts in vocabulary, style and language. Specifically, the task vector τ_d represents the language- and domain-specific knowledge that Θ_0 learned during further pre-training with masked language modelling on domain d. By applying the task vector τ_d to Θ_t, this domain- and language-specific knowledge is transferred into the ranking model, yielding a new domain-specific ranking model. This approach effectively combines the model's ability to judge relevance (Θ_t) with its capacity to handle the linguistic characteristics of the target domain d.

8.3 Experimental Setup

Our study investigates the application of task vectors in information retrieval, with the aim of improving ranking model performance on scientific retrieval tasks without additional fine-tuning or labelled data. In particular, we focus on three types of ranking models, dense, learned sparse and cross-encoders, leading to the following research questions:

- RQ1: Do task vectors enhance the performance of dense models on scientific retrieval tasks?
- RQ2: Do task vectors enhance the performance of learned sparse models on scientific retrieval tasks?
- RQ3: Do task vectors enhance the performance of cross-encoder on scientific retrieval tasks?
- RQ4: Do task vectors enhance the performance of cross-encoder on multilingual retrieval tasks?
- RQ5: What are the impacts of the introduced hyper-parameter on the performance of task vectors?

Pre-trained Model Θ_0	Domain/Language-specific Model Θ_d	Training Dataset	Retriever or Re-ranker Θ_t
DistilBERT	Bio-DistillBERT	PubMed Abstract	DistillBERT-MS-MARCO, TAS-B, SPLADE
BERT	BLEU-BERT	PubMed Abstract	BERT-MS-MARCO, TCT, DeepCT
RoBERTa	BioMed-RoBERTa	S2ORC	RoBERTa-MS-MARCO, ANCE
T5	SciFive	PubMed Abstract, PMC articles	GTR, Doc2Query, MonoT5
MT5	MT5-German, Spanish, French, English	WikiNews	MT5-MS-MARCO
LLama2	LLama2-MedTuned-7b	Medical NER and RE, PubMedQA	RankingGPT-Llama2-7b

Table 8.1: All models used to evaluate our approach: we report the pre-trained model (Θ_0), its version fine-tuned on a specific domain or language Θ_d and the dataset on which it is trained, and finally the model trained on MS-MARCO for learning relevance Θ_t

Datasets To evaluate our approach, we use four scientific and biomedical datasets from BEIR [442]. We use SciFact [459], SciDocs [92], NFCorpus [42] and TREC-Covid [456]. SciDocs and SciFact focus on scientific citation prediction and fact checking, respectively, while both TREC-Covid and NFCorpus address biomedical retrieval. SciFact and NFCorpus are both based on PubMed [122], with SciFact covering a wide range of scientific articles from basic science to clinical medicine [92], and NFCorpus featuring non-technical queries over documents with specialised terminology [42]; TREC-Covid also relies on PubMed articles focused on Covid-19 [456]. SciDocs, in contrast, is based on the Microsoft Academic Graph [92, 459] for citation retrieval. We use their test versions available through the ir-datasets package [296]. We focus on the biomedical and scientific domains, where publicly available domain-specific models allow the definition and application of task vectors. In contrast, we exclude BEIR datasets based on Wikipedia since the PLMs in our experiments have already been trained on Wikipedia [103, 279], and Task Arithmetic cannot add domain-specific knowledge. To further asses the effectiveness of Task Arithmetic behind domain adaptation, we evaluate our approach on language-specific retrieval, specifically on GermanQuAD [318], which targets question answering in German, and three subsets (English, French, and Spanish) from the MIRACL multilingual IR challenge [525].

Models Details For our experiments, we rely on seven different pre-trained language models (i.e. Θ_0) and seventeen Dense, Learned-Sparse and Cross-Encoder models. In detail, we use `DistillBERT-base` [409], `BERT-base` [103], `RoBERTa-base` [279], `T5-base` and `T5-Large` [376], `MT5-base` [501] and `LLama-2-7b` [448] as pre-trained language models Θ_0, spanning encoder-only, encoder-decoder and decoder-only architectures. For each pre-trained model, we compute task vectors τ_d by subtracting the weights of the original pre-trained model Θ_0 from those of the publicly available domain-specific variant Θ_D. Specifically, for the biomedical and scientific domains, we use `Bio-DistilBERT` [398], `BLUEBERT` [352], `BioMed-RoBERTa` [152], `SciFive-base` and `SciFive-Large` [356] and `LLama2-MedTuned-7b` [397], as provided by HuggingFace [201]. For our multilingual evaluation, we rely on `MT5-base-german`, `MT5-base-spanish`, `MT5-base-french`,

and `MT5-base-english` [59].

Table 8.1 reports all the pre-trained language models, their domain-specific versions and the datasets on which these domain- or language- specific versions are trained. One of the most used datasets for continual training of a PLM on a scientific domain is PubMed [122], which contains citations and abstracts of biomedical literature, is used for four out of five models. While Bio-DistillBERT and BLUE-BERT are trained only on the PubMed abstracts, SciFive uses both PubMed abstracts and PubMed Central (PMC) articles [197], which is a corpus of full-text articles in the domain of biomedical and life sciences. In contrast, while LLama2-MedTuned-7b has been trained of PubMedQA [211], which is a question-answering dataset based on PubMED, it has been further trained on Medical Named Entity Recognition and Relation Extraction tasks [397]. BioMed-RoBERTa [152] is the only model that is not pre-trained on PubMed abstracts, but it is trained on a subset of 2.68M full-text papers from S2ORC [283].

For our experiments, as ranking models Θ_t, we focus on Dense, Learned Sparse and Cross-Encoders models. Specifically, we use eleven Dense retrieval models:

- **DistilBERT-msmarco** [389], **BERT-msmarco** [389] and **RoBERTa-msmarco** [389] are defined by training DistilBERT, BERT and RoBERTa, respectively, on MS MARCO for dense retrieval.
- **ANCE** [496] constructs, during training, negative examples from an approximate nearest neighbour index of the corpus.
- **TCT** [264] employs ColBERT[231] as the teacher model and distils knowledge to a bi-encoder based student model.
- **TAS-B** [178] performs k-nearest neighbour search over a flat index to avoid variability introduced in using approximation techniques.
- **BGE-base-en** [494] is pre-trained using RetroMAE [495] and then trained MS MARCO using contrastive learning.
- **GTR-base** [540] is based on two T5-encoders pre-trained on question-answering pairs and then further fine-tuned on MS MARCO.
- **Dragon-base** [266] is a BERT-base model initialized from RetroMAE and further trained on the data augmented from MS MARCO
- **E5-BERT** [469], which is based on BERT-base, is trained in a contrastive manner with weak supervision signals.
- **Co-Condenser** [132] is a BERT-base model trained pre-trained towards the bi-encoder structure.

Furthermore, we evaluate task vectors on three Learned Sparse retrieval models:

- **Doc2Query** [335] is a document expansion technique based on T5 to generate synthetic queries and append them to the original document.
- **DeepCT** [98] is a BERT-based model that generates a pseudo-document with keywords multiplied by the learnt term frequencies.
- **SPLADE-v3** [237], which is based on DistilBERT, balances efficiency, effectiveness, and representation size. We index and retrieve using PISA [304].

Finally, we evaluate task vectors on three Cross-encoder models:

- **MonoT5-base** and Large [336] are based on T5-base and Large and encode both the query and the document with the structured prompt `Query: [q] Document: [d] Relevant:`.
- **MT5-base-MS-MARCO** [39] is the multilingual version of MonoT5.
- **RankingGPT-Llama2-7b** [516], which is based on Llama2 with 7B parameters, encode both the query and the document with the prompt `Document: [d] Query: [q]`.

We tune the scaling factor α from 0.1 to 1.0 in steps of 0.1, selecting the optimal value based on the highest average retrieval performance over two development sets: the official NFCorpus split and a 20% subset of SciFact training queries. Since GermanQuAD and MIRACL do not provide development sets, we apply a fully zero-shot scenario by not optimizing the value of α, i.e. we put $\alpha = 1$. All re-ranking experiments begin by retrieving the top 100 documents via BM25. Following a two-stage retrieval paradigm, the final rankings are then computed using a weighted sum of BM25 and LLM scores, with λ_{BM25} and λ_{LLM} optimized in [0,1] on the NFCorpus and SciFact development sets. We take the average score for all remaining datasets, i.e. $\lambda_{BM25} = \lambda_{LLM} = 0.5$.

Baselines As baselines, we employ a lexical ranker, namely BM25, and each model Θ_t, i.e. a model fine-tuned on MS MARCO [328], as provided by PyTerrier [297]. In detail, BM25 is a lexical retriever based on a variant of TF-IDF-based term weighting coupled with document length normalisation.

Evaluation Metrics We measure the quality of retrievers with $nDCG$@10 [203], which are the official measure for BEIR benchmarks [442]. For significance testing with agreement measures, we use a paired t-test to calculate p-values. We use $p < 0.05$ and Bonferroni correction for multiple tests [112].

Model \Dataset	SciFact	NFCorpus	TREC-Covid	SciDocs
BM25	.691	.343	.688	.165
Dense First-stage retrievers				
DistilBERT	.473+**.066***	.255+**.013***	.604+**.054***	.100+**.011***
BERT	.577+**.024***	.298−.011	.672−.073	.129+**.007***
RoBERTa	.456+**.065***	.237+**.009***	.389+**.051***	.100+**.007***
ANCE	.493+**.029***	.236+**.007**	.654+**.029**	.113+**.006***
TCT	.539+**.036***	.266+**.004**	.616+**.012**	.112+**.004**
TAS-B	.610+**.004**	.310−.004	.610+**.022**	.128+**.008***
BGE-en	.698−.018	.296+**.010***	.491+**.051***	.153+**.015***
GTR-base	.574+**.038***	.300+**.006***	.641+**.052***	.132+**.004**
Dragon	.654-0.06	.323+**.003**	.648+**.010**	.144+**.001**
Co-Condenser	.577+**.007**	.316-.020	.642-.072	.122+**.008***
E5-BERT	.712+ **.001**	.343-.012	.707-.027	.171+.0
Dense re-rankers				
DistilBERT	.703+**.017**	.357+**.002**	.744+**.021**	.168+**.003**
RoBERTa	.707+**.013**	.359+**.002**	.732+**.025***	.170+**.003**
Learned Sparse First-stage retrievers				
Doc2Query	.676+**009**	.325+**.006***	.631−.032	.152+**.001**
DeepCT	.667−.011	.320−.016	.582−.005	.141−.002
SPLADE	.658+**.008**	.337−.006	.632+**.018**	.147−.006
Cross-Encoders re-rankers				
T5-base	.726+**.022**	.359-.001	.712+**.041**	.173+**.003**
T5-Large	.743-.013	.368-.010	.735+**.024**	.182+**.005***
Llama-2-7B	.770-.005	.373-.008	.810+**.002**	.188+**.001**

Table 8.2: Experimental results (nDCG@10) on biomedical and scientific domains for dense, learned sparse retrieval and cross-encoder models. Numbers before + or − are the zero-shot performance of each model, while numbers after these symbols show the Δ effect of applying the task vectors. Bold (*) values denote (significant) improvements.

8.4 Results

In this section, we answer our research questions outlined in Section 8.3 and provide further analysis of our proposed approach. More specifically, we examine whether domain transfer consistently benefits ranking effectiveness across four target datasets. We report the results of our evaluation in Table 8.2. Numbers

before + or − are the zero-shot performance of each dense or learned sparse model (i.e. Θ_t), while numbers after these symbols show the effect of applying task vector τ_d as in Equation (8.2). We denote statistically significant improvements with *. Firstly, to answer RQ1 RQ2 and RQ3, we apply task vectors to eleven dense, three learned sparse retrieval and three cross-encoder models and evaluate them across four scientific and biomedical benchmarks. As described in Section 8.3, the domain models Θ_t are mainly trained on PubMed abstracts, PMC articles, and S2ORC. Therefore, we hypothesise that task vectors will yield stronger improvements on datasets whose corpora are derived from PubMed or related scientific resources, namely SciFact, NFCorpus, and SciDocs, whereas their effectiveness may be limited on TREC-Covid, since the scientific models were trained before 2021 and their corpora do not include Covid-related information.

8.4.1 RQ1: Do task vectors enhance the performance of dense models on scientific retrieval tasks?

We show the results of our evaluation in Table 8.2. For all dense models, the application of task vectors improves the performance in $nDCG@10$ of the model trained on MS MARCO (i.e., Θ_t) on all evaluated datasets, except for BERT on NFCorpus and TREC-Covid. In detail, we observe that, for SciFact and SciDocs, most models benefit from task vectors, with significant gains in $nDCG@10$ for DistilBERT, BERT, RoBERTa, ANCE and GTR-base on both the datasets. The application of task vectors significantly improves TCT and BGE-en only on SciDocs. For NFCorpus, the impact of task vectors is more varied: while DistilBERT, RoBERTa, GTR-base, and BGE-en are significantly improved by the task vectors, BERT and TAS-B are not. This mixed behaviour can be attributed to the specific design of NFCorpus: documents are sourced from PubMed, but queries are written in non-technical English. Thus, task vectors, which inject domain-specific knowledge primarily tailored to biomedical terminology, may help models better understand documents but hinder their ability to bridge the lexical gap with the query style. TREC-Covid highlights the challenges of applying task vectors when the target task lies outside the temporal scope of the domain model training. Here, DistilBERT, RoBERTa, BGE-en, and GTR-base exhibit significant improvements in $nDCG@10$. BERT, however, exhibits a decrease in both metrics, suggesting that the introduction of information not aligned with the target domain can reduce retrieval effectiveness. From a model-focused perspective, the application of task vectors to DistilBERT, RoBERTa, and GTR-base consistently exhibits significant improvements across all four datasets compared to the baseline ranking model version trained on MS MARCO. DistilBERT achieves the highest gains overall: +14.0% $nDCG@10$ on SciFact. ANCE, TCT, and BERT are improved significantly by task vectors only on SciFact and SciDocs, confirming our hypothesis that task vectors are most effective when the target dataset shares the same domain as

the scientific model. Broadly speaking, all dense model baselines perform worse than BM25 due to the domain shift from MS MARCO to scientific and biomedical corpora. Nevertheless, in specific cases, the application of task vectors allows dense models to outperform BM25, such as DistilBERT on TREC-Covid, TCT on NFCorpus and TREC-Covid, and BGE-en on SciDocs. Overall, to answer RQ1, we find that task vectors improve all the evaluated dense models trained on MS MARCO on scientific and biomedical datasets (except for BERT on NFCorpus and TREC-Covid), with the largest gains observed on models and datasets closely aligned with the same domain. For datasets or tasks with partial or no domain overlap, improvements are mixed, emphasising the importance of both model choice and dataset characteristics for effective domain adaptation.

8.4.2 RQ2: Do task vectors enhance the performance of learned sparse models on scientific retrieval tasks?

Table 8.2 reports the results of our learned sparse experiments. Considering first Doc2Query, we observe that applying task vectors generally improves $nDCG@10$ across most datasets, with a significant gain on NFCorpus. Doc2Query does not achieve improvement on TREC-Covid, which is expected given the temporal and topical shift: the scientific model Θ_d lacks Covid-specific knowledge, so task vectors cannot fully compensate for this domain gap. For SPLADE and DeepCT, both models already achieve strong baseline performance on most datasets compared to BM25. However, applying task vectors results in a small decrease in $nDCG@10$ across all datasets. We hypothesise that this decrease in performance for Learned Sparse is caused by the way task vectors inject domain-specific knowledge: these models rely on sparse, term-level representations learned from MS MARCO, which may already capture robust general relevance patterns. Introducing additional domain-specific information can potentially distort these representations, particularly when the target dataset queries and documents differ lexically from the training corpus. In other words, the task vector may shift the model's focus toward biomedical terminology in ways that do not always align with the existing retrieval signal, reducing effectiveness. On inspection of these results, we observe that the impact of task vectors on learned sparse retrieval is highly dependent on the interplay between model architecture, training data, and dataset characteristics. Doc2Query benefits in datasets where domain adaptation can bridge gaps between query and document styles, while SPLADE and DeepCT are more sensitive to perturbations of their already strong sparse representations. This suggests that, for learned sparse retrieval models, additional domain adaptation needs to carefully consider the alignment between injected knowledge and the model's existing representation space. To concretely answer RQ2, task vectors provide slight improvements for Doc2Query, particularly on NFCorpus, but can reduce performance for SPLADE and DeepCT across datasets.

To further assess the impact of task vectors on learned sparse retrieval, we compare the queries generated by Doc2Query on NFCorpus before and after applying task vectors, focusing on this setting because it is the only case where task vectors yield significant improvements over the model fine-tuned on MS MARCO. For example, for a document about epilepsy as a common neurological disorder, the Doc2Query model generates queries like: *What is seizure? Which type of disorder is a neurological disorder?*, while the application of task vector to Doc2Query gives a query like: *which is the most common neurological disorder? Which diseases are associated with epilepsy?*. Thus, the queries given after the application of the task vector directly focus on epilepsy, which is the main topic of the document, while the queries generated without the task vector are more generic ('*What is seizure*') and fail to focus on epilepsy. This qualitative example shows how applying the task vector to Doc2Query leads to generating more precise and targeted queries in the scientific domain compared to those generated by the original model.

8.4.3 RQ3: Do task vectors enhance the performance of cross-encoder on scientific retrieval tasks?

Table 8.2 reports the performance of our approach on biomedical and scientific benchmarks, evaluated across five different backbone models. Across all datasets and metrics, the Task Arithmetic consistently outperform BM25, confirming its effectiveness as a domain adaptation strategy for IR. For bi-encoder architectures (DistilBERT and RoBERTa-base), Task Arithmetic yields consistent gains over all considered baselines, demonstrating that task vectors provide a robust mechanism to inject domain-specific knowledge into dense retrieval models. For cross-encoders (T5-base and T5-Large), Task Arithmetic improves over MonoT5 (Θ_T) on SCIDOCS and TREC-COVID, while achieving comparable performance on SciFact and NFCorpus. Notably, the proposed method achieves statistically significant improvements in nDCG@10 on both TREC-COVID and SCIDOCS. Regarding decoder-only models (LLama-2), our approach achieves the strongest results on TREC-COVID compared to all baselines and remains competitive on SCIDOCS. Interestingly, the pre-trained LLama-2 (Θ_0) attains the highest nDCG@10 on SciFact and SCIDOCS, which likely reflects the extensive domain knowledge acquired during large-scale pre-training [448]. Overall, to answer RQ3, these results show that task vectors improve the performance of MS MARCO-trained re-ranking models on scientific and biomedical datasets (except for LLama-2), with the largest gains observed on BERT- and T5-based architectures. In contrast, LLama-2 exhibits less consistent behaviour, suggesting that its adaptation dynamics require further analysis.

8.4.4 RQ4: Do task vectors enhance the performance of cross-encoder on multilingual retrieval tasks?

Model		GermanQuAD	MIRACL Spanish	MIRACL French	MIRACL English
BM25		.437	.270	.174	.302
MT5-base	Θ_0: Pre-trained	.336	.191	.127	.212
	Θ_D: Language specific	.353	.191	.143	.225
	Θ_T: MS-MARCO (en)	.513	.379	.234	.398
	Θ': Task Arithmetic ($\alpha = 1$)	**.537***	**.405***	**.278***	**435***

Table 8.3: Main results in nDCG@10 on four different languages, i.e. German, Spanish, French and English, with MT5 as base model.

Table 8.3 extends the evaluation to multilingual IR by using MT5-base as a Cross-Encoder on language-specific retrieval tasks. Both our proposed approach and the IR-specific baseline Θ_t consistently outperform all other baselines across every evaluation metric. In particular, our method achieves statistically significant improvements over the IR-specialised model, reaching gains of up to 18% in NDCG@10, which highlights its effectiveness in adapting to new language settings. In contrast, the pre-trained MT5-base model and its language-adapted variant fail to surpass BM25, suggesting that these models are not inherently optimised for retrieval. Our approach instead successfully injects language-specific knowledge into the multilingual IR model, thereby substantially improving its retrieval effectiveness. To concretely answer RQ4, these findings confirm Task Arithmetic as a lightweight yet effective strategy for zero-shot multilingual adaptation in IR.

8.4.5 RQ5: What are the impacts of the introduced hyperparameter on the performance of task vectors?

Recall that Equation (8.2) includes a trade-off hyperparameter α to balance the domain-specific knowledge provided by the task vector τ_d and the relevance capabilities of the ranking model Θ_t. To answer RQ5, we conduct experiments by varying the value of α from 0 to 1.0 and examining its impact on $nDCG$@10. When $\alpha = 0$, no task vector is applied and the baseline model Θ_t is evaluated. Our hypothesis is that if the value of α is close to 1, we are adding too much domain knowledge into the ranking model, overshadowing the relevance signals learned by the model. Thus, we expect that the retrieval performance will decrease as $\alpha \to 1$.

How to chose the optimal α We conduct an ablation study to examine scaling factor impact while applying Task Arithmetic. Table 8.4 presents NDCG@10 scores on NFCorpus and SciFact development sets for α values from 0.1 to 1.0 in increments of 0.1. We conduct these experiments in a re-ranking setting, i.e. DistilBERT, RoBERTa, T5-base and Large, and LLaMA-2, with BM25 as a fixed

Model	Dataset	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
LLama 2-7B	**SciFact**	.631	.643	.660	.675	.687	.705	**.712**	.711	.705	.704
	NFCorpus	.235	.238	.239	.241	**.242**	.241	.239	**.242**	.241	.241
T5-Large	**SciFact**	.728	.732	.727	.731	.730	.732	.728	.730	**.737**	.710
	NFCorpus	.307	.307	.307	.308	.308	.306	.306	**.311**	**.311**	.309
T5-base	**SciFact**	.712	.710	.712	.711	.709	.718	**.722**	.708	.673	.640
	NFCorpus	.302	.301	.304	.306	.306	.311	**.313**	.312	.303	.298
RoBERTa-base	**SciFact**	.713	.717	**.725**	.723	.722	.710	.715	.706	.695	.687
	NFCorpus	.334	.331	.332	.332	.333	**.334**	.330	.323	.315	.307
DistilBERT	**SciFact**	.723	.720	.723	.722	**.724**	.712	.705	.699	.688	.652
	NFCorpus	.334	**.336**	.333	.333	.333	.329	.325	.306	.297	.286

Table 8.4: Impact of the scaling factor in cross-encoder models on SciFact and NFCorpus validation sets.

first-stage retriever. By keeping the candidate generation stage unchanged, we ensure that performance variations are attributable to the Task Arithmetic adaptation of the re-ranker, rather than to changes in recall induced by an adapted dense or learned sparse retriever. The results reveal that retrieval performance varies considerably with changes in α. On SciFact, for instance, T5-base improves from .640 at $\alpha = 1.0$ to .722 at $\alpha = 0.7$, representing a notable difference of approximately 12%. In general, while values in the 0.7–0.9 range often yield strong results for models such as T5-base and T5-Large, no single α value is consistently optimal across all models or datasets. This variability underscores the importance of model-specific calibration. Moreover, $\alpha = 1.0$ rarely provides the best performance, suggesting that excessive emphasis on domain parameters can overshadow the IR-specific knowledge embedded in IR-tuned weights. RoBERTa-base and DistilBERT sometimes peak at moderate values between 0.3 and 0.5, whereas T5-based models and LLama-2 commonly favor higher settings. These observations indicate that a small grid search over α can be effective in identifying a tradeoff between the IR and the domain-specific knowledge.

What are the impacts of α on dense and learned sparse models? Figures 8.2 and 8.3 present the results for dense and learned sparse retrieval models, respectively, expressed as the percentage gain or loss relative to the baseline Θ_t, i.e. when the task vector is not applied and $\alpha = 0$.

We observe that in most cases, increasing α leads to a steady decrease in performance, with relative $nDCG@10$ gains turning negative as the task vector begins to overshadow the relevance signals captured by Θ_t. However, this trend does not hold universally. For example, in TREC-Covid with RoBERTa and GTR, as well as in SciFact, effectiveness actually improves at higher values of α (around 0.6–0.7), indicating that stronger domain adaptation can complement relevance modelling

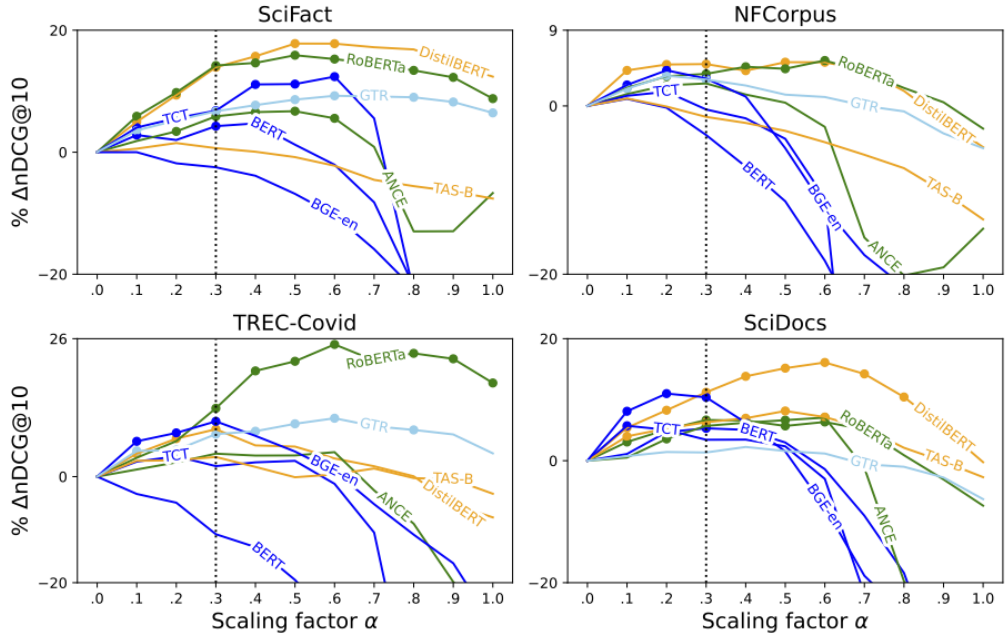

Figure 8.2: Relative difference (%) in $nDCG@10$ of eight dense retrieval pipelines by increasing the value of hyperparameter α. Effectiveness measurements that are significant improvements over the MS MARCO model (i.e. $\alpha = 0$) are marked with •.

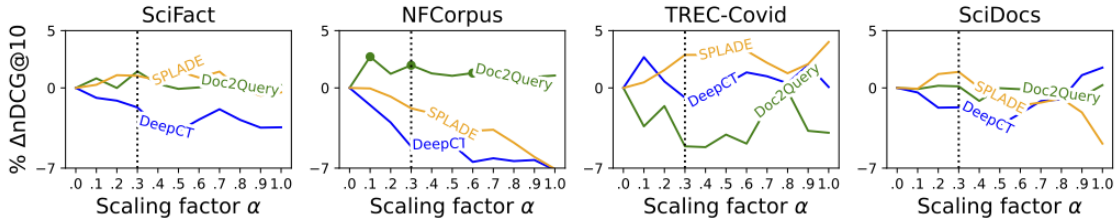

Figure 8.3: Relative difference (%) in $nDCG@10$ of three learned sparse retrieval pipelines by increasing the value of hyperparameter α. Notation as in Figure 8.2.

when specialised terminology is critical. In contrast, NFCorpus reaches its best performance around $\alpha = 0.3$, with no substantial gains at larger values, while SciDocs shows a monotonic decline as α increases. These findings show that the optimal value of α is dataset-dependent, and a fixed choice does not generalise well across domains. Furthermore, no single dense model dominates across datasets: DistilBERT achieves the strongest results in SciFact and SciDocs, whereas RoBERTa performs best in TREC-Covid. Regarding learned sparse retrieval, we find that applying task vectors generally reduces performance. Significant improvements over the baseline are observed only for Doc2Query on NFCorpus, while for TREC-Covid, both SPLADE and Doc2Query show performance gains when $\alpha > 0.8$. In all other

cases, effectiveness declines, suggesting that sparse models are more sensitive to over-weighting domain knowledge than dense models. Overall, these results indicate that increasing $\alpha > 0.3$ typically harms retrieval performance, but there are a few exceptions, where additional domain signals enhance effectiveness. Importantly, both the optimal value of α and the best-performing model vary by dataset, underscoring that no single configuration fits all scenarios. To concretely answer RQ3, the optimal value for α to balance domain adaptation without overshadowing the relevance signals depends on the model and the specific domain.

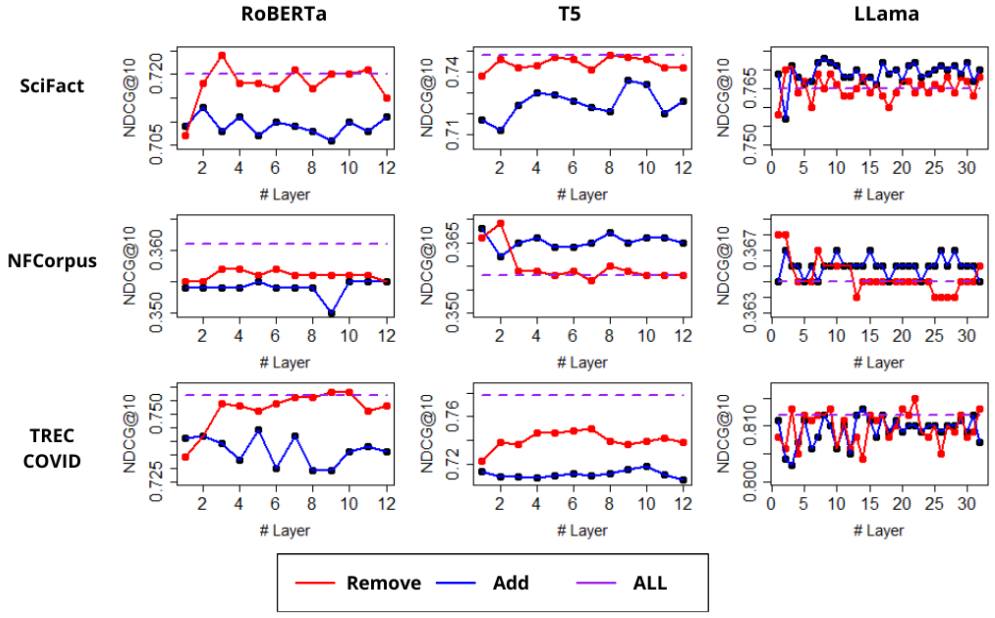

Figure 8.4: Layer impact on nDCG@10 while applying Task Arithmetic. We evaluate both adding and removing layers in red and blue, respectively. In purple, we show the results while adding all the layers.

Analysis of Layer Impact. We further investigate which Transformer layers contribute most to the transfer, and conversely the suppression, of domain-specific knowledge. To this end, we perform a layer-wise Task Arithmetic analysis on three benchmarks, namely SciFact, NFCorpus, and TREC-COVID. Due to the computational cost of evaluating LLama-2-7b, we exclude SCIDOCS from this study. Since this analysis focuses on the effect of task vectors, we adopt a fixed BM25 first-stage retrieval pipeline to isolate the impact of layer-wise Task Vector manipulations on ranking quality, avoiding confounding effects that would arise if retrieval recall were simultaneously altered by adapting the first-stage retriever. We compute layer-specific Task Vectors that selectively activate or suppress the parameters of a single layer L. Formally, let $\tau_t = \{\tau_1, \ldots, \tau_N\}$ denote the full Task Vector. To

evaluate the contribution of an individual layer, we construct a sparse Task Vector by enforcing $\tau_i \neq 0$ if and only if the corresponding parameters $(\theta_i)_T$ belong to layer L, while setting all remaining components to zero. This setting measures the extent to which domain knowledge can be transferred through that layer alone. Conversely, we evaluate the effect of excluding layer L by setting $\tau_i = 0$ if and only if $(\theta_i)_t$ belongs to layer L, while retaining all other components. This variant reveals which layers are most critical for removing domain-specific information from the adapted model. We conduct these experiments using RoBERTa-base, T5-base, and LLama-2-7b, covering encoder-only, encoder–decoder, and decoder-only architectures, respectively. Figure 8.4 reports NDCG@10 for each configuration. Overall, the results show that selectively adding or removing a single layer generally yields performance that is comparable to, but often lower than, the full Task Vector baseline. In particular, on TREC-COVID and NFCorpus with RoBERTa-base, and on SciFact and TREC-COVID with T5-base, both layer-restricted and layer-excluded variants consistently underperform the full-layer configuration. A clear trend emerges across architectures: later transformer layers appear more influential for retrieval, as manipulating these layers typically produces larger performance variations than modifying earlier layers. This behaviour suggests that domain-specific information relevant to ranking is predominantly encoded in higher layers, which are responsible for modelling more abstract semantic representations. NFCorpus constitutes a partial exception for T5-base and LLama-2-7b, where selectively activating or excluding individual layers occasionally surpasses the full-layer Task Vector. This indicates that layer-wise interactions may vary across datasets and model families, potentially due to domain-specific distribution shifts or differences in how deeper layers encode retrieval-relevant semantic features. Overall, these findings support the use of full-layer Task Vectors as the most stable strategy, while highlighting that a subset of higher layers can have a disproportionate impact on retrieval effectiveness.

