Trusting deep learning natural-language models via local and global explanations

Noname manuscript No.
(will be inserted by the editor)

Trusting deep learning natural-language models via local and global explanations

Francesco Ventura* · **Salvatore Greco*** ·
Daniele Apiletti · **Tania Cerquitelli**

Received: 22 Dec 2020 / Revised: 12 Apr 2022 / Accepted: 23 Apr 2022

Abstract Despite the high accuracy offered by state-of-the-art deep natural-language models (e.g., LSTM, BERT), their application in real-life settings is still widely limited, as they behave like a black-box to the end-user. Hence, explainability is rapidly becoming a fundamental requirement of future-generation data-driven systems based on deep-learning approaches. Several attempts to fulfill the existing gap between accuracy and interpretability have been made. However, robust and specialized XAI (eXplainable Artificial Intelligence) solutions, tailored to deep natural-language models, are still missing. We propose a new framework, named T-EBAnO, which provides innovative prediction-local and class-based model-global explanation strategies tailored to deep learning natural-language models. Given a deep NLP model and the textual input data, T-EBAnO provides an objective, human-readable, domain-specific assessment of the reasons behind the automatic decision-making process. Specifically, the framework extracts sets of *interpretable features* mining the inner knowledge of the model. Then, it quantifies the influence of each feature during the prediction process by exploiting the *normalized Perturbation Influence Relation* index at the local level and the novel *Global Absolute Influence* and *Global Relative Influence* indexes at the global level. The effectiveness and the quality of the local and global explanations obtained with T-EBAnO are proved on an extensive set of experiments addressing

Francesco Ventura*
Department of Control and Computer Engineering, Politecnico di Torino, Turin, Italy
E-mail: francesco.ventura@polito.it

Salvatore Greco* (Corresponding author)
Department of Control and Computer Engineering, Politecnico di Torino, Turin, Italy
E-mail: salvatore_greco@polito.it

Daniele Apiletti
Department of Control and Computer Engineering, Politecnico di Torino, Turin, Italy
E-mail: daniele.apiletti@polito.it

Tania Cerquitelli
Department of Control and Computer Engineering, Politecnico di Torino, Turin, Italy
E-mail: tania.cerquitelli@polito.it

*These authors contributed equally

different tasks, such as a sentiment-analysis task performed by a fine-tuned BERT model and a toxic-comment classification task performed by an LSTM model. The quality of the explanations proposed by T-EBAnO, and, specifically, the correlation between the influence index and human judgment, has been evaluated by humans in a survey with more than 4000 judgments. To prove the generality of T-EBAnO and its model/task-independent methodology, experiments with other models (ALBERT, ULMFit) on popular public datasets (Ag News and Cola) are also discussed in detail.

Keywords eXplainable Artificial Intelligence · Natural Language Processing · Text Classification · Black-Box Classifier · Neural Network

Declaration

Funding: Not applicable.

Conflicts of interest/Competing interests: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Availability of data and material: Not applicable.

Code availability: The code for T-EBAnO is available at

`https://github.com/EBAnO-Ecosystem/Text-EBAnO-Express`

1 Introduction

Nowadays, more and more deep learning models such as BERT [12] and LSTM [19] are exploited as the ground basis to build new powerful automatic decision-making systems to automatically address complex natural language processing (NLP) tasks, e.g., text classification, question answering (QA), and sentiment analysis. Although deep learning models are often very accurate, even exceeding human performance (e.g., in [49,39,36,5]), they are very opaque and defined as "black-boxes": given an input, deep learning models provide an output, without any human-understandable insight about their inner behavior. The huge amount of data required to train these black-box models is usually collected from people's daily lives (e.g., web searches, social networks, e-commerce), increasing the risk of inheriting human prejudices, racism, gender discrimination, and other forms of bias [27,6]. For these reasons, new *eXplainable Artificial Intelligence* (XAI) solutions are needed to produce more credible and reliable information and services. XAI components will become, shortly, a design requirement in most data-driven decision-making processes [11], and they will be rewarded by increased trust, interaction, and access to new forms of data.

Table 1 shows a clear example of a misleading prediction provided by an LSTM model[1]. In the example, both sentences express *Clean* language. However, the

[1] Details on the experiments leading to the reported result are provided in Section 6.1 trained to distinguish between *Clean* and *Toxic* comments.

predictions are extremely contradictory, and the black-box nature of the LSTM model does not allow us to understand why. Thus, the complexity and the opacity of the learning process significantly reduce the adoption of those neural networks in real-life scenarios where a higher level of transparency is needed. The new *eXplainable Artificial Intelligence* (XAI) field of research is currently trying to close the gap between model accuracy and model interpretability to effectively increase the adoption of those models in real-life settings.

This work proposes T-EBAnO (Text-Explaining BlAck-box mOdels), a novel explanation framework that allows understanding the decisions made by deep neural networks in the context of *Natural Language Processing*.

Human-readable *prediction-local* and *model-global* explanations are offered to users to understand why and how a prediction is made, hence allowing them to consciously trust the model's outcomes. With the term *prediction-local explanation*, we mean to provide the relation of a specific input text with the predicted label: the explanation is local to the label and the input, and it aims at identifying which regions of the inputs, i.e., the tokens for NLP models or pixels for computer vision models, are mostly impacting/influencing the output prediction of the model. Instead, with the term *model-global explanation*, we mean to obtain general insights about the model behavior by globally analyzing many local explanations over different input texts.

T-EBAnO produces *prediction-local* explanations through a perturbation process applied on different sets of *interpretable features*, i.e., parts of speech, sentences, and multi-layer word embedding clusters, which are accurately selected to be meaningful for the model and understandable by humans. Then, T-EBAnO evaluates the model's performance in the presence of the perturbed inputs, quantifying the contribution that each feature had in the prediction process through qualitative and objective indexes. The proposed explanations enable end-users to decide whether a specific local prediction made by a deep learning model is reliable and to evaluate the general behavior of the global model across predictions. *Prediction-local* and *model-global* explanations are summarized in reports consisting of *textual* and *quantitative* contributions, allowing both expert and non-expert users to understand the reasons why a certain decision has been taken by the model under analysis.

Experimentally, T-EBAnO has been applied to explain: (i) the well-known state-of-the-art transformer-based language model BERT [12] in a sentiment analysis task, (ii) a custom sequence LSTM [19] model trained to solve a toxic comment binary classification task, i.e., detecting whether a document contains threats, obscenity, insults, or hate speech, and (iii) additional models like ALBERT [25] and ULMFit [20] on other two classification tasks of popular public datasets (*Ag News* topic classification and *Cola* sentence acceptability). Experimental results show the effectiveness of T-EBAnO in providing human-readable, local vs global interpretations of different model outcomes.

The novel contributions of the current work are provided in the following.

- The design and development of a new XAI methodology, named T-EBAnO, tailored to NLP tasks, to produce both prediction-local and model-global explanations, consisting of textual and numerical human-readable reports.

Sentence	$P(Toxic)$
`Politician-1 is an awesome man`	0.17
`Politician-1 is an intellectual`	0.89

Table 1: Misleading prediction example of a clean/toxic comment classification. The surname of a well-know politician is anonymized.

- The design of effective strategies to describe input textual documents through a set of model-wise interpretable features exploiting specific inner-model and domain-specific knowledge (Section 4.1).
- The definition of a cutting-edge model-global explanation strategy, analyzing the influence of inter- and intra- class concepts, based on two new metrics, the *Global Absolute Influence* and the *Global Relative Influence* scores (Section 5.2).
- A thorough experimental evaluation on many state-of-the-art black-box deep-learning models, such as BERT, LSTM, ALBERT, and ULMFit, on different textual data collections and text classification tasks. Results show that the proposed approach is general and widely applicable, independently from the model or task.
- A human evaluation of the correlation between the influence index exploited by T-EBAnO (*normalized Perturbation Influence Relation*) and human judgment. We collected 4320 user evaluations from 108 participants, each evaluating 2 explanations from 20 input texts, showing that the proposed index is highly correlated with human judgment.

The paper is organized as follows. Section 2 discusses XAI literature, Section 3 provides an overview of the proposed solution, Section 4 provides the details about the *interpretable features* extracted by our framework, and Section 5 describes how the local and global explanations are computed. Section 6 presents the experimental results and discusses the prediction-local and model-global explanation reports produced by T-EBAnO. Finally, Section 7 concludes this work and presents future research directions.

2 Literature review

Research activities in XAI can be classified based on [18,41,2] data-type (e.g., structured data, images, texts), machine learning task (e.g., classification, forecasting, clusterization), and characteristics of the explanations (e.g., local vs global). More generally, explanation frameworks can be grouped into (i) model-agnostic, (ii) domain-specific, and (iii) task-specific approaches.

Up to now, many efforts have been devoted to explaining the prediction process in the context of structured data (e.g., measuring quantitative input influence [10], by means of local rules in [37,8]) and of deep learning models for image classification (e.g., [43,16,48]). In contrast, less attention has been devoted to domain-specific explanation frameworks for textual data analytics.

Model-agnostic approaches. Tools like [40,21,32] can be applied to explain the decisions made by a black-box model on unstructured inputs (e.g., images or texts), and they provide interesting and human-readable results. LIME [40] is a

model-agnostic strategy that allows a local explanation to be generated for any predictive model. It approximates the prediction performed by the model with an interpretable model built locally to the data object to be predicted. However, the interpretable model approximates the prediction locally, and it could not represent faithfully what the real model has effectively learned. SHAP [32], instead, is a unified framework able to interpret predictions produced by any machine learning model, exploiting a game-theoretic approach based on the concept of *Shapley Values* [44], by iteratively removing possible combinations of input features and measuring the impact that the removal of the features has over the outcome of the prediction task. PALEX [21] is a model-agnostic explanation method that provides multiple *local explanations* for individual predictions. It uses frequent input patterns to generate a precise neighborhood of the prediction and exploits intrinsic interpretability of contrast patterns to capture locally important information. Since the above-mentioned techniques are *model-agnostic*, they might not fully exploit the specific characteristics of the data domain and the latent semantic information specifically learned by the predictive models when computing an explanation. Although they can be applied in the context of NLP, they do not provide *inner-model awareness*, i.e., they are not able to deeply explain what the model has specifically learned since they do not exploit such information in their explanation process, leading to less specific explanations. Moreover, in the specific case of NLP, model-agnostic techniques analyze the impact of singular words over the prediction without taking into account the complex semantic relations that exist in textual documents (i.e., semantically correlated portions of text) and that is actually learned by modern neural networks. Also, perturbing singular words can have a very limited impact on the prediction process, in particular when dealing with long texts, other than being very computationally intensive, compromising the quality of the explanations.

T-EBAnO addresses such limitations and is able to increase the precision of the produced explanations and limit the feature search space by i) using *domain-specific* feature extraction techniques and ii) exploiting the *inner knowledge* of the neural network to identify meaningful inter-word relations learned by the NLP model.

Domain-specific approaches. An exhaustive overview of the existing XAI techniques for NLP models, applied in different contexts, such as social networks, medical, and cybersecurity, is presented in [34]. Many works exploit feature-perturbation strategies in the explanation process, analyzing the model reactions to produce prediction-local explanations, like in [3,48,32,40,29,35]. This straightforward idea is very powerful but requires a careful selection of the input features to be perturbed.

