UNIVERSITÀ
DEGLI STUDI
DI MILANO

Doctoral Dissertation
Doctoral Program in Artificial Intelligence ($38^{th}$cycle)

# Theory and Tools for Structure-Aware Machine Learning on Biological Data

By

**Davide D'Ascenzo**

******

**Supervisor(s):**
Prof. Sebastiano Vigna, Supervisor
Prof. Nicolò Cesa-Bianchi, Co-Supervisor

**Doctoral Examination Committee:**
Prof. Maurizio Parton, Referee, University of Chieti-Pescara
Prof. Hernan Makse, Referee, City University of New York
Prof. Alessandro Rizzo, Politecnico di Torino

Politecnico di Torino ◇ Università degli Studi di Milano
2025

*This dissertation was written at the end of the doctoral programme funded by PNRR, Mission 4, Component 1 "Strengthening the provision of education services: from nurseries to universities" – Investment 4.1 "Extension of the number of PhD programmes and innovative doctoral programmes for public administration and cultural heritage", through Ministerial Decree no. 351 of April 9, 2022.*



# Declaration

I hereby declare that, the contents and organization 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.

Davide D'Ascenzo
2025

* This dissertation is presented in partial fulfillment of the requirements for **Ph.D. degree** in the Graduate School of Politecnico di Torino (ScuDo).

*To the curious and those hungry for knowledge, to the people of science*

# Acknowledgements

I would like to express my deepest gratitude to my advisor, Sebastiano Vigna, for his unconditional support, guidance, and the freedom he granted me throughout my research. He enabled me to explore multiple topics and grow as an independent researcher. I would also like to thank my co-advisor, Nicolò Cesa-Bianchi, for his support throughout these years. I am grateful to my informal mentor, Paolo Boldi, for the many inspiring conversations that have greatly influenced my research path.

I would like to thank the people with whom I shared most of the time in my lab at the University of Milan: Flavio, Dario, and Chiara, for the stimulating (and provocative) discussions, the help, and the friendship. I have shared my PhD journey with Flavio since the beginning, and I am grateful to have met a friend like him. I also want to thank Liz for the great time we spent together and for all the positive energy she always brings.

Special thanks go to my advisor, Tomaso Poggio, during my research visit at MIT. His passion for science, his dedication to research and to his students, and the inspiring discussions we had have fueled my passion for research even more. I also want to thank Lorenzo Rosasco, with whom I shared many stimulating discussions during my visit. My thanks also go to John, Dan, Pier, Ziyin, and all the people at CBMM for the engaging conversations we had.

I also thank all the wonderful people I met at the Schmidt Center and Microsoft Research, Sebastiano, Peter, Lorin, and all the others, for the memorable time we spent together. I first met Sebastiano in London during a summer school more than a year ago and we have been collaborating closely ever since. He is a great scientist and a great friend.

I want to thank my advisor Stefano Zacchiroli, who hosted me at Télécom Paris for six months. I had the pleasure to meet him and some of his students and collaborators, Stefano, Andrea, and Federico, with whom I spent enjoyable time.

There are many other colleagues and friends whom I want to thank: Martin, Sergio, Claudio, Julie, Guido, and all the other people I met during these years.

I am also thankful to the colleagues I met at the Politecnico di Torino, and especially to Stefano Di Carlo, for coordinating the PhD program and supporting me with all the administrative aspects.

Finally, I thank my family, my partner, and all my friends for the continuous support and encouragement throughout these years.

# Abstract

The exponential growth of biological data has created unprecedented opportunities for machine learning to advance our understanding of complex biological systems. However, traditional machine learning approaches often treat biological data as collections of independent features, ignoring the rich structural relationships that characterize biological systems across all scales of organization. This dissertation develops theory and methods for structure-aware machine learning, demonstrating how incorporating structural knowledge into deep learning models can improve both theoretical understanding and practical performance.

We begin by studying centrality measures under network growth, proving that closeness, harmonic centrality, and betweenness satisfy rank semi-monotonicity when an edge is added to an undirected network. We then address the "curse of dimensionality", proving that the success of deep learning stems from its ability to exploit compositional sparsity, a hierarchical property inherent to all efficiently Turing-computable functions. Moving from theory to practice, we demonstrate that structural knowledge can be encoded directly into training objectives. We introduce a hierarchical cross-entropy loss that embeds the cell ontology into the learning process, improving out-of-distribution generalization by 12-15% for atlas-scale single-cell annotation. Evaluated on over 6 million cells across diverse architectures, this approach recovers roughly half of the performance drop observed when models are applied to newly released studies. Finally, we address the practical bottlenecks of training on large-scale datasets by developing a high-throughput data loading solution that uses quasi-random sampling to enable efficient training on disk-resident datasets comprising hundreds of millions of cells.

# Contents

# List of Figures

# List of Tables

# Introduction

Machine learning succeeds when it captures the structural priors of the domain it models [1, 2]. Spatial locality enables convolutional neural networks to excel at image processing [3], translational equivariance underlies their robustness [4], and homophily drives graph neural networks success on social networks [5, 6].

The deep learning revolution of the past decade can be largely understood through this lens. Transformers dominate natural language processing not simply due to scale, but because self-attention mechanisms naturally capture the compositional and contextual structure of language [7]. Protein structure prediction achieved breakthrough accuracy with AlphaFold by incorporating geometric and physical constraints directly into the architecture [8, 9], exploiting the fact that protein folding is governed by physical forces that respect spatial symmetries. In each case, the decisive factor was not merely to apply more compute or data, but to design architectures whose inductive biases align with the underlying structure of the problem domain.

Yet biological systems exhibit fundamentally richer patterns than those captured by current architectures. Beyond spatial locality and homophily, biological organization manifests compositional hierarchies and symmetries that span molecular, cellular, and tissue scales [10, 11]; sparsity in regulatory networks where genes interact with only a small fraction of other genes [12, 13]; temporal dynamics that govern development and disease progression [14]; and explicit taxonomic structures refined through decades of experimental research [15]. These structural regularities remain often unexploited in today's models, despite being documented, formalized, and made computationally accessible through community ontologies and knowledge bases. This thesis develops theory and tools that enable machine learning to harness these structural inductive biases, with the ultimate goal of building predictive models that can reason about biological complexity.

# The Limitations of the Foundation Model Paradigm

Large-scale neural networks pre-trained on massive datasets, known as foundation models, have recently transformed multiple domains from natural language processing to computer vision. This paradigm promises a similar revolution in biology: train a single large model on all available biological data, then adapt it to specific prediction tasks through fine-tuning. The appeal is compelling, as biological data, particularly single-cell omics, is now being produced at unprecedented scale. Single-cell RNA sequencing technologies have become sufficiently affordable that datasets now routinely contain millions of cells [16, 17]. The Tahoe-100M dataset profiles over 100 million cells across 379 drug perturbations and 50 cancer cell lines [18], while CELLxGENE hosts over 93 million cells spanning diverse tissues and conditions. This scale has motivated the development of numerous foundation models for single-cell transcriptomics, including scBERT [19], Geneformer [20], scGPT [21], UCE [22], scFoundation [23] and CellFM [24], among others.

However, a growing body of evidence reveals that these foundation models consistently fail to generalize to out-of-distribution settings and are frequently outperformed by simple linear baselines. Multiple independent benchmarking studies have now documented this phenomenon. Ahlmann-Eltze et al. [25] compared five foundation models (scGPT, scFoundation, UCE, scBERT, and Geneformer) against simple baselines for predicting transcriptional responses to genetic perturbations. Strikingly, none of the deep learning approaches outperformed a simple additive model for combinatorial perturbations, nor did they exceed the baseline of predicting the mean response for unseen gene perturbations. Kedzierska et al. [26] performed zero-shot evaluation of Geneformer and scGPT, finding substantial limitations in their ability to generalize without fine-tuning. Bendidi et al. [27] introduced a biologically motivated evaluation framework comparing transcriptomics foundation models to classical techniques, concluding that scVI [28] and even simple PCA consistently outperform transformer-based foundation models on perturbation analysis tasks. Wu et al. [29] developed PerturBench, a comprehensive benchmarking platform, revealing that simple models frequently outperform complex architectures and that many published models suffer from mode collapse or posterior collapse.

The failure modes are systematic and reveals that foundation models trained on observational and perturbational single-cell data struggle with causal inference tasks such as perturbation prediction. They often fail to capture biologically meaningful

relationships that simpler, task-specific methods recover naturally. When evaluated on structural integrity, that is, whether the learned latent space preserves known biological relationships, transformer models lag behind methods that explicitly model gene-gene covariance structures [27]. These findings suggest that current foundation models, largely adapted from architectures developed for language (transformers) or computer vision (convolutional networks), do not adequately capture the structure of biological systems.

## The Limitations of the Scaling Hypothesis

The foundation model paradigm implicitly relies on scaling laws, which predict that model performance improves predictably as model size, data, and compute increase [30, 31]. However, recent empirical studies reveal important limitations to this scaling paradigm. First, scaling laws exhibit task-dependent behavior: while next-token prediction loss follows predictable power laws, downstream task performance shows inconsistent scaling patterns, with some capabilities emerging unpredictably and others plateauing [32]. Second, practical constraints increasingly limit continued scaling: high-quality human-generated text data is projected to be exhausted within years [33], while training models on synthetic data introduces risks of model collapse when models are iteratively trained on their own outputs [34].

More fundamentally, scaling does not address the structure mismatch problem. A larger model trained on more data will better memorize patterns present in the training distribution, but this does not guarantee generalization to out-of-distribution settings that require reasoning about causal mechanisms or physical constraints. In biology, this limitation is particularly acute. Predicting cellular responses to novel drug combinations requires understanding pathway interactions and feedback mechanisms. This structure cannot be simply learned from observational correlations but must be explicitly represented and reasoned about. This suggests that progress in biological machine learning requires not merely larger models, but models whose architectures and training objectives are designed to exploit biological structure.

## Toward Structure-Aware Models for Biology

If foundation models adapted from other domains prove insufficient, what principles should guide the design of models for biological prediction tasks? This thesis argues that the answer lies in explicitly modeling the structural regularities that characterize biological systems. Rather than treating biological data as unstructured high-dimensional vectors and hoping that scale will recover relevant patterns, we should build architectures and training procedures that directly encode known biological principles.

This approach departs from the foundation model paradigm in different ways. First, rather than pursuing domain-general architectures, we should develop task-specific inductive biases that exploit biological structure. Second, rather than relying solely on data-driven learning, we should incorporate decades of accumulated biological knowledge through ontologies, pathway databases, and physical constraints. Third, rather than treating all prediction errors as equivalent, we should design loss functions that respect the semantic relationships among biological entities, recognizing, for instance, that confusing two proteins in the same complex is less severe than confusing proteins from unrelated pathways.

This dissertation develops structure-aware approaches across different levels of abstraction, moving from axioms about (biological) networks, to the class of functions that can be learned, to concrete learning objectives and scalable training pipelines. Taken together, these levels form a coherent programme for building machine learning systems whose inductive biases are aligned with biological structure.

Chapter 1 is devoted to network analysis. Centrality measures have long served as tools for ranking nodes in biological networks, from gene regulatory graphs to protein-protein interaction networks, and are widely used to prioritize drug targets or essential genes. Yet most centralities were designed with social or technological networks in mind. By studying how centrality behaves under network growth, in particular under edge additions, we derive axioms such as semi-monotonicity and basin dominance that capture desirable stability properties when networks evolve. This analysis provides guidance for designing or selecting centrality-based inductive biases in biological settings, where new interactions are routinely added as knowledge accumulates or when networks evolve through developmental, disease, or evolutionary processes.

Beyond descriptive analysis, modern biological research increasingly demands predictive models capable of mapping high-dimensional molecular data to functional outcomes. Classical statistical learning theory predicts that such tasks should be intractable due to the exponential growth in sample and parameter complexity with increasing data dimensionality. Yet, deep neural networks routinely achieve strong generalization in these settings, defying classical expectations. Chapter 2 addresses this apparent paradox by examining a different aspect of structure—not the topology of a given graph, but the internal organization of the functions we aim to learn. Building on recent work on compositional sparsity, we show that all efficiently Turing-computable functions admit a hierarchical decomposition into low-arity constituents. This result explains why architectures that mirror known biological modularity, from signaling pathways to regulatory circuits, can approximate complex cellular input-output maps without incurring the curse of dimensionality, and it frames such architectures as explicit inductive biases on the space of admissible hypotheses. The analysis also sheds light on contemporary architectural choices more broadly, from convolutional networks to transformer models, and connects to phenomena such as Chain-of-Thought reasoning in large language models.

Chapter 3 shifts focus from the properties of functions to the properties of their output spaces. Canonical machine learning tasks treat output classes as independent and equidistant, but biological systems exhibit rich hierarchical organization encoded in ontologies developed over decades. Standard cross-entropy loss ignores this structure, forcing models to make artificial choices between related labels. We introduce a hierarchical cross-entropy loss that incorporates ontological structure directly into the training objective, focussing on automated cell type annotation in single-cell RNA sequencing data. Empirical results demonstrate that encoding known biological relationships significantly improves generalization to newly released studies, addressing the practical challenge of annotating data from newly generated experiments.

Finally, Chapter 4 addresses the practical constraints that arise when training deep learning models at the scale of contemporary biological datasets. Single-cell atlases now contain over 100 million cells, but efficient training typically requires loading entire datasets into memory. Alternative approaches either drastically increase storage requirements through format conversion or achieve prohibitively low throughput through random disk access. The chapter introduces scDataset, a framework that implements quasi-random sampling through block sampling and

batched fetching strategies. Block sampling reduces random disk reads by accessing contiguous chunks, while batched fetching amortizes latency and enables in-memory reshuffling for minibatch diversity. Benchmarks demonstrate up to 129-fold speed improvements while maintaining sampling diversity comparable to true random shuffling.

# Chapter 1

# Axiomatization of Network Centrality Measures

*The content of this chapter is largely based on the research originally presented in [35, 36].*

Networks provide the mathematical language for representing relationships in complex systems, offering a powerful framework for modeling everything from the intricate web of protein-protein interactions that govern cellular processes [37, 38] to the vast social networks that shape the flow of information and influence in human societies [39, 40]. Understanding which actors, be they proteins, people, or corporations, occupy the most "important" positions in these networks is of compelling interest across scientific domains. In epidemiology, identifying central individuals can guide strategies for disease control. In molecular biology, pinpointing central proteins can highlight key targets for drug therapies. In social science, understanding the locus of power can reveal the dynamics of influence and organization [41].

Network science formalizes the analysis of these relational architectures, and the concept of centrality has emerged as the primary tool for translating a node's structural position into a quantitative measure of its importance. However, the notion of "importance" is not monolithic; it is a multifaceted concept that has evolved through a series of conceptual shifts, each representing a different understanding of what constitutes a valuable connection. Next, we sketch a brief history that provides the essential context for the questions at the heart of this chapter.

## 1.1 History of Centrality

The most intuitive and direct measure of importance is *degree centrality*, which simply counts the number of connections a node possesses. Rooted in the early work of sociometry [42], this measure captures the idea of activity or immediate connectivity. A person with many friends or a protein that interacts with many other proteins is, in this local sense, highly engaged and potentially important. While simple, degree centrality has proven to be a surprisingly powerful predictor in many domains, such as correlating with gene essentiality in protein interaction networks [43, 44].

Yet for all its utility, degree centrality provides only a local perspective. The systematic effort to quantify importance by considering a node's role in mediating connections across the entire network began in the mid-20th century with early work in social network analysis. The first major conceptual leap was to look beyond immediate connections and consider a node's position within the global topology of the entire network. This shift, pioneered by Alex Bavelas [45, 46] and formalized by Linton Freeman [47], focused on the role of shortest paths, or geodesics, in mediating network processes.

**Closeness Centrality** Introduced by Bavelas in the context of communication networks, closeness centrality redefines importance as efficiency. A node is considered central not because of its number of connections, but because of its ability to reach (or being reached by) all other nodes in the network quickly. It is calculated as the reciprocal of the sum of shortest path distances to all other nodes. A node with high closeness has high "access" to the rest of the network, making it an efficient hub for information dissemination [45, 46]. As with degree centrality, closeness has been used to identify essential genes in regulatory networks [48].

**Betweenness Centrality** Freeman's seminal paper provided a comprehensive framework that introduced betweenness centrality. This measure captures the notion of brokerage or control. A node is important if it lies on a high proportion of the shortest paths connecting other pairs of nodes. Such nodes act as gatekeepers or intermediaries, potentially controlling the flow of information, resources, or influence throughout the network. Freeman's work was pivotal because it established that

centrality was not a single property but a class of concepts—degree (local activity), closeness (global reachability), and betweenness (intermediary role)—each capturing a distinct facet of structural importance [47]. In biological contexts, betweenness has been used to highlights bottleneck proteins whose disruption can fragment interaction networks [49, 48].

The next conceptual shift moved beyond the restrictive focus on shortest paths to a more holistic view where influence or status could flow along all possible paths. This approach treats influence as a kind of signal that propagates through the network, decaying with distance.

**Katz index** Developed by sociologist Leo Katz in 1953, this measure was one of the first to formalize this idea. It defines a node's centrality as a weighted sum of all paths ending at that node and introduces an attenuation factor, $\alpha$, which ensures that longer paths contribute less to the final score. This innovation was not only necessary for the mathematical convergence of the infinite sum of paths but also intuitively realistic: an endorsement from a close neighbor is more valuable than one from a distant acquaintance. Katz index thus provides a more nuanced measure of influence that accounts for the multiplicity of connection pathways [50].

Finally, one can define importance recursively: a node's status is a function of the status of the nodes to which it is connected. This captures the social and economic reality that a connection's value is not intrinsic but is derived from the importance of the entity it connects to.

**Eigenvector Centrality** Pioneered by Phillip Bonacich, eigenvector centrality formalizes this recursive intuition directly. It posits that a node's centrality is proportional to the sum of the centralities of its neighbors. The solution to this system of linear equations is the principal eigenvector of the network's adjacency matrix. This measure elegantly captures the idea that receiving a link from a highly central node contributes more to one's own centrality than receiving a link from a peripheral one [51].

**PageRank** Developed by Sergey Brin and Lawrence Page for ranking web pages, PageRank is a highly influential and robust variant of eigenvector centrality. It builds upon the same recursive foundation but introduces two modifications to handle the

unique challenges of the web graph: a damping factor and a mechanism for handling dangling nodes (pages with no outgoing links). This is often conceptualized through the "random surfer" model, where a user randomly clicks on links but occasionally "teleports" to a random page in the network [52]. These modifications ensure that the resulting centrality score is unique, always positive, and can be computed efficiently on massive, disconnected graphs, demonstrating how abstract centrality concepts could be scaled to have profound commercial and societal impact.

## 1.2 Axiomatization

The proliferation of centrality measures has led to a rich but fragmented landscape, where different measures often yield divergent rankings of node importance. This diversity raises questions about the principles that should guide the choice and interpretation of centrality measures. To address these questions, researchers have turned to axiomatic approaches, seeking to identify the core properties that a centrality measure should satisfy to be considered valid or useful in various contexts [53, 54]. These axiomatic frameworks typically focus on desirable properties, a fundamental one being anonymity (or isomorphic invariance), which guarantees that centrality depends only on a node's position in the network structure rather than on arbitrary node labels or identities [53]. By formalizing these intuitions, researchers aim to create a more coherent understanding of centrality and its implications across different domains.

While these axiomatic approaches provide a rigorous foundation for understanding centrality in static networks, they leave open the question of what happens when networks evolve. Real-world systems, particularly biological ones, are inherently dynamic. Protein interactions form and dissolve [55, 56], gene regulatory relationships evolve during development [57], and cellular networks rewire in response to environmental perturbations [58]. Understanding how node importance changes as networks evolve is important for modeling disease progression [59], predicting therapeutic targets [60], and designing robust biological circuits [61]. In this chapter, we address this question in the simplest case: when a network grows by the addition of a single edge.

Intuitively, any reasonable measure of importance should satisfy two basic properties when a network grows by the addition of an edge. First, *score monotonicity*

requires that the centrality scores of the two endpoints of the new edge should increase. Gaining a new connection should not make one less important. Second, *rank monotonicity* requires that the relative ranking of the endpoints should not deteriorate; any node that was previously less central than an endpoint should remain so. In the *strict* version of rank monotonicity, any ties in ranking are additionally resolved in favor of the endpoints.

For directed networks, where an arc represents a one-way relationship like a "follow" on social media or a citation, these properties generally hold. Gaining a new incoming link is almost always beneficial to a node's score and rank [54, 62]. However, undirected networks, which represent symmetric relationships like friendships or physical protein interactions, present a far more complex and counter-intuitive picture. While score monotonicity often holds for many measures, rank monotonicity frequently fails: adding an undirected edge can actually *reduce* the rank of one of its endpoints relative to other nodes in the network [63].

This "monotonicity paradox" has implications for interpreting centrality-based analyses. In social network terms, while getting a new follower (directed relationship) is always beneficial, getting a new friend (undirected relationship) might sometimes be detrimental to one's relative importance [63]. For biological networks, this suggests that forming new protein interactions or regulatory connections may have non-monotonic effects on a gene's or protein's importance.

A key unanswered question from Boldi et al. [63] is whether adding an edge can decrease the rank of both endpoints simultaneously. In other words: when an edge is added to an *undirected* network, does the importance of *at least one* of its two endpoints increase?

This chapter provides a positive answer to this question by introducing *semi-monotonicity*, a weaker condition than monotonicity in which we require that *at least one* endpoint of the new edge in an undirected network enjoys monotonicity. We demonstrate that several of the most important centralities, including closeness, harmonic and betweenness, do satisfy rank semi-monotonicity.

The chapter is structured as follows. Section 1.3 defines score semi-monotonicity and (strict) rank semi-monotonicity. Section 1.4 introduces $\delta$-monotonicity and $\delta$-semi-monotonicity, linking these concepts to rank (semi-)monotonicity. Section 1.5 presents basin dominance, a stronger property that applies to most centralities discussed here. We then show in Section 1.6 that basin dominance holds also for

*distance-decay centralities* satisfying a (discrete) convexity condition on their decay function. The condition is necessary and sufficient, thus characterizing basin dominance for those centralities. This leads to results for closeness, harmonic centrality, and centralities with exponential or power-law decay. Finally, in Section 1.7 we show that betweenness is basin dominant. Along the way, we show that, unlike harmonic centrality, closeness and betweenness are not strictly semi-monotonic, echoing similar results from the directed case. In Section 1.8, we end the chapter by drawing some conclusions and interpretations of our results.

## 1.3 Score and Rank Semi–Monotonicity

Although the chapter focuses on undirected graphs, we occasionally reference directed graphs for comparison; the following definitions about graphs are standard (see, for instance, [64]). A *directed graph* is a pair $G = (V_G, A_G)$ where $V_G$ is a set of nodes and $A_G \subseteq V_G \times V_G$ is a set of ordered pairs, called *arcs*; we will denote the existence of an arc from $x$ to $y$ with the notation $x \to y$. An *undirected graph* is a directed graph such that $x \to y$ iff $y \to x$; when this happens, we say that there is an *edge* between $x$ and $y$, denoted by the notation $x \text{ — } y$, or, equivalently, we say that $x$ and $y$ are *adjacent*.

We use $G + x \to y$ for the directed graph $G$ with the added arc $x \to y$, and $G + x \text{ — } y$ for the undirected graph $G$ with the added edge $x \text{ — } y$.

A (simple) *path* of length $k$ from $x$ to $y$ in an undirected graph $G$ is a sequence of distinct vertices $x_0, x_1, \ldots, x_k$ such that $x_0 = x$, $x_k = y$, and $x_{i-1} \text{ — } x_i$ for every $1 \leq i \leq k$. The *distance* $d_G(x,y)$ between $x$ and $y$ in $G$ is the length of a shortest path from $x$ to $y$ in $G$; if there is no such path, we set $d_G(x,y) = \infty$. When $G$ is clear from the context, we abbreviate to $d_{xy}$.

A *centrality measure* is a function $c : V_G \to \mathbf{R}$, assigning real values, which we also refer to as *scores*, to all the vertices of a graph $G$, where vertices with larger scores should be interpreted as having a greater structural importance in the network. We assume all centrality measures to be invariant under isomorphism.

We start by recalling the definition of score and rank monotonicity on undirected graphs, which were introduced in [63] as a natural extension of score and rank

monotonicity on directed graphs [62, 65]. Score monotonicity requires that both endpoints see their scores increase when an edge is added. Formally:

**Definition 1.1** (Score monotonicity)**.** Given an undirected graph $G$, a centrality $c$ is said to be *score monotone on G* if for every pair of distinct,[1] non-adjacent vertices $x$ and $y$ we have that

$$c(x) < c'(x) \quad \textit{and} \quad c(y) < c'(y),$$

where $c'$ is the value of the centrality on the graph $G + x \text{ — } y$.

Note that, in general, we will give definitions of properties for a centrality $c$ on a graph $G$, and we will say that the same property holds *on a set of graphs* if it is true on all the graphs from the set.

An increase in score does not guarantee that the relative positions of the endpoints and other nodes remain stable. This observation motivates the definition of rank monotonicity [62, 63], which ensures that nodes previously outranked by the endpoints continue to be so after adding the edge.[2] Formally, we can consider two versions of rank monotonicity:

**Definition 1.2** (Rank monotonicity)**.** Given an undirected graph $G$, a centrality $c$ is said to be *rank monotone on G* if for every pair of distinct, non-adjacent vertices $x$ and $y$ the following two statements hold:

- for all vertices $z \neq x, y$:

$$\begin{aligned} c(z) < c(x) \quad &\text{implies} \quad c'(z) < c'(x) \text{ and} \\ c(z) = c(x) \quad &\text{implies} \quad c'(z) \leq c'(x), \end{aligned}$$

[1] It is unfortunate that the assumption $x \neq y$ was not stated explicitly in the definition of score monotonicity given in [54], although being part of Sabidussi's original definition [53]; the same restriction is necessary for strict rank monotonicity [62]. Indeed, adding loops cannot change the value of any centrality depending on shortest paths, so no such centrality could be score monotone or strictly rank monotone without the assumption $x \neq y$. On the other hand, the results in [62, 65] show that spectral centralities satisfy a stronger definition of strict rank monotonicity that includes the possibility of adding loops.

[2] The original definition of rank monotonicity for directed graphs was given in [66].

- for all vertices $z \neq x, y$:

$$\begin{aligned} c(z) < c(y) \quad &\text{implies} \quad c'(z) < c'(y) \text{ and} \\ c(z) = c(y) \quad &\text{implies} \quad c'(z) \leq c'(y). \end{aligned}$$

where $c'$ is the value of the centrality on the graph $G + x \text{ — } y$.

A strict (in fact, simpler) version of the previous definition can be introduced, requiring that adding the new edge breaks all ties in favor of the two endpoints of the new edge itself:

**Definition 1.3** (Strict rank monotonicity)**.** Given an undirected graph $G$, a centrality $c$ is said to be *strictly rank monotone on* $G$ if for every pair of distinct, non-adjacent vertices $x$ and $y$ the following two statements hold:

- for all vertices $z \neq x, y$: $\quad c(z) \leq c(x) \quad \text{implies} \quad c'(z) < c'(x)$,
- for all vertices $z \neq x, y$: $\quad c(z) \leq c(y) \quad \text{implies} \quad c'(z) < c'(y)$.

where $c'$ is the value of the centrality on the graph $G + x \text{ — } y$.

Semi-monotonicity is a weaker condition than monotonicity, originating from the observation that in undirected networks some centrality measures do not satisfy score or rank monotonicity. Indeed, adding an edge to a graph can reduce the score or the rank of one of the two endpoints of the edge [63]. Thus, we just require that adding a new edge increases the score of *at least one* of the two endpoints:

**Definition 1.4** (Score semi-monotonicity)**.** Given an undirected graph $G$, a centrality $c$ is said to be *score semi-monotone on* $G$ if for every pair of distinct, non-adjacent vertices $x$ and $y$ we have that

$$c(x) < c'(x) \quad \textit{or} \quad c(y) < c'(y).$$

where $c'$ is the value of the centrality on the graph $G + x \text{ — } y$.

Similarly, rank semi-monotonicity means that adding a new edge increases the rank of *at least one* of the two endpoints, and as before comes in two versions:

**Definition 1.5** (Rank semi-monotonicity)**.** Given an undirected graph $G$, a centrality $c$ is said to be *rank semi-monotone on* $G$ if for every pair of distinct, non-adjacent vertices $x$ and $y$ at least one of the following two statements holds:

- for all vertices $z \neq x, y$:

$$\begin{aligned} c(z) < c(x) \quad &\text{implies} \quad c'(z) < c'(x) \text{ and} \\ c(z) = c(x) \quad &\text{implies} \quad c'(z) \leq c'(x), \end{aligned}$$

- for all vertices $z \neq x, y$:

$$\begin{aligned} c(z) < c(y) \quad &\text{implies} \quad c'(z) < c'(y) \text{ and} \\ c(z) = c(y) \quad &\text{implies} \quad c'(z) \leq c'(y). \end{aligned}$$

where $c'$ is the value of the centrality on the graph $G + x \text{ — } y$.

Again, the strict version is simpler:

**Definition 1.6** (Strict rank semi-monotonicity)**.** Given an undirected graph $G$, a centrality $c$ is said to be *strictly rank semi-monotone on* $G$ if for every pair of distinct, non-adjacent vertices $x$ and $y$ at least one of the following two statements holds:

- for all vertices $z \neq x, y$: $\quad c(z) \leq c(x) \quad \text{implies} \quad c'(z) < c'(x)$,
- for all vertices $z \neq x, y$: $\quad c(z) \leq c(y) \quad \text{implies} \quad c'(z) < c'(y)$.

where $c'$ is the value of the centrality on the graph $G + x \text{ — } y$.

## 1.4 $\delta$–Monotonicity and $\delta$–Semi–Monotonicity

When proving rank monotonicity of a centrality measure, working purely on order relationship on the whole graph can be tricky: it is often easier to prove a stronger but more amenable property that provides a sufficient condition for rank monotonicity based on value differences, rather than on order: indeed, the authors of [62] used such a property to prove rank monotonicity for closeness centrality, harmonic centrality, PageRank, and Katz index, without explicitly naming it.

This observation motivates us to introduce the concepts of $\delta$-monotonicity and $\delta$-semi-monotonicity, and to show that they indeed provide a sufficient condition for, respectively, rank monotonicity and rank semi-monotonicity. The idea underlying $\delta$-monotonicity is that the score increase of the target of a new arc should be at least as large as the score increase of all other vertices. This approach formalizes the idea underlying the proofs of [62] and factors out the common structure of the proofs of rank semi-monotonicity in the rest of the chapter.

We start with the definition of $\delta$-monotonicity for directed graphs:

**Definition 1.7** ($\delta$-monotonicity (directed))**.** Given a directed graph $G$, a centrality $c$ is said to be *$\delta$-monotone on $G$* if for every pair of distinct vertices $x$ and $y$ such that $x \not\to y$ the following holds:

$$c'(z) - c(z) \leq c'(y) - c(y) \qquad \text{for every } z \neq y.$$

where $c'$ is the value of the centrality on the graph $G + x \to y$. It is said to be *strictly $\delta$-monotone* if the inequality is strict.

The undirected version of $\delta$-monotonicity follows the same pattern of the monotonicity properties of the previous section:

**Definition 1.8** ($\delta$-monotonicity (undirected))**.** Given an undirected graph $G$, a centrality $c$ is said to be *$\delta$-monotone on $G$* if for every pair of distinct, non-adjacent vertices $x$ and $y$ the following holds:

$$\begin{aligned} &c'(z) - c(z) \leq c'(x) - c(x) \qquad \text{for every } z \neq x, y \qquad \text{and} \\ &c'(z) - c(z) \leq c'(y) - c(y) \qquad \text{for every } z \neq x, y. \end{aligned}$$

where $c'$ is the value of the centrality on the graph $G + x \text{ — } y$. It is said to be *strictly $\delta$-monotone* if the inequalities are strict.

Finally, only for undirected graphs we define $\delta$-semi-monotonicity, whose definition parallels the weakening of monotonicity we discussed in the previous section:

**Definition 1.9** ($\delta$-semi-monotonicity)**.** Given an undirected graph $G$, a centrality $c$ is said to be *$\delta$-semi-monotone on $G$* if for every pair of distinct, non-adjacent vertices $x$

and $y$ the following holds:

$$\begin{aligned} c'(z) - c(z) &\leq c'(x) - c(x) \qquad \text{for every } z \neq x, y \qquad \text{or} \\ c'(z) - c(z) &\leq c'(y) - c(y) \qquad \text{for every } z \neq x, y. \end{aligned}$$

where $c'$ is the value of the centrality on the graph $G + x \text{—} y$. It is said to be *strictly $\delta$-semi-monotone* if the inequalities are strict.

We now prove that $\delta$-monotonicity implies rank monotonicity, and $\delta$-semi-monotonicity implies rank semi-monotonicity. This is intuitive, as the score increase of the target of a new arc (or of the endpoints of a new edge, in the undirected case) is at least as large as the increase in score of all other vertices, so its order relationship with other vertices can only improve.

**Theorem 1.10.** *If a centrality measure is (strictly) $\delta$-monotone on a (directed or undirected) graph then it is (strictly) rank monotone on the same graph.*

*Proof.* We prove this result for a directed graph: a similar proof can be obtained for the undirected case. By the definition of $\delta$-monotonicity, we know that when adding an arc $x \to y$ to $G$, the inequality

$$c'(z) - c(z) \leq c'(y) - c(y)$$

holds for every $z \neq y$, the inequality being strict in the strict case. Adding this inequality to the left-hand sides of the implication appearing in the definition of rank monotonicity, we obtain

$$\begin{aligned} c(z) < c(y) \quad &\text{implies} \quad c'(z) < c'(y) \quad \text{for every } z \neq y, \\ c(z) = c(y) \quad &\text{implies} \quad c'(z) \leq c'(y) \quad \text{for every } z \neq y. \end{aligned}$$

The proof for the strict case is analogous. □

**Theorem 1.11.** *If a centrality measure is (strictly) $\delta$-semi-monotone on a graph then it is (strictly) rank semi-monotone on the same graph.*

The proof can be straightforwardly derived as for Theorem 1.10 using the appropriate hypotheses and definitions.

## 1.5 Basin Dominance

Finally, we introduce an even stronger property than $\delta$-semi-monotonicity, *basin dominance*: this property is related to the notion of distance in a graph, and it will be instrumental in the proofs of rank semi-monotonicity for centralities based on shortest paths in the next sections.

We start with the notion of *basin*, formalizing the idea that given two vertices $x$, $y$ in an undirected graph we can classify the vertices of the graph depending on whether they are closer to $x$ or to $y$, with possibly an overlap for the vertices that are equidistant from $x$ and $y$. This concept appeared implicitly in some recent works [67, 68] where graph distances are used to determine winners of pairwise comparisons, in a *graph-as-an-election* fashion. In particular, between two candidate vertices $x$ and $y$ of a graph, voters give their preference to the one whom they are strictly closer to. In fact, we can trace the notion back at least to [69, Property 2.2] (albeit in [69, 67, 68] equidistant vertices are not considered).

**Definition 1.12** (Basin)**.** Given an undirected graph $G$ with vertex set $V_G$ and two vertices $x$ and $y$, we define the *basin of x (with respect to y)* $B_{xy}$ and the *basin of y (with respect to x)* $B_{yx}$ as

$$B_{xy} = \{\, u \in V_G \mid d_{ux} \leq d_{uy} \,\}$$
$$B_{yx} = \{\, u \in V_G \mid d_{uy} \leq d_{ux} \,\}$$

Figure 1.1 shows an example of a graph with the basins of two vertices; the vertices that are equidistant from $x$ and $y$ are included in both basins. We note a few useful facts:

- if $x$ is not adjacent to $y$, adding the edge $x$ — $y$ to the graph leaves the basins unchanged; moreover, the shortest paths between $x$ and the vertices in its basin $B_{xy}$ do not change (the same is true for $y$);
- if $z$ is equidistant from $x$ and $y$ and they are non-adjacent, the shortest paths between $z$ and other vertices do not change because of the addition of the edge $x$ — $y$; as a consequence, the score of $z$ will remain the same after the addition of $x$ — $y$ in any centrality depending only on shortest paths;

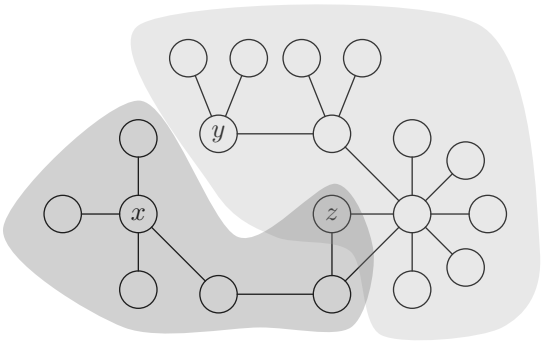


Fig. 1.1 An undirected graph $G$, with $B_{xy}$ (the basin of x w.r.t. y) shown in dark grey and $B_{yx}$ (the basin of y w.r.t. x) in light grey. Note that $z$ is equidistant from $x$ and $y$, and thus it is included in both basins.