8.5 Conclusions

This Chapter investigates Task Arithmetic for domain adaptation of dense, learned sparse retrieval and cross-encoder models, encompassing encoder-only, encoder-decoder and decoder-only architecture and evaluating task vectors across seventeen retrieval paradigms in scientific and biomedical domains. Our evaluation shows that Task Arithmetic yields consistent improvements over MS MARCO-trained ranking models, with gains up to 24% in $nDCG$@10. Furthermore, we show the effectiveness of the proposed approach even in multilingual IR tasks. Finally, we evaluate the impact of hyperparameter α and each layer in the Task Vectors. Task Arithmetic thus provides a training-free method to transfer domain knowledge and reduce reliance on costly labelled datasets.

Chapter 9

Revealing MonoT5's Learning Mechanisms via Prompt-Token Adaptation

The introduction of the Transformer architecture [454] and the subsequent development of large Pre-trained Language Models (PLMs), such as T5 [376], have significantly advanced neural approaches to Information Retrieval (IR). Among these models, MonoT5 [336] has emerged as a strong cross-encoder reranker by fine-tuning T5 on the MS MARCO dataset [328], achieving state-of-the-art effectiveness across several retrieval benchmarks. MonoT5 follows the cross-encoder paradigm [195], where the query and the document are jointly encoded as a single input sequence. Unlike BERT-based reranking models [337], MonoT5 relies on a prompt-based formulation that explicitly introduces task-specific tokens, such as `Query`, `Document`, and `Relevant`, which guide the model during fine-tuning. The model is then trained to generate the label `true` when the document is relevant to the query, and `false` otherwise. Despite its strong empirical performance, the internal adaptation dynamics of MonoT5 remain relatively underexplored. In particular, it is unclear how relevance signals are encoded during fine-tuning and which components of the model contribute most to its retrieval behaviour. To address this gap, in this Chapter, we analyse the parameter shifts induced when fine-tuning T5 as MonoT5 on MS MARCO [328], i.e. its corresponding task vector. Our analysis shows that these changes are not uniformly distributed across the model, but instead concentrate in a small subset of parameters, primarily within the embedding layer. Notably, the embeddings associated with the prompt and label tokens, i.e. `Query`, `Document`, `Relevant`, `true`, and `false`, exhibit the largest shifts. This suggests that a substantial amount of retrieval-specific behaviour is encoded directly into these token representations, while the majority of the model remains comparatively stable.

Building on these findings, in this Chapter, we propose a new training strategy

that updates only the embedding vectors associated with the prompt tokens, while keeping the remaining parameters frozen. This approach falls within the broader family of Parameter-Efficient Fine-Tuning (PEFT) methods, which aim to adapt large models by training only a limited subset of parameters. Our strategy requires updating only 0.002% of the model parameters, while preserving retrieval effectiveness. We refer to the resulting model as Light-MonoT5, highlighting its lightweight adaptation mechanism. We evaluate Light-MonoT5 on both in-domain benchmarks (TREC DL 19 [94], DL 20 [95], and DL Hard [298]) and out-of-domain collections from BEIR [442], showing that its performance is not statistically significantly different from that of full MonoT5 fine-tuning. Finally, based on our empirical observations, we hypothesise that MonoT5 primarily learns to assess passage quality rather than learning semantic relevance, since the base T5 model [376] already captures semantic matching during pre-training. To test this hypothesis, we leverage Quality T5 (QT5) [65], a T5-based model designed to identify low-quality passages that are unlikely to be relevant to any query. When applying Light-MonoT5 to a QT5-pruned corpus, we obtain retrieval performance comparable to MonoT5. These results further support the hypothesis that MonoT5's effectiveness is strongly driven by its ability to model passage quality, and that once low-quality passages are removed, only minimal parameter adaptation is sufficient to recover full MonoT5 performance. To summarise, the contributions of this Chapter are as follows:

- We analyse the parameter dynamics of fine-tuning T5 on MS MARCO, showing that the largest shifts concentrate in the embeddings of prompt-related tokens.
- We introduce a novel parameter-efficient tuning strategy that adapts only prompt embeddings, reducing trainable parameters to 0.002% while maintaining retrieval effectiveness.
- We evaluate the proposed approach on both in-domain and out-of-domain benchmarks, showing its robustness across various retrieval settings.
- Building on these findings, we show that MonoT5 largely captures passage quality signals rather than modelling query-document relevance.

The remainder of the Chapter is organised as follows. Section 9.1 reviews related work on T5 and parameter-efficient fine-tuning techniques. Section 9.2 presents our analysis of the parameter shifts induced by fine-tuning T5 on MS MARCO. Section 9.3 describes Light-MonoT5 in detail. Section 9.4 outlines the experimental setup, Section 9.5 reports and discusses the results, and Section 9.6 concludes the Chapter.

9.1 Related Work

This section reviews prior work on T5 and its application in Information Retrieval (IR).

T5 [376] is a Transformer-based encoder-decoder model [454] that formulates all tasks under a unified text-to-text paradigm. Its architecture consists of an encoder, which processes the input sequence, and a decoder, which generates the output sequence conditioned on the encoder representations. T5 is pre-trained on the C4 corpus [376], a large collection of cleaned English web documents, and subsequently fine-tuned on a diverse set of supervised tasks, including translation, question answering, and classification. To specify the target task, Raffel et al. [376] prepend task-specific textual prefixes to the input sequence. This design enables the model to reuse a single architecture while adapting its behaviour through natural language prompts. Building on this paradigm, Nogueira et al. [336] proposed MonoT5, which fine-tunes T5 for document reranking on MS MARCO [328]. MonoT5 follows the cross-encoder setting, jointly encoding the query q and the document d as a single input prompt of the form `Query: [q] Document: [d] Relevant:`, and training the model to generate the tokens `true` or `false` as relevance labels. Recent work has shown that MonoT5 strongly relies on the prompt structure and its associated tokens. In particular, while MonoT5 can adapt to arbitrary relevance label semantics [336], it is also vulnerable to prompt-based adversarial manipulations such as stuffing, rewriting, and preemption [346], highlighting the central role of prompt tokens in controlling its ranking behaviour. Beyond query-document relevance modelling, Cheng et al. [65] introduced Quality T5 (QT5), a T5-based model trained to estimate passage quality rather than query relevance. QT5 predicts whether a document is unlikely to be relevant to any query, capturing signals such as repetition or low-information content. Unlike MonoT5, QT5 is trained using prompts that exclude the query, e.g., `Document: [x] Relevant: [true/false]`. QT5 has been shown to support effective corpus pruning by removing low-quality passages prior to indexing, reducing storage and indexing costs while maintaining statistically comparable retrieval effectiveness.

9.2 Preliminary Analysis of MonoT5

Recent work [196] has introduced the notion of a *task vector*, defined as the parameter difference between a pre-trained language model (PLM) and its fine-tuned counterpart on a specific downstream task. Task vectors are commonly interpreted as capturing the knowledge acquired during fine-tuning. Inspired by these studies, we analyse the task vector between T5-base and MonoT5-base in order to address the following research question:

- RQ0: Which parameters change more when fine-tuning MonoT5?

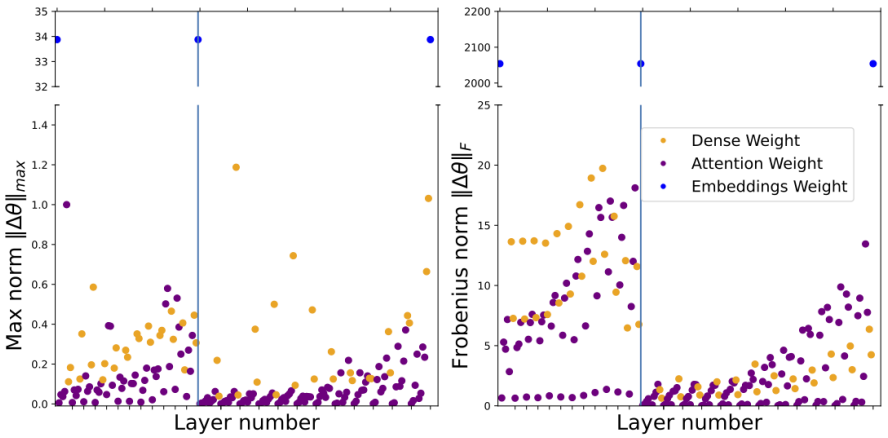

Figure 9.1: Frobenius and Max norm differences between the parameters of MonoT5 and T5. The vertical blue lines separate encoder and decoder components.

For each weight matrix $\theta_{MonoT5} \in \mathbb{R}^{n \times m}$ in MonoT5, we compute its corresponding task vector with respect to the matching parameter matrix $\theta_{T5} \in \mathbb{R}^{n \times m}$ in the original T5 model as:

$$\Delta\theta = \theta_{MonoT5} - \theta_{T5}. \tag{9.1}$$

To quantify the magnitude of these parameter shifts, we first compute the Frobenius norm [175]:

$$||\Delta\theta||_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} |(\Delta\theta)_{i,j}|^2}. \tag{9.2}$$

However, the Frobenius norm is sensitive to the dimensionality of the parameter matrix, meaning that larger matrices tend to yield larger values even when the per-parameter changes are small. This is particularly relevant in MonoT5, where parameter matrices vary significantly in size: for instance, the embedding layer has dimensions 768×32128, whereas attention matrices are typically 768×768. To obtain a complementary measure that is independent of matrix size, we also compute the Max norm [57]:

$$||\Delta\theta||_{max} = \max_{i,j} |(\Delta\theta)_{i,j}|. \tag{9.3}$$

Figure 9.1 reports the results of our analysis. We observe that the largest differences, under both Frobenius and Max norms, occur in the embedding weights of the encoder, the decoder, and the language modelling head. These components are tied in T5, meaning that they share the same parameter matrix and therefore learn a unified representation space used across encoding, decoding, and output generation. While the high Frobenius norm values can partly be attributed to the large dimensionality of the embedding layer, the dominance of these matrices

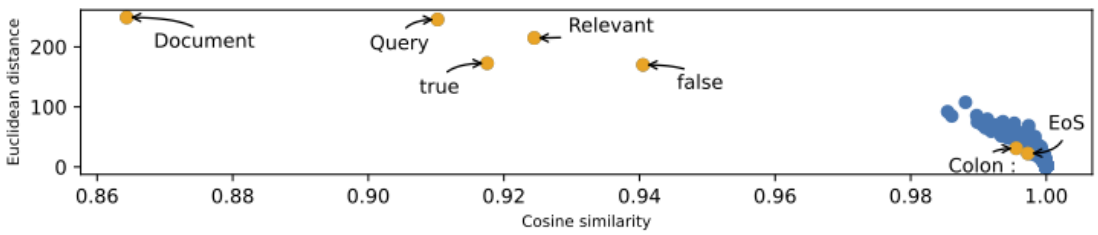

Figure 9.2: Euclidean distance and cosine similarity between MonoT5 and T5 token embeddings. Each point corresponds to a token in the vocabulary; prompt and label tokens used during MonoT5 training are highlighted in orange.

under the Max norm confirms that the most substantial fine-tuning updates are concentrated in the embedding parameters. In contrast, attention and feed-forward layers exhibit considerably smaller shifts, with Max norm values consistently below 1 and Frobenius norms below 20, indicating only limited adaptation. Thus, we can conclude that the training process in MonoT5 is dominated by changes in the embedding layer, with minimal adaptation occurring in the other parameters.

To better understand which embeddings are most affected, we further analyse token-level changes in the embedding layer. Given a token t, we compare its embedding vector in MonoT5, $e^t_{MonoT5} \in \mathbb{R}^{768}$, with the corresponding embedding in T5, $e^t_{T5} \in \mathbb{R}^{768}$, using both cosine similarity and Euclidean distance. These measures provide complementary information, as cosine similarity captures directional changes, while Euclidean distance reflects the magnitude of the update.

The results are shown in Figure 9.2. Overall, most embeddings remain highly aligned after fine-tuning, with cosine similarity values close to 1. Nevertheless, a small subset of tokens exhibits substantially larger shifts. In particular, the token with the lowest cosine similarity is `Document`. More generally, the most affected embeddings correspond to the prompt tokens `Query`, `Document`, `Relevant`, as well as the target label tokens `true` and `false`. These tokens are central to the prompt template used to formulate reranking as a text generation task, and their strong updates suggest that relevance modelling is largely encoded through these prompt-specific representations. Interestingly, the colon symbol ':' also appears among the most shifted tokens, which is consistent with its repeated use in the prompt formatting. Overall, this analysis indicates that MonoT5 fine-tuning primarily modifies the embedding representations associated with prompt and label tokens, while leaving the majority of the model parameters relatively unchanged. This observation motivates the investigation of whether effective reranking can be achieved by selectively updating only a minimal subset of T5 parameters.

9.3 Training Prompt Tokens is All You Need

In this section, we present our proposed parameter-efficient fine-tuning strategy, which leads to a lightweight T5-based reranker that we refer to as *Light-MonoT5*. We then describe how Light-MonoT5 can be combined with QT5 to reveal the learning mechanisms of MonoT5.

Formal Definition Let f_Θ denote a pre-trained T5 model parameterised by Θ. Fine-tuning f_Θ on a downstream task produces an adapted model $f_{\Theta'}$ with updated parameters Θ'. In our setting, f_Θ is used as a ranking function that, given a query q and a corpus $\mathcal{C} = \{d_i\}_{i=1}^{|\mathcal{C}|}$, assigns a relevance score to each document and induces a ranking over $\mathcal{C}$. Training is performed using triples $\langle q, d^+, d^- \rangle$, where $d^+ \in \mathcal{C}$ is a relevant document and $d^- \in \mathcal{C}$ is a non-relevant one. Classical fine-tuning updates the full parameter set Θ.

Instead, we consider a more granular perspective by partitioning the parameters into two disjoint subsets: the embedding parameters Θ_{Emb} and all remaining parameters Θ^c (including attention and feed-forward weights), such that $\Theta = \Theta_{Emb} \cup \Theta^c$.

Light-MonoT5 The analysis in Section 9.2 shows that the fine-tuning of MonoT5 is largely dominated by changes in the embedding layer, and in particular by the prompt-related token embeddings. This observation motivates a selective adaptation strategy in which we freeze all parameters Θ^c and restrict training to only a small subset of vectors in Θ_{Emb}, reducing both training cost and time. Concretely, Light-MonoT5 updates only the embeddings corresponding to the tokens used in the MonoT5 prompt template: `Query`, `Document`, `Relevant`, ':', `true`, `false`, and `[EOS]`, where `[EOS]` denotes the end-of-sequence token. All remaining parameters of the model are kept frozen throughout training. The motivation behind this approach is rooted in the nature of T5 pre-training. Since T5 has already been exposed to a wide variety of text-to-text tasks, including question answering [376], it likely contains latent semantic matching capabilities that can be activated with minimal task-specific supervision. Under this view, fine-tuning for reranking does not require rewriting the full parameter space, but rather adjusting a small number of prompt-related embeddings that define the task formulation. As a result, Light-MonoT5 can retain the full inference behaviour of MonoT5 while drastically reducing the number of trainable parameters.

Relevance vs. Quality The strong empirical similarity between MonoT5 and Light-MonoT5 raises a fundamental question: *what does MonoT5 learn during fine-tuning?* If comparable effectiveness can be achieved by updating only a small set of prompt embeddings, then MonoT5 may not primarily learn query-document relevance. Instead, we hypothesise that MonoT5 largely captures passage-level

quality signals, i.e. the ability to identify texts that are generally informative and well-formed, and to demote passages that are unlikely to be relevant to any query.

To test this hypothesis, we leverage Quality T5 (QT5) [65], a T5-based model trained to detect passages that are unlikely to be relevant to any query. We apply QT5 to a corpus $\mathcal{C} = \{p_1, \ldots, p_n\}$ to obtain a quality score for each passage. Given a pruning threshold t, we remove all passages whose quality score falls below t, obtaining a filtered corpus:

$$\mathcal{C}' = \{p \mid p \in \mathcal{C} \wedge QT5(p) \geq t\}. \tag{9.4}$$

We then index $\mathcal{C}'$ and perform retrieval using a first-stage ranker, followed by reranking with Light-MonoT5. The application of QT5 enables us to disentangle relevance modelling from quality modelling. If the combination of QT5 and Light-MonoT5 yields effectiveness comparable to MonoT5, it would suggest that MonoT5's performance is largely driven by passage quality estimation rather than fine-grained query-document matching.

9.4 Experimental Setup

Our experiments investigate the role of prompt tokens embedding in MonoT5 through the following research questions:

RQ1: Can we achieve effectiveness comparable to MonoT5 by training only prompt-token embeddings?

- (a) Does Light-MonoT5 achieve performance on par with MonoT5 on in-domain benchmarks?
- (b) Does Light-MonoT5 preserve effectiveness under domain shift?

RQ2: What is the role of passage quality in MonoT5?

Model Details and Baselines We define Light-MonoT5 by fine-tuning three T5 backbone variants [376] (small, base, and large) on the MS MARCO training triples. We follow the original MonoT5 training set up [336], using a constant learning rate of 10^{-3} for 100K iterations with class-balanced batches of size 128. Inputs are truncated to a maximum length of 512 tokens, and the decoder produces two output tokens (the prediction token and the end-of-sequence token). As in MonoT5, relevance probabilities are computed by applying the softmax only over the `true` and `false` output tokens. To study the passage quality effects, we rely on two publicly available QT5 models [65], namely QT5-small and QT5-base. In addition, following the original QT5 setting [65], we train QT5-large for 10K iterations on MS MARCO triples without any query.

As baselines, we employ a lexical ranker, namely BM25, and three different versions of MonoT5 rerankers [336], i.e. small, base and large. Moreover, we include three PEFT baselines: LoRA [187], LoRA + *Embedding layer*, and *Embedding layer* tuning. For all LoRA-based variants, we use rank $r = 8$ [187]. All PEFT baselines are trained under the same configuration as Light-MonoT5. To evaluate the efficiency of the proposed approaches, we report the number of trainable parameters (*# Params.*) and the training speed-up relative to full MonoT5 fine-tuning.

Test Collections We evaluate in-domain performance on the judged versions of the TREC Deep Learning tracks 2019 [94], 2020 [95], and DL Hard [298], all derived from the MS MARCO passage retrieval collection [328]. We refer to these test sets as DL 19, DL 20, and DL Hard, respectively, and access them through the `ir-datasets` framework [296]. To evaluate robustness under domain shift, we adopt several collections from BEIR [442], including NFCorpus [42], TREC-COVID [456], SciFact [459], SciDocs [92], and FiQA [301]. These datasets cover a range of tasks: SciDocs and SciFact focus on scientific citation prediction and fact checking, TREC-Covid and NFCorpus address biomedical retrieval, and FiQA targets financial question answering.

Evaluation Metrics We evaluate all methods using nDCG@10 [112]. Since our main goal is to determine whether Light-MonoT5 preserves the effectiveness of full fine-tuning, we perform statistical testing using both: (i) a Two One-Sided t-Test (TOST) [412] with $p < 0.05$ to test equivalence between Light-MonoT5 and MonoT5; and (ii) a paired t-test with $p < 0.05$ and Bonferroni correction to detect statistically significant differences across multiple comparisons [112].

9.5 Results

This section reports the results of our experiments and answers the research questions introduced in Section 9.4. We present results on in-domain benchmarks (DL 19, DL 20, DL Hard), out-of-domain BEIR collections, and additional analyses aimed at clarifying the role of passage quality. Results are summarised in Tables 9.1, 9.2, and 9.3. In all tables, the model before $\gg$ denotes the first-stage retriever, while the model after denotes the reranker applied to the top-100 retrieved documents. In the following, Section 9.5.1, addresses RQ1(a), which asks whether we may achieve comparable in-domain performance to MonoT5 while updating only a restricted subset of token embeddings, while Section 9.5.2 addresses RQ1(b), to study the impact of Light-MonoT5 on out-of-domain benchmarks. Finally, Section 9.5.3 investigates the role of passage quality in MonoT5's learning mechanism to answer RQ2.

Model								# Params.	Training Speed Up	DL 19	DL 20	DL Hard
BM25								/	/	.479	.494	.274
BM25 ≫ T5								0	0	.343	.231	.110
	Query	Document	Relevant	true	false	EoS	Colon (:)					
BM25 ≫ Light-MonoT5	✔	✘	✘	✘	✘	✘	✘	768	1.50×	.610	.582	.335
	✘	✔	✘	✘	✘	✘	✘	768	1.50×	.555	.516	.298
	✘	✘	✔	✘	✘	✘	✘	768	1.50×	.547	.446	.243
	✘	✘	✘	✘	✘	✔	✘	768	1.50×	.363	.255	.135
	✔	✔	✘	✘	✘	✘	✘	1536	1.48×	.669	.609	.351
	✘	✘	✘	✔	✔	✘	✘	1536	1.48×	.457	.381	.164
	✔	✔	✔	✘	✘	✘	✘	2304	1.49×	.681	.617	.344
	✔	✔	✔	✘	✘	✔	✘	3072	1.48×	.664	.624	.337
	✔	✔	✔	✔	✔	✘	✘	3840	1.47×	.686*	.639	.356
	✔	✔	✔	✔	✔	✔	✘	4608	1.47×	.696*†	.631	.347
	✔	✔	✔	✔	✔	✔	✔	5376	1.46×	.694*†	.625	.350
BM25 ≫ T5 + LoRA								885k	1.18×	.675	.630	.355
BM25 ≫ T5 + LoRA + Embedding layer								25.6M	1.12×	.690	.594	.329
BM25 ≫ T5 + Embedding layer								24.7M	1.15×	.698*†	.640	.343
BM25 ≫ MonoT5								223M	1×	.701	.679	.400

Table 9.1: Effectiveness in nDCG@10 and efficiency in terms of the number of updated parameters (#) and training speed up of all proposed configurations of Light-MonoT5-base. * and † mean not statistically significant difference compared to MonoT5 with t-test and TOST-test, respectively.

9.5.1 RQ1(a): Does Light-MonoT5 achieve performance on par with MonoT5 on in-domain benchmarks?

We begin by evaluating Light-MonoT5-base on DL 19, DL 20, and DL Hard. To isolate the contribution of individual prompt tokens, we perform a sequence of controlled experiments in which only selected embedding vectors are updated while all remaining parameters are frozen. We first fine-tune a single token embedding, e.g. `Query`, `Document`, `Relevant`, or `EOS`), then progressively expand to multiple prompt tokens until reaching the full MonoT5 prompt set: `Query`, `Document`, `Relevant`, `true`, `false`, `EOS`, and ':'. Table 9.1 reports the effectiveness of each configuration (BM25 ≫ Light-MonoT5) in terms of nDCG@10 compared to BM25, T5 and MonoT5 reranker. The green check (✔) marks the embeddings that are updated, while the grey cross (✘) indicates frozen embeddings.

Table 9.1 reports the effectiveness of each configuration in terms of nDCG@10. The results show that T5 used in a zero-shot manner performs substantially worse than BM25, confirming that some form of task adaptation is needed. However, even minimal tuning can yield strong improvements: updating only `Query` or `Document` is sufficient to outperform BM25 across all three benchmarks, while updating only `Relevant` or `EOS` leads to weak performance, often below the lexical baseline. This observation aligns with recent findings [24, 144], which suggest that pre-trained language models rely heavily on the initial instruction tokens, as, for example, T5 is fine-tuned with task-specific prompts defined at the beginning of the input [376]. Jointly updating `Query` and `Document` further improves effectiveness, while training

Model								# Params.	Training Speed Up	DL 19	DL 20	DL Hard
TCT-ColBERT								/	/	.718	.683	.371
TCT-ColBERT ≫ T5								0	0	.313	.224	.123
	`Query`	`Document`	`Relevant`	`true`	`false`	`EoS`	Colon (:)					
TCT-ColBERT ≫ Light-MonoT5-base	✔	✘	✘	✘	✘	✘	✘	768	1.50×	.651	.590	.341
	✘	✔	✘	✘	✘	✘	✘	768	1.50×	.563	.549	.306
	✘	✘	✔	✘	✘	✘	✘	768	1.50×	.527	.451	.243
	✘	✘	✘	✘	✘	✔	✘	768	1.50×	.338	.241	.121
	✔	✔	✘	✘	✘	✘	✘	1536	1.48×	.684*	.611	.347
	✘	✘	✘	✔	✔	✘	✘	1536	1.48×	.438	.360	.174
	✔	✔	✔	✘	✘	✘	✘	2304	1.49×	.711*	.631	.365*
	✔	✔	✔	✘	✘	✔	✘	3072	1.48×	.691*	.637	.351
	✔	✔	✔	✔	✔	✘	✘	3840	1.47×	.691*	.637	.364
	✔	✔	✔	✔	✔	✔	✘	4608	1.47×	.702*	.639	.375*
	✔	✔	✔	✔	✔	✔	✔	5376	1.46×	.710*	.635	.360
TCT-ColBERT ≫ T5 + LoRA								885k	1.18×	.687*	.651	.375*
TCT-ColBERT ≫ T5 + LoRA + Embedding layer								25.6M	1.12×	.670*	.608	.327
TCT-ColBERT ≫ T5 + Embedding layer								24.7M	1.15×	.697*	.648	.370*
TCT-ColBERT ≫ MonoT5								223M	1×	.717	.700	.413

Table 9.2: Effectiveness in nDCG@10 of all proposed configurations of Light-MonoT5 with TCT-ColBERT as first-stage ranker. Notation as per Table 9.1.

only the output tokens `true` and `false` fails to outperform BM25, suggesting that the reranking behaviour cannot be learned from label tokens alone. Expanding training to `Query`, `Document`, and `Relevant` yields additional gains, and including `true`/`false` enables Light-MonoT5 to reach effectiveness that is statistically equivalent to MonoT5-base on DL 19, while updating only 4608 parameters, i.e. 0.002% of the model, and achieving a 1.5× training speed-up. Including `EOS` and/or the colon token further improves performance, reaching statistical equivalence with MonoT5-base under the TOST test on DL 19. However, these configurations show weaker generalisation to DL 20 and DL Hard. Finally, we have evaluated our approach by training three new special tokens instead of `Query`, `Document`, and `Relevant`, achieving performance on par with MonoT5 (.647 in nDCG@10 on DL 19), suggesting that the specific semantic meaning of the prompt tokens is not critical. We also compare against standard PEFT baselines: LoRA, LoRA+*Embedding layer*, and full *Embedding layer* tuning. While these methods achieve competitive effectiveness, with the embedding-layer baseline showing particularly strong results on DL 20, they require substantially more trainable parameters and at least 1.3× longer training compared to Light-MonoT5.

Finally, extend our analysis to a stronger semantic first-stage dense retriever, TCT-ColBERT [264]. As shown in Table 9.2, Light-MonoT5 preserves the same effectiveness trends, and when training the full prompt-token set, it achieves nDCG@10 values that are statistically equivalent to MonoT5. These results show that training only prompt-token embeddings is sufficient to capture the core relevance signals learned by MonoT5, even when the first-stage retriever already provides semantically rich passages.

Model	DL 19	DL 20	DL Hard	NFCorpus	SciDocs	SciFact	TREC-Covid	FiQA
BM25	.479	.494	.274	.328	.158	.684	.629	.253
BM25 ≫ Light-MonoT5-small	.629*	.540	.298	.319	.118	.628	.632	.277
BM25 ≫ MonoT5-small	.673	.640	.354	.334	.146	.672	.694	.344
BM25 ≫ Light-MonoT5-base	.696*†	.631	.347	.341*	.145	.707*	.723*	.354
BM25 ≫ MonoT5-base	.701	.679	.400	.347	.157	.720	.725	.386
BM25 ≫ Light-MonoT5-large	.685*	.665*	.350*	.359*	.162	.718	.729*	.391
BM25 ≫ MonoT5-large	.711	.685	.359	.367	.174	.737	.759	.421

Table 9.3: Effectiveness in nDCG@10 of Light-MonoT5 on out-of-domain benchmarks. Notation for * and † as per Table 9.1.

Overall, to concretely answer RQ1(a), Light-MonoT5 match MonoT5 performance on in-domain benchmarks, showing that training prompt-token embeddings is sufficient to reproduce the effectiveness of full MonoT5 fine-tuning.

9.5.2 RQ1(b): Does Light-MonoT5 preserve effectiveness under domain shift?

To understand if our approach can generalise well on different T5 variants and out-of-domain datasets, we evaluate Light-MonoT5 under domain shift by training three variants (small, base, and large) and testing them on BEIR collections. Based on the in-domain results, we adopt the strongest configuration, which updates `Query`, `Document`, `Relevant`, `true`, `false`, and `EOS`. As shown in Table 9.3, Light-MonoT5-base and Light-MonoT5-large maintain strong out-of-domain performance, achieving effectiveness that is statistically equivalent to MonoT5 on NFCorpus, SciFact, and TREC-COVID. In contrast, Light-MonoT5-small degrades substantially on most out-of-domain benchmarks, showing that limited model capacity amplifies the effect of vocabulary and distribution shifts. To concretely answer RQ1(b), Light-MonoT5-base and Large preserve MonoT5's robustness under domain shift, while the small variant is less reliable, suggesting that prompt-token adaptation alone may be insufficient when model capacity is limited.

9.5.3 RQ2: What is the role of passage quality in MonoT5?

The strong performance of Light-MonoT5 shows that T5 can learn effective relevance judgements even when training on MS MARCO is restricted to a small and carefully selected subset of parameters. This suggests that updating the whole parameters may not be necessary to learn retrieval-specific relevance signals. Indeed, the generalisation ability of T5 can be partly attributed to its prior fine-tuning on different supervised tasks such as question answering (e.g., SQuAD [381]) and semantic similarity (e.g., STS-B [61]). These tasks require modelling semantic relationships and contextual alignment, which is similar to query-document matching.

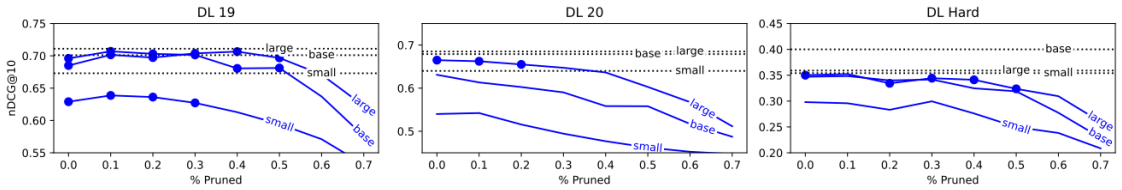

Figure 9.3: nDCG@10 of Light-MonoT5 by the percentage of a corpus pruned by QT5. Effectiveness measurements that are not statistically significantly different from MonoT5 are marked with •.

As a result, T5 likely already encodes much of the capability required for relevance estimation, even before being fine-tuned on MS MARCO. Building on this insight, RQ2 investigates what MonoT5 actually learns during full fine-tuning on MS MARCO. Unlike Light-MonoT5, MonoT5 updates the entire set of 223M parameters, which raises the possibility that its effectiveness improvements do not primarily derive from learning fine-grained query-document relevance, but rather from acquiring a strong ability to discriminate passage quality. To test this hypothesis, we use QT5 as a document filter, pruning low-quality passages from MS MARCO before indexing with a first-stage retriever. We then apply Light-MonoT5 on the pruned corpus: if its performance are on par to fully fine-tuned MonoT5, this would support the interpretation that full fine-tuning mainly enhances the model's capacity to identify low-quality passages. Following Chen et al. [65], we prune up to 70% of the corpus, since pruning beyond this threshold substantially changes the index size due to lexicon effects. The results, reported in Figure 9.3, show that both Light-MonoT5-base and Light-MonoT5-large achieve effectiveness on DL 2019 that is not statistically significantly different from their fully fine-tuned MonoT5 counterparts, even when up to 50% of the MS MARCO passages are removed. A similar pattern is observed on DL Hard, particularly for the base model. In contrast, Light-MonoT5-small reaches statistical equivalence with MonoT5 only on DL 20 and only for pruning levels up to 30%, suggesting that smaller models may rely more heavily on explicit relevance learning.

Overall, these findings show that once low-quality passages are removed, extensive parameter updates are no longer needed to retrieve highly relevant documents. This reinforces our hypothesis that full fine-tuning MonoT5 learns how to classify passage quality rather than query-document relevance, thereby answering RQ2.