Differently from *model-agnostic* and *domain-agnostic* frameworks [40,32], some strategies have been explored by *domain-specific* works to determine the information contained in the target model, with the aim to select the most relevant features to be perturbed. The feature extraction process is of utmost importance in the explanation process since the quality of the produced explanations strictly depends on this step. In [35], the authors propose the use of an approximate brute-force strategy to analyze the impact that phrases in the input text have over the predictions made by LSTM models. Also, they define an importance score that exploits the parameters learned by an LSTM model to select the phrases which

consistently provided a large contribution in the prediction of a specific class. However, this approach has been tailored to LSTM models, making it difficult to generalize the solution. In [3], the authors proposed an explanation strategy tailored to structured and sequential data models with a perturbation strategy that exploits the training of a *variational autoencoder* to perturb the input data with semantically related variations, introducing controlled perturbations. However, this explanation strategy has been mainly focused on explaining sequence to sequence scenarios (e.g., machine translation), and the perturbation requires the training, in advance, of a variational autoencoder model, introducing a further level of opacity and complexity in the explanation process. The authors in [26] propose to learn how to explain a predictive model jointly with the training of the predictor. To this aim, they introduce an *encoder-generator* framework that extracts a subset of inputs from the original text as an interpretable summary of the prediction process. Again, the training of a separate model is required to extract the whole explanation, also making this solution equivocal for the end-user. The authors in [29] proposed an explanation process based on a novel strategy to select the minimal set of words to perturb what causes a change in the model's decision. To this aim, a reinforcement learning approach has been exploited. However, as in previous cases, this method requires the training of an external model to extract features to be perturbed, increasing the complexity and affecting the reliability of the explanation process itself. The authors in [13] propose a framework called CREX that allows regularizing the training of DNNs using prior human knowledge. The prior human knowledge, consisting of a subset of features highlighted by domain experts, is exploited to let the model focus more on what actually matters for the task. However, the highlighting operation is time consuming, it is not always feasible, and it is not applicable to already trained models. The authors in [9] propose LS-Tree, a model-agnostic but domain-specific game-theoretic technique based on the *Banzhaf* value [4] and parse trees to analyze several aspects of NLP models such as the nonlinearity, adversarial relationship captured, and overfitting. However, it is more suitable to acquire global insights about the model behavior instead of explaining single predictions of the model. Moreover, it has high complexity, especially for long sentences. Finally, its explanations are more suited for an expert audience.

Instead, other techniques are *gradient-based* and, thus, exploit gradients to produce explanations [43, 45]. In [43], the authors propose Grad-CAM, a gradient-based approach that highlights the important regions in the image for the prediction. However, it is suitable for convolutional-based neural networks, and thus, for the computer vision domain. The authors in [28] propose a Grad-CAM implementation for text classification named Grad-CAM-Text. However, it is only applicable to 1D convolutional neural networks for text classification. Therefore, it is inapplicable for sequence models such as RNNs or transformer-based models as BERT, which are currently the most widespread architectures for NLP tasks. Finally, the authors in [45] propose DeepLIFT, a gradient-based technique that computes importance scores by explaining the difference between the outputs of the input to explain from the outputs obtained by a 'reference' input. However, it requires prior knowledge to make assumptions on reference data. Moreover, it has been only tested on convolutional neural networks, and, again, the version presented in the paper is unsuitable for sequence or transformer-based models.

Different from the above-mentioned works, T-EBAnO implements a feature-extraction process that exploits the specific information learned by the predictive deep natural-language model, without the need to train external resources. T-EBAnO exploits the embedding representation of the textual input data, available in the inner layer of the neural network, to identify correlated portions of input text accordingly to the model, which are used in the explanation process. To support this choice, we recall that textual embeddings have interesting interpretable properties, as described in [46]. Following the insights discussed by the authors in [14], modern natural-language models incorporate most of the context-specific information in the latest and inmost layers. T-EBAnO exploits the textual embedding representations as interpretable features to explain model outcomes.

Task-specific approaches. Finally, not every task can be explained with model-agnostic or domain-specific approaches. This is why interpretable task-specific solutions are also relevant. In [53], the authors focused their attention on explaining the duplicate question detection task developing a specific model based on the attention mechanism, proposing to interpret the model results by visually analyzing their attention matrix to understand the inter-words relations learned by the model. However, exploiting attention can be performed only for black-box models that are based on this mechanism, and it can be hard to interpret for non-expert users. The authors of [52] developed an explainable tag-based recommendation model that increases the interpretability of its results by proposing an overview of user's preference correlated with learned topics and predicted tags, but without actually focusing on the reliability of the model or on the possible presence of bias. In [1], the authors introduced a specific linguistic explanation approach for fuzzy classifier decisions, which are shown in textual form to users. They focus on a high abstraction level of explanations providing reasons, confidence, coverage, and feature importance. However, their approach does not take into account the complexity of deep learning models. In [23], the authors propose a framework for recognizing symptoms of cognitive decline that provides natural language explanations of the detected anomalies generated from a trained tree regression algorithm. However, this solution is customized for this specific task and not easily extendable to other contexts. In [22], the authors propose two solutions, for the k-nearest neighbor and the random shapelet forest algorithms, solving the problem of locally and globally explainable time-series tweaking. These solutions are suitable for time-series classification, and they are not easily applicable for different tasks.

T-EBAnO proposes a new local and global explanation process for state-of-the-art deep NLP models. By exploiting a perturbation-based strategy similar to that described in [48], which was successfully tailored to image data, T-EBAnO fills in the gap of missing customized solutions for explaining deep NLP models by introducing a totally redesigned architecture and experimental section. Specifically, we introduce (i) a novel feature extraction process specifically tailored to textual data and deep natural language models, (ii) new perturbation strategies, (iii) an improved version of the index proposed in [48] able to quantify the influence of the input feature over local predictions tested in a new domain (NLP) ùà. , and (iv) novel class-based global explanations, besides extending the experiments to new models and use cases, and presenting a human evaluation of the exploited index and proposed explanations.

3 T-EBAnO overview

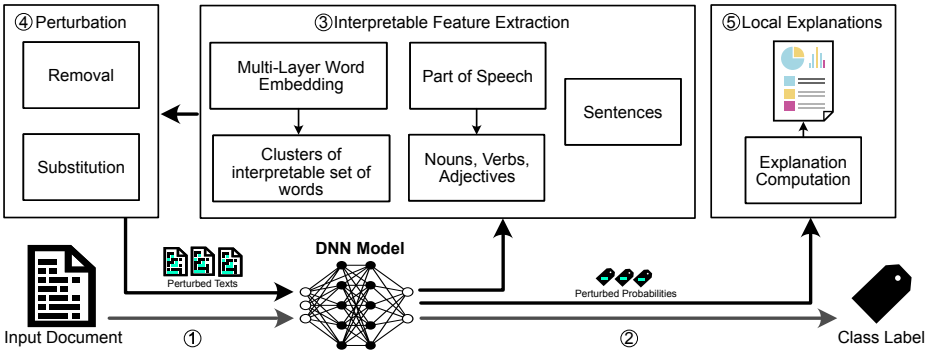


Fig. 1: T-EBAnO local explanation process.

T-EBAnO explains the inner functionalities of deep neural networks (DNN) models in the context of NLP analytics tasks. Deep learning models act as a black-box to the end-user because the model's internal decision process is obscure [18]. However, T-EBAnO requires that the model's architecture is known. For instance, for explaining the decision-making process of a transformer-based model, its architecture is known, but why it produces certain predictions is unknown and requires an explanation. Thus, T-EBAnO exploits the knowledge about the architecture of the specific model to make more reliable and faithful explanations, in contrast to completely model-agnostic methodologies that could be applied to arbitrary models but that cannot exploit the knowledge hidden into the model.

T-EBAnO's architecture is shown in Figure 1 and includes different building blocks. Both *model-agnostic* (i.e., part of speech, sentences) and *model-aware* (i.e., multi-layer word embeddings) features are extracted by T-EBAnO. The *model-aware* technique is the one that requires and exploits the knowledge about the model's architecture, while the *model-agnostic* techniques are completely independent of the model. However, all the techniques are *domain-specific* and exploit the semantic feature of textual data.

For a given classification task, an input document is provided to the pre-trained deep learning model ① that outputs its predicted *class label* ②. Thus, T-EBAnO extracts a set of *interpretable features* ③ by exploiting either NLP techniques or the analysis of the knowledge hidden in the model itself (Section 4.1). Then, it performs the *perturbation* of the set of interpretable features and tests the model's outcomes on the perturbed inputs ④ (Section 4.2). Specifically, the perturbed inputs are new texts produced by applying the perturbation to each interpretable feature extracted. Then the model's predictions (perturbed probabilities) for the perturbed texts are evaluated to measure each feature's impact. The perturbation of the interpretable features can influence the model outcome in different ways, as described in the following:

– Case (a): the probability of the class under analysis *increases*. It means that the analyzed features were negatively impacting the process;

- Case (b): the predicted probability *decreases*. It means that the perturbed features were positively impacting the class under analysis;
- Case (c): the predicted probability *remains roughly unchanged*. It means that the portion of the input is irrelevant to the predictive model under analysis.

The significance of the difference in the prediction process before and after the perturbation is evaluated through the $nPIR$ index, a quantitative metric to estimate the effect of the perturbation strategy (Section 4.3). Thus, T-EBAnO generates the *local explanation* report ⑤, showing the results of the analysis of the perturbations through an informative dashboard (Section 5.1).

Finally, aggregating the local explanations produced for a corpus of input documents, T-EBAnO provides model-global explanations highlighting relevant inter- and intra- class semantic concepts that are influencing the deep neural network decision-making process at a model-global level (Section 5.2).

4 Interpretable features

This Section describes the interpretable feature extraction (Section 4.1) and perturbation (Section 4.2). Then, it introduces the quantitative index that measures the feature importance (Section 4.3) exploited by T-EBAnO. Finally, it details the Multi-layer Word Embedding feature extraction technique (Section 4.4).

4.1 Interpretable feature extraction

The interpretable feature extraction block identifies meaningful and correlated sets of words (tokens) having an influence on the outcomes of the NLP model under the exam. It represents the most critical and complex phase in the explanation process workflow. A set of words is meaningful for the model if its perturbation in the input document produces a meaningful change in the prediction outcome. On the other hand, a set of words is meaningful for a user if s/he can easily understand and use it to support the decision-making process.

T-EBAnO considers both word (tokens) and sentence granularity levels to extract the set of interpretable features. Moreover, T-EBAnO records the position of the extracted features in the input text since the context in which words appear is often very important for NLP models.

T-EBAnO includes three different kinds of *interpretable feature extraction* techniques:

1. *Multi-layer Word Embedding* (MLWE) feature extraction. This strategy is the most powerful technique since it exploits the inner knowledge learned by the model to perform the prediction. Specifically, it performs an unsupervised clustering analysis to group related input tokens based on the inner representation (i.e., embedding) assigned by the model. Each group of tokens could have influenced the prediction of the model in a similar way. The unsupervised analysis performed by the MLWE figures out by itself which and the right number of tokens to assign to each cluster and which cluster of tokens is the most influential. To access the inner knowledge of the network, this technique needs to know the inner details of the model under analysis. However, the process can

be easily adapted to be compliant with different deep architectures (e.g., as reported in [48]) and their hidden layers. A detailed description of the MLWE feature extraction technique is provided in Section 4.4.

2. *Part-of-Speech* (PoS) feature extraction. This strategy explores the semantic meaning of words by looking at which part-of-speech they belong to (e.g., nouns, adjectives). The intuition behind this type of feature extraction is that the semantic difference corresponding to distinct parts-of-speech can differently influence the model outcome. Firstly, the input text is tokenized, leading to three features: the *token* itself, its *position* in the text, and its *pos-tag* (i.e., part-of-speech tagging). Then, tokens are divided into correlated groups: adjectives, nouns, verbs, adverbs, and others. Each group is considered as a separate *interpretable feature* by T-EBAnO in the perturbation phase (e.g., the POS-Adjectives interpretable feature extracts all the adjectives present in the input text and not a partial subset of it). This is because the main objective of the POS is to measure the influence of each entire part-of-speech, while the MLWE feature extraction discovers the exact more influential tokens. Understanding which POS most influenced the original prediction of the model can be useful to understand if the model is looking to the correct semantic aspect. Indeed, different tasks are usually influenced by different parts-of-speech. For instance, a well-trained model for sentiment analysis usually exploits adjectives to predict the sentiment. Therefore, adjectives should be the most influential and important part-of-speech in the model's decision-making process. Thus, T-EBAnO creates an interpretable feature for each analyzed part-of-speech.

3. *Sentence-based* (SEN) feature extraction. This strategy considers each sentence separately to assess its influence on the model decisions. The straightforward intuition behind this strategy is to verify if the model captures the complete meaning of a sentence and uses it to derive the outcome. The *sentence* feature extraction characterizes the input text with the *position* of the sentence and the *sentence* itself. In this case, T-EBAnO creates a feature for each sentence in the input text.