- let $u$ be a vertex in the basin of $x$; then, a shortest path between $u$ and $x$ passes exclusively through vertices in the basin of $x$; the same holds symmetrically for a shortest path between $v \in B_{yx}$ and $y$; moreover, a shortest path between $u$ and $y$ passes first exclusively through the basin of $x$, and then exclusively through the basin of $y$; again, everything holds symmetrically for a shortest path between $v \in B_{yx}$ and $x$;
- shortest paths between arbitrary vertices can zig-zag between basins multiple times (see Figure 1.2);
- if $x$ — $y$, the difference of the sum of distances from $x$ and from $y$ is the opposite of the difference of the sizes of their basins [69]:

$$\sum_z d_{zx} - \sum_z d_{zy} = \left|B_{yx}\right| - \left|B_{xy}\right|.$$

Given the notion of basin, we can define the *basin dominance* property, which requires that the increase in score of $x$ and $y$ is at least as large as (or larger than, in the strict case) the increase in score of all other vertices in their corresponding basin:

**Definition 1.13** (Basin dominance)**.** A centrality $c$ is said to be *basin dominant* on an undirected graph $G$ if for every pair of distinct, non-adjacent vertices $x$ and $y$ we have that

$$\begin{aligned} c'(u) - c(u) &\leq c'(x) - c(x) \qquad \text{for every } u \in B_{xy} \setminus \{x\} \text{ and} \\ c'(v) - c(v) &\leq c'(y) - c(y) \qquad \text{for every } v \in B_{yx} \setminus \{y\}. \end{aligned}$$

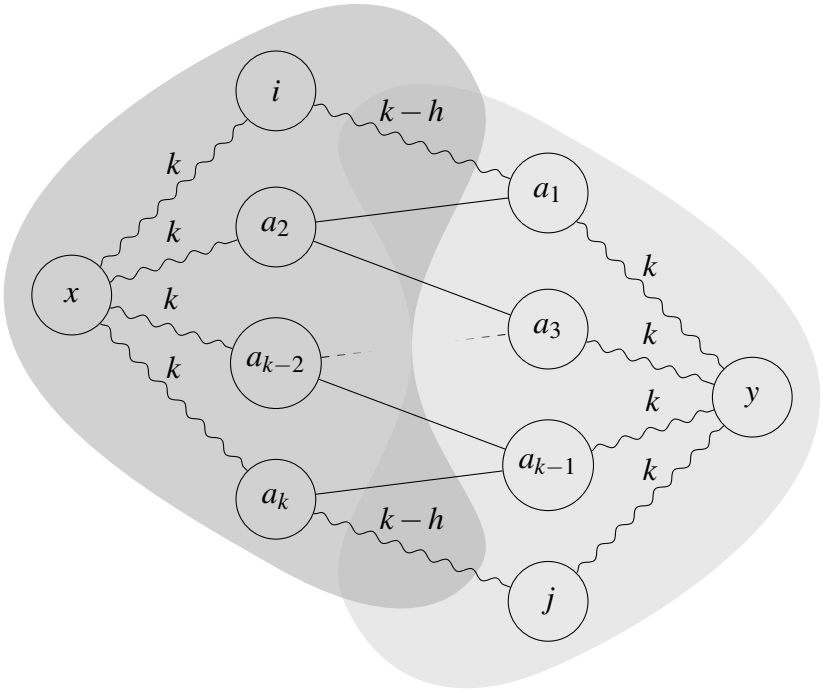


Fig. 1.2 A family of graphs where the shortest path between $i$ and $j$ can change basin $k+1$ times, for $k \geq 2$. $B_{xy}$ is shown in dark grey and $B_{yx}$ is shown in light grey. Each squiggly line with label $\ell$ represents a path of length $\ell$ between its two endpoints. Tuning the parameter $h$ we can make the shortest path between $i$ and $j$ change basin $k+1$ times even after the addition of the edge $x$ — $y$. For example, choosing $h = k-1$ we get $d_{ij} = d'_{ij} = k+1$ (where $d'_{uv}$ is the distance between $u$ and $v$ *after* the addition of the edge $x$ — $y$); moreover, $d_{ix} + d_{xj} = d_{iy} + d_{yj} = d'_{ix} + 1 + d'_{yj} = 2k+1$. On the other hand, choosing $h = 0$, the shortest path between $i$ and $j$ after the addition of the edge $x$ — $y$ will pass through $x$ and $y$.

where $c'$ is the value of the centrality on the graph $G + x — y$. It is *strictly basin dominant* if both inequalities are strict.

At first glance, this property might seem weaker than $\delta$-semi-monotonicity, as it requires a larger increase only for a subset of the vertices. However, it is immediate to notice that either $x$ or $y$ has the largest increase, and with respect to *that* vertex $c$ will be $\delta$-semi-monotone. Combining this observation with the results of the previous section we have that:

**Theorem 1.14.** *If a centrality measure is (strictly) basin dominant on a graph then it is (strictly) $\delta$-semi-monotone. Hence, if a centrality measure is (strictly) basin dominant on a graph then it is (strictly) rank semi-monotone on the same graph.*

As it will become apparent in the following section, basins will enable us to reason about the score increase of a vertex in a more geometric way.

## 1.6 Distance–Decay Centralities

**Conventions.** In this section, and in the following ones, we will often be in a situation where $x$ and $y$ are distinct, non-adjacent vertices of an undirected graph $G$: we will then uniformly use $c$ for the value of a centrality on $G$, and $c'$ for the value of the same centrality on the graph $G' = G + x — y$. Analogously, we will denote distances in $G$ with $d_{uv}$ and in $G'$ with $d'_{uv}$. More in general, we will use the *prime* symbol on quantities and functions related to $G'$.

Our first goal, now, is to prove that a large class of centrality measures are basin dominant, and thus $\delta$-semi-monotone (hence, rank semi-monotone). In fact, we will be able to characterize basin dominance in terms of a discrete convexity condition. As a bonus, we will obtain indirectly that closeness centrality is rank semi-monotone.[3]

A *geometric centrality* [54] is a centrality measure that depends only on distances between vertices. A special class of geometric centralities is the following:

**Definition 1.15** (Distance-decay centrality)**.** A *distance-decay centrality* is a centrality measure $c$ for which there exists a nonincreasing *decay function* $\alpha : \mathbf{N} \setminus \{0\} \to \mathbf{R}$

[3] Some of these results were previously and independently obtained in [70].

such that[4]

$$c(v) = \sum_{\substack{u \neq v \\ d_{uv} < \infty}} \alpha(d_{uv}).$$

The idea that the influence of a vertex on the centrality of another is computed additively on some nonincreasing function of its distance is very natural and it appeared several times in the literature ([71–76], just to cite a few), but there is no standard definition. We took the name and definition used from [72], where it is declined as "distance-decay closeness", but given that closeness does not satisfy the definition (see Section 1.6.1), we prefer to use "distance-decay centrality".[5]

To state our results, we need to recall the definition of the *discrete derivative* operator $\Delta$ [77], which given a function $f : \mathbf{N} \to \mathbf{R}$ returns the function $\mathbf{N} \to \mathbf{R}$ defined by

$$(\Delta f)(i) = f(i+1) - f(i).$$

The operator can be iterated, and we denote with $\Delta^k$ the $k$-th iteration of the operator. As in the case of the standard derivative, $\Delta f \leq 0$ if and only if $f$ is nonincreasing[6].

While the operator is normally defined for functions with domain $\mathbf{N}$, we will use it for functions with domain $\mathbf{N} \setminus \{0\}$ to avoid shifting the input of decay functions. For instance, the condition on the decay function is just that $\Delta\alpha \leq 0$. Observe that when adding an edge $x$ — $y$ all distances decrease, so no score can decrease. Thus, the condition of the derivative of $\alpha$ being nonnegative is fairly natural; otherwise, adding a new edge might decrease the score of some vertex—a quite counterintuitive behavior.

By the same considerations, if we further require that $(\Delta\alpha)(1) < 0$ we have the following result:

**Theorem 1.16.** *A distance-decay centrality measure $c$ is score monotone on connected undirected graphs iff the first derivative of its decay function $\alpha$ is negative at 1, that is, iff $\alpha$ satisfies $(\Delta\alpha)(1) < 0$.*

[4] In the directed case we have a *positive* and a *negative* (distance-decay) centrality, the latter being usually that of interest, depending on whether we use $d_{vu}$ or $d_{uv}$ in the definition.

[5] The reader should not confuse the term *distance-decay centrality* with *decay centrality* [74, 73], a measure that became popular in the economic literature. Note that, in Section 1.6.3, we will identify *decay centrality* as a special case of distance-decay centrality exhibiting *exponential decay*.

[6] For a function $f : \mathbf{N} \setminus \{0\} \to \mathbf{R}$, we write $f \leq 0$ ($f \geq 0$, respectively) if $f(x) \leq 0$ ($f(x) \geq 0$, respectively) holds for every $x \in \mathbf{N} \setminus \{0\}$.

*Proof.* For the right-to-left implication, we have $\alpha(1) > \alpha(2) \geq \alpha(k)$ for all $k \geq 2$. Thus, when adding the edge $x$ — $y$ to the graph, for all $z \neq x, y$ we have $\alpha\big(d'_{xz}\big) \geq \alpha(d_{xz})$, and $\alpha\big(d'_{xy}\big) > \alpha(d_{xy})$. For the reverse, assume by contradiction $\alpha(1) = \alpha(2)$ and consider the graph $x$ — $u$ — $y$: when adding an edge $x$ — $y$ we have $c'(x) = 2\alpha(1) = \alpha(1) + \alpha(2) = c(x)$. □

The reader could be puzzled by the fact that a condition at a single point is equivalent to score monotonicity. However, the gap between $\alpha(1)$ and $\alpha(2)$ has a special role because when adding an edge $x$ — $y$ there is exactly one distance that turns into 1, that is, $d_{xy}$. Since in the graph $x$ — $u$ — $y$ that distance is 2, the gap between $\alpha(1)$ and $\alpha(2)$ must be larger than zero for score monotonicity to happen. On the other hand, since all other coefficients are smaller than or equal to $\alpha(2)$ by definition, the nonzero gap between $\alpha(1)$ and $\alpha(2)$ induces a nonzero gap between $\alpha(1)$ and all other coefficients, which makes the condition sufficient. We will observe a similar phenomenon with strictness in basin dominance.

Interestingly, the *second* derivative gives us further insights into the inner workings of the centrality. As in the continuous case, a *convex* function $f$ is a function such that $\Delta^2 f \geq 0$. Convexity gives us information about what happens when a distance changes because of the addition of an edge.

Indeed, since $(\Delta^2 f)(i) = f(i+2) - 2f(i+1) + f(i)$, we have that $\Delta^2 f \geq 0$ if and only if

$$f(i) - f(i+1) \geq f(i+1) - f(i+2) \quad \text{for every } i, \tag{1.1}$$

that is, shortening a distance from $i+2$ to $i+1$ provides an increase in score that is not larger than the one given by a distance shortening from $i+1$ to $i$. We can generalize this fact by making the shortened distances more far apart, and the gap in the shortenings different:

**Lemma 1.17.** *If $f : \mathbf{N} \setminus \{0\} \to \mathbf{R}$ is convex, then for every $i \leq j$, $k \geq 0$, and $0 \leq \ell \leq j - i$ we have*

$$f(i+k) - f(j+k-\ell) \leq f(i) - f(j).$$

*Moreover, if $i < j$, $k > 0$, and $\big(\Delta^2 f\big)(i) > 0$ (i.e., $f$ is* strictly convex *at $i$), then the inequality is strict.*

*Proof.* By telescoping,

$$\begin{aligned} f(i)-f(j) &= \\ &= f(i)-f(i+1)+f(i+1)-f(i+2)+f(i+2)-\cdots-f(j-1)+f(j-1)-f(j) \\ &\geq f(i+k)-f(i+k+1)+f(i+k+1)-f(i+k+2)-\cdots-f(j+k) \\ &= f(i+k)-f(j+k) \geq f(i+k)-f(j+k-\ell), \end{aligned}$$

where the second inequality is strict if $f$ is strictly convex at $i$, as

$$f(i)-f(i+1) > f(i+1)-f(i+2) \geq \cdots \geq f(i+k)-f(i+k-1). \quad \square$$

$\square$

We remark that if $\alpha$ is strictly convex at 1, that is, $\left(\Delta^2\alpha\right)(1) > 0$, we have that $\alpha(1)-\alpha(2) > \alpha(2)-\alpha(3) \geq 0$ by definition, so $(\Delta\alpha)(1) = \alpha(2)-\alpha(1) < 0$; we just proved that

**Theorem 1.18.** *If the decay function $\alpha$ of a distance-decay centrality is strictly convex at 1, that is, $\left(\Delta^2\alpha\right)(1) > 0$, then $c$ is score monotone on connected undirected graphs.*

This is not all: convexity will give us results about basin dominance, and thus about rank monotonicity. This happens because Lemma 1.17, when read in the context of a decay function, tells us that when adding an arc from $x$ to $y$, if some distance shortens from $j$ to $i$, then it provides an increase in score that is at least the one given by a distance shortening from $j+k$ to $j+k-\ell$, and in fact more in the strict case. Thus, $x$ will benefit of a new edge $x$ — $y$ more than any other vertex in the basin of $x$ with respect to $y$, and analogously for $y$.

Using this observation, we can prove the following result:

**Theorem 1.19.** *If the decay function $\alpha$ of a distance-decay centrality $c$ is convex, then $c$ is basin dominant on connected undirected graphs. Thus, it is $\delta$-semi-monotone and rank semi-monotone. If it is furthermore strictly convex at 1, that is, $\left(\Delta^2\alpha\right)(1) > 0$, then $c$ is strictly basin dominant. Thus, it is strictly $\delta$-semi-monotone and strictly rank semi-monotone.*

*Proof.* Let $c$ be a distance-decay centrality, and let $x$ and $y$ be two distinct non-adjacent vertices. We have to prove that $c'(u)-c(u) \leq c'(x)-c(x)$ for every vertex

$u \in B_{xy}$ (or $<$, if $\big(\Delta^2\alpha\big)(1) > 0$); we have that

$$c'(u) - c(u) = \sum_{z \neq u} \alpha\big(d'_{uz}\big) - \sum_{z \neq u} \alpha(d_{uz}) = \sum_{z \neq u} \Big(\alpha\big(d'_{uz}\big) - \alpha(d_{uz})\Big).$$

We now consider each term of the summation, assuming without loss of generality that $c'(u) - c(u) > 0$ (the case $c'(u) = c(u)$ yields obviously the result we want, using Theorem 1.18 in the strict case).

If $d'_{uz} = d_{uz}$ then $\alpha\big(d'_{uz}\big) - \alpha(d_{uz}) = 0 \leq \alpha\big(d'_{xz}\big) - \alpha(d_{xz})$ (because adding a new edge cannot make the distance increase, and $\alpha$ is nonincreasing).

If instead $d'_{uz} < d_{uz}$, then $d'_{uz} = d'_{ux} + d'_{xz} > d'_{xz}$. Moreover, $d_{uz} \leq d_{ux} + d_{xz}$ and $d_{ux} = d'_{ux}$, so subtracting $d_{uz} \leq d_{ux} + d_{xz}$ from $d'_{uz} = d'_{ux} + d'_{xz}$ we obtain

$$d'_{uz} - d_{uz} \geq d'_{xz} - d_{xz}.$$

We now note that by the convexity of $\alpha$, we can apply Lemma 1.17 with $i = d'_{xz}$, $j = d_{xz}$, $k = d'_{uz} - i$ and $\ell = j + k - d_{uz}$, obtaining

$$\alpha\big(d'_{uz}\big) - \alpha(d_{uz}) \leq \alpha\big(d'_{xz}\big) - \alpha(d_{xz}),$$

where the inequality is strict if $\big(\Delta^2\alpha\big)(1) > 0$ and $z = y$.

Hence, by combining the two cases we have

$$\begin{aligned} c'(u) - c(u) &= \sum_{z \neq u,x} \Big(\alpha\big(d'_{uz}\big) - \alpha(d_{uz})\Big) + \alpha\big(d'_{ux}\big) - \alpha(d_{ux}) \\ &\leq \sum_{z \neq x,u} \Big(\alpha\big(d'_{xz}\big) - \alpha(d_{xz})\Big) + \alpha\big(d'_{xu}\big) - \alpha(d_{xu}) \\ &= c'(x) - c(x), \end{aligned}$$

where, once again, inequality is strict if $\big(\Delta^2\alpha\big)(1) > 0$, as it is strict when $z = y$. □

We can also prove the converse of Theorem 1.19, showing that basin dominance and convexity conditions are tightly coupled:

**Theorem 1.20.** *If a distance-decay centrality $c$ is basin dominant, then its decay function is convex. If it is strictly basin dominant, its decay function is further strictly convex at* 1*, that is,* $\big(\Delta^2\alpha\big)(1) > 0$.

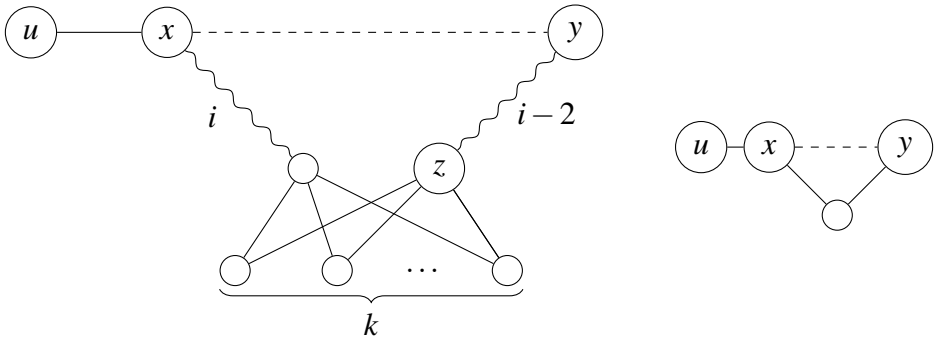


Fig. 1.3 Graphs used in the proof of Theorem 1.20: the one on the right is used for the case $i = 1$, whereas the left one covers $i \geq 2$. Note that, when $i = 2$, the vertices labeled with $y$ and $z$ on the left graph collapse to the same vertex. Each squiggly line with label $\ell$ represents a path of length $\ell$ between its two endpoints.

*Proof.* By contradiction, assume that for some $i \geq 1$ we have $\alpha(i) - \alpha(i+1) < \alpha(i+1) - \alpha(i+2)$. For the case $i \geq 2$, consider the graph on the left in Figure 1.3. We have that after the addition of the edge $x$ — $y$, the variation of the centrality of $x$ is of the form $t_x + k \cdot (\alpha(i) - \alpha(i+1))$, whereas the variation of the centrality of $u$ is of the form $t_u + k \cdot (\alpha(i+1) - \alpha(i+2))$. For $k$ sufficiently large, the latter is greater than the former.

For the case $i = 1$, assume again by contradiction $\alpha(1) - \alpha(2) < \alpha(2) - \alpha(3)$ and consider the graph on the right in Figure 1.3. We have

$$c(x) = 2\alpha(1) + \alpha(2) \qquad c'(x) = 3\alpha(1)$$
$$c(u) = \alpha(1) + \alpha(2) + \alpha(3) \qquad c'(u) = \alpha(1) + 2\alpha(2)$$

so $c'(x) - c(x) = \alpha(1) - \alpha(2) < \alpha(2) - \alpha(3) = c'(u) - c(u)$.

The last statement is easily proved using the same argument we just used for $i = 1$, but assuming by contradiction $\alpha(1) - \alpha(2) = \alpha(2) - \alpha(3)$ and concluding $c'(x) - c(x) = c'(u) - c(u)$ as a consequence. □

In the next few sections, we will apply our characterization of basin dominance to a few geometric centrality measures.

### 1.6.1 Closeness Centrality

Closeness centrality [45, 46] is one of the oldest centrality measures in the literature. It was shown to be score monotone but not rank monotone on connected undirected networks [63]. Moreover, in the latter work it was left as an open problem whether closeness was (in our terminology) rank semi-monotone or not. In the rest of this section, we will solve this open problem by showing that closeness is in fact rank semi-monotone, but not in a strict way.

Recall that the *peripherality* of a vertex $x$ is the sum of the distances between $v$ and all the other vertices of $G$:

$$p(v) = \sum_u d_{uv}.$$

As usual, in the directed case there is a positive and a negative peripherality, but we will be concerned with the connected, undirected case only, for which the two notions coincide. The *closeness centrality* of $v$ is just the reciprocal of its peripherality:

$$C(v) = \frac{1}{p(v)}.$$

To prove rank semi-monotonicity for closeness, we will have to pass through the centrality defined by negated peripherality. This is because

$$-p(v) < -p(u) \iff C(v) < C(u),$$

so negated peripherality is (strictly) semi-rank monotone if and only if closeness is.

**Lemma 1.21.** *Negated peripherality is basin dominant on connected undirected graphs. Thus, it is $\delta$-semi-monotone and rank semi-monotone on the same graphs.*

*Proof.* By Theorem 1.19, since negated peripherality is a distance-decay centrality with decay function $\alpha(i) = -i$, which satisfies $\Delta^2\alpha = 0$. □

We thus obtain

**Theorem 1.22.** *Closeness centrality is rank semi-monotone on connected undirected graphs.*

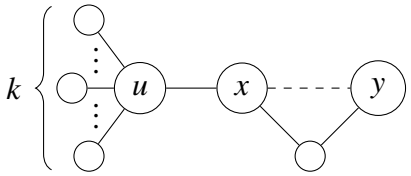


Fig. 1.4 A graph showing that closeness centrality is not basin dominant. For all $k \geq 3$, the closeness centrality of $u$ increases more than that of $x$ when we add the edge $x$ — $y$.

Note that, maybe counterintuitively, closeness centrality is *not* basin dominant. An erroneous statement to this effect appears in [78, Lemma 1], but the counterexample in Figure 1.4 shows that this is not the case, as

$$C'(u) - C(u) = \frac{1}{k+5} - \frac{1}{k+6} > \frac{1}{2k+3} - \frac{1}{2k+4} = C'(x) - C(x)$$

for all $k \geq 3$. The main statement of [78, Theorem 3], however, is correct: closeness is rank semi-monotone, as we have just stated.

The lack of basin dominance is part of the counterintuitive behavior of closeness: it is not strictly rank monotone, not even on directed graphs [63]; it is not score monotone, either, on directed graphs unless the graph is strongly connected [54].

The problem lies in the reciprocation used to make closeness increase when peripherality decreases: because of reciprocation, when we add an edge $x$ — $y$ the distances that are shortened by a certain amount $d$ have the same influence on peripherality, but the effect on scores will depend on the original score of the vertex. In the example, $u$ has a greater centrality than $x$, and thus benefits more than $x$ by the reduction of 1 of peripherality.

Harmonic centrality (see the next section) solves this problem by reciprocating distances instead of the whole sum.

We conclude this section by showing that:

**Theorem 1.23.** *Closeness centrality is not strictly rank semi-monotone on (an infinite family of) connected undirected graphs.*

*Proof.* Consider the graphs in Figure 1.5, where $u \in B_{xy}$. This is an infinite family of graphs with a parameter $k$ which controls the sizes of the two stars around vertices $w$ and $y$. Computing the peripheralities of $u$, $x$ and $y$ before and after the addition of

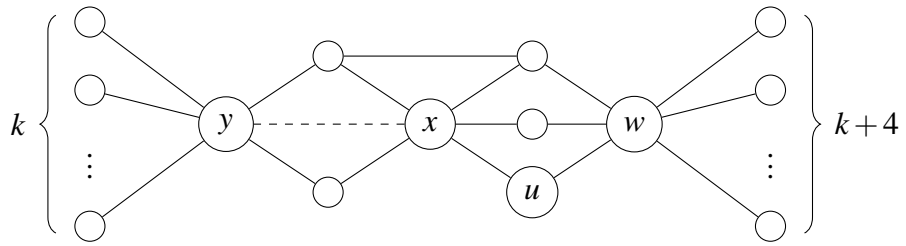


Fig. 1.5 A counterexample to strict rank semi-monotonicity for closeness centrality. For all $k \geq 10$, $u$ and $x$ have the same score before and after the addition of the edge $x$ — $y$. Moreover, $u$ has the same score of $y$ (or smaller) before the addition, but a higher score after the addition, breaking strict rank semi-monotonicity.

$x$ — $y$, we obtain

$$\begin{aligned} p(u) &= 2\cdot(k+4)+4\cdot k+13 & p'(u) &= 2\cdot(k+4)+3\cdot k+12 \\ p(x) &= 3\cdot(k+4)+3\cdot k+9 & p'(x) &= 3\cdot(k+4)+2\cdot k+8 \\ p(y) &= 4\cdot(k+4)+k+15 & p'(y) &= 4\cdot(k+4)+k+12. \end{aligned}$$

For all $k \geq 10$, we have that

$$p(x) = p(u), \quad p'(x) = p'(u), \quad p(y) \leq p(u), \quad p'(y) > p'(u),$$

showing that closeness is not semi-monotone at $y$ (because $y$ used to be at least as central as $u$, but it is less central after the addition of the edge) and not strictly rank semi-monotone at $x$ (it is always as central as $x$, before and after adding the edge). □

Note how the counterexample shows a graph where $y$ has a smaller basin than $x$ but a greater score, losing rank after the addition of the edge (see the last fact we reported about basins in Section 1.5). Moreover, note that the degree of $y$ can become arbitrarily large, while the degree of $x$ and $u$ are fixed. Nevertheless, $x$ and $u$ have a greater score than $y$ in $G'$ for $k \geq 10$, notwithstanding its degree.

We will see that harmonic centrality behaves very differently for the graph in Figure 1.5, yet showing some counterintuitive results.

### 1.6.2 Harmonic Centrality

Harmonic centrality [71] solves the issue of unreachable vertices in closeness centrality. In particular, if we assume $\infty^{-1} = 0$, we can define it as

$$h(v) = \sum_{u \neq v} \frac{1}{d_{uv}},$$

so that unreachable vertices have no impact on the summation and, thus, on the final centrality score of the node. On undirected graphs, it is score monotone but not rank monotone, as shown in [63], where the same counterexample disproving rank monotonicity for closeness centrality also shows that harmonic centrality fails at satisfying this axiom. We can however leverage our results on distance-decay centralities to prove strict basin dominance:

**Theorem 1.24.** *Harmonic centrality is strictly basin dominant on connected undirected graphs. Thus, it is strictly $\delta$-semi-monotone and strictly rank semi-monotone on the same graphs.*

*Proof.* By Theorem 1.19, as harmonic centrality is a distance-decay centrality with decay function $\alpha(i) = 1/i$, which is strictly convex, as $1/i - 1/(i+1) > 1/(i+1) - 1/(i+2)$. The rest follows by Theorem 1.14. □

The stronger result we can give for harmonic centrality should be compared to the fact that on strongly connected graphs harmonic centrality is strictly rank monotone, whereas closeness centrality is just rank monotone [63]. Following the example in Figure 1.5, we can show different behaviors for harmonic centrality as $k$ varies. Computing the harmonic centrality of $u$, $x$ and $y$ before and after the addition of $x$ — $y$ as a function of $k$, we have:

$$\begin{aligned} h(u) &= \frac{1}{2} \cdot (k+4) + \frac{1}{4} \cdot k + \frac{13}{3} & h'(u) &= \frac{1}{2} \cdot (k+4) + \frac{1}{3} \cdot k + \frac{9}{2} \\ h(x) &= \frac{1}{3} \cdot (k+4) + \frac{1}{3} \cdot k + 6 & h'(x) &= \frac{1}{3} \cdot (k+4) + \frac{1}{2} \cdot k + \frac{13}{2} \\ h(y) &= \frac{1}{4} \cdot (k+4) + k + 4 & h'(y) &= \frac{1}{4} \cdot (k+4) + k + \frac{29}{6}. \end{aligned}$$

Note that for such a graph, the increase in score of $y$ is a constant, while the increase in score of $x$ and $u$ is a function of $k$. For $k \geq 2$, we have that $h'(y) - h(y) \leq h'(x) - h(x)$.

In particular, for $k = 4$, $h(y) = h(x)$ hence $h'(y) < h'(x)$, even if the size of the neighborhood of $y$ is larger than that of $x$. For $k \geq 5$, $h'(y) > h'(x)$ even if $y$ increase its score less than $x$. Finally, for $k \geq 8$, $h'(y) > h'(u)$. This example shows how the behavior of harmonic centrality can be quite tricky to predict.

### 1.6.3 Further Distance–Decay Centralities

As we already mentioned, other decay functions have been considered, for example *exponential decay*, where $\alpha(i) = \xi^i$ for some $0 < \xi < 1$ [74, 73]. One can also consider *power-law decay*, where $\alpha(i) = 1/i^k$ for some $k > 0$ (note that the case $k = 1$ is harmonic centrality). Armed with our results, we can easily prove the following theorem:

**Theorem 1.25.** *Distance-decay centralities with exponential decay or power-law decay are score monotone, strictly basin dominant, strictly $\delta$-semi-monotone, and strictly rank semi-monotone.*

*Proof.* By Theorem 1.18 and Theorem 1.19, as

$$\xi^i - \xi^{i+1} > \xi\left(\xi^i - \xi^{i+1}\right) = \xi^{i+1} - \xi^{i+2}$$

and

$$\begin{aligned}\frac{1}{i^k} - \frac{1}{(i+1)^k} = \frac{(i+1)^k - i^k}{i^k(i+1)^k} &= \left(\left(\frac{i+1}{i}\right)^k - 1\right)\frac{1}{(i+1)^k} \\ &> \left(\left(\frac{i+2}{i+1}\right)^k - 1\right)\frac{1}{(i+2)^k} = \frac{(i+2)^k - (i+1)^k}{(i+1)^k(i+2)^k} = \frac{1}{(i+1)^k} - \frac{1}{(i+2)^k},\end{aligned}$$

so in both cases the decay function is strictly convex everywhere. □

## 1.7 Betweenness Centrality

Betweenness centrality [79, 80] tries to capture the idea that a vertex is central if it lies on many shortest paths between other vertices. It does not focus on the length of shortest paths but on how many of them involve a given node, trying to estimate the

amount of flow passing through nodes in a network. For this reason, betweenness is not a geometric measure.

Formally, if we call $\sigma_{ij}$ the number of shortest paths between two vertices $i$ and $j$ and $\sigma_{ij}(v)$ the number of such paths passing through a vertex $v$, then we can define the betweenness centrality of $v$ as

$$b(v) = \sum_{\substack{i,j \neq v \\ \sigma_{ij} > 0}} \frac{\sigma_{ij}(v)}{\sigma_{ij}}.$$

A few variants of this definition appear in the literature:

- Contrarily to Anthonisse's original definition [79], which was stated on directed graphs in the form above, Freeman's later definition [80] was stated on undirected graphs using *unordered* pairs, and indeed his summations have the condition $i < j$. The form above, however, is the one usually found in current literature. With respect to Freeman's definition, however, the value is doubled; this difference is irrelevant as far as score and rank monotonicity are concerned.
- Some authors exclude explicitly the case $i = j$, others do not. Since $\sigma_{ii} = 1$ and $\sigma_{ii}(v) = 0$ for all $v \neq i$, the resulting value does not change, and we will thus assume $i \neq j$ in our proofs.
- Some authors give the definition without the condition $\sigma_{ij} > 0$, in which case the definition applies only to (strongly) connected graphs. In the undirected case, however, the resulting definition is equivalent to defining the centrality on each connected component separately, so we will consider the connected case only, and drop the condition $\sigma_{ij} > 0$ in the proofs.

As in the previous sections, we denote with $\sigma$ and $\sigma'$ the number of shortest paths before and after the addition of an edge $x$ — $y$, and with $b$ and $b'$ the betweenness centrality before and after the addition of the edge.

We know from [63] that betweenness is neither rank nor score monotone. Nonetheless, we can show that the betweenness of two vertices can never decrease after we link them with a new edge: this result was also proved in [81, Theorem 5.2], although with a slightly weaker statement; we thus provide a full proof here for the sake of completeness:

**Lemma 1.26.** *The following property holds when adding the edge $x$ — $y$:*

$$\frac{\sigma'_{ij}(x)}{\sigma'_{ij}} - \frac{\sigma_{ij}(x)}{\sigma_{ij}} \geq 0 \qquad \text{for all } i, j \neq x.$$

*As a consequence, $b'(x) \geq b(x)$. The same statements are true for $y$.*

*Proof.* Given vertices $i, j \neq x$, let us call $p_x$ ($p_{\bar{x}}$, respectively) the number of shortest paths between $i$ and $j$ passing (not passing, resp.) through $x$ in $G$. As usual, let us refer to the same quantities in $G'$ with $p'_x$ ($p'_{\bar{x}}$, resp.). We have to show that the following holds:

$$\frac{\sigma'_{ij}(x)}{\sigma'_{ij}} - \frac{\sigma_{ij}(x)}{\sigma_{ij}} = \frac{p'_x}{p'_x + p'_{\bar{x}}} - \frac{p_x}{p_x + p_{\bar{x}}} \geq 0.$$

Summing over all $i, j \neq x$ proves the second part of the statement.

We consider two cases:

- if $d'_{ij} < d_{ij}$, all shortest paths in $G'$ between $i$ and $j$ pass through the edge $x$ — $y$ (hence, through $x$). Thus, we obtain:

  $$1 - \frac{p_x}{p_x + p_{\bar{x}}} \geq 0,$$

  which is clearly true.

- if $d'_{ij} = d_{ij}$, then all shortest paths between $i$ and $j$ in $G$ are still shortest paths in $G'$, but there may be some new ones passing through $x$, so $p'_{\bar{x}} = p_{\bar{x}}$ and $p'_x \geq p_x$. Letting $\alpha = p'_x - p_x \geq 0$, we obtain:

  $$\frac{p_x + \alpha}{p_x + \alpha + p_{\bar{x}}} - \frac{p_x}{p_x + p_{\bar{x}}} = \frac{p_x^2 + p_x p_{\bar{x}} + \alpha p_x + \alpha p_{\bar{x}} - (p_x^2 + \alpha p_x + p_x p_{\bar{x}})}{(p_x + \alpha + p_{\bar{x}})(p_x + p_{\bar{x}})} = \\ = \frac{\alpha p_{\bar{x}}}{(p_x + \alpha + p_{\bar{x}})(p_x + p_{\bar{x}})} \geq 0,$$

  which is again true, concluding the proof. □

□

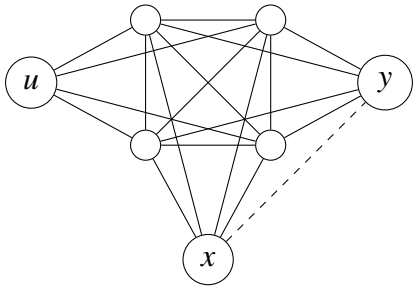


Fig. 1.6 Simple counterexample for score semi-monotonicity and strict rank semi-monotonicity for betweenness centrality. The dashed edge is the $x$ — $y$ edge that we add to $G$, obtaining $G'$. The betweenness score of vertices $x$, $y$ and $u$ is 0 both in $G$ and $G'$. This is true regardless of the size $k$ of the clique (in the picture, $k = 4$).

Incidentally, observe that we can always tell if the betweenness centrality of a vertex is zero without actually computing it. In fact, denoting with $N_G(v)$ the *neighborhood* of $v$ in $G$, that is, the set of vertices $z$ such that $v$ — $z$, we have:

**Lemma 1.27.** *Let $G$ be a connected undirected graph and $v$ a vertex of $G$. Then $b(v) = 0$ iff $v$ is simplicial, that is, iff the subgraph induced by $N_G(v)$ is a clique.*

*Proof.* If $b(v) = 0$ no shortest paths are passing through $v$: but then any two neighbors of $v$ must be adjacent, or otherwise they would have distance 2, and there would be a path through $v$ of length 2. Conversely, suppose that $v$ is simplicial and consider vertices $i, j \neq v$. A shortest path between $i$ and $j$ cannot involve $v$, otherwise it would touch two neighbors of $v$, say $i'$, $j'$, and we might shorten it by skipping $v$. □

An immediate consequence is the following result:

**Theorem 1.28.** *Betweenness centrality is neither score semi-monotone, nor strictly rank semi-monotone on (an infinite family of) connected undirected graphs.*

*Proof.* Consider the family of graphs shown in Figure 1.6, in which we have a clique connected to three vertices $u$, $x$ and $y$ (the unnamed nodes form a clique of arbitrary size $k$, in the picture we show the case $k = 4$). Since they all have the same neighborhood, and that neighborhood is a clique, their betweenness is zero. But when we add the edge $x$ — $y$ this property is still true, so their betweenness remains zero. □

We now prove a sufficient condition for the betweenness of $x$ and $y$ to remain unchanged when adding the edge $x$ — $y$:

**Lemma 1.29.** *Let $x$ and $y$ be two distinct non-adjacent vertices of $G$. If $N_G(x) \setminus \{x\} = N_G(y) \setminus \{y\}$ then $b'(x) = b(x)$ and $b'(y) = b(y)$.*

*Proof.* Assume by contradiction, without loss of generality (see Lemma 1.26), that $b'(x) > b(x)$. Then, there must be at least one pair of distinct vertices $i$, $j$ with a shortest path linking them passing through both $x$ and $y$ via the new edge $x$ — $y$. Thus, $d_{ij} \geq d'_{ij} = d_{iu} + 2 + d_{yj}$ for some neighbor $u$ of $x$. However, at the same time $d_{ij} \leq d_{iu} + 1 + d_{yj}$ because $u$ is also a neighbor of $y$, leading to a contradiction. □

Note that we remove $x$ and $y$ from their neighborhoods to avoid that loops prevent the application of the lemma.