9.6 Conclusions

In this Chapter, we analysed the fine-tuning dynamics of MonoT5 showing that the largest parameter updates are focused in a small subset of prompt-token embeddings, while the majority of the model remains largely unchanged. Motivated by

this observation, we introduced Light-MonoT5, a parameter-efficient reranker that fine-tunes only prompt-related embedding vectors, reducing the number of trainable parameters to 0.002% while preserving effectiveness across multiple benchmarks. Our experiments show that Light-MonoT5 achieves effectiveness statistically equivalent to MonoT5 on in-domain evaluations, and that base and large variants retain strong transfer performance under domain shift. Furthermore, by combining Light-MonoT5 with QT5, we show that MonoT5 learns to classify document quality rather than query-passage relevance.

Part IV

Resources and Benchmarks

Chapter 10

Introduction

This part of the thesis introduces and motivates the development of new resources and benchmarks aimed at advancing personalised and domain-specific search evaluation across heterogeneous application scenarios. Although recent advances in parameter-efficient adaptation and large language models have reduced the reliance on full-scale retraining, effective fine-tuning of Information Retrieval (IR) systems still fundamentally depends on the availability of high-quality labelled data. In practice, the development of domain-specific and Personalised Information Retrieval (PIR) models is constrained by a lack of large-scale annotated datasets that capture both semantic relevance and user-specific signals.

To address this limitation, this part of the thesis focuses on bridging this gap by introducing novel datasets designed to support robust training and evaluation of PIR systems. The overarching goal is to provide structured, richly annotated resources that reflect the complexity of real-world interactions and enable systematic benchmarking of personalization methods in IR. Chapter 11 introduces SE-PQA, a large-scale dataset for Personalised Community Question Answering (cQA). The dataset comprises over one million queries and two million answers, enriched with fine-grained annotations capturing social and interactional signals extracted from a large-scale community-based QA platform. In addition to describing the dataset construction process, the Chapter establishes a set of baseline models leveraging deep learning architectures and personalization mechanisms tailored to the cQA setting. Empirical results show that incorporating personalization consistently improves retrieval and ranking performance, particularly in terms of robustness and generalization when models are exposed to heterogeneous user communities. These findings highlight the relevance of community-aware signals for effective PIR modelling.

Building on this resource, Chapter 12 presents Sy-SE-PQA, a synthetic extension of SE-PQA designed to mitigate the scarcity of human-annotated data. In this Chapter, we explore the use of large language models (LLMs) to generate synthetic personalised answers conditioned on user–query interactions from the original

dataset. This approach is motivated by the observation that, in many realistic settings, manually annotated personalised relevance judgments are either expensive or unavailable. Experimental results show that, despite approximately 30% of generated synthetic documents containing hallucinated or partially incorrect content, models trained on synthetic data achieve performance comparable to those trained on human-annotated datasets. This suggests that LLM-generated supervision can serve as an alternative when gold-standard annotations are limited.

Overall, this part of the thesis provides the foundational data resources necessary to support the development and evaluation of advanced PIR models. While Part II investigated parameter-efficient fine-tuning strategies and Part III explored training-free adaptation methods for domain-specific and task-oriented use of large language models, Part IV complements these contributions by introducing the datasets required to rigorously evaluate domain-specific and personalised approaches.

Chapter 11

SE-PQA: Personalized Community Question Answering

Personalization is a central research topic in both Information Retrieval (IR) [40, 58, 428, 190, 294, 49] and Natural Language Processing (NLP) [49]. In personalised search, the goal is to adapt retrieval results to a specific user or group of users by leveraging information about their interests, preferences, and online behaviour. Given the ability of Deep Neural Networks (DNNs) in learning complex representations from both text and structured information [314], there is strong interest in applying such models to Personalised IR (PIR) and Recommender Systems (RS). However, the lack of publicly-available, large-scale datasets that include user-related information is one of the biggest obstacles to the training and evaluation of DNN-based personalised models. Several real-world datasets have been used to study personalization, including the AOL query log [347], the Yandex query log, and the CIKM Cup 2016 dataset. In addition, some synthetically enriched datasets have been proposed, such as PERSON [435], the Amazon product search dataset [7], and a dataset derived from the Microsoft Academic Knowledge Graph [29]. Despite their usefulness, these resources present significant limitations. For instance, the AOL query log raises serious ethical and privacy concerns [23], while the strong anonymization applied to the Yandex query log makes it unsuitable for training or fine-tuning neural language models.

This Chapter addresses these limitations by introducing `SE-PQA` (StackExchange – Personalized Question Answering), a large-scale dataset enriched with user-level signals that enable the training and evaluation of personalised models for community Question Answering (cQA). `SE-PQA` is derived from StackExchange, a widely used cQA platform composed of 178 public communities. The StackExchange data dump of user-generated content is publicly available under the CC BY-SA 4.0 license. We process the original dump and construct `SE-PQA` as a curated resource comprising approximately one million questions and two million answers, enriched

with extensive metadata capturing social and behavioural aspects of user interactions. These include, among others, upvotes and downvotes on questions and answers, view counts, favourite counts, post tags describing topical content, and user comments. In addition, to support personalization, each user is associated with their historical activity (previous questions and answers), self-reported biography, reputation score, and profile-level statistics such as profile view counts. The cQA problem on SE-PQA can be formulated using different information sources, including textual content, tags, social signals, or their combinations. In this work, we focus on information retrieval-based formulations of cQA, modelling the task as ad-hoc retrieval where a question serves as a query and answers are retrieved from a pre-existing collection. The objective is to produce a ranked list of answers that best address the information need expressed in the question. Since multiple answers may be relevant for the same query, personalization can be incorporated to exploit user-specific context and preferences, thereby favouring responses that better align with individual user profiles. In summary, the main contributions of this Chapter are:

- We introduce SE-PQA, a publicly available dataset containing over one million questions and two million answers generated by approximately 600k users, together with rich metadata that enables both standard and personalised cQA research.
- We provide an in-depth analysis of SE-PQA and compare it against existing datasets used in personalised information retrieval studies.
- We present initial experimental results on SE-PQA, showing that neural retrieval methods outperform traditional approaches and that incorporating personalization signals leads to substantial performance improvements.

The remainder of this Chapter is organised as follows. Section 11.1 introduces SE-PQA, presents its statistics, defines the supported cQA tasks, and compares it with related datasets. Section 11.2 reports preliminary experimental results using both classical and personalised retrieval models. Finally, Section 11.3 concludes the Chapter.

11.1 The SE-PQA Dataset

StackExchange posts are enriched with a wide range of social and interaction signals. When posting a question, users typically attach tags that indicate its topic, thereby facilitating retrieval and improving discoverability for other users with related interests. In addition, questions may receive upvotes or downvotes reflecting their perceived usefulness and compliance with community standards. Community members can also contribute feedback aimed at improving questions

that are unclear or poorly formulated. Similar treatment is given to the answers: they can be up-voted or down-voted, and the user who asked the question can explicitly mark one response as the best answer. This choice does not necessarily coincide with the answer that received the highest number of up-votes. Overall, we find that 87.6% of questions and answers are associated with a score computed as the difference between the number of upvotes and downvotes. A positive value indicates that a post received more favourable than unfavourable feedback, whereas a negative value reflects the opposite.

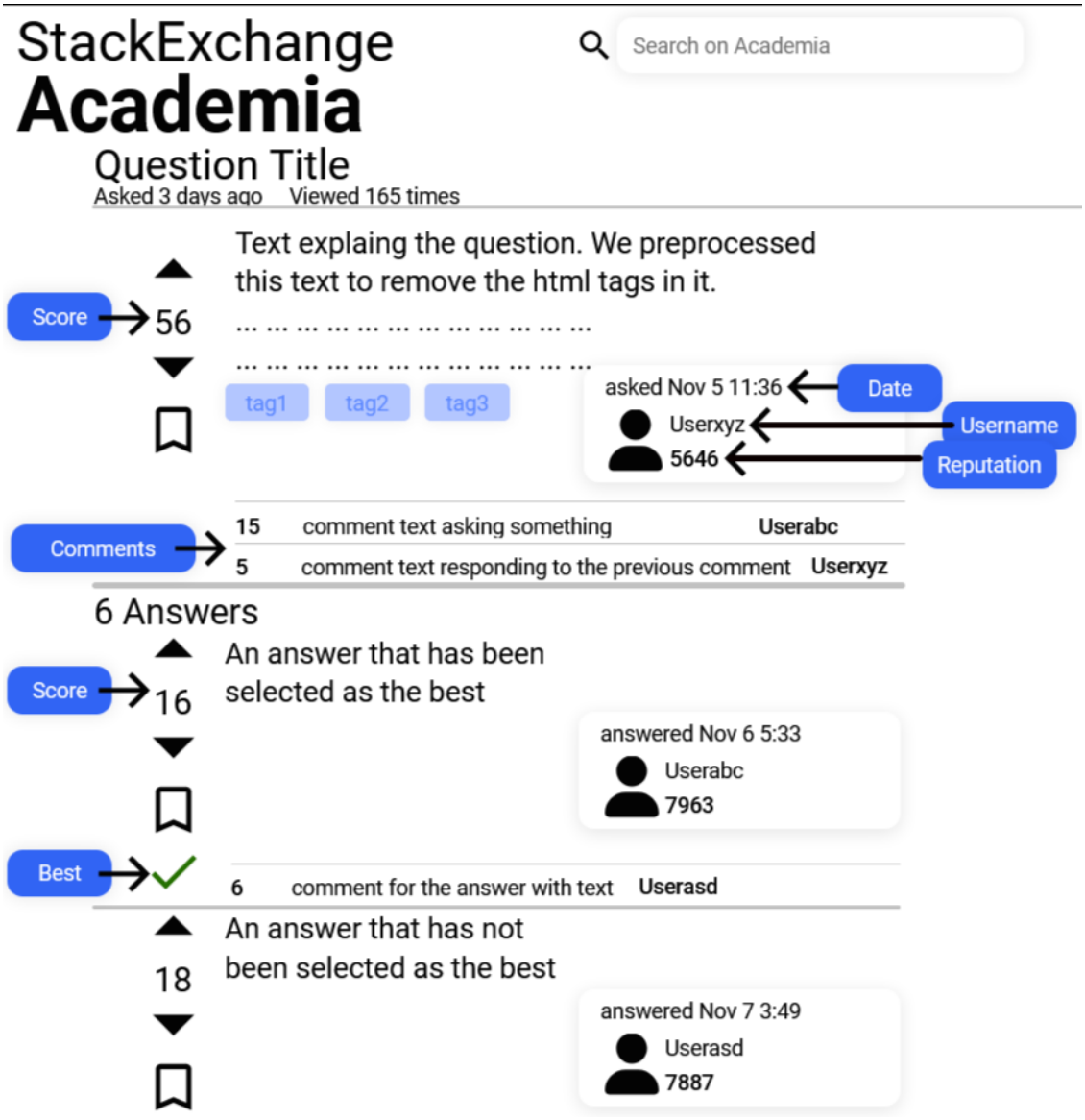

Figure 11.1: Illustration of StackExchange data.

StackExchange data has previously been utilised within the information retrieval community, for instance as a source of training data for sentence similarity models [194]. However, its use in Q&A research has largely been limited to deriving text pairs, without exploiting the rich social and user-level information that could support personalised retrieval. Other work has used StackExchange for duplicate question retrieval [181], but such datasets typically focus on a limited number of communities and do not study cross-community interactions. In contrast, SE-PQA offers a unified, curated collection of questions and answers gathered from multiple heterogeneous forums. A notable characteristic of this dataset is that users are not restricted to a single community and may participate across different domains. Among users who have asked at least two questions, approximately 50% have contributed to more than one community. This proportion increases when focusing on more active users. For instance, among the 62k users with at least five posts (questions or answers), only 23k (37%) are active in a single community, while 40k (63%) participate in at least two communities, 26k (42%) in at least three, and 18k (28%) in more than three communities. We claim that personalization becomes particularly valuable in such multi-domain settings, where user interests span multiple topics. In contrast, when retrieval is performed within a single domain or topic, personalization signals may have a weaker impact. We provide empirical evidence supporting this claim in Section 11.2.2, where we show that personalization yields larger improvements when applied to the full multi-domain dataset compared to isolated single-community subsets. To promote diversity while maintaining linguistic consistency, SE-PQA combines data from multiple communities broadly belonging to humanistic and lifestyle-related domains. In particular, we include the following 50 StackExchange communities:
writers, workplace, woodworking, vegetarianism, travel, sustainability, sports, sound, skeptics, scifi, rpg, politics, philosophy, pets, parenting, outdoors, opensource, musicfans, music, movies, money, martialarts, literature, linguistics, lifehacks, law, judaism, islam, interpersonal, hsm, history, hinduism, hermeneutics, health, genealogy, gardening, gaming, freelancing, fitness, expatriates, english, diy, cooking, christianity, buddhism, boardgames, bicycles, apple, anime, academia.

To avoid data leakage, we construct training, validation, and test splits using a temporal partition. The training set contains questions posted between *2008-09-10* and *2019-12-31* (inclusive), the validation set contains questions posted between *2019-12-31* and *2020-12-31* (inclusive), and the test set contains questions posted between *2020-12-31* and *2022-09-25* (inclusive). Overall, the dataset contains 1,125,407 questions, of which 1,001,706 (89%) are associated with at least one answer. Among these, 525,030 questions (47% of the total) have an answer explicitly marked as the best by the question author. After applying the temporal split, we obtain 822,974 training questions, 78,854 validation questions, and 99,878 test questions. In total, SE-PQA consists of 2,173,139 answers and 588,688 users. User

	Type	mean	std	median	25%	75%	99%
Questions	Length	125.69	112.90	94	60	153	553
	Score	5.13	10.73	2	1	6	45
#docs =	Answers Count	1.93	1.92	1	1	2	9
1,125,407	Comments Count	2.78	3.37	2	0	4	15
	Favorite Count	2.07	4.88	1	1	2	16
	Tags Count	2.45	1.21	2	1	3	5
Answers	Length	178.15	210.09	117	61	218	1000
#docs =	Score	5.13	12.43	2	1	5	51
2,173,139	Comments Count	1.62	2.62	1	0	2	12

Table 11.1: Feature statistics for questions and answers.

activity is highly skewed: the median user contributes only a single post (question or answer), and around 80% of users publish no more than two posts in total. All textual content is cleaned by removing HTML markup present in the original posts. Table 11.1 reports aggregate statistics, including document length (in words), post score (computed as upvotes minus downvotes), number of answers per question, number of comments, favourite counts, and tag counts. Figure 11.2 further shows that posts are generally short, with answers tending to be longer than questions on average. The dataset is publicly released on Zenodo.

11.1.1 Task Definition

While SE-PQA enables multiple information retrieval tasks, including duplicate question detection, related question retrieval, and expert finding, in this Chapter we focus on community Question Answering (cQA). The objective of cQA is to address an information need expressed in a user question by retrieving relevant answers from a historical repository of community responses. We derive relevance signals from community feedback, primarily through upvote counts. For experiments involving personalised cQA models, we adopt a stricter criterion and consider only the answer explicitly selected as the best by the question author as relevant. Formally, let $\mathcal{A} = a_1, \ldots, a_n$ denote the set of historical answers and let $\mathbf{q}$ be a question issued by user $\mathbf{u}$. The task is to retrieve a ranked list of k answers $a_{q,1}, \ldots, a_{q,k} \subset \mathcal{A}$ that are most relevant to $\mathbf{q}$. In the personalised setting, relevance is reduced to a single ground-truth item $a_{q,u} \in \mathcal{A}$, corresponding to the answer selected as best by the question author. To support this formulation, we filter the answer collection by removing all answers with negative scores, as these are likely to be of low quality and not suitable for retrieval. This step eliminates approximately 100k answers, leaving 2,073,370 answers. We also discard questions that have no associated answers.

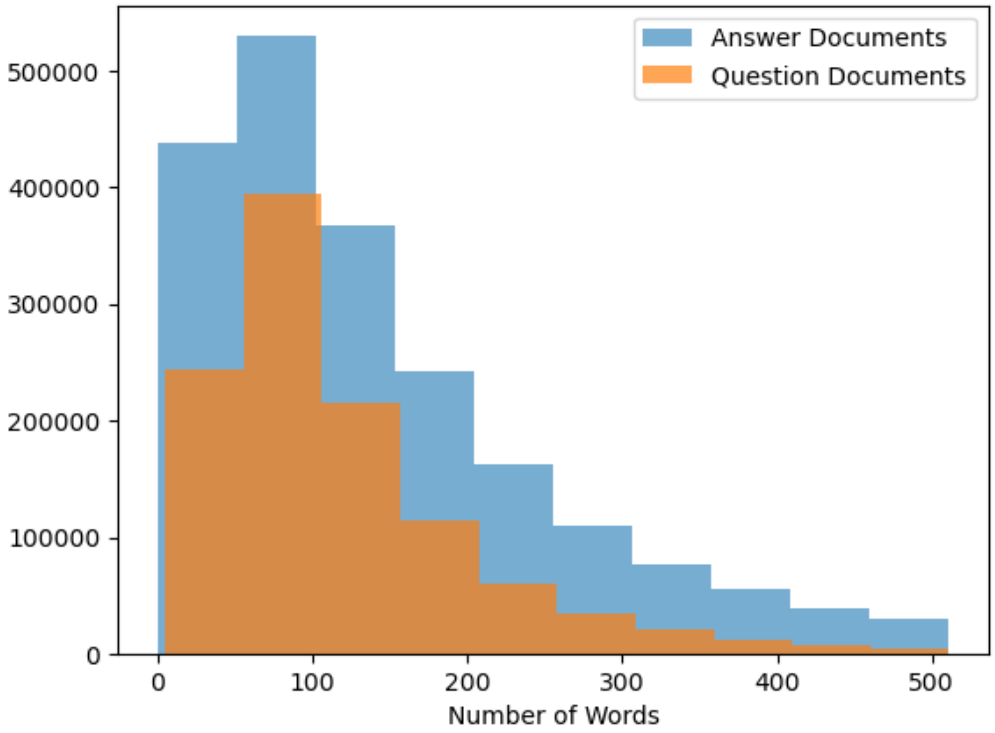

Figure 11.2: Word length distribution for questions and answers.

Relevance judgments are then constructed based on the remaining question–answer associations. Out of 1,001,706 questions, 525,030 include an answer marked as best by the user. Since negative-scored answers have already been removed, these selected answers also exhibit positive community feedback. As a result, the answer associated with a question **q** and user **u** can be interpreted both as relevant to the query (due to community endorsement) and aligned with the user's preference (due to explicit selection). Based on this, we define two dataset variants: a base setting, where all answers with positive scores are treated as relevant, and a personalised (pers) setting, where only the user-selected best answer is considered relevant for each query. Both formulations support cQA, but they differ in the availability of explicit user preference signals: the pers variant provides direct user-level supervision, while the base variant includes many queries where such explicit signals are absent, limiting its suitability for training personalised models. For personalization purposes, `SE-PQA` also provides rich user-level information that can be exploited during training and inference. For each query, we include the historical activity of the corresponding user prior to the query timestamp, including earlier questions and

Dataset	Documents	Train Queries	Val Queries	Test Queries	Users	Avg. user docs	Avg. # relevant	Rel. Assessment
AOLIA[295]	1 291 695	212 386	31 064	36 052	30 166	136.62 ± 134.17	1.15 ± 0.46	inferred from click
PERSON[435]	616 889	-	-	-	558 898	-	5.5 ± 5.3	inferred from citation
MAG Computer Science[29]	4 809 684	552 798	5 583	6 497	5 260 279	61.94 ± 60.32	3.25 ± 3.27	inferred from citations
MAG Physics	4 926 753	728 171	7 355	6 366	5 835 016	60.98 ± 56.54	4.17 ± 4.15	inferred from citations
MAG Political Science	4 814 084	162 597	1 642	5 715	6 347 092	40.64 ± 29.32	3.88 ± 5.17	inferred from citations
MAG Psychology	4 215 384	544 882	5 503	12 625	4 825 578	61.66 ± 62.72	4.73 ± 4.4	inferred from citations
Amazon Electorincs[7]	1 689 188	904	-	85	192 403	8.78 ± 8.26	1.12 ± 0.48	synthetic
Amazon Kindle Store	982 618	3 313	-	1 290	68 223	35.65 ± 37.48	1.87 ± 3.3	synthetic
Amazon CDs	1 097 591	534	-	160	75 258	21.75 ± 16.53	2.57 ± 6.59	synthetic
Amazon Cell Phones	194 439	134	-	31	27 879	4.95 ± 2.6	1.52 ± 1.13	synthetic
SE-PQA [Base]	2 073 370	822 974	78 854	99 878	588 688	34.71 ± 107.33	2.07 ± 1.81	inferred from answer scores
SE-PQA [Pers]	2 073 370	224 366	18 086	19 811	17 963	75.67 ± 142.41	1	manually selected by user

Table 11.2: Comparison of SE-PQA with other English text-based datasets for personalized information retrieval.

answers, in order to avoid temporal leakage. The dataset further includes social interaction signals, tag histories, badge information, and user biographies (about me sections). In addition, it provides numerical attributes such as reputation, upvote and downvote counts, and view statistics, as well as temporal metadata including account creation time, last activity, and post timestamps.

11.1.2 Comparison with available datasets

We next examine existing datasets that are frequently adopted in personalised information retrieval research and highlight their main limitations. These datasets cover diverse application scenarios, including web search, product search, and academic retrieval. Table 11.2 reports key statistics for several widely used benchmarks in the personalised IR literature. The AOL query log, originally released in 2006, remains one of the most commonly used resources despite well-documented privacy concerns [23]. It contains approximately 20 million web queries issued by more than 657,000 users over a three-month period (03/01/2006 to 05/31/2006). Each record includes an anonymised user identifier, a clicked URL (when available), and the rank position of the clicked result. A major limitation of this dataset is that it provides only URLs rather than the actual textual content of the retrieved web pages. To mitigate this issue, a later effort constructed a document collection in 2017 by crawling the pages associated with the URLs [6]. However, this introduces temporal inconsistencies, as web pages may have substantially changed between 2006 and 2017. More recently, an updated version was proposed using the Internet Archive to retrieve page snapshots closer to their 2006 content [295]. The AOLIA dataset [295] is a cleaned and refined version of the original AOL query log [347], designed to improve data quality. Queries without clicks are removed, since they do not provide relevance signals. Queries containing explicit domain indicators, such as .com or .org, are filtered out, along with those related to adult or illegal content. Extremely short queries (fewer than three characters) are also discarded, and users with fewer

than twenty queries are excluded to ensure the availability of sufficient historical interaction data. After preprocessing, the dataset contains roughly 1.3M documents and around 30k users. Given the limitations of real-world logs, several synthetic personalised IR datasets and evaluation frameworks have been introduced, including PERSON [435], the Amazon product search dataset [7], and datasets derived from the Microsoft Academic Graph (MAG) [29]. PERSON is a synthetic benchmark built on the ArnetMiner citation network [439], where the title or abstract of a paper is treated as a query and its cited papers are assumed relevant. Author publication histories are used to represent user profiles. A similar strategy is adopted in MAG-based datasets, which exploit a larger citation graph to generate multiple discipline-specific collections (e.g., Political Science, Psychology, Computer Science, and Physics). In both cases, authors with fewer than twenty prior publications are removed to ensure sufficient profile context. As discussed in Tabrizi et al. [435], such datasets are useful for studying user modelling approaches, but they are less suitable for evaluating personalised retrieval effectiveness, since relevance is derived from strong and potentially unrealistic assumptions. The Amazon product search dataset [7], built on the Amazon Review corpus [166], is also constructed synthetically. Queries are generated by concatenating product category terms along the hierarchical taxonomy. Stopwords and repeated terms are removed, and lower-level category terms are prioritised, e.g. *Camera* → *Photo* → *Digital Camera Lenses* becomes *"photo digital camera lenses"*. Given a product i purchased by a user u, the product is treated as relevant for u, and the corresponding synthetic query is generated as described above. Nevertheless, this process has important limitations: it produces a limited variety of unique queries and does not reflect realistic search behaviour, as users rarely issue queries that explicitly list a full taxonomy path. As shown in Table 11.2, SE-PQA is comparable in scale to existing benchmarks in terms of corpus size and user population, while offering the largest number of queries among the considered datasets. Moreover, it is the only resource in which relevance judgments are explicitly provided by users: the best answer is selected directly by the question author, while additional relevance signals are implicitly captured through community voting behaviour. This combination makes SE-PQA a valuable benchmark for studying personalised retrieval under realistic, community-driven conditions.

11.2 Preliminary experiments with SE-PQA

This section describes the experimental setup and the retrieval methods adopted to show the applicability of SE-PQA to the community Question Answering task defined in Section 11.1.1. We then report and discuss the results of a set of preliminary experiments.

11.2.1 Experimental settings

We employ a two-stage ranking pipeline designed to balance effectiveness and efficiency by combining a fast retrieval component with a more accurate but computationally heavier re-ranking stage. The first stage is recall-oriented and retrieves an initial pool of candidate documents for each query, while the second stage focuses on precision by re-ranking these candidates using more expressive models. In the first stage, we use BM25 as a lightweight lexical retrieval method. To maximise recall, BM25 hyperparameters are tuned via grid search by optimising Recall@100 on a set of 5,000 queries randomly sampled from the validation split. The best-performing configuration corresponds to $b = 1$ and $k1 = 1.75$. In the second stage, we re-rank the top 100 BM25-retrieved answers using a linear combination of multiple scoring signals, including the BM25 score, a neural re-ranking score, and, when enabled, a personalization score derived from user history based on tag information.

Neural models

We evaluate the following neural re-ranking approaches in the second stage:

- **MiniLM.** We use MiniLM [390], a sentence-transformer model trained on large-scale text pair data. Since its pre-training corpus includes StackExchange content, we directly apply MiniLM without additional fine-tuning.
- **DistilBERT.** We fine-tune DistilBERT [409] on the training queries of SE-PQA. For each query, we sample one positive answer and two negative examples: one retrieved by BM25 and one random in-batch negative [172]. The model is trained for 10 epochs using Triplet Margin Loss [390], with margin $\gamma = 0.5$, batch size 16, and learning rate 10^{-6}.
- **MonoT5.** We adopt MonoT5 [336], a cross-encoder re-ranking model based on T5 [377], originally fine-tuned on MS MARCO. We evaluate both MonoT5-small and MonoT5-base. MonoT5-small is fine-tuned following the training configuration proposed in [336], using batch size 128 and learning rate 10^{-3}. For each query, one positive and one negative document are sampled from the BM25 candidate set. Due to computational constraints, MonoT5-base is trained with batch size 64. Both models are fine-tuned for 10 epochs. Instead of updating all model parameters, we train Pfeiffer adapter modules [365]. In line with previous work on T5-based adaptation [217], we set the adapter bottleneck dimension to 48.

For DistilBERT and the T5-based models, we use AdamW [284] as the optimiser. For reproducibility, the random seed is fixed to 42 for DistilBERT training and to 0 for MonoT5 training.

Personalised TAG model for cQA

To incorporate personalization, we introduce a simple tag-based user modelling strategy. For each candidate answer **a** retrieved for a question **q** submitted by user **u**, we compute a personalization score that is subsequently combined with the BM25 and neural ranking signals. Let **q** be posted at time **t** by user **u**. We define $T_{u,t}$ as the set of tags assigned by user **u** to all questions posted prior to time **t** (including **q**). This tag set serves as a representation of the user's historical topical interests. Since answers are not directly associated with tags on StackExchange, we represent each answer **a** authored by user $\mathbf{u}'$ using a tag set $T_{u',t}$, defined as the set of tags associated with questions answered by $\mathbf{u}'$ before time **t** (excluding the current question **q**). Excluding **q** prevents query-level information leakage. The tag-based personalization score for answer **a** is defined as:

$$s_a = \frac{|T_{u',t} \cap T_{u,t}|}{|T_{u,t}| + 1},$$

where the additive constant +1 is introduced as smoothing to handle cases in which $T_{u,t}$ is empty. The underlying intuition is that answers authored by users with similar topical interests to the question author should be ranked more highly.

Score combination

Final rankings are obtained by combining normalised scores through a weighted linear interpolation. Specifically, we combine the BM25 score, the neural model score (MiniLM, DistilBERT, or T5), and the TAG-based personalization score using weights λ_{BM25}, λ_{Neural}, and λ_{TAG}, respectively, subject to the constraint $\sum_i \lambda_i = 1$. These weights are tuned on the validation split using grid search over the interval [0,1] with step size 0.1.

Evaluation Metrics

Retrieval performance is evaluated using P@1, nDCG@3, nDCG@10, Recall@100, and MAP@100. These relatively low cutoffs are well suited to cQA scenarios, where placing the most relevant answers at the top of the ranking is particularly important. Statistical significance is evaluated using a Bonferroni-corrected two-sided paired Student's t-test with a 99% confidence level. All metrics are computed using the *ranx* library [25, 26].

11.2.2 Experimental Results

Table 11.4 reports the experimental results obtained on the *base* and *pers* variants of the dataset, respectively. In both tables, the symbol * indicates statistically significant improvements over the corresponding non-personalised baseline, i.e., the

Model	P@1	NDCG@3	NDCG@10	R@100	MAP@100	λ
BM25	0.330	0.325	0.359	0.615	0.320	-
BM25 + TAG	0.355*	0.349*	0.383*	0.615	0.342	(.7;.3)
BM25 + DistilBERT	0.404	0.400	0.435	0.615	0.389	(.3;.7)
BM25 + DistilBERT + TAG	0.422*	0.415*	0.448*	0.615	0.402*	(.3;.5;.2)
BM25 + MiniLM	0.473	0.459	0.486	0.615	0.443	(.1;.9)
BM25 + MiniLM + TAG	0.493*	0.475*	0.500*	0.615	0.457*	(.1;.8;.1)
BM25 + T5-small + TAG	0.448	0.442	0.471	0.615	0.426	(.1;.9;.0)
BM25 + T5-base + TAG	**0.497**	**0.491**	**0.514**	0.615	**0.470**	(.1;.9;.0)
Model	P@1	NDCG@3	NDCG@10	R@100	MAP@100	λ
BM25	0.279	0.353	0.394	0.707	0.362	-
BM25 + TAG	0.306*	0.383*	0.425*	0.707	0.392*	(.7;.3)
BM25 + DistilBERT	0.351	0.437	0.478	0.707	0.441	(.3;.7)
BM25 + DistilBERT + TAG	0.375*	0.460*	0.500*	0.707	0.463*	(.3;.5;.2)
BM25 + MiniLM	0.403	0.491	0.525	0.707	0.490	(.1;.9)
BM25 + MiniLM + TAG	0.426*	0.512*	0.543*	0.707	0.509*	(.1;.8;.1)
BM25 + T5-small	0.376	0.469	0.506	0.707	0.468	(.1;.9)
BM25 + T5-small + TAG	0.400*	0.491*	0.525*	0.707	0.489*	(.1;.8;.1)
BM25 + T5-base	0.417	0.517	0.548	0.707	0.510	(.1;.9)
BM25 + T5-base + TAG	**0.440***	**0.535***	**0.563***	0.707	**0.528***	(.1;.8;.1)

Table 11.4: Performance on the cQA task using the Pers split of SE-PQA.

same retrieval model without the TAG component. The column λ reports the optimal weights used for combining BM25, the neural re-ranking score, and the TAG-based personalization signal.

Overall, the results confirm that neural re-ranking substantially outperforms BM25. Among the evaluated models, MiniLM achieves stronger performance than DistilBERT and MonoT5-small, which is expected given that MiniLM benefits from large-scale pre-training on extensive sentence-pair supervision [194]. After task-specific fine-tuning, MonoT5-base yields the best overall results, surpassing MiniLM across most metrics. Notably, after only 10 training epochs on SE-PQA, MonoT5-small and MonoT5-base improve MAP@100 over BM25 by 33% and 47%, respectively. The central observation is that incorporating TAG-based personalization consistently improves effectiveness across all neural re-ranking methods. In particular, on the *pers* dataset variant (Table 11.4), where queries without explicit personalization signals are removed, the TAG component produces statistically significant gains across all evaluation metrics and for every model. The improvements obtained through this personalization strategy reach up to 8% MAP@100 compared to the corresponding non-personalised baselines, highlighting the strong potential of user-aware modelling in community question answering.