Then, separately for each feature extraction method, T-EBAnO tests *pairwise combinations of features* to create larger groups of tokens corresponding to more complex concepts. For instance, for *Part-of-Speech*, it creates a feature with the combination of *Adjectives* and *Verbs*, *Adjectives* and *Nouns*, etc. For the *Sentence-based* feature extraction, it creates a feature with the combination of the first sentence and the second, another with the first sentence and the third, and so on. Finally, for the *Multi-layer Word Embedding* feature extraction, it creates a feature with the combination of the first cluster of words and the second, the first and the third, and so on (more details on MLWE features are provided in Section 4.4). T-EBAnO creates *pairwise combinations* of features only within the same feature extraction method and not among different feature extraction methods because each of them considers different aspects of the input text, i.e., *PoS* features are combined with *PoS* features and not with *MLWE* features. This allows T-EBAnO to efficiently explore a wider search space of interpretable features, hence finding even more relevant prediction-local explanations.

4.2 Interpretable feature perturbation

After the extraction of the interpretable feature sets, a perturbation phase is performed by introducing noise and consequently assessing the impact of the perturbed features on the model outcomes. Adding noise to the model input is a well-known technique adopted by different state-of-the-art approaches [3, 48, 32, 40] to study the model behavior through the effects on the outcomes. Different input data types require different perturbation strategies. In case of textual data, the perturbation can be performed by *feature removal* or *feature substitution*.

In the *feature removal perturbation* approach provided by T-EBAnO, all the interpretable features are iteratively removed from the input text, producing new perturbed variations of the input itself. The perturbed variations of the input are then fed back into the model under analysis, and its predictions are collected and analyzed by T-EBAnO to produce the *local explanation* report (see Section 5.1). For instance, for *multi-layer word embedding* features, each cluster of tokens is removed (one cluster at a time) from the input text, each one producing a new perturbed text. For *part-of-speech* features, each part-of-speech removal produces a new perturbed text. Finally, for *sentence-based* features, the removal of each sentence, one at a time, produces a new perturbed text.

Examples of explanations produced by feature removal perturbation are shown in Figures 3b, 3c, and 3d. From the input text in Figure 3a, the words highlighted in Figure 3b, the sentence highlighted in Figure 3c and the words identified by MLWE in Figure 3d are removed. A discussion on these examples is provided in Section 5.1

The *feature substitution perturbation* was also explored by T-EBAnO. While the removal perturbation causes an absence of the concept associated with the removed words, the substitution perturbation introduces a new, possibly related, concept that can cause a change in the prediction. The *feature substitution* perturbation requires an additional step to select new words that will replace the current ones. In T-EBAnO, the substitution of words with their *antonyms* is exploited. This strategy turned out to be very powerful in some specific cases (e.g., Adjective-POS perturbation), but in general, it has several limitations: (i) some words can have many antonyms and the optimal choice might depend on the context, (ii) antonyms do not exist for some words (e.g., nouns), and (iii) the choice of the new words to be inserted in the substitution of the feature is task-specific (e.g., antonyms work with opposite class labels like *Positive* and *Negative* in sentiment analysis, but are not suited with independent class labels as in topic detection). Thus, the effectiveness of this perturbation strategy is affected by these limitations. Figures 3e and 3f show two examples of explanations performed using this technique. For the Adjective-POS features, it is straightforward to find meaningful antonyms. On the contrary, for Verb-POS features, the result is very difficult to evaluate since verbs like {`was, have`} are substituted with {`differ, lack`}. This feature perturbation strategy remains an open task left for further inspection in future works. For instance, we plan to analyze *task-specific* and *expert-driven* substitution perturbations. For example, for a *comment toxicity* classification (i.e., predicting if an input text contains toxic or clean language), the effects of substitution w.r.t. gender, minority, named entity, or other possible biases is of absolute

interest. For a *sentence grammar acceptability* classification task (i.e., predicting if a sentence is grammatically acceptable or unacceptable), introducing expert-driven substitutions to understand if the classifier is robust to critical linguistic aspects is another example. In this paper, such implementations are out of scope because we currently devise T-EBAnO to be as general as possible across different classification tasks without requiring human expertise, and we reach this goal by means of the *removal* perturbation.

4.3 Interpretable feature influence measurement

T-EBAnO exploits an improved version of the quantitative index proposed in [48], namely *normalized Perturbation Influence Relation* (*nPIR*) to measure the influence of each interpretable feature extracted. This improved index solves the issues of *asymmetry* and *unbounded values*, which affect the index previously proposed in [48]. It assesses the importance of an input feature for a given prediction, analyzing its performance before and after the perturbation of a feature (or set of features) extracted from the input data.

Formally, given a model able to distinguish between a set of classes $c \in C$. Let $ci \in C$ be the *class-of-interest* for which the local-explanation has to be computed. Given the input sample I, the explanation process extracts the set of interpretable features F. For each feature $f \in F$, the perturbation is applied, and the reactions of the predictive model are evaluated. These reactions represent the contribution of f to the prediction process. We quantify the influence of f over ci through the *nPIR* index.

Let $p_{o,ci}$ be the output probability of the original input I (the unperturbed input) to belong to the class-of-interest ci, and $p_{f,ci}$ the probability of the same input, with the feature f perturbed, to belong to the same class. Let consider the predicted class distributions as $\sum_c^C \mathbb{P}_{o,c} = 1$ and similarly $\sum_c^C \mathbb{P}_{f,ci} = 1$. For instance, the output of the model is given by a SoftMax layer.

We introduce a generic definition of influence relation for a feature f by combining the outcomes of the model $p_{o,ci}$ and $p_{f,ci}$ before and after the perturbation process. We want such influence relation (i.e., the *nPIR*) to range in the $[-1; 1]$ interval. An *nPIR* value for f close or equal to 1 represents a positive relevance for the concept in f over the prediction of class ci. On the opposite, an *nPIR* value for f close or equal to -1 represents a negative impact of that feature over the prediction of class ci. An *nPIR* value close to 0 means that f is neutral w.r.t. the prediction of class ci.

The *nPIR* derives from the combination of two sub-indicators: the *Amplitude of Influence* ΔI and the *Symmetric Relative Influence* SRI. The ΔI for a feature f is defined as in Equation 1 and ranges from -1 to 1 since the domain for probability values is included in $[0, 1]$.

$$\Delta I_f = p_{o,ci} - p_{f,ci} \tag{1}$$

A $\Delta I_f > 0$ represents a positive influence of the feature f for class ci since the perturbation of the corresponding portion of input causes a decrease of its probability to belong to the class-of-interest. Thus, f is relevant for class ci. Similar reasoning could be made for $\Delta I_f < 0$ representing a negative influence of the feature f for ci.

The amplitude alone does not reflect the overall contribution of f completely. In particular, the absolute distance between two values can be low if the values are small w.r.t. the probability values domain, but, their relative distance can still be significant. This effect should not be ignored as well. Because of this, we need to consider also the relative influence of f. To capture the relative influence of f, a straightforward approach would be to compute the ratio between the probabilities. However, as shown in [48], such score is asymmetric: the ratio $\frac{p_{o,ci}}{p_{f,ci}}$ will range from 0 to 1 in case of negative influence and from 1 to ∞ in the other case. So, it will be difficult to quantitatively compare positive and negative influences. To overcome this problem, we define the *Symmetric Relative Influence* for a feature f as in Equation 2. This index evaluates the relative influence that f has over $p_{o,ci}$ and $p_{f,ci}$. The symmetry of this score allows measuring the relative influence of the feature f before and after the perturbation regardless of its positiveness or negativeness.

$$SRI_f = \frac{p_{o,ci}}{p_{f,ci}} + \frac{p_{f,ci}}{p_{o,ci}} \tag{2}$$

By combining Equations 1 and 2, we define the *Perturbation Influence Relation* for f in the range $(-\infty, +\infty)$. We finally add the *Softsign* [17] function to obtain a linear approximation of the influence close to 0 and to bound in a non-linear way the very high positive or negative values in the $[-1; 1]$ range. Hence, the *normalized Perturbation Influence Relation* ($nPIR$) of a feature f for a class-of-interest ci is defined in Equation 3.

$$\begin{aligned} nPIR_f(ci) &= softsign(\Delta I_f * SRI_f) \\ &= softsign(p_{f,ci} * b - p_{o,ci} * a) \end{aligned} \tag{3}$$

$$a = 1 - \frac{p_{o,ci}}{p_{f,ci}}; b = 1 - \frac{p_{f,ci}}{p_{o,ci}} \tag{4}$$

The coefficient a is the contribution of input o w.r.t. the perturbed input. Similarly, b represents the contribution of the perturbation of f w.r.t. the original feature. The higher the $nPIR_f$ (close to 1), the more the feature f is positively influencing the class-of-interest. On the opposite, the lower the $nPIR_f$ (close to -1), the more the feature f is negatively influencing the class-of-interest.

4.4 Multi-layer Word Embedding (MLWE) feature extraction

In this Section, the terms words and tokens are often used interchangeably. However, the tokenization process of the explained model also drives T-EBAnO. For example, if the tokenizer of the explained model removes the punctuation and stopwords, T-EBAnO-MLWE does not consider it. Otherwise, if the tokenization step keeps the punctuation and stopwords, then also T-EBAnO-MLWE considers them as possible influential tokens/words.

Deep Neural networks are trained to extract knowledge from training data learning a complex numerical model spreading this knowledge on multiple hidden layers. During the prediction process of previously unseen data, all these layers

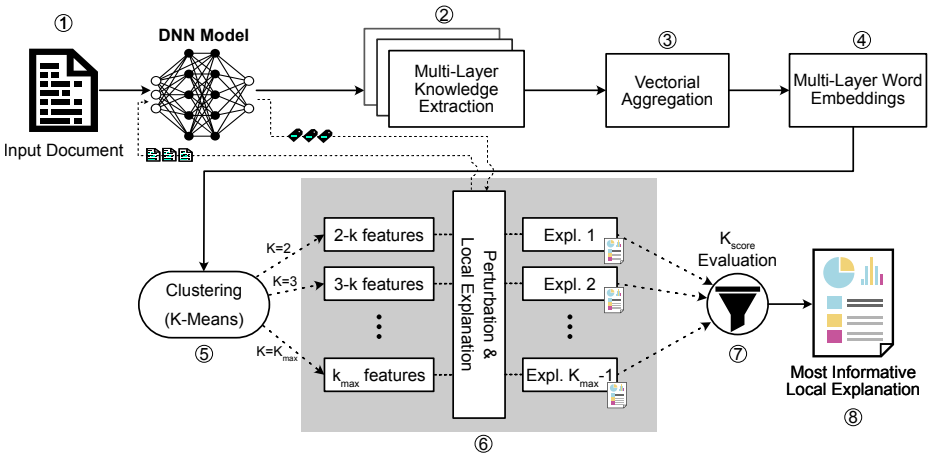


Fig. 2: T-EBAnO MLWE feature extraction process.

contribute to the outcome. Thus, to get a reliable explanation, it is necessary to mine all the knowledge hidden along with the layers of the model. Thanks to the Multi-layer Word Embedding (MLWE) feature extraction, T-EBAnO can achieve this goal. Specifically, T-EBAnO analyzes the outcomes of multiple hidden layers to extract the numerical representation of the input at different levels of the network. The *Multi-layer Word Embedding* feature extraction process is shown in Figure 2.

Embedding knowledge extraction and aggregation.
Firstly, given an input document ①, a tensor containing the numerical embedding representations of different tokens in different layers is extracted ②. Then, the intermediate embeddings of each layer are aggregated (e.g., through average, sum or concatenation) on the layers' axis to a single-layer vector representation for each token. Then, only in the case of sub-words representation, another aggregation is performed to reconstruct full-words from the sub-words tokens. The aggregation of the multiple channels' and the sub-tokens representations compose the *vectorial aggregation* step and depends on the specific model's architecture. Finally, their dimensionalities are further reduced through PCA to obtain an embedding vector representation for each input token ③. The outcomes of the vectorial aggregation and the dimensionality reduction steps are the *Multi-layer Word Embedding* representation of the input document ④, where each full-token is represented with a small and dense vector that approximates the meaning and knowledge learned by the model. The intuition is that words with a similar MLWE representation are considered highly correlated by the model, and, if grouped together, they represent key input concepts that most probably are influencing the current prediction. The MLWE feature extraction, and in particular the extraction of the aggregated word embeddings from multiple layers, has to be achieved in different ways depending on the neural network architecture under the exam. Further details about MLWE feature extraction tailored to LSTM and to BERT and how MLWE fits other NLP

architectures are provided in Section 6.2.