We are now going to show that betweenness centrality is rank semi-monotone on connected undirected graphs. In fact, we show that it even enjoys basin dominance, which is rather surprising given that basin dominance depends on the length of shortest paths, while betweenness depends on the fraction of shortest paths passing through a vertex. Moreover, it is rather hard to find interesting axioms satisfied by betweenness [54].

We start by proving a technical lemma that relates certain products of numbers of shortest paths. There are several cases to consider, but the most interesting one is depicted in Figure 1.7. Consider a situation where $i$ is strictly closer to $x$ than to $y$, and $j$ is strictly closer to $y$ than to $x$. Moreover, fix some (arbitrary) subset of vertices $\alpha$ in the basin of $x$ that we need to go through, and some subset of vertices $\mu$ in the basin of $x$ that we need to avoid. The dotted line represents a (generic) path from[7] $i$ to $j$ satisfying these traversal conditions and passing through $x$, whereas the dashed line represents a (generic) shortest path from $i$ to $j$ passing through $x$.

The key observation is that when we add the edge $x$ — $y$, the only part of shortest paths going from $i$ to $j$ and passing through $x$ that can change is the part from $x$ to $j$ (regardless of whether we are looking at the paths of dotted or dashed type). So the number of dashed/dotted paths in $G$ or $G'$, given that they are all nonzero, is the product of the number of the paths from $i$ to $x$ by the number of the paths from $x$ to $j$, and only the second factor changes when we add the edge $x$ — $y$. Moreover, this second factor is the same for both counts. Thus, if we multiply the counts of the dashed and dotted paths, as long as one count is taken in $G$ and the other is taken in

[7] Note that paths are sequences of vertices, so they have a direction also in an undirected graph. Of course, any path from $i$ to $j$ can be reversed in an undirected graph, obtaining a path from $j$ to $i$.

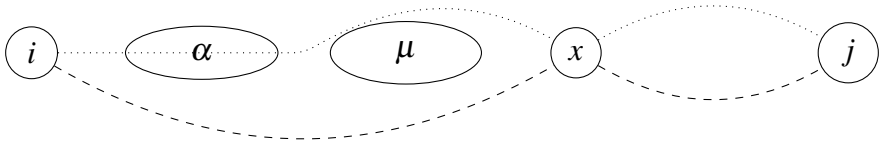


Fig. 1.7 A graph showing the main case of Lemma 1.30.

$G'$ the product is the same—we're just pairing the paths from $i$ to $x$ and from $x$ to $j$ in different ways, or, with a slogan, we can "move the prime":

**Lemma 1.30.** *Let $G$ be a connected undirected graph, $x, y \in V_G$ two non-adjacent vertices and $i, j \in V_G$ two vertices. Let also $\alpha, \mu \subseteq B_{xy} \setminus \{x\}$ be two subsets of the basin of $x$ not containing $x$. Call $p_x$ ($p_{\bar{x}}$, respectively) the number of shortest paths from $i$ to $j$ in $G$ passing through $x$ (not passing through $x$, resp.). Let also $p_{\alpha\bar{\mu}x}$ be the number of paths from $i$ to $j$ passing through all the vertices of $\alpha$ and then through $x$, but never passing through any of the vertices of $\mu$ before reaching $x$, and $p_{\alpha\bar{\mu}\bar{x}}$ the number of paths from $i$ to $j$ passing through all the vertices of $\alpha$ but never passing through $x$ or any of the vertices of $\mu$. Define similarly $p'_x$, $p'_{\bar{x}}$ $p'_{\alpha\bar{\mu}x}$, and $p'_{\alpha\bar{\mu}\bar{x}}$ for the paths satisfying the same conditions, but in $G'$. Then, the following properties hold:*

$$
\begin{aligned}
&(1.) \quad p'_{\alpha\bar{\mu}x} p_x = p_{\alpha\bar{\mu}x} p'_x \\
&(2.) \quad p'_{\alpha\bar{\mu}\bar{x}} p_{\bar{x}} = p_{\alpha\bar{\mu}\bar{x}} p'_{\bar{x}} \\
&(3.) \quad p'_{\alpha\bar{\mu}\bar{x}} p_x \leq p_{\alpha\bar{\mu}\bar{x}} p'_x \\
&(4.) \quad p'_{\alpha\bar{\mu}x} p_{\bar{x}} \geq p_{\alpha\bar{\mu}x} p'_{\bar{x}}
\end{aligned}
$$

*Proof.* We can assume w.l.o.g. that $i \in V_G \setminus B_{yx}$ and $j \in V_G \setminus B_{xy}$, otherwise everything holds trivially because the new edge $x \text{ — } y$ does not create new shortest paths from $i$ to $j$, hence all the counts of shortest paths from $i$ to $j$ in $G$ and in $G'$ are the same.

We now prove separately each of the four statements.

(1.) If $p_x = 0$ then $p_{\alpha\bar{\mu}x} = 0$, so the statement holds, and the same happens for $p'_x = 0$, so we can assume $p_x, p'_x \neq 0$, and in particular $d_{ij} = d_{ix} + d_{xj}$, $d'_{ij} = d'_{ix} + d'_{xj}$, and $d'_{ix} = d_{ix}$, as shortest paths between $i$ and $x$ are not affected by the new edge.

Thus, $p_{\alpha\bar{\mu}x} \neq 0$ iff there is a path in $G$ from $i$ to $x$ of length $d_{ix}$ that passes through $\alpha$ but not through $\mu$, but since $d'_{ix} = d_{ix}$ this is true iff there is a path in $G'$ from $i$ to $x$

of length $d'_{ix}$ that passes through $\alpha$ but not through $\mu$, that is, iff $p'_{\alpha\bar{\mu}x} \neq 0$. Thus, we can also assume $p_{\alpha\bar{\mu}x}, p'_{\alpha\bar{\mu}x} \neq 0$.

Now let $s$ be the number of shortest paths in $G$ from $i$ to $x$ , and $s_{\alpha\bar{\mu}}$ the number of such paths that pass through vertices of $\alpha$ and do not pass through any of the vertices of $\mu$ (see Figure 1.7): these values do not change in $G'$ because the new edge $x$ — $y$ can only shorten the paths from $x$ to $j$. Let also $t$ and $t'$ be the number of shortest paths from $x$ to $j$ in $G$ and $G'$, respectively. Then,

$$p'_{\alpha\bar{\mu}x} p_x = s_{\alpha\bar{\mu}} t' st = s_{\alpha\bar{\mu}} t s t' = p_{\alpha\bar{\mu}x} p'_x.$$

(2.) For the shortest paths not passing through $x$, there are two possibilities:

- If $d'_{ij} = d_{ij}$ then $p'_{\alpha\bar{\mu}\bar{x}} = p_{\alpha\bar{\mu}\bar{x}}$ and $p'_{\bar{x}} = p_{\bar{x}}$, since all the shortest paths in $G$ are also shortest in $G'$, and more are possibly added but they all pass through $x$.
- If $d'_{ij} < d_{ij}$ then $p'_{\alpha\bar{\mu}\bar{x}} = 0 = p'_{\bar{x}}$, because in $G'$ all the shortest paths between $i$ and $j$ must pass through $x$.

In both cases, the equality holds.

(3.) When we add $x$ — $y$, we have two possibilities:

- If $d'_{ij} = d_{ij}$ then $p'_{\alpha\bar{\mu}\bar{x}} = p_{\alpha\bar{\mu}\bar{x}}$ and $p_x \leq p'_x$;
- If $d'_{ij} < d_{ij}$ then $p'_{\alpha\bar{\mu}\bar{x}} = 0$.

In both cases, the inequality holds.

(4.) Multipying (1.) and (2.) we have $p'_{\alpha\bar{\mu}x} p_x p'_{\alpha\bar{\mu}\bar{x}} p_{\bar{x}} = p_{\alpha\bar{\mu}x} p'_x p_{\alpha\bar{\mu}\bar{x}} p'_{\bar{x}}$, and dividing by (3.) we obtain the statement. □ □

We are now ready to prove basin dominance for betweenness centrality: the proof goes through a case-by-case analysis of the contribution of each summand to the difference in centrality, with the main case being covered by Lemma 1.30:

**Theorem 1.31.** *Betweenness centrality is basin dominant on connected undirected graphs. Thus, betweenness centrality is $\delta$-semi-monotone and rank semi-monotone on connected undirected graphs.*

*Proof.* Let us call $\Delta_u = b'(u) - b(u)$ the score difference for a vertex $u$ and for every pair of distinct vertices $i \neq j$, let also

$$\Delta_u(i,j) = \frac{\sigma'_{ij}(u)}{\sigma'_{ij}} - \frac{\sigma_{ij}(u)}{\sigma_{ij}}.$$

Obviously

$$\Delta_u = \sum_{i,j \neq u} \Delta_u(i,j).$$

We want to show that $\Delta_u \leq \Delta_x$ for every $u \in B_{xy}$. For vertices $u$ being equidistant from $x$ and $y$, everything holds trivially because the new edge $x$ — $y$ does not create any shortest path between $i$ and $j$ passing through $u$. Then, we can restrict our attention to the case where $u$ is strictly closer to $x$ than to $y$, that is, $u \in V_G \setminus B_{yx}$.

The two summations giving $\Delta_u$ and $\Delta_x$ happen on a different set of pairs of indices, and we treat the common and non-common pairs separately.

The easiest case is that of pairs $i, j$ that appear in the summation of $\Delta_x$ but not in the summation of $\Delta_u$, because we know from Lemma 1.26 that those summands are non-negative.

Then we consider the pairs $i, j$ that appear in the summation of $\Delta_u$ but not in the summation of $\Delta_x$, that is, those where either $i$ or $j$ are equal to $x$; without loss of generality let us assume $j = x$. We want to show that in this case, instead, we have

$$\Delta_u(i,x) = \frac{\sigma'_{ix}(u)}{\sigma'_{ix}} - \frac{\sigma_{ix}(u)}{\sigma_{ix}} \leq 0.$$

When $i \in B_{xy}$, the new $x$ — $y$ edge does not create any new shortest path between $i$ and $x$, so $\Delta_u(i,x) = 0$. Conversely, when $i \notin B_{xy}$ new shortest paths cannot pass through $u$ (remember that $u \in B_{xy}$); thus,

- if $d_{ix} = d'_{ix}$ then $\sigma_{ix} \leq \sigma'_{ix}$ and $\sigma_{ix}(u) = \sigma'_{ix}(u)$;
- if $d_{ix} > d'_{ix}$ then $\sigma'_{ix}(u) = 0$.

We are now left with the pairs $i, j$ that appear in both summations, that is, $i, j \neq u$ and $i, j \neq x$. In this case, we want to prove a term-by-term bound, that is:

$$\Delta_u(i,j) = \frac{\sigma'_{ij}(u)}{\sigma'_{ij}} - \frac{\sigma_{ij}(u)}{\sigma_{ij}} \leq \frac{\sigma'_{ij}(x)}{\sigma'_{ij}} - \frac{\sigma_{ij}(x)}{\sigma_{ij}} = \Delta_x(i,j). \tag{1.2}$$

Note that if $i$ and $j$ belong to the same basin the edge $x$ — $y$ does not create any new shortest path between $i$ and $j$, so (1.2) holds because both sides are zero; this includes the case in which $i$ and $j$ are in the intersection of the basins, so we can assume, without loss of generality, that $i \in V_G \setminus B_{yx}$ and $j \in V_G \setminus B_{xy}$.

The case $\Delta_u(i,j) \leq 0$ is trivial because of Lemma 1.26; we now analyze the case $\Delta_u(i,j) > 0$.

Since $i, u \in V_G \setminus B_{yx}$ and $j \in V_G \setminus B_{xy}$, we can state two facts:

- all shortest paths in $G$ and $G'$ from $i$ to $j$ passing through $x$ and $y$ must pass through $x$ before $y$;
- all shortest path in $G$ and $G'$ from $i$ to $j$ passing through $u$ and $x$ must pass through $u$ before $x$.

As mentioned in Section 1.5, a shortest path between $i$ and $y$ passes exclusively through vertices in the basin of $x$ and then exclusively through vertices in the basin of $y$, until it reaches $y$. Moreover, a shortest path between $j$ and $y$ passes exclusively through vertices in the basin of $y$. Combining these two facts is enough to show that the first statement holds.

The second one, instead, can be proven by contradiction: if $u$ is after $x$ in a shortest path from $i$ to $j$, then $u$ is necessarily between $x$ and $y$. This means that $d'_{ij} < d_{ij}$ since we might shorten the path from $x$ to $y$ passing through $u$ by taking $x$ — $y$. Hence $\sigma'_{ij}(u) = 0$ and thus $\Delta_u(i,j) < 0$—a contradiction.

We now define a few counters of shortest paths from $i$ to $j$ in $G$ satisfying certain conditions:

- $p_x$: passing through $x$;
- $p_{\bar{x}}$: not passing through $x$;
- $p_{ux}$: passing through $u$ and then through $x$;
- $p_{\bar{u}x}$: passing through $x$ but not through $u$;
- $p_{u\bar{x}}$ passing through $u$ but not through $x$;
- $p_{\bar{u}\bar{x}}$ passing through neither.

The same notations are used for $G'$, but we use $p'$ instead of $p$.

With these notations, we can write the single terms appearing in (1.2) as follows:

$$\begin{aligned} \sigma_{ij}(u) &= p_{ux} + p_{u\bar{x}} & \sigma'_{ij}(u) &= p'_{ux} + p'_{u\bar{x}} & \sigma_{ij} &= p_x + p_{\bar{x}} \\ \sigma_{ij}(x) &= p_{ux} + p_{\bar{u}x} & \sigma'_{ij}(x) &= p'_{ux} + p'_{\bar{u}x} & \sigma'_{ij} &= p'_x + p'_{\bar{x}} \end{aligned}$$

which makes us able to rewrite (1.2) as

$$\frac{p'_{ux} + p'_{u\bar{x}}}{\sigma'_{ij}} - \frac{p_{ux} + p_{u\bar{x}}}{\sigma_{ij}} \leq \frac{p'_{ux} + p'_{\bar{u}x}}{\sigma'_{ij}} - \frac{p_{ux} + p_{\bar{u}x}}{\sigma_{ij}},$$

which is equivalent to

$$\frac{p'_{u\bar{x}} - p'_{\bar{u}x}}{\sigma'_{ij}} \leq \frac{p_{u\bar{x}} - p_{\bar{u}x}}{\sigma_{ij}}.$$

Multiplying both sides by $\sigma'_{ij}\sigma_{ij}$, we obtain

$$p'_{u\bar{x}} \cdot \sigma_{ij} - p'_{\bar{u}x} \cdot \sigma_{ij} \leq p_{u\bar{x}} \cdot \sigma'_{ij} - p_{\bar{u}x} \cdot \sigma'_{ij}.$$

Now, it is enough to show that the two following inequalities hold:

$$\begin{aligned} p'_{u\bar{x}} \cdot (p_x + p_{\bar{x}}) &\leq p_{u\bar{x}} \cdot (p'_x + p'_{\bar{x}}) \\ p'_{\bar{u}x} \cdot (p_x + p_{\bar{x}}) &\geq p_{\bar{u}x} \cdot (p'_x + p'_{\bar{x}}), \end{aligned}$$

for which, in turn, it is sufficient to show that the following four statements hold:

$$\begin{aligned} p'_{u\bar{x}} p_x &\leq p_{u\bar{x}} p'_x & p'_{u\bar{x}} p_{\bar{x}} &= p_{u\bar{x}} p'_{\bar{x}} \\ p'_{\bar{u}x} p_x &= p_{\bar{u}x} p'_x & p'_{\bar{u}x} p_{\bar{x}} &\geq p_{\bar{u}x} p'_{\bar{x}}. \end{aligned}$$

The latter are immediate by Lemma 1.30 if we set appropriately $\alpha$ and $\mu$ to $\varnothing$ or $\{u\}$. □

## 1.8 Conclusions

This chapter answers positively some questions raised in [62, 63, 54] about closeness, harmonic centrality and betweenness. Table 1.1 puts the positive results about rank semi-monotonicity of this chapter in context with the positive results of the directed case. It is interesting to note that rank semi-monotonicity is in fact the only property of this kind that is true for betweenness. All the results are consequences of a stronger result of a similar type about basin dominance.

| | undirected | | directed [62, 54] | |
|---|---|---|---|---|
| | score | rank | score | rank |
| Closeness | monotone [63] | **semi-monotone** | monotone | monotone |
| Harmonic centrality | monotone [63] | **strictly semi-mon.** | monotone | strictly monotone |
| Betweenness | **not semi-monotone** | **semi-monotone** | not monotone | not monotone |

Table 1.1 Summary of the results about semi-monotonicity obtained in this chapter (in boldface) corresponding to negative results about monotonicity in [63]. All results are about (strongly) connected graphs, except for the monotonicity property of harmonic centrality on directed graphs, which is true on all graphs.

We also proved results about basin dominance, score monotonicity, and rank semi-monotonicity for distance-decay centralities whose decay function is convex; this result enabled us to prove rank semi-monotonicity for other centralities defined in the literature. In particular, we characterised score monotonicity in terms of a simple condition on the first discrete derivative of the decay function, and showed that discrete convexity of the decay function is necessary and sufficient for basin dominance. As a consequence, several families of distance-based centralities, such as exponential decay centralities and power-law decay centralities, turn out to be basin dominant, $\delta$-semi-monotone, and hence rank semi-monotone.

Our negative results are about strictness of rank semi-monotonicity, in particular of closeness and betweenness, and lack of score semi-monotonicity for betweenness. For all the negative results we exhibit an infinite family of counterexamples.

The notion of basin dominance proved to be the key idea in all proofs of semi-monotonicity. It provides a geometric way to control how score increments are distributed after the addition of an edge: under basin dominance, both endpoints of the new edge gain more than any other vertex in their respective basins, which in turn implies $\delta$-semi-monotonicity and rank semi-monotonicity. It would be interesting to investigate whether basin dominance applies to other geometric measures, or

even other centrality measures based on shortest paths, as in that case one gets immediately rank semi-monotonicity.

Beyond their combinatorial interest, these results also have implications for the use of centrality in biological network analysis. Centrality measures are routinely used to prioritise genes, proteins, or pathways in protein-protein interaction networks, regulatory networks, metabolic networks, and disease-gene association graphs, where highly central nodes are more likely to be essential, to play key regulatory roles, or to be attractive drug targets [43, 44, 48, 49, 59–61, 58, 82–84]. In this setting, centrality effectively plays the role of an inductive bias: scores determine which nodes are treated as more plausible candidates for follow-up experiments, for inclusion in disease modules, or for in silico drug-repurposing pipelines based on network proximity or diffusion [59, 60, 58, 85]. From this perspective, the axioms studied in this chapter constrain the behaviour we are willing to accept from such priors as the underlying networks evolve. When a new interaction is added, at least one endpoint must improve relative to others, and within each endpoint's basin, no other vertex gains more than that endpoint. These minimal stability conditions are particularly valuable for noisy, incomplete biological networks subject to sampling biases, where centrality rankings are sensitive to missing data [83, 84, 86].

In drug discovery and therapeutic target prioritization, centrality measures rank candidate drug targets in protein-protein interaction networks, where highly central proteins are prioritized for experimental validation [60, 84]. However, these networks are notoriously incomplete: current estimates suggest that experimentally validated protein interactions represent only $\sim$10% of the true interactome [82, 87, 88]. As new high-throughput screens incrementally add edges to these networks, target rankings can shift dramatically [84]. In network-based drug repurposing, methods compute proximity between disease modules and drug targets using centrality-weighted propagation or shortest-path distances [89–91]. When underlying networks are updated, proximity scores can change non-monotonically, potentially invalidating predictions. Our axioms provide theoretical grounding for identifying repurposing methods robust to network updates.

## 1.9 Future Works

Proving or disproving score and (strict) rank semi-monotonicity for other measures remains an open problem. Of particular interest are spectral measures, such as PageRank and other feedback centralities, which were shown not to be rank monotone in undirected graphs [63]. The primary difficulty in analyzing spectral measures stems from their algebraic, rather than combinatorial, definition. Unlike the path-based centralities examined in this chapter, the centrality vector of spectral measures is defined as the principal eigenvector of a matrix derived from the graph's adjacency matrix. When an edge is added to an undirected graph, the adjacency matrix undergoes a rank-two symmetric update. Consequently, determining whether PageRank or Katz index satisfies semi-monotonicity will likely require techniques from matrix perturbation theory, specifically analyzing how the principal eigenvector changes in response to such low-rank updates.

Spectral centralities have demonstrated substantial utility in biological applications. PageRank has been applied to gene regulatory networks for identifying regulatory elements [48, 92], protein-protein interaction networks for disease-gene associations [83], and cancer gene identification through pathway analysis [93]. Establishing semi-monotonicity properties would provide theoretical guarantees about their behavior as biological networks evolve.

**Graph neural networks and biological inductive biases.** Understanding the semi-monotonicity and stability properties of centrality measures has direct implications for the design of graph neural networks (GNNs) applied to biological data. Recent work has revealed connections between graph neural networks and spectral centralities: GNNs can be interpreted as discrete dynamical systems whose behavior relates to spectral properties of the underlying graph [94, 95]. The message-passing operations in GNN architectures conceptually mirror the iterative update schemes that define spectral centralities, with both frameworks capturing how information propagates through network structure. Moreover, spectral centralities correspond naturally to random walk processes on graphs, providing theoretical foundations for understanding information propagation in graph convolutional networks [96]. Recent work has further emphasized these connections by interpreting neural networks themselves as relational graphs, revealing structural relationships between network architecture and graph-theoretic properties [97].

Several models explicitly incorporated personalized PageRank as a propagation scheme [98, 99], separating feature transformation from diffusion to aggregate information from large, adjustable neighborhoods. If centrality measures used in propagation schemes satisfy semi-monotonicity, resulting GNNs could inherit stability guarantees: node representations would change predictably under edge additions, preventing unstable reranking when new interactions are discovered.

Recent GNN-based perturbation prediction models exemplify these considerations. GEARS (Graph-Enhanced gene Activation and Repression Simulator) predicts transcriptional responses to genetic perturbations by learning over Gene Ontology-derived knowledge graphs [100]. PDGrapher predicts combinatorial therapeutic targets by solving an inverse problem: given diseased and treated cell states, it identifies perturbagens to achieve desired phenotypic transitions using protein-protein interaction or gene regulatory networks as causal graph proxies [101]. Both models use GNN message passing over biological networks that are continuously refined: Gene Ontology annotations update quarterly, and pathway databases see hundreds of additions per release. As these knowledge graphs evolve, understanding which graph properties ensure prediction stability becomes crucial. While these models don't explicitly use centrality measures as features, the information propagation mechanisms they employ are governed by the graph's structural properties, making semi-monotonicity results potentially relevant for understanding their robustness. Establishing formal connections between centrality semi-monotonicity and GNN stability could guide architecture design for biological applications. When biological knowledge graphs are incomplete or noisy, architectures leveraging propagation schemes with proven stability properties may yield more reliable predictions for drug discovery and therapeutic target identification.

# Chapter 2

# Compositional Sparsity of Learnable Functions

*The content of this chapter is largely based on the research originally presented in [102].*

Chapter 1 developed the mathematical foundations for understanding how centrality measures behave when networks change, proving that measures like closeness, harmonic centrality, and betweenness satisfy rank semi-monotonicity. However, centrality measures represent just one facet of network analysis, addressing the fundamentally *descriptive* challenge of characterizing the importance of network components.

The biological scientific goals extend far beyond description to *prediction*: Can we predict protein function from network topology? Can we identify disease-causing mutations? Can we design therapeutic interventions? These challenges require learning complex functions that map high-dimensional biological data to meaningful biological outcomes.

Centrality measures are highly sensitive to the sampling biases and missing edges that pervade molecular interaction data, limiting their reliability for biological inference [84]. On the other hand, AI-driven models, which can adapt to new data and capture non-linear relationships across genome-scale networks, offer a viable path from description to prediction [103, 104].

Historically, many attempts to build intelligent systems relied on logical or rule-based approaches [105, 106]. In contrast, the rise of deep neural networks from around 2012 onward ushered in learning-based architectures that surpass traditional algorithms in numerous domains. For several years, the focus has been on pushing performance boundaries, achieving remarkable success across domains ranging from computer vision [107, 108], to playing strategic games (AlphaGo [109], AlphaZero [110]), to natural language processing and complex reasoning [111–117]. In biological applications, deep networks have revolutionized our ability to predict protein folding and interactions [8, 9, 118–120], and promise to revolutionize biology with the help of single-cell omics [104]. Although these models continue to break benchmark after benchmark, their development often outpaces the theory needed to explain or predict their performance.

Like images and text, biological data has features with thousands or even millions of dimensions and classical statistical learning theory, grounded in the framework of empirical risk minimization, provides sample complexity bounds that scale exponentially with the ambient dimension of the input space [121, 122]. For a function $f : \mathscr{X}^d \to \mathscr{Y}$ in $d$ dimensions, standard approximation results suggest that achieving $\varepsilon$-accuracy requires $O((1/\varepsilon)^d)$ parameters or samples—an exponential dependence known as the curse of dimensionality [123, 124].

Yet deep neural networks routinely violate these predictions. Modern architectures with polynomially many parameters achieve excellent generalization performance on high-dimensional tasks where classical theory predicts failure [125, 126]. The resolution, we argue, lies in recognizing that real-world functions are not arbitrary mappings over high-dimensional spaces. Instead, they exhibit *structural regularity* that can be exploited by appropriately designed architectures [1, 127]. The most fundamental form of such regularity is *compositional sparsity* [128–132].

The compositional principle, that complex systems can be understood as hierarchical arrangements of simpler components, represents one of the most pervasive organizing principles in nature [133, 134]. In physics, atoms compose molecules, which compose materials, which exhibit emergent properties at macroscopic scales [135]. In biology, this principle manifests at every level: amino acids fold into protein domains that combine to create functional enzymes [136]; regulatory motifs compose into circuits that govern developmental programs [137]; cellular modules aggregate into tissues and organs with specialized functions [138].

This hierarchical organization is not merely descriptive but reflects fundamental constraints on how complex systems can be efficiently constructed and understood [133]. A protein with 1000 amino acids could, in principle, adopt $20^{1000}$ possible sequences, yet evolution has discovered functional sequences that can be understood through modular domain organization [139]. Similarly, gene regulatory networks with thousands of components exhibit modular structure that enables both robustness and evolvability [140].

Compositional structure also pervades human-designed systems. Software architectures rely on modular design principles [141] and natural language exhibits hierarchical organization from phonemes to morphemes to words to phrases [142]. This ubiquity suggests that compositional organization is not accidental but reflects fundamental constraints on complexity and learnability.

In this chapter, we tie Turing-computability to compositional sparsity. We prove that all efficiently Turing-computable functions exhibit compositional sparsity, providing a possible explanation for why deep learning succeeds where classical methods fail. Our framework offers new theoretical insights into the effectiveness of hierarchical architectures and provides a unifying perspective on phenomena ranging from the success of convolutional networks to the emergence of Chain-of-Thought reasoning in large language models.

We begin by reviewing classical learning theory and the curse of dimensionality in Section 2.1. Section 2.2 then introduces compositional sparsity as the key to understanding why deep learning succeeds, formally defining this property and proving that all efficiently Turing-computable functions exhibit compositional sparsity. We demonstrate how this can resolve the deep learning paradox by showing that deep networks can approximate such functions with polynomial rather than exponential complexity. In Section 2.3 we shift from representation to learning, exploring the theoretical challenges of discovering compositional structure from data and examining how modern architectures like convolutional neural networks and transformers exploit sparsity through architectural biases. We conclude by connecting our framework to contemporary phenomena in large language models, particularly Chain-of-Thought reasoning, which we argue naturally exploits compositional sparsity by decomposing complex problems into learnable subcomponents. Throughout, we highlight open questions that bridge theory and practice, pointing toward future research directions in deep learning theory.

## 2.1 Preliminaries

We briefly introduce statistical learning viewed through the lens of empirical risk minimization following standard literature [143–145]. We also introduce the notion of the *curse of dimensionality*, to which classical learners such as shallow networks are subject.

In a supervised learning problem, an unknown probability measure $\mu$ in the space of input-output pairs $\mathscr{X} \times \mathscr{Y}$ gives rise to the target function $f_\mu : \mathscr{X} \to \mathscr{Y}$ assumed to underlie an observable i.i.d. finite training set $S = \{(x_i, y_i)\}_{i=1}^m$ with $y_i = f_\mu(x_i)$. It is the goal to approximate $f_\mu$ using some $f$ best, as measured by an expected risk over the probability measure $\mu$

$$\mathscr{R}(f) \;=\; \int \ell(f(x), y)\, d\mu(x, y) \tag{2.1}$$

constructed from some per-instance loss $\ell$. Since $\mu$ is commonly unknown, the expected risk is approximated empirically via the observable training set,

$$\mathscr{R}_{\text{emp}}(f) \;=\; \frac{1}{m} \sum_{i=1}^{m} \ell(f(x_i), y_i). \tag{2.2}$$

A low empirical risk does not necessarily imply that the approximation predicts well on unseen data. However, the generalization performance of this empirical risk minimization procedure can be bounded by taking the complexity of the hypothesis class $\mathscr{H}$, over which $f$ is optimized, into account. The Rademacher complexity $\mathscr{R}_m$ of $\mathscr{H}$, defined on the independent Rademacher random variables $\sigma_i$,

$$\mathscr{R}_m(\mathscr{H}) = \mathbb{E}_\sigma \left[ \sup_{f \in \mathscr{H}} \frac{1}{m} \sum_{i=1}^{m} \sigma_i f(x_i) \right] \tag{2.3}$$

probabilistically bounds the gap between the expected risk $\mathscr{R}(f)$ and the empirical risk $\mathscr{R}_{\text{emp}}$

$$\mathscr{R}(f) \leq \mathscr{R}_{\text{emp}}(f) + \mathscr{R}_m(\mathscr{H}_f) + o(\delta) \tag{2.4}$$

where the low-order error terms depend on the certainty of the guarantee $1 - \delta$ and the training set size $m$.

From an approximation-theoretic perspective, the key challenge is that classical learners such as kernel machines or shallow networks are affected by the curse

of dimensionality. As the dimension $d$ of a function $f$ grows, these methods may require an exponentially large number of parameters to approximate $f$ arbitrarily well [129].

Consequently, a central question arises: *Which property of real-world target functions ensures that a suitable deep neural network can approximate them without requiring a number of parameters that grows exponentially with $d$?* The crucial insight is that deep networks can exploit suitable structural assumptions about the target function—most notably, that it admits a decomposition which avoids the need for an exponential number of parameters. As we will demonstrate in the following section, this requirement turns out to be surprisingly mild encompassing a broad class of functions. Thus, while traditional approaches suffer in high dimensions, deep neural networks can circumvent an exponential blow-up in the number of parameters by leveraging such structure [146].

## 2.2 Compositional Sparsity and Deep Learning

A central concept in our argument is that of *efficient Turing-computability*. Throughout this chapter, we use this term to refer to functions in the complexity class **FP**, that is, function problems solvable by a deterministic Turing machine in polynomial time. While the decision problem class **P** involves single-bit yes/no answers, **FP** encompasses functions whose outputs can be computed in polynomial time. For real-valued functions, computability can be formalized in multiple ways (see Poggio and Fraser [132] for a detailed discussion), but for the purposes of this chapter, we equate "efficient Turing-computable functions" with those in **FP**.

Subsequently, we introduce the notion of compositionally sparse functions and their relevance. We then state how this property helps deep neural networks to break the curse of dimensionality. For an alternative set of proofs see [147].

### 2.2.1 Compositionally Sparse Functions

**Definition 2.1** (Compositionally Sparse Function)**.** A function $f : \mathscr{X}^d \to \mathscr{X}$ is *compositionally sparse* if it can be represented as the composition of at most $\mathscr{O}(\operatorname{poly} d)$

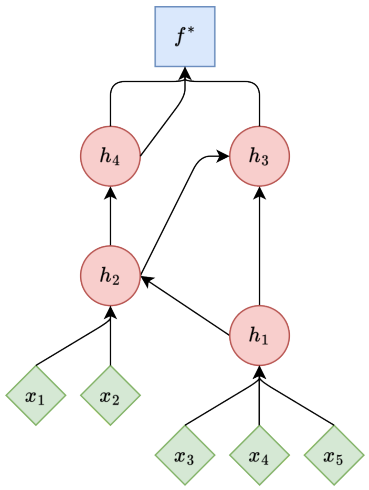


Fig. 2.1 A directed acyclic graph representing a compositionally sparse function. The green diamonds denote the $d = 5$ input variables, the red dots constituent functions and the blue square the final output. Each function depends on at most $3 = c \ll d$ variables.

constituent functions each of which is sparse, i.e., depends on at most a (small) constant number of variables $c$.

A compositionally sparse function can be visually depicted as a directed acyclic graph, in which the leaves represent inputs, the root denotes the output function, and the internal nodes represent the constituent functions. The maximum in-degree of the directed acyclic graph is equivalent to $c$. Figure 2.1 illustrates this definition for a small toy example with total input dimension $d = 5$, and input dimension of at most $c = 3$ for all constituent functions. Functions with constituents of bounded dimensionality were earlier referred to as "hierarchically local compositional functions" [129]. Interestingly, this property is not very restrictive and is satisfied by a large class of functions, as the following theorem states.

**Theorem 2.2** (Efficient Computability implies Compositional Sparsity, restated from Poggio and Fraser [132])**.** *Any function that is efficiently Turing-computable is compositionally sparse.*

We prove this conjecture from Poggio and Fraser [132]. The proof uses the fact that efficiently Turing-computable functions may be translated into Boolean circuits of a polynomial number of gates with a bounded number of input variables.

### 2.2.2 Proof of Theorem 2.2

**Theorem** (Efficient Computability implies Compositional Sparsity)**.** *Any function* $f \in \mathbf{FP}$ *is compositionally sparse.*

*Proof.* Assume a function $f \in \mathbf{FP}$, i.e., $f$ is efficiently Turing-computable. By definition, there exists a deterministic Turing machine that computes $f$ in time $T(n) = \mathcal{O}(\text{poly}(n))$.

A deterministic Turing machine running in time $T(n)$ can be converted into a Boolean circuit with $\mathcal{O}(T(n)\log T(n))$ gates [148]. For $T(n) = \mathcal{O}(\text{poly}\, n)$, this results in a circuit $C$ of size $\mathcal{O}(\text{poly}\, n)$.

The circuit $C$ may include gates with unbounded fan-in. To ensure compositional sparsity, we transform $C$ into an equivalent circuit with 2 inputs.

In particular:

- Any gate with $k > 2$ inputs can be replaced with a binary tree of $\lceil log_2(k) \rceil$-depth and $k-1$ gates (see Figure 2.2 for an example visualization).
- Since $k$ in the original circuit is bounded by $\mathcal{O}(T(n)) = \mathcal{O}(\text{poly}(n))$, transforming $C$ into a circuit with fan-in 2 increases the circuit size by at most a polynomial factor. Thus, the final circuit has still polynomial size.

The final circuit is a Boolean circuit of $\mathcal{O}(\text{poly}(n))$ gates with fan-in 2. Since this circuit does not contain any cycles by construction, it can be translated into a directed acyclic graph, where the Boolean inputs constitute the leaves, the gates are the internal nodes, and the output is the root. The circuit therefore computes a compositionally sparse function in the sense of Definition 2.1, which concludes the proof.

□

### 2.2.3 Deep Learning under Compositional Sparsity

While compositionally sparse functions may be approximated by suitable deep networks while avoiding the curse of dimensionality, this is generally not the case for shallow networks.