Finally, to test the hypothesis that personalization is more beneficial in heterogeneous multi-domain collections than in single-domain settings, we conduct additional experiments on community-specific subsets of SE-PQA. Concretely, we partition the dataset into 50 single-community collections (considering only the *base*

Community	Model (BM25 +)	P@1	NDCG@3	NDCG@10	R@100	MAP@100	λ
Academia	MiniLM	0.438	0.382	0.395	0.489	0.344	(.1,.9)
	MiniLM + TAG	**0.453***	**0.392***	**0.403***	0.489	**0.352***	(.1,.8,.1)
Apple	MiniLM	0.327	0.351	0.381	0.514	0.349	(.1,.9)
	MiniLM + TAG	**0.335***	**0.361***	**0.389***	0.514	**0.357***	(.1,.8,.1)
Bicycles	MiniLM	0.405	0.380	0.421	0.600	0.365	(.1,.9)
	MiniLM + TAG	**0.436***	**0.405***	**0.441***	0.600	**0.386***	(.1,.8,.1)
Christianity	MiniLM	0.534	0.505	0.555	0.783	0.497	(.2,.8)
	MiniLM + TAG	**0.549***	**0.521***	**0.564***	0.783	**0.507***	(.1,.8,.1)
Cooking	MiniLM	0.600	0.567	0.600	0.719	0.553	(.1,.9)
	MiniLM + TAG	**0.619***	**0.583***	**0.614***	0.719	**0.568***	(.1,.8,.1)
DIY	MiniLM	0.323	0.313	0.346	0.501	0.302	(.1,.9)
	MiniLM + TAG	**0.335***	**0.324***	**0.356***	0.501	**0.312***	(.1,.8,.1)
Hermeneutics	MiniLM	0.589	0.538	0.593	0.828	0.526	(.2,.8)
	MiniLM + TAG	**0.632***	**0.570***	**0.617***	0.828	**0.552***	(.1,.8,.1)
Law	MiniLM	0.663	0.647	0.678	0.803	0.639	(.2,.8)
	MiniLM + TAG	**0.677***	**0.657***	**0.687***	0.803	**0.649***	(.1,.8,.1)
Money	MiniLM	0.545	0.535	0.563	0.706	0.515	(.2,.8)
	MiniLM + TAG	**0.559***	**0.542***	**0.571***	0.706	**0.523***	(.1,.8,.1)
Music	MiniLM	0.508	0.447	0.476	0.602	0.418	(.2,.8)
	MiniLM + TAG	**0.522***	**0.460***	**0.486***	0.602	**0.427***	(.1,.8,.1)
Rpg	MiniLM	0.657	0.646	0.685	0.849	0.640	(.2,.8)
	MiniLM + TAG	**0.677***	**0.660***	**0.695***	0.849	**0.651***	(.1,.8,.1)
Scifi	MiniLM	0.532	0.563	0.596	0.745	0.559	(.2,.8)
	MiniLM + TAG	**0.549***	**0.574***	**0.606***	0.745	**0.569***	(.1,.8,.1)
$\lambda_{TAG} = 0$	english, health, history, travel, workplace, writers, woodworking, vegetarianism, skeptics, politics, philosophy, parenting, outdoors, musicfans, literature, linguistics, judaism, interpersonal, hsm, genealogy, freelancing, fitness, expatriates, buddhism, anime						
No Statistical Improvement	boardgames, gardening, gaming, hinduism, islam, lifehacks, martialarts, movies, opensource, pets, sound, sports, sustainability						

Table 11.5: Results for the cQA task on single-community data extracted from Base SE-PQA.

version) and evaluate both personalised and non-personalised retrieval pipelines using MiniLM as the neural re-ranker. Due to computational limitations, we exclude the T5-based models in this setting. For each community, the interpolation weights λ are tuned independently using that community's validation data. Compared to the multi-domain results shown in Table 11.4, the contribution of TAG is substantially smaller in the single-domain scenario and is often negligible. In particular, for 25 of the 50 communities, personalization provides no measurable improvement and the optimised weights assign $\lambda_{TAG} = 0$. For an additional 13 communities, improvements in P@1 are not statistically significant. Table 11.5 reports results for the remaining 12 communities where personalization leads to statistically significant gains. Since statistical significance is influenced by sample size, we also compute average performance across all single-domain experiments. While overall effectiveness is slightly higher in the single-domain setting—mainly due to improved candidate recall when indexing smaller collections—the average improvement provided by TAG is reduced. Specifically, TAG improves P@1 by 2% on the full

multi-domain dataset, whereas the gain decreases to 1.1% when evaluating each community independently.

11.3 Conclusions

This Chapter introduced `SE-PQA` (StackExchange – Personalised Question Answering), a large-scale real-world dataset containing approximately one million questions and two million answers contributed by StackExchange users. The dataset is enriched with extensive user-level and interaction metadata, including post scores derived from community voting, tags, comments, user biographies, reputation scores, and profile view statistics. We described the dataset structure and discussed how it can support the development and evaluation of both standard and personalised retrieval approaches for the cQA task. As an initial baseline, we evaluated a two-stage retrieval pipeline combining BM25, neural models (DistilBERT, MiniLM, or T5), and a TAG-based personalization model. Our preliminary evaluation shows that personalization is highly effective in this setting: across all metrics and models, the inclusion of TAG yields statistically significant improvements.

Chapter 12

Synthetic Data Generation with Large Language Models for Personalized Community Question Answering

As discussed in the previous Chapter (Chapter 11), one of the main challenges in Personalised Information Retrieval (PIR) is the scarcity of large-scale public datasets that include rich user-level information. This limitation has historically constrained the development and evaluation of neural PIR models, whose effectiveness strongly depends on the availability and quality of training data. To address this gap, in Chapter 11, we introduce SE-PQA, a large-scale benchmark for personalised community Question Answering (cQA), derived from StackExchange. In parallel, another promising direction for alleviating data scarcity is the use of synthetic data generation. Recent advances in Large Language Models (LLMs) have enabled the automatic creation of high-quality synthetic datasets, particularly in settings where manual annotation is costly or limited. Models such as GPT [54] have shown strong capabilities in generating training data for a variety of tasks, including text classification [406], medical relation extraction [440], dialogue systems [154, 9], and Information Retrieval [38, 14]. In IR, prior work has mainly explored synthetic query generation [38] and synthetic document generation [14]. However, no attention has been devoted to generating answers or documents explicitly tailored to individual users.

In this Chapter, we investigate the use of LLMs to generate synthetic training data for personalised community question answering. Building on SE-PQA, we propose a pipeline, shown in Figure 12.1, that leverages both query content and user history to guide answer generation. Specifically, we use GPT-3.5 [54] and Phi-3 [2] to produce synthetic answers conditioned on user interests, with the goal of creating training instances that better capture personalised relevance signals.

To evaluate the usefulness of the generated synthetic data, we train neural retrieval models on LLM-generated answers and assess their effectiveness on a human-annotated test set. This evaluation allows us to quantify whether synthetic data can improve personalised retrieval performance compared to models trained only on human-authored answers. A key concern in LLM-based generation is hallucination, i.e., the tendency of models to produce fluent but incorrect or misleading content. Since hallucinations may affect the reliability of the resulting dataset, we manually analyse a sample of synthetic answers to estimate the rate of hallucinated answers. Interestingly, we find that retrieval models trained on synthetic answers can outperform those trained on human-authored data, even though more than 35% of the generated answers contain hallucinations. This result suggests that strict factual correctness may not be strictly necessary for training effective retrieval models, although this finding requires further investigation. In summary, this Chapter makes the following contributions:

1. We release Sy-SE-PQA, a synthetic dataset for personalised community question answering, containing LLM-generated answers for 100k StackExchange questions.

2. We provide an extensive experimental evaluation by fine-tuning a neural reranker on both human-authored and LLM-generated answers, and comparing performance on a human-written test set.

3. We analyse the diversity of the generated data across different LLMs and prompting strategies, and we manually assess the generated answers to quantify hallucination and factual inconsistency.

The remainder of this Chapter is organised as follows. Section 12.1 reviews the relevant related work. Section 12.2 describes our synthetic data generation pipeline in detail. Section 12.3 reports the results of our experimental evaluation, while Section 12.4 discusses the impact of different prompting strategies and LLM choices on data diversity. Finally, Section 12.5 concludes the Chapter.

12.1 Related Work

This section reviews prior work on synthetic data generation for training machine learning models, with a particular focus on IR and personalization tasks.

Recent progress in generative AI has motivated researchers to investigate the use of generative models for producing synthetic training data, particularly in Computer Vision and NLP. In this Chapter, we focus on the generation of textual synthetic data. Several studies have explored the use of LLMs to generate synthetic datasets for text classification, reporting mixed findings depending on the task, domain, and evaluation setting [236, 508, 4, 72, 131, 311, 404, 507, 88]. Puri et al.

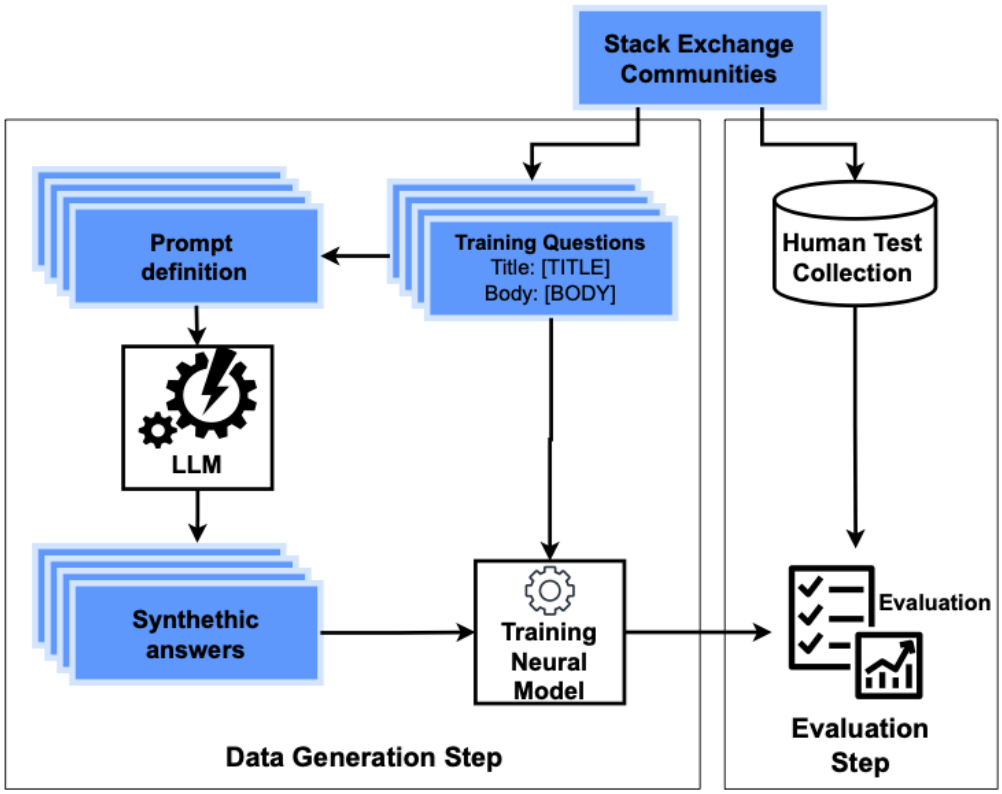

Figure 12.1: Illustration of the data generation and evaluation pipeline. In the data generation stage, synthetic answers are produced for questions sampled from StackExchange. These question–answer pairs are then used to train neural retrieval models, where the questions serve as queries and the synthetic answers as documents. Finally, in the evaluation stage, the trained models are assessed on a human-written test set.

[369] study question-answer generation as a form of data augmentation for improving QA models, motivated by the scarcity of human-annotated supervision. Their experiments show that models trained on synthetic QA pairs can outperform those trained on the original datasets in terms of Exact Match (EM) and F1. Similarly, Li et al. [255] analyse the effectiveness of LLM-generated synthetic data for text classification across tasks with varying degrees of subjectivity. Their results indicate that synthetic data can provide competitive performance for low-subjectivity tasks, such as news and topic classification, but tends to degrade effectiveness on high-subjectivity tasks such as sarcasm detection. In Information Retrieval, synthetic data generation has mainly focused on query and document creation. For instance, LLMs have been used to generate synthetic queries for training retrievers [38] and

to produce synthetic documents for expanding corpora [14]. Wang et al. [468] propose generating documents via few-shot prompting and concatenating them with the original query, effectively using synthetic text as a form of query expansion. Despite these advances, comparatively little work has addressed the generation of synthetic documents or answers that explicitly incorporate user-specific context, which is a key requirement in personalised retrieval settings.

We focus in particular on personalization tasks, which aim to tailor model outputs to individual users (or groups of users) based on information about their interests and behaviour. As LLMs are increasingly deployed in real-world applications, recent work has also highlighted both the potential and the risks of personalization in language models [192]. Salemi et al. [406] introduce the LaMP benchmark, a synthetic dataset designed to evaluate the ability of LLMs to produce personalised outputs. LaMP covers seven tasks, including three classification problems (Citation Identification, Movie Tagging, Product Rating) and four text generation tasks (News Headline, Scholarly Title, Email Subject, and Tweet Paraphrasing). Chan et al. [62] propose a pipeline where an LLM first generates a textual description of a user based on web data, and then uses this user profile as conditioning information to generate synthetic datasets, including mathematical and logical reasoning problems. In this Chapter, we focus on a specific personalization task: *Personalised Information Retrieval* (PIR). PIR aims to overcome the one-size-fits-all behaviour of traditional search engines by adapting retrieval results to individual users [271]. Two common strategies in PIR are personalised query expansion and personalised re-ranking. Personalised query expansion [27] leverages user-related documents to extract user-specific vocabulary and augment the original query with expansion terms. In contrast, personalised re-ranking operates in a two-stage architecture: a first-stage retriever produces an initial ranked list of documents, and a second-stage re-ranker re-scores these candidates by exploiting richer representations and additional signals. Personalised re-rankers can incorporate user context by modelling similarities between user preferences and document topics [227]. Since personalised IR inherently relies on sensitive user-related information, privacy concerns and anonymization constraints remain major challenges [22]. In this context, the possibility of generating synthetic data conditioned on user history represents a promising direction, as it may enable the development of personalised models without requiring direct access to private behavioural logs. Aliannejadi et al. [9] propose the TREC Interactive Knowledge Assistance Track (iKAT), a test collection designed to evaluate conversational search agents in settings that require personalization and context-awareness. The collection includes 36 dialogues across 20 topics, each associated with a personal text knowledge base describing a user persona. Passage retrieval is performed over ClueWeb22-B [340], and the dialogues are human-annotated to enable reliable evaluation of conversational and personalised retrieval systems.

12.2 Methodology

This section describes the methodology adopted to generate synthetic training data for personalised community question answering. We first detail the pipeline used to construct the synthetic dataset (Section 12.2.1), and then describe the retrieval models and evaluation framework used to assess the effectiveness of the generated data (Section 12.2.1).

12.2.1 Dataset Generation

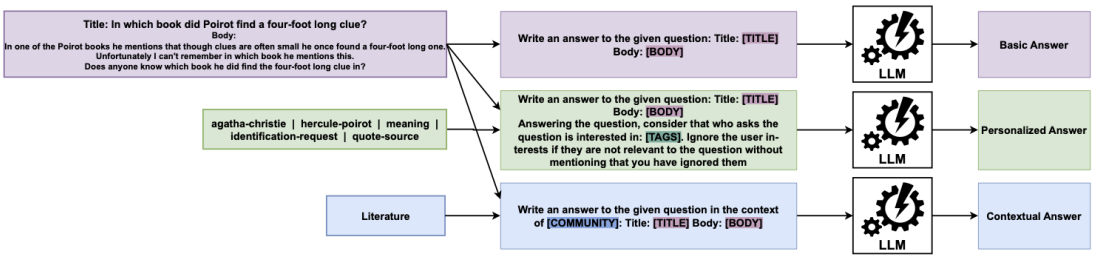

Figure 12.2: Example of a StackExchange question alongside the three prompting strategies used for synthetic answer generation.

The Sy-SE-PQA dataset is based on the personalised version of SE-PQA [227], which contains more than 200k training questions collected from 50 StackExchange communities. In this setting, each query consists of a *Title* and a *Body*. The Title provides a concise description of the user's information need, while the Body includes additional context, background details, and clarifications that help fully specify the question. Together, these two fields represent the full query formulation. In addition, StackExchange users can assign tags to their questions. Tags are selected from a predefined vocabulary and serve both as a topical categorization of the query and as an additional contextual feature. They make questions easier to browse and retrieve, and they can also be exploited by retrieval models to better interpret the underlying intent. In our dataset, each question is associated on average with 2 to 3 tags. To simulate different scenarios and user preferences, we generate three variants of synthetic answers for each question. These variants differ in the amount and type of additional information provided to the LLM, allowing us to evaluate how personalization signals affect the generated answers and their usefulness for retrieval training.

Basic Answer Generation In the first approach, the LLM is asked to generate an answer relying only on the question's Title and Body, without any personalization or additional context. The prompt is:

"Write an answer to the given question: Title: [TITLE] Body: [BODY]."

Personalized Answer Generation In the second setting, we incorporate user-specific information to generate personalised answers. Since prior work on SE-PQA showed that tag-based user profiles can improve personalised ranking [227], we model user interests using the five most frequent tags employed by the user in their previous interactions. We follow the same choice of five tags as in [227], since StackExchange questions typically contain up to five tags. The prompt used in this setting is: *"Write an answer to the given question: Title: [TITLE] Body: [BODY]. Answering the question, consider that who asks the question is interested in: [TAGS]. Ignore the user interests if they are not relevant to the question without mentioning that you have ignored them."*

This personalisation mechanism encourages the model to adapt the response to the user's topical background. This is particularly useful in scenarios where a user's history of engagement with certain topics can influence the type of response they are seeking. At the same time, the instruction to ignore irrelevant interests reduces the risk of the model generating off-topic content when the user profile does not match the current question.

Contextual Answer Generation In the third setting, we incorporate information about the community in which the question is posted. Communities represent domain-specific environments where users share expertise and similar interests. Conditioning the prompt on the community context may therefore produce answers that better align with the style and expectations of that specific community. The prompt is:

"Write an answer to the given question in the context of [COMMUNITY]: Title: [TITLE] Body: [BODY]."

Unlike the personalised setting, this approach does not aim to tailor the answer to a single user. Instead, it can be interpreted as adaptation to domain-specific and topic knowledge, where the generated answer reflects the domain and context of the target community. For instance, a question about historical figures posted in a History community might be answered differently than the same question posted in a general knowledge or movies community. By explicitly injecting the community context, the LLM can generate responses that delves into the historical aspects, thereby better aligning with the user's underlying intent and providing a more meaningful and satisfactory response.

Models and Baselines To generate synthetic answers, we rely on one closed-source LLM, *GPT-3.5 Turbo* [54], and two open-source models, i.e. *Phi-3-mini-4k-instruct* (3.82B parameters) and *Phi-3-medium-4k-instruct* (14B parameters) [2]. We selected Phi-3 as our open-source family due to its strong performance across a broad range of tasks, including reasoning [479], long-context understanding [54, 90, 110, 536], mathematical problem solving, and code generation [2]. Since synthetic generation is computationally expensive, we limit the number of questions

sampled from each community. Specifically, we sample at most 3,000 questions per community, and include all questions for smaller communities. This ensures a broad yet manageable variance of questions for our analysis. Under this setup, we generate approximately 100k synthetic answers per prompt type for each of the three LLMs. For both Phi-3-mini and Phi-3-medium we set the temperature to 1 and cap the maximum number of generated tokens to 500, following the recommendations in the Phi-3 documentation. To make a fair comparison, we apply the same generation parameters to GPT-3.5.

To evaluate the effectiveness of the synthetic answers for personalised retrieval, we adopt the same two-stage ranking architecture proposed in the SE-PQA benchmark [227]. This architecture balances efficiency and effectiveness by combining a lightweight recall-oriented retriever with a more expensive precision-oriented neural re-ranker. The first stage produces an initial candidate set for each query, which is then re-scored by the second stage. The *first stage* uses BM25 implemented through elasticsearch. We follow the same BM25 hyperparameters as in the original SE-PQA work [227], setting $b = 1$ and $k_1 = 1.75$. In the *second stage*, we apply a neural re-ranking model based on DistilBERT [409]. For each query, the re-ranker is applied to the top-100 documents retrieved by BM25. The final relevance score is computed as a convex combination of the BM25 score and the DistilBERT score. This fusion allows the system to combine lexical matching signals with semantic representations learned by the neural model. To determine the best fusion weights, we perform a grid search over the interval [0,1] with step size 0.1, optimizing the weights on the original SE-PQA validation set. The selected weights are reported in Tables 12.1, 12.2, and 12.3. We fine-tune DistilBERT for 20 epochs using a batch size of 128 and a learning rate of $5 \cdot 10^{-6}$. Training is performed using Triplet Margin Loss with in-batch random negatives and margin $\gamma = 0.5$. During training, the model is exposed only to documents generated using the same prompting strategy, meaning that each training run uses a consistent synthetic corpus (basic, personalised, or contextual). All non-relevant answers in the batch are treated as in-batch negatives. We use AdamW as optimizer and set the random seed to 42 to ensure reproducibility. All experiments are conducted on a single A100 GPU.

12.2.2 Evaluation

We assess retrieval effectiveness using P@1, nDCG@3, nDCG@10, and MAP@100, which capture both early precision and overall ranking quality. All metrics are computed using the *ranx* library [25].

12.3 Results

In this section, we report the results of our experimental evaluation. Ta-

Training	Model	P@1	NDCG@3	NDCG@10	MAP@100	λ
-	BM25	0.279	0.353	0.394	0.362	-
Real Answer	DistilBERT	0.253	0.329	0.378	0.344	-
	BM25 + DistilBERT	0.333*	0.415*	0.456*	0.421*	.3
Basic	DistilBERT	0.264	0.340	0.388	0.352	-
	BM25 + DistilBERT	0.336*	0.419*	0.460*	0.425*	.3
Personalized	DistilBERT	0.299*	0.374*	0.418*	0.384*	-
	BM25 + DistilBERT	0.345*	0.426*	0.465*	0.431*	.3
Contextual	DistilBERT	0.294*	0.369*	0.413*	0.379*	-
	BM25 + DistilBERT	0.342*	0.425*	0.464*	0.429*	.3

Table 12.1: Performance on the SE-PQA test set using models trained on data generated with Phi-mini.

Training	Model	P@1	NDCG@3	NDCG@10	MAP@100	λ
-	BM25	0.279	0.353	0.394	0.362	-
Real Answer	DistilBERT	0.253	0.329	0.378	0.344	-
	BM25 + DistilBERT	0.333*	0.415*	0.456*	0.421*	.3
Basic	DistilBERT	0.299*	0.373*	0.418*	0.383*	-
	BM25 + DistilBERT	0.344*	0.425*	0.465*	0.430*	.3
Personalized	DistilBERT	0.299*	0.374*	0.419*	0.384*	-
	BM25 + DistilBERT	0.347*	0.429*	0.468*	0.433*	.3
Contextual	DistilBERT	0.301*	0.375*	0.419*	0.385*	-
	BM25 + DistilBERT	0.338*	0.423*	0.463*	0.427*	.4

Table 12.2: Performance on the SE-PQA test set using models trained on data generated with Phi-medium.

bles 12.1, 12.2, and 12.3 summarize the performance of DistilBERT models trained on synthetic answers generated respectively with Phi-mini, Phi-medium, and GPT-3.5. In all tables, the symbol (*) denotes statistically significant improvements over BM25, assessed with a Bonferroni-corrected two-sided paired Student's t-test at 99% confidence. The column λ reports the optimized weight assigned to BM25 in the second-stage score combination, while the neural component weight is equal to $1 - \lambda$. Models are evaluated on the SE-PQA validation and test sets, and therefore their performance reflects their ability to retrieve human-written answers.

To directly compare synthetic and human-annotated training data, we fine-tune DistilBERT on human-written answers, i.e., the original answers marked as relevant for the user in the SE-PQA training set. These results are reported in the rows labeled *Real Answers*. Across Tables 12.1, 12.2, and 12.3, we observe that

Training	Model	P@1	NDCG@3	NDCG@10	MAP@100	λ
-	BM25	0.279	0.353	0.394	0.362	-
Real Answer	DistilBERT	0.269	0.346	0.394	0.358	-
	BM25 + DistilBERT	0.336*	0.419*	0.459*	0.424*	.3
Basic	DistilBERT	0.274	0.347	0.394	0.360	-
	BM25 + DistilBERT	0.335*	0.415*	0.455*	0.420*	.3
Personalized	DistilBERT	0.263	0.333	0.380	0.347	-
	BM25 + DistilBERT	0.325*	0.406*	0.448*	0.413*	.4
Contextual	DistilBERT	0.275	0.346	0.392	0.359	-
	BM25 + DistilBERT	0.331*	0.411*	0.452*	0.417*	.4

Table 12.3: Performance on the SE-PQA test set using models trained on data generated with GPT-3.5.

DistilBERT trained on synthetic data consistently and significantly outperforms BM25, indicating that the generated answers provide a useful training signal for neural retrieval models. In particular, when fine-tuned on *Personalised* synthetic answers, DistilBERT achieves improvements of up to 24% in P@1 and 18% in nDCG@10 over BM25. Interestingly, even without incorporating user or community information, the model trained on *Basic* synthetic answers reaches effectiveness comparable to the model trained on *Real Answers*. Moreover, for both Phi-mini and Phi-medium, training on *Personalised* answers yields clear gains over training on human-written answers, suggesting that personalization signals injected during generation can improve the usefulness of the synthetic corpus. For GPT-3.5, the model trained on *Basic* synthetic answers achieves the best performance among the GPT-based variants, although all prompt types lead to similar results overall. Compared to models trained on *Real Answers*, synthetic-data-trained models show only slightly lower effectiveness. However, differently from the Phi-based results, the *Personalised* answers generated by GPT-3.5 lead to the weakest performance among the three prompting strategies, indicating that the quality of personalization may depend strongly on the specific LLM used.

12.4 Exploratory Analysis

In this section, we perform an exploratory investigation of the synthetic answers generated by the different LLMs. We first analyse how much the generated answers differ across prompt types, and then we examine the factual reliability of the generated content by manually checking for hallucinations.

Model	Contextual	Personalized
Phi-mini		
Basic	0.29	0.24
Contextual	–	0.23
Phi-medium		
Basic	0.34	0.29
Contextual	–	0.27
GPT-3.5		
Basic	0.17	0.16
Contextual	–	0.15

Table 12.4: Diversity of synthetically generated data under different input prompts, measured using BLEU.

Model	Contextual	Personalized
Phi-mini		
Basic	58.57	54.01
Contextual	–	52.71
Phi-medium		
Basic	61.50	56.26
Contextual	–	53.52
GPT-3.5		
Basic	49.07	48.73
Contextual	–	47.44

Table 12.5: Diversity of synthetically generated data under different input prompts, measured using chrF.

12.4.1 Data Diversity

To quantify how different the answers produced by the three prompting strategies are, we rely on BLEU [343] and chrF [363], two widely adopted metrics that measure n-gram overlap between a candidate and a reference text. BLEU outputs a value between 0 and 1, where higher scores indicate stronger similarity between texts. Similarly, chrF ranges from 0 to 100, with larger values corresponding to greater lexical overlap. The results are reported in Table 12.4 and 12.5. The BLEU scores show that both Phi-mini and Phi-medium generate the most diverse outputs when comparing answers produced using the *Contextual* and *Personalised* prompts. This indicates that the addition of community context tends to produce answers

that differ substantially from those generated using user features. Overall, GPT-3.5 produces lower BLEU values than the Phi models, suggesting a higher variability in its generated answers. In particular, the BLEU score between *Contextual* and *Personalised* answers is 0.15, which is the lowest value observed across all comparisons. The chrF results confirm the same trend. Both Phi-mini and Phi-medium exhibit the highest diversity between *Contextual* and *Personalised* prompts, while the *Personalised-Basic* and *Basic-Contextual* comparisons lead to slightly more similar outputs. GPT-3.5 again produces the most diverse generations overall, reaching its lowest chrF score (47.44) when comparing *Contextual* and *Personalised* answers. Finally, we measure the lexical overlap between the question body and the corresponding answers, considering both human-written and LLM-generated responses. On average, human-written answers contain 23.4% of the words appearing in the question body. In contrast, Phi-3-mini answers contain 34.0%, 35.4%, and 35.5% of query words for the *Basic*, *Contextual*, and *Personalised* prompts, respectively. Phi-3-medium yields similar values, with overlaps of 35.4%, 35.5%, and 34.0%. GPT-3.5 answers contain 33.8%, 34.5%, and 33.2% of query words for the three prompts. We hypothesise that this higher overlap compared to human answers is due to the tendency of LLMs to repeat parts of the question in the response [14]. However, lexical overlap alone is not a reliable indicator of answer usefulness, especially in question answering scenarios where relevant human responses may contain paraphrases or implicit information. Since our retrieval models are evaluated on a human-written test set, it is possible that relevant answers with low lexical overlap remain highly informative.

12.4.2 Factual Information in Generated Answers

Although LLMs are capable of generating fluent and coherent text, they are known to hallucinate, producing plausible but factually incorrect statements [447]. This issue is further amplified by the possibility that LLMs may contain outdated knowledge, which can reduce the reliability of synthetic answers. In sensitive domains such as health, finance, or law, hallucinated responses may increase the risk of misinformation. To estimate the extent of hallucinations in our synthetic dataset, we randomly sample 100 questions uniformly across all communities and manually evaluate the generated answers. We find that Phi-mini, Phi-medium, and GPT-3.5 generate incorrect answers in 41%, 35%, and 36% of cases, respectively. Table 12.6 reports an example of hallucination. The question asks to identify an Italian movie in which a cartoonist is gradually transformed into an animated character. While both Phi-mini and Phi-medium propose Italian movie titles, none of the models correctly identify the actual answer. GPT-3.5, using the *Basic* and *Personalised* prompts, instead suggests *Monkeybone*, which is an American movie but includes both live-action and animated characters, making it a plausible yet incorrect guess. Interestingly, GPT-3.5 with the *Contextual* prompt generates a fictional movie title,

Question Title	Identify the movie where a cartoonist turns into an animation	
Question Body	I remember a movie, probably Italian or some similar Latin language, where a cartoonist is slowly transformed into an animated character. [...]	
Model	**Prompt type**	**Answer**
Human		I'm pretty sure you're talking about **Volere Volare (1991)**, an Italian movie where the main character has a date with a girl, but he's so nervous that he transforms into an animated character.
GPT-3.5	*Personalised*	The movie you are thinking of is likely **Monkeybone** (2001). [...]
	Basic	The movie you are referring to is likely **Monkeybone**, a 2001 American dark comedy film starring Brendan Fraser. [...]
	Contextual	The movie you are referring to is likely **Il Lupo perde il pelo**, also known as "The Wolf Loses His Fur". This Italian film was released in 1987 and features a cartoonist who is transformed into an animated character. [...]
Phi-mini	*Personalised*	The movie you are referring to is **La vita è bella** (Life Is Beautiful), directed by Roberto Benigni. However, it does not involve a cartoonist turning into an animated character. [...]
	Basic	[...] The movie you're referring to is likely **La vita è bella**. [...]
	Contextual	The movie you're referring to is likely **La vita è bella**. [...]
Phi-medium	*Personalised*	The movie you're referring to is likely **Lucky and Zorba** (1998). [...]
	Basic	The movie you're referring to is likely **Lucky and Zorba**. [...]
	Contextual	The movie you're referring to is likely **Lucky and Zorba**. It was released in 1998. [...]

Table 12.6: Examples of hallucinations in the Movies community.

Il lupo perde il pelo, which corresponds to a well-known Italian proverb rather than an actual film. This example highlights how LLMs may hallucinate both realistic but wrong answers and entirely fabricated content. Despite these limitations, the results reported in Section 12.3 show that retrieval models trained on synthetic answers can still achieve comparable, and in some cases superior, performance to models trained on human-written answers.

12.5 Conclusions

In this Chapter, we investigate the use of LLMs for synthetic data generation for Personalised Information Retrieval. Building on the SE-PQA dataset, we generate synthetic answers to user questions and introduce Sy-SE-PQA, a new synthetic resource designed for personalised community question answering. Our experimental evaluation shows that models trained on LLM-generated data consistently outperform BM25 and achieve retrieval performance that is comparable to, and in some cases better than, models trained on human-written answers. Finally, our exploratory analysis highlights an important limitation of LLM-generated data: more than 30% of synthetic answers contain hallucinated or factually incorrect information. Despite this, retrieval effectiveness remains competitive, suggesting that factual correctness of training answers may not be strictly necessary for learning useful retrieval representations, although this observation should be further validated in more safety-critical domains.