Unsupervised embedding analysis.
Once the MLWEs are extracted, they are analyzed through an *unsupervised clustering* analysis ⑤ to identify sets of correlated words that share common behaviors inside the model under exam. Specifically, the unsupervised analysis aims to identify the smallest groups of input words (tokens) that have the highest impact on the model outcome. For this purpose, T-EBAnO exploits the K-Means [31] clustering algorithm since it provided good performance in a similar context [48] and represents a good trade-off with computational time. A critical parameter when dealing with K-Means is setting the desired number of groups K to correctly model interesting subsets of data. T-EBAnO applies K-Means to identify a number of groups ranging in $[2, K_{max}]$, where the max number of clusters K_{max} is a function of the input size and has been empirically set to:

$$K_{max} = \sqrt{n_{tk} + 1} \tag{5}$$

On the one hand, using small fixed values of K with large input texts leads to large clusters of words containing both influential and less impacting words, and consequently, the explanation provided will be of low interest. On the other hand, the number of tokens n_{tk} in a text can be very high, and it would be neither feasible nor useful to evaluate partitioning that takes into account values of K as large as the number of tokens n_{tk}. For this reason, the evaluation of a number of clusters K that is at most equal to the root of the number of tokens n_{tk} in a text allows maintaining a good trade-off between partitioning size and performance. This allows for reducing the search space, without affecting the quality of the features. T-EBAnO produces a *quantitative explanation* (as detailed in Section 5.1) exploiting the *normalized Perturbation Influence Relation* (nPIR) index (introduced in Section 4.3) for each K ⑥. Specifically, for each value of $K \in [2, K_{max}]$, K perturbations will be analyzed, each one producing a new version of the input text applying the perturbation over the tokens of the current cluster. Then, the outcomes of the model by presenting the new perturbed texts are evaluated, producing the nPIR index for each cluster perturbation of each possible K (dot lines in ⑥). In this way, a large number of potentially useful *local explanations* are produced by T-EBAnO.

Most informative local explanation evaluation.
The objective, however, is to provide only the best explanation to the end-user. T-EBAnO selects the *most informative local explanations* as those extracting the most valuable knowledge from the behavior of the model over a single prediction. To this aim, firstly, T-EBAnO assigns a *feature informative score (FIS)* to each feature (i.e., each cluster of words), exploiting the nPIR index, as follows:

$$FIS(\kappa) = \max\Big(\big(\alpha(nPIR_\kappa) + \beta(1 - \kappa_{tk}/n_{tk})\big), 0\Big) \tag{6}$$

Where κ is the current cluster, κ_{tk} is the number of tokens inside the cluster κ, n_{tk} is the total number of tokens and $nPIR_\kappa$ is the influence score of the current cluster κ, which measures the positive or negative influence of perturbing the tokens in κ (as discussed in Section 4.3). The ratio κ_{tk}/n_{tk} represents the percentage of tokens inside the cluster over the total number of tokens. The $FIS(\kappa)$ score tends

to *maximize the influence* of the feature ($nPIR$) and *minimize the size* of the feature κ_{tk}/n_{tk} (maximizing $(1-(\kappa_{tk}/n_{tk}))$).

The hyper-parameters α and β are the *weights* assigned respectively to the $nPIR$ and the tokens ratio score $(1-(\kappa_{tk}/n_{tk}))$. They determine the relative contribution of the *influence* of the feature and its *size*. In our settings, we assigned a weight of 0.60 to the *influence* and 0.40 to the *size* of the features ($\alpha = 0.60$ and $\beta = 0.40$) because selecting influential features is of prevalent importance, and only secondly, we would like to minimize the number of tokens. On the contrary, selecting small-size features which are not influential would be useless. An experimental evaluation of α and β hyperparameters is provided in Section 6.7.

The range of nPIR is [-1,1] (as discussed in section 4.3), where 1 indicates a very high positive influence for the class of interest. The range of $(1-(\kappa_{tk}/n_{tk}))$ is [0,1]. Therefore, the *feature informative score FIS*, with $\alpha = 0.60$ and $\beta = 0.40$ (or any values of α and β whose sum is 1), is in the range [0,1]. The negative values are undesired because we are looking for positively influential features for the class of interest. A $FIS = 0$ is obtained by a feature whose size is towards the 100% of the tokens and whose influence is towards 0. A $FIS = 1$ is obtained by a feature with few tokens (e.g., less then 1%) and with an influential score towards 1. The higher the FIS score, the more informative and shorter the feature is.

Then, for each value of K (i.e., each possible partition analyzed), a score is computed ⑦ by taking the max of the FIS score over its clusters of words.

$$\begin{aligned} K_{score} &= \max_{\kappa \in K} \Big(FIS(\kappa) \Big) \\ &= \max_{\kappa \in K} \Big(\max \big((\alpha(nPIR_{\kappa}) + \beta(1-\kappa_{tk}/n_{tk})), 0 \big) \Big) \end{aligned} \tag{7}$$

The K with the highest K_{score} is selected as the best. Hence, K clusters of words are created, with each $\kappa \in K$ being a feature including some neutral features (or negative influential, i.e., $nPIR \leq 0$) and, generally, one very highly influential feature. Finally, the cluster κ with the highest $FIS(\kappa)$ will be the *most informative local explanation* ⑧.

Table 2 shows an example of the analysis made by T-EBAnO using MLWE with a short input text consisting of 9 tokens predicted with the label *Positive* with high confidence (≈ 0.99) in a sentiment analysis task. The column K represents the different numbers of clusters analyzed by T-EBAnO, cluster $\kappa \in K$ is denoted as $K.k$ (e.g., 2.1 is the first cluster of the division $K = 2$), $\mathrm{k_{tk}}$ represents the number of tokens inside the cluster, $\mathrm{k_{tk}/n_{tk}}$ represents the ratio between the tokens in the cluster and the total number of tokens, *nPIR* and *FIS* are the influence score and the feature informative score obtained by cluster *k*, respectively. The tokens of each cluster are highlighted in cyan in the input text (column *Highlighted Clusters*).

The partitions analyzed by T-EBAnO are $K \in [2, 3]$ ($K_{max} = \sqrt{9+1} \approx 3$). The first partition ($K = 2$) finds two clusters of tokens: *cluster 2.1* containing 3 tokens and *cluster 2.2* containing 6 tokens (highlighted in cyan). The current *most informative local explanation* is *cluster 2.1*, because it has the highest *FIS* score among the clusters of $K = 2$. Then, T-EBAnO analyzes the clustering results with

K	k	Highlighted Clusters	k_{tk}	k_{tk}/n_{tk}	nPIR	FIS
2	2.1	Yesterday I saw a movie that positively surprised me	3	3/9	0.990	0.861
2	2.2	Yesterday I saw a movie that positively surprised me	6	6/9	0.001	0.134
3	3.1	Yesterday I saw a movie that positively surprised me	2	2/9	0.999	**0.911**
3	3.2	Yesterday I saw a movie that positively surprised me	4	4/9	0.001	0.228
3	3.3	Yesterday I saw a movie that positively surprised me	3	3/9	0.000	0.267

Table 2: Example of the *most informative local explanation*(*cluster 3.1*) and *best K division*($K = 3$) selection using MLWE for an input text with 9 tokens predicted as *Positive* by an NLP model fine-tuned for sentiment analysis

.

$K = 3$. The current *most informative local explanation* is *cluster 3.1*, because it has the highest *FIS* score among the clusters of $K = 3$. Overall, the local explanation *cluster 3.1* has a higher FIS score than *cluster 2.1* ($0.911 > 0.861$), then $K = 3$ is selected as the best K value, and *cluster 3.1* is the final *most informative local explanation.*

5 Explanations

This Section presents the prediction-local (Section 5.1) and the model-global (Section 5.2) explanation processes implemented in T-EBAnO.

5.1 Prediction-local explanations

To produce the local explanations, T-EBAnO exploits the outcomes of the model when fed with the original input and its perturbed versions. A local explanation consists of two main parts: a *textual explanation* (Figure 3) and a *quantitative explanation* (Table 3), as detailed in the following.

Textual explanation.
The *textual explanation* highlights the most relevant sets of features for the model under analysis, also allowing the understanding of the context in which they appear. Many sets of features can be extracted for each interpretable feature extraction technique. Figure 3 shows a simple example of textual explanations. For this example, the BERT model has been trained to detect the sentiment of a textual document, either positive (P) or negative (N). Given the input document in Figure 3a, the model outputs a negative sentiment. So, the user can inspect the highlighted features (in cyan) in the textual explanations in Figures 3b, 3c, 3d, 3e, and 3f to find out which are the most important sections of the input that have been exploited by the model to make its decision.

Quantitative explanation.
The *quantitative explanation* shows the influence of each set of extracted features

This film was very awful. I have never seen such a bad movie.

(a) Original text

This film was very awful. I have never seen such a bad movie.

(b) EXP1: Adjective - POS feature extraction with removal perturbation.

This film was very awful. I have never seen such a bad movie.

(c) EXP2: Sentence feature extraction with removal perturbation.

This film was very awful. I have never seen such a bad movie

(d) EXP3: Multi-layer word embedding feature extraction with removal perturbation.

This film was very [awful] nice. I have never seen such a [bad] good movie

(e) EXP4: Adjective-POS feature extraction with substitution perturbation.

This film [was] differ very awful. I [have] lack never seen such a bad movie.

(f) EXP5: Verb-POS feature extraction with substitution perturbation.

Fig. 3: Examples of a *textual explanation* report. The original text was labeled by BERT as *Negative* with a probability of 0.99. The most relevant features are reported and highlighted in cyan. Removed tokens for the substitution perturbation are in squared brackets and followed by the new inserted tokens.

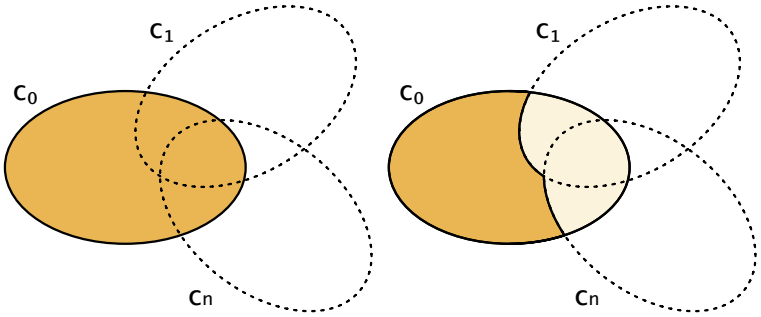
Explanation	Feature f	L_o	L_f	$nPIR_f(N)$
EXP1	POS-Adjective	N	P	**0.998**
EXP2	Sentence	N	N	0.000
EXP3	MLWE	N	P	**0.984**
EXP4	POS-Adjective (sub.)	N	P	**0.999**
EXP5	POS-Verb (sub.)	N	N	0.000

Table 3: Quantitative explanation for example in Figure 3. P is the positive label, N is the negative label. The (sub.) suffix indicates that the substitution perturbation has been applied. Otherwise the removal perturbation has been applied.

for the prediction (separately) by evaluating the *nPIR* index (*normalized Perturbation Influence Relation*) introduced in Section 4.3. The *nPIR* index is computed by T-EBAnO for each feature extracted by all the feature extraction techniques, for the class-of-interest (usually the predicted label for the input text).

Exploiting the $nPIR$ index, we can define *thresholds* to identify highly influential and *informative* explanations. For instance, considering a threshold $nPIR_t > 0$, if $-nPIR_t \leq nPIR_f \leq nPIR_t$, then the difference between the probabilities before and after the perturbation of f could be considered not sufficiently informative. Instead, values of $nPIR_f < -nPIR_t$ (or $nPIR_f > nPIR_t$) mean that the perturbation of feature f is contributing negatively (or positively) to the prediction by decreasing (or increasing) the probability of belonging to the class-of-interest.