**Theorem 2.3** (Poggio et al. [129], informal)**.** *Suppose $f$ is a* compositionally sparse *function of input dimension $d$ with directed acyclic graph $G_f$. Let the complexity of a network be the number of its trainable parameters. Then*

1. ***Shallow Networks*** *with one hidden layer require complexity $\mathcal{O}(\varepsilon^{-d})$ to approximate $f$ to an accuracy $\varepsilon > 0$, which is the best bound possible, whereas*
2. ***Deep Networks*** *mimicking $G_f$ require complexity $\mathcal{O}(d\varepsilon^{-2})$ to approximate $f$ to an accuracy $\varepsilon > 0$.*

Combining Theorem 2.2 on the compositional property of all efficiently computable functions with Theorem 2.3 from above, yields the following corollary:

**Corollary 2.4** (cf. Poggio and Fraser [132])**.** *Any efficiently Turing-computable function (Boolean or real-valued) may be approximated to an accuracy of $\varepsilon > 0$ with a deep, sparse network of polynomial complexity in $d$ and $\varepsilon^{-1}$.*

As such, deep neural networks are universal approximators for all practically computable functions, capable of avoiding the curse of dimensionality. Observe that the argument in the case of approximation only requires the in-degree of the function nodes in the directed acyclic graph to be constrained, but does not further restrict the function class from which the "node functions" are drawn.

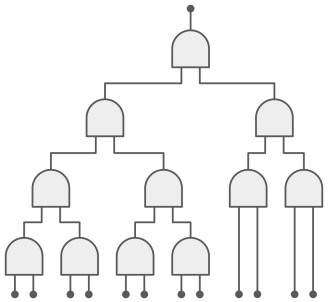

Fig. 2.2 An AND gate with fan-in 12 decomposed in a tree of AND gates with fan-in 2.

## 2.3 Learnability and Optimization of Compositionally Sparse Functions

It is noteworthy, that the curse of dimensionality occurs not only in the context of network complexity, i.e., a question related to approximation. It also extends to optimization, namely, convergence rates and sample complexity. While compositional sparsity explains why deep networks can *represent* efficiently computable functions with polynomial complexity, it does not guarantee that such representations can be *learned* efficiently from input-output pairs. This section explores the theoretical and practical challenges of learning compositionally sparse functions and connects these insights to modern training paradigms.

### 2.3.1 Theoretical Challenges

Learning general compositionally sparse functions from input-output examples faces fundamental complexity barriers. A foundational result by Goldreich et al. [149] demonstrates that, under standard cryptographic assumptions (specifically, the existence of one-way functions), there exist families of Boolean circuits of polynomial size that are not efficiently learnable in the distribution-free setting by any polynomial-time algorithm, regardless of the representation used. This result is representation-independent and applies directly to the class of functions computable by polynomial-size Boolean circuits, which encompasses compositionally sparse functions as defined in this work.

**Theorem 2.5** (Goldreich et al. [149], informal)**.** *Assuming the existence of one-way functions, there exists a polynomial $p$ such that the class of Boolean circuits with at most $p(n)$ gates is not learnable in polynomial time by any polynomial-time evaluable representation class.*

This cryptographic hardness result implies that, without additional structural assumptions or access to more than just input-output pairs, learning arbitrary compositionally sparse functions is infeasible in the worst case. In other words, even though such functions can be efficiently represented and computed, their learnability from examples alone is fundamentally limited by computational complexity barriers.

However, this worst-case hardness does not preclude the efficient learnability of many important subclasses of compositionally sparse functions. For example, Mansour [150] showed that most sparse Boolean functions are easy to learn, and further work has identified tractable subclasses such as staircase functions [151] or sparse polynomials [152]. The key insight is that while compositional sparsity is necessary for efficient representation, it is not sufficient for efficient learnability in the absence of further assumptions or structural information. In practice, providing partial supervision, architectural biases, or exploiting additional properties of the constituent functions can make learning feasible.

Despite these theoretical barriers, deep neural networks that likely represent compositionally sparse functions often achieve impressive performance on real-world tasks with high-dimensional data. This empirical success suggests that, in practice, the structure of real-world problems and the inductive biases of deep neural networks frequently circumvent worst-case hardness. Some theoretical results support this phenomenon: Bauer and Kohler [130], for example, show that in non-parametric regression, the convergence rate of shallow networks diminishes exponentially with the input dimension, whereas deep neural networks can achieve convergence rates that depend only on the interaction order of the underlying compositional function, not the ambient dimension. Further work [131, 153–155] demonstrates that deep neural networks can overcome the curse of dimensionality and efficiently learn compositional functions under suitable assumptions.

An interesting example of sparse Boolean functions is provided by staircase functions, which are hierarchically structured Boolean functions over a high-dimensional hypercube. Such functions can be learned in polynomial time using layer-wise stochastic gradient descent on specialized deep neural networks [151]. The hierarchical structure enables gradient descent to progressively combine low-level features into higher-level ones through network depth. Negahban and Shah [152] show that functions representable as $s$-sparse Boolean polynomials over $n$ variables can be learned via $L_1$-constrained convex optimization inspired by compressed sensing, achieving $O(s^2 n)$ sample complexity. Similarly, Boolean functions with Fourier spectra concentrated on $k$ non-zero coefficients (out of a predefined set $P$ of potential basis elements) can be learned via $L_1$-constrained regression with $O(k \log^4 |P|)$ samples [156].

These results collectively highlight that, although the worst-case learnability of compositionally sparse functions is computationally hard, many natural and structured subclasses remain efficiently learnable. This dichotomy underscores the importance of leveraging additional structure (whether through architectural design, training paradigms, or supervision) to circumvent the limitations imposed by general hardness results.

### 2.3.2 Implications for Architecture Design

Convolutional neural networks address compositional sparsity not because of translational invariance but because the filters are constrained to local patches, which induces sparse Toeplitz weight matrices. This architectural bias, which is independent of the convolutional property, ensures that the network computes a compositionally sparse function. Recent work by Xu et al. [157] demonstrates that such sparsity can be leveraged to obtain significantly tighter generalization bounds than those derived from naive Rademacher complexity, for both sparse and dense networks. This suggests that, in domains where convolutional neural networks excel, that is, where the underlying function is well-approximated by a compositionally sparse structure, deep sparse learners are essential for strong generalization. Notably, these generalization results depend on the sparsity of the weight matrices, not on the convolution per se.

In biological applications, compositional sparsity provides a natural explanation for why architectures that mirror known modular structure often outperform generic dense models, even when trained on the same data. Gene regulatory networks, signaling cascades, and metabolic pathways are organized into modules where each component interacts with a small subset of partners [158, 159], and where layers of regulation compose into higher-level phenotypic effects [10, 160]. Empirical studies have consistently demonstrated that biological networks exhibit sparse, hierarchical connectivity patterns [161, 162] and that this modular organization directly influences how perturbations propagate through regulatory circuits [163]. When we design neural architectures that respect this organization, for instance by constraining connectivity along curated pathway graphs [164] or by factorizing prediction tasks into low-arity modules that correspond to known biological processes [100], we are not merely regularizing the model. Rather, we are aligning its hypothesis class with the compositional sparse structure that any efficiently computable cellular input-

output map must satisfy. In this sense, architecture becomes an explicit encoding of a biological inductive bias [165] that is justified both by empirical modularity and by the theoretical results of this chapter.

On the optimization side, compositional sparsity also impacts the symmetry properties of the network. Dense architectures, such as fully connected networks, possess large symmetry groups: permutations of neurons or layers often leave the function unchanged, resulting in highly degenerate loss landscapes with many equivalent minima [166–169]. In contrast, compositionally sparse architectures, including convolutional neural networks, have much smaller symmetry groups, as their constrained connectivity restricts the set of permissible permutations. For example, convolutional neural networks avoid permutation symmetries in weight space by enforcing translational equivariance, which simplifies the optimization landscape and can accelerate convergence. Recent works have begun to explore general principles of learning in- and equivariances in deep neural networks [170] and the interplay between sparsity, equivariance, and symmetry reduction [171], suggesting that architectural biases that reduce or exploit symmetry can induce useful structure and constraints on learning, thereby serving as a general mechanism for improving learnability and optimization.

While the structural sparsity induced by convolutional neural networks has empirically been shown to be an exceptional fit for vision tasks [172–174], it can generally be challenging to impose the correct sparse structure *a priori* in architectures intended for other domains. Considering the natural language domain, it is reasonable to assume that the entities sharing a meaningful relationship in a sentence or text may shift position and have variable relative pair-wise distances. The rigid sparse structure of a convolutional neural network may therefore not be well-suited.
Transformers, on the other hand, may address this by dynamically learning to *focus* on a still small, but flexible input-dependent subset of tokens via attention [7, 175, 176]. Song et al. [177] rigorously characterize how sparse attention mechanisms in transformers approximate exact attention, revealing that attention patterns naturally exhibit sparsity despite the architecture being dense by design. Recent empirical studies have further shown that transformers often exhibit emergent compositional structure across layers [178–180].
We posit that the key lies in the autoregressive training framework: by training models to predict each token conditioned on previously generated tokens, autoregressive training encourages the incremental construction of complex outputs from simpler

components. This process can naturally lead to the emergence of compositional sparsity through learned intermediate representations.

### 2.3.3 Universality of Auto-Regressive Predictors and Chain-of-Thought

A recent result by Malach [181] can be derived as a corollary of Theorem 2.2. Malach's theorem, obtained independently and in a somewhat different context, establishes that autoregressive next-token predictors are universal learners for any efficiently Turing computable function. Specifically, Malach [181] states:

**Theorem 2.6** (Malach [181], informal)**.** *For any function $f$ that is efficiently Turing-computable, there exists a dataset $\mathscr{D}$ such that training a (linear) auto-regressive next-token predictor on $\mathscr{D}$ results in a predictor that approximates $f$.*

Because Boolean sparse functions are easy to learn [150, Theorem 4.9], it is easy to show that our Theorem 2.2 implies the following statement (which is effectively equivalent to Theorem 2.6):

**Corollary 2.7** (informal)**.** *Any function $f$ that is efficiently Turing-computable, can be learned if training sets are available for each of the sparse constituent functions in one of its decompositions.*

Interestingly, decision trees [150, Theorem 5.10] are sparse Boolean functions because they have small $L_1$-norm. In a smoothed complexity setting where the input distribution is drawn from the biased binary hypercube, and the bias of each variable is randomly chosen, the class of $\log(n)$-depth decision trees satisfies the general staircase property. In turn, staircase functions with a small number of terms are sparse, since they are a sum of parity functions of increasing degree.

Poggio and Fraser [132] argue that a dataset containing the step-by-step output and therefore the intermediate results of constituent functions should suffice to learn any compositionally sparse function conditioned on the learnability of its constituents. The structure of natural language is commonly regarded to be sparse s.t. subsequent tokens only depend on few prior tokens [182, 178]. This compositional structure can be picked up by auto-regressive next-token predictors [183]. The vast corpora of natural language [184, 185], on which modern-day large language models are

trained on, are therefore natural candidates for such datasets, that contain not only end-to-end examples of input-output pairs of functions, but also intermediate results. This may explain complex reasoning observed in present-day large language models.

Recent empirical work by Lindsey et al. [186] provides direct evidence that large language models, such as Claude 3.5 Haiku, internally perform genuine multi-step reasoning in practice. For example, when prompted with *Fact: the capital of the state containing Dallas is*, the model produces the correct answer "Austin" by first inferring that Dallas is in Texas, and then that the capital of Texas is Austin. Attribution graph analysis reveals that the model's computation proceeds through distinct intermediate representations corresponding to "Texas" and "capital," rather than relying solely on memorized shortcuts. This multi-hop reasoning is reflected in the model's internal feature activations and their interactions, supporting the view that compositional sparsity and hierarchical reasoning are not only theoretically motivated but also empirically realized in state-of-the-art models [186].

Chain-of-Thought is one of the most compelling phenomena discovered during the recent study of large language models. In brief, it can be shown that eliciting a series of intermediate reasoning steps significantly improves the ability of large language models to perform complex reasoning [187]. The reasoning steps may be hierarchically structured themselves to break up involved problems into simpler subproblems [188, 189].

**Conjecture 2.8** (Chain-of-Thought exploits Compositionally Sparse Functions)**.** Chain-of-Thought explicitly decomposes a compositionally sparse learning problem into sparse subproblems, each one of which can be learned. As such, it overcomes the complexity of one-shot learning.

The following sketch shows how Chain-of-Thought fits into the compositional sparsity framework. Let $f_\theta : \mathbb{T}^d \to \mathbb{T}$ be a token-to-token predictor, $S[s:e]$ be the subsequence operator on a token sequence $S$ selecting all tokens from the $s$-th to the $e$-th token (inclusive), and $[\cdot, \dots]$ be the concatenation operator concatenating sequences of tokens. Then, using Chain-of-Thought prediction can be understood as repeatedly applying the same predictor to a changing sequence of inputs.

$$f_\theta([\dots, f_\theta([X[2:d], f_\theta([X[1:d], f_\theta(X)])])]) \tag{2.5}$$

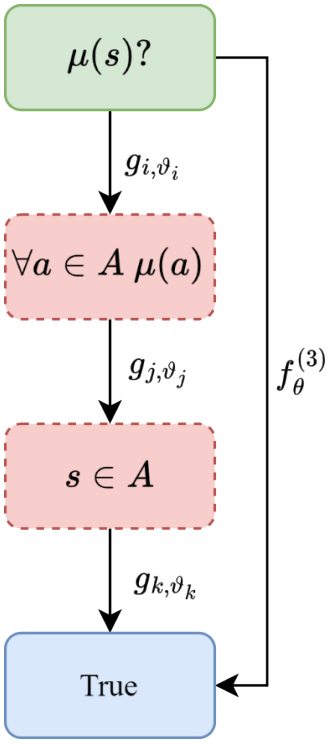


Fig. 2.3 *Is Socrates mortal?* Chain-of-Thought-style intermediate solving steps can simplify this famous question to a sequence of general reasoning steps of less complexity than the specific question at hand.

It is always possible to factor $f_\theta(\cdot)$ over a partition $\bigcup_{i\in I} P_i = \mathbb{T}^d$, s.t.,

$$f_\theta(x) = g_{i,\vartheta_i}(x) \quad \text{with } i \in I, x \in P_i \tag{2.6}$$

and thus understand $f_\theta(\cdot)$ to implicitly be a decision tree assigning each input to the leaf predictor $g_{i,\vartheta_i}(x)$ of the input's partition set (construction akin to Belcak and Wattenhofer [190]). The plausibility of this idea is supported by, e.g., the work of Gan et al. [183], who demonstrated that decision tree structures naturally emerge in auto-regressive language models. In the framework of compositional sparsity, each $g_{i,\vartheta_i}$ constitutes a simpler function of (bounded) density $c \ll d$. As such, the token-to-token predictor is faced with sparse, learnable functions $g_{i,\vartheta_i}$ of limited complexity and can solve the problem via repeated prediction. Note that decision trees themselves are sparse [150], implying that the full function directed acyclic graph is compositionally sparse. To directly predict the result in a single prediction step, i.e., evaluating $f_\theta$ on the input sequence $X$ once, on the other hand constitutes a dense problem and is therefore not (easily) learnable if at all.

However, there remain open questions, e.g., how exactly the subfunctions are represented, selected, and learned. Viewing this problem through the lens of compositional sparsity may be helpful in identifying promising research directions. The work of Cheung et al. [191] shows how a single model can represent several functions using superposition and is thus possibly an answer to the question of representation.

### 2.3.4 Open Questions

Despite significant theoretical and empirical progress, several critical gaps remain in our understanding of how compositional sparsity interacts with deep learning in practice. We highlight below a set of open questions that, if addressed, could unify the theory of compositional sparsity with the practical realities of modern deep learning.

**Which Functions Are Learnable? And by which deep neural networks?** A central challenge is to characterize the classes of compositional sparse functions, represented as directed acyclic graphs, that can be efficiently inferred from data. While compositional sparsity provides a powerful representational framework, it remains unclear for which directed acyclic graph topologies the underlying structure can be reliably discovered by learning algorithms. Notably, recent work suggests that certain function classes, such as staircase functions, may not require a strictly compositional structure, but rather an overlap in the subsets of variables they use. Careful analysis of these cases, as discussed in the literature, may reveal subtler forms of compositionality relevant for learnability.
Optimization in the compositional sparsity framework suggests that the directed acyclic graph of the target function must be reflected in the deep neural network's computational graph as a subgraph implying that only certain architectures can efficiently be optimized for certain target functions. This conjecture constitutes a direct pathway to empirical study of the relationship shared by the graph topologies of the target function and the network. Resulting insights could aid in delimiting the scope of our proposition and competing frameworks of understanding deep neural network optimization behavior.

**How Much Supervision Is Needed for Efficient Learning?** Another open question concerns the amount and type of supervision required to efficiently learn compositional functions. In practice, providing intermediate supervision, such as Chain-of-Thought steps or explicit labels for subproblems, can dramatically reduce the complexity of learning. However, it is not yet well understood how much and what kind of intermediate supervision is necessary to avoid exponential scaling in sample or computational complexity. Determining the minimal supervision needed for efficient learning remains an important direction, both for theory and practice.

**Why Are Multiple Layers Essential in Transformers?** The empirical success of multilayer transformers in pretraining tasks suggests that predicting the next word in natural language may, in general, require learning a compositionally sparse function that cannot be represented by a single threshold layer. This observation raises further questions about the depth and architectural requirements for capturing compositional structure in practice, and about the specific mechanisms by which transformers exploit or induce such sparsity.

**Can SGD Alone Discover Compositional Structure?** Recent work by Beneventano et al. [192] shows that stochastic gradient descent (SGD) can naturally recover the support of the target function at the input layer. This raises a fundamental question: can deep networks, in practice, also discover the sparse constituent functions that would appear in a compositional decomposition of the target function? Is the implicit bias of SGD alone sufficient to induce this structure, or are additional biases, such as explicit sparsity constraints (e.g., $L_1$-regularization) or architectural priors, necessary to reliably recover compositional sparsity across layers? Addressing these questions is crucial for understanding the mechanisms by which deep networks learn and represent compositional structure.

**How Do Neural Networks Choose Among Multiple Decompositions?** Beyond these challenges, another subtle but important issue arises: a given function may admit many distinct compositional sparse decompositions, each corresponding to a different hierarchical arrangement of constituent functions. If a neural network learns such a decomposition, which one does it select among the many possibilities? Are certain decompositions favored due to architectural inductive biases, optimization

dynamics, or properties of the data? What determines this preference, and can it be controlled or predicted? Understanding the factors that bias the learning process toward particular decompositions is important for both interpretability and control.

Addressing these open questions is essential for bridging the gap between the theoretical foundations of compositional sparsity and the practical achievements of deep learning. Progress in these areas will not only deepen our understanding of why deep neural networks work so well, but may also guide the design of more efficient, interpretable, and robust learning systems.

## 2.4 Conclusions

In this chapter, we established a formal connection between computational complexity and the representational power of deep neural networks. We proved the conjecture that all efficiently Turing-computable functions are compositionally sparse. This class of functions can be decomposed into a polynomial number of constituent functions, each depending on only a small, constant number of variables.

This theoretical finding provides a possible explanation for the empirical success of deep learning on high-dimensional problems where classical methods fail. While classical learning theory predicts an exponential "curse of dimensionality", deep networks can avoid this curse in *approximation* by exploiting the inherent compositional sparsity of the target function. Consequently, deep, sparse networks can approximate any efficiently computable function with polynomial, rather than exponential, complexity.

However, the ability to *represent* a function efficiently is distinct from the ability to *learn* it from data. We have seen that architectural biases, like the local connectivity in convolutional neural networks, and training paradigms, such as the incremental reasoning in Chain-of-Thought, provide powerful mechanisms for discovering this implicit structure. In these cases, the learning algorithm is tasked with inferring a hidden hierarchy from the data.

In biological applications, the challenge is to instantiate this principle in concrete tasks. Predicting cellular responses to genetic or chemical perturbations, annotating cell types in continuously updated atlases, or inferring latent trajectories from time-resolved single-cell data all amount to learning complex functions under strong

compositional and modular constraints. Recent work has demonstrated that models incorporating explicit biological structure outperform generic approaches. The GEARS model [100] combines deep learning with a knowledge graph of gene-gene relationships to predict transcriptional responses to novel multi-gene perturbations, achieving 40% higher precision than prior methods by explicitly modeling sparse gene regulatory interactions. Similarly, PDGrapher [101] leverages graph neural networks to map relationships among genes, proteins, and pathways, successfully predicting therapeutic targets across 19 datasets and 11 cancer types by respecting the modular organization of cellular signaling cascades.

Beyond perturbation prediction, compositional structure has proven essential for single-cell analysis tasks. Pathway-guided neural networks [193] that constrain layer connectivity to follow known signaling pathways achieve superior interpretability while maintaining predictive performance in cell-type classification. The Compositional Perturbation Autoencoder (CPA) [194] explicitly models perturbation effects as compositional, allowing accurate prediction of 97.6% of missing genetic combinations in Perturb-seq experiments. Models such as SigPrimedNet [195] and CellTICS [196] incorporate hierarchical biological pathways directly into network architectures, demonstrating that aligning model structure with known biological modularity improves both accuracy and biological interpretability. Gene regulatory network inference methods like GRACE [197] and CEFCON [198] use graph neural networks with explicit causal constraints to recover compositionally sparse regulatory programs from single-cell data. The scapGNN framework [199] transforms sparse single-cell profiles into gene-cell association networks to infer pathway activities, while d-scIGM [200] builds deep generative models guided by hierarchical pathway knowledge to enhance biological interpretability.

The results of this chapter suggest that architectures which expose compositional constraints explicitly, whether by mirroring known pathway structure, by factoring predictions across ontological hierarchies, or by decomposing dynamics into interacting subsystems, should be able to approximate the underlying biological maps with polynomial complexity. These empirical validations across diverse biological tasks confirm that compositional sparsity is not merely a theoretical construct but a practical design principle that, when properly instantiated, enables efficient learning even in high-dimensional biological spaces.

Throughout this chapter, we have focused on compositional sparsity as a property of the function being learned, whether encoded through architectural constraints like pathway-guided connectivity, or emerging through optimization dynamics in transformers and autoregressive training. But many biological domains also exhibit explicit compositional structure in their output taxonomies. Cell types form nested hierarchies from broad lineages to specialized subtypes, diseases are classified into increasingly specific categories, and proteins belong to families with shared evolutionary origins. When such hierarchical relationships among labels are known a priori, can we leverage this structure not through network architecture, but through the learning objective itself?

The next chapter explores this complementary perspective, shifting from compositional structure in function space to hierarchical structure in label space. We investigate how encoding known ontological relationships directly into the loss function can improve generalization in single-cell type annotation, where the output taxonomy naturally exhibits the compositional principles developed in this chapter.

# Chapter 3

# Hierarchical Cross-Entropy Loss for Single-Cell Annotation

*The content of this chapter is largely based on the research originally presented in [201].*

While the compositional sparsity framework provides a compelling explanation for the representational efficiency of deep networks, it addresses only one side of the learning problem. An equally potent source of prior knowledge often lies not in the hidden computational mechanics of the function $f$, but in the explicit relationships that govern its possible outcomes in the output space $\mathscr{Y}$. This chapter pivots from the properties of the function to the properties of its codomain, arguing that a vast and frequently overlooked opportunity for improving machine learning models involves directly modeling the known structure of the output space itself.

This perspective aligns with the established field of structured output prediction [202–204]. In contrast to standard classification or regression, which typically treats the output space $\mathscr{Y}$ as an unstructured set of independent labels or a simple Euclidean space, structured prediction acknowledges that $\mathscr{Y}$ often possesses a rich internal grammar, geometry, or topology. The learning task is consequently transformed from merely selecting the most probable label to finding the highest-scoring valid structure $y \in \mathscr{Y}$ given an input $x$. This often involves an inference step to ensure that the predicted output is globally consistent with the constraints of the output space. In many scientific domains, this structure is not a hidden property to

be inferred but is instead the product of decades of cumulative scientific consensus, explicitly encoded in knowledge bases and ontologies.

In canonical machine learning tasks, particularly in computer vision and natural language processing, the output space is treated as a flat set of mutually exclusive categories. The standard cross-entropy loss function embodies this assumption: it treats all classes as equidistant and all misclassifications as equally erroneous. A model trained on ImageNet learns to map an image to one of a thousand discrete labels, with mistaking a cat for a dog penalized identically to mistaking a cat for a truck [205, 107]. While this simplification suffices for many applications, it breaks down in domains where labels possess rich, known relational structure.

Biology is a preeminent example where ignoring output structure proves most detrimental. Biological ground truth is frequently complex, multi-leveled, and governed by known relational principles. Consider cell biology, where a CD4+ T helper cell is a subtype of T cell, which is a lymphocyte, which is an immune cell. Forcing a classifier to choose between "T cell" and "CD4+ T cell" as if they were unrelated classes is fundamentally flawed. Similarly, protein function prediction relies on the Gene Ontology (GO), where functional terms are organized in a directed acyclic graph representing a multi-level hierarchy of biological processes, molecular functions, and cellular components [206, 207]. In CRISPR-Cas9 screens, perturbations of two genes in the same pathway often produce identical phenotypes, yet a standard classifier would be incorrectly penalized for confusing them when such confusion represents a biologically meaningful insight [208, 209]. The central premise is that this domain knowledge should be a core component of the learning process, encoded directly into the training objective.

To demonstrate this principle, we focus on automated cell type annotation in single-cell RNA sequencing data. Section 3.1 briefly introduces the fundamentals of single-cell RNA sequencing technology, including modern experimental workflows, data characteristics, and preprocessing steps. Section 3.2 introduces the task of cell type annotation and its hierarchical nature. Section 3.3 describes an out-of-distribution (OOD) evaluation framework that reflects real-world usage scenarios. Section 3.4 presents the hierarchical cross-entropy (HCE) loss function, which incorporates ontological structure into model training. In Section 3.5 we reports empirical results showing that HCE significantly improves annotation performance across multiple model architectures in the OOD setting. Finally, Section 3.6 discusses

the broader implications of structured output modeling in biology and potential avenues for future research.

## 3.1 Single-Cell RNA Sequencing

Single-cell RNA sequencing (scRNA-seq) has revolutionized our ability to profile gene expression at the resolution of individual cells, revealing cellular heterogeneity that is obscured by traditional bulk RNA sequencing [210, 211]. Unlike bulk approaches that measure averaged expression across millions of cells, scRNA-seq captures the complete transcriptome of thousands to millions of individual cells, enabling the identification of rare cell types, the mapping of cellular differentiation trajectories, and the discovery of cell-type-specific responses to perturbations [212].

Modern scRNA-seq workflows, particularly droplet-based platforms such as 10x Genomics, encapsulate individual cells within nanoliter-sized droplets along with barcoded beads [213, 214]. Each bead carries unique molecular identifiers that tag mRNA molecules from a single cell, allowing massively parallel sequencing of thousands of cells in a single experiment. The resulting data takes the form of a high-dimensional gene expression matrix, where rows represent genes (often 20,000+ protein-coding genes) and columns represent individual cells. However, scRNA-seq data exhibits distinctive characteristics that distinguish it from bulk sequencing. First, the data is inherently sparse: due to low capture efficiency and the stochastic nature of gene expression, many genes appear as zero counts even when they are expressed at low levels. Second, the data displays substantial technical noise from amplification biases and library preparation artifacts. Third, biological variability across cells, even those of the same nominal type, can be considerable, reflecting genuine differences in cell state, cell cycle phase, and microenvironmental context [215].

To address these technical challenges, scRNA-seq data typically undergoes a series of preprocessing steps before analysis. Size-factor normalization adjusts for differences in sequencing depth across cells, ensuring that total counts are comparable. Log-transformation with a pseudocount (e.g., $\log(x+1)$) stabilizes variance and makes the data more suitable for downstream statistical modeling. Quality control procedures filter out low-quality cells with anomalously low gene counts or high mitochondrial content, which can indicate cell damage or stress.

These preprocessing steps are essential for making the data amenable to machine learning models that assume inputs are on comparable scales [216–218].

## 3.2 Automated Cell Type Annotation

Cell type annotation is a core step in scRNA-seq pipelines. The quality of annotations directly impacts downstream analyses, including mapping cellular diversity across tissues and deciphering cell-type-specific regulatory mechanisms. Manual annotation remains time-consuming and dependent on domain-specific expertise, but the rapid adoption of scRNA-seq as a standard lab technique has created an urgent need for automated, scalable solutions [210]. With repositories like the Human Cell Atlas (HCA) [16] and CELLxGENE [17] now containing over 100 million cells, accurate and robust annotation methods are a critical first step in translating these large-scale datasets into actionable biological insights [219, 220].

Automated atlas-level cell type annotation can be framed as a supervised classification problem, where models assign labels to individual cells based on gene expression profiles, using reference annotations provided by original studies [221–223]. A defining feature of this task is that cell types are organized within a hierarchical ontology [224, 225], forming a multi-level taxonomy. For example, "leukocytes" represent a broad category that contains "lymphocytes", which in turn includes more specific subtypes such as "B cells". However, annotation practices vary substantially between studies—some assign broad categories, while others distinguish fine-grained subtypes. This inconsistency in label granularity introduces ambiguity into the training signal, as models must infer the appropriate level of resolution without explicit guidance. More formally, the annotation task can be viewed as learning a function $f : \mathscr{X} \to \mathscr{Y}$, where $\mathscr{X}$ is the space of gene expression profiles and $\mathscr{Y}$ is a structured label space defined by a directed acyclic graph (DAG). In this graph, each node corresponds to a cell type and directed edges represent `is_a` subtype relationships. This structure captures relationships across varying levels of annotation granularity [226–228].

We use the cell ontology obtained from the Ontology Lookup Service (OLS) at EMBL-EBI[1] as the hierarchical scaffold for all analyses [225]. We restrict the

[1] https://www.ebi.ac.uk/ols/ontologies/cl

ontology to the 164 distinct cell types observed in the training set (see Section 3.3). In CELLxGENE, which is the atlas used in our study, cell types are annotated by the original data contributors and then harmonized by mapping each label to the closest cell ontology term as specified by the portal's data schema. While the cell ontology offers a valuable scaffold for representing hierarchical relationships among cell types, it is important to note that its structure is continuously being revised where certain definitions and mappings between cell types remain under active refinement.

Because each cell type corresponds to a node in the DAG, we can further classify them based on the type of node they represent. A node is defined as a leaf if it has no children in the pruned ontology and as an internal node if it has at least one child. We also distinguish between connected nodes, which has at least one parent or child present in the curated training set, and isolated nodes, which has none of their ancestors or descendants represented in the training data. These definitions were used to assess how the hierarchical loss propagates information across the ontology (e.g., Fig. 3.5).

## 3.3 The Out-of-Distribution Setting

Many methods have been developed to perform automated cell type annotation, ranging from logistic regression to deep learning architectures [28, 19, 229, 230]. Recent benchmarking studies have shown that deep learning models outperform simpler methods as the number of cells in a dataset increases [226]. Importantly, these evaluations were conducted using donor-partitioned training and test splits, a design we refer to as the in-distribution (ID) setting (Fig. 3.1a). While useful for controlled comparisons, such splits do not reflect how cell atlases evolve in practice, where new studies are continually added and must be annotated upon release.

To better evaluate generalization to newly released studies, we consider an out-of-distribution (OOD) setup in which models are tested on datasets not seen during training (Fig. 3.1b). The dataset used in this study originates from the same filtered subset of the CELLxGENE census (version 2023-05-15) [17] that was curated for the scTab study [226]. This subset was constructed by applying strict inclusion criteria to the full census: only primary human cells profiled with 10x Genomics technologies were retained and the feature space was limited to 19,331 human protein-coding genes. Cell types were required to appear in at least 5,000 cells

drawn from a minimum of 30 donors. All gene expression profiles were size-factor normalized to 10,000 counts per cell and log-transformed with a pseudocount of 1 (i.e., $f(x) = \log(x+1)$). The resulting dataset included 22,189,056 cells annotated with 164 distinct cell types, spanning 5,052 donors and 56 tissues. For the in-distribution (ID) task, we adopted the same donor-partitioned data split used in Fischer et al. [226]—that is, 15,240,192 cells for training, 3,500,032 for validation, and 3,448,832 for testing.

We trained three methods with increasingly complex architectures (a linear classifier, a multilayer perceptron (MLP), and TabNet [231]) on the training set of 15.2 million cells (Fig. 3.1c). We then evaluate each method on 2.6 million human cells from 21 studies newly added during the 2023-12-15 release of the CELLxGENE census, spanning 470 donors, 16 tissues, and 80 of the original 164 cell types represented in the training set. Despite being evaluated on the same cell types profiled with the same assays, macro F1-scores dropped by 24-32% for the linear classifier, MLP, and TabNet when moving from the ID case (Fig. 3.1d) to the OOD setting (Fig. 3.1e), underscoring the limitations of current modeling strategies in generalizing across studies.

Classification performance are evaluated using the macro-averaged F1-score (macro F1-score) which computes the unweighted average of the F1-scores across all cell types. This metric ensures that each cell type contributes equally to the overall score, regardless of class imbalance or prevalence in the dataset. For $C$ cell types, the macro-averaged F1-score is computed as

$$\text{macro F1-score} = \frac{1}{C}\sum_{i=1}^{C}\frac{2\cdot\text{precision}_i\cdot\text{recall}_i}{\text{precision}_i+\text{recall}_i} \tag{3.1}$$

where $\text{precision}_i$ and $\text{recall}_i$ are defined for the $i$-th class as the following

$$\text{precision}_i = \frac{\text{TP}_i}{\text{TP}_i+\text{FP}_i}, \qquad \text{recall}_i = \frac{\text{TP}_i}{\text{TP}_i+\text{FN}_i}. \tag{3.2}$$

Here, the terms $\text{TP}_i$, $\text{FP}_i$, and $\text{FN}_i$ denote the number of true positives, false positives, and false negatives for the $i$-th cell type, respectively. We followed the evaluation framework introduced by Fischer et al. [226] in the scTab study, particularly because of how the authors handled differences in the granularity of annotations that can occur across different studies. Namely, a predicted label is considered correct if it

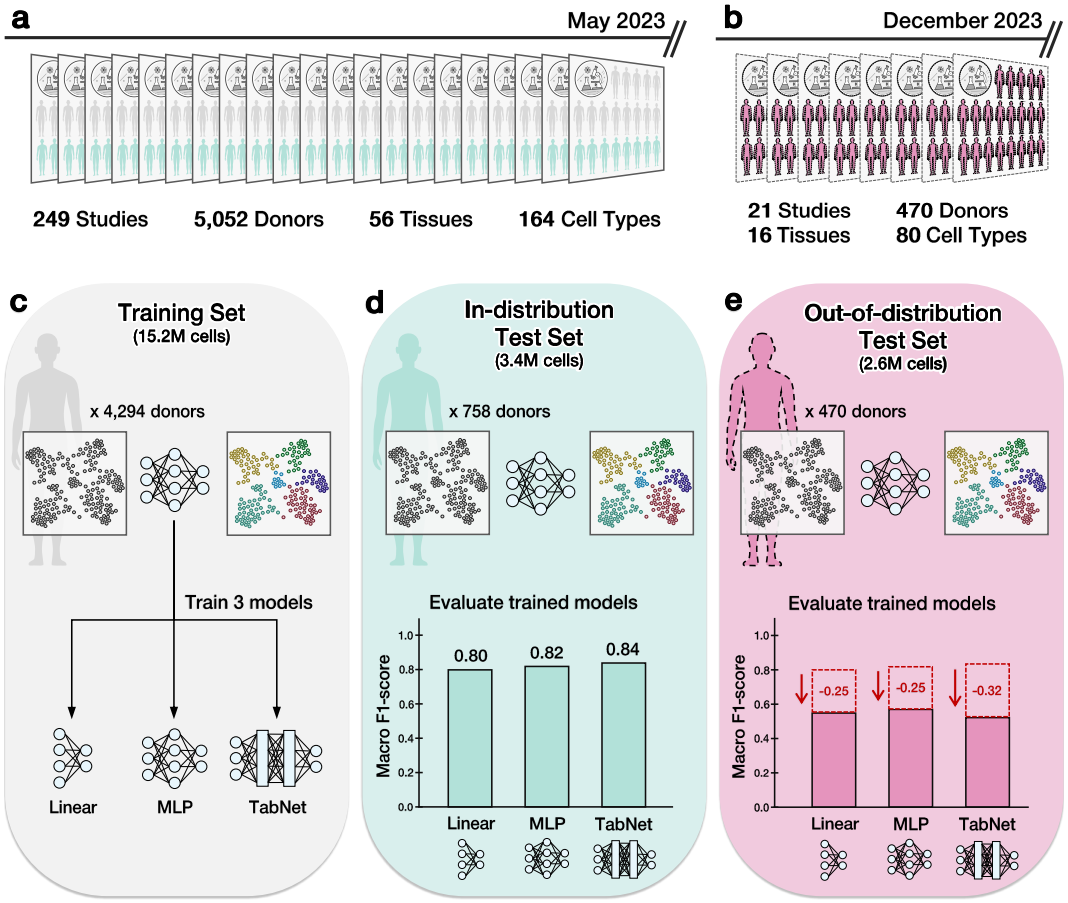


Fig. 3.1 Evaluating model generalization in continuously updated single-cell atlases reveals sharp out-of-distribution performance drops for the annotation task. **a** A curated subset of the CELLxGENE census (May 2023 release) consisting of 22.2 million human cells annotated with 164 curated cell types, spanning 5,052 donors, 56 tissues, and 249 studies. All cells were profiled using 10x Genomics platforms. **b** Out-of-distribution (OOD) test set comprised of 2.6 million newly added cells from 21 studies in the December 2023 release. These cells span 470 donors and 16 tissues, and are annotated with 80 of the 164 original training cell types. All cells are also profiled using 10x Genomics platforms. **c** We train three models (linear classifier, multilayer perceptron (MLP), and TabNet) on a donor-partitioned training set comprised of 15.2 million cells from the May 2023 CELLxGENE census. **d** In-distribution (ID) test set comprised of 3.4 million cells from the May 2023 release of the CELLxGENE census, held out by donor. The linear model, MLP, and TabNet achieve 80%, 82%, and 84% macro F1-scores, respectively. **e** All models exhibit substantial performance drops out-of-distribution (OOD): macro F1-scores decrease to 55%, 57%, and 52% for the linear model, MLP, and TabNet, respectively. The dashed red bars indicate the in-distribution performances for comparison.

exactly matches the ground-truth label or if it corresponds to a descendant of the ground-truth label in the cell ontology (i.e., the prediction is a more specific subtype).