Part V

Conclusions and Insights

Chapter 13

Conclusions

This Chapter summarises the main findings, contributions, and broader implications of the research presented in this dissertation. It begins by revisiting the research objectives introduced in Chapter 1 and by outlining how the work conducted throughout the thesis addresses the challenges of efficiently adapting Large Language Models (LLMs) to downstream tasks and domains. It then provides a consolidated discussion of the principal outcomes obtained in each contribution. Finally, it highlights a set of open challenges and promising directions for future research.

13.1 Overview of our Contributions and Results

The work undertaken during the Ph.D. aimed to advance research in Natural Language Processing and Information Retrieval by developing methods that improve the adaptability of LLMs while reducing computational and storage requirements. The work was motivated by the increasing cost of training and fine-tuning modern neural architectures, which limits their applicability in resource-constrained environments and their scalability to multi-task and multi-domain scenarios. The contributions presented in this thesis address this issue through complementary perspectives, ranging from systematic analysis of existing parameter-efficient fine-tuning techniques, to the design of new structured adapter architectures, to modular retrieval systems, model merging strategies, interpretability-driven adaptation, and the creation of resources for personalised retrieval. The main contributions and results are summarised below.

Efficiency in NLP The systematic literature review presented in Chapter 3 investigated how efficiency is defined in the context of NLP, with a particular focus on Parameter-Efficient Fine-Tuning (PEFT) approaches. The analysis of 174 studies resulted in a structured taxonomy that categorised PEFT methods into additive,

selective, reparameterised, and hybrid families. This review highlighted that efficiency is often discussed primarily in terms of the number of trainable parameters, while other dimensions such as training time, inference latency, energy consumption, and environmental impact are less consistently evaluated. A key outcome of this contribution is the identification of methodological fragmentation in current PEFT research, driven by inconsistent experimental settings and the lack of standardised evaluation protocols. Overall, this Chapter provides a consolidated overview of the state of the art and motivates the need for more reproducible and comprehensive benchmarking practices.

AdaKron and MAdaKron Chapter 5 introduced AdaKron, a parameter-efficient adapter architecture that integrates Kronecker product decomposition into adapter design. AdaKron can be used as an efficient approach to fine-tune a LLMs on different downstream tasks. The motivation behind AdaKron was to increase the expressiveness of adapters while maintaining a very small number of trainable parameters. Experimental results showed that AdaKron can update less than 1% of the parameters of the underlying backbone while achieving competitive performance across multiple tasks, including question answering and language generation, and transformer architectures. Building upon this contribution, the Chapter proposed MAdaKron, a Mixture-of-Experts extension of AdaKron that introduces a routing mechanism to dynamically select specialised adapter components. This design enables the model to allocate capacity adaptively depending on the input, improving adaptation quality while preserving efficiency. The results showed that combining structured parameterisation with expert routing provides a favourable trade-off between performance and computational cost, supporting the broader hypothesis that Kronecker-based adapters are a strong alternative to classical bottleneck approaches.

DRAMA: Domain Retrieval using Adaptive Module Allocation Chapter 6 presented DRAMA, a parameter-efficient retrieval framework designed to support multi-domain Information Retrieval. Different from the previous Chapter, where the dissertation focused about task-specific models, DRAMA is based on domain-specific PEFT modules combined with a gating function that dynamically selects the most appropriate components depending on the domain of the input query. The framework addresses the limitation of unified multi-domain retrievers, which may fail to capture domain-specific patterns, while avoiding the redundancy and storage cost of training separate models for each domain. The experimental evaluation showed that DRAMA improves retrieval effectiveness compared to single multi-domain baselines and significantly reduces storage requirements relative to full domain-specific ensembles. In addition, the selective activation of adapters at inference time enables reductions in energy consumption, making the approach particularly relevant for large-scale and sustainable retrieval applications.

Task Arithmetic for Zero-Shot Information Retrieval Chapter 8 goes beyond efficient fine-tuning by exploring training-free adaptation methods, in particular by focusing on task arithmetic and task vector approaches. The Chapter investigated the application of Task Arithmetic to Information Retrieval and proposed an approach for defining domain- and language-specific retrievers through the linear composition of task vectors. This approach enables the definition of specialised retrieval models without further fine-tuning, making it suitable for zero-shot scenarios. Experimental results showed that task arithmetic can improve retrieval effectiveness for dense, sparse, and cross-encoder ranking models across scientific, biomedical, and multilingual benchmarks. This contribution extends the concept of efficiency beyond reducing trainable parameters, showing that parameter-space composition can offer an alternative solution to efficient adaptation.

Prompt-Token Adaptation and MonoT5 Analysis Chapter 9 further analyses the task vector between two specific models, i.e. MonoT5 and T5, to investigate how neural re-ranking models learn relevance signals during fine-tuning. The Chapter showed that the majority of the adaptation signal is concentrated in a small subset of parameters, particularly prompt-token embeddings. Motivated by this finding, the Chapter proposed a lightweight adaptation strategy based on fine-tuning only prompt-token embeddings, achieving retrieval performance on both in-domain and out-of-domain scenarios comparable to full fine-tuning, while significantly reducing training cost and the number of updated parameters. Furthermore, the analysis suggested that MonoT5's performance can be interpreted as the combination of two signals: the ability to identify low-quality documents and the ability to learn fine-grained relevance signals. This contribution provides interpretability insights into neural ranking models and suggests new directions for designing minimal adaptation strategies grounded in empirical analysis of model behaviour.

SE-PQA: Personalised Community Question Answering Chapter 11 introduced SE-PQA, a large-scale dataset designed to support reproducible experimentation in personalised Information Retrieval. The dataset was constructed from StackExchange data and enriched with user interaction signals, enabling retrieval models to incorporate user-specific context such as activity history and behavioural patterns. SE-PQA provides a benchmark that integrates textual relevance with personalisation features, enabling systematic evaluation of personalised ranking methods. Baseline experiments showed that incorporating personalisation signals can improve retrieval effectiveness compared to non-personalised approaches, while also showing that the impact of personalisation differs across communities, reflecting varying degrees of user dependency in the underlying data.

Sy-SE-PQA: Synthetic Data Generation for Personalised cQA Chapter 12 extended the SE-PQA contribution by introducing Sy-SE-PQA, a synthetic dataset generated using large language models to support scalable training in personalised retrieval tasks. The study explored different prompting strategies to generate answers conditioned on user and question context and evaluated whether retrieval models trained on synthetic data can learn effective ranking signals. Experimental results showed that models trained on LLM-generated answers achieve competitive performance and can outperform classical baselines, indicating that synthetic generation can be an effective tool for dataset expansion when human annotations are expensive. However, the analysis revealed that hallucinations and factual errors remain a significant limitation, with more than 30% of generated answers containing incorrect information. This finding suggests that retrieval models may learn useful relevance representations even when training data is not fully factually reliable, although this observation raises important concerns for applications where correctness is critical.

13.2 Directions for Future Research

The results presented in this dissertation highlight several open challenges that motivate future work in efficient adaptation of LLMs. While the contributions of this thesis provide evidence that strong effectiveness can be achieved under strict efficiency constraints, further research is needed to improve robustness, evaluation practices, and scalability across tasks and domains. The most relevant future directions are outlined below.

Standardisation of Efficiency Evaluation A key limitation identified in Chapter 3 is the lack of unified evaluation protocols for PEFT methods. Future research should focus on defining shared benchmarks and reporting standards that measure efficiency beyond the number of trainable parameters. In particular, evaluation frameworks should incorporate training time, inference latency, GPU memory usage, energy consumption, and environmental footprint. Standardised protocols would enable fairer comparisons between methods and support the development of adaptation techniques that account for real-world deployability in addition to effectiveness.

Parameter-Efficient Fine-Tuning and Modular Architectures The proposed adapter-based contributions suggest that structured parameterisation and modular allocation are promising strategies for scalable adaptation. Future work should explore how structured adapters such as AdaKron can be combined with quantisation-aware approaches, potentially enabling further reductions in memory footprint while maintaining expressive capacity.

In addition, DRAMA motivates the study of more flexible routing strategies for modular retrieval systems. Extending gating mechanisms beyond discrete domain classification to continuous or soft allocation strategies could enable smoother transitions across domains and improve robustness in mixed-domain scenarios. A particularly relevant direction is the integration of modular PEFT architectures with model merging techniques, enabling systems that dynamically combine multiple specialised components based on uncertainty or classifier confidence.

Model Merging and Task Vector Robustness Although Task Arithmetic showed the potential of training-free adaptation, its effectiveness remains inconsistent across model families and retrieval paradigms. Future research should investigate methods for improving the robustness of task vector composition, particularly by mitigating interference effects and sensitivity to scaling hyperparameters. Automatic weight selection and more stable composition mechanisms could prevent degradation in mismatched domains. Additionally, extending task arithmetic beyond linear operations represents an open direction, as non-linear merging strategies may better capture complex adaptation signals and reduce negative transfer.

Understanding Learning Dynamics in Neural Retrieval Models The analysis of MonoT5 suggests that fine-tuning updates are not uniformly distributed across model parameters and that a small subset of embeddings may encode a large portion of the relevance adaptation signal. Future work should generalise this analysis to other retrieval architectures, including dense bi-encoders and decoder-only ranking models, in order to determine whether similar patterns hold across model families. A deeper understanding of where task-specific information is encoded could lead to more principled and generalisable minimal fine-tuning strategies, improving both interpretability and efficiency.

Bibliography

[1] Hervé Abdi. "Singular value decomposition (SVD) and generalized singular value decomposition". In: *Encyclopedia of measurement and statistics* 907.912 (2007), p. 44.

[2] Marah Abdin et al. "Phi-4 technical report". In: *arXiv preprint arXiv:2412.08905* (2024).

[3] Alessandro Achille and Stefano Soatto. "Where is the Information in a Deep Neural Network?" In: *CoRR* abs/1905.12213 (2019). URL: http://arxiv.org/abs/1905.12213.

[4] Karan Aggarwal, Henry Jin, and Aitzaz Ahmad. "Entity-Controlled Synthetic Text Generation using Contextual Question and Answering with Pretrained Language Models". In: *NeurIPS 2022 Workshop on Synthetic Data for Empowering ML Research*. 2022.

[5] Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. "Intrinsic Dimensionality Explains the Effectiveness of Language Model Fine-Tuning". In: *CoRR* abs/2012.13255 (2020). arXiv: 2012.13255. URL: https://arxiv.org/abs/2012.13255.

[6] Wasi Uddin Ahmad, Kai-Wei Chang, and Hongning Wang. "Context Attentive Document Ranking and Query Suggestion". In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR'19. Paris, France: Association for Computing Machinery, 2019, pp. 385–394. ISBN: 9781450361729. DOI: 10.1145/3331184.3331246. URL: https://doi-org.unimib.idm.oclc.org/10.1145/3331184.3331246.

[7] Qingyao Ai et al. "Learning a Hierarchical Embedding Model for Personalized Product Search". In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '17. Shinjuku, Tokyo, Japan: Association for Computing Machinery, 2017, pp. 645–654. ISBN: 9781450350228. DOI: 10.1145/3077136.3080813. URL: https://doi-org.unimib.idm.oclc.org/10.1145/3077136.3080813.

[8] Samuel Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. "Git Re-Basin: Merging Models modulo Permutation Symmetries". In: *The Eleventh International Conference on Learning Representations*. 2023.

[9] Mohammad Aliannejadi et al. "Trec ikat 2023: The interactive knowledge assistance track overview". In: *arXiv preprint arXiv:2401.01330* (2024).

[10] Shun-ichi Amari. "Neural learning in structured parameter spaces-natural Riemannian gradient". In: *Advances in neural information processing systems* 9 (1996).

[11] Alan Ansell et al. "Composable Sparse Fine-Tuning for Cross-Lingual Transfer". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Smaranda Muresan, Preslav Nakov, and Aline Villavicencio. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 1778–1796. DOI: 10.18653/v1/2022.acl-long.125. URL: https://aclanthology.org/2022.acl-long.125/.

[12] Alan Ansell et al. "MAD-G: Multilingual adapter generation for efficient cross-lingual transfer". In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. 2021, pp. 4762–4781.

[13] Akari Asai et al. "ATTEMPT: Parameter-Efficient Multi-task Tuning via Attentional Mixtures of Soft Prompts". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 6655–6672. DOI: 10.18653/v1/2022.emnlp-main.446. URL: https://aclanthology.org/2022.emnlp-main.446/.

[14] Arian Askari et al. "A Test Collection of Synthetic Documents for Training Rankers: ChatGPT vs. Human Experts". In: CIKM '23. 2023.

[15] Seyedarmin Azizi, Souvik Kundu, and Massoud Pedram. "LaMDA: Large Model Fine-Tuning via Spectrally Decomposed Low-Dimensional Adaptation". In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 9635–9646. DOI: 10.18653/v1/2024.findings-emnlp.563. URL: https://aclanthology.org/2024.findings-emnlp.563/.

[16] Jimmy Lei Ba. "Layer normalization". In: *arXiv preprint* (2016).

[17] Vassileios Balntas et al. "Learning local feature descriptors with triplets and shallow convolutional neural networks." In: *Procedings of the British Machine Vision Conference 2016*. Vol. 1. York, 2016, p. 3.

[18] Krisztian Balog. *Entity-oriented search*. Springer Nature, 2018.

[19] Satanjeev Banerjee and Alon Lavie. "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments". In: *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*. 2005, pp. 65–72.

[20] Srijan Bansal et al. "PRO-CS : An Instance-Based Prompt Composition Technique for Code-Switched Tasks". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 10243–10255. DOI: 10.18653/v1/2022.emnlp-main.698. URL: https://aclanthology.org/2022.emnlp-main.698/.

[21] Roy Bar-Haim et al. "The second PASCAL recognising textual entailment challenge". In: Jan. 2006.

[22] Michael Barbaro, Tom Zeller, and Saul Hansell. "A face is exposed for AOL searcher no. 4417749". In: *New York Times* (2006).

[23] Michael Barbaro, Tom Zeller, and Saul Hansell. "A face is exposed for AOL searcher no. 4417749". In: *New York Times* 9.2008 (2006), p. 8.

[24] Federico Barbero et al. "Why do LLMs attend to the first token?" In: *Second Conference on Language Modeling*. 2025.

[25] Elias Bassani. "ranx: A blazing-fast python library for ranking evaluation and comparison". In: *European Conference on Information Retrieval*. Springer. 2022, pp. 259–264.

[26] Elias Bassani and Luca Romelli. "Ranx.Fuse: A Python Library for Metasearch". In: *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. CIKM '22. Atlanta, GA, USA: Association for Computing Machinery, 2022, pp. 4808–4812. ISBN: 9781450392365. DOI: 10.1145/3511808.3557207. URL: https://doi-org.unimib.idm.oclc.org/10.1145/3511808.3557207.

[27] Elias Bassani, Nicola Tonellotto, and Gabriella Pasi. "Personalized query expansion with contextual word embeddings". In: *ACM Transactions on Information Systems* (2023).

[28] Elias Bassani et al. "A Multi-Domain Benchmark for Personalized Search Evaluation". In: *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. CIKM '22. Atlanta, GA, USA: Association for Computing Machinery, 2022, pp. 3822–3827. ISBN: 9781450392365. DOI: 10.1145/3511808.3557536. URL: https://doi.org/10.1145/3511808.3557536.

[29] Elias Bassani et al. “A Multi-Domain Benchmark for Personalized Search Evaluation”. In: *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. CIKM ’22. Atlanta, GA, USA: Association for Computing Machinery, 2022, pp. 3822–3827. ISBN: 9781450392365. DOI: 10.1145/3511808.3557536. URL: https://doi.org/10.1145/3511808.3557536.

[30] Madeleine Bates. “Models of natural language understanding.” In: *Proceedings of the National Academy of Sciences* 92.22 (1995), pp. 9977–9982.

[31] Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. “BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 1–9. DOI: 10.18653/v1/2022.acl-short.1. URL: https://aclanthology.org/2022.acl-short.1.

[32] Nadav Benedek and Lior Wolf. “PRILoRA: Pruned and Rank-Increasing Low-Rank Adaptation”. In: *Findings of the Association for Computational Linguistics: EACL 2024*. Ed. by Yvette Graham and Matthew Purver. St. Julian’s, Malta: Association for Computational Linguistics, Mar. 2024, pp. 252–263. URL: https://aclanthology.org/2024.findings-eacl.18/.

[33] Luisa Bentivogli et al. “The Fifth PASCAL Recognizing Textual Entailment Challenge”. In: *Proceedings of the Second Text Analysis Conference, TAC 2009, Gaithersburg, Maryland, USA, November 16-17, 2009*. NIST, 2009. URL: https://tac.nist.gov/publications/2009/additional.papers/RTE5%5C_overview.proceedings.pdf.

[34] Kartikeya Bhardwaj et al. “Sparse High Rank Adapters”. In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=6hY60tkiEK.

[35] Rishabh Bhardwaj, Duc Anh Do, and Soujanya Poria. “Language Models are Homer Simpson! Safety Re-Alignment of Fine-tuned Language Models through Task Arithmetic”. In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, 2024, pp. 14138–14149. URL: https://aclanthology.org/2024.acl-long.762/.

[36] Rishabh Bhardwaj, Tushar Vaidya, and Soujanya Poria. “Adapter Pruning using Tropical Characterization”. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec.

2023, pp. 1699–1706. DOI: 10.18653/v1/2023.findings-emnlp.116. URL: https://aclanthology.org/2023.findings-emnlp.116/.

[37] Yonatan Bisk et al. "Piqa: Reasoning about physical commonsense in natural language". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 05. 2020, pp. 7432–7439.

[38] Luiz Bonifacio et al. "Inpars: Unsupervised dataset generation for information retrieval". In: *SIGIR 2022*. 2022.

[39] Luiz Bonifacio et al. "mmarco: A multilingual version of the ms marco passage ranking dataset". In: *arXiv preprint arXiv:2108.13897* (2021).

[40] Alexey Borisov et al. "A Context-Aware Time Model for Web Search". In: *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '16. Pisa, Italy: Association for Computing Machinery, 2016, pp. 205–214. ISBN: 9781450340694. DOI: 10.1145/2911451.2911504. URL: https://doi.org/10.1145/2911451.2911504.

[41] Shubhankar Borse et al. "FouRA: Fourier Low-Rank Adaptation". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=qCJ1dq5M7N.

[42] Vera Boteva et al. "A full-text learning to rank dataset for medical information retrieval". In: *European Conference on Information Retrieval*. Springer. 2016, pp. 716–722.

[43] Samuel Bowman et al. "Generating Sentences from a Continuous Space". In: *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*. 2016, pp. 10–21.

[44] Marco Braga. "Personalized Large Language Models through Parameter Efficient Fine-Tuning Techniques". In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '24. Washington DC, USA: Association for Computing Machinery, 2024, p. 3076. ISBN: 9798400704314. DOI: 10.1145/3626772.3657657. URL: https://doi.org/10.1145/3626772.3657657.

[45] Marco Braga. "Personalized Large Language Models through Parameter Efficient Fine-Tuning Techniques". In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '24. Washington DC, USA: Association for Computing Machinery, 2024, p. 3076. ISBN: 9798400704314. DOI: 10.1145/3626772.3657657. URL: https://doi.org/10.1145/3626772.3657657.

[46] Marco Braga, Gian Carlo Milanese, and Gabriella Pasi. "Investigating Large Language Models' Linguistic Abilities for Text Preprocessing". In: *2025 IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. IEEE. 2025.

[47] Marco Braga, Alessandro Raganato, and Gabriella Pasi. "AdaKron: An Adapter-based Parameter Efficient Model Tuning with Kronecker Product". In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. 2024, pp. 350–357.

[48] Marco Braga, Alessandro Raganato, and Gabriella Pasi. "MAdaKron: A mixture-of-AdaKron adapters". In: *Knowledge-Based Systems* 334 (2026), p. 115086. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2025.115086. URL: https://www.sciencedirect.com/science/article/pii/S0950705125021240.

[49] Marco Braga, Alessandro Raganato, Gabriella Pasi, et al. "Personalization in bert with adapter modules and topic modelling". In: *Proceedings of the 13th Italian Information Retrieval Workshop (IIR 2023). Pisa, Italy*. 2023, pp. 24–29.

[50] Marco Braga, Alessandro Raganato, and Gabriella Pasi. "Personalization in BERT with Adapter Modules and Topic Modelling." In: *IIR*. 2023, pp. 24–29.

[51] Marco Braga et al. "Investigating Task Arithmetic for Zero-Shot Information Retrieval". In: *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '25. Padua, Italy: Association for Computing Machinery, 2025, pp. 2738–2743. ISBN: 9798400715921. DOI: 10.1145/3726302.3730216. URL: https://doi.org/10.1145/3726302.3730216.

[52] Marco Braga et al. "Revealing MonoT5s Learning Mechanisms via Prompt-Token Adaptation". In: *European Conference on Information Retrieval*. Springer. 2026, pp. 450–465.

[53] Marco Braga et al. "Synthetic Data Generation with Large Language Models for Personalized Community Question Answering". In: *2024 IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. IEEE. 2024, pp. 360–366.

[54] Tom Brown et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[55] Lucas Caccia et al. “Multi-head adapter routing for cross-task generalization”. In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS ’23. New Orleans, LA, USA: Curran Associates Inc., 2023.

[56] RIZHAO CAI, Haoliang Li, and Alex Kot. “Towards domain generalized pruning by scoring out-of-distribution importance”. In: *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*. 2022.

[57] Tony Cai and Wen-Xin Zhou. “A Max-Norm Constrained Minimization Approach to 1-Bit Matrix Completion”. In: *Journal of Machine Learning Research* 14 (2013), pp. 3619–3647.

[58] Silvia Calegari and Gabriella Pasi. “Personal ontologies: Generation of user profiles based on the YAGO ontology”. In: *Information Processing & Management* 49.3 (2013). Personalization and Recommendation in Information Access, pp. 640–658. ISSN: 0306-4573. DOI: https://doi.org/10.1016/j.ipm.2012.07.010. URL: https://www.sciencedirect.com/science/article/pii/S0306457312001070.

[59] Rémi Calizzano et al. “Generating Extended and Multilingual Summaries with Pre-trained Transformers”. In: *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. Ed. by Nicoletta Calzolari et al. Marseille, France: European Language Resources Association, June 2022, pp. 1640–1650. URL: https://aclanthology.org/2022.lrec-1.175/.

[60] Xavier Carreras and Lluís Màrquez. “Introduction to the CoNLL-2004 Shared Task: Semantic Role Labeling”. In: *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*. Boston, Massachusetts, USA: Association for Computational Linguistics, May 2004, pp. 89–97. URL: https://aclanthology.org/W04-2412.

[61] Daniel Cer et al. “SemEval-2017 Task 1: Semantic Textual Similarity Multilingual and Crosslingual Focused Evaluation”. In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Ed. by Steven Bethard et al. Vancouver, Canada: Association for Computational Linguistics, Aug. 2017, pp. 1–14.

[62] Xin Chan et al. *Scaling Synthetic Data Creation with 1,000,000,000 Personas*. 2024. arXiv: 2406.20094.

[63] Aofei Chang et al. “BIPEFT: Budget-Guided Iterative Search for Parameter Efficient Fine-Tuning of Large Pretrained Language Models”. In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 7429–7440. DOI:

10.18653/v1/2024.findings-emnlp.437. URL: https://aclanthology.org/2024.findings-emnlp.437/.

[64] Wei-Cheng Chang et al. "Pre-training Tasks for Embedding-based Large-scale Retrieval". In: *International Conference on Learning Representations*. 2020.

[65] Xuejun Chang et al. "Neural Passage Quality Estimation for Static Pruning". In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '24. Washington DC, USA: Association for Computing Machinery, 2024, pp. 174–185. ISBN: 9798400704314. URL: https://doi.org/10.1145/3626772.3657765.

[66] Derek Chen et al. "Mixture of Soft Prompts for Controllable Data Generation". In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 14815–14833. DOI: 10.18653/v1/2023.findings-emnlp.988. URL: https://aclanthology.org/2023.findings-emnlp.988/.

[67] Guanzheng Chen et al. "Revisiting Parameter-Efficient Tuning: Are We Really There Yet?" In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 2612–2626. DOI: 10.18653/v1/2022.emnlp-main.168. URL: https://aclanthology.org/2022.emnlp-main.168.

[68] Hailin Chen et al. "Learning Label Modular Prompts for Text Classification in the Wild". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 1677–1690. DOI: 10.18653/v1/2022.emnlp-main.109. URL: https://aclanthology.org/2022.emnlp-main.109/.

[69] Jiaao Chen et al. "Parameter-Efficient Fine-Tuning Design Spaces". In: *The Eleventh International Conference on Learning Representations*. 2023.

[70] Jianlyu Chen et al. "M3-Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation". In: *Findings of the Association for Computational Linguistics: ACL 2024*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 2318–2335.

[71] Lichang Chen et al. "PTP: Boosting Stability and Performance of Prompt Tuning with Perturbation-Based Regularizer". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 13512–13525. DOI: 10.18653/v1/2023.emnlp-main.833. URL: https://aclanthology.org/2023.emnlp-main.833/.

[72] Maximillian Chen et al. "Weakly Supervised Data Augmentation Through Prompting for Dialogue Understanding". In: *NeurIPS 2022 Workshop on Synthetic Data for Empowering ML Research*. 2022.

[73] Nuo Chen et al. "Pass-Tuning: Towards Structure-Aware Parameter-Efficient Tuning for Code Representation Learning". In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 577–591. DOI: 10.18653/v1/2023.findings-emnlp.42. URL: https://aclanthology.org/2023.findings-emnlp.42/.

[74] Shijie Chen, Yu Zhang, and Qiang Yang. "Multi-Task Learning in Natural Language Processing: An Overview". In: *ACM Comput. Surv.* 56.12 (July 2024). ISSN: 0360-0300. DOI: 10.1145/3663363. URL: https://doi.org/10.1145/3663363.

[75] Xinyun Chen et al. "Targeted backdoor attacks on deep learning systems using data poisoning". In: *arXiv preprint arXiv:1712.05526* (2017).

[76] Xuxi Chen et al. "DSEE: Dually Sparsity-embedded Efficient Tuning of Pre-trained Language Models". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 8208–8222. DOI: 10.18653/v1/2023.acl-long.456. URL: https://aclanthology.org/2023.acl-long.456/.

[77] Yifan Chen et al. "Empowering parameter-efficient transfer learning by recognizing the kernel structure in self-attention". In: *Findings of the Association for Computational Linguistics: NAACL 2022*. Ed. by Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 1375–1388. DOI: 10.18653/v1/2022.findings-naacl.102. URL: https://aclanthology.org/2022.findings-naacl.102/.

[78] Yifan Chen et al. "Inducer-tuning: Connecting Prefix-tuning and Adapter-tuning". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and

Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 793–808. DOI: 10.18653/v1/2022.emnlp-main.50. URL: https://aclanthology.org/2022.emnlp-main.50/.

[79] Yukang Chen et al. "LongLoRA: Efficient Fine-tuning of Long-Context Large Language Models". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=6PmJoRfdaK.

[80] Yuyan Chen et al. "Hadamard Adapter: An Extreme Parameter-Efficient Adapter Tuning Method for Pre-trained Language Models". In: *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*. CIKM '23. Birmingham, United Kingdom: Association for Computing Machinery, 2023, pp. 276–285. ISBN: 9798400701245. DOI: 10.1145/3583780.3614904. URL: https://doi.org/10.1145/3583780.3614904.

[81] Zhuo Chen et al. "QuanTA: Efficient High-Rank Fine-Tuning of LLMs with Quantum-Informed Tensor Adaptation". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=EfpZNpkrm2.

[82] Hao Cheng et al. "Task-Aware Specialization for Efficient and Robust Dense Retrieval for Open-Domain Question Answering". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 1864–1875. URL: https://aclanthology.org/2023.acl-short.159/.

[83] Davide Chicco and Giuseppe Jurman. "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation". In: *BMC genomics* 21 (2020), pp. 1–13.

[84] Jae Won Cho et al. "Empirical study on using adapters for debiased Visual Question Answering". In: *Computer Vision and Image Understanding* 237 (2023), p. 103842. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2023.103842. URL: https://www.sciencedirect.com/science/article/pii/S1077314223002229.

[85] Joon-Young Choi et al. "SMoP: Towards Efficient and Effective Prompt Tuning with Sparse Mixture-of-Prompts". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 14306–14316. DOI: 10.18653/v1/2023.emnlp-main.884. URL: https://aclanthology.org/2023.emnlp-main.884/.

[86] Alexandra Chronopoulou et al. "AdapterSoup: Weight Averaging to Improve Generalization of Pretrained Language Models". In: *Findings of the Association for Computational Linguistics: EACL 2023*. 2023, pp. 2054–2063.

[87] Alexandra Chronopoulou et al. "Language and Task Arithmetic with Parameter-Efficient Layers for Zero-Shot Summarization". In: *Proceedings of the Fourth Workshop on Multilingual Representation Learning (MRL 2024)*. Ed. by Jonne Sälevä and Abraham Owodunni. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 114–126. URL: https://aclanthology.org/2024.mrl-1.7/.

[88] John Chung, Ece Kamar, and Saleema Amershi. "Increasing Diversity While Maintaining Accuracy: Text Data Generation with Large Language Models and Human Interventions". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2023, pp. 575–593.

[89] Christopher Clark et al. "BoolQ: Exploring the Surprising Difficulty of Natural Yes-No Questions". In: *NAACL*. 2019.

[90] Elizabeth Clark et al. "All That's 'Human'Is Not Gold: Evaluating Human Evaluation of Generated Text". In: *Proceedings of the ACL '21*. 2021, pp. 7282–7296.

[91] Peter Clark et al. "Think you have solved question answering? try arc, the ai2 reasoning challenge". In: *arXiv preprint arXiv:1803.05457* (2018).

[92] Arman Cohan et al. "SPECTER: Document-level Representation Learning using Citation-informed Transformers". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 2020, pp. 2270–2282. DOI: 10.18653/v1/2020.acl-main.207. URL: https://aclanthology.org/2020.acl-main.207/.

[93] Harris Cooper, Larry V Hedges, and Jeffrey C Valentine. *The handbook of research synthesis and meta-analysis*. Russell Sage Foundation, 2019.

[94] Nick Craswell et al. *Overview of the TREC 2019 deep learning track*. 2020. arXiv: 2003.07820 [cs.IR]. URL: https://arxiv.org/abs/2003.07820.

[95] Nick Craswell et al. *Overview of the TREC 2020 deep learning track*. 2021. arXiv: 2102.07662 [cs.IR]. URL: https://arxiv.org/abs/2102.07662.

[96] Ido Dagan, Oren Glickman, and Bernardo Magnini. "The PASCAL Recognising Textual Entailment Challenge". In: *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*. Ed. by Joaquin Quiñonero-Candela et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 177–190.

[97] Damai Dai et al. “Mixture-of-Experts for Biomedical Question Answering”. In: *Natural Language Processing and Chinese Computing: 12th National CCF Conference, NLPCC 2023, Foshan, China, October 12–15, 2023, Proceedings, Part I*. Foshan, China: Springer-Verlag, 2023, pp. 604–615. ISBN: 978-3-031-44692-4. DOI: 10.1007/978-3-031-44693-1_47. URL: https://doi.org/10.1007/978-3-031-44693-1_47.

[98] Zhuyun Dai and Jamie Callan. “Context-aware sentence/passage term importance estimation for first stage retrieval”. In: *arXiv preprint arXiv:1910.10687* (2019).

[99] Zhuyun Dai et al. “Promptagator: Few-shot Dense Retrieval From 8 Examples”. In: *The Eleventh International Conference on Learning Representations*. 2023.

[100] Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. “The commitmentbank: Investigating projection in naturally occurring discourse”. In: *proceedings of Sinn und Bedeutung*. Vol. 23. 2. 2019, pp. 107–124.

[101] Tim Dettmers et al. “Qlora: Efficient finetuning of quantized llms”. In: *Advances in neural information processing systems* 36 (2023), pp. 10088–10115.

[102] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423.

[103] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. URL: https://aclanthology.org/N19-1423/.

[104] Shizhe Diao et al. “Mixture-of-Domain-Adapters: Decoupling and Injecting Domain Knowledge to Pre-trained Language Models’ Memories”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 5113–5129. DOI: 10.18653/v1/2023.acl-long.280. URL: https://aclanthology.org/2023.acl-long.280/.

[105] Ning Ding et al. “Parameter-efficient fine-tuning of large-scale pre-trained language models”. In: *Nature Machine Intelligence* 5.3 (2023), pp. 220–235.

[106] Ning Ding et al. "Sparse Low-rank Adaptation of Pre-trained Language Models". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 4133–4145. DOI: 10.18653/v1/2023.emnlp-main.252. URL: https://aclanthology.org/2023.emnlp-main.252/.

[107] George Doddington. "Automatic evaluation of machine translation quality using n-gram co-occurrence statistics". In: *Proceedings of the second international conference on Human Language Technology Research*. 2002, pp. 138–145.