Table 3 shows the *quantitative explanations* for the *textual explanations* in Figure 3. For each *interpretable feature* f the labels assigned by the model before and



(a) Absolute influential words (GAI). (b) Relative influential words (GRI).

Fig. 4: Influential set of words at model-global level for class-of-interest c_0.

after their perturbation are reported in columns L_o and L_f respectively along with the $nPIR$ value calculated for the class-of-interest negative (N). Perturbing the POS adjectives in Figure 3b (EXP1) or the MLWE cluster in Figure 3d (EXP3) the $nPIR$ is very close to 1. This means that these sets of features are very relevant for the model outcome: *removing* one of these features will cause completely different outcomes from the model, changing the prediction from negative (N) to positive (P). Instead, the perturbation of the sentence in Figure 3c (EXP2) is not relevant at all for the model, showing a value of $nPIR$ equal to 0. We can conclude that the feature sets {`awful, bad`} and { `was, awful, bad, movie` } are the real reason why the model is predicting the negative class. The information contained in the sentence { `This film was very awful` } instead does not justify the model outcome alone, like the rest of the text that is also contributing to the prediction. The *quantitative explanations* obtained through the substitution perturbation (EXP4 and EXP5) have been also reported in Table 3. Even from these results, it is evident that the substitution perturbation has great potential in expressiveness when it is possible to find suitable antonyms. In the case of Adjective-POS substitution (EXP4), the quantitative explanation shows a $nPIR$ value close to 1. On the contrary, in the case of EXP5, verbs are replaced with semantically incorrect words (not antonyms) in the context of the phrases, showing no impact in the prediction process with a $nPIR$ equal to 0. Therefore, as discussed in Section 4.2, this perturbation strategy remains an open task left for further inspection in future works.

5.2 Per class model-global explanation

T-EBAnO is able to provide per-class *model-global* explanations of the prediction process. The local explanations computed for a corpus of input documents are aggregated and analyzed together, highlighting possible misleading behaviors of the predictive model.

Two indices have been introduced to measure the global influence of the corpus of input documents: (i) the *Global Absolute Influence* (GAI) described by Algorithm 1, and (ii) the *Global Relative Influence* (GRI) defined in Equation 8.

The GAI score measures the global importance of *all* the words impacting the class-of-interest, without distinction concerning other classes (Figure 4a). On the other hand, the GRI score evaluates the relevance of the words influential *only* (or mostly) for the class-of-interest, differently from other classes (Figure 4b).

The global explanations are computed for each available class $c \in C$, analyzing the set of local explanations produced by T-EBAnO from a dataset of input texts D. For each document $d \in D$, the set of local explanations E_d are produced, where each explanation $e_{d,f} \in E_d$ is the explanation computed over the feature f (e.g., cluster of tokens) containing the list of tokens of the current feature and their influence value (nPIR). Only MLWE explanations are exploited to produce the global explanations since it is the only feature extraction strategy that exploits inner model knowledge (as discussed in Section 4.4).

Algorithm 1: Global Absolute Influence.

Input: Dataset D, Classes C .
Output: GAI score $\forall$ class label $c \in C$ and lemma $l \in L$.

1 $GAI \leftarrow$ initHashMap(0);
2 $PredictionsCounter(C) \leftarrow$ init(0);
3 $L \leftarrow$ empty list;
4 **for** d ***in*** D **do**
5 $\quad \hat{c} \leftarrow$ Model.Predict(d);
6 $\quad PredictionsCounter(\hat{c}) \leftarrow PredictionsCounter(\hat{c}) + 1$;
7 $\quad E_d \leftarrow$ T-EBAnO.LocalExplanation($Model$, d, $\hat{c}$);
8 $\quad \hat{e}_{d,f} \leftarrow$ T-EBAnO.GetMostInfluentialExplanation(E_d, $\hat{c}$, "MLWE");
9 $\quad$ **for** tk ***in*** $\hat{e}_{d,f}.featureTokens$ **do**
10 $\quad\quad l \leftarrow$ Lemmatize(tk);
11 $\quad\quad L$.insert(l);
12 $\quad\quad GAI(\hat{c}, l) \leftarrow GAI(\hat{c}, l) + Max[0, \hat{e}_{d,f}.nPIR]$;
13 $\quad$ **end**
14 **end**
15 **for** c ***in*** C **do**
16 $\quad$ **for** l ***in*** L **do**
17 $\quad\quad GAI(c, l) \leftarrow GAI(c, l)/PredictionsCounter(c)$;
18 $\quad$ **end**
19 **end**
20 **return** GAI;

Global Absolute Influence.
The Global Absolute Influence value is computed following the process described in Algorithm 1. Firstly, are initialized the HashMap containing the GAI score for each class C and each lemma L (line 1), the counter of predictions for each class (line 2) and the list of unique lemmas (line 3). Then, given a corpus of documents D, for each input textual document $d \in D$ the following steps are repeated (line 4). Firstly, the estimated class label $\hat{c} \in C$ for the input text d is predicted by the DNN model to explain (line 5) and the counter value for the class $\hat{c}$ is incremented (line 6). Then, T-EBAnO produces the local explanation set E_d for the input d and the class-of-interest $\hat{c}$ (line 7). Thus, the most influential explanation $\hat{e}_{d,f}$, i.e., the one with the highest $nPIR$, is selected (line 8). We recall that T-EBAnO exploits only the MLWE features to produce the global explanations. Therefore, the most

influential explanation $\hat{e}_{d,f}$ is the cluster of tokens with highest influence, measured with the nPIR, for the original predicted class label $\hat{c}$. Finally, for each token tk belonging the most influential feature $\hat{e}_{d,f}.featureTokens$ (line 9), T-EBAnO extracts the lemma l (line 10) of each token tk, adds it to the list of unique lemmas L (line 11) and updates the GAI score $GAI(\hat{c}, l)$ for the class $\hat{c}$ of the lemma l (line 12) by summing the $nPIR$ score of the the explanation $\hat{e}_{d,f}$, only if it is positively impacting the prediction (i.e., if $nPIR > 0$). The algorithm analyzes *lemmas* instead of *tokens* (words) in order to group together their inflected forms, obtaining more significant results. Finally, T-EBAnO normalizes the GAI score of each lemma $l \in L$ and each class $c \in C$ dividing by the number of inputs predicted with the class label c (lines 15,16,17). This normalization step is required to handle also unbalance classes cases. The output of the algorithm is the set of *Global Absolute Influence* scores. Specifically, for each lemma found in corpus D, a GAI score is computed for each possible class $c \in C$. The value $GAI(c, l)$ is in range $[0, +\infty]$ and measures the absolute global influence of the lemma l for the class c. Notice that, in the current T-EBAnO implementation, the GAI score can exceed 1 because if a lemma is present n times in an influential feature, its global score is updated by summing the $nPIR$ n times. We could obtain a score in range $[0, 1]$ by taking the list of *unique* lemmas of the feature (in lines 9 and 10). However, we preferred to reward lemmas that are highlighted multiple times as important for the prediction in a single explanation.

In conclusion, the GAI score, will be 0 for all the lemmas that have always brought a negative influence on class c, and it will grow proportionally to the frequency and to the positive influence of each lemma positively influencing class c. The higher the GAI score, the most positively influential a lemma is for the model under analysis with respect to class c.

Global Relative Influence.
The Global Relative Influence score highlights the most influential and differentiating lemmas for each class-of-interest, discarding lemmas with multiple impact on other classes. The GRI for a class-of-interest c, for a specific lemma l, and for a classification task with n_C classes is defined as:

$$GRI(c, l) = Max\left[0, \left(GAI(c, l) - \sum_{c_i \neq c}^{C} GAI(c_i, l)/(n_C - 1)\right)\right] \tag{8}$$

The GRI score is 0 when a lemma is more relevant for other classes than for the one under exam, while $GRI > 0$ if its influence is higher for class c than all the other classes. The higher the GRI value, the more specific the lemma influence is with respect to the class-of-interest. The normalization over the number of predicted samples for each class performed on the GAI allows the GRI to be fair in case of unbalanced classes, while the division by $n_C - 1$ allows handling multi-class tasks.

Analyzing GAI and GRI scores, the user can extrapolate which are the most relevant inter- and intra- class semantic concepts that are affecting the decision-making process at a model-global level. For example, if a word is influential for all the possible classes, it will have a high GAI score and a GRI score close to 0 for all the classes. On the contrary, if a word is most influential for a specific class,

the *GAI* score will be higher for that specific class. Therefore, the *GRI* score for that class will be greater than 0 for that class and usually 0 (or close to 0) for all the other classes. Section 6.4 provides an experimental analysis of the insights provided by T-EBAnO at a model-global level.

6 Experimental results

In this section, we present the experiments performed to assess the ability of T-EBAnO to provide useful and human-readable insights on the decisions made by deep learning NLP models. Firstly, we describe the experimental use cases in terms of NLP models and datasets (Section 6.1). Before discussing the core results, i.e., the explanations, we show how *MLWE* adapts to different NLP-model architectures (Section 6.2). The effectiveness of T-EBAnO in extracting useful *local explanations* is presented in Section 6.3, whereas results for *global insights* are discussed in Section 6.4. Then, we evaluate the application of T-EBAnO to different use cases (Section 6.5), the *effectiveness* of *MLWE* with respect to a random choice of the features (Section 6.6), and we perform a hyperparameters analysis (Section 6.7). Finally, we evalaute the capacity of the proposed influence index (*nPIR*) to model the human judgment (Section 6.8), and we perform an experimental comparison with two model-agnostic XAI techniques (Section 6.9).

6.1 Use cases

To discuss how T-EBAnO is able to provide useful *prediction-local explanations* and *model-global explanations*, we selected two main use cases consisting of different NLP models and classification tasks (*Use cases 1-2*). We chose a sequence model and a transformer-based model from the state-of-the-art, specifically LSTM and BERT, applied on two different binary text classification tasks: sentiment analysis and toxic comment classification. Then, to evaluate the flexibility of T-EBAnO, independently of the specific deep learning model and the classification task, we selected additional classification tasks (*Ag News* topic classification and *Cola* sentence acceptability) on different models like BERT, ALBERT, and ULMFit (*Use cases 3-8*). The removal perturbation has been exploited for all the experiments. Table 4 summarize all the experimental use cases.

Use case 1. The first task is a binary *toxic comment classification*, and it consists of predicting whether the input comment is *clean* or *toxic*, i.e., it contains inappropriate content. The *toxic* class label contains several subtypes of toxic comments such as identity attacks, insults, explicit sexuality, obscenity, insult, and threats. An LSTM model applied to a civil comments dataset [7] has been used. The LSTM model is composed of an embedding 300-dimensional layer, two bidirectional LSTM layers (with 256 units for each direction), and finally, a dense layer with 128 hidden units. Transfer learning has been exploited using GloVe [38] (with 300-dimensional vectors) for the embedding layer. After training, the custom LSTM model reached an accuracy of 90%.

Use case 2. The second selected task is *sentiment analysis*, and it consists of predicting if the underlying sentiment of an input text is either positive or negative.

Use case	Model	Dataset	Task (Classification)	Test accuracy
1	LSTM	Civil Comments	Comment Toxicity	90%
2	BERT	Imdb	Sentiment Analysis	86%
3	BERT	Ag News	Topic Classification	94%
4	BERT	Cola	Sentence Acceptability	81%
5	ALBERT	Ag News	Topic Classification	93%
6	ALBERT	Cola	Sentence Acceptability	77%
7	ULMFit	Ag News	Topic Classification	92%
8	ULMFit	Cola	Sentence Acceptability	71%

Table 4: Experimental use cases.

The BERT base (uncased) pre-trained model [12] has been chosen as deep learning predictive model with obscure decision-making process, and it has been applied to the *IMDB dataset* [33], which is a reference set of data for sentiment analysis. We performed a fine-tuning step of the BERT model [12] by adding a classification layer on top of the last encoder transformer's stack. The BERT model, fine-tuned on the IMDB textual reviews, reached an accuracy of 86%.