This accounts for the fact that some datasets provide coarse-grained annotations (e.g., "T cell") while others include more detailed subtypes (e.g., "CD4-positive, $\alpha$-$\beta$ T cell"). In such cases, predicting a valid subtype is treated as correct, as it remains consistent with the original label. Any other prediction including a coarser label (i.e., a parent node) or an unrelated class is considered incorrect.

## 3.4 Hierarchical Cross-Entropy Loss

While there are methods that leverage ontological information for cell type annotation, they do not enforce hierarchical consistency as an integral part of their predictive framework. For example, OnClass maps both transcriptomic profiles and cell ontology structure into a joint embedding space, enabling both the annotation of unseen cell types and the identification of marker genes [227]. However, it operates primarily as a nearest-neighbor or embedding search algorithm and does not couple hierarchical relationships to the learned probabilities for each cell. As a result, sibling classes or intermediate states can still be misassigned if their embeddings overlap in feature space. As another example, popV aggregates predictions from multiple classifiers using ontology-based voting, producing robust consensus labels and uncertainty estimates for ambiguous or outlier populations [222]. Yet, the ontology is used only as a scaffold for post hoc reconciliation and not as a guide for model optimization. This means that hierarchical constraints are not encoded in training and possible conflicts or inconsistencies in the ensemble are resolved heuristically. In contrast, SCimilarity focuses on metric learning for scalable, cross-study retrieval of transcriptionally similar cells, using the ontology at training time to exclude ambiguous annotation pairs when sampling triplets for a contrastive loss function [232]. The learned representation supports high-quality search and transfer tasks but is not directly optimized for hierarchical or taxonomic consistency when determining class probabilities.

To address these shortcomings, we introduce a hierarchical cross-entropy (HCE) loss that explicitly incorporates the structural relationships between cell types. With the standard cross-entropy, the loss is computed directly from raw model predictions, treating all cell types as independent classes. Let $\mathbf{p} = (p_1, \ldots, p_C)$ denote the raw predicted probabilities for $C$ different cell types. The standard cross-entropy loss is

given by

$$\mathscr{L}_{\text{CE}} = -\sum_{i=1}^{C} 1\{\text{label} = i\} \log p_i \tag{3.3}$$

where $1\{\text{label} = i\}$ is an indicator function that equals 1 if the true class label is the $i$-th cell type and 0 otherwise. The HCE adjusts these predictions to reflect hierarchical dependencies encoded in the ontology's directed acyclic graph (DAG). The adjusted score $s_i$ for the $i$-th cell type is computed as the sum of the predicted probability for its label and the predicted probabilities of all its descendant subtypes

$$s_i = p_i + \sum_{j \in \mathscr{D}(i)} p_j \tag{3.4}$$

where $\mathscr{D}(i)$ denotes the set of all descendants of cell type $i$ in the DAG. This adjustment ensures that the probability of a parent node reflects its entire subgraph. The hierarchical loss is then

$$\mathscr{L}_{\text{HCE}} = -\sum_{i=1}^{C} 1\{\text{label} = i\} \log s_i \,. \tag{3.5}$$

This formulation directly parallels the evaluation framework where predictions are considered correct if they match the ground-truth label or any of its descendants. By aligning the training objective with the assessment criterion, HCE encourages cell type classification models to distribute probability mass in a way that respects biological hierarchy and annotation granularity.

Consider an ontology subgraph that is rooted at the node *T cell* which includes subtype labels such as *CD4+ T cell*, *CD8+ T cell*, and *γ-δ T cell*. The HCE enables classifications models to predict fine-grained subtypes when available, while also deferring to parent categories when annotations are coarse or ambiguous. For example, if some studies annotate cells as *T cell* while others use more specific labels such as *CD4+ T cell* or *CD8+ T cell*, the adjusted score is computed as

$$s_{\text{T cell}} = p_{\text{T cell}} + p_{\text{CD4+}} + p_{\text{CD8+}} + p_{\gamma\text{-}\delta} + \cdots \tag{3.6}$$

This hierarchical setup allows the model to aggregate subtype information upward, improving consistency across annotations with varying granularity (Fig. 3.2a).

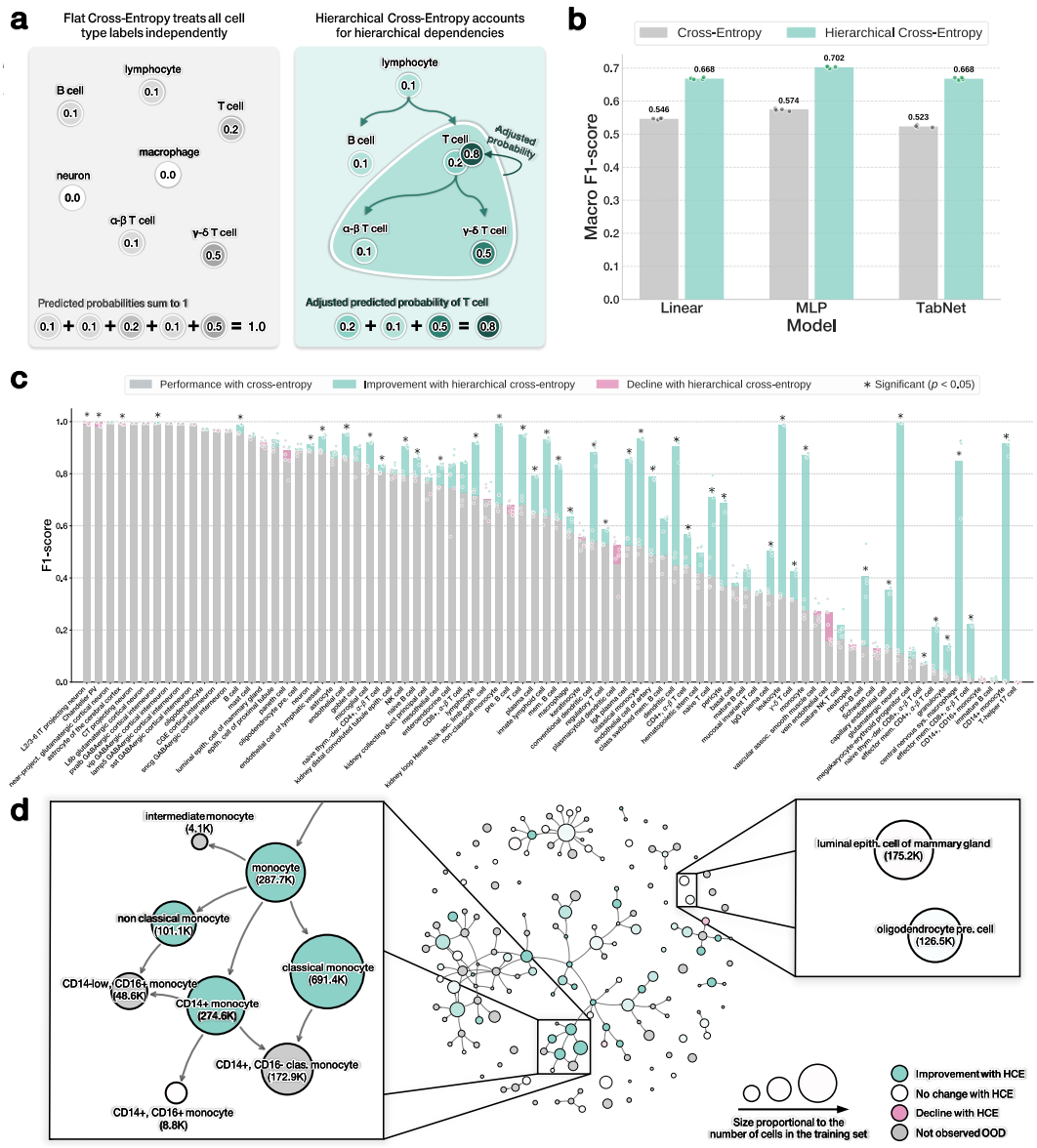


Fig. 3.2 Hierarchical cross-entropy (HCE) loss improves perfomances across architectures. **a** The standard cross-entropy (CE) loss defines a probability distribution over a flat label set, treating each cell type independently and requiring that probabilities sum to one across the ontology. The hierarchical cross-entropy (HCE) loss modifies these predictions by propagating probability mass up the directed acyclic graph (DAG) of the cell ontology: parent nodes such as "T cell" accumulate mass from their more specific descendants, such as "$\alpha$-$\beta$ T cell" and "$\gamma$-$\delta$ T cell", encouraging biologically coherent predictions. **b** The HCE loss improves macro F1-scores by 12-15% on out-of-distribution (OOD) evaluations across the linear classifier, multilayer perceptron (MLP), and TabNet. **c** Per-cell type performance changes induced by the HCE loss strategy for the MLP model, shown relative to standard cross-entropy. Each dot represents the performance of an individual run (color coding remains the same as in the legend). **d** Improvements from hierarchical cross-entropy loss for the MLP model visualized directly on the cell ontology DAG consisting of all 164 cell types seen in the training set. Node size reflects the number of cells of that type seen in training, while color indicates the change in F1-score (green for improvement, red for decline). Grey nodes correspond to cell types not observed in the OOD test set.

We implemented the hierarchical cross-entropy loss using a reachability matrix $\mathbf{R} \in \{0,1\}^{C \times C}$, where element $R_{ij} = 1$ if the $j$-th class is reachable from the $i$-th class (meaning $j$ is either $i$ itself or $j$ is a descendant of $i$ in the hierarchy), and $R_{ij} = 0$ otherwise. The reachability relation encoded in this matrix is a partial order and has the following mathematical properties:

- reflexive: every class is reachable from itself (diagonal elements are 1);
- antisymmetric: if class $i$ can reach $j$ and $j$ can reach $i$, then $i = j$;
- transitive: if class $i$ can reach $j$ and $j$ can reach $k$, then $i$ can reach $k$.

Indeed, the reachability matrix represents the transitive closure of the inverted adjacency matrix of the hierarchical DAG structure. Since the original DAG encodes `is_a` relationships from child to parent, we invert the edge directions to enable parent-to-descendant reachability, ensuring reflexivity by setting the diagonal to 1. Each trained model outputs a raw probability distribution $\mathbf{p} = (p_1, \ldots, p_C)$ over the class labels. The adjusted scores are computed via matrix-vector multiplication: $\mathbf{s} = \mathbf{R}\mathbf{p}$, which efficiently aggregates descendant probabilities for each class. We then apply a log transformation with numerical stability $\log(\mathbf{s} + \varepsilon)$, where $\varepsilon = 10^{-6}$. The final loss uses a weighted negative log-likelihood as implemented in PyTorch, with class weights computed following scikit-learn's `compute_class_weight` approach: $w_i = N/(C \cdot n_i)$, where $N$ is the total number of samples, $C$ is the number of classes, and $n_i$ is the count of samples for the class $i$. The complete loss for a single training sample $x$ with true label $t$ is

$$\mathscr{L}_{\mathrm{HCE}}(x) = -w_t \log(s_t + \varepsilon). \tag{3.7}$$

This formulation maintains consistency with the models trained with the weighted cross-entropy, while incorporating hierarchical structure through efficient matrix operations.

## 3.5 Results

We evaluated three model architectures of increasing complexity: a linear classifier, a multi-layer perceptron (MLP), and the TabNet transformer model. Each model takes

as input the full set of 19,331 human protein-coding genes. To ensure a fair comparison across models and with prior work, we adopted the architecture configurations and hyperparameters used in the scTab benchmarking study from Fischer et al. [226] (see Tables 3.1-3.3). The models using cross-entropy (CE) versus hierarchical cross-entropy (HCE) share identical architecture and hyperparameter settings; the loss term is the only change that is different between them. Specifically, for the models with CE, we used the best hyperparameters available according to the original scTab study. For the models using the HCE loss, we did not perform additional hyperparameter tuning and instead kept the (possibly suboptimal) hyperparameters used for the models with CE. Note that, while recent efforts have explored large-scale foundation models to learn transferable embeddings for single-cell data, such approaches have not yet demonstrated clear advantages over simpler, task-specific approaches for cell type annotation [26, 226]. We therefore focused on methods where we could easily isolate and study the direct effects of implementing the hierarchical cross-entropy strategy.

Table 3.1 Values of the hyperparameters used to run the linear classifier. During training, the learning schedule was linear, the maximum learning rate was 0.0005, the optimizer was AdamW, and the weight decay parameter was set to 0.01.

| **Parameter** | **Value** |
|---|---|
| **batch_size** | 2048 |
| **learning_rate** | 0.0005 |
| **learning rate scheduler** | torch.optim.lr_scheduler.StepLR |
| | gamma = 0.9 |
| | step_size = 1 epoch |
| **optimizer** | AdamW |
| **weight_decay** | 0.01 |
| **augment_training_data** | False |

To assess changes in predictive performance induced by the ontology-aware training strategy, we computed per-cell type differences in macro F1-score between models trained with standard cross-entropy and hierarchical cross-entropy across four independent training runs. For each cell type, a paired $t$-test was performed and $p$-values were adjusted using the Holm-Bonferroni method to correct for multiple hypothesis testing [233]. Statistically significant differences indicate cell types

Table 3.2 Values of the hyperparameters used to run the multilayer perceptron (MLP). This model had 8 hidden layers (`n_hidden`) each with 128 neurons (`hidden_size`). During training, the learning schedule was linear, the maximum learning rate was 0.002, the optimizer was AdamW, the hidden layer dropout was 0.1, and the weight decay parameter was set to 0.05.

| Parameter | Value |
|---|---|
| **batch_size** | 2048 |
| **learning_rate** | 0.002 |
| **learning rate scheduler** | torch.optim.lr_scheduler.StepLR<br>gamma = 0.9<br>step_size = 1 epoch |
| **optimizer** | AdamW |
| **weight_decay** | 0.05 |
| **n_hidden** | 8 |
| **hidden_size** | 128 |
| **dropout** | 0.1 |
| **augment_training_data** | True |

for which ontology-aware training produces consistent changes beyond random variability.

The source code is available under the MIT license at https://github.com/microsoft/hce-classification. Model checkpoints needed to reproduce the results can be found at https://zenodo.org/records/17211022.

Applying the HCE loss improved out-of-distribution macro F1-scores by 12-15% for the linear classifier, MLP, and TabNet, without modifying their architecture or tuning any hyperparameters (Fig. 3.2b). These consistent gains demonstrate the widespread benefits of hierarchy-aware training for cell type annotation tasks. The HCE loss function enables the recovery of roughly half of the performance drop observed when models are applied to new studies, underscoring the practical value of aligning training objectives with ontology structure. To further assess the consistency of this effect, we evaluated performance across each of the 21 held-out studies individually (Fig. 3.3). Outside of just one study while using the linear model, all HCE-trained models showed statistically significant improvements, highlighting the robustness of this approach across diverse experimental settings.

Table 3.3 Values of the hyperparameters used to run TabNet presented in Fischer et al. [226]. This model has three main components: (1) a feature transformer, which is a multi-layer perceptron with batch normalization, (2) skip connections, and (3) a gated linear unit nonlinearity. The feature transformer maps the input gene expression data into a latent space of `n_a`+`n_d` dimensions, where the `n_a` = 64 portion is used to calculate attention masks and the `n_d` = 128 is used for cell type annotation. During training, the learning schedule was linear, the maximum learning rate was 0.005, the optimizer was AdamW, the feature attention mask is obtained by applying the 1.5-entmax function, and the weight decay parameter was set to 0.05.

| **Parameter** | **Value** |
|---|---|
| **batch_size** | 2048 |
| **learning_rate** | 0.005 |
| **learning rate scheduler** | torch.optim.lr_scheduler.StepLR<br>gamma = 0.9<br>step_size = 1 epoch |
| **optimizer** | AdamW |
| **weight_decay** | 0.05 |
| **n_d** | 128 |
| **n_a** | 64 |
| **n_shared** | 3 |
| **n_independent** | 5 |
| **n_steps** | 1 |
| **lambda_sparse** | $10^{-5}$ |
| **mask_type** | entmax |
| **virtual_batch_size** | 256 |
| **augment_training_data** | True |

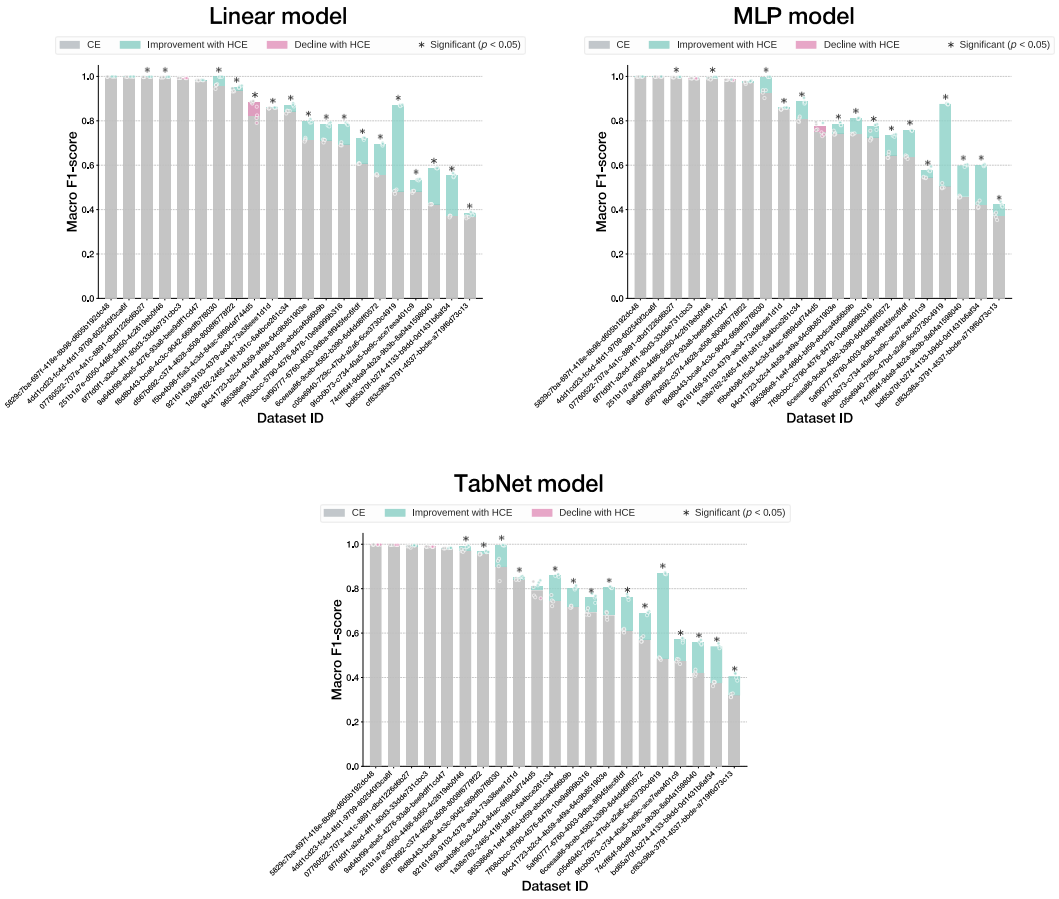


Fig. 3.3 Performance gains from the hierarchical cross-entropy (HCE) loss across 21 out-of-distribution test datasets for the linear classifier, multilayer perceptron (MLP), and TabNet. Improvements are measured relative to the same models trained with standard cross-entropy loss. Each dot represents the performance of an individual run (color coding remains the same as in the legend).

To better understand where the improvements from hierarchical training arise, we identified cell types that exhibited statistically significant changes in performance between models trained with and without the HCE loss. In the MLP model, for example, HCE led to improvements of up to 0.9 in F1-score for cell types such as “glutamatergic neuron” and “CD14+ monocyte” (Fig. 3.2c). Examining these effects in relation to the cell ontology, we found that the largest gains occurred for internal nodes, particularly those embedded in densely connected regions of the DAG where related types were annotated in the training data. In contrast, leaf nodes, especially structurally isolated ones, showed more modest gains (Fig. 3.2d and Figs. 3.4 and

3.5). This aligns with the intuition that the hierarchical loss is most effective when it can propagate signal across nearby cell types. Similar trends were observed for the linear and transformer-based models, highlighting the architecture-agnostic nature of the effect (Fig. 3.6). Importantly, gains were largely unaffected by a cell type's rarity, the number of contributing studies or tissues, and the diversity of sequencing technologies used, further underscoring the robustness of the approach (Fig. 3.7). Finally, these gains extend to cells observed in new contexts, including across diseases and tissues not seen in the training set where we also observe consistent improvements (Figs. 3.8 and 3.9).

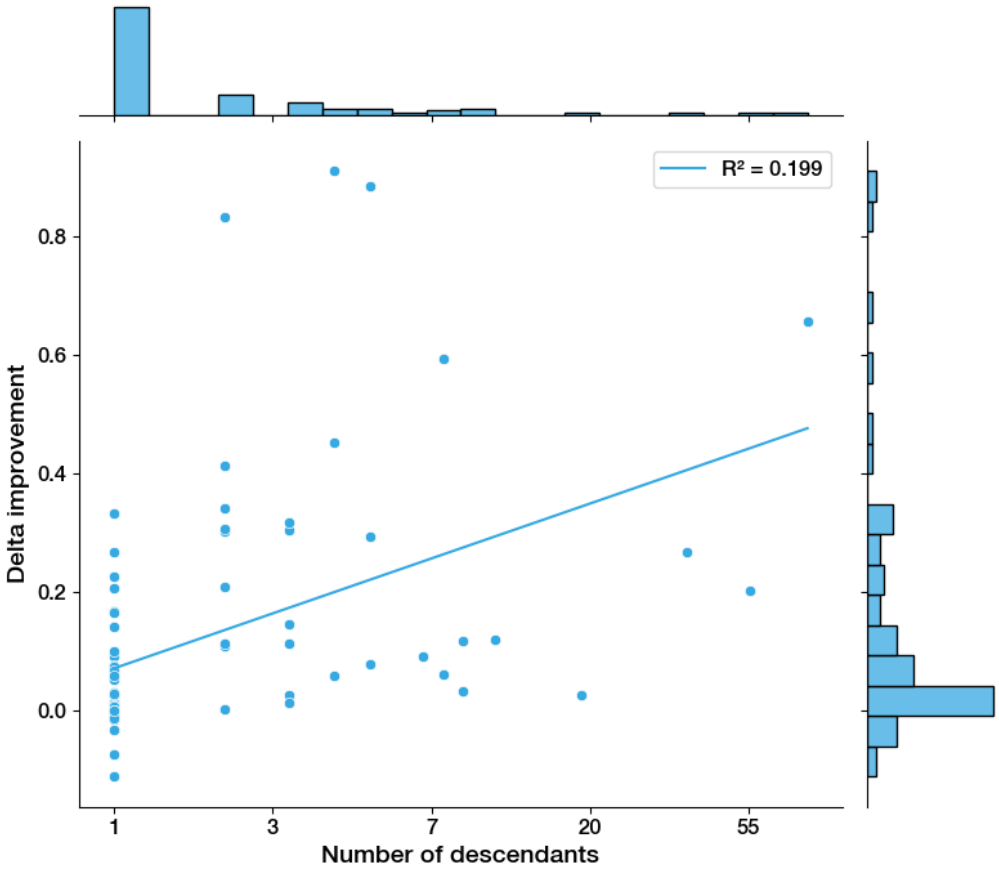


Fig. 3.4 Performance gains from the hierarchical cross-entropy loss in the multilayer perceptron (MLP) as a function of the number of descendants of a given cell type in the cell ontology directed acyclic graph (DAG) (in log scale).

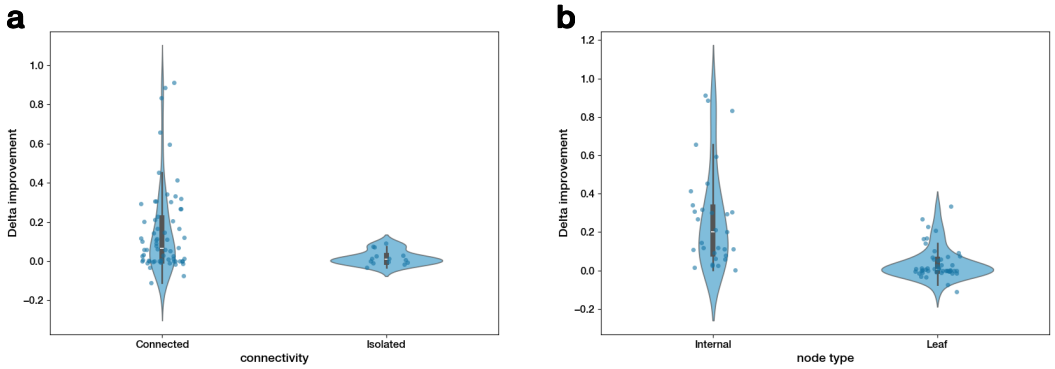


Fig. 3.5 Performance gains from the hierarchical cross-entropy loss in the multilayer perceptron (MLP) as a function of structural properties of the cell ontology directed acyclic graph (DAG). **a** Performance gains from the hierarchical loss for the MLP model on connected versus isolated nodes. **b** Performance gains from the hierarchical loss for the MLP model on internal nodes versus leaves.

## 3.6 Discussion

Our results challenge the view that increasing model complexity is the primary route to improved cell type annotation at atlas scale. Instead, we demonstrate that aligning the training objective with biological structure through a hierarchical cross-entropy loss consistently improves generalization across model classes, from linear classifiers to transformers. This finding highlights that explicit incorporation of domain knowledge into learning objectives can yield performance gains that rival or exceed those achieved through architectural complexity alone.

Beyond architectural considerations, the hierarchical cross-entropy framework suggests a strategy for building more effective training sets that extends beyond the conventional approach of simply accumulating data. Rather than prioritizing dataset size, efforts should focus on increasing connectivity among annotated cell types, particularly in sparsely represented regions of the ontology. This connectivity-driven approach amplifies the generalization capabilities of learning architectures by enabling information propagation across hierarchical relationships, thereby allowing models to learn from the structural properties of the label space itself. The practical implication is that carefully curated smaller datasets with high ontological coverage may prove more valuable than larger datasets with redundant or isolated annotations.

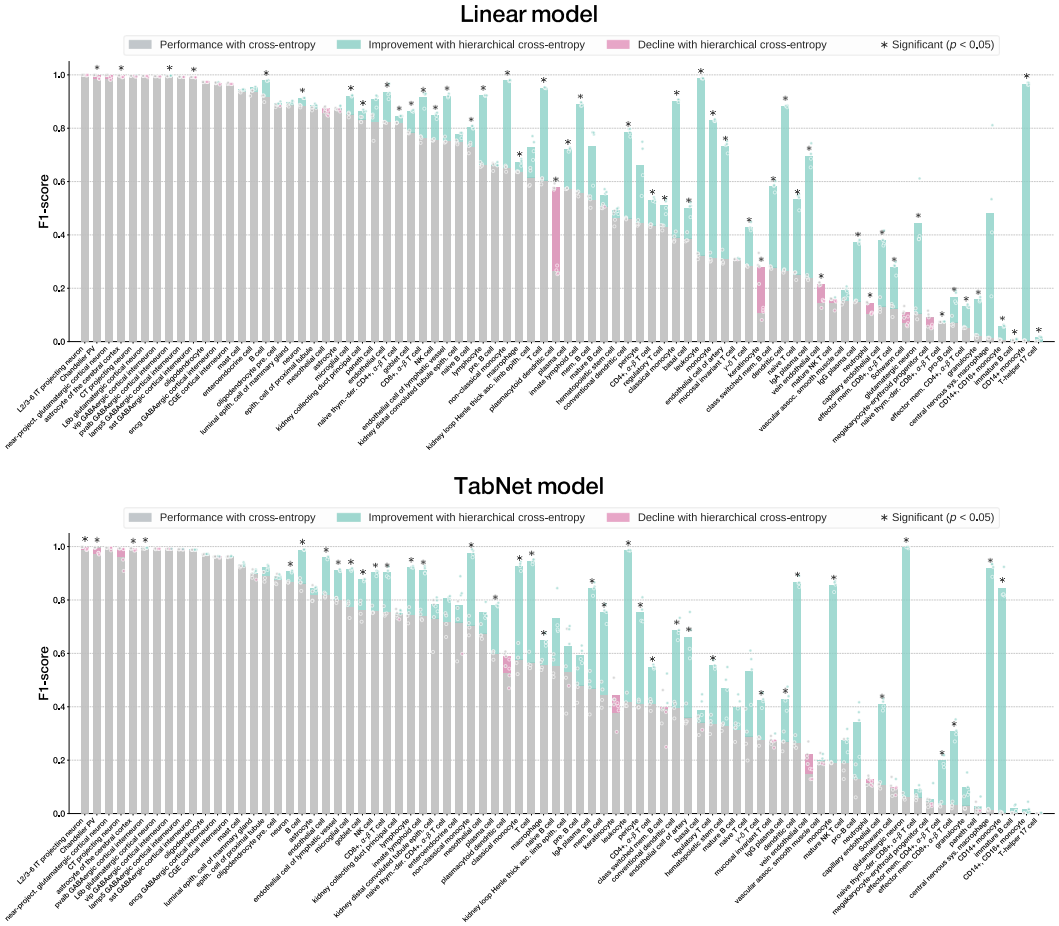


Fig. 3.6 Per-cell type performance changes induced by the hierarchical cross-entropy (HCE) loss strategy for the linear model and TabNet, shown relative to standard cross-entropy. Each dot represents the performance of an individual run (color coding remains the same as in the legend).

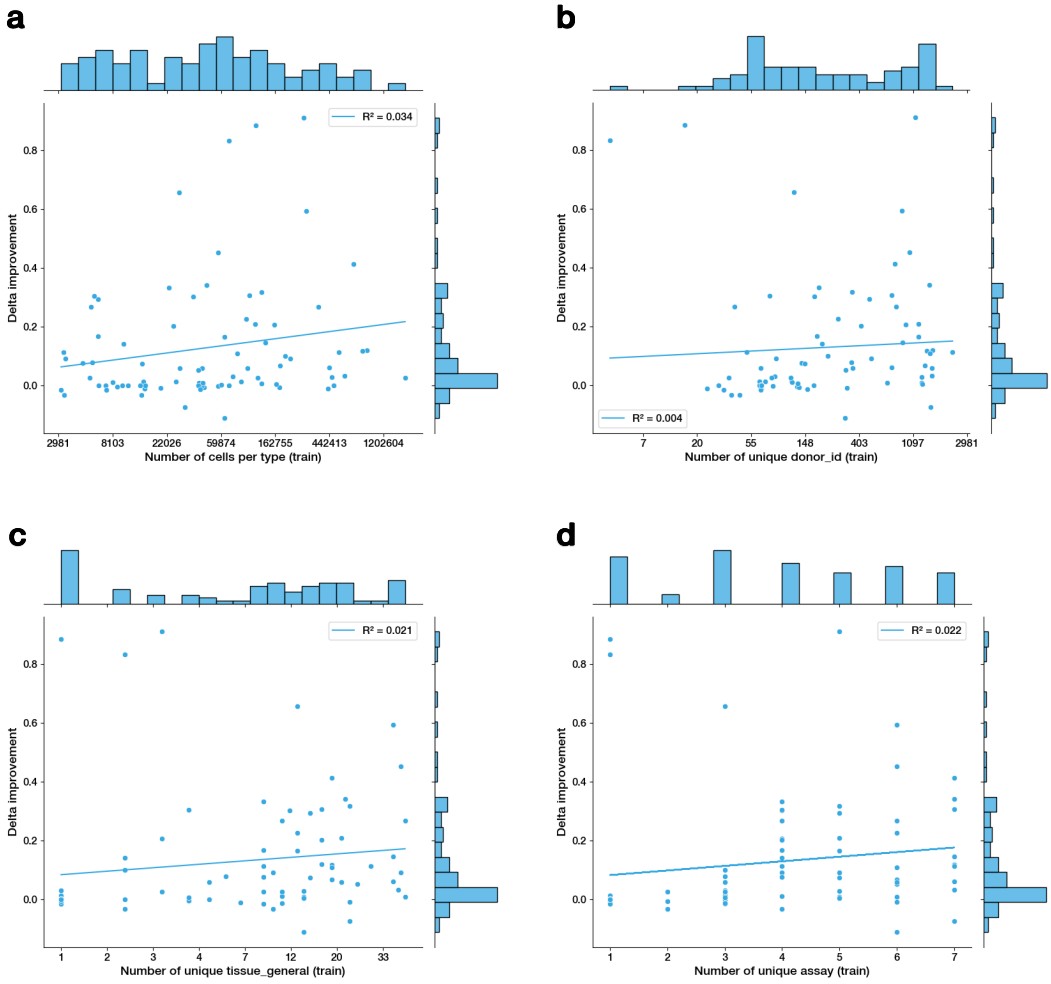


Fig. 3.7 Performance gains from hierarchical training in the multilayer perceptron (MLP) model as a function of several training-set properties. These include: **a** cell type rarity (in log scale), **b** number of donors (in log scale), **c** number of tissues (in log scale), and **d** number of sequencing technologies (in linear scale).

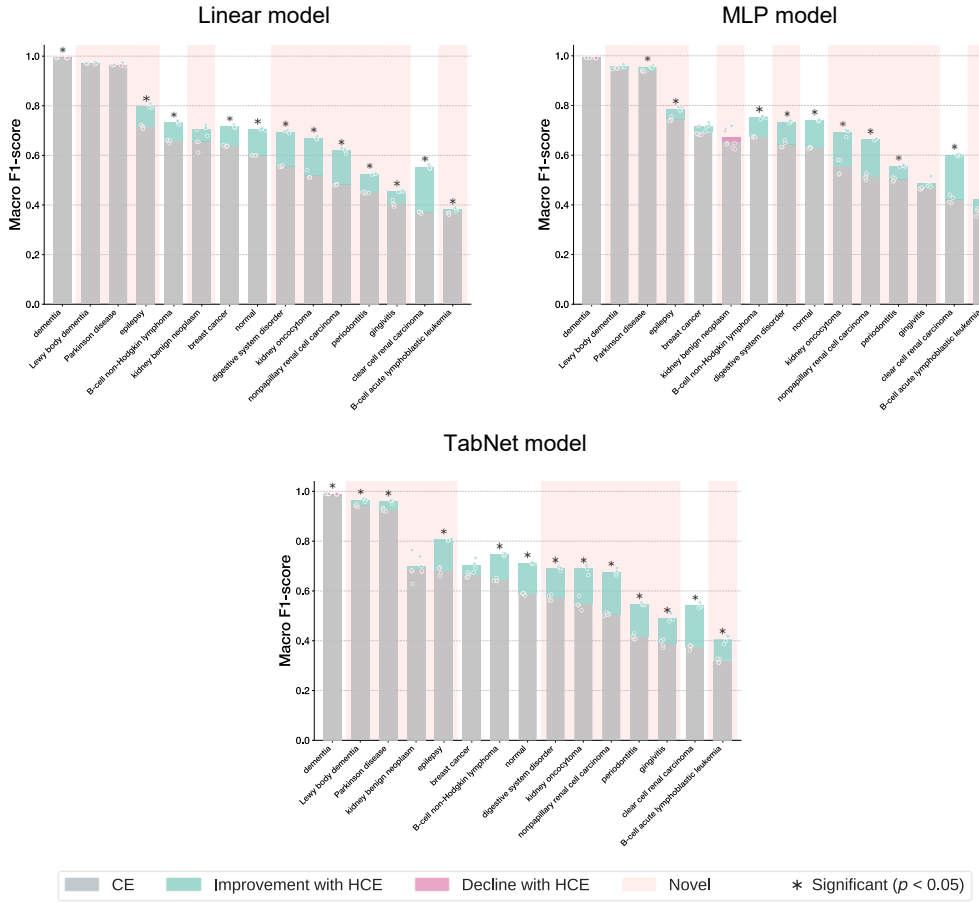


Fig. 3.8 Performance gains from the hierarchical cross-entropy (HCE) loss for different diseases for the linear classifier, multilayer perceptron (MLP), and TabNet. Improvements are measured relative to the same models trained with standard cross-entropy loss. Highlighted in pink are novel diseases in the test set that were not seen in the training set. Each dot represents the performance of an individual run (color coding remains the same as in the legend).