[108] William B. Dolan and Chris Brockett. "Automatically Constructing a Corpus of Sentential Paraphrases". In: *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*. 2005. URL: https://aclanthology.org/I05-5002.

[109] Jeff Donahue et al. "Decaf: A deep convolutional activation feature for generic visual recognition". In: *International conference on machine learning*. PMLR. 2014, pp. 647–655.

[110] Yao Dou et al. "Is GPT-3 Text Indistinguishable from Human Text? Scarecrow: A Framework for Scrutinizing Machine Text". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2022.

[111] Mengnan Du et al. "Robustness Challenges in Model Distillation and Pruning for Natural Language Understanding". In: *Proc. of the 17th EACL*. 2023. DOI: 10.18653/v1/2023.eacl-main.129.

[112] Olive Jean Dunn. "Multiple Comparisons among Means". In: *Journal of the American Statistical Association* 56 (1961), pp. 52–64.

[113] Ali Edalati et al. "KronA: Parameter Efficient Tuning with Kronecker Adapter". In: *The Third Workshop on Efficient Natural Language and Speech Processing (ENLSP-III)*. 2022.

[114] Ali Edalati et al. "Kronecker Decomposition for GPT Compression". In: *Annual Meeting of the Association for Computational Linguistics*. 2021. URL: https://api.semanticscholar.org/CorpusID:239009526.

[115] Aleksandra Edwards and Jose Camacho-Collados. "Language Models for Text Classification: Is In-Context Learning Enough?" In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. 2024, pp. 10058–10072.

[116] Guglielmo Faggioli et al. "Perspectives on large language models for relevance judgment". In: *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval*. 2023, pp. 39–50.

[117] Louis Falissard, Vincent Guigue, and Laure Soulier. "Improving generalization in large langue model by learning prefix subspaces". In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 11474–11483. DOI: 10.18653/v1/2023.findings-emnlp.768. URL: https://aclanthology.org/2023.findings-emnlp.768/.

[118] M Farzan and Derek A Waller. "Kronecker products and local joins of graphs". In: *Canadian Journal of Mathematics* 29.2 (1977), pp. 255–269.

[119] Wenfeng Feng et al. "Mixture-of-LoRAs: An Efficient Multitask Tuning Method for Large Language Models". In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. Ed. by Nicoletta Calzolari et al. Torino, Italia: ELRA and ICCL, May 2024, pp. 11371–11380. URL: https://aclanthology.org/2024.lrec-main.994/.

[120] John Fields, Kevin Chovanec, and Praveen Madiraju. "A Survey of Text Classification With Transformers: How Wide? How Large? How Long? How Accurate? How Expensive? How Safe?" In: *IEEE Access* 12 (2024), pp. 6518–6531. DOI: 10.1109/ACCESS.2024.3349952.

[121] Chelsea Finn, Pieter Abbeel, and Sergey Levine. "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. Ed. by Doina Precup and Yee Whye Teh. Vol. 70. Proceedings of Machine Learning Research. PMLR, June 2017, pp. 1126–1135. URL: https://proceedings.mlr.press/v70/finn17a.html.

[122] Nicolas Fiorini et al. "How user intelligence is improving PubMed". In: *Nature biotechnology* 36.10 (2018), pp. 937–945.

[123] Ronald A Fisher. "On the mathematical foundations of theoretical statistics". In: *Philosophical transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character* 222.594-604 (1922), pp. 309–368.

[124] Jonathan Frankle and Michael Carbin. "The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=rJl-b3RcF7.

[125] Peter I Frazier. "A tutorial on Bayesian optimization". In: *arXiv preprint arXiv:1807.02811* (2018).

[126] Markus Frohmann et al. "ScaLearn: Simple and Highly Parameter-Efficient Task Transfer by Learning to Scale". In: *Findings of the Association for Computational Linguistics: ACL 2024*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 11743–11776. DOI: 10.18653/v1/2024.findings-acl.699. URL: https://aclanthology.org/2024.findings-acl.699/.

[127] Chin-Lun Fu et al. "AdapterBias: Parameter-efficient Token-dependent Representation Shift for Adapters in NLP Tasks". In: *Findings of the Association for Computational Linguistics: NAACL 2022*. Ed. by Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 2608–2621. DOI: 10.18653/v1/2022.findings-naacl.199. URL: https://aclanthology.org/2022.findings-naacl.199/.

[128] Zihao Fu et al. "On the effectiveness of parameter-efficient fine-tuning". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 37. 11. 2023, pp. 12799–12807.

[129] Chongyang Gao et al. "Higher layers need more lora experts". In: *arXiv preprint arXiv:2402.08562* (2024).

[130] Ze-Feng Gao et al. "Parameter-Efficient Mixture-of-Experts Architecture for Pre-trained Language Models". In: *Proceedings of the 29th International Conference on Computational Linguistics*. Ed. by Nicoletta Calzolari et al. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 3263–3273. URL: https://aclanthology.org/2022.coling-1.288/.

[131] Jiahui Gao et al. "Self-guided noise-free data generation for efficient zero-shot learning". In: *arXiv preprint arXiv:2205.12679* (2022).

[132] Luyu Gao and Jamie Callan. "Condenser: a pre-training architecture for dense retrieval". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 2021, pp. 981–993.

[133] Ziqi Gao et al. "Parameter-Efficient Fine-Tuning with Discrete Fourier Transform". In: *International Conference on Machine Learning*. PMLR. 2024, pp. 14884–14901.

[134] Ziqi Gao et al. "Parameter-efficient fine-tuning with discrete fourier transform". In: *Proceedings of the 41st International Conference on Machine Learning*. ICML'24. Vienna, Austria: JMLR.org, 2024.

[135] Claire Gardent et al. “The WebNLG challenge: Generating text from RDF data”. In: *10th International Conference on Natural Language Generation*. ACL Anthology. 2017, pp. 124–133.

[136] Yao Ge et al. “Few-shot learning for medical text: A review of advances, trends, and opportunities”. In: *Journal of Biomedical Informatics* 144 (2023), p. 104458. ISSN: 1532-0464. DOI: https://doi.org/10.1016/j.jbi.2023.104458. URL: https://www.sciencedirect.com/science/article/pii/S153204642300179X.

[137] Mozhdeh Gheini, Xiang Ren, and Jonathan May. “Cross-Attention is All You Need: Adapting Pretrained Transformers for Machine Translation”. In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Ed. by Marie-Francine Moens et al. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 1754–1765. URL: https://aclanthology.org/2021.emnlp-main.132/.

[138] Danilo Giampiccolo et al. “The Third PASCAL Recognizing Textual Entailment Challenge”. In: *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*. Prague: Association for Computational Linguistics, June 2007, pp. 1–9. URL: https://aclanthology.org/W07-1401.

[139] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. “Deep sparse rectifier neural networks”. In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2011, pp. 315–323.

[140] Lorraine Goeuriot et al. “Medical information retrieval: introduction to the special issue”. In: *Information Retrieval Journal* 19.1 (2016), pp. 1–5.

[141] Reyhaneh Goli, Alistair Moffat, and George Buchanan. “Refined Medical Search via Dense Retrieval and User Interaction”. In: *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR ’25. Padua, Italy: Association for Computing Machinery, 2025, pp. 3315–3324. ISBN: 9798400715921. URL: https://doi.org/10.1145/3726302.3730292.

[142] Hao Gu et al. “NLoPT: N-gram Enhanced Low-Rank Task Adaptive Pretraining for Efficient Language Model Adaption”. In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. Ed. by Nicoletta Calzolari et al. Torino, Italia: ELRA and ICCL, May 2024, pp. 12259–12270. URL: https://aclanthology.org/2024.lrec-main.1072/.

[143] Jihao Gu et al. “From Bottom to Top: Extending the Potential of Parameter Efficient Fine-Tuning”. In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 3488–3500. DOI: 10.18653/v1/2024.emnlp-main.204. URL: https://aclanthology.org/2024.emnlp-main.204/.

[144] Xiangming Gu et al. “When Attention Sink Emerges in Language Models: An Empirical View”. In: *The Thirteenth International Conference on Learning Representations*. 2025.

[145] Yuxian Gu et al. “PPT: Pre-trained Prompt Tuning for Few-shot Learning”. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Smaranda Muresan, Preslav Nakov, and Aline Villavicencio. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 8410–8423. DOI: 10.18653/v1/2022.acl-long.576. URL: https://aclanthology.org/2022.acl-long.576/.

[146] Almog Gueta et al. “Knowledge is a Region in Weight Space for Fine-tuned Language Models”. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 1350–1370. DOI: 10.18653/v1/2023.findings-emnlp.95. URL: https://aclanthology.org/2023.findings-emnlp.95/.

[147] Anchun Gui and Han Xiao. “HiFi: High-Information Attention Heads Hold for Parameter-Efficient Model Adaptation”. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 8521–8537. DOI: 10.18653/v1/2023.acl-long.475. URL: https://aclanthology.org/2023.acl-long.475/.

[148] Demi Guo, Alexander Rush, and Yoon Kim. “Parameter-Efficient Transfer Learning with Diff Pruning”. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Ed. by Chengqing Zong et al. Online: Association for Computational Linguistics, Aug. 2021, pp. 4884–4896. DOI: 10.18653/v1/2021.acl-long.378. URL: https://aclanthology.org/2021.acl-long.378/.

[149] Umang Gupta, Aram Galstyan, and Greg Ver Steeg. “Jointly Reparametrized Multi-Layer Adaptation for Efficient and Private Tuning”. In: *Findings of the Association for Computational Linguistics: ACL*

2023. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 12612–12629. DOI: 10.18653/v1/2023.findings-acl.799. URL: https://aclanthology.org/2023.findings-acl.799/.

[150] Suchin Gururangan et al. "DEMix Layers: Disentangling Domains for Modular Language Modeling". In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 5557–5576. DOI: 10.18653/v1/2022.naacl-main.407. URL: https://aclanthology.org/2022.naacl-main.407.

[151] Suchin Gururangan et al. "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Ed. by Dan Jurafsky et al. Online: Association for Computational Linguistics, July 2020, pp. 8342–8360.

[152] Suchin Gururangan et al. "Don't Stop Pretraining: Adapt Language Models to Domains and Tasks". In: *Proceedings of ACL*. 2020.

[153] Michael Gutmann and Aapo Hyvärinen. "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings. 2010, pp. 297–304.

[154] Perttu Hämäläinen, Mikke Tavast, and Anton Kunnari. "Evaluating Large Language Models in Generating Synthetic HCI Research Data: a Case Study". In: *Proceedings of the CHI '23*. CHI '23. <conf-loc>, <city>Hamburg</city>, <country>Germany</country>, </conf-loc>: ACM, 2023. ISBN: 9781450394215. DOI: 10.1145/3544548.3580688. URL: https://doi.org/10.1145/3544548.3580688.

[155] Zeyu Han et al. "Parameter-Efficient Fine-Tuning for Large Models: A Comprehensive Survey". In: *Transactions on Machine Learning Research* (2024). ISSN: 2835-8856.

[156] Allan Hanbury. "Medical information retrieval: an instance of domain-specific search". In: *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*. 2012, pp. 1191–1192.

[157] Jitai Hao et al. "MEFT: Memory-Efficient Fine-Tuning through Sparse Adapter". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association

for Computational Linguistics, Aug. 2024, pp. 2375–2388. DOI: 10.18653/v1/2024.acl-long.129. URL: https://aclanthology.org/2024.acl-long.129/.

[158] Yongchang Hao, Yanshuai Cao, and Lili Mou. "FLORA: low-rank adapters are secretly gradient compressors". In: *Proceedings of the 41st International Conference on Machine Learning*. ICML'24. Vienna, Austria: JMLR.org, 2024.

[159] Helia Hashemi et al. "Dense retrieval adaptation using target domain description". In: *Proceedings of the 2023 ACM SIGIR International Conference on Theory of Information Retrieval*. 2023, pp. 95–104.

[160] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. "LoRA+: Efficient Low Rank Adaptation of Large Models". In: *Forty-first International Conference on Machine Learning*. 2024. URL: https://openreview.net/forum?id=NEv8YqBROO.

[161] Junxian He et al. "Towards a Unified View of Parameter-Efficient Transfer Learning". In: *International Conference on Learning Representations*. 2022.

[162] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[163] Pengcheng He, Jianfeng Gao, and Weizhu Chen. "Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing". In: *arXiv preprint arXiv:2111.09543* (2021).

[164] Pengcheng He et al. "DEBERTA: DECODING-ENHANCED BERT WITH DISENTANGLED ATTENTION". In: *ICLR 2021*. 2021. URL: https://openreview.net/forum?id=XPZIaotutsD.

[165] Pengcheng He et al. "Deberta: Decoding-enhanced bert with disentangled attention". In: *arXiv preprint arXiv:2006.03654* (2020).

[166] Ruining He and Julian McAuley. "Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering". In: *Proceedings of the 25th International Conference on World Wide Web*. WWW '16. Montréal, Québec, Canada: International World Wide Web Conferences Steering Committee, 2016, pp. 507–517. ISBN: 9781450341431. DOI: 10.1145/2872427.2883037. URL: https://doi.org/10.1145/2872427.2883037.

[167] Shwai He et al. "Mera: Merging pretrained adapters for few-shot learning". In: *arXiv preprint arXiv:2308.15982* (2023).

[168] Shwai He et al. "SparseAdapter: An Easy Approach for Improving the Parameter-Efficiency of Adapters". In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 2184–2190. DOI: 10.18653/v1/2022.findings-emnlp.160. URL: https://aclanthology.org/2022.findings-emnlp.160/.

[169] Xuehai He et al. *Parameter-efficient Model Adaptation for Vision Transformers*. 2023. arXiv: 2203.16329 [cs.CV]. URL: https://arxiv.org/abs/2203.16329.

[170] Yun He et al. "Hyperprompt: Prompt-based task-conditioning of transformers". In: *International conference on machine learning*. PMLR. 2022, pp. 8678–8690.

[171] Harold V. Henderson, Friedrich Pukelsheim, and Shayle R. Searle. "On the history of the kronecker product". In: *Linear & Multilinear Algebra* 14 (1983), pp. 113–120. URL: https://api.semanticscholar.org/CorpusID:123284178.

[172] Matthew Henderson et al. *Efficient Natural Language Response Suggestion for Smart Reply*. 2017. DOI: 10.48550/ARXIV.1705.00652. URL: https://arxiv.org/abs/1705.00652.

[173] Dan Hendrycks and Kevin Gimpel. "Gaussian error linear units (gelus)". In: *arXiv preprint arXiv:1606.08415* (2016).

[174] Karl Moritz Hermann et al. "Teaching machines to read and comprehend". In: *Advances in neural information processing systems* 28 (2015).

[175] Roland Herzog et al. "Frobenius-type norms and inner products of matrices and linear maps with applications to neural network training". In: *arXiv preprint arXiv:2311.15419* (2023).

[176] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. *Distilling the Knowledge in a Neural Network*. 2015. URL: http://arxiv.org/abs/1503.02531.

[177] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[178] Sebastian Hofstätter et al. "Efficiently Teaching an Effective Dense Retriever with Balanced Topic Aware Sampling". In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '21. Virtual Event, Canada: Association for Computing Machinery, 2021, pp. 113–122. ISBN: 9781450380379. URL: https://doi.org/10.1145/3404835.3462891.

[179] Sebastian Hofstätter et al. *Improving Efficient Neural Ranking Models with Cross-Architecture Knowledge Distillation*. 2021. arXiv: 2010.02666 [cs.IR]. URL: https://arxiv.org/abs/2010.02666.

[180] Seung-Kyu Hong, Jae-Seok Jang, and Hyuk-Yoon Kwon. "Enhancing performance of transformer-based models in natural language understanding through word importance embedding". In: *Knowledge-Based Systems* 304 (2024), p. 112404. ISSN: 0950-7051. DOI: https://doi.org/10.1016/j.knosys.2024.112404. URL: https://www.sciencedirect.com/science/article/pii/S0950705124010384.

[181] Doris Hoogeveen, Karin M. Verspoor, and Timothy Baldwin. "CQADupStack: A Benchmark Data Set for Community Question-Answering Research". In: *Proceedings of the 20th Australasian Document Computing Symposium (ADCS)*. ADCS '15. Parramatta, NSW, Australia: ACM, 2015, 3:1–3:8. ISBN: 978-1-4503-4040-3. DOI: 10.1145/2838931.2838934. URL: http://doi.acm.org/10.1145/2838931.2838934.

[182] Yutai Hou et al. "MetaPrompting: Learning to Learn Better Prompts". In: *Proceedings of the 29th International Conference on Computational Linguistics*. Ed. by Nicoletta Calzolari et al. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 3251–3262. URL: https://aclanthology.org/2022.coling-1.287/.

[183] Neil Houlsby et al. "Parameter-Efficient Transfer Learning for NLP". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, Sept. 2019, pp. 2790–2799. URL: https://proceedings.mlr.press/v97/houlsby19a.html.

[184] Neil Houlsby et al. "Parameter-Efficient Transfer Learning for NLP". In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, Sept. 2019, pp. 2790–2799. URL: https://proceedings.mlr.press/v97/houlsby19a.html.

[185] Jeremy Howard and Sebastian Ruder. "Universal Language Model Fine-tuning for Text Classification". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Iryna Gurevych and Yusuke Miyao. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 328–339. DOI: 10.18653/v1/P18-1031. URL: https://aclanthology.org/P18-1031/.

[186] Edward J Hu et al. "LoRA: Low-Rank Adaptation of Large Language Models". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=nZeVKeeFYf9.

[187] Edward J Hu et al. "Lora: Low-rank adaptation of large language models." In: *ICLR* 1.2 (2022), p. 3.

[188] Shengding Hu et al. "Sparse structure search for delta tuning". In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*. NIPS '22. New Orleans, LA, USA: Curran Associates Inc., 2022. ISBN: 9781713871088.

[189] Shengding Hu et al. "Sparse structure search for parameter-efficient tuning". In: *arXiv preprint arXiv:2206.07382* (2022).

[190] Minghui Huang, Wei Peng, and Dong Wang. *TPRM: A Topic-based Personalized Ranking Model for Web Search*. 2021. DOI: 10.48550/ARXIV.2108.06014. URL: https://arxiv.org/abs/2108.06014.

[191] Po-Sen Huang et al. "Learning deep structured semantic models for web search using clickthrough data". In: *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*. 2013, pp. 2333–2338.

[192] Xiaolei Huang et al. "UserNLP'22: 2022 International Workshop on User-centered Natural Language Processing". In.

[193] Yufei Huang et al. "FPT: Improving Prompt Tuning Efficiency via Progressive Training". In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 6877–6887. DOI: 10.18653/v1/2022.findings-emnlp.511. URL: https://aclanthology.org/2022.findings-emnlp.511/.

[194] HuggingFace. *Train a Sentence Embedding Model with 1B Training Pairs*. HuggingFace. 2021. URL: https://huggingface.co/blog/1b-sentence-embeddings (visited on 09/01/2021).

[195] Samuel Humeau et al. "Poly-encoders: Transformer architectures and pre-training strategies for fast and accurate multi-sentence scoring". In: *arXiv preprint arXiv:1905.01969* (2019).

[196] Gabriel Ilharco et al. "Editing models with task arithmetic". In: *The Eleventh International Conference on Learning Representations*. 2022.

[197] R Islamaj Dogan, J Wilbur, and DC Comeau. "BioC and simplified use of the PMC open access dataset for biomedical text mining". In: *Proceedings of the 4th Workshop on Building and Evaluating Resources for Health and Biomedical Text Processing. LREC, Reykjavik, Iceland*. 2014.

[198] Hamish Ivison and Matthew Peters. "Hyperdecoders: Instance-specific decoders for multi-task NLP". In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 1715–1730. DOI: 10.18653/v1/2022.findings-emnlp.124. URL: https://aclanthology.org/2022.findings-emnlp.124/.

[199] Robert A. Jacobs et al. "Adaptive Mixtures of Local Experts". In: *Neural Computation* 3.1 (Mar. 1991), pp. 79–87. ISSN: 0899-7667. DOI: 10.1162/neco.1991.3.1.79.

[200] Abhinav Jain et al. "Prompt tuning strikes back: Customizing foundation models with low-rank prompt adaptation". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 47297–47316.

[201] Shashank Mohan Jain. "Hugging face". In: *Introduction to transformers for NLP: With the hugging face library and models to solve problems*. Springer, 2022, pp. 51–67.

[202] Eric Jang, Shixiang Gu, and Ben Poole. "Categorical Reparametrization with Gumble-Softmax". In: *International Conference on Learning Representations (ICLR 2017)*. OpenReview. net. 2017.

[203] Kalervo Järvelin and Jaana Kekäläinen. "Cumulated gain-based evaluation of IR techniques". In: *ACM Transactions on Information Systems (TOIS)* 20.4 (2002), pp. 422–446.

[204] Vitor Jeronymo et al. "Inpars-v2: Large language models as efficient dataset generators for information retrieval". In: *arXiv preprint arXiv:2301.01820* (2023).

[205] Albert Q Jiang et al. "Mixtral of experts". In: *arXiv preprint arXiv:2401.04088* (2024).

[206] Albert Q. Jiang et al. *Mixtral of Experts*. 2024. arXiv: 2401.04088 [cs.LG]. URL: https://arxiv.org/abs/2401.04088.

[207] Jingjing Jiang and Nanning Zheng. "MixPHM: Redundancy-Aware Parameter-Efficient Tuning for Low-Resource Visual Question Answering". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 24203–24213.

[208] Jingjing Jiang and Nanning Zheng. "MixPHM: Redundancy-Aware Parameter-Efficient Tuning for Low-Resource Visual Question Answering". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2023, pp. 24203–24213.

[209] Tangyu Jiang, Haodi Wang, and Chun Yuan. "DiffoRA: Enabling Parameter-Efficient Fine-Tuning via Differential Module Selection". In: *arXiv preprint arXiv:2502.08905* (2025).

[210] Feihu Jin, Yifan Liu, and Ying Tan. "Derivative-Free Optimization for Low-Rank Adaptation in Large Language Models". In: *IEEE/ACM Trans. Audio, Speech and Lang. Proc.* 32 (Oct. 2024), pp. 4607–4616. ISSN: 2329-9290. DOI: 10.1109/TASLP.2024.3477330. URL: https://doi.org/10.1109/TASLP.2024.3477330.

[211] Qiao Jin et al. "Pubmedqa: A dataset for biomedical research question answering". In: *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*. 2019, pp. 2567–2577.

[212] Xisen Jin et al. "Dataless Knowledge Fusion by Merging Weights of Language Models". In: *The Eleventh International Conference on Learning Representations*. 2023.

[213] Michael I Jordan and Robert A Jacobs. "Hierarchical mixtures of experts and the EM algorithm". In: (2001).

[214] Navya Jose et al. "A Survey of Current Datasets for Code-Switching Research". In: *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*. 2020, pp. 136–141. DOI: 10.1109/ICACCS48705.2020.9074205.

[215] Euna Jung, Jaekeol Choi, and Wonjong Rhee. "Semi-Siamese Bi-encoder Neural Ranking Model Using Lightweight Fine-Tuning". In: *Proceedings of the ACM Web Conference 2022*. WWW '22. Virtual Event, Lyon, France: Association for Computing Machinery, 2022, pp. 502–511. ISBN: 9781450390965. DOI: 10.1145/3485447.3511978. URL: https://doi.org/10.1145/3485447.3511978.

[216] Jared Kaplan et al. "Scaling laws for neural language models". In: *arXiv preprint arXiv:2001.08361* (2020).

[217] Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. "Compacter: Efficient low-rank hypercomplex adapter layers". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 1022–1035.

[218] Vladimir Karpukhin et al. "Dense Passage Retrieval for Open-Domain Question Answering". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2020. URL: https://aclanthology.org/2020.emnlp-main.550/.

[219] Pranav Kasela, Gabriella Pasi, and Raffaele Perego. "SE-PEF: a Resource for Personalized Expert Finding". In: *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*. SIGIR-AP '23. Beijing, China: Association for Computing Machinery, 2023, pp. 288–309. ISBN: 9798400704086. DOI: 10.1145/3624918.3625335. URL: https://doi.org/10.1145/3624918.3625335.

[220] Pranav Kasela et al. "DESIRE-ME: Domain-Enhanced Supervised Information Retrieval Using Mixture-of-Experts". In: *European Conference on Information Retrieval*. Ed. by Nazli Goharian et al. Springer. Cham: Springer Nature Switzerland, 2024, pp. 111–125. ISBN: 978-3-031-56060-6.

[221] Pranav Kasela et al. "DIETA: A Decoder-only Transformer-based Model for Italian-English Machine TrAnslation". In: *Proceedings of the Eleventh Italian Conference on Computational Linguistics (CLiC-it 2025)*. Ed. by Cristina Bosco et al. Cagliari, Italy: CEUR Workshop Proceedings, Sept. 2025, pp. 539–549. ISBN: 979-12-243-0587-3. URL: https://aclanthology.org/2025.clicit-1.52/.

[222] Pranav Kasela et al. "DRAMA: Domain Retrieval using Adaptive Module Allocation". In: *arXiv preprint arXiv:2602.14960* (2026).

[223] Pranav Kasela et al. "Overview of the PIR Track at FIRE 2024: Evaluation of Personalised Information Retrieval". In: *Proceedings of the 16th Annual Meeting of the Forum for Information Retrieval Evaluation*. 2024.

[224] Pranav Kasela et al. "Overview of the PIR Track at FIRE 2024: Evaluation of Personalised Information Retrieval". In: *Proceedings of the 16th Annual Meeting of the Forum for Information Retrieval Evaluation*. FIRE '24. Association for Computing Machinery, 2025, pp. 11–13. ISBN: 9798400713187. DOI: 10.1145/3734947.3735665. URL: https://doi.org/10.1145/3734947.3735665.

[225] Pranav Kasela et al. "SE-PQA: Personalized Community Question Answering". In: *The Web Conference 2024 (WWW '24)*. 2024.

[226] Pranav Kasela et al. "SE-PQA: Personalized Community Question Answering". In: *Companion Proceedings of the ACM on Web Conference 2024*. WWW '24. Singapore, Singapore: Association for Computing Machinery, 2024, pp. 1095–1098. ISBN: 9798400701726. DOI: 10.1145/3589335.3651445. URL: https://doi.org/10.1145/3589335.3651445.

[227] Pranav Kasela et al. "SE-PQA: StackExchange Personalized Community Question Answering". In: *CEUR WORKSHOP PROCEEDINGS*. Vol. 3802. 2024, pp. 99–102.

[228] Wenjun Ke et al. "Unveiling LoRA Intrinsic Ranks via Salience Analysis". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=vU512K8vrR.

[229] Daniel Khashabi et al. "Looking beyond the surface: A challenge set for reading comprehension over multiple sentences". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 2018, pp. 252–262.

[230] Omar Khattab and Matei Zaharia. "ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT". In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '20. Virtual Event, China: Association for Computing Machinery, 2020, pp. 39–48. ISBN: 9781450380164. DOI: 10.1145/3397271.3401075. URL: https://doi.org/10.1145/3397271.3401075.

[231] Omar Khattab and Matei Zaharia. "Colbert: Efficient and effective passage search via contextualized late interaction over bert". In: *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*. 2020, pp. 39–48.

[232] Minyoung Kim and Timothy Hospedales. "Bayestune: Bayesian sparse deep model fine-tuning". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 65317–65365.

[233] James Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *Proceedings of the national academy of sciences* 114.13 (2017), pp. 3521–3526.

[234] Dawid Jan Kopiczko, Tijmen Blankevoort, and Yuki M Asano. "VeRA: Vector-based Random Matrix Adaptation". In: *The Twelfth International Conference on Learning Representations*. 2024.

[235] Solomon Kullback and Richard A Leibler. "On information and sufficiency". In: *The annals of mathematical statistics* 22.1 (1951), pp. 79–86.

[236] Varun Kumar, Ashutosh Choudhary, and Eunah Cho. "Data Augmentation using Pre-trained Transformer Models". In: *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*. 2020, pp. 18–26.

[237] Carlos Lassance et al. "SPLADE-v3: New baselines for SPLADE". In: *arXiv preprint arXiv:2403.06789* (2024).

[238] Neal Lawton et al. "Neural Architecture Search for Parameter-Efficient Fine-tuning of Large Pre-trained Language Models". In: *Findings of the Association for Computational Linguistics: ACL 2023*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 8506–8515. DOI: 10.18653/v1/2023.findings-acl.539. URL: https://aclanthology.org/2023.findings-acl.539/.

[239] Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. "SNIP: SINGLE-SHOT NETWORK PRUNING BASED ON CONNECTION SENSITIVITY". In: *International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=B1VZqjAcYX.

[240] Tao Lei et al. "Conditional adapters: Parameter-efficient transfer learning with fast inference". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 8152–8172.

[241] Zijian Lei, Dong Qian, and William Cheung. "Fast Randomized Low-Rank Adaptation of Pre-trained Language Models with PAC Regularization". In: *Findings of the Association for Computational Linguistics: ACL 2024*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 5236–5249. DOI: 10.18653/v1/2024.findings-acl.310. URL: https://aclanthology.org/2024.findings-acl.310/.

[242] Brian Lester, Rami Al-Rfou, and Noah Constant. "The Power of Scale for Parameter-Efficient Prompt Tuning". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Ed. by Marie-Francine Moens et al. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 3045–3059. DOI: 10.18653/v1/2021.emnlp-main.243. URL: https://aclanthology.org/2021.emnlp-main.243/.

[243] Hector Levesque, Ernest Davis, and Leora Morgenstern. "The winograd schema challenge". In: *Thirteenth international conference on the principles of knowledge representation and reasoning*. 2012.

[244] Mike Lewis et al. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Ed. by Dan Jurafsky et al. Online: Association for Computational Linguistics, July 2020, pp. 7871–7880. DOI: 10.18653/v1/2020.acl-main.703. URL: https://aclanthology.org/2020.acl-main.703/.

[245] Chunyuan Li et al. "Measuring the intrinsic dimension of objective landscapes". In: *arXiv preprint arXiv:1804.08838* (2018).

[246] Jonathan Li et al. "Prefix Propagation: Parameter-Efficient Tuning for Long Sequences". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 1408–1419. DOI: 10.18653/v1/2023.acl-short.120. URL: https://aclanthology.org/2023.acl-short.120/.

[247] Junyi Li et al. "Pre-Trained Language Models for Text Generation: A Survey". In: *ACM Comput. Surv.* 56.9 (Apr. 2024). ISSN: 0360-0300. DOI: 10.1145/3649449. URL: https://doi.org/10.1145/3649449.

[248] Raymond Li, Gabriel Murray, and Giuseppe Carenini. "Mixture-of-Linguistic-Experts Adapters for Improving and Interpreting Pre-trained Language Models". In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 9456–9469. DOI: 10.18653/v1/2023.findings-emnlp.634. URL: https://aclanthology.org/2023.findings-emnlp.634/.

[249] Raymond Li, Gabriel Murray, and Giuseppe Carenini. "Mixture-of-Linguistic-Experts Adapters for Improving and Interpreting Pre-trained Language Models". In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. 2023, pp. 9456–9469.

[250] Shiwei Li et al. "Beyond Higher Rank: Token-wise Input-Output Projections for Efficient Low-Rank Adaptation". In: *The Thirty-ninth Annual Conference on Neural Information Processing Systems*. 2025. URL: https://openreview.net/forum?id=w2xl15ZbD3.

[251] Shiwei Li et al. "BoRA: Towards More Expressive Low-Rank Adaptation with Block Diversity". In: *arXiv preprint arXiv:2508.06953* (2025).

[252] Siwei Li et al. "LoRASC: Expressive and Generalizable Low-rank Adaptation for Large Models via Slow Cascaded Learning". In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 12806–12816. DOI: 10.18653/v1/2024.findings-emnlp.748. URL: https://aclanthology.org/2024.findings-emnlp.748/.

[253] Xiang Lisa Li and Percy Liang. "Prefix-Tuning: Optimizing Continuous Prompts for Generation". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 4582–4597.

DOI: 10.18653/v1/2021.acl-long.353. URL: https://aclanthology.org/2021.acl-long.353.

[254] Yang Li, Shaobo Han, and Shihao Ji. "VB-LoRA: Extreme Parameter Efficient Fine-Tuning with Vector Banks". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=kuCY0mW4Q3.

[255] Zhuoyan Li et al. "Synthetic Data Generation with Large Language Models for Text Classification: Potential and Limitations". In: *EMNLP 2023*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. 2023.

[256] Vladislav Lialin, Vijeta Deshpande, and Anna Rumshisky. *Scaling Down to Scale Up: A Guide to Parameter-Efficient Fine-Tuning*. 2023. arXiv: 2303.15647 [cs.CL]. URL: https://arxiv.org/abs/2303.15647.