Other use cases. For the additional use cases, we selected different models and classification tasks. We kept BERT from the transformer-encoder family of models as a reference milestone of the state-of-the-art, then we added ALBERT [25] as a representative of the variations proposed for the BERT model (like RoBERTa [30], DistilBERT [42]), and ULMFit [20] as a representative of the general language model family, with a completely different architecture. The two additional tasks are (i) a binary classification, predicting the grammatical acceptability or unacceptability of the sentence with the *Cola* (Corpus of Linguistic Acceptability) dataset [50], and (ii) a multi-class news topic classification task consisting of four classes (*World*, *Sport*, *Business* and *Science/Technology*) of the *Ag News* dataset [51] (a subset version with the 4 largest classes of the original corpus).

6.2 Multi-layer word embedding model-specific implementations

In this section, we discuss the model-specific MLWE implementations for the deep learning models used in the experimental use cases.

LSTM. RNNs with LSTM units are robust architectures that can learn both the time sequence dimension and the feature vector dimension. Multiple LSTM layers usually characterize them, and they can take as input an embedded representation of the text. As highlighted in Section 6.1, the developed LSTM model exploits one embedding layer that works with full tokens and two bidirectional LSTM layers. For these reasons, the MLWE exploits the single embedding layer to extract a tensor of shape $(tk \times 300 \times 1)$. In this case, the *vectorial aggregation* step (Section 4.4) is unnecessary because the embedding is extracted from a single layer, and the model does not present sub-tokens. Then, a Principal Component Analysis is used to reduce the embedding matrix shape to $(tk \times c)$, obtaining the *multi-layer word embedding* representation for the custom LSTM model.

BERT. Figure 5 shows all the steps of the *multi-layer word embedding* (MLWE) feature extraction process in BERT. The *base* version of the BERT model [12]

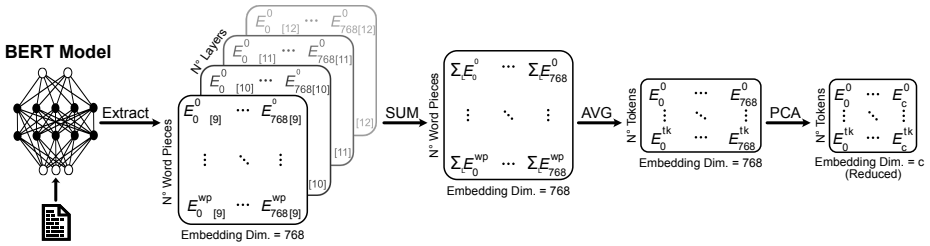


Fig. 5: BERT multi-layer word embedding feature extraction process. With: $E^{<wp \text{ or } tk>}_{<768 \text{ or } c>\ [L_{id}]}$ such that: E is the word embedding matrix, wp and tk indicate the position of the word-piece and token respectively in the input text, 768 is the original embedding dimension, c is the number of reduced principal components of the word embedding vector and L_{id} is the layer from which is extracted.

is composed of 12 transformer layers [47], each producing an output of shape $(wp \times 768)$, where wp is the number of word pieces extracted by BERT in its pre-processing phase. The MLWE, in this case, analyzes the word embeddings extracted from the last four transformer layers of the model. It has been motivated in literature [14] that modern natural language models incorporate most of the context-specific information in the last and deepest layers. Thus, the joint analysis of these layers allows the MLWE to extract features more related to the task under exam, avoiding too specific (if analyzing only the last layer) or too general (if analyzing only the first layers) word embeddings. In the first step of the MLWE feature extraction, the last four transformer layer outputs (i.e., L_9, L_{10}, L_{11}, L_{12}) are extracted (Figure 5-left), resulting in a tensor of shape $(wp \times 768 \times 4)$. Each row is the embedding representation for each word piece in each layer. Then, the outputs of the four layers are aggregated, summing the values of the embeddings over the layer axes in a matrix of shape $wp \times 768$ (Figure 5-center-left), as suggested by [15]. Since BERT works with word pieces but T-EBAnO objective is to extract full tokens (words), the embedding of word pieces belonging to the same word are aggregated, averaging their values over the word-piece axes, and obtaining a new matrix of tokens embedding of shape $tk \times 768$, where tk is the number of input tokens (Figure 5-center-right). The 4-layers to single-layer and word-pieces to full-tokens aggregations compose the *vectorial aggregation* step (Section 4.4) for the BERT model. In the end, due to the sparse nature of data, the dimensionality reduction technique, i.e., Principal Component Analysis, is exploited, reducing the final shape of the tokens embeddings matrix to $(tk \times c)$, where c is the reduced number of principal components extracted (Figure 5-right). This last result is the *Multi-layer word embedding* representation for the BERT model.

In general, to adapt T-EBAnO to different NLP deep-learning architectures, the MLWE approach requires providing one or more layers of word embedding (a vector or a tensor for each token), an aggregation function if there are more layers of word embedding (i.e., to represent each token from the n-dimensional tensor to a 1-dimensional vector) and, finally, an aggregation function if wordpieces tokenization is performed (i.e., some tokens are divided into sub-tokens) to create full tokens representations instead of wordpieces representations (vectorial aggregation step).

Feature extraction type	No combination	Pairwise combination
Part-of-speech	33%	70%
Sentence	22%	30%
MLWE	75%	86%
Overall	80%	90%

Table 5: Explanations of the BERT model: percentage of documents for which each feature extraction strategy produces at least one highly influential local explanation (i.e., with $nPIR \geqslant 0.5$), with and without combination of features. The *pairwise combinations* are inner feature extraction methods (like *Adjs* with Verbs for *POS*, *Sentence 1* with *Sentence 2* for *SEN* and *Cluster 1* with *Cluster 2* for *MLWE*). *Overall* is the percentage of documents for which at least one method provided a local explanation with $nPIR \geqslant 0.5$.

T-EBAnO provides an interface to be implemented with such specifications, hence allowing T-EBAnO-MLWE to potentially work with any NLP deep-learning model. This interface has been used to exploit MLWE with all the models included in the experiments (LSTM, BERT, ALBERT, ULMFit). For instance, the MLWE implementation for the ALBERT architecture is exactly the same as used for BERT. It extracts the last four transformer-encoder layers, aggregates the multi-layer to a single vectorial representation for each wordpiece (*sum*), and, finally, aggregates the wordpieces vectorial representation to full token representation (*avg*) before the dimensionality reduction. For the ULMFit model, instead, the MLWE implementation is very similar to the LSTM implementation. T-EBAnO extracts a representation for each input token by the LSTM-encoder part of ULMFit (a vector of length 400 for each token) and then applies the dimensionality reduction. The two aggregation functions, in this case, are not necessary because a single-layer representation is extracted for each token, and ULMFit already works with full tokens.

6.3 Local Explanations

For each input document, the local explanations were computed exploiting all the feature extraction methods described in Section 4.1 and the *removal perturbation* for use cases 1 and 2.

Overview of use cases 1 and 2.
In the explanation process of the sentiment analysis task with the BERT model (use case 2), T-EBAnO has been experimentally evaluated on 400 textual documents, 202 belonging to the class *Positive* and 198 to the class *Negative*, for a total of almost 100,000 local explanations, with an average of 250 local explanations for each input document. However, only the highly influential local explanations are automatically shown by the engine to the user. A local explanation has been defined to be *highly influential* when having an $nPIR$ value equal to or higher than the threshold $nPIR_t = 0.5$. All the rest of the local explanations produced by T-EBAnO are still available to the users, should they like to investigate further insights into the prediction process. To show the effectiveness of the proposed feature extraction techniques, we analyzed the percentage of documents for which T-EBAnO computed local explanations with at least one highly influential feature

Feature Extraction Type	Clean	Toxic	Clean/Toxic
Part-of-speech	8%	98%	53%
Sentence	2%	76%	39%
MLWE	12%	98%	55%
Overall	15%	99%	58%

Table 6: Explanation of the custom LSTM model: percentage of documents for which each feature extraction strategy produces at least one highly influential local explanation (i.e., with $nPIR \geqslant 0.5$), with combination of features, for the class labels *Clean* and *Toxic* separately, and together (*Clean/Toxic*). The *pairwise combinations* are inner feature extraction methods (like *Adjs* with Verbs for *POS*, *Sentence 1* with *Sentence 2* for *SEN*, and *Cluster 1* with *Cluster 2* for *MLWE*).

for the class-of-interest. Experiments on the same input texts have been repeated twice, firstly without combining the different features, then including the pairwise combinations for each feature extraction method. Table 5 shows the percentage of documents required to find at least one highly influential feature ($nPIR \geqslant 0.5$) with and without combinations of pairwise features. The MLWE method leads to abundantly better results than the other methods. The part-of-speech strategy benefits the most from the pairwise combinations, allowing the creation of features representing more complex concepts. For example, the combination of *adjectives* and *nouns* allows the creation of features composed of words like {`bad, film`} that, together, can better express a sentiment.

In the explanation process of the toxic comment task with the custom LSTM model (use case 1), T-EBAnO has been experimentally evaluated on 2250 documents, 1121 belonging to the class *Toxic* and 1129 to class *Clean*, leading to almost 160,000 local explanations in total. Table 6 shows the percentage of input documents for which T-EBAnO has been able to extract at least one highly influential local explanation ($nPIR \geqslant 0.5$). For the *Toxic* class, T-EBAnO has been able to identify at least one highly influential explanation for almost all the documents, with most of the feature extraction strategies. Only the sentence-based feature extraction has a lower percentage of highly influential explanations w.r.t. the other techniques. This suggests that toxic words tend to be sparse in the input text and not concentrated in a single sentence. Finding highly influential explanations for the *Clean* class has proven to be harder. None of the feature extraction techniques can explain more than 15% of the predictions for the *Clean* input texts. The nature of the use case under exam can explain possible causes: usually, a document is considered clean; it can become toxic because of the presence of specific words or linguistic expressions. Thus, the hypothesis is that there is no specific pattern of words that represents the *Clean* class (see Section 6.4 for further details).

In the following, we present and discuss some specific local explanations provided by T-EBAnO in different conditions to show their relevance and usefulness in explaining the deep NLP model behavior for both the custom LSTM and the BERT models of the use cases 1 and 2.

Criticize a black man and the left calls you a racist. Criticize a woman and you are a sexist. Now I will criticize you as a fool and you can call me intolerant.

(a) Original text

Criticize a black man and the left calls you a racist. Criticize a woman and you are a sexist. Now I will criticize you as a fool and you can call me intolerant.

(b) `EXP1:` Adjective & Noun - POS feature extraction

Criticize a black man and the left calls you a racist. Criticize a woman and you are a sexist. Now I will criticize you as a fool and you can call me intolerant.

(c) `EXP2:` Multi-layer word embedding feature extraction

Fig. 6: Examples of *textual explanation* report for the input in Figure 6a originally labeled by custom LSTM model as *Toxic* with a probability of 0.98. The most relevant features are highlighted in cyan.

Explanation	**Feature f**	$\mathbf{L_o}$	$\mathbf{L_f}$	$\mathbf{nPIR_f(N)}$
EXP1	POS-Adj&Noun	T	C	**0.839**
EXP2	MLWE	T	C	**0.883**

Table 7: Quantitative explanation for the example reported in Figure 6. T is the *Toxic* label, C is the *Clean* label. Positively highly influential features ($nPIR \geq 0.5$) for the L_o class are highlighted in green in the $nPIR_f(N)$ column.

6.3.1 Local Explanation: Example 1

In the first example, reported in Figure 6, the custom LSTM model classifies the input comment in Figure 6a as *Toxic*. The most influential features identified by T-EBAnO are shown in Figures 6b and 6c. The different feature extraction strategies find that the most positively influential features for the *Toxic* class labels are {`black man, left, racist, woman, sexist, fool, intolerant`}. In particular, the most influential explanations are extracted with the combination of adjectives and nouns (Table 7-EXP1) and with MLWE (Table 7-EXP2). It is interesting to notice that in this case, the combination of *adjectives* and *nouns* is very relevant for this model, e.g., it is not just the word `black` that makes a comment toxic, but the combination `black man`. Furthermore, the POS feature extraction and the MLWE highlighted very similar sets of words. In this case, the prediction is trustful, and in particular, it is relevant that the model learned features like `black man` and `woman` to be influential for the *Toxic* class.