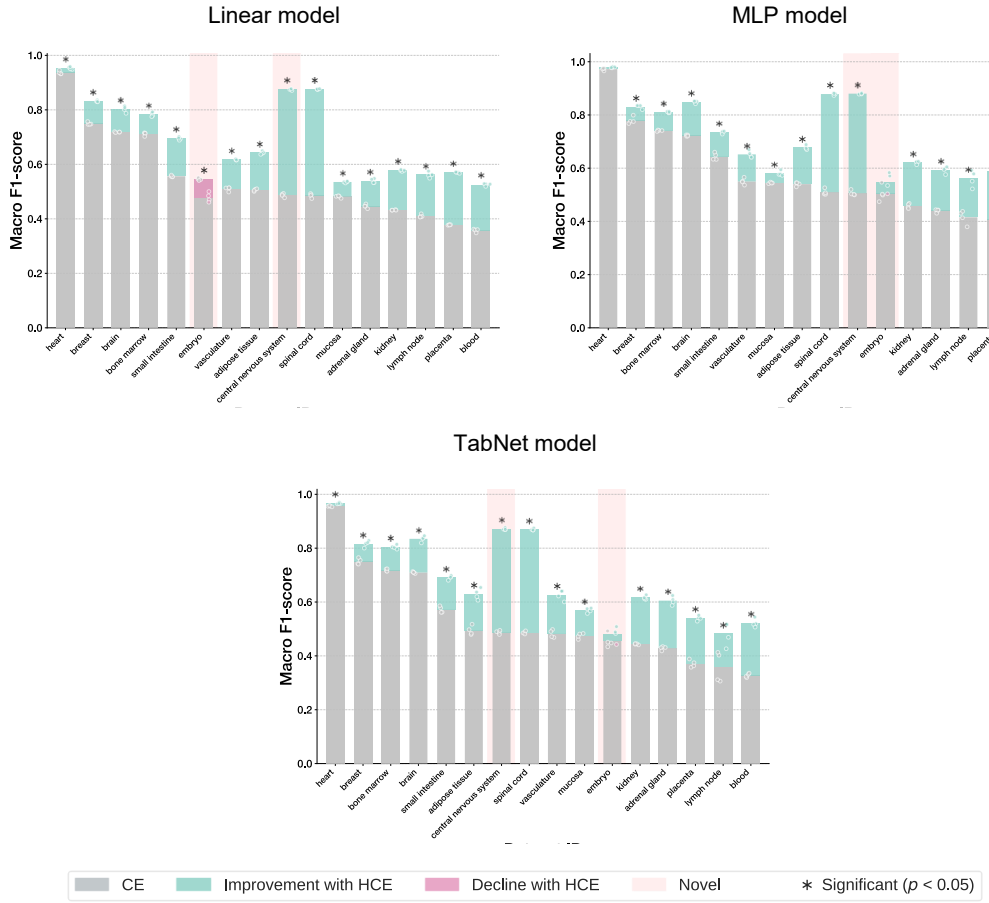


Fig. 3.9 Performance gains from the hierarchical cross-entropy (HCE) loss for different tissues for the linear classifier, multilayer perceptron (MLP), and TabNet. Improvements are measured relative to the same models trained with standard cross-entropy loss. Highlighted in pink are novel tissues in the test set that were not seen in the training set. Each dot represents the performance of an individual run (color coding remains the same as in the legend).

This chapter also exemplifies how biological problems can inspire foundational advances in machine learning methodology. As argued by Uhler, the unique challenges posed by biological data, including hierarchical organization, multi-scale structure, and the availability of expert-curated knowledge bases, present opportunities to develop broadly applicable machine learning innovations [234]. The hierarchical cross-entropy loss represents one such example: while motivated by the specific problem of cell type annotation, it generalizes to any classification task with a structured label space in the form of inclusion relationships, offering a simple drop-in replacement for standard cross-entropy that brings domain knowledge into model training. This broader applicability points to an increasingly important consideration as models are trained on ever-growing single-cell atlases [235, 21, 232]: the need to systematically incorporate biological priors into learning objectives. While this study centers on cell type classification, hierarchically structured losses extend naturally to other prediction tasks with well-defined ontological structures, such as protein function annotation or disease phenotype prediction, representing a promising direction for future work.

# Chapter 4

# Fast Quasi-Random Data Loading for Large-Scale On-Disk Datasets

*The content of this chapter is largely based on the research originally presented in [236].*

Previous chapters have been devoted to theoretical aspects of machine learning and its applications in biology. In this chapter, we shift our focus to a practical challenge that arises when training deep learning models on large-scale datasets. Motivated by the recent surge of large-scale single-cell omics datasets [16–18], we develop a novel data loading framework to overcome the slow random data access patterns that hinder efficient training on disk-resident data. As with the hierarchical cross-entropy loss, along the way we realize the broad applicability of the techniques of this chapter, which can be used in any context where large datasets are stored on disk.

We introduce scDataset[1], a PyTorch IterableDataset designed for efficient and scalable training on large single-cell omics datasets. scDataset operates directly on one or more AnnData files, without format conversion or full in-memory loading, and implements a quasi-random sampling strategy that combines block sampling with batched fetching. Block sampling reduces the number of random disk reads by accessing contiguous chunks, while batched fetching amortizes I/O latency and enables in-memory reshuffling for minibatch diversity. While scDataset was devel-

[1]The source code and documentation for scDataset are publicly available on GitHub at https://github.com/scDataset/scDataset.

oped to address the limitations of AnnData in deep learning applications, its design is broadly applicable to any large-scale dataset stored on disk.

The remainder of this chapter proceeds as follows. Section 4.1 elaborates on the computational challenges posed by large-scale single-cell datasets and the limitations of existing data formats for deep learning workflows. Section 4.2 reviews current solutions, analyzing their trade-offs in terms of performance, storage requirements, and ecosystem compatibility. Section 4.3 details the design of scDataset, including its block sampling and batched fetching strategies that enable efficient quasi-random data access. Finally, in Section 4.4 we present comprehensive benchmarks demonstrating the performance advantages of scDataset over existing data loading solutions.

## 4.1 Motivation

The advent of single-cell omics has transformed molecular biology by enabling the high-throughput measurement of gene expression, chromatin accessibility, and protein abundance at the resolution of individual cells. Among these, single-cell transcriptomics has emerged as the most widely adopted modality, providing genome-wide snapshots of gene expression that reveal cellular identity, state, and function. These technologies have driven major discoveries across cancer biology [237], neurodevelopment [238], and immunology [239], uncovering rare cell types, lineage hierarchies, and dynamic transcriptional programs [240, 241, 16].

As the scale and accessibility of single-cell experiments continue to grow, public datasets have expanded rapidly. As of October 2024, CELLxGENE Discover hosts over 1,550 datasets and more than 93 million unique cells, spanning a wide range of tissues, diseases, and experimental conditions [17]. This resource is primarily observational, capturing the natural diversity of cell states across biological contexts. More recently, the Tahoe-100M dataset marked a new milestone for interventional single-cell datasets, combining both scale and experimental richness: it profiles over 100 million cells across 379 drug perturbations, 50 cancer cell lines, and multiple dosages, totaling almost 60,000 perturbation contexts [18]. Together, these large-scale resources provide a foundation for modeling cell states and biological responses at unprecedented resolution.

To extract structure from these datasets, the field is increasingly turning to deep learning models that can learn expressive representations of cellular states from raw omics data. Recent efforts have focused on building biological foundation models—large, pre-trained architectures designed to generalize across tissues, conditions, and experimental settings [242]. Models such as GeneFormer [20] and scGPT [21] leverage transformer-based architectures to capture gene-gene dependencies, and can be fine-tuned for tasks such as cell type classification, batch correction, and perturbation prediction. This direction has created strong demand for scalable infrastructure to support model training on datasets containing hundreds of millions of cells, with the long-term goal of building in silico "virtual cells" that simulate cellular behavior across modalities and contexts [104].

The AnnData format has become the community standard for storing single-cell omics data, offering efficient handling of sparse matrices, metadata, and annotations [243]. However, its integration with deep learning workflows remains underdeveloped. Standard training algorithms such as stochastic gradient descent (SGD) require diverse, randomly sampled minibatches to ensure stable convergence and generalization [244, 245]. Ensuring this diversity at hundred-million-cell scale typically requires loading the full dataset into memory, which is infeasible for most practitioners. Alternative strategies, such as converting to dense formats or relying on random disk access, suffer from inflated storage costs or low throughput. For instance, AnnLoader, an experimental PyTorch data loader developed by the AnnData team to enable on-disk sampling, achieves only 20 samples/second on the Tahoe-100M dataset, requiring *more than 58 days* for a single training epoch.

From the perspective of this thesis, efficient data loading is not an isolated systems problem, but an enabling technology for structure-aware models. The ontological loss functions, graph-based architectures, and compositional designs advocated in the previous chapters become practically useful only if they can be trained on the full scale of modern single-cell resources, such as CELLxGENE or Tahoe-100M, without resorting to aggressive subsampling or lossy format conversions. Moreover, data loading strategies themselves encode implicit inductive biases: streaming data plate by plate without shuffling, for instance, encourages models to exploit plate- or batch-specific correlations rather than biological structure, while quasi-random sampling restores the stochastic gradients required for robust optimization. scDataset removes a systems bottleneck that would otherwise prevent the systematic exploration and deployment of biologically motivated inductive biases at atlas scale.

## 4.2 Current Solutions

The popularity of the AnnData format stems from its flexible support for sparse matrices, rich metadata, and seamless integration with the Python data science ecosystem [246]. While AnnData supports on-disk access via backing modes, most deep learning tools, including frameworks like scvi-tools [247], often require loading the entire dataset into memory. As modern single-cell atlases span hundreds of millions of cells and multiple terabytes of storage, this is increasingly infeasible. The only native option, *AnnLoader*, preserves format compatibility but suffers from slow throughput and lacks multiprocessing support, making it impractical for large-scale use.

In response, various groups have developed custom data loading pipelines tailored for deep learning. These typically convert AnnData to alternative formats, introducing new technical and interoperability challenges. For example, GeneFormer [20] converts data to *HuggingFace Datasets*, while scTab [248] transforms AnnData into Parquet files for use with NVIDIA Merlin. These conversions often require densifying sparse matrices, significantly increasing storage requirements. Other groups have developed custom storage infrastructures, such as the Zarr-based backend in SCimilarity [249] or the LaminDB-based backend in scDataloader [250], or adopted entirely new formats like TileDB's SOMA in CZ CELLxGENE [17]. A notable tailored solution is *BioNeMo-SCDL*, developed by NVIDIA to support large-scale single-cell training [251]. It introduces a custom format and leverages NumPy memory mapping for efficient loading of the `X` matrix. However, it still requires conversion from AnnData and does not natively handle the full metadata stack, limiting compatibility with the broader analysis ecosystem.

In summary, while AnnData remains central to the field and does support on-disk access in principle, no efficient, standard solution exists for deep learning workflows. Existing alternatives either sacrifice compatibility and flexibility or fall short on performance. Our work addresses this gap by introducing a faster and more flexible solution that operates directly on AnnData files, eliminating format conversion and enabling scalable model training on modern single-cell datasets.

## 4.3 Method

scDataset is designed to enable efficient, randomized data loading from large-scale datasets. It serves as a flexible and extensible interface that connects diverse data backends, such as AnnData and HuggingFace Datasets, to PyTorch's DataLoader. The core technical innovations of scDataset are its *block sampling* and *batched fetching* strategies, which together balance I/O efficiency with minibatch diversity. To accommodate a wide range of data sources and workflows, scDataset is implemented as a PyTorch IterableDataset with a modular architecture. The design centers on four user-configurable functions that govern data retrieval and transformation, enabling seamless integration with arbitrary data backends (see Figure 4.1).

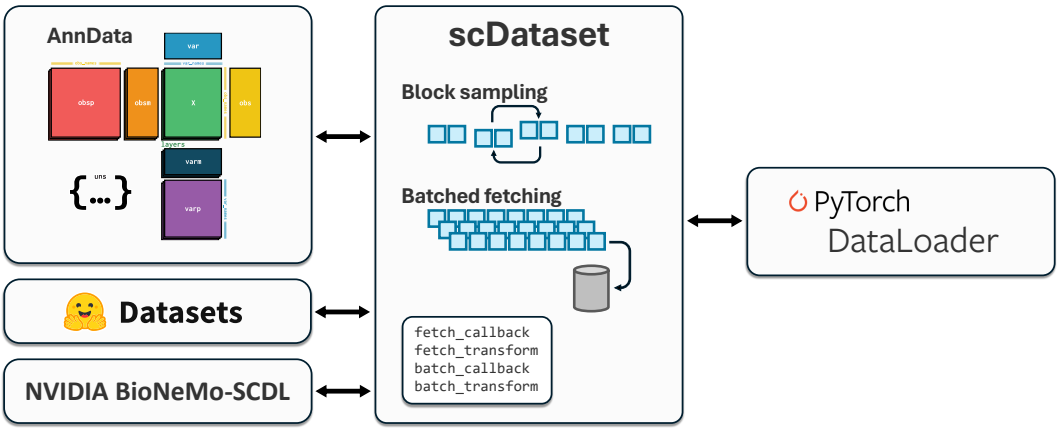


Fig. 4.1 scDataset bridges diverse data backends with PyTorch's DataLoader through a modular interface. Data retrieval is managed by a configurable `fetch_callback`, followed by preprocessing with `fetch_transform` (e.g., sparse-to-dense conversion). Batches are selected using `batch_callback` and further processed with `batch_transform` before being yielded to the training pipeline.

In the following sections, we first discuss the rationale for adopting an iterable-style dataset over the traditional map-style paradigm in PyTorch, and then describe the block sampling and batched fetching strategies in detail.

### 4.3.1 Map-style vs Iterable-style PyTorch Datasets

PyTorch supports two primary paradigms for dataset implementation: *map-style* and *iterable-style* datasets.

**Map-style datasets** implement the `__getitem__` method, allowing retrieval of individual samples by index. Batching is managed by the DataLoader, which assembles minibatches by collecting samples one at a time and merging them via a `collate_fn`. This approach is efficient when the dataset resides in memory, as random access is fast. However, for on-disk datasets, this results in a large number of random I/O operations per minibatch—a significant bottleneck, especially on hard disk drives (HDDs). For instance, SCimilarity employs a map-style dataset, which constrains disk throughput [249].

An exception is the experimental AnnLoader and its underlying AnnCollection dataset, which extend `__getitem__` to accept both single indices and batches of indices. By providing a `batch_sampler` to the DataLoader's `sampler` argument, minibatches can be retrieved in a single call, reducing I/O overhead. While this approach deviates from standard PyTorch API usage, it does improve throughput by minimizing disk accesses.

**Iterable-style datasets**, in contrast, require implementation of the `__iter__` method, returning an iterator over samples. This paradigm offers maximal flexibility, enabling custom sampling and loading strategies. However, it precludes the use of standard PyTorch samplers, which rely on map-style indexing, and necessitates careful handling of multiprocessing within the iterable implementation.

Given the limitations of map-style datasets for large, on-disk single-cell data, we adopt an iterable-style dataset for scDataset. This design empowers us to implement efficient sampling strategies optimized for disk-based access patterns. Our implementation natively supports reading from single or multiple AnnData files without format conversions and integrates custom multiprocessing logic aligned with PyTorch's DataLoader specifications.

### 4.3.2 Block Sampling

Training deep learning models relies on stochastic gradient descent (SGD), where the diversity of each minibatch is crucial for unbiased updates and robust convergence [244]. Achieving this diversity via random sampling is straightforward in-memory, but becomes challenging when datasets are too large to fit in memory and must be accessed from disk.

Disk drives, particularly HDDs, are optimized for reading large contiguous chunks of data, but are inefficient when required to perform frequent, small, non-sequential reads. While solid-state drives (SSDs) mitigate this limitation to some extent, contiguous access remains significantly faster than random access.

To reconcile the need for random sampling with the realities of disk I/O, we introduce a *block sampling* strategy (see Algorithm 1). Instead of sampling each data point independently, we randomly select contiguous blocks of size $b$ (e.g., 8 cells) from the dataset. For a minibatch size $m$ (e.g., $m = 256$) and block size $b = 8$, only $m/b = 32$ random disk reads are required per minibatch, each retrieving a contiguous chunk of $b$ samples. These blocks are then assembled to form the final minibatch.

The choice of block size $b$ governs the trade-off between I/O efficiency and minibatch diversity. Larger blocks improve throughput by reducing the number of random disk operations, but may group together cells with correlated metadata (e.g., from the same tissue or batch), potentially reducing diversity. Conversely, smaller blocks enhance randomness but increase I/O overhead. This parameter can be tuned based on dataset properties and hardware capabilities.

**Algorithm 1:** Block Sampling

**Input :** Dataset size $n$, block size $b$, minibatch size $m$
(where $n$ is a multiple of $b$ and $m$ for simplicity)

**Output :** Sequence of minibatches $\mathscr{M}_0, \mathscr{M}_1, \ldots$

Generate full index array: $I = [0, 1, \ldots, n-1]$;
Split $I$ into $k = \frac{n}{b}$ blocks $[B_0, B_1, \ldots, B_{k-1}]$, where $B_i = [i \cdot b, \ldots, (i+1) \cdot b - 1]$;
Shuffle block order: $[B_{\sigma(0)}, \ldots, B_{\sigma(k-1)}] \leftarrow \text{RandomPermutation}([B_0, \ldots, B_{k-1}])$;
Concatenate shuffled blocks: $I_{\text{shuffled}} \leftarrow B_{\sigma(0)} \| \ldots \| B_{\sigma(k-1)}$;
Split $I_{\text{shuffled}}$ into minibatches $[M_0, \ldots, M_{\frac{n}{m}-1}]$ of size $m$;
**for** *each* $M_i$ **do**
    Load data: $\mathscr{M}_i \leftarrow \text{ReadFromDisk}(M_i)$;
    **yield** $\mathscr{M}_i$;

### 4.3.3 Batched Fetching

While block sampling reduces the number of random disk reads, further improvements in throughput and randomness can be achieved through *batched fetching*. In

this approach, multiple blocks are prefetched into memory in a single I/O operation, forming a buffer that is subsequently reshuffled to construct diverse minibatches.

Batched fetching amortizes the latency of disk access across larger data transfers, which is particularly beneficial on high-latency storage devices. After fetching a batch of blocks, the samples are randomly permuted in memory before being yielded as minibatches, ensuring that each minibatch contains a diverse set of cells.

The detailed procedure for batched fetching is presented in **Algorithm 2**. This algorithm outlines how scDataset preloads blocks, reshuffles their contents, and yields randomized minibatches, balancing I/O efficiency with quasi-random sampling.

**Algorithm 2:** Block Sampling with Batched Fetching

**Input :** Dataset size $n$, block size $b$, minibatch size $m$, fetch factor $f$ (where $n$ is a multiple of $b$ and $m \cdot f$ for simplicity)

**Output :** Sequence of minibatches $\mathscr{M}_0, \mathscr{M}_1, \ldots$

Generate full index array: $I = [0, 1, \ldots, n-1]$;

Split $I$ into $k = \frac{n}{b}$ blocks $[B_0, B_1, \ldots, B_{k-1}]$, where $B_i = [i \cdot b, \ldots, (i+1) \cdot b - 1]$;

Shuffle block order: $[B_{\sigma(0)}, \ldots, B_{\sigma(k-1)}] \leftarrow \text{RandomPermutation}([B_0, \ldots, B_{k-1}])$;

Concatenate shuffled blocks: $I_{\text{shuffled}} \leftarrow B_{\sigma(0)} \| \ldots \| B_{\sigma(k-1)}$;

Split $I_{\text{shuffled}}$ into batches $[F_0, \ldots, F_{\frac{n}{m \cdot f}-1}]$ of size $m \cdot f$;

**for** *each* $F_i$ **do**

  Load data: $\mathscr{F}_i \leftarrow \text{ReadFromDisk}(F_i)$;

  Shuffle $\mathscr{F}_i$ in memory;

  Split $\mathscr{F}_i$ into minibatches $\mathscr{M}_0, \ldots, \mathscr{M}_j$;

  **for** *each* $\mathscr{M}_j$ **do**

    **yield** $\mathscr{M}_j$;

## 4.4 Benchmarks

We benchmarked scDataset on the Tahoe-100M dataset [18], which is available in three formats: AnnData (14 files of approximately 7 million cells each), HuggingFace Datasets, and BioNeMo. The AnnData files (downloadable from the official GitHub[2])

[2] https://github.com/ArcInstitute/arc-virtual-cell-atlas

occupy 314GB on disk. The HuggingFace version[3] requires 1.9TB, and the BioNeMo format, generated using the official conversion script[4], occupies 1.1TB, with peak storage usage of 2.2TB during conversion.

For the HuggingFace dataset, a custom script was used to load the `X` matrix. The BioNeMo dataset stores only the `X` matrix, so handling metadata requires additional custom logic. When applied to AnnData, scDataset yields batches as AnnData objects, preserving all original metadata. In contrast, when used with HuggingFace or BioNeMo, scDataset currently returns only the `X` matrix, as metadata must be managed separately.

All experiments were conducted on an NVIDIA DGX Station with 256GB RAM, an Intel Xeon E5-2698 v4 CPU, and 5TB of SSD storage. Unless otherwise specified, a fixed batch size of 64 was used for all benchmarks.

### 4.4.1 Data Loading Throughput

We evaluated data loading speed by measuring single-core throughput (samples per second). Each data loader was warmed up for 30 seconds, followed by 120 seconds of measurement per condition. We benchmarked scDataset using seven block sizes (1, 2, 4, 8, 16, 32, 64) and seven fetch factors (1, 2, 4, 8, 16, 32, 64) on the AnnData, HuggingFace, and BioNeMo datasets. For AnnData, AnnLoader served as a baseline.

The performance of scDataset depends not only on the sampling strategy but also on how efficiently the storage backend serves data requests. While scDataset delivers requests in an optimal format for batch retrieval, the actual data access pattern, whether batched or as individual queries, is determined by the backend implementation. As a result, improvements from increasing the fetch factor are observed only when the backend supports efficient batched reads. Notably, very large fetch factors can slightly degrade throughput due to the increased computational overhead of in-memory shuffling of large buffers.

Results are presented in Figure 4.2 (AnnData), Figure 4.3 (HuggingFace), and Figure 4.4 (BioNeMo). As expected, scDataset with block size 1 and fetch factor

[3] https://huggingface.co/datasets/tahoebio/Tahoe-100M
[4] https://nvidia.github.io/bionemo-framework/API_reference/bionemo/scdl/scripts/convert_h5ad_to_scdl/

1 matches the baseline performance. On AnnData, increasing both block size and fetch factor yields up to a **48**× speed-up over AnnLoader (block size 64, fetch factor 64). For HuggingFace and BioNeMo, throughput improves with larger block sizes, reaching up to **27**× and **18**× speed-ups, respectively, at block size 64.

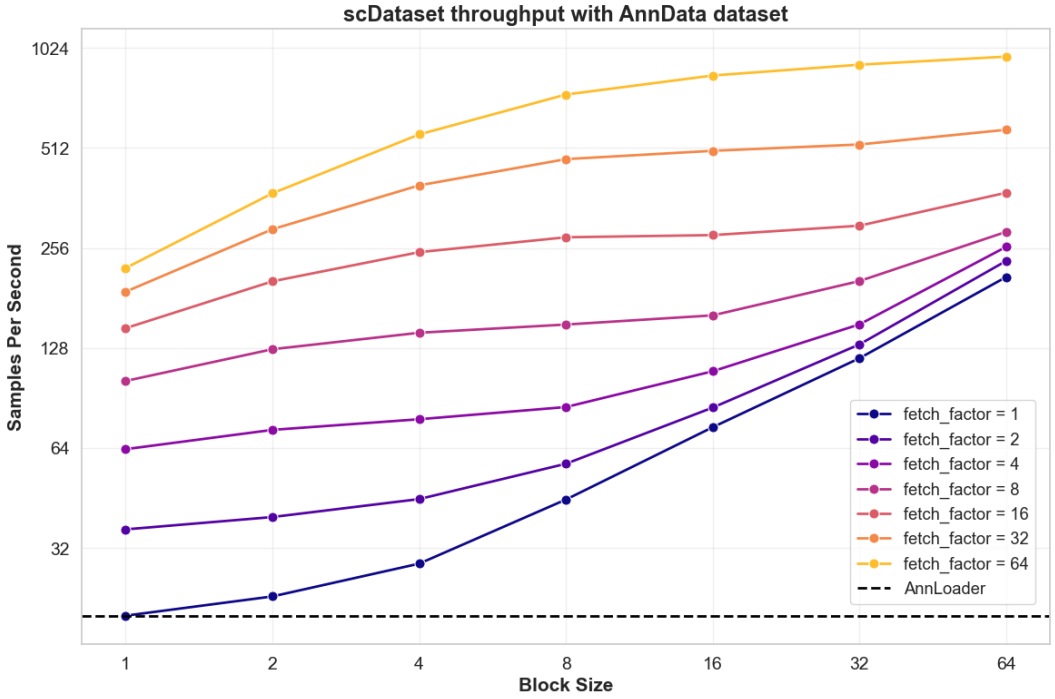


Fig. 4.2 Data loading throughput for scDataset on the AnnData dataset as a function of block size and fetch factor. Throughput (samples/sec) increases substantially with larger block sizes and higher fetch factors, demonstrating that both parameters synergistically improve I/O efficiency. At the largest tested values, scDataset achieves over 48× higher throughput compared to the baseline.

### 4.4.2 Minibatch Diversity

We assessed sampling quality by measuring plate label entropy within minibatches. In the Tahoe-100M dataset, each of the 14 AnnData files corresponds to a unique plate label, and because the dataset is concatenated without prior shuffling, adjacent cells share the same label.

We evaluated scDataset using the same seven block sizes and fetch factors as in the throughput benchmarks. Since minibatch diversity is determined solely by the sampling strategy, this analysis was performed once, independent of backend.

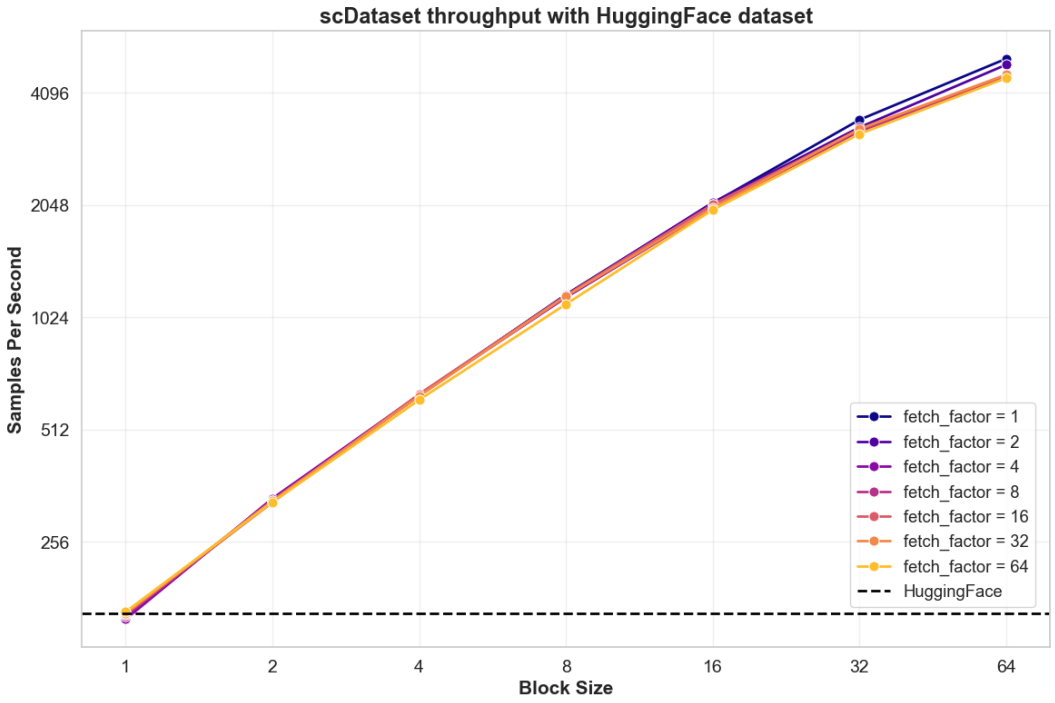


Fig. 4.3 Data loading throughput for scDataset on the HuggingFace dataset as a function of block size and fetch factor. Throughput increases with larger block sizes, but remains unaffected by the fetch factor. At the largest block size, scDataset achieves a 27× speed-up over the baseline.

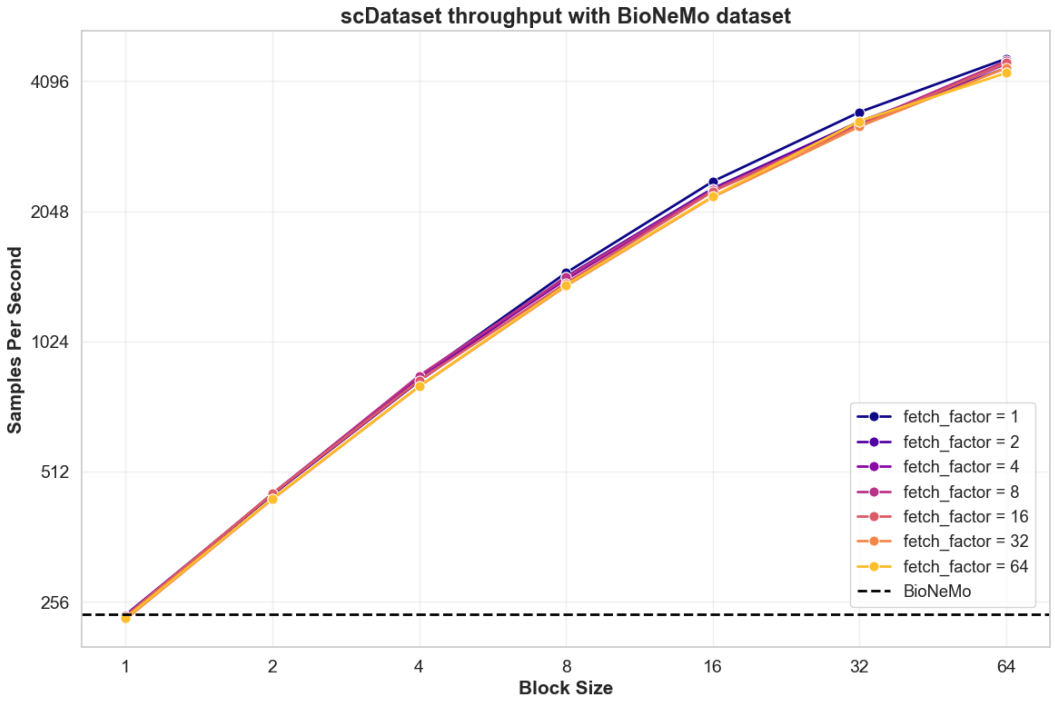


Fig. 4.4 Data loading throughput for scDataset on the BioNeMo dataset as a function of block size and fetch factor. Throughput increases with larger block sizes, but remains unaffected by the fetch factor. At the largest block size, scDataset achieves an 18× speed-up over the baseline.

Both AnnLoader and scDataset with block size 1 and fetch factor 1 achieve random shuffling, yielding an entropy of approximately 3.63. As shown in Figure 4.5, increasing block size reduces entropy, reaching zero at block size 64, while higher fetch factors counteract this effect and help maintain diversity.

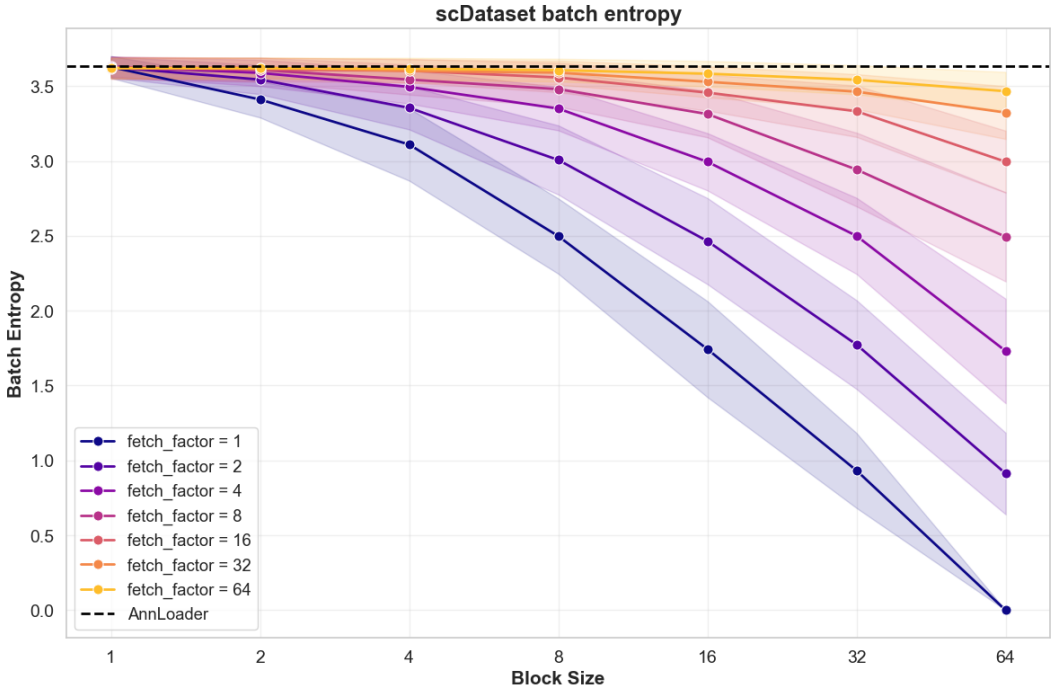


Fig. 4.5 Plate label entropy within minibatches for scDataset as a function of block size and fetch factor. Higher entropy indicates greater minibatch diversity.

### 4.4.3 Scaling Throughput with Multiprocessing

We evaluated the maximum throughput of scDataset on the AnnData dataset using multiprocessing, a feature not natively supported by AnnLoader. The hyperparameter search space is summarized in Table 4.1, with full results in Table 4.2. While no single configuration is universally optimal, we highlight a setting with `block_size`=4, `fetch_factor`=16, and `num_workers`=12, which achieves approximately 2593 samples/sec and maintains an entropy of 3.59, comparable to random sampling. This represents a **129**× speed-up over AnnLoader. Notably, training for one epoch on the full Tahoe-100M dataset would take **over 58 days** with AnnLoader, but **less than 11 hours** with scDataset.

Table 4.1 Hyperparameter search space for throughput experiments with multiprocessing on the AnnData dataset.

| Parameter | Values |
|---|---|
| Block size ($b$) | 4, 8, 16, 32 |
| Fetch factor ($f$) | 4, 8, 16, 32 |
| Number of workers | 8, 12, 16 |

### 4.4.4 Real-world Classification Tasks

To evaluate the practical impact of sampling strategies on model performance, we conducted classification experiments on the Tahoe-100M dataset. We trained independent linear classifiers for four different prediction tasks: cell line classification (50 classes), drug classification (380 classes), mechanism of action classification at broad resolution (4 classes), and mechanism of action classification at fine resolution (27 classes). The mechanism of action labels were provided by the dataset authors.

We compared four data loading strategies: streaming without shuffle, streaming with buffer-based shuffle (analogous to approaches used by HuggingFace Datasets and Ray Dataset for local shuffling), scDataset with block size 4 and fetch factor 16 (the optimal configuration identified in Section 4.4.3), and random sampling with full dataset shuffling. To isolate the effect of sampling strategy from confounding factors such as model selection or hyperparameter tuning, we used a simple linear model architecture across all experiments. Each model was trained for exactly one epoch using the Adam optimizer [252] with learning rate $1 \times 10^{-5}$. The training set comprised plates 1-13 (approximately 94 million cells), while plate 14 (approximately 6.5 million cells) served as the test set. Notably, the test set contained at least one occurrence of every cell line and drug, ensuring representation of all classes despite the plate-based split.