[257] Vladislav Lialin et al. "ReLoRA: High-Rank Training Through Low-Rank Updates". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=DLJznSp6X3.

[258] Baohao Liao, Yan Meng, and Christof Monz. "Parameter-Efficient Fine-Tuning without Introducing New Latency". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 4242–4260. DOI: 10.18653/v1/2023.acl-long.233. URL: https://aclanthology.org/2023.acl-long.233/.

[259] Baohao Liao and Christof Monz. "3-in-1: 2D Rotary Adaptation for Efficient Finetuning, Efficient Batching and Composability". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=rYjYwuM6yH.

[260] Baohao Liao and Christof Monz. "3-in-1: 2d rotary adaptation for efficient finetuning, efficient batching and composability". In: *Advances in Neural Information Processing Systems* 37 (), pp. 35018–35048.

[261] Yu-Chen Lin et al. "ACCEPT: Adaptive Codebook for Composite and Efficient Prompt Tuning". In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 15345–15358. DOI: 10.18653/v1/2024.findings-emnlp.900. URL: https://aclanthology.org/2024.findings-emnlp.900/.

[262] Chin-Yew Lin. "Rouge: A package for automatic evaluation of summaries". In: *Text summarization branches out*. 2004, pp. 74–81.

[263] Qingyun Lin et al. "WGLora:Efficient fine-tuning method integrating weights and gradient low-rank adaptation". In: *Proceedings of the 2024 12th International Conference on Communications and Broadband Networking*. ICCBN '24. Nyingchi, China: Association for Computing Machinery, 2024, pp. 108–114. ISBN: 9798400717109. DOI: 10.1145/3688636.3688654. URL: https://doi.org/10.1145/3688636.3688654.

[264] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. "Distilling dense representations for ranking using tightly-coupled teachers". In: *arXiv preprint arXiv:2010.11386* (2020).

[265] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. "In-Batch Negatives for Knowledge Distillation with Tightly-Coupled Teachers for Dense Retrieval". In: *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*. Ed. by Anna Rogers et al. Online: Association for Computational Linguistics, Aug. 2021, pp. 163–173. DOI: 10.18653/v1/2021.repl4nlp-1.17. URL: https://aclanthology.org/2021.repl4nlp-1.17.

[266] Sheng-Chieh Lin et al. "How to train your dragon: Diverse augmentation towards generalizable dense retrieval". In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. 2023, pp. 6385–6400.

[267] Zhisheng Lin et al. "PEMT: Multi-Task Correlation Guided Mixture-of-Experts Enables Parameter-Efficient Transfer Learning". In: *Findings of the Association for Computational Linguistics: ACL 2024*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 6869–6883. DOI: 10.18653/v1/2024.findings-acl.410. URL: https://aclanthology.org/2024.findings-acl.410/.

[268] Vijay Lingam et al. "SVFT: Parameter-Efficient Fine-Tuning with Singular Vectors". In: *2nd Workshop on Advancing Neural Network Training: Computational Efficiency, Scalability, and Resource Optimization (WANT@ICML 2024)*. 2024. URL: https://openreview.net/forum?id=DOUskwCqg5.

[269] Haokun Liu et al. "Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning". In: *Advances in Neural Information Processing Systems*. Ed. by S. Koyejo et al. Vol. 35. Curran Associates, Inc., 2022, pp. 1950–1965. URL: https://proceedings.neurips.cc/paper%5C_files/paper/2022/file/0cde695b83bd186c1fd456302888454c-Paper-Conference.pdf.

[270] Haokun Liu et al. "Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 1950–1965.

[271] Jingjing Liu, Chang Liu, and Nicholas J Belkin. "Personalization in text information retrieval: A survey". In: *Journal of the Association for Information Science and Technology* (2020).

[272] Pengfei Liu et al. "Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing". In: *ACM Comput. Surv.* 55.9 (Jan. 2023). ISSN: 0360-0300. DOI: 10.1145/3560815. URL: https://doi.org/10.1145/3560815.

[273] Qidong Liu et al. "MOELoRA: An MOE-based Parameter Efficient Fine-Tuning Method for Multi-task Medical Applications". In: *CoRR* abs/2310.18339 (2023). URL: https://doi.org/10.48550/arXiv.2310.18339.

[274] Qidong Liu et al. "When MOE Meets LLMs: Parameter Efficient Fine-tuning for Multi-task Medical Applications". In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '24. Washington DC, USA: Association for Computing Machinery, 2024, pp. 1104–1114. ISBN: 9798400704314. DOI: 10.1145/3626772.3657722. URL: https://doi.org/10.1145/3626772.3657722.

[275] Shih-Yang Liu et al. "DoRA: weight-decomposed low-rank adaptation". In: *Proceedings of the 41st International Conference on Machine Learning*. ICML'24. Vienna, Austria: JMLR.org, 2024.

[276] Xiangyang Liu et al. "Late Prompt Tuning: A Late Prompt Could Be Better Than Many Prompts". In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 1325–1338. DOI: 10.18653/v1/2022.findings-emnlp.95. URL: https://aclanthology.org/2022.findings-emnlp.95/.

[277] Xiao Liu et al. "GPT understands, too". In: *AI Open* (2023).

[278] Xiao Liu et al. "P-Tuning: Prompt Tuning Can Be Comparable to Fine-tuning Across Scales and Tasks". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics. 2022.

[279] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *ArXiv* abs/1907.11692 (2019). URL: https://api.semanticscholar.org/CorpusID:198953378.

[280] Zequan Liu et al. "ALoRA: Allocating Low-Rank Adaptation for Fine-tuning Large Language Models". In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Ed. by Kevin Duh, Helena Gomez, and Steven Bethard. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 622–641. DOI: 10.18653/v1/2024.naacl-long.35. URL: https://aclanthology.org/2024.naacl-long.35/.

[281] Zequan Liu et al. "PARA: Parameter-Efficient Fine-tuning with Prompt-Aware Representation Adjustment". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Ed. by Franck Dernoncourt, Daniel Preoţiuc-Pietro, and Anastasia Shimorina. Miami, Florida, US: Association for Computational Linguistics, Nov. 2024, pp. 728–737. DOI: 10.18653/v1/2024.emnlp-industry.55. URL: https://aclanthology.org/2024.emnlp-industry.55/.

[282] Zeyu Liu et al. "AFLoRA: Adaptive Freezing of Low Rank Adaptation in Parameter Efficient Fine-Tuning of Large Models". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 161–167. DOI: 10.18653/v1/2024.acl-short.16. URL: https://aclanthology.org/2024.acl-short.16/.

[283] Kyle Lo et al. "S2ORC: The Semantic Scholar Open Research Corpus". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Ed. by Dan Jurafsky et al. Online: Association for Computational Linguistics, July 2020, pp. 4969–4983. URL: https://aclanthology.org/2020.acl-main.447/.

[284] Ilya Loshchilov and Frank Hutter. "Decoupled Weight Decay Regularization". In: *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL: https://openreview.net/forum?id=Bkg6RiCqY7.

[285] Wenxuan Lu et al. "Hit the Nail on the Head: Parameter-Efficient Multi-task Tuning via Human Language Intervention". In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 8873–8885. DOI: 10.18653/v1/2024.findings-emnlp.518. URL: https://aclanthology.org/2024.findings-emnlp.518/.

[286] Ximing Lu et al. "Inference-Time Policy Adapters (IPA): Tailoring Extreme-Scale LMs without Fine-tuning". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 6863–6883. DOI: 10.18653/v1/2023.emnlp-main.424. URL: https://aclanthology.org/2023.emnlp-main.424/.

[287] Dan Luo et al. "ERAT-DLoRA: Parameter-efficient tuning with enhanced range adaptation in time and depth aware dynamic LoRA". In: *Neurocomputing* 614 (2025), p. 128778. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2024.128778. URL: https://www.sciencedirect.com/science/article/pii/S0925231224015492.

[288] Xindi Luo et al. "KnowLA: Enhancing Parameter-efficient Finetuning with Knowledgeable Adaptation". In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Ed. by Kevin Duh, Helena Gomez, and Steven Bethard. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 7153–7166. DOI: 10.18653/v1/2024.naacl-long.396. URL: https://aclanthology.org/2024.naacl-long.396/.

[289] Mihai Lupu, Michail Salampasis, and Allan Hanbury. "Domain specific search". In: *Professional search in the modern world: Cost action IC1002 on multilingual and multifaceted interactive information access*. Springer, 2014, pp. 96–117.

[290] Chuancheng Lv et al. "HyperLoRA: Efficient Cross-task Generalization via Constrained Low-Rank Adapters Generation". In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 16376–16393. DOI: 10.18653/v1/2024.findings-emnlp.956. URL: https://aclanthology.org/2024.findings-emnlp.956/.

[291] Fang Ma et al. "XPrompt: Exploring the Extreme of Prompt Tuning". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 11033–11047. DOI: 10.18653/v1/2022.emnlp-main.758. URL: https://aclanthology.org/2022.emnlp-main.758/.

[292] Xinyu Ma et al. "Scattered or Connected? An Optimized Parameter-efficient Tuning Approach for Information Retrieval". In: *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. CIKM '22. Atlanta, GA, USA: Association for Computing Machinery, 2022,

pp. 1471–1480. ISBN: 9781450392365. DOI: 10.1145/3511808.3557445. URL: https://doi.org/10.1145/3511808.3557445.

[293] Yufei Ma et al. "MoDULA: Mixture of Domain-Specific and Universal LoRA for Multi-Task Learning". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 2758–2770. DOI: 10.18653/v1/2024.emnlp-main.161. URL: https://aclanthology.org/2024.emnlp-main.161/.

[294] Zhengyi Ma et al. "PSTIE: Time Information Enhanced Personalized Search". In: *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. CIKM '20. Virtual Event, Ireland: Association for Computing Machinery, 2020, pp. 1075–1084. ISBN: 9781450368599. DOI: 10.1145/3340531.3411877. URL: https://doi.org/10.1145/3340531.3411877.

[295] Sean MacAvaney, Craig Macdonald, and Iadh Ounis. "Reproducing Personalised Session Search Over the AOL Query Log". In: *Advances in Information Retrieval: 44th European Conference on IR Research, ECIR 2022, Stavanger, Norway, April 10–14, 2022, Proceedings, Part I*. Stavanger, Norway: Springer-Verlag, 2022, pp. 627–640. ISBN: 978-3-030-99735-9. DOI: 10.1007/978-3-030-99736-6_42. URL: https://doi.org/10.1007/978-3-030-99736-6_42.

[296] Sean MacAvaney et al. "Simplified Data Wrangling with ir_datasets". In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '21. Virtual Event, Canada: Association for Computing Machinery, 2021, pp. 2429–2436. ISBN: 9781450380379. URL: https://doi.org/10.1145/3404835.3463254.

[297] Craig Macdonald et al. "PyTerrier: Declarative Experimentation in Python from BM25 to Dense Retrieval". In: *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. CIKM '21. Virtual Event, Queensland, Australia: Association for Computing Machinery, 2021, pp. 4526–4533. ISBN: 9781450384469. URL: https://doi.org/10.1145/3459637.3482013.

[298] Iain Mackie, Jeffrey Dalton, and Andrew Yates. "How Deep is your Learning: the DL-HARD Annotated Deep Learning Dataset". In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2021.

[299] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. "The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables". In: *International Conference on Learning Representations*. 2017.

[300] Rabeeh Karimi Mahabadi et al. "Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks". In: *arXiv preprint arXiv:2106.04489* (2021).

[301] Macedo Maia et al. "Www'18 open challenge: financial opinion mining and question answering". In: *Companion proceedings of the the web conference 2018*. 2018, pp. 1941–1942.

[302] Eran Malach et al. "Proving the Lottery Ticket Hypothesis: Pruning is All You Need". In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 6682–6691. URL: https://proceedings.mlr.press/v119/malach20a.html.

[303] Bhavitvya Malik et al. "UDAPTER - Efficient Domain Adaptation Using Adapters". In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Ed. by Andreas Vlachos and Isabelle Augenstein. Dubrovnik, Croatia: Association for Computational Linguistics, May 2023, pp. 2249–2263. DOI: 10.18653/v1/2023.eacl-main.165. URL: https://aclanthology.org/2023.eacl-main.165/.

[304] Antonio Mallia et al. "PISA: Performant indexes and search for academia". In: *Proceedings of the Open-Source IR Replicability Challenge* (2019).

[305] Yuning Mao et al. "UniPELT: A Unified Framework for Parameter-Efficient Language Model Tuning". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 6253–6264. DOI: 10.18653/v1/2022.acl-long.433. URL: https://aclanthology.org/2022.acl-long.433.

[306] Charles H Martin and Michael W Mahoney. "Heavy-tailed universality predicts trends in test accuracies for very large pre-trained deep neural networks". In: *Proceedings of the 2020 SIAM International Conference on Data Mining*. SIAM. 2020, pp. 505–513.

[307] Charles H. Martin and Michael W. Mahoney. *Traditional and Heavy Tailed Self Regularization in Neural Network Models*. 2019. URL: https://openreview.net/forum?id=SJeFNoRcFQ.

[308] Brian W Matthews. "Comparison of the predicted and observed secondary structure of T4 phage lysozyme". In: *Biochimica et Biophysica Acta (BBA)-Protein Structure* 405.2 (1975), pp. 442–451.

[309] Hansa Meghwani et al. “Hard Negative Mining for Domain-Specific Retrieval in Enterprise Systems”. In: *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*. Ed. by Georg Rehm and Yunyao Li. Vienna, Austria: Association for Computational Linguistics, July 2025, pp. 1013–1026. ISBN: 979-8-89176-288-6. DOI: 10.18653/v1/2025.acl-industry.72. URL: https://aclanthology.org/2025.acl-industry.72/.

[310] Fanxu Meng, Zhaohui Wang, and Muhan Zhang. “PiSSA: Principal Singular Values and Singular Vectors Adaptation of Large Language Models”. In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=6ZBHIEtdP4.

[311] Yu Meng et al. “Generating training data with language models: Towards zero-shot language understanding”. In: *Advances in Neural Information Processing Systems* (2022).

[312] Todor Mihaylov et al. “Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering”. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Ed. by Ellen Riloff et al. Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 2381–2391. DOI: 10.18653/v1/D18-1260. URL: https://aclanthology.org/D18-1260/.

[313] Gian Carlo Milanese et al. “Fact-driven health information retrieval: Integrating LLMs and knowledge graphs to combat misinformation”. In: *European Conference on Information Retrieval*. Springer. 2025, pp. 192–200.

[314] Bhaskar Mitra and Nick Craswell. “An Introduction to Neural Information Retrieval”. In: *Foundations and Trends® in Information Retrieval* 13.1 (2018), pp. 1–126. ISSN: 1554-0669. DOI: 10.1561/1500000061. URL: http://dx.doi.org/10.1561/1500000061.

[315] Bhaskar Mitra, Fernando Diaz, and Nick Craswell. “Learning to Match using Local and Distributed Representations of Text for Web Search”. In: *International Conference on World Wide Web*. WWW ’17. Perth, Australia: International World Wide Web Conferences Steering Committee, 2017. ISBN: 9781450349130.

[316] Bhaskar Miutra and Nick Craswell. “An introduction to neural information retrieval”. In: *Foundations and Trends$\hat{W}$ in Accounting* 13.1 (2018), pp. 1–126.

[317] P Molchanov et al. “Pruning convolutional neural networks for resource efficient inference”. In: *5th International Conference on Learning Representations, ICLR 2017-Conference Track Proceedings*. 2019.

[318] Timo Möller, Julian Risch, and Malte Pietsch. "GermanQuAD and GermanDPR: Improving Non-English Question Answering and Passage Retrieval". In: *Proceedings of the 3rd Workshop on Machine Reading for Question Answering*. 2021, pp. 42–50.

[319] Calvin N Mooers. "Zatocoding applied to mechanical organization of knowledge". In: *American documentation* 2.1 (1951), pp. 20–32.

[320] Nafise Sadat Moosavi et al. "Adaptable Adapters". In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2022, pp. 3742–3753.

[321] Lia Morra et al. "Designing Logic Tensor Networks for Visual Sudoku puzzle classification". In: *CEUR WORKSHOP PROCEEDINGS*. Vol. 3432. CEUR. 2023, pp. 223–232.

[322] Shamsuddeen Hassan Muhammad et al. *SemEval-2025 Task 11: Bridging the Gap in Text-Based Emotion Detection*. 2025. arXiv: 2503.07269 [cs.CL]. URL: https://arxiv.org/abs/2503.07269.

[323] J. Pablo Muñoz, Jinjie Yuan, and Nilesh Jain. "Shears: Unstructured Sparsity with Neural Low-rank Adapter Search". In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 6: Industry Track)*. Ed. by Yi Yang et al. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 395–405. DOI: 10.18653/v1/2024.naacl-industry.34. URL: https://aclanthology.org/2024.naacl-industry.34/.

[324] Linyong Nan et al. "DART: Open-Domain Structured Data Record to Text Generation". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 2021, pp. 432–447.

[325] Shashi Narayan, Shay B Cohen, and Mirella Lapata. "Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. 2018.

[326] Humza Naveed et al. "A comprehensive overview of large language models". In: *ACM Transactions on Intelligent Systems and Technology* 16.5 (2025), pp. 1–72.

[327] Thanh-Do Nguyen et al. "Passage-based BM25 Hard Negatives: A Simple and Effective Negative Sampling Strategy For Dense Retrieval". In: *Proceedings of the 37th Pacific Asia Conference on Language, Information and Computation*. Ed. by Chu-Ren Huang et al. Hong Kong, China: Association for Computational Linguistics, Dec. 2023, pp. 591–599. URL: https://aclanthology.org/2023.paclic-1.59/.

[328] Tri Nguyen et al. "MS MARCO: A Human Generated MAchine Reading COmprehension Dataset". In: *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016*. Ed. by Tarek Richard Besold et al. Vol. 1773. CEUR Workshop Proceedings. CEUR-WS.org, 2016.

[329] Tuc Van Nguyen and Thai Le. "Adapters Mixup: Mixing Parameter-Efficient Adapters to Enhance the Adversarial Robustness of Fine-tuned Pre-trained Text Classifiers". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 21183–21203. DOI: 10.18653/v1/2024.emnlp-main.1180. URL: https://aclanthology.org/2024.emnlp-main.1180/.

[330] Jianmo Ni et al. "Large Dual Encoders Are Generalizable Retrievers". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 9844–9855. URL: https://aclanthology.org/2022.emnlp-main.669/.

[331] Alex Nichol, Joshua Achiam, and John Schulman. "On first-order meta-learning algorithms". In: *arXiv preprint arXiv:1803.02999* (2018).

[332] Binling Nie, Yiming Shao, and Yigang Wang. "Know-Adapter: Towards Knowledge-Aware Parameter-Efficient Transfer Learning for Few-shot Named Entity Recognition". In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. Ed. by Nicoletta Calzolari et al. Torino, Italia: ELRA and ICCL, May 2024, pp. 9777–9786. URL: https://aclanthology.org/2024.lrec-main.854/.

[333] Mahdi Nikdan et al. "RoSA: accurate parameter-efficient fine-tuning via robust adaptation". In: *Proceedings of the 41st International Conference on Machine Learning*. ICML'24. Vienna, Austria: JMLR.org, 2024.

[334] Rodrigo Nogueira and Kyunghyun Cho. *Passage Re-ranking with BERT*. 2020. arXiv: 1901.04085 [cs.IR]. URL: https://arxiv.org/abs/1901.04085.

[335] Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. "From doc2query to docTTTTTquery". In: *Online preprint* 6.2 (2019).

[336] Rodrigo Nogueira et al. "Document Ranking with a Pretrained Sequence-to-Sequence Model". In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Ed. by Trevor Cohn, Yulan He, and Yang Liu. Online: Association for Computational Linguistics, Nov. 2020, pp. 708–718. URL: https://aclanthology.org/2020.findings-emnlp.63/.

[337] Rodrigo Nogueira et al. "Multi-stage document ranking with BERT". In: *arXiv preprint arXiv:1910.14424* (2019).

[338] Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. "The E2E dataset: New challenges for end-to-end generation". In: 2017.

[339] I. V. Oseledets. "Tensor-Train Decomposition". In: *SIAM Journal on Scientific Computing* 33.5 (2011), pp. 2295–2317. DOI: 10.1137/090752286. URL: https://doi.org/10.1137/090752286.

[340] Arnold Overwijk et al. "Clueweb22: 10 billion web documents with visual and semantic information". In: *arXiv preprint arXiv:2211.15848* (2022).

[341] Kaihang Pan et al. "Self-supervised Meta-Prompt Learning with Meta-Gradient Regularization for Few-shot Generalization". In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 1059–1077. DOI: 10.18653/v1/2023.findings-emnlp.75. URL: https://aclanthology.org/2023.findings-emnlp.75/.

[342] Abhishek Panigrahi et al. "Task-specific skill localization in fine-tuned language models". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 27011–27033.

[343] Kishore Papineni et al. "BLEU: a Method for Automatic Evaluation of Machine Translation". In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. 2002.

[344] Kishore Papineni et al. "Bleu: a method for automatic evaluation of machine translation". In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002, pp. 311–318.

[345] Marinela Parovic et al. "Cross-Lingual Transfer with Target Language-Ready Task Adapters". In: *Findings of the Association for Computational Linguistics: ACL 2023*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 176–193. DOI: 10.18653/v1/2023.findings-acl.13. URL: https://aclanthology.org/2023.findings-acl.13/.

[346] Andrew Parry et al. "Analyzing adversarial attacks on sequence-to-sequence relevance models". In: *European Conference on Information Retrieval*. Springer. 2024, pp. 286–302.

[347] Greg Pass, Abdur Chowdhury, and Cayley Torgeson. "A Picture of Search". In: *Proceedings of the 1st International Conference on Scalable Information Systems*. InfoScale '06. Hong Kong: Association for Computing Machinery, 2006, 1–es. ISBN: 1595934286. DOI: 10.1145/1146847.1146848. URL: https://doi-org.unimib.idm.oclc.org/10.1145/1146847.1146848.

[348] Panupong Pasupat and Percy Liang. "Compositional Semantic Parsing on Semi-Structured Tables". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Ed. by Chengqing Zong and Michael Strube. Beijing, China: Association for Computational Linguistics, July 2015, pp. 1470–1480. DOI: 10.3115/v1/P15-1142. URL: https://aclanthology.org/P15-1142.

[349] Karl Pearson. "VII. Note on regression and inheritance in the case of two parents". In: *proceedings of the royal society of London* 58.347-352 (1895), pp. 240–242.

[350] Georgios Peikos, Wojciech Kusa, and Symeon Symeonidis. "ASPIRE: assistive system for performance evaluation in IR". In: *European Conference on Information Retrieval*. Springer. 2025, pp. 65–71.

[351] Georgios Peikos, Gabriella Pasi, et al. "A Decision-Theoretic Framework to Multidimensional Relevance Estimation". In: *International Journal of Information Technology & Decision Making* (2025), pp. 1–43.

[352] Yifan Peng, Shankai Yan, and Zhiyong Lu. "Transfer Learning in Biomedical Natural Language Processing: An Evaluation of BERT and ELMo on Ten Benchmarking Datasets". In: *Proceedings of the 18th BioNLP Workshop and Shared Task*. Ed. by Dina Demner-Fushman et al. Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 58–65. URL: https://aclanthology.org/W19-5006/.

[353] Jonas Pfeiffer et al. "AdapterFusion: Non-Destructive Task Composition for Transfer Learning". In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Online: Association for Computational Linguistics, Apr. 2021, pp. 487–503. DOI: 10.18653/v1/2021.eacl-main.39. URL: https://aclanthology.org/2021.eacl-main.39.

[354] Jonas Pfeiffer et al. "AdapterHub: A Framework for Adapting Transformers". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020): Systems Demonstrations*. Online: Association for Computational Linguistics, 2020, pp. 46–54. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.7.

[355] Jonas Pfeiffer et al. "MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, Nov. 2020, pp. 7654–7673. DOI: 10.18653/v1/2020.emnlp-main.617. URL: https://aclanthology.org/2020.emnlp-main.617.

[356] Long N Phan et al. "Scifive: a text-to-text transformer model for biomedical literature". In: *arXiv preprint arXiv:2106.03598* (2021).

[357] Jason Phang et al. "Hypertuning: Toward adapting large language models without back-propagation". In: *International Conference on Machine Learning*. PMLR. 2023, pp. 27854–27875.

[358] Mohammad Taher Pilehvar and Jose Camacho-Collados. "WiC: the Word-in-Context Dataset for Evaluating Context-Sensitive Meaning Representations". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2019, pp. 1267–1273.

[359] Ruben Piperno et al. "Cross-lingual distillation for domain knowledge transfer with sentence transformers". In: *Knowledge-Based Systems* 311 (2025), p. 113079.

[360] Emmanouil Antonios Platanios et al. "Contextual Parameter Generation for Universal Neural Machine Translation". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Ed. by Ellen Riloff et al. Brussels, Belgium: Association for Computational Linguistics, Oct. 2018, pp. 425–435. DOI: 10.18653/v1/D18-1039. URL: https://aclanthology.org/D18-1039/.

[361] Edoardo M Ponti et al. "Combining modular skills in multitask learning". In: *arXiv preprint arXiv:2202.13914* (2022).

[362] Ben Poole et al. "On variational bounds of mutual information". In: *International conference on machine learning*. PMLR. 2019, pp. 5171–5180.

[363] Maja Popović. "chrF: character n-gram F-score for automatic MT evaluation". In: *Proceedings of the Tenth Workshop on Statistical Machine Translation*. 2015.

[364] Clifton Poth et al. "Adapters: A Unified Library for Parameter-Efficient and Modular Transfer Learning". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. 2023, pp. 149–160.

[365] Clifton Poth et al. "Adapters: A Unified Library for Parameter-Efficient and Modular Transfer Learning". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 149–160. URL: https://aclanthology.org/2023.emnlp-demo.13.

[366] Sameer Pradhan et al. "Towards robust linguistic analysis using ontonotes". In: *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*. 2013, pp. 143–152.

[367] Sai Prasanna, Anna Rogers, and Anna Rumshisky. "When BERT Plays the Lottery, All Tickets Are Winning". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2020, pp. 3208–3229.

[368] Sai Prasanna, Anna Rogers, and Anna Rumshisky. "When BERT Plays the Lottery, All Tickets Are Winning". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Bonnie Webber et al. Online: Association for Computational Linguistics, Nov. 2020, pp. 3208–3229. DOI: 10.18653/v1/2020.emnlp-main.259. URL: https://aclanthology.org/2020.emnlp-main.259/.

[369] Raul Puri et al. "Training Question Answering Models From Synthetic Data". In: *EMNLP 2020*. 2020.

[370] Ye Qi et al. "When and Why Are Pre-Trained Word Embeddings Useful for Neural Machine Translation?" In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. Ed. by Marilyn Walker, Heng Ji, and Amanda Stent. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 529–535. DOI: 10.18653/v1/N18-2084. URL: https://aclanthology.org/N18-2084/.

[371] Hong Qian, Yi-Qi Hu, and Yang Yu. "Derivative-Free Optimization of High-Dimensional Non-Convex Functions by Sequential Random Embeddings." In: *IJCAI*. 2016, pp. 1946–1952.

[372] Yujia Qin et al. "Exploring Universal Intrinsic Task Subspace for Few-Shot Learning via Prompt Tuning". In: *IEEE/ACM Trans. Audio, Speech and Lang. Proc.* 32 (July 2024), pp. 3631–3643. ISSN: 2329-9290. DOI: 10.1109/TASLP.2024.3430545. URL: https://doi.org/10.1109/TASLP.2024.3430545.

[373] Peijun Qing et al. "AlphaLoRA: Assigning LoRA Experts Based on Layer Training Quality". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 20511–20523. DOI: 10.18653/v1/2024.emnlp-main.1141. URL: https://aclanthology.org/2024.emnlp-main.1141/.

[374] Yingqi Qu et al. "RocketQA: An Optimized Training Approach to Dense Passage Retrieval for Open-Domain Question Answering". In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Kristina Toutanova et al. Online: Association for Computational Linguistics, June 2021, pp. 5835–5847. DOI: 10.18653/v1/2021.naacl-main.466. URL: https://aclanthology.org/2021.naacl-main.466.

[375] Alec Radford et al. "Language models are unsupervised multitask learners". In: *OpenAI blog* 1.8 (2019), p. 9.

[376] Colin Raffel et al. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *Journal of machine learning research* 21.140 (2020), pp. 1–67.

[377] Colin Raffel et al. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *J. Mach. Learn. Res.* 21.1 (Jan. 2020). ISSN: 1532-4435.

[378] Alessandro Raganato et al. "Reasoning Capabilities and Invariability of Large Language Models". In: *2024 IEEE/WIC International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT)*. IEEE. 2024, pp. 125–132.

[379] Hossein A Rahmani et al. "Llm4eval: Large language model for evaluation in ir". In: *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2024, pp. 3040–3043.

[380] Pranav Rajpurkar, Robin Jia, and Percy Liang. "Know What You Don't Know: Unanswerable Questions for SQuAD". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 2018, pp. 784–789.

[381] Pranav Rajpurkar et al. "SQuAD: 100,000+ Questions for Machine Comprehension of Text". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Ed. by Jian Su, Kevin Duh, and Xavier Carreras. Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 2383–2392. URL: https://aclanthology.org/D16-1264/.

[382] Pranav Rajpurkar et al. "SQuAD: 100,000+ Questions for Machine Comprehension of Text". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. 2016.

[383] Anastasiia Razdaibiedina et al. "Residual Prompt Tuning: improving prompt tuning with residual reparameterization". In: *Findings of the Association for Computational Linguistics: ACL 2023*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 6740–6757. DOI: 10.18653/v1/2023.findings-acl.421. URL: https://aclanthology.org/2023.findings-acl.421/.

[384] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. "Efficient parametrization of multi-domain deep neural networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 8119–8127.

[385] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. "Learning multiple visual domains with residual adapters". In: *Advances in neural information processing systems* 30 (2017).

[386] Waleed Reda, Abhinav Jangda, and Krishna Chintalapudi. "Task Specific Pruning with LLM-Sieve: How Many Parameters Does Your Task Really Need?" In: *arXiv preprint arXiv:2505.18350* (2025).

[387] Nils Reimers and Iryna Gurevych. "Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Bonnie Webber et al. Online: Association for Computational Linguistics, Nov. 2020, pp. 4512–4525. DOI: 10.18653/v1/2020.emnlp-main.365. URL: https://aclanthology.org/2020.emnlp-main.365.

[388] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Ed. by Kentaro Inui et al. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3982–3992. DOI: 10.18653/v1/D19-1410. URL: https://aclanthology.org/D19-1410.

[389] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: http://arxiv.org/abs/1908.10084.

[390] Nils Reimers and Iryna Gurevych. *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks*. 2019. DOI: 10.48550/ARXIV.1908.10084. URL: https://arxiv.org/abs/1908.10084.

[391] Pengjie Ren et al. "MELoRA: Mini-Ensemble Low-Rank Adapters for Parameter-Efficient Fine-Tuning". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 3052–3064. DOI: 10.18653/v1/2024.acl-long.168. URL: https://aclanthology.org/2024.acl-long.168/.

[392] Adithya Renduchintala, Tugrul Konuk, and Oleksii Kuchaiev. "Tied-LoRA: Enhancing parameter efficiency of LoRA with Weight Tying". In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Ed. by Kevin Duh, Helena Gomez, and Steven Bethard. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 8694–8705. DOI: 10.18653/v1/2024.naacl-long.481. URL: https://aclanthology.org/2024.naacl-long.481/.

[393] Luis Miguel Rios and Nikolaos V Sahinidis. "Derivative-free optimization: a review of algorithms and comparison of software implementations". In: *Journal of Global Optimization* 56.3 (2013), pp. 1247–1293.

[394] Stephen E. Robertson et al. "Okapi at TREC-3". In: *Text Retrieval Conference*. 1994. URL: https://api.semanticscholar.org/CorpusID:41563977.

[395] Stephen E. Robertson et al. "Okapi at TREC-3". In: *Proceedings of The Third Text REtrieval Conference, TREC 1994, Gaithersburg, Maryland, USA, November 2-4, 1994*. Ed. by Donna K. Harman. Vol. 500-225. NIST Special Publication. National Institute of Standards and Technology (NIST), 1994, pp. 109–126. URL: http://trec.nist.gov/pubs/trec3/papers/city.ps.gz.

[396] Anna Rogers and Sasha Luccioni. "Position: Key Claims in LLM Research Have a Long Tail of Footnotes". In: *Forty-first International Conference on Machine Learning*. 2024. URL: https://openreview.net/forum?id=M2cwkGleRL.