6.3.2 Local Explanation: Example 2

In the second example, the BERT model makes a wrong prediction by classifying the sentiment of the input text in Figure 7a as *Negative*, while the expected label (ground-truth) is *Positive*. A user requiring to decide whether to trust such prediction can take advantage of T-EBAnO to understand which are the words influencing the outcome. Figure 7 shows the *textual explanations* provided by the most influential features. Table 8 contains the corresponding *quantitative explanations* with the $nPIR$ values. T-EBAnO identified three local explanations for the

How many movies are there that you can think of when you see a movie like this? I can't count them but it sure seemed like the movie makers were trying to give me a hint. I was reminded so often of other movies, it became a big distraction. One of the borrowed memorable lines came from a movie from 2003 - Day After Tomorrow. One line by itself, is not so bad but this movie borrows so much from so many movies it becomes a bad risk. BUT... See The Movie! Despite its downfalls there is enough to make it interesting and maybe make it appear clever. While borrowing so much from other movies it never goes overboard. In fact, you'll probably find yourself battening down the hatches and riding the storm out. Why? ...Costner and Kutcher played their characters very well. I have never been a fan of Kutcher's and I nearly gave up on him in The Guardian, but he surfaced in good fashion. Costner carries the movie swimmingly with the best of Costner's ability. I don't think Mrs. Robinson had anything to do with his success. The supporting cast all around played their parts well. I had no problem with any of them in the end. But some of these characters were used too much. From here on out I can only nit-pick so I will save you the wear and tear. Enjoy the movie, the parts that work, work well enough to keep your head above water. Just don't expect a smooth ride. 7 of 10 but almost a 6.

(a) Original text

How many movies are there that you can think of when you see a movie [...] I was reminded so often of other movies, it became a big distraction. One of the borrowed memorable lines came from a movie from 2003 - Day After Tomorrow. One line by itself, is not so bad but this movie borrows so much from so many movies it becomes a bad risk. BUT ... See The Movie! Despite its downfalls there is enough to make it interesting and maybe make it appear clever. While borrowing so much from other movies it never goes overboard. [...] I have never been a fan of Kutcher 's and I nearly gave up on him in The Guardian, but he surfaced in good fashion. Costner carries the movie swimmingly with the best of Costner 's ability. [...] But some of these characters were used too much. [...] Just do n't expect a smooth ride. 7 of 10 but almost a 6.

(b) `EXP1:` Adjective - POS feature extraction

How many movies are there that you can think of when you see a movie like this? I can't count them but it sure seemed like the movie makers were trying to give me a hint. I was reminded so often of other movies, it became a big distraction. One of [...]

(c) `EXP2:` Sentence feature extraction

How many movies are there that you can think of when you see a movie like this? [...] See the movie despite its downfalls there is enough to make it interesting and maybe make it appear clever. [...]

(d) `EXP3:` Multi-layer word embedding feature extraction

Fig. 7: Examples of *textual explanation* report for the input in Figure 7a, wrongly labeled by BERT as *Negative* with a probability of 0.99. The most relevant features are highlighted in cyan.

Explanation	**Feature f**	$\mathbf{L_o}$	$\mathbf{L_f}$	$\mathbf{nPIR_f(N)}$
EXP1	POS-Adjective	N	P	**0.884**
EXP2	Sentence	N	P	**0.663**
EXP3	MLWE	N	P	**0.651**

Table 8: Quantitative explanation for the example in Figure 7. P is the positive label, N is the negative label. Positively highly influential features ($nPIR \geq 0.5$) for the L_o class are highlighted in green in the $nPIR_f(N)$ column.

Negative class with $nPIR$ values higher than 0.5, whose perturbation would cause a change in the predicted label from *Negative* to *Positive*). The top relevant features were extracted exploiting Adjectives-POS (Figure 7b), Sentence (Figure 7c), and MLWE (Figure 7d). Regarding the Adjectives-POS feature extraction, Figure 7b shows that general words like {`many, other, big, ..., smooth`} have an $nPIR$ value for the class *Negative* close to 0.88 (Table 8-EXP1). General words with a very strong impact on the final prediction for this specific input text are not a trustful indicator: their absence might lead to entirely different outcomes.

Regarding the sentence-based feature extraction, the negative prediction is triggered by only one specific phrase (Figure 7c), whose absence leads to a *Positive* prediction with a $nPIR$ value of 0.66 (Table 8-EXP2).

Finally, the MLWE feature extraction strategy identifies a cluster composed of only two instances of a very general single word {`there`} as the *most informative feature* (Figure 7d). By removing the two occurrences of the word {`there`}, the prediction changes from *Negative* to *Positive* with an $nPIR$ value of 0.651 (Table 8-EXP3).

Since the output of the prediction model can be drifted (from *Negative* to *Positive*) by simply removing occurrences of general words such as {`there, many, other, big, smooth, ...`} from the input text (actually removing only {`there`} is enough!), doubts on the predicted class reliability are reasonable. More details related to the global behavior and the robustness of the model are addressed in Section 6.4.

6.3.3 Local Explanation: Example 3

The example is reported in Figure 8, where the BERT model correctly classifies the input text in Figure 8a as *Negative*. The *textual explanations* produced by T-EBAnO exploiting different feature extraction strategies are reported as follows: adjective-POS in Figure 8b, verb-POS in Figure 8c, adjective-verb-POS in Figure 8d, sentence in Figure 8e, and multi-layer word embedding in Figure 8f. Their $nPIR$ values are reported in the *quantitative explanations* of Table 9. We note that only the adjective-verb-POS, sentence, and MLWE techniques provide informative explanations, whereas the adjective-POS and verb-POS yield uninformative explanations, yet we include them in the example for discussion.

The POS feature analysis (Figures 8b, 8c) shows that the different parts-of-speech, taken separately one at a time, are not influential for the prediction of the class *Negative*. From the *quantitative explanation* of EXP1 and EXP2 in Table 9 indeed it can be observed that they achieve an *nPIR* close to 0.003 and 0.137 respectively. A similar result was obtained for all the other POS features considered individually. Consequently, T-EBAnO explores the *pairwise combinations* (as explained in Section 4.1) of the parts-of-speech to create more sophisticated features and to analyze more complex semantic concepts. In this case, the feature composed of *Adjectives* and *Verbs* (Figure 8d) is reported to be impacting for the predicted class label reaching a *nPIR* value close to 0.915 (EXP3 in Table 9).
The sentence feature extraction strategy, instead, identifies the feature composed of the phrase in Figure 8e as positively influential for the predicted class with an *nPIR* score of about 0.638 (EXP4 in Table 9).

There were so many classic movies that were made where the leading people were out-and- out liars and yet they are made to look good. I never bought into that stuff. The "screwball comedies" were full of that stuff and so were a lot of the Fred Astaire films. Here, Barbara Stanwyck plays a famous "country" magazine writer who has been lying to the public for years, and feels she has to keep lying to keep her persona (and her job). She even lies to a guy about getting married, another topic that was always trivialized in classic films. She's a New York City woman who pretends she's a great cook and someone who knows how to handle babies, etc. Obviously she knows nothing and the lies pile up so fast you lose track. I guess all of that is supposed to be funny because lessons are learned in the end and true love prevails, etc. etc. Please pass the barf bag. Most of this film is NOT funny. Stanwyck was far better in the film noir genre. As for Dennis Morgan, well, pass the bag again.

(a) Original text

There were so many classic movies that were made where the leading people were out-and- out liars and yet they are made to look good. I never bought into that stuff. The ‘‘screwball comedies’’ were full of [...] plays a famous ‘‘country’’ [...] getting married, another topic that was always trivialized in classic films. [...] she's a great cook and someone [...] supposed to be funny because lessons are learned in the end and true love [...] bag. Most of this film is NOT funny. Stanwyck was far better in the film noir genre. [...]

(b) `EXP1:` Adjective - POS feature extraction

There were so [...] that were made where the leading people were out-and- out liars and yet they are made to look good. I never bought into that stuff. The ‘‘screwball comedies’’ were full of that stuff and so were a lot [...] Barbara Stanwyck plays a famous ‘‘country’’ magazine writer who has been lying to [...] she has to keep lying to keep her persona (and her job). She even lies to a guy about getting married, another topic that was always trivialized in classic films. She 's a New York City woman who pretends she's a great cook and someone who knows how to han dle babies, etc. Obviously she knows nothing and the lies pile up so fast you lose track. I guess all of that is supposed to be funny because lessons are learned in the [...] Please pass the barf bag. Most of this film is NOT funny. Stanwyck was far [...] well, pass the bag again.

(c) `EXP2:` Verb - POS feature extraction

There were so many classic movies that were made where the leading people were out-and- out liars and yet they are made to look good. I never bought into that stuff. The ‘‘screwball comedies’’ were full of that stuff and so were a lot of the Fred Astaire films. Here, Barbara Stanwyck plays a famous ‘‘country’’ magazine writer who has been lying to the public for years, and feels she has to keep lying to keep her persona (and her job). She even lies to a guy about getting married, another topic that was always trivialized in classic films. She 's a New York City woman who pretends she 's a great cook and someone who knows how to handle babies, etc. Obviously she knows nothing and the lies pile up so fast you lose track. I guess all of that is supposed to be funny because lessons are learned in the end and true love prevails, etc. etc. Please pass the barf bag. Most of this film is NOT funny. Stanwyck was far better in the film noir genre. As for Dennis Morgan, well, pass the bag again.

(d) `EXP3:` Adjective & Verb - POS feature extraction

Fig. 8: Examples of *textual explanation* report for the input in Figure 8a originally labeled by BERT as *Negative* with a probability of 0.99. Features found are highlighted in cyan.(Continue)

Finally, the MLWE feature extraction identifies $K = 15$ as the best K partitioning of words, and the *most informative feature* (i.e., cluster of words that maximize nPIR and minimize tokens ratio) with a significant impact on the output prediction is that showed in Figure 8f, reaching an *nPIR* of 0.899 (EXP5 in

[...] She even lies to a guy about getting married, another topic that was always trivialized in classic films. [...]

(e) EXP4: Sentence feature extraction

[...] I never bought into that stuff. The "screwball comedies" were full of that stuff and so were a lot of the Fred Astaire films. Here, Barbara Stanwyck plays a famous "country" magazine writer who has been lying to the public for years, and feels she has to keep lying to keep her persona (and her job). she even lies to a guy about getting married, another topic that was always trivialized in classic films. she's a new york city woman who pretends she's a great cook and someone who knows how to handle babies, etc. Obviously she knows nothing and the lies pile up so fast you lose track. I guess all of that is supposed to be funny because lessons are learned in the end and true love prevails, etc. [...] Most of this film is not funny. Stanwyck was far better in the film noir genre. as for Dennis Morgan, well, pass the bag again

(f) EXP5: Multi-layer word embedding feature extraction

Fig. 8: (Continued) Examples of *textual explanations* for the input in Figure 8a, originally labeled by BERT as *Negative* with a probability of 0.99. Features extracted by T-EBAnO are highlighted in cyan.

Explanation	Feature f	$\mathbf{L_o}$	$\mathbf{L_f}$	$\mathbf{nPIR_f(N)}$
EXP1	POS-Adjective	N	N	0.003
EXP2	POS-Verb	N	N	0.137
EXP3	POS-Adj&Verb	N	P	**0.915**
EXP4	Sentence	N	P	**0.638**
EXP5	MLWE	N	P	**0.899**

Table 9: Quantitative explanations for the example reported in Figure 8. P is the positive label, N is the negative label. Positively highly influential features ($nPIR \geq 0.5$) for the L_o class are highlighted in green in the $nPIR_f(N)$ column.

Table 9).

Analyzing the content of the most influential textual explanations (EXP3, EXP4 and EXP5), it can be observed that, interestingly, all the local explanations with high values of $nPIR$ contain the word {trivialized}. It might seem that a single word can be the only one responsible for the original prediction. However, also the explanation EXP2 contains the same word but is not influential for the class label. Therefore, it emerges that the output predictions are not influenced by single words, but is the combination of different words that allows creating more complex concepts which determine the predicted class label. Moreover, it is possible to say that, in this specific prediction, the model is not sensible to the perturbation of adjectives (EXP1 in figure 8b) or verbs (EXP2 in Figure 8c) separately, highlighting that the proposed prediction has been produced taking into account the whole context of the input text. Only in EXP3 (Figure 8d) is it possible to notice that, when adjectives and verbs are perturbed together, changing the meaning of the input text radically, the predicted class changes. The joint perturbation can be considered a good measure of robustness for the prediction performed by the fine-tuned BERT model under analysis.
However, as for the previous example, it is shown in EXP4 (Figure 8e) that exist a

singular phrase more relevant than the others in the decision-making process. The perturbation of the sentence in EXP4 will bring the model to change the prediction from class *Negative* to *Positive*. Furthermore, EXP5 (Figure 8f), obtained through the MLWE feature extraction technique, shows an apparently random pool of words very relevant in the prediction process. The MLWE feature extraction is able to find the influential feature with higher precision concerning EXP3 (obtained by the combination of all verbs and adjectives), with a very small penalty on the nPIR score. Indeed, the MLWE strategy is able to find a small number of words belonging to different part-of-speeches and different sentences that are affecting the model's output. So, also the resulting explanations are more understandable and meaningful for the end-user.

As in the previous example, this last experiment shows that the predictive model is particularly sensitive to a few specific variations of, apparently not correlated, input words.

From these examples, it emerges that the different feature extraction strategies should be used in a complementary manner, as they look at different aspects of the input text and provide different kinds of explanations. Furthermore, the proposed examples showed that:

- T-EBAnO can be successfully applied to different deep learning models;
- the proposed prediction explanation process can be applied with success to different use cases and NLP tasks;
- T-EBAnO can extract meaningful explanations from both long and short text documents without limiting their interpretability;
- the end-user is provided with informative details to analyze critically and judge the quality of the model outcomes, being supported in deciding whether its decision-making process is trustful.

6.4 Model-global explanations

Exploiting the prediction-local explanations computed by T-EBAnO for all the input documents, *model-global* insights can be provided.

Use case 1. For the toxic comment classification, Figure 9 shows the *GAI* and *GRI* scores for each influential word under the form of word clouds for the classes *Toxic* (Figure 9a and 9c) and *Clean* (Figure 9b and 9d), respectively. They are generated by analyzing all the local explanations produced over the 2250 texts of use case 1 (as discussed in Section 5.2). The font size of words is proportional to the *GAI* or *GRI* scores obtained for each class separately. The proportion of the font size is relative only to the single word cloud (i.e., two words with the same size in different word clouds do not necessarily have the same score, while two words with the same size in the same word cloud have almost the same score).

Firstly, as discussed in Section 5.2, T-EBAnO analyzes the most influential local explanations produced and computes the GAI score for each lemma and the labels *Toxic* and *Clean*. Then, it generates the word clouds (Figures 9a and 9b) to provide a visual impact of the most important lemmas for each class. The *GAI* word clouds (Figures 9a and 9b) show that the two classes are influenced by a non-overlapping set of words. Indeed, the most important lemmas for the *Toxic* class

(a) GAI *Toxic* class. (b) GAI *Clean* class.

(c) GRI *Toxic* class. (d) GRI *Clean* class.

Fig. 9: Global explanation of toxic comment classification with LSTM.

(i.e., with higher *GAI* for the *Toxic* class) are `stupid` (0.31), `Politician1` (0.28), `people` (0.26), `idiot` (0.17), and `white` (0.15). Instead, the lemmas with higher GAI for the *Clean* class are `the` (0.04), `people` (0.02), `and` (0.01), `if` (0.01), and `like` (0.01). This confirms that the model learned that if a word is attributable to toxic language in some context, it is unlikely to be associated with clean language in others. Toxic comments are identified by terms that are strongly related to toxic language, discrimination, or racism. Instead, there is no specific pattern of words that identifies clean comments. Just a few concepts like `people` have an inter-class influence.

Then, T-EBAnO computes the *GRI* score for each lemma and the *Toxic* and *Clean* classes and generates the corresponding word clouds (Figures 9c and 9d) to determine which are the more differentiating concepts between the two classes, among those selected by the model. The *GRI* word cloud highlights, even more, the impact of words like `stupid`, `idiot`, and `ignorant` which obtained a GRI score for the *Toxic* class of 0.31, 0.17, and 0.12, respectively. But also terms related to minorities and genders like `woman`, `black`, `white`, `gay`, (which obtained a GRI for the *Toxic* class of 0.10, 0.10, 0.15, and 0.06 respectively) meaning that the model has learned to recognize racists or sexists comments when these terms are present. Also, the presence of specific politician family names, anonymized as `Politician1`, `Politician2`, etc., highlights that those people's names are related to toxic comments. In particular, `Politician1` achieved the second higher GRI score for the *Toxic* class with 0.28. These results demonstrate that a deep learning model, if not carefully trained, can learn from sensible content, including prejudices and various forms of bias that should be avoided in critical contexts. Finally, associating a specific person's family name to a class also raises ethical issues.

Use case 2. Analyzing the prediction-local explanations produced for the 400 input texts in the *sentiment analysis* use case is possible to extract global insights

(a) GAI *Positive* class. (b) GAI *Negative* class.

(c) GRI *Positive* class. (d) GRI *Negative* class.

Fig. 10: Global explanation of sentiment analysis with BERT.

regarding the fine-tuned BERT model. Figure 10 shows the *GAI* and *GRI* word clouds for the *Positive* (Figure 10a and 10c) and *Negative* (Figure 10b and 10d) class labels.

Again, T-EBANO firstly produces the GAI score for each lemma for the *Positive* and *Negative* classes analyzing all the *most influential* local explanations (as discussed in Section 5.2). The most important lemmas for the *Positive* class (i.e., with higher *GAI* for the *Positive* class) are `film` (0.60), `movie` (0.48), `one` (0.34), `like` (0.22), `story` (0.21), `good` (0.21), `great` (0.20), and `love` (0.19). Instead, the lemmas with higher GAI for the *Negative* class are `movie` (0.37), `film` (0.25), `like` (0.17), `one` (0.16), `even` (0.14), and `story` (0.12). From these values, T-EBANO generates the GAI word clouds for the classes *Positive* (Figure 10a) and *Negative* (Figure 10b).

Differently from the previous example, the *GAI* word clouds for the *Positive* and the *Negative* class labels show that several words like `story`, `movie`, `film`, `like` are impacting on both classes. This means that the model exploits overlapping concepts that do not directly express a sentiment but that, if considered together in their context, can be associated with words that express the mood of the writer (e.g., `This film is not as good as expected`). Thus, to understand which are the lemmas that mostly impact one class with respect to the other, it computes the GRI score for each lemma for the two classes and generates the word clouds.

The *GRI* word cloud for the *Positive* class (Figure 10c) shows that words like `movie` and `film` are still very relevant for it, while they do not appear anymore for the *Negative* class (Figure 10d) that is now highly characterized by the concept of `book`. Indeed, `movie` and `film` obtain a GRI for the *Positive* class of 0.35 and 0.11 respectively (while for the *Negative* class is 0). Instead, `book` achieved a GRI score for the *Negative* class of 0.07 (while for the *Positive* is 0). Exploring the dataset, we noticed that movies inspired by books are used to be associated with negative comments, as typically, the original book is more detailed or slightly different.

Therefore, this can be considered a form of bias that the model has learned, in the sense that a movie evaluation might not be based on its comparison with a book. However, the *GRI* shows also that most of the influential words for positive input texts are concepts strictly related to positive sentiments like `good`, `great`, `best`, `love` achieving a GRI score for the *Positive* class of 0.12, 0.17, 0.12, and 0.17, respectively. Similarly, the negative sentiment is associated with words like `worst`, `bad`, `awful` achieving a GRI score for the *Negative* class of 0.07, 0.06, and 0.05, respectively. For these concepts, the model behaves as expected.

Thanks to the model-global explanation process, the user can better understand how the predictive model is taking its decisions, identifying the presence of prejudice and/or bias, and allowing to decide if and which corrective actions have to be taken to make the decision-making process more reliable.

6.5 Framework Extendibility

In this section, we evaluate the ability of T-EBAnO to adapt to different architectures and different tasks (use cases 3-8). For this purpose, we defined the following additional tasks.

- *Ag News*: a multi-class news topic classification task consisting of four classes: *World*, *Sport*, *Business* and *Science/Technology* [51].
- *Cola*: Corpus of Linguistic Acceptability, a binary classification task that consists of predicting the grammatical *acceptability* or *unacceptability* of the sentence [50].

Both tasks differ from the previous ones (sentiment analysis and toxic comment) because they do not strictly depend on a specific part of the speech. Furthermore, *Ag News* is a multi-class classification problem.

For each task, we trained 3 different models:

- *BERT*: Bidirectional Encoder Representations from Transformers
- *ALBERT*: A Lite BERT [25]
- *ULMfit*: Universal Language Model Fine-tuning [20]

For each model and task, corresponding to the use cases 3-8 of Table 4, we produced with T-EBAnO the *local explanations* of 512 input texts exploiting the *removal* perturbation. Then, for each local explanation, we selected the most influential feature (i.e., with the highest nPIR) and the least influential feature (i.e., with the lowest nPIR).

Figure 11 shows the nPIR distribution of the most influential features (*Max nPIR*) and the least influential features (*Min nPIR*) for all input texts, separately by each model-task. The nPIR values of the least influential features are close to zero for all models, whereas the most influential features have nPIR values close to 1 for all models and generally higher than 0.5. BERT performs better on these tasks and, consequently, T-EBAnO is able to find features having extreme nPIR values. A model like ULMfit, instead, is more uncertain in the prediction, and T-EBAnO finds features with variable *Max nPIR* values from 0.5 to 1. Such results show that T-EBAnO is able to extract different features from the input texts, both highly influential and neutral ones, for the prediction of the class label. Moreover,

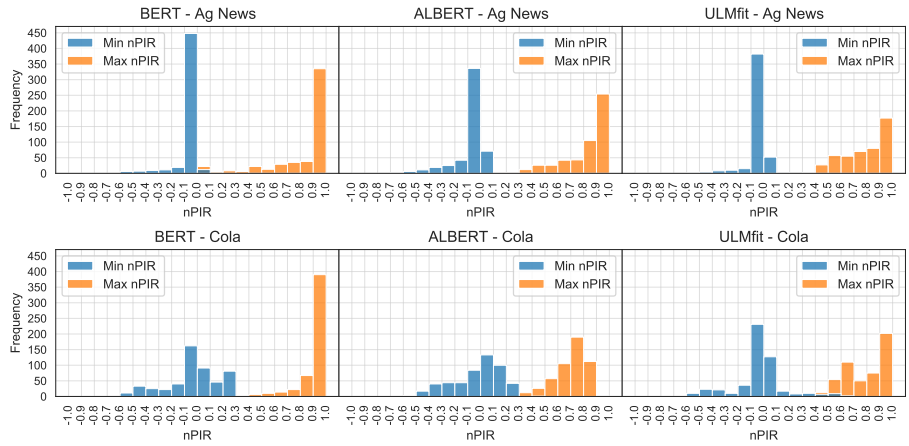


Fig. 11: nPIR distribution of the most influential features (*Max nPIR*) and the least influential features (*Min nPIR*) over 512 input texts for each model and task.

T-EBAnO is able to find explanations for models with different architectures and different classification tasks.

Table 10 shows examples of local explanations for the different models and tasks. For each input text i, one highly influential feature ($i.a$) and one neutral feature (or less influential) ($i.b$) are reported. The original predicted label L_o, the label predicted after the perturbation L_o, and the relative nPIR score obtained by the feature (with respect to the original predicted label L_o) applying the removal perturbation are also reported.

For *Ag News*, inputs from 1 to 6 show that all models correctly learned the concepts of *World*, *Business*, *Sport*, and *Science/Technology*. All the influential features *1-6.a* contain concepts related to the predicted class L_0, whereas less influential features *1-6.b* contain neutral concepts or tokens. The only exception is the *6.a* example, which shows that the ULMFit model overfits some tokens, as the specific name of the London's agency {`Reuters`} has been learned as important for the class label *Business*. The behavior of the explanations is also coherent with the performance in