Figure 4.6 presents the macro F1-scores across all four tasks. The results reveal similar performances between streaming without shuffle and buffer-based shuffle, indicating that local shuffling does not significantly enhance model generalization. In contrast, both scDataset (with block sampling and batched fetching) and true random sampling yield similar substantial improvements in macro F1-scores across all tasks. This validates that scDataset achieves performance parity with true random

Table 4.2 Results for throughput experiments with multiprocessing on the AnnData dataset. The experiment highlighted in **bold** corresponds to the configuration referenced in the main text.

| Block size | Fetch factor | Num workers | Samples/sec | Avg. batch entropy | Std. batch entropy |
|---|---|---|---|---|---|
| **4** | 4 | 8 | 597 | 3.51 | 0.11 |
| | | 12 | 882 | 3.49 | 0.11 |
| | | 16 | 1175 | 3.50 | 0.11 |
| | 8 | 8 | 1061 | 3.56 | 0.10 |
| | | 12 | 1626 | 3.57 | 0.09 |
| | | 16 | 2131 | 3.56 | 0.09 |
| | **16** | 8 | 1876 | 3.59 | 0.08 |
| | | **12** | **2593** | **3.59** | **0.09** |
| | | 16 | 2492 | 3.59 | 0.08 |
| | 32 | 8 | 1852 | 3.60 | 0.08 |
| | | 12 | 1768 | 3.61 | 0.08 |
| | | 16 | 1761 | 3.61 | 0.08 |
| 8 | 4 | 8 | 666 | 3.34 | 0.15 |
| | | 12 | 993 | 3.34 | 0.16 |
| | | 16 | 1323 | 3.33 | 0.16 |
| | 8 | 8 | 1181 | 3.48 | 0.12 |
| | | 12 | 1766 | 3.49 | 0.12 |
| | | 16 | 2297 | 3.48 | 0.12 |
| | 16 | 8 | 2093 | 3.55 | 0.09 |
| | | 12 | 2603 | 3.56 | 0.10 |
| | | 16 | 2529 | 3.55 | 0.10 |
| | 32 | 8 | 1873 | 3.59 | 0.08 |
| | | 12 | 1774 | 3.59 | 0.09 |
| | | 16 | 1789 | 3.59 | 0.08 |
| 16 | 4 | 8 | 861 | 3.00 | 0.23 |
| | | 12 | 1308 | 3.00 | 0.23 |
| | | 16 | 1697 | 3.01 | 0.22 |
| | 8 | 8 | 1319 | 3.33 | 0.16 |
| | | 12 | 1940 | 3.32 | 0.16 |
| | | 16 | 2573 | 3.32 | 0.16 |
| | 16 | 8 | 2266 | 3.48 | 0.12 |
| | | 12 | 2612 | 3.48 | 0.12 |
| | | 16 | 2549 | 3.47 | 0.12 |
| | 32 | 8 | 1878 | 3.54 | 0.10 |
| | | 12 | 1834 | 3.55 | 0.10 |
| | | 16 | 1775 | 3.54 | 0.10 |
| 32 | 4 | 8 | 1309 | 2.47 | 0.27 |
| | | 12 | 1932 | 2.47 | 0.27 |
| | | 16 | 2510 | 2.47 | 0.28 |
| | 8 | 8 | 1662 | 3.00 | 0.23 |
| | | 12 | 2493 | 3.00 | 0.22 |
| | | 16 | 3122 | 3.00 | 0.22 |
| | 16 | 8 | 2474 | 3.32 | 0.16 |
| | | 12 | 2570 | 3.32 | 0.16 |
| | | 16 | 2458 | 3.31 | 0.16 |
| | 32 | 8 | 1837 | 3.48 | 0.12 |
| | | 12 | 1740 | 3.46 | 0.13 |
| | | 16 | 1781 | 3.46 | 0.12 |

shuffling while maintaining the computational efficiency demonstrated in previous benchmarks.

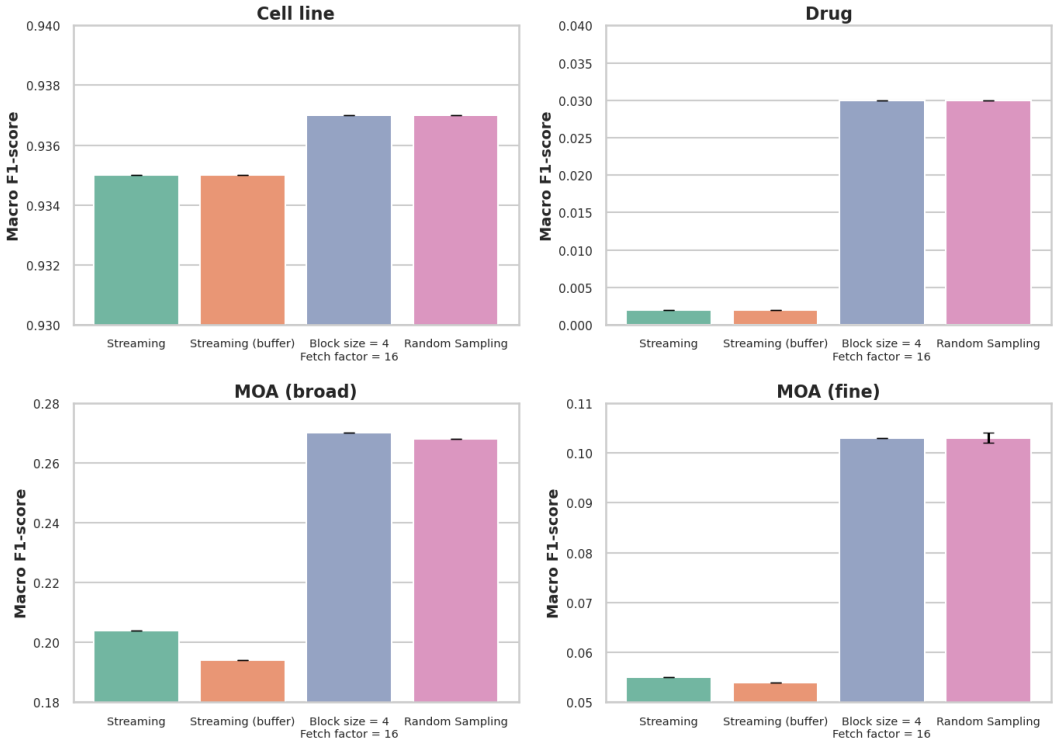


Fig. 4.6 Macro F1-scores for four classification tasks on the Tahoe-100M dataset using different data loading strategies. scDataset with block sampling and batched fetching achieves performance comparable to true random sampling, significantly outperforming streaming without shuffle and buffer-based shuffle.

## 4.5 Discussion

This chapter introduces **scDataset**, a scalable and flexible data loader for training deep learning models on large-scale single-cell omics datasets. By combining block sampling and batched fetching, scDataset enables randomized, high-throughput training directly from disk without requiring format conversion or full in-memory loading. The implementation integrates directly with PyTorch, provides native support for multiprocessing, and consistently delivers substantial speed-ups over existing solutions such as AnnLoader, HuggingFace Datasets, and BioNeMo. By operating directly on formats like AnnData, scDataset enables shuffled training on

commodity hardware and lowers the barrier to large-scale deep learning in single-cell biology.

While scDataset was developed to address specific challenges in single-cell omics, its design principles have broad applicability to any domain requiring efficient training on large-scale on-disk datasets. The combination of block sampling and batched fetching is domain-agnostic and can benefit fields facing similar computational bottlenecks, particularly where datasets cannot fit entirely in memory and random sampling is critical for model convergence. The open-source release and active maintenance of scDataset on GitHub facilitates community contributions and extensions, and the modular architecture allows for easy adaptation to new data backends and formats as they emerge.

Hierarchical losses and graph-based architectures are only as useful as our ability to expose them to the full diversity of cells, tissues, and perturbations captured in current atlases. By enabling high-throughput, quasi-random access to on-disk datasets, scDataset makes it feasible to train models that incorporate ontologies, pathway structure, and compositional decompositions as first-class inductive biases, rather than as small-scale proofs of concept. In this sense, the framework complements the earlier chapters: it supplies the infrastructural bias needed for stochastic optimization to faithfully realize the structural priors that the rest of the thesis advocates.

# Conclusion

This thesis has examined theoretical foundations and practical applications of deep learning in computational biology, demonstrating how mathematical frameworks can illuminate both the behavior of network analysis methods and the representational power of neural architectures, and showing how domain knowledge and computational constraints shape effective machine learning systems for biological data.

Chapter 1 established that closeness, harmonic centrality, and betweenness centrality satisfy rank semi-monotonicity under edge addition in undirected networks. This result guarantees that when an edge is added, at least one endpoint increases in relative importance, resolving a question about the behavior of centrality measures in evolving networks. For biological networks that evolve through formation of protein interactions or regulatory connections, these results provide theoretical grounding for interpreting changes in node importance.

Chapter 2 proved that all efficiently Turing-computable functions exhibit compositional sparsity, offering an explanation for why deep neural networks overcome the curse of dimensionality. By showing that functions computable in polynomial time can be represented as compositions of polynomially many constituent functions, each depending on few variables, the chapter establishes that the success of deep learning reflects structural regularity in target functions. This framework connects to architectural design principles, explaining why convolutional networks exploit local structure and why transformers benefit from Chain-of-Thought reasoning that decomposes complex problems into learnable subcomponents. Several questions remain open, including characterizing which compositional structures can be efficiently discovered from data and determining the minimal supervision needed to avoid exponential sample complexity.

Chapter 3 demonstrated that incorporating known hierarchical structure into training objectives significantly improves model performance in biological domains. The hierarchical cross-entropy loss explicitly encodes cell ontology relationships, leading to improved generalization when models trained on one set of studies are evaluated on newly released datasets. This result validates the principle that domain knowledge should not be an afterthought but a core component of the learning process. The out-of-distribution evaluation framework reflects how cell atlases actually evolve, where new studies are continuously added and must be annotated upon release. The substantial performance drops observed when moving from in-distribution to out-of-distribution evaluation underscore the importance of designing methods that generalize across studies rather than merely fitting training distributions.

Chapter 4 provided infrastructure for training deep learning models on large-scale datasets that cannot fit in memory. The scDataset framework implements quasi-random sampling strategies that balance I/O efficiency with minibatch diversity, achieving speed improvements of up to two orders of magnitude compared to existing solutions. Block sampling and batched fetching enable efficient training directly on disk-resident data without format conversion, lowering the barrier to large-scale deep learning in single-cell biology. The modular architecture allows adaptation to new data backends and formats, and the design principles apply broadly to any domain requiring efficient training on datasets stored on disk.

The convergence of large-scale biological datasets, novel deep learning architectures, and mature domain knowledge creates opportunities for building models that capture biological complexity at unprecedented resolution. This thesis has contributed theoretical frameworks for understanding when such models can succeed, methodological principles for incorporating biological structure, and practical infrastructure for training at scale. Together, these contributions demonstrate how insights from graph theory, machine learning, and biology can be integrated to advance computational approaches to understanding living systems.

# References


[1] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, 2013.

[2] Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

[3] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86 (11):2278–2324, 1998.

[4] Taco Cohen and Max Welling. Group equivariant convolutional networks. *International Conference on Machine Learning*, pages 2990–2999, 2016.

[5] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*, 2016.

[6] William L Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications. *IEEE Data Engineering Bulletin*, 40 (3):52–74, 2017.

[7] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Neural Information Processing Systems*, 2017.

[8] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, August 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-03819-2.

[9] Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, Sebastian W Bodenstein, David A Evans, Chia-Chun Hung, Michael O'Neill, David Reiman, Kathryn Tunyasuvunakool, Zachary Wu, Akvilė Žemgulytė, Eirini Arvaniti, Charles Beattie, Ottavia Bertolli, Alex Bridgland, Alexey Cherepanov, Miles Congreve, Alexander Imani Cowen-Rivers, Andrew Cowie, Michael Figurnov, Fabian B Fuchs, Hannah Gladman, Rishub Jain, Yousuf A. Khan, Caroline M R Low, Kuba Perlin, Anna Potapenko, Pascal Savy, Sukhdeep Singh, Adrian Stecula, Ashok Thillaisundaram, Catherine Tong, Sergei Yakneen, Ellen D. Zhong, Michal Zielinski, Augustin Žídek, Vic-613 tor Bapst, Pushmeet Kohli, Max Jaderberg, Demis Hassabis, and John M. Jumper. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630(8016):493–500, June 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-07487-w.

[10] Leland H Hartwell, John J Hopfield, Stanislas Leibler, and Andrew W Murray. From molecular to modular cell biology. *Nature*, 402(6761):C47–C52, 1999.

[11] Hernan A. Makse, Paolo Boldi, Francesco Sorrentino, and Ian Stewart. Symmetries of living systems: Symmetry fibrations and synchronization in biological networks, 2025.

[12] Robert D. Leclerc. Survival of the sparsest: robust gene networks are parsimonious. *Molecular Systems Biology*, 4:213 – 213, 2008.

[13] Daniel M. Busiello, Samir Suweis, Jorge Hidalgo, and Amos Maritan. Explorability and the origin of network sparsity in living systems. *Scientific Reports*, 7(1):12323, Sep 2017. ISSN 2045-2322. doi: 10.1038/s41598-017-12521-1.

[14] Koen Van den Berge, Hector Roux de Bézieux, Kelly Street, Wouter Saelens, Robrecht Cannoodt, Yvan Saeys, Sandrine Dudoit, and Lieven Clement. Trajectory-based differential expression analysis for single-cell sequencing data. *Nature Communications*, 11(1):1201, Mar 2020. ISSN 2041-1723. doi: 10.1038/s41467-020-14766-3.

[15] Alexander D Diehl, Terrence F Meehan, Yvonne M Bradford, Matthew H Brush, Wasila M Dahdul, David S Dougall, Yongqun He, David Osumi-Sutherland, Alan Ruttenberg, Sirarat Sarntivijai, et al. The cell ontology 2016: enhanced content, modularization, and ontology interoperability. *Journal of Biomedical Semantics*, 7(1):1–10, 2016.

[16] Aviv Regev, Sara A Teichmann, Eric S. Lander, Ido Amit, Christophe Benoist, Ewan Birney, Bernd Bodenmiller, Peter Campbell, Piero Carninci, Menna R. Clatworthy, Hans Clevers, Bart Deplancke, Ian Dunham, James Eberwine, Roland Eils, Wolfgang Enard, Andrew Farmer, Lars Fugger, Berthold Göttgens, Nir Hacohen, Muzlifah A. Haniffa, Martin Hemberg, Seung Kim, Paul Klenerman, Arnold R. Kriegstein, Ed S. Lein, Sten Linnarsson, Emma Lundberg, Joakim Lundeberg, Partha P. Majumder, John C. Marioni,

Miriam Merad, Musa M. Mhlanga, Martijn C. Nawijn, Mihai Netea, Garry P Nolan, Dana Pe'er, Anthony Phillipakis, Chris P. Ponting, Stephen R. Quake, Wolf Reik, Orit Rozenblatt-Rosen, Joshua R Sanes, Rahul Satija, Ton N. Schumacher, Alex N Shalek, Ehud Y. Shapiro, Padmanee Sharma, Jay W. Shin, Oliver Stegle, Michael Stratton, Michael J. T. Stubbington, Fabian J Theis, Matthias Uhlen, Alexander van Oudenaarden, Allon Wagner, Fiona M. Watt, Jonathan S. Weissman, Barbara Wold, Ramnik Xavier, and Nir Yosef. The human cell atlas. *eLife*, 6, 2017.

[17] CZI Cell Science Program, Shibla Abdulla, Brian Aevermann, Pedro Assis, Seve Badajoz, Sidney M Bell, Emanuele Bezzi, Batuhan Cakir, Jim Chaffer, Signe Chambers, J. Michael Cherry, Tiffany Chi, Jennifer Chien, Leah Dorman, Pablo Garcia-Nieto, Nayib Gloria, Mim Hastie, Daniel Hegeman, Jason Hilton, Timmy Huang, Amanda Infeld, Ana-Maria Istrate, Ivana Jelic, Kuni Katsuya, Yang Joon Kim, Karen Liang, Mike Lin, Maximilian Lombardo, Bailey Marshall, Bruce Martin, Fran McDade, Colin Megill, Nikhil Patel, Alexander Predeus, Brian Raymor, Behnam Robatmili, Dave Rogers, Erica Rutherford, Dana Sadgat, Andrew Shin, Corinn Small, Trent Smith, Prathap Sridharan, Alexander Tarashansky, Norbert Tavares, Harley Thomas, Andrew Tolopko, Meghan Urisko, Joyce Yan, Garabet Yeretssian, Jennifer Zamanian, Arathi Mani, Jonah Cool, and Ambrose Carr. Cz cellxgene discover: a single-cell data platform for scalable exploration, analysis and modeling of aggregated data. *Nucleic Acids Research*, 53(D1):D886–D900, 11 2024. ISSN 1362-4962. doi: 10.1093/nar/gkae1142.

[18] Jesse Zhang, Airol A Ubas, Richard de Borja, Valentine Svensson, and Nicole Thomas et al. Tahoe-100m: A giga-scale single-cell perturbation atlas for context-dependent gene function and cellular modeling. *bioRxiv*, 2025. doi: 10.1101/2025.02.20.639398.

[19] Fan Yang, Wenchuan Wang, Fang Wang, Yuan Fang, Duyu Tang, Junzhou Huang, Hui Lu, and Jianhua Yao. scbert as a large-scale pretrained deep language model for cell type annotation of single-cell rna-seq data. *Nature Machine Intelligence*, 4(10):852–866, Oct 2022. ISSN 2522-5839. doi: 10.1038/s42256-022-00534-z.

[20] Christina V. Theodoris, Ling Xiao, Anant Chopra, Mark D. Chaffin, Zeina R. Al Sayed, Matthew C. Hill, Helene Mantineo, Elizabeth M. Brydon, Zexian Zeng, X. Shirley Liu, and Patrick T. Ellinor. Transfer learning enables predictions in network biology. *Nature*, 618(7965):616–624, Jun 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-06139-9.

[21] Haotian Cui, Chloe Wang, Hassaan Maan, Kuan Pang, Fengning Luo, Nan Duan, and Bo Wang. scgpt: toward building a foundation model for single-cell multi-omics using generative ai. *Nature Methods*, 21(8):1470–1480, Aug 2024. ISSN 1548-7105. doi: 10.1038/s41592-024-02201-0.

[22] Yanay Rosen, Yusuf H. Roohani, Ayush Agrawal, Leon Samotoran, Stephen R. Quake, and Jure Leskovec. Universal cell embeddings: A foundation model for cell biology. *bioRxiv*, 2023.

[23] Minsheng Hao, Jing Gong, Xin Zeng, Chiming Liu, Yucheng Guo, Xingyi Cheng, Taifeng Wang, Jianzhu Ma, Xuegong Zhang, and Le Song. Large-scale foundation model on single-cell transcriptomics. *Nature Methods*, 21(8): 1481–1491, Aug 2024. ISSN 1548-7105. doi: 10.1038/s41592-024-02305-7.

[24] Yuansong Zeng, Jiancong Xie, Ningyuan Shangguan, Zhuoyi Wei, Wenbing Li, Yun Su, Shuangyu Yang, Chengyang Zhang, Jinbo Zhang, Nan Fang, Hongyu Zhang, Yutong Lu, Huiying Zhao, Jue Fan, Weijiang Yu, and Yuedong Yang. Cellfm: a large-scale foundation model pre-trained on transcriptomics of 100 million human cells. *Nature Communications*, 16(1):4679, May 2025. ISSN 2041-1723. doi: 10.1038/s41467-025-59926-5.

[25] Constantin Ahlmann-Eltze, Wolfgang Huber, and Simon Anders. Deep-learning-based gene perturbation effect prediction does not yet outperform simple linear baselines. *Nature Methods*, 22(8):1657–1661, Aug 2025. ISSN 1548-7105. doi: 10.1038/s41592-025-02772-6.

[26] Kasia Z. Kedzierska, Lorin Crawford, Ava P. Amini, and Alex X. Lu. Zero-shot evaluation reveals limitations of single-cell foundation models. *Genome Biology*, 26(1):101, Apr 2025. ISSN 1474-760X. doi: 10.1186/s13059-025-03574-x.

[27] Ihab Bendidi, Shawn T. Whitfield, Kian Kenyon-Dean, Hanene Ben Yedder, Yassir El Mesbahi, Emmanuel Noutahi, and Alisandra Kaye Denton. Benchmarking transcriptomics foundation models for perturbation analysis : one PCA still rules them all. In *NeurIPS 2024 Workshop on AI for New Drug Modalities*, 2024.

[28] Romain Lopez, Jeffrey Regier, Michael B. Cole, Michael I. Jordan, and Nir Yosef. Deep generative modeling for single-cell transcriptomics. *Nature Methods*, 15(12):1053–1058, Dec 2018. ISSN 1548-7105. doi: 10.1038/s41592-018-0229-2.

[29] Yan Wu, Esther Wershof, Sebastian M Schmon, Marcel Nassar, Błażej Osiński, Ridvan Eksi, Kun Zhang, and Thore Graepel. Perturbench: Benchmarking machine learning models for cellular perturbation analysis. In *NeurIPS 2024 Workshop on AI for New Drug Modalities*, 2024.

[30] Jared Kaplan, Sam McCandlish, T. J. Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeff Wu, and Dario Amodei. Scaling laws for neural language models. *ArXiv*, abs/2001.08361, 2020.

[31] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican,

George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and L. Sifre. Training compute-optimal large language models. *ArXiv*, abs/2203.15556, 2022.

[32] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856.

[33] Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. Position: Will we run out of data? limits of llm scaling based on human-generated data. In *International Conference on Machine Learning*, 2024.

[34] Ilia Shumailov, Zakhar Shumaylov, Yiren Zhao, Nicolas Papernot, Ross Anderson, and Yarin Gal. Ai models collapse when trained on recursively generated data. *Nature*, 631(8022):755–759, Jul 2024. ISSN 1476-4687. doi: 10.1038/s41586-024-07566-y.

[35] Paolo Boldi, Davide D'Ascenzo, Flavio Furia, and Sebastiano Vigna. Score and rank semi-monotonicity for closeness, betweenness and harmonic centrality. In *Complex Networks & Their Applications XII*. Springer Nature Switzerland, 2024.

[36] Paolo Boldi, Davide D'Ascenzo, Flavio Furia, and Sebastiano Vigna. Score and rank semi-monotonicity for closeness, betweenness, and distance–decay centralities. *Social Network Analysis and Mining*, 14(1):183, Sep 2024. ISSN 1869-5469. doi: 10.1007/s13278-024-01285-y.

[37] Mark Newman. *Networks: An Introduction*. Oxford University Press, 03 2010. ISBN 9780199206650.

[38] Albert-László Barabási. *Network Science*. Cambridge University Press, 2016.

[39] Stanley Wasserman and Katherine Faust. *Social network analysis: Methods and applications*. Cambridge University Press, 1994.

[40] Alexandre Bovet and Hernán A. Makse. Influence of fake news in twitter during the 2016 us presidential election. *Nature Communications*, 10(1):7, Jan 2019. ISSN 2041-1723. doi: 10.1038/s41467-018-07761-2.

[41] Noah E. Friedkin. Theoretical foundations for centrality measures. *American Journal of Sociology*, 96(6):1478–1504, 1991. ISSN 00029602, 15375390.

[42] J L Moreno. *Who shall survive?: A new approach to the problem of human interrelations*. Nervous and Mental Disease Publishing Co, Washington, 1934.

[43] H. Jeong, S. P. Mason, A.-L. Barabási, and Z. N. Oltvai. Lethality and centrality in protein networks. *Nature*, 411(6833):41–42, May 2001. ISSN 1476-4687. doi: 10.1038/35075138.

[44] Elena Zotenko, Julian Mestre, Dianne P O'Leary, and Teresa M Przytycka. Why do hubs in the yeast protein interaction network tend to be essential: reexamining the connection between the network topology and essentiality. *PLoS Computational Biology*, 4(8):e1000140, 2008.

[45] Alex Bavelas. A mathematical model for group structures. *Applied Anthropology*, 7(3):16–30, 1948.

[46] Alex Bavelas. Communication patterns in task-oriented groups. *Journal of the Acoustical Society of America*, 22:725–730, 1950.

[47] Linton C. Freeman. Centrality in social networks conceptual clarification. *Social Networks*, 1:215–239, 1978.

[48] Dirk Koschützki and Falk Schreiber. Centrality analysis methods for biological networks and their application to gene regulatory networks. *Gene Regulation and Systems Biology*, 2:193 – 201, 2008.

[49] Maliackal Poulo Joy, Amy Brock, Donald E. Ingber, and Sui Huang. High-betweenness proteins in the yeast protein interaction network. *Journal of Biomedicine and Biotechnology*, 2005:96 – 103, 2005.

[50] Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, Mar 1953. doi: 10.1007/BF02289026.

[51] Phillip Bonacich. Power and centrality: A family of measures. *American Journal of Sociology*, 92(5):1170–1182, 1987. ISSN 00029602, 15375390.

[52] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

[53] Gert Sabidussi. The centrality index of a graph. *Psychometrika*, 31:581–603, 1966.

[54] Paolo Boldi and Sebastiano Vigna. Axioms for centrality. *Internet Mathematics*, 10:222 – 262, 2014.

[55] James R. Perkins, Ilhem Diboun, Benoit H. Dessailly, Jon G. Lees, and Christine Orengo. Transient protein-protein interactions: Structural, functional, and network properties. *Structure*, 18(10):1233–1243, Oct 2010. ISSN 0969-2126. doi: 10.1016/j.str.2010.08.007.

[56] Kevin A. Murgas, Emil Saucan, and Romeil Sandhu. Hypergraph geometry reflects higher-order dynamics in protein interaction networks. *Scientific Reports*, 12(1):20879, Dec 2022. ISSN 2045-2322. doi: 10.1038/s41598-022-24584-w.

[57] Eric H. Davidson. Emerging properties of animal gene regulatory networks. *Nature*, 468(7326):911–920, Dec 2010. ISSN 1476-4687. doi: 10.1038/nature09645.

[58] Albert-László Barabási, Natali Gulbahce, and Joseph Loscalzo. Network medicine: a network-based approach to human disease. *Nature Reviews Genetics*, 12(1):56–68, 2011.

[59] Kwang-Il Goh, Michael E. Cusick, David Valle, Barton Childs, Marc Vidal, and Albert-László Barabási. The human disease network. *Proceedings of the National Academy of Sciences*, 104(21):8685–8690, 2007. doi: 10.1073/pnas.0701361104.

[60] Andrew L Hopkins. Network pharmacology: the next paradigm in drug discovery. *Nature Chemical Biology*, 4(11):682–690, 2008.

[61] Uri Alon. *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Chapman and Hall/CRC, 2006.

[62] Paolo Boldi, Alessandro Luongo, and Sebastiano Vigna. Rank monotonicity in centrality measures. *Network Science*, 5(4):529–550, 2017. doi: 10.1017/nws.2017.21.

[63] Paolo Boldi, Flavio Furia, and Sebastiano Vigna. Monotonicity in undirected networks. *Network Science*, pages 1–23, 2023. doi: 10.1017/nws.2022.42.

[64] Claude Berge. *Théorie des graphes et ses applications*. Dunod, Paris, France, 1958.

[65] Paolo Boldi and Sebastiano Vigna. Rank monotonicity in centrality measures—Corrigendum. *Network Science*, 7(2):265–268, 2019. doi: 10.1017/nws.2019.11.

[66] Steve Chien, Cynthia Dwork, Ravi Kumar, Daniel Simon, and D. Sivakumar. Link evolution: Analysis and algorithms. *Internet Mathematics*, 1:277–304, 01 2003. doi: 10.1080/15427951.2004.10129090.

[67] Ulrik Brandes, Christian Laußmann, and Jörg Rothe. Voting for centrality. In *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, AAMAS '22, pages 1554–1556, Richland, SC, 2022. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450392136.

[68] Oskar Skibski. Closeness centrality via the Condorcet principle. *Social Networks*, 74:13–18, 2023. ISSN 0378-8733. doi: https://doi.org/10.1016/j.socnet.2023.01.005. URL https://www.sciencedirect.com/science/article/pii/S0378873323000059.

[69] Roger C. Entringer, Douglas E. Jackson, and D.A. Snyder. Distance in graphs. *Czechoslovak Mathematical Journal*, 26(2):283–296, 1976.

[70] G. Kishi. On centrality functions of a graph. In N. Saito and T. Nishizeki, editors, *Graph Theory and Algorithms*, pages 45–52, Berlin, Heidelberg, 1981. Springer Berlin Heidelberg. ISBN 978-3-540-38661-2.

[71] Murray A. Beauchamp. An improved index of centrality. *Behavioral Science*, 10(2):161–163, 1965.

[72] Edith Cohen, Daniel Delling, Thomas Pajor, and Renato F. Werneck. Computing classic closeness centrality, at scale. In *Proceedings of the Second ACM Conference on Online Social Networks*, pages 37–50, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450331982.

[73] Chavdar Dangalchev. Residual closeness in networks. *Physica A: Statistical Mechanics and its Applications*, 365(2):556–564, 2006.

[74] Matthew O. Jackson. *Social and Economic Networks*, volume 3. Princeton university press, 2008.

[75] Chauncy D. Harris. The market as a factor in the localization of industry in the United States. *Annals of the Association of American Geographers*, 44(4): 315–348, 1954.

[76] Raj Kumar Pan and Jari Saramäki. Path lengths, correlations, and centrality in temporal networks. *Phys. Rev. E*, 84(1):016105, 2011.

[77] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics*. Addison–Wesley, second edition, 1994. ISBN ISBN 0-201-55802-5.

[78] Paolo Boldi, Davide D'Ascenzo, Flavio Furia, and Sebastiano Vigna. Score and rank semi-monotonicity for closeness, betweenness and harmonic centrality. In Hocine Cherifi, Luis M. Rocha, Chantal Cherifi, and Murat Donduran, editors, *Complex Networks & Their Applications XII*, number 1143 in Studies in Computational Intelligence, pages 102–113. Springer Nature Switzerland, 2024.

[79] Jac M. Anthonisse. The rush in a directed graph. *Journal of Computational Physics*, pages 1–10, 1971.

[80] Linton Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40:35–41, 03 1977. doi: 10.2307/3033543.

[81] Elisabetta Bergamini, Pierluigi Crescenzi, Gianlorenzo D'Angelo, Henning Meyerhenke, Lorenzo Severini, and Yllka Velaj. Improving the betweenness centrality of a node by adding links. *Journal of Experimental Algorithmics (JEA)*, 23:1–32, 2018.

[82] Marc Vidal, Michael E. Cusick, and Albert-László Barabási. Interactome networks and human disease. *Cell*, 144(6):986–998, Mar 2011. ISSN 0092-8674. doi: 10.1016/j.cell.2011.02.016.

[83] Pourya Naderi Yeganeh, Christine Richardson, Erik Saule, Ann Loraine, and M. Taghi Mostafavi. Revisiting the use of graph centrality models in biological pathway analysis. *BioData Mining*, 13(1):5, Jun 2020. ISSN 1756-0381. doi: 10.1186/s13040-020-00214-x.

[84] Ali Salehzadeh-Yazdi and Marc-Thorsten Hütt. Assessing the impact of sampling bias on node centralities in synthetic and biological networks. *npj Systems Biology and Applications*, 11(1):47, May 2025. ISSN 2056-7189. doi: 10.1038/s41540-025-00526-w.

[85] Sepideh Sadegh, James Skelton, Elisa Anastasi, Judith Bernett, David B. Blumenthal, Gihanna Galindez, Marisol Salgado-Albarrán, Olga Lazareva, Keith Flanagan, Simon Cockell, Cristian Nogales, Ana I. Casas, Harald H. H. W. Schmidt, Jan Baumbach, Anil Wipat, and Tim Kacprowski. Network medicine for disease module identification and drug repurposing with the nedrex platform. *Nature Communications*, 12(1):6848, Nov 2021. ISSN 2041-1723. doi: 10.1038/s41467-021-27138-2.

[86] Eric de Silva, Thomas Thorne, Piers Ingram, Ino Agrafioti, Jonathan Swire, Carsten Wiuf, and Michael PH Stumpf. The effects of incomplete protein interaction data on structural and evolutionary inferences. *BMC Biology*, 4(1): 39, Nov 2006. ISSN 1741-7007. doi: 10.1186/1741-7007-4-39.

[87] Thomas Rolland, Murat Taşan, Benoit Charloteaux, Samuel J. Pevzner, Quan Zhong, Nidhi Sahni, Song Yi, Irma Lemmens, Celia Fontanillo, Roberto Mosca, Atanas Kamburov, Susan D. Ghiassian, Xinping Yang, Lila Ghamsari, Dawit Balcha, Bridget E. Begg, Pascal Braun, Marc Brehme, Martin P. Broly, Anne-Ruxandra Carvunis, Dan Convery-Zupan, Roser Corominas, Jasmin Coulombe-Huntington, Elizabeth Dann, Matija Dreze, Amélie Dricot, Changyu Fan, Eric Franzosa, Fana Gebreab, Bryan J. Gutierrez, Madeleine F. Hardy, Mike Jin, Shuli Kang, Ruth Kiros, Guan Ning Lin, Katja Luck, Andrew MacWilliams, Jörg Menche, Ryan R. Murray, Alexandre Palagi, Matthew M. Poulin, Xavier Rambout, John Rasla, Patrick Reichert, Viviana Romero, Elien Ruyssinck, Julie M. Sahalie, Annemarie Scholz, Akash A. Shah, Amitabh Sharma, Yun Shen, Kerstin Spirohn, Stanley Tam, Alexander O. Tejeda, Shelly A. Wanamaker, Jean-Claude Twizere, Kerwin Vega, Jennifer Walsh, Michael E. Cusick, Yu Xia, Albert-László Barabási, Lilia M. Iakoucheva, Patrick Aloy, Javier De Las Rivas, Jan Tavernier, Michael A. Calderwood, David E. Hill, Tong Hao, Frederick P. Roth, and Marc Vidal. A proteome-scale

map of the human interactome network. *Cell*, 159(5):1212–1226, Nov 2014. ISSN 0092-8674. doi: 10.1016/j.cell.2014.10.050.

[88] Marc Vidal. How much of the human protein interactome remains to be mapped? *Science Signaling*, 9(427):eg7–eg7, 2016. doi: 10.1126/scisignal.aaf6030.

[89] Muhammed A. Yıldırım, Kwang-Il Goh, Michael E. Cusick, Albert-László Barabási, and Marc Vidal. Drug—target network. *Nature Biotechnology*, 25 (10):1119–1126, Oct 2007. ISSN 1546-1696. doi: 10.1038/nbt1338.

[90] Emre Guney, Jörg Menche, Marc Vidal, and Albert-László Barábasi. Network-based in silico drug efficacy screening. *Nature Communications*, 7(1):10331, Feb 2016. ISSN 2041-1723. doi: 10.1038/ncomms10331.

[91] Feixiong Cheng, Rishi J. Desai, Diane E. Handy, Ruisheng Wang, Sebastian Schneeweiss, Albert-László Barabási, and Joseph Loscalzo. Network-based approach to prediction and population-based validation of in silico drug repurposing. *Nature Communications*, 9(1):2691, Jul 2018. ISSN 2041-1723. doi: 10.1038/s41467-018-05116-5.

[92] Konstantin Voevodski, Shang-Hua Teng, and Yu Xia. Spectral affinity in protein networks. *BMC Systems Biology*, 3:112 – 112, 2009.

[93] José Lages, Dima L. Shepelyansky, and Andrei Yu. Zinovyev. Inferring hidden causal relations between pathway members using reduced google matrix of directed biological networks. *PLoS ONE*, 13, 2017.

[94] Louis-Pascal Xhonneux, Meng Qu, and Jian Tang. Continuous graph neural networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 10432–10441. PMLR, 13–18 Jul 2020.

[95] Francesco Di Giovanni, James Rowbottom, Benjamin Paul Chamberlain, Thomas Markovich, and Michael M. Bronstein. Understanding convolution on graphs via energies. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856.

[96] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. Representation learning on graphs with jumping knowledge networks. In *Proceedings of the 35th International Conference on Machine Learning*, PMLR.

[97] Jiaxuan You, Jure Leskovec, Kaiming He, and Saining Xie. Graph structure of neural networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 10881–10891. PMLR, 13–18 Jul 2020.

[98] Johannes Gasteiger, Aleksandar Bojchevski, and Stephan Günnemann. Predict then propagate: Graph neural networks meet personalized pagerank. In *International Conference on Learning Representations (ICLR)*, 2019.

[99] Aleksandar Bojchevski, Johannes Gasteiger, Bryan Perozzi, Amol Kapoor, Martin Blais, Benedek Rózemberczki, Michal Lukasik, and Stephan Günnemann. Scaling graph neural networks with approximate pagerank. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2464–2473. Association for Computing Machinery, 2020. ISBN 9781450379984. doi: 10.1145/3394486.3403296.

[100] Yusuf Roohani, Kexin Huang, and Jure Leskovec. Predicting transcriptional outcomes of novel multigene perturbations with gears. *Nature Biotechnology*, 42(6):927–935, Jun 2024. ISSN 1546-1696. doi: 10.1038/s41587-023-01905-6.

[101] Guadalupe Gonzalez, Xiang Lin, Isuru Herath, Kirill Veselkov, Michael Bronstein, and Marinka Zitnik. Combinatorial prediction of therapeutic perturbations using causally inspired neural networks. *Nature Biomedical Engineering*, Sep 2025. ISSN 2157-846X. doi: 10.1038/s41551-025-01481-x.