[397] Omid Rohanian et al. "Exploring the Effectiveness of Instruction Tuning in Biomedical Language Processing". In: *Artificial Intelligence in Medicine* 158 (2024), p. 103007. ISSN: 0933-3657. DOI: 10.1016/j.artmed.2024.103007. URL: https://www.sciencedirect.com/science/article/pii/S0933365724002495.

[398] Omid Rohanian et al. “On the effectiveness of compact biomedical transformers”. In: *Bioinformatics* 39.3 (2023), btad103.

[399] Amir Rosenfeld and John K Tsotsos. “Incremental learning through deep adaptation”. In: *IEEE transactions on pattern analysis and machine intelligence* 42.3 (2018), pp. 651–663.

[400] Simone Rossi, Sebastien Marmin, and Maurizio Filippone. “Walsh-hadamard variational inference for bayesian deep learning”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 9674–9686.

[401] Andreas Rücklé et al. “Adapterdrop: On the efficiency of adapters in transformers”. In: *arXiv preprint arXiv:2010.11918* (2020).

[402] Stuart J Russell and Peter Norvig. “Artificial Intelligence: A Modern Approach, Global Edition 4e”. In: (2021).

[403] Mohammadreza Sadeghi et al. “MoKA: mixture of Kronecker adapters”. In: *arXiv preprint arXiv:2508.03527* (2025).

[404] Gaurav Sahu et al. “Data Augmentation for Intent Classification with Off-the-shelf Large Language Models”. In: *Proceedings of the 4th Workshop on NLP for Conversational AI*. 2022, pp. 47–57.

[405] Keisuke Sakaguchi et al. “Winogrande: An adversarial winograd schema challenge at scale”. In: *Communications of the ACM* 64.9 (2021), pp. 99–106.

[406] Alireza Salemi et al. “Lamp: When large language models meet personalization”. In: *arXiv preprint arXiv:2304.11406* (2023).

[407] Gerard Salton. *Automatic information organization and retrieval.* McGraw Hill Text, 1968.

[408] Erik Tjong Kim Sang and Fien De Meulder. “Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition”. In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*. 2003, pp. 142–147.

[409] V Sanh. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.” In: *Proceedings of Thirty-third Conference on Neural Information Processing Systems (NIPS2019)*. 2019.

[410] Victor Sanh et al. *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.* 2019. eprint: 1910.01108. URL: https://arxiv.org/pdf/1910.01108.

[411] Maarten Sap et al. “Social IQa: Commonsense Reasoning about Social Interactions”. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. 2019, pp. 4463–4473.

[412] Donald J Schuirmann. "A comparison of the two one-sided tests procedure and the power approach for assessing the equivalence of average bioavailability". In: *Journal of pharmacokinetics and biopharmaceutics* 15.6 (1987), pp. 657–680.

[413] Roy Schwartz et al. "Green AI". In: *Commun. ACM* 63.12 (Nov. 2020), pp. 54–63. ISSN: 0001-0782. DOI: 10.1145/3381831. URL: https://doi.org/10.1145/3381831.

[414] Arijit Sehanobish et al. "Structured Unrestricted-Rank Matrices for Parameter Efficient Finetuning". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=MXOzgjlWDF.

[415] Noam Shazeer and Mitchell Stern. "Adafactor: Adaptive learning rates with sublinear memory cost". In: *International conference on machine learning*. PMLR. 2018, pp. 4596–4604.

[416] Noam Shazeer et al. *Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer*. 2017. arXiv: 1701.06538 [cs.LG]. URL: https://arxiv.org/abs/1701.06538.

[417] Noam Shazeer et al. "Outrageously large neural networks: The sparsely-gated mixture-of-experts layer". In: *arXiv preprint arXiv:1701.06538* (2017).

[418] Haoxiang Shi et al. "LayerConnect: Hypernetwork-Assisted Inter-Layer Connector to Enhance Parameter Efficiency". In: *Proceedings of the 29th International Conference on Computational Linguistics*. Ed. by Nicoletta Calzolari et al. Gyeongju, Republic of Korea: International Committee on Computational Linguistics, Oct. 2022, pp. 3120–3126. URL: https://aclanthology.org/2022.coling-1.276/.

[419] Zhengxiang Shi and Aldo Lipani. "DePT: Decomposed Prompt Tuning for Parameter-Efficient Fine-tuning". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=KjegfPGRde.

[420] Karen Simonyan. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[421] Sunayana Sitaram et al. "A survey of code-switched speech and language processing". In: *arXiv preprint arXiv:1904.00784* (2019).

[422] Matthew Snover et al. "A study of translation edit rate with targeted human annotation". In: *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*. 2006, pp. 223–231.

[423] Richard Socher et al. "Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank". In: *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1631–1642. URL: https://aclanthology.org/D13-1170.

[424] Effrosyni Sokli et al. "Investigating mixture of experts in dense retrieval". In: *arXiv preprint arXiv:2412.11864* (2024).

[425] Effrosyni Sokli et al. "Leveraging Cognitive Complexity of Texts for Contextualization in Dense Retrieval". In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. 2025, pp. 27071–27084.

[426] Effrosyni Sokli et al. "Mixture of Experts Approaches in Dense Retrieval Tasks". In: *arXiv preprint arXiv:2510.15683* (2025).

[427] Haobo SONG et al. "Increasing Model Capacity for Free: A Simple Strategy for Parameter Efficient Fine-tuning". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=H3IUunLy8s.

[428] Mirco Speretta and Susan Gauch. "Personalized Search Based on User Search Histories". In: *The 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*. Vol. 2005. Oct. 2005, pp. 622–628. ISBN: 0-7695-2415-X. DOI: 10.1109/WI.2005.114.

[429] Yusheng Su et al. "On Transferability of Prompt Tuning for Natural Language Processing". In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 3949–3969. DOI: 10.18653/v1/2022.naacl-main.290. URL: https://aclanthology.org/2022.naacl-main.290/.

[430] Sainbayar Sukhbaatar et al. *Branch-Train-MiX: Mixing Expert LLMs into a Mixture-of-Experts LLM*. 2024. arXiv: 2403.07816 [cs.CL]. URL: https://arxiv.org/abs/2403.07816.

[431] Tianxiang Sun et al. "BBTv2: Towards a Gradient-Free Future with Large Language Models". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 3916–3930. DOI: 10.18653/v1/2022.emnlp-main.259. URL: https://aclanthology.org/2022.emnlp-main.259/.

[432] Tianxiang Sun et al. "Black-box tuning for language-model-as-a-service". In: *International Conference on Machine Learning*. PMLR. 2022, pp. 20841–20855.

[433] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. "Lst: Ladder side-tuning for parameter and memory efficient transfer learning". In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 12991–13005.

[434] Yi-Lin Sung, Varun Nair, and Colin A Raffel. "Training Neural Networks with Fixed Sparse Masks". In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 24193–24205. URL: https://proceedings.neurips.cc/paper_files/paper/2021/file/cb2653f548f8709598e8b5156738cc51-Paper.pdf.

[435] Shayan A. Tabrizi et al. "PERSON: Personalized information retrieval evaluation based on citation networks". In: *Information Processing & Management* 54.4 (2018), pp. 630–656. ISSN: 0306-4573. DOI: https://doi.org/10.1016/j.ipm.2018.04.004. URL: https://www.sciencedirect.com/science/article/pii/S0306457317307811.

[436] Marzieh S Tahaei et al. "Kroneckerbert: Learning kronecker decomposition for pre-trained language models via knowledge distillation". In: *arXiv preprint arXiv:2109.06243* (2021).

[437] Alon Talmor et al. "CommonsenseQA 2.0: Exposing the Limits of AI through Gamification". In: *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*. 2021. URL: https://openreview.net/forum?id=qF7FlUT5dxa.

[438] Zeqi Tan et al. "Learning Global Controller in Latent Space for Parameter-Efficient Fine-Tuning". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 4044–4055. DOI: 10.18653/v1/2024.acl-long.222. URL: https://aclanthology.org/2024.acl-long.222/.

[439] Jie Tang et al. "ArnetMiner: Extraction and Mining of Academic Social Networks". In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '08. Las Vegas, Nevada, USA: Association for Computing Machinery, 2008, pp. 990–998. ISBN: 9781605581934. DOI: 10.1145/1401890.1402008. URL: https://doi-org.unimib.idm.oclc.org/10.1145/1401890.1402008.

[440] Ruixiang Tang et al. "Does synthetic data generation of llms help clinical text mining?" In: *arXiv preprint arXiv:2303.04360* (2023).

[441] Nandan Thakur et al. "BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models". In: *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*. 2021.

[442] Nandan Thakur et al. "BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models". In: *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*. 2021.

[443] Nandan Thakur et al. "SPRINT: A Unified Toolkit for Evaluating and Demystifying Zero-shot Neural Sparse Retrieval". In: *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '23. Taipei, Taiwan: Association for Computing Machinery, 2023, pp. 2964–2974. ISBN: 9781450394086. DOI: 10.1145/3539618.3591902. URL: https://doi.org/10.1145/3539618.3591902.

[444] Aaron Xuxiang Tian et al. "FanLoRA: Fantastic LoRAs and Where to Find Them in Large Language Model Fine-tuning". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*. Ed. by Franck Dernoncourt, Daniel Preoţiuc-Pietro, and Anastasia Shimorina. Miami, Florida, US: Association for Computational Linguistics, Nov. 2024, pp. 515–528. DOI: 10.18653/v1/2024.emnlp-industry.38. URL: https://aclanthology.org/2024.emnlp-industry.38/.

[445] Chunlin Tian et al. "HydraLoRA: An Asymmetric LoRA Architecture for Efficient Fine-Tuning". In: *Advances in Neural Information Processing Systems*. Ed. by A. Globerson et al. Vol. 37. Curran Associates, Inc., 2024, pp. 9565–9584. URL: https://proceedings.neurips.cc/paper_files/paper/2024/file/123fd8a56501194823c8e0dca00733df-Paper-Conference.pdf.

[446] Andrea Tirico et al. "Modelling Children's Language: Vikidia-based Resources to Support Italian Text Simplification". In: *2026 IEEE Conference on Artificial Intelligence (CAI)*. IEEE. 2026, pp. 1659–1664.

[447] SM Tonmoy et al. "A comprehensive survey of hallucination mitigation techniques in large language models". In: *arXiv preprint arXiv:2401.01313* (2024).

[448] Hugo Touvron et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. 2023. arXiv: 2307.09288 [cs.CL]. URL: https://arxiv.org/abs/2307.09288.

[449] Hugo Touvron et al. "Llama 2: Open foundation and fine-tuned chat models". In: *arXiv preprint arXiv:2307.09288* (2023).

[450] Marcos Treviso et al. "Efficient Methods for Natural Language Processing: A Survey". In: *Transactions of the Association for Computational Linguistics* 11 (2023), pp. 826–860. DOI: 10.1162/tacl_a_00577. URL: https://aclanthology.org/2023.tacl-1.48.

[451] Alan M Turing. "Computing machinery and intelligence". In: *Parsing the Turing test: Philosophical and methodological issues in the quest for the thinking computer*. Springer, 2007, pp. 23–65.

[452] Mojtaba Valipour et al. "DyLoRA: Parameter-Efficient Tuning of Pre-trained Models using Dynamic Search-Free Low-Rank Adaptation". In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Ed. by Andreas Vlachos and Isabelle Augenstein. Dubrovnik, Croatia: Association for Computational Linguistics, May 2023, pp. 3274–3287. DOI: 10.18653/v1/2023.eacl-main.239. URL: https://aclanthology.org/2023.eacl-main.239/.

[453] Mathias Vast et al. "Simple Domain Adaptation for Sparse Retrievers". In: *European Conference on Information Retrieval*. Springer. 2024, pp. 403–412.

[454] Ashish Vaswani et al. "Attention is all you need". In: *Advances in neural information processing systems* 30 (2017).

[455] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. "Cider: Consensus-based image description evaluation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 4566–4575.

[456] Ellen Voorhees et al. "TREC-COVID: constructing a pandemic information retrieval test collection". In: *ACM SIGIR Forum*. Vol. 54. 1. ACM New York, NY, USA. 2021, pp. 1–12.

[457] Tu Vu et al. "SPoT: Better Frozen Model Adaptation through Soft Prompt Transfer". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Smaranda Muresan, Preslav Nakov, and Aline Villavicencio. Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 5039–5059. DOI: 10.18653/v1/2022.acl-long.346. URL: https://aclanthology.org/2022.acl-long.346.

[458] Danilo Vucetic et al. "Efficient Fine-Tuning of BERT Models on the Edge". In: *2022 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2022, pp. 1838–1842. DOI: 10.1109/ISCAS48785.2022.9937567.

[459] David Wadden et al. "Fact or Fiction: Verifying Scientific Claims". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Ed. by Bonnie Webber et al. Online: Association for Computational Linguistics, Nov. 2020, pp. 7534–7550. URL: https://aclanthology.org/2020.emnlp-main.609/.

[460] Yixin Wan, Kuan-Hao Huang, and Kai-Wei Chang. "PIP: Parse-Instructed Prefix for Syntactically Controlled Paraphrase Generation". In: *Findings of the Association for Computational Linguistics: ACL 2023*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 10372–10380. DOI: 10.18653/v1/2023.findings-acl.659. URL: https://aclanthology.org/2023.findings-acl.659/.

[461] Alex Wang et al. "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding". In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 353–355. DOI: 10.18653/v1/W18-5446. URL: https://aclanthology.org/W18-5446.

[462] Alex Wang et al. "SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[463] Alex Wang et al. "Superglue: A stickier benchmark for general-purpose language understanding systems". In: *Advances in neural information processing systems* 32 (2019).

[464] Haowen Wang et al. "Customizable Combination of Parameter-Efficient Modules for Multi-Task Learning". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=G1Hlubz1fR.

[465] Haoyu Wang et al. "RoseLoRA: Row and Column-wise Sparse Low-rank Adaptation of Pre-trained Language Model for Knowledge Editing and Fine-tuning". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 996–1008. DOI: 10.18653/v1/2024.emnlp-main.57. URL: https://aclanthology.org/2024.emnlp-main.57/.

[466] Kexin Wang et al. "GPL: Generative Pseudo Labeling for Unsupervised Domain Adaptation of Dense Retrieval". In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 2345–2360. URL: https://aclanthology.org/2022.naacl-main.168/.

[467] Kexin Wang et al. "GPL: Generative Pseudo Labeling for Unsupervised Domain Adaptation of Dense Retrieval". In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 2345–2360. DOI: 10.18653/v1/2022.naacl-main.168. URL: https://aclanthology.org/2022.naacl-main.168.

[468] Liang Wang, Nan Yang, and Furu Wei. "Query2doc: Query Expansion with Large Language Models". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 2023, pp. 9414–9423.

[469] Liang Wang et al. "Text embeddings by weakly-supervised contrastive pre-training". In: *arXiv preprint arXiv:2212.03533* (2022).

[470] Qifan Wang et al. "APrompt: Attention Prompt Tuning for Efficient Adaptation of Pre-trained Language Models". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 9147–9160. DOI: 10.18653/v1/2023.emnlp-main.567. URL: https://aclanthology.org/2023.emnlp-main.567/.

[471] Shaowen Wang, Linxi Yu, and Jian Li. "LoRA-GA: Low-Rank Adaptation with Gradient Approximation". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=VaLAWrLHJv.

[472] Sheng Wang et al. "PRoLoRA: Partial Rotation Empowers More Parameter-Efficient LoRA". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 2829–2841. DOI: 10.18653/v1/2024.acl-long.156. URL: https://aclanthology.org/2024.acl-long.156/.

[473] Tuowei Wang et al. "Long Exposure: Accelerating Parameter-Efficient Fine-Tuning for LLMs under Shadowy Sparsity". In: *Proceedings of the International Conference for High Performance Computing, Networking, Storage, and Analysis*. SC '24. Atlanta, GA, USA: IEEE Press, 2024. ISBN: 9798350352917. DOI: 10.1109/SC41406.2024.00081. URL: https://doi.org/10.1109/SC41406.2024.00081.

[474] Wenhui Wang et al. "MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 5776–5788. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[475] Yaqing Wang et al. "AdaMix: Mixture-of-Adaptations for Parameter-efficient Model Tuning". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 5744–5760. DOI: 10.18653/v1/2022.emnlp-main.388. URL: https://aclanthology.org/2022.emnlp-main.388.

[476] Zhen Wang et al. "Multitask Prompt Tuning Enables Parameter-Efficient Transfer Learning". In: *The Eleventh International Conference on Learning Representations*. 2023. URL: https://openreview.net/forum?id=Nk2pDtuhTq.

[477] Zihan Wang et al. "Let the Expert Stick to His Last: Expert-Specialized Fine-Tuning for Sparse Architectural Large Language Models". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 784–801. DOI: 10.18653/v1/2024.emnlp-main.46. URL: https://aclanthology.org/2024.emnlp-main.46/.

[478] Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. "Neural Network Acceptability Judgments". In: vol. 7. Cambridge, MA: MIT Press, 2019, pp. 625–641. DOI: 10.1162/tacl_a_00290. URL: https://aclanthology.org/Q19-1040.

[479] Jason Wei et al. "Finetuned language models are zero-shot learners". In: *arXiv preprint arXiv:2109.01652* (2021).

[480] Yeming Wen and Swarat Chaudhuri. "Batched Low-Rank Adaptation of Foundation Models". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=w4abltTZ2f.

[481] Colin White et al. "Neural architecture search: Insights from 1000 papers". In: *arXiv preprint arXiv:2301.08727* (2023).

[482] Adina Williams, Nikita Nangia, and Samuel Bowman. "A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 1112–1122. DOI: 10.18653/v1/N18-1101. URL: https://aclanthology.org/N18-1101.

[483] Laurin Wischounig, Abdelrahman Abdallah, and Adam Jatowt. "Negative Sampling Techniques in Dense Retrieval: A Survey". In: *Findings of the Association for Computational Linguistics: EACL 2026*. Ed. by Vera Demberg, Kentaro Inui, and Lluís Marquez. Rabat, Morocco: Association for Computational Linguistics, Mar. 2026, pp. 3003–3020. ISBN: 979-8-89176-386-9. DOI: 10.18653/v1/2026.findings-eacl.157. URL: https://aclanthology.org/2026.findings-eacl.157/.

[484] Thomas Wolf et al. "Transformers: State-of-the-art natural language processing". In: *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*. 2020, pp. 38–45.

[485] Mitchell Wortsman et al. "Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time". In: *International conference on machine learning*. PMLR. 2022, pp. 23965–23998.

[486] Chengyue Wu et al. "π-Tuning: Transferring Multimodal Foundation Models with Optimal Multi-task Interpolation". In: *Proceedings of the 40th International Conference on Machine Learning*. Ed. by Andreas Krause et al. Vol. 202. Proceedings of Machine Learning Research. PMLR, 23–29 Jul 2023, pp. 37713–37727. URL: https://proceedings.mlr.press/v202/wu23t.html.

[487] Junda Wu et al. "Infoprompt: Information-theoretic soft prompt tuning for natural language understanding". In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 61060–61084.

[488] Junde Wu et al. "Medical sam adapter: Adapting segment anything model for medical image segmentation". In: *arXiv preprint arXiv:2304.12620* (2023).

[489] Muling Wu et al. "Advancing Parameter Efficiency in Fine-tuning via Representation Editing". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 13445–13464. DOI: 10.18653/v1/2024.acl-long.726. URL: https://aclanthology.org/2024.acl-long.726/.

[490] Taiqiang Wu et al. "Mixture-of-Subspaces in Low-Rank Adaptation". In: *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 7880–7899. DOI: 10.18653/v1/2024.emnlp-main.450. URL: https://aclanthology.org/2024.emnlp-main.450/.

[491] Zhengxuan Wu et al. "Reft: Representation finetuning for language models". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 63908–63962.

[492] Zhuofeng Wu et al. "IDPG: An Instance-Dependent Prompt Generation Method". In: *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Ed. by Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz. Seattle, United States: Association for Computational Linguistics, July 2022, pp. 5507–5521. DOI: 10.18653/v1/2022.naacl-main.403. URL: https://aclanthology.org/2022.naacl-main.403/.

[493] Zongru Wu et al. "Acquiring Clean Language Models from Backdoor Poisoned Datasets by Downscaling Frequency Space". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 8116–8134. DOI: 10.18653/v1/2024.acl-long.441. URL: https://aclanthology.org/2024.acl-long.441/.

[494] Shitao Xiao et al. *C-Pack: Packaged Resources To Advance General Chinese Embedding*. 2023. arXiv: 2309.07597 [cs.CL].

[495] Shitao Xiao et al. "RetroMAE: Pre-Training Retrieval-oriented Language Models Via Masked Auto-Encoder". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 538–548. URL: https://aclanthology.org/2022.emnlp-main.35/.

[496] Lee Xiong et al. "Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval". In: *International Conference on Learning Representations*. 2021.

[497] Jing Xu and Jingzhao Zhang. "Random Masking Finds Winning Tickets for Parameter Efficient Fine-tuning". In: *Forty-first International Conference on Machine Learning*. 2024. URL: https://openreview.net/forum?id=VrwIrAa1Lc.

[498] Lingling Xu et al. *Parameter-Efficient Fine-Tuning Methods for Pretrained Language Models: A Critical Review and Assessment*. 2023. arXiv: 2312.12148 [cs.CL]. URL: https://arxiv.org/abs/2312.12148.

[499] Runxin Xu et al. "Raise a Child in Large Language Model: Towards Effective and Generalizable Fine-tuning". In: *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Ed. by Marie-Francine Moens et al. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 9514–9528. DOI: 10.18653/v1/2021.emnlp-main.749. URL: https://aclanthology.org/2021.emnlp-main.749/.

[500] Xilie Xu, Jingfeng Zhang, and Mohan Kankanhalli. "AutoLoRa: An Automated Robust Fine-Tuning Framework". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=09xFexjhqE.

[501] Linting Xue et al. "mT5: A massively multilingual pre-trained text-to-text transformer". In: *Proceedings of the 2021 conference of the North American chapter of the association for computational linguistics: Human language technologies*. 2021, pp. 483–498.

[502] Xiaocong Yang et al. "Parameter-Efficient Tuning with Special Token Adaptation". In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Ed. by Andreas Vlachos and Isabelle Augenstein. Dubrovnik, Croatia: Association for Computational Linguistics, May 2023, pp. 865–872. DOI: 10.18653/v1/2023.eacl-main.60. URL: https://aclanthology.org/2023.eacl-main.60/.

[503] Yibo Yang et al. "CorDA: Context-Oriented Decomposition Adaptation of Large Language Models for Task-Aware Parameter-Efficient Fine-tuning". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=Gi00NVru6n.

[504] Yifan Yang et al. "LoRETTA: Low-Rank Economic Tensor-Train Adaptation for Ultra-Low-Parameter Fine-Tuning of Large Language Models". In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Ed. by Kevin Duh, Helena Gomez, and Steven Bethard. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 3161–3176. DOI: 10.18653/v1/2024.naacl-long.174. URL: https://aclanthology.org/2024.naacl-long.174/.

[505] Yongxin Yang and Timothy Hospedales. "A Unified Perspective on Multi-Domain and Multi-Task Learning". In: *3rd International Conference on Learning Representations (ICLR 2025)*. 2015.

[506] Can Yaras et al. "Compressible Dynamics in Deep Overparameterized Low-Rank Learning and Adaptation". In: *Proceedings of the 41st International Conference on Machine Learning*. Ed. by Ruslan Salakhutdinov et al. Vol. 235. Proceedings of Machine Learning Research. PMLR, 21–27 Jul 2024, pp. 56946–56965. URL: https://proceedings.mlr.press/v235/yaras24a.html.

[507] Jiacheng Ye et al. "ZeroGen: Efficient Zero-shot Learning via Dataset Generation". In: *EMNLP 2022*. 2022, pp. 11653–11669.

[508] Kang Min Yoo et al. "GPT3Mix: Leveraging Large-scale Language Models for Text Augmentation". In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. 2021, pp. 2225–2239.

[509] Beiming Yu, Zhenfei Yang, and Xiushuang Yi. "MoKA:Parameter Efficiency Fine-Tuning via Mixture of Kronecker Product Adaption". In: *Proceedings of the 31st International Conference on Computational Linguistics*. Ed. by Owen Rambow et al. Abu Dhabi, UAE: Association for Computational Linguistics, Jan. 2025, pp. 10172–10182. URL: https://aclanthology.org/2025.coling-main.679/.

[510] Ted Zadouri et al. "Pushing Mixture of Experts to the Limit: Extremely Parameter Efficient MoE for Instruction Tuning". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=EvDeiLv7qc.

[511] Rowan Zellers et al. "HellaSwag: Can a Machine Really Finish Your Sentence?" In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2019, pp. 4791–4800.

[512] Jingtao Zhan et al. "Disentangled modeling of domain and relevance for adaptable dense retrieval". In: *arXiv preprint arXiv:2208.05753* (2022).

[513] Dacao Zhang et al. "Optimizing low-rank adaptation with decomposed matrices and adaptive rank allocation". In: *Frontiers of Computer Science* 19.5 (2025), p. 195337.

[514] Fangzhao Zhang and Mert Pilanci. "Spectral Adapter: Fine-Tuning in Spectral Space". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=UoxuaOGV6B.

[515] Jingfan Zhang et al. "MiLoRA: Efficient Mixture of Low-Rank Adaptation for Large Language Models Fine-tuning". In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, Nov. 2024, pp. 17071–17084. DOI: 10.18653/v1/2024.

findings-emnlp.994. URL: https://aclanthology.org/2024.findings-emnlp.994/.

[516] Longhui Zhang et al. *RankingGPT: Empowering Large Language Models in Text Ranking with Progressive Enhancement*. 2023. arXiv: 2311.16720 [cs.IR].

[517] Mingyang Zhang et al. "LoRAPrune: Structured Pruning Meets Low-Rank Parameter-Efficient Fine-Tuning". In: *Findings of the Association for Computational Linguistics: ACL 2024*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 3013–3026. DOI: 10.18653/v1/2024.findings-acl.178. URL: https://aclanthology.org/2024.findings-acl.178/.

[518] Qingru Zhang et al. "Adaptive Budget Allocation for Parameter-Efficient Fine-Tuning". In: *The Eleventh International Conference on Learning Representations*. 2023.

[519] Renrui Zhang et al. *LLaMA-Adapter: Efficient Fine-tuning of Language Models with Zero-init Attention*. 2023. arXiv: 2303.16199 [cs.CV]. URL: https://arxiv.org/abs/2303.16199.

[520] Ruiyi Zhang et al. "AutoLoRA: Automatically Tuning Matrix Ranks in Low-Rank Adaptation Based on Meta Learning". In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 2024, pp. 5048–5060.

[521] Ruiyi Zhang et al. "AutoLoRA: Automatically Tuning Matrix Ranks in Low-Rank Adaptation Based on Meta Learning". In: *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Ed. by Kevin Duh, Helena Gomez, and Steven Bethard. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 5048–5060. DOI: 10.18653/v1/2024.naacl-long.282. URL: https://aclanthology.org/2024.naacl-long.282/.

[522] Sheng Zhang et al. "Record: Bridging the gap between human and machine commonsense reading comprehension". In: *arXiv preprint arXiv:1810.12885* (2018).

[523] Tianyi Zhang et al. "Revisiting Few-sample {BERT} Fine-tuning". In: *International Conference on Learning Representations*. 2021. URL: https://openreview.net/forum?id=cO1IH43yUF.

[524] Wenxuan Zhang et al. "DimA: A Parameter-efficient Fine-tuning Method with Knowledge Transfer Based on Transformer". In: *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*. Ed. by Nicoletta Calzolari et al. Torino, Italia: ELRA and ICCL, May 2024, pp. 4922–4934. URL: https://aclanthology.org/2024.lrec-main.441/.

[525] Xinyu Zhang et al. "MIRACL: A Multilingual Retrieval Dataset Covering 18 Diverse Languages". In: *Transactions of the Association for Computational Linguistics* 11 (2023), pp. 1114–1131.

[526] Yu Zhang and Qiang Yang. "A survey on multi-task learning". In: *IEEE transactions on knowledge and data engineering* 34.12 (2021), pp. 5586–5609.

[527] Zhen-Ru Zhang et al. "Towards Adaptive Prefix Tuning for Parameter-Efficient Language Model Fine-tuning". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 1239–1248. DOI: 10.18653/v1/2023.acl-short.107. URL: https://aclanthology.org/2023.acl-short.107/.

[528] Zhengkun Zhang et al. "HyperPELT: Unified Parameter-Efficient Language Model Tuning for Both Language and Vision-and-Language Tasks". In: *Findings of the Association for Computational Linguistics: ACL 2023*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 11442–11453. DOI: 10.18653/v1/2023.findings-acl.725. URL: https://aclanthology.org/2023.findings-acl.725/.

[529] Hao Zhao, Jie Fu, and Zhaofeng He. "Prototype-based HyperAdapter for Sample-Efficient Multi-task Tuning". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 4603–4615. DOI: 10.18653/v1/2023.emnlp-main.280. URL: https://aclanthology.org/2023.emnlp-main.280/.

[530] Hongyu Zhao, Hao Tan, and Hongyuan Mei. "Tiny-Attention Adapter: Contexts Are More Important Than the Number of Parameters". In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 6626–6638. DOI: 10.18653/v1/2022.emnlp-main.444. URL: https://aclanthology.org/2022.emnlp-main.444/.

[531] Ziwang Zhao et al. "Knowledgeable Parameter Efficient Tuning Network for Commonsense Question Answering". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki. Toronto, Canada: Association for Computational Linguistics, July 2023, pp. 9051–9063. DOI: 10.18653/v1/2023.acl-long.503. URL: https://aclanthology.org/2023.acl-long.503/.

[532] Ziyu Zhao et al. "LoraRetriever: Input-Aware LoRA Retrieval and Composition for Mixed Tasks in the Wild". In: *Findings of the Association for Computational Linguistics: ACL 2024*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 4447–4462. DOI: 10.18653/v1/2024.findings-acl.263. URL: https://aclanthology.org/2024.findings-acl.263/.

[533] Victor Zhong, Caiming Xiong, and Richard Socher. "Seq2sql: Generating structured queries from natural language using reinforcement learning". In: *arXiv preprint arXiv:1709.00103* (2017).

[534] Han Zhou et al. *AutoPEFT: Automatic Configuration Search for Parameter-Efficient Fine-Tuning*. 2024. arXiv: 2301.12132 [cs.CL]. URL: https://arxiv.org/abs/2301.12132.

[535] Hattie Zhou et al. "Deconstructing Lottery Tickets: Zeros, Signs, and the Supermask". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper_files/paper/2019/file/1113d7a76ffceca1bb350bfe145467c6-Paper.pdf.

[536] Jiawei Zhou et al. "Synthetic lies: Understanding ai-generated misinformation and evaluating algorithmic and human solutions". In: *CHI Conference on Human Factors in Computing Systems 2023*.

[537] Wei Zhu and Ming Tan. "SPT: Learning to Selectively Insert Prompts for Better Prompt Tuning". In: *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Ed. by Houda Bouamor, Juan Pino, and Kalika Bali. Singapore: Association for Computational Linguistics, Dec. 2023, pp. 11862–11878. DOI: 10.18653/v1/2023.emnlp-main.727. URL: https://aclanthology.org/2023.emnlp-main.727/.

[538] Wei Zhu et al. "IAPT: Instance-Aware Prompt Tuning for Large Language Models". In: *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by Lun-Wei Ku, Andre Martins, and Vivek Srikumar. Bangkok, Thailand: Association for Computational Linguistics, Aug. 2024, pp. 14285–14304. DOI: 10.18653/v1/2024.acl-long.771. URL: https://aclanthology.org/2024.acl-long.771/.

[539] Yaoming Zhu et al. "Counter-Interference Adapter for Multilingual Machine Translation". In: *Findings of the Association for Computational Linguistics: EMNLP 2021*. Ed. by Marie-Francine Moens et al. Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 2812–2823. DOI: 10.18653/v1/2021.findings-emnlp.240. URL: https://aclanthology.org/2021.findings-emnlp.240/.

[540] Yutao Zhu et al. "Large language models for information retrieval: A survey". In: *arXiv preprint arXiv:2308.07107* (2023).

[541] Simiao Zuo et al. *Taming Sparsely Activated Transformer with Stochastic Experts*. 2022. arXiv: 2110.04260 [cs.CL]. URL: https://arxiv.org/abs/2110.04260.

This Ph.D. thesis has been typeset by means of the TEX-system facilities. The typesetting engine was pdfLATEX. The document class was `toptesi`, by Claudio Beccari, with option `tipotesi=scudo`. This class is available in every up-to-date and complete TEX-system installation.