[102] David A. Danhofer, Davide D'Ascenzo, Rafael Dubach, and Tomaso A Poggio. Position: A theory of deep learning must include compositional sparsity. In *Forty-second International Conference on Machine Learning Position Paper Track*, 2025.

[103] Mauro DiNuzzo. How artificial intelligence enables modeling and simulation of biological networks to accelerate drug discovery. In *Frontiers in Drug Discovery*, 2022.

[104] Charlotte Bunne, Yusuf Roohani, Yanay Rosen, Ankit Gupta, Xikun Zhang, Marcel Roed, Theo Alexandrov, Mohammed AlQuraishi, Patricia Brennan, Daniel B. Burkhardt, Andrea Califano, Jonah Cool, Abby F. Dernburg, Kirsty Ewing, Emily B. Fox, Matthias Haury, Amy E. Herr, Eric Horvitz, Patrick D. Hsu, Viren Jain, Gregory R. Johnson, Thomas Kalil, David R. Kelley, Shana O. Kelley, Anna Kreshuk, Tim Mitchison, Stephani Otte, Jay Shendure, Nicholas J. Sofroniew, Fabian Theis, Christina V. Theodoris, Srigokul Upadhyayula, Marc Valer, Bo Wang, Eric Xing, Serena Yeung-Levy, Marinka Zitnik, Theofanis Karaletsos, Aviv Regev, Emma Lundberg, Jure Leskovec, and Stephen R. Quake. How to build the virtual cell with artificial intelligence: Priorities and opportunities. *Cell*, 187(25):7045–7063, Dec 2024. ISSN 0092-8674. doi: 10.1016/j.cell.2024.11.015.

[105] Allen Newell and Herbert A. Simon. Computer science as empirical inquiry: symbols and search. *Commun. ACM*, 19(3):113–126, March 1976. ISSN 0001-0782. doi: 10.1145/360018.360022.

[106] Frederick Hayes-Roth, Donald A. Waterman, and Douglas B. Lenat. *Building expert systems*. Addison-Wesley Longman Publishing Co., Inc., USA, 1983. ISBN 9780201106862.

[107] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60:84 – 90, 2012.

[108] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. pages 770–778, 2016.

[109] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587): 484–489, Jan 2016. ISSN 1476-4687. doi: 10.1038/nature16961.

[110] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018. doi: 10.1126/science.aar6404.

[111] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. 33:1877–1901, 2020.

[112] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, DeepMind Nand, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

[113] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[114] OpenAI (2023). Gpt-4 technical report. 2023.

[115] Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, Alhussein Fawzi, Josh Grochow, Andrea Lodi, Jean-Baptiste Mouret, Talia Ringer, and Tao Yu. Mathematical discoveries from program search with large language models. *Nature*, 625:468 – 475, 2023.

[116] Trieu H. Trinh, Yuhuai Wu, Quoc V. Le, He He, and Thang Luong. Solving olympiad geometry without human demonstrations. *Nature*, 625(7995):476–482, January 2024. ISSN 1476-4687. doi: 10.1038/s41586-023-06747-5.

[117] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, ..., and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025.

[118] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023. doi: 10.1126/science.ade2574.

[119] Jeremy Wohlwend, Gabriele Corso, Saro Passaro, Mateo Reveiz, Ken Leidal, Wojtek Swiderski, Tally Portnoi, Itamar Chinn, Jacob Silterra, Tommi Jaakkola, and Regina Barzilay. Boltz-1 democratizing biomolecular interaction modeling. *bioRxiv*, 2024. doi: 10.1101/2024.11.19.624167.

[120] Saro Passaro, Gabriele Corso, Jeremy Wohlwend, Mateo Reveiz, Stephan Thaler, Vignesh Ram Somnath, Noah Getz, Tally Portnoi, Julien Roy, Hannes Stark, David Kwabi-Addo, Dominique Beaini, Tommi Jaakkola, and Regina Barzilay. Boltz-2: Towards accurate and efficient binding affinity prediction. *bioRxiv*, 2025. doi: 10.1101/2025.06.14.659707.

[121] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, NY, 2000. ISBN 9781441931603 9781475732641. doi: 10.1007/978-1-4757-3264-1.

[122] Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations*. Cambridge University Press, 1999.

[123] David Donoho. High-dimensional data analysis: The curses and blessings of dimensionality. pages 1–32.

[124] Sergios Theodoridis and Konstantinos D. Koutroumbas. Pattern recognition, fourth edition. 2008.

[125] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*, 2017.

[126] Behnam Neyshabur, Srinadh Bhojanapalli, David Mcallester, and Nati Srebro. Exploring generalization in deep learning. 30, 2017.

[127] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[128] Hrushikesh Mhaskar, Qianli Liao, and Tomaso Poggio. Learning real and boolean functions: When is deep better than shallow? CBMM Memo #45, arXiv preprint, 2016.

[129] Tomaso Poggio, Hrushikesh Mhaskar, Lorenzo Rosasco, Brando Miranda, and Qianli Liao. Why and when can deep-but not shallow-networks avoid the curse of dimensionality: A review. *International Journal of Automation and Computing*, 14(5):503–519, October 2017. ISSN 1751-8520.

[130] Benedikt Bauer and Michael Kohler. On deep learning as a remedy for the curse of dimensionality in nonparametric regression. *The Annals of Statistics*, 47(4):2261–2285, August 2019. ISSN 0090-5364, 2168-8966.

[131] Johannes Schmidt-Hieber. Nonparametric regression using deep neural networks with ReLU activation function. *The Annals of Statistics*, 48(4):1875–1897, August 2020. ISSN 0090-5364, 2168-8966.

[132] Tomaso Poggio and Maia Fraser. Compositional sparsity of learnable functions. *Bulletin of the American Mathematical Society*, 61(3):438–456, 2024. doi: 10.1090/bull/1820.

[133] Herbert A Simon. The architecture of complexity. *Proceedings of the American Philosophical Society*, 106(6):467–482, 1962.

[134] Günter P Wagner, Mihaela Pavlicev, and James M Cheverud. The road to modularity. *Nature Reviews Genetics*, 8(12):921–931, 2007.

[135] Philip W. Anderson. More is different. *Science*, 177(4047):393–396, 1972. doi: 10.1126/science.177.4047.393.

[136] Matthew Bashton and Cyrus Chothia. The generation of new protein functions by the combination of domains. *Structure*, 15(1):85–99, Jan 2007. ISSN 0969-2126. doi: 10.1016/j.str.2006.11.009.

[137] Eric H. Davidson. Emerging properties of animal gene regulatory networks. *Nature*, 468(7326):911–920, Dec 2010. ISSN 1476-4687. doi: 10.1038/nature09645.

[138] Jason William Nichol and Ali Khademhosseini. Modular tissue engineering: Engineering biological tissues from the bottom up. *Soft matter*, 5 7:1312–1319, 2009.

[139] Cyrus Chothia and Arthur M. Lesk. The evolution of protein structures. *Cold Spring Harbor Symposia on Quantitative Biology*, 52:399–405, 1987.

[140] Andreas Wagner. The origins of evolutionary innovations: a theory of transformative change in living systems. 2011.

[141] D. L. Parnas. On the criteria to be used in decomposing systems into modules. *Commun. ACM*, 15(12):1053–1058, December 1972. ISSN 0001-0782. doi: 10.1145/361598.361623.

[142] Victoria Fromkin, Robert Rodman, Nina M. Hyams, Peter Collins, Mengistu Amberber, and Felicity Cox. *An introduction to language*. Cengage Learning, 7th edition, 2012. ISBN 9780170212984. Australia and New Zealand edition.

[143] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2009.

[144] Vladimir N Vapnik. *The Nature of Statistical Learning Theory*. Springer, 2013.

[145] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning (2nd Ed.)*. Springer, 2023.

[146] Pierfrancesco Beneventano, Patrick Cheridito, Robin Graeber, Arnulf Jentzen, and Benno Kuckuck. Deep neural network approximation theory for high-dimensional functions, 2021.

[147] Tomaso Poggio. On efficiently computable functions, deep networks and sparse compositionality. CBMM Memo #156, arXiv preprint, 2025.

[148] S. Arora and B. Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.

[149] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, August 1986. ISSN 0004-5411. doi: 10.1145/6490.6503.

[150] Yishay Mansour. *Learning Boolean Functions via the Fourier Transform*, pages 391–424. Springer US, Boston, MA, 1994. ISBN 978-1-4615-2696-4. doi: 10.1007/978-1-4615-2696-4_11.

[151] Emmanuel Abbe, Enric Boix-Adserà, Matthew Brennan, Guy Bresler, and Dheeraj M. Nagaraj. The staircase property: How hierarchical structure can guide deep learning. *ArXiv*, abs/2108.10573, 2021.

[152] Sahand N. Negahban and Devavrat Shah. Learning sparse boolean polynomials. *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 2032–2036, 2012.

[153] Michael Kohler and Sophie Langer. On the rate of convergence of fully connected deep neural network regression estimates. *The Annals of Statistics*, 2021.

[154] Wolfgang Dahmen. Compositional Sparsity, Approximation Classes, and Parametric Transport Equations, June 2023. arXiv:2207.06128.

[155] Francesco Cagnetta, Leonardo Petrini, Umberto M. Tomasini, Alessandro Favero, and Matthieu Wyart. How Deep Neural Networks Learn Compositional Data: The Random Hierarchy Model. *Physical Review X*, 14(3):031001, July 2024. ISSN 2160-3308. doi: 10.1103/PhysRevX.14.031001.

[156] Peter Stobbe and Andreas Krause. Learning fourier sparse set functions. In *International Conference on Artificial Intelligence and Statistics*, 2012.

[157] Mengjia Xu, Akshay Rangamani, Andrzej Banburski, Qianli Liao, Tomer, Galanti, and Tomaso A. Poggio. Dynamics in deep classifiers trained with the square loss: Normalization, low rank, neural collapse, and generalization bounds. *Research*, 6, 2023.

[158] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A.-L. Barabási. Hierarchical organization of modularity in metabolic networks. *Science*, 297 (5586):1551–1555, 2002. doi: 10.1126/science.1073374.

[159] Albert-László Barabási and Zoltán N. Oltvai. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics*, 5(2):101–113, Feb 2004. ISSN 1471-0064. doi: 10.1038/nrg1272.

[160] Uri Alon. Network motifs: theory and experimental approaches. *Nature Reviews Genetics*, 8(6):450–461, Jun 2007. ISSN 1471-0064. doi: 10.1038/nrg2102.

[161] Claus Kadelka, Matthew Wheeler, Alan Veliz-Cuba, David Murrugarra, and Reinhard Laubenbacher. Modularity of biological systems: a link between structure and function. *Journal of The Royal Society Interface*, 20(207): 20230505, 10 2023. ISSN 1742-5689. doi: 10.1098/rsif.2023.0505.

[162] Matthew Aguirre, Jeffrey P. Spence, Guy Sella, and Jonathan K. Pritchard. Gene regulatory network structure informs the distribution of perturbation effects. *PLOS Computational Biology*, 21(9):e1013387, September 2025. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1013387.

[163] Boyoung Shin and Ellen V. Rothenberg. Multi-modular structure of the gene regulatory network for specification and commitment of murine t cells. *Frontiers in Immunology*, 14, 2023. ISSN 1664-3224. doi: 10.3389/fimmu.2023.1108368.

[164] Teng Ma and Jianxin Wang. Graphpath: a graph attention model for molecular stratification with interpretability based on the pathway–pathway interaction network. *Bioinformatics*, 40(4):btae165, 03 2024. ISSN 1367-4811. doi: 10.1093/bioinformatics/btae165.

[165] Christian W. Omlin and Sean Snyders. Inductive bias strength in knowledge-based neural networks: application to magnetic resonance spectroscopy of breast tissues. *Artificial Intelligence in Medicine*, 28(2):121–140, 2003. ISSN 0933-3657. Knowledge-based Neurocomputing in Medicine.

[166] Kenji Kawaguchi. Deep learning without poor local minima. *Advances in Neural Information Processing Systems (NeurIPS)*, 2016.

[167] Johanni Brea and Wulfram Gerstner. Weight space symmetry in deep networks gives rise to permutation saddles, connected by error valleys of non-interfering solutions, 2019.

[168] Quynh N. Nguyen and Matthias Hein. Optimization landscape and expressivity of deep cnns. In *International Conference on Machine Learning*, 2017.

[169] Daniel Gissin, Shai Shalev-Shwartz, and Amit Daniely. The implicit bias of depth: How incremental learning drives generalization. *ArXiv*, abs/1909.12051, 2019.

[170] Tycho FA van der Ouderaa and Mark van der Wilk. Learning invariant weights in neural networks. In *Uncertainty in Artificial Intelligence*, pages 1992–2001. PMLR, 2022. ISBN 2640-3498.

[171] Liu Ziyin. Symmetry induces structure and constraint of learning. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[172] Daniel Yamins, Ha Hong, Charles F. Cadieu, Ethan A. Solomon, Darren Seibert, and James J. DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*, 111:8619 – 8624, 2014.

[173] Radoslaw Martin Cichy, Aditya Khosla, Dimitrios Pantazis, Antonio Torralba, and Aude Oliva. Comparison of deep neural networks to spatio-temporal cortical dynamics of human visual object recognition reveals hierarchical correspondence. *Scientific Reports*, 6, 2016.

[174] Michael Eickenberg, Alexandre Gramfort, Gaël Varoquaux, and Bertrand Thirion. Seeing it all: Convolutional network layers map the function of the human visual system. *NeuroImage*, 152:184–194, 2017.

[175] Chenglei Han, Ziyang Wang, Hongyu Zhao, and Heng Ji. In-context learning of large language models explained as kernel regression. *arXiv preprint*, 2023.

[176] Tomaso A. Poggio. How deep sparse networks avoid the curse of dimensionality: Efficiently computable functions are compositionally sparse. 2023.

[177] Zhao Song, Jing Xiong, and Chiwun Yang. How sparse attention approximates exact attention?your attention is naturally $n^c$-sparse. In *Sparsity in LLMs (SLLM): Deep Dive into Mixture of Experts, Quantization, Hardware, and Inference*, 2025.

[178] Shikhar Murty, Pratyusha Sharma, Jacob Andreas, and Christopher D Manning. Characterizing intrinsic compositionality in transformers with tree projections. *arXiv preprint arXiv:2211.01288*, 2022.

[179] Jesse Vig and Yonatan Belinkov. Analyzing the structure of attention in a transformer language model. In *BlackboxNLP@ACL*, 2019.

[180] Jackson Petty, Sjoerd van Steenkiste, Ishita Dasgupta, Fei Sha, Dan Garrette, and Tal Linzen. The impact of depth on compositional generalization in transformer language models. In *North American Chapter of the Association for Computational Linguistics*, 2023.

[181] Eran Malach. Auto-regressive next-token predictors are universal learners. *ArXiv*, abs/2309.06979, 2023.

[182] Bingbin Liu, Jordan T Ash, Surbhi Goel, Akshay Krishnamurthy, and Cyril Zhang. Transformers learn shortcuts to automata. *arXiv preprint arXiv:2210.10749*, 2022.

[183] Yulu Gan, Tomer Galanti, Tomaso A. Poggio, and Eran Malach. On the power of decision trees in auto-regressive language modeling. *ArXiv*, abs/2409.19150, 2024.

[184] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[185] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[186] Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025.

[187] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, F. Xia, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. *arXiv preprint*, 2022.

[188] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint*, 2023.

[189] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, John A. Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuan-Fang Li, Scott M. Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint*, 2023.

[190] Peter Belcak and Roger Wattenhofer. Fast feedforward networks. *arXiv preprint arXiv:2308.14711*, 2023.

[191] Brian Cheung, Alexander Terekhov, Yubei Chen, Pulkit Agrawal, and Bruno Olshausen. Superposition of many models into one. *Advances in neural information processing systems*, 32, 2019.

[192] Pierfrancesco Beneventano, Andrea Pinto, and Tomaso A. Poggio. How neural networks learn the support is an implicit regularization effect of sgd. *ArXiv*, abs/2406.11110, 2024.

[193] Pelin Gundogdu, Carlos Loucera, Inmaculada Alamo-Alvarez, Joaquin Dopazo, and Isabel Nepomuceno. Integrating pathway knowledge with deep neural networks to reduce the dimensionality in single-cell rna-seq data. *BioData Mining*, 15(1):1, Jan 2022. ISSN 1756-0381. doi: 10.1186/s13040-021-00285-4.

[194] Mohammad Lotfollahi, Anna Klimovskaia Susmelj, Carlo De Donno, Leon Hetzel, Yuge Ji, Ignacio L. Ibarra, Sanjay R. Srivatsan, Mohsen Naghipourfar, Riza M. Daza, Beth Martin, Jay Shendure, Jose L. McFaline-Figueroa, Pierre Boyeau, F. Alexander Wolf, Nafissa Yakubova, Stephan Günnemann, Cole Trapnell, David Lopez-Paz, and Fabian J. Theis. Predicting cellular responses to complex perturbations in high-throughput screens. *Molecular Systems Biology*, 19(6):MSB202211517, May 2023. ISSN 1744-4292. doi: 10.15252/msb.202211517.

[195] Pelin Gundogdu, Inmaculada Alamo, Isabel A. Nepomuceno-Chamorro, Joaquin Dopazo, and Carlos Loucera. Sigprimednet: A signaling-informed neural network for scrna-seq annotation of known and unknown cell types. *Biology*, 12(4), 2023. ISSN 2079-7737. doi: 10.3390/biology12040579.

[196] Qingyang Yin and Liang Chen. Celltics: an explainable neural network for cell-type identification and interpretation based on single-cell rna-seq data. *Briefings in Bioinformatics*, 25(1):bbad449, 12 2023. ISSN 1477-4054. doi: 10.1093/bib/bbad449.

[197] Jia-Cheng Wang, Yao-Jia Chen, and Quan Zou. Grace: Unveiling gene regulatory networks with causal mechanistic graph neural networks in single-cell rna-sequencing data. *IEEE Transactions on Neural Networks and Learning Systems*, 36(5):9005–9017, 2025. doi: 10.1109/TNNLS.2024.3412753.

[198] Peizhuo Wang, Xiao Wen, Han Li, Peng Lang, Shuya Li, Yipin Lei, Hantao Shu, Lin Gao, Dan Zhao, and Jianyang Zeng. Deciphering driver regulators

of cell fate decisions from single-cell transcriptomics data with cefcon, Dec 2023. ISSN 2041-1723.

[199] Xudong Han, Bing Wang, Chenghao Situ, Yaling Qi, Hui Zhu, Yan Li, and Xuejiang Guo. scapgnn: A graph neural network-based framework for active pathway and gene module inference from single-cell multi-omics data. *PLOS Biology*, 21, 2023.

[200] Hegang Chen, Yuyin Lu, Zhiming Dai, Yuedong Yang, Qing Li, and Yanghui Rao. Comprehensive single-cell rna-seq analysis using deep interpretable generative modeling guided by biological hierarchy knowledge. *Briefings in Bioinformatics*, 25(4):bbae314, 07 2024. ISSN 1477-4054. doi: 10.1093/bib/bbae314.

[201] Sebastiano Cultrera di Montesano, Davide D'Ascenzo, Srivatsan Raghavan, Ava P. Amini, Peter S. Winter, and Lorin Crawford. Improving atlas-scale single-cell annotation models with hierarchical cross-entropy loss. *Nature Computational Science*, Jan 2026. ISSN 2662-8457. doi: 10.1038/s43588-025-00945-z.

[202] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6(50):1453–1484, 2005.

[203] Gükhan H. Bakir, Thomas Hofmann, Bernhard Scholkopf, Alex Smola, Ben Taskar, and S. V. N. Vishwanathan. Predicting structured data. 2007.

[204] Sebastian Nowozin and Christopher H. Lampert. *Structured Learning and Prediction in Computer Vision*. 2011.

[205] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[206] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, Gerald M. Rubin, and Gavin Sherlock. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, May 2000. ISSN 1546-1718. doi: 10.1038/75556.

[207] Giorgio Valentini. Hierarchical ensemble methods for protein function prediction. *ISRN Bioinformatics*, 2014.

[208] Carlos le Sage, Steffen Lawo, and Benedict C. S. Cross. Crispr: A screener's guide. *Slas Discovery*, 25:233 – 240, 2019.

[209] Kyuho Han, Edwin E Jeng, Gaelen T. Hess, David W. Morgens, Amy Li, and Michael C. Bassik. Synergistic drug combinations for cancer identified in a crispr screen for pairwise genetic interactions. *Nature biotechnology*, 35:463 – 474, 2017.

[210] Michael J. T. Stubbington, Orit Rozenblatt-Rosen, Aviv Regev, and Sarah A. Teichmann. Single-cell transcriptomics to explore the immune system in health and disease. *Science*, 358:58 – 63, 2017.

[211] Byungjin Hwang, Ji Hyun Lee, and Duhee Bang. Single-cell rna sequencing technologies and bioinformatics pipelines. *Experimental & Molecular Medicine*, 50(8):1–14, Aug 2018. ISSN 2092-6413. doi: 10.1038/s12276-018-0071-8.

[212] Hyunmin Woo and Seong-il Eyun. Applications and techniques of single-cell rna sequencing across diverse species. *Briefings in Bioinformatics*, 26(4), 07 2025. ISSN 1477-4054. doi: 10.1093/bib/bbaf354.

[213] Allon M. Klein, Linas Mazutis, Ilke Akartuna, Naren Tallapragada, Adrian Veres, Victor Li, Leonid Peshkin, David A. Weitz, and Marc W. Kirschner. Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. *Cell*, 161(5):1187–1201, 2015. ISSN 0092-8674. doi: https://doi.org/10.1016/j.cell.2015.04.044.

[214] Xiannian Zhang, Tianqi Li, Feng Liu, Yaqi Chen, Jiacheng Yao, Zeyao Li, Yanyi Huang, and Jianbin Wang. Comparative analysis of droplet-based ultra-high-throughput single-cell rna-seq systems. *Molecular Cell*, 73(1):130–142.e5, 2019. ISSN 1097-2765. doi: https://doi.org/10.1016/j.molcel.2018.10.020.

[215] Stephanie C Hicks, F William Townes, Mingxiang Teng, and Rafael A Irizarry. Missing data and technical variability in single-cell rna-sequencing experiments. *Biostatistics*, 19(4):562–578, 11 2017. ISSN 1465-4644. doi: 10.1093/biostatistics/kxx053.

[216] Michael B. Cole, Davide Risso, Allon Wagner, David DeTomaso, John Ngai, Elizabeth Purdom, Sandrine Dudoit, and Nir Yosef. Performance assessment and selection of normalization procedures for single-cell rna-seq. *Cell Systems*, 8(4):315–328.e8, Apr 2019. ISSN 2405-4712. doi: 10.1016/j.cels.2019.03.010.

[217] Robert A. Amezquita, Aaron T. L. Lun, Etienne Becht, Vince J. Carey, Lindsay N. Carpp, Ludwig Geistlinger, Federico Marini, Kevin Rue-Albrecht, Davide Risso, Charlotte Soneson, Levi Waldron, Hervé Pagès, Mike L. Smith, Wolfgang Huber, Martin Morgan, Raphael Gottardo, and Stephanie C. Hicks. Orchestrating single-cell analysis with bioconductor. *Nature Methods*, 17(2):137–145, Feb 2020. ISSN 1548-7105. doi: 10.1038/s41592-019-0654-x.

[218] Malte D Luecken and Fabian J Theis. Current best practices in single-cell rna-seq analysis: a tutorial. *Molecular Systems Biology*, 15(6):e8746, 2019. doi: https://doi.org/10.15252/msb.20188746.

[219] Tamim Abdelaal, Lieke Michielsen, Davy Cats, Dylan Hoogduin, Hailiang Mei, Marcel J. T. Reinders, and Ahmed Mahfouz. A comparison of automatic cell identification methods for single-cell rna sequencing data. *Genome Biology*, 20, 2019.

[220] Giovanni Pasquini, Jesus Eduardo Rojo Arias, Patrick Schäfer, and Volker Busskamp. Automated methods for cell type annotation on scrna-seq data. *Computational and Structural Biotechnology Journal*, 19:961 – 969, 2021.

[221] Cecilia Domínguez Conde, C. F. Xu, LB Jarvis, DB Rainbow, SB Wells, Tomás Gomes, S Howlett, Ondrej Suchanek, Krzysztof Polański, HW King, Lira Mamanova, Ni Huang, PA Szabo, Laura Richardson, Liam Bolt, ES Fasouli, K. T. Mahbubani, Martin Prete, Liz Tuck, Nathan Richoz, ZK Tuong, L Campos, HS Mousa, EJ Needham, Sophie Pritchard, T. Li, Rasa Elmentaite, J. Park, E. Rahmani, D. Chen, D. Menon, OA Bayraktar, LK James, K. B. Meyer, Nir Yosef, MR Clatworthy, Patrick C. Sims, DL Farber, Kourosh Saeb-Parsy, Jl Jones, and Sandra Teichmann. Cross-tissue immune cell analysis reveals tissue-specific features in humans. *Science (New York, N.Y.)*, 376: eabl5197 – eabl5197, 2022.

[222] Can Ergen, Galen K. Xing, Chenling Xu, Martin Kim, Michael Jayasuriya, Erin McGeever, Angela Oliveira Pisco, Aaron M. Streets, and Nir Yosef. Consensus prediction of cell type labels in single-cell data with popv. *Nature Genetics*, 56:2731 – 2738, 2024.

[223] Yixuan Huang and Peng Zhang. Evaluation of machine learning approaches for cell-type identification from single-cell transcriptomics data. *Briefings in bioinformatics*, 2021.

[224] David Osumi-Sutherland, Chuan Xu, Maria Keays, Adam P. Levine, Peter V. Kharchenko, Aviv Regev, Ed Lein, and Sarah A. Teichmann. Cell type ontologies of the human cell atlas. *Nature Cell Biology*, 23(11):1129–1135, Nov 2021. ISSN 1476-4679.

[225] Simon Jupp, Tony Burdett, Catherine Leroy, and Helen E. Parkinson. A new ontology lookup service at embl-ebi. In *Workshop on Semantic Web Applications and Tools for Life Sciences*, 2015.

[226] Felix Fischer, David S Fischer, R. S. Mukhin, A. V. Isaev, Evan Biederstedt, Alexandra-Chloé Villani, and Fabian J. Theis. sctab: Scaling cross-tissue single-cell annotation models. *Nature Communications*, 15, 2024.

[227] Sheng Wang, Angela Oliveira Pisco, Aaron McGeever, Maria Brbić, Marinka Zitnik, Spyros Darmanis, Jure Leskovec, Jim Karkanias, and Russ B. Altman. Leveraging the cell ontology to classify unseen cell types. *Nature Communications*, 12, 2021.

[228] Matthew N. Bernstein, Zhongjie Ma, Michael Gleicher, and Colin N. Dewey. Cello: comprehensive and hierarchical cell type classification of human cells with the cell ontology. *iScience*, 24, 2019.

[229] Jiawei Chen, Hao Xu, Wanyu Tao, Zhaoxiong Chen, Yuxuan Zhao, and Jing-Dong J. Han. Transformer for one stop interpretable cell type annotation. *Nature Communications*, 14, 2023.

[230] Jing Xu, Aidi Zhang, Fang Liu, Liang Chen, and Xiujun Zhang. Ciform as a transformer-based model for cell-type annotation of large-scale single-cell rna-seq data. *Briefings in bioinformatics*, 2023.

[231] Sercan Ö. Arik and Tomas Pfister. Tabnet: Attentive interpretable tabular learning. *ArXiv*, abs/1908.07442, 2019.

[232] Graham Heimberg, Tony Kuo, Daryle J. DePianto, Omar Salem, Tobias Heigl, Nathaniel Diamant, Gabriele Scalia, Tommasso Biancalani, Shannon J. Turley, Jason R. Rock, Héctor Corrada Bravo, Josh Kaminker, Jason A. Vander Heiden, and Aviv Regev. A cell atlas foundation model for scalable search of similar human cells. *Nature*, 638:1085 – 1094, 2024.

[233] Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979.

[234] Caroline Uhler. Building a two-way street between cell biology and machine learning. *Nature Cell Biology*, 26(1):13–14, Jan 2024. ISSN 1476-4679. doi: 10.1038/s41556-023-01279-6.

[235] Christina V. Theodoris, Ling Xiao, Anant Chopra, Mark D. Chaffin, Zeina R. Al Sayed, Matthew C. Hill, Helene Mantineo, Elizabeth M Brydon, Zexian Zeng, X. Shirley Liu, and Patrick T. Ellinor. Transfer learning enables predictions in network biology. *Nature*, 618:616–624, 2023.

[236] Davide D'Ascenzo and Sebastiano Cultrera di Montesano. scdataset: Scalable data loading for deep learning on large-scale single-cell omics, 2025.

[237] Itay Tirosh, Benjamin Izar, Sanjay M. Prakadan, Marc H. Wadsworth, Daniel J. Treacy, John J. Trombetta, Asaf Rotem, Christopher Rodman, Christine G. Lian, George Murphy, Mohammad Fallahi-Sichani, Ken Dutton-Regester, Jia-Ren Lin, Ofir Cohen, Parin Shah, Diana Lu, Alex S Genshaft, Travis Hughes, Carly G. K. Ziegler, Samuel W. Kazer, Aleth Gaillard, Kellie E. Kolb, Alexandra-Chloé Villani, Cory M. Johannessen, Aleksandr Andreev, Eliezer M. Van Allen, Monica M. Bertagnolli, Peter K. Sorger, Ryan J. Sullivan, Keith T. Flaherty, Dennie T. Frederick, Judit Jané-Valbuena, Charles H. Yoon, Orit Rozenblatt-Rosen, Alex K. Shalek, Aviv Regev, and Levi A. Garraway. Dissecting the multicellular ecosystem of metastatic melanoma by single-cell rna-seq. *Science*, 352:189 – 196, 2016.

[238] Tomasz J. Nowakowski, Aparna Bhaduri, Alex A. Pollen, Beatriz Alvarado, Mohammed Andres Mostajo-Radji, Elizabeth Di Lullo, Maximilian Haeussler, Carmen Sandoval-Espinosa, Siyuan John Liu, Dmitry Velmeshev, Johain R Ounadjela, Joe Shuga, Xiaohui Wang, Daniel A. Lim, Jay A. A. West, Anne A. Leyrat, W. James Kent, and Arnold R. Kriegstein. Spatiotemporal gene expression trajectories reveal developmental hierarchies of the human cortex. *Science*, 358:1318 – 1323, 2017.

[239] Alexandra-Chloé Villani, Rahul Satija, Gary Reynolds, Siranush Sarkizova, Karthik Shekhar, James Fletcher, Morgane Griesbeck, Andrew Butler, Shiwei Zheng, Suzan Lazo, Laura Jardine, David Dixon, Emily Stephenson, Emil Nilsson, Ida Grundberg, David McDonald, Andrew Filby, Weibo Li, Philip Lawrence De Jager, Orit Rozenblatt-Rosen, Andrew A. Lane, Muzlifah A. Haniffa, Aviv Regev, and Nir Hacohen. Single-cell rna-seq reveals new types of human blood dendritic cells, monocytes, and progenitors. *Science*, 356, 2017.

[240] Yong Xin Wang and Nicholas E. Navin. Advances and applications of single-cell sequencing technologies. *Molecular cell*, 58 4:598–609, 2015.

[241] Jason D. Buenrostro, Beijing Wu, Ulrike Litzenburger, David W. Ruff, Michael L. Gonzales, Michael Paul Snyder, Howard Y. Chang, and William James Greenleaf. Single-cell chromatin accessibility reveals principles of regulatory variation. *Nature*, 523:486 – 490, 2015.

[242] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, S. Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen A. Creel, Jared Davis, Dora Demszky, Chris Donahue, Moussa Koulako Bala Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah D. Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas F. Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, O. Khattab, Pang Wei Koh, Mark S. Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Benjamin Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, J. F. Nyarko, Giray Ogut, Laurel J. Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Robert Reich, Hongyu Ren, Frieda Rong, Yusuf H. Roohani, Camilo Ruiz, Jack Ryan, Christopher R'e, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishna Parasuram Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei A.

Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models. *ArXiv*, abs/2108.07258, 2021.

[243] Isaac Virshup, Sergei Rybakov, Fabian J. Theis, Philipp Angerer, and F. Alexander Wolf. anndata: Access and store annotated data matrices. *Journal of Open Source Software*, 9(101):4371, 2024. doi: 10.21105/joss.04371.

[244] Léon Bottou. *On-line learning and stochastic approximations*, pages 9–42. Cambridge University Press, USA, 1999. ISBN 0521652634.

[245] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *ArXiv*, abs/1609.04836, 2016.

[246] F. Alexander Wolf, Philipp Angerer, and Fabian J. Theis. Scanpy: large-scale single-cell gene expression data analysis. *Genome Biology*, 19(1):15, Feb 2018. ISSN 1474-760X. doi: 10.1186/s13059-017-1382-0.

[247] Adam Gayoso, Romain Lopez, Galen Xing, Pierre Boyeau, Katherine Wu, Michael Jayasuriya, Edouard Melhman, Maxime Langevin, Yining Liu, Jules Samaran, Gabriel Misrachi, Achille Nazaret, Oscar Clivio, Chenling Xu, Tal Ashuach, Mohammad Lotfollahi, Valentine Svensson, Eduardo da Veiga Beltrame, Carlos Talavera-López, Lior Pachter, Fabian J. Theis, Aaron Streets, Michael I. Jordan, Jeffrey Regier, and Nir Yosef. scvi-tools: a library for deep probabilistic analysis of single-cell omics data. *bioRxiv*, 2021. doi: 10.1101/2021.04.28.441833.

[248] Felix Fischer, David S. Fischer, Roman Mukhin, Andrey Isaev, Evan Biederstedt, Alexandra-Chloé Villani, and Fabian J. Theis. sctab: Scaling cross-tissue single-cell annotation models. *Nature Communications*, 15(1):6611, Aug 2024. ISSN 2041-1723. doi: 10.1038/s41467-024-51059-5.

[249] Graham Heimberg, Tony Kuo, Daryle J. DePianto, Omar Salem, Tobias Heigl, Nathaniel Diamant, Gabriele Scalia, Tommaso Biancalani, Shannon J. Turley, Jason R. Rock, Héctor Corrada Bravo, Josh Kaminker, Jason A. Vander Heiden, and Aviv Regev. A cell atlas foundation model for scalable search of similar human cells. *Nature*, 638(8052):1085–1094, Feb 2025. ISSN 1476-4687. doi: 10.1038/s41586-024-08411-y.

[250] Jérémie Kalfon, Jules Samaran, Gabriel Peyré, and Laura Cantini. scprint: pretraining on 50 million cells allows robust gene network predictions. *Nature Communications*, 16(1):3607, Apr 2025. ISSN 2041-1723. doi: 10.1038/s41467-025-58699-1.

[251] Peter St. John, Dejun Lin, Polina Binder, Malcolm Greaves, Vega Shah, John St. John, Adrian Lange, Patrick D. Hsu, Rajesh Illango, Arvind Ramanathan, Anima Anandkumar, David H Brookes, Akosua Busia, Abhishaike Mahajan, Stephen Malina, Neha Prasad, Sam Sinai, Lindsay Edwards, Thomas

Gaudelet, Cristian Regep, Martin Steinegger, Burkhard Rost, Alexander Brace, Kyle Hippe, Luca Naef, Keisuke Kamata, George Armstrong, Kevin Boyd, Zhonglin Cao, Han-Yi Chou, Simon Chu, Allan dos Santos Costa, Sajad Darabi, Eric Dawson, Kieran Didi, Cong Fu, Mario Geiger, Michelle Gill, Darren J. Hsu, Gagan Kaushik, Maria Korshunova, Steven T. Kothen-Hill, Youhan Lee, Meng Liu, Micha Livne, Zachary McClure, Jonathan Mitchell, Alireza Moradzadeh, Ohad Mosafi, Youssef L. Nashed, Saee Paliwal, Yuxing Peng, Sara Rabhi, Farhad Ramezanghorbani, Danny Reidenbach, Camir Ricketts, Brian Roland, Kushal Shah, Tyler Shimko, Hassan Sirelkhatim, Savitha Srinivasan, Abraham C Stern, Dorota Toczydlowska, Srimukh Prasad Veccham, Niccolo Alberto Elia Venanzi, Anton Vorontsov, Jared Wilber, Isabel Wilkinson, Wei Jing Wong, Eva Xue, Cory Ye, Xin Yu, Yang Zhang, Guoqing Zhou, Becca Zandstein, Christian Dallago, Bruno Trentini, Emine Kucukbenli, Timur Rvachov, Eddie Calleja, Johnny Israeli, Harry Clifford, Risto Haukioja, Nicholas Haemel, Kyle Tretina, Neha Tadimeti, and Anthony B Costa. Bionemo framework: a modular, high-performance library for ai model development in drug discovery. *ArXiv*, abs/2411.10548, 2024.

[252] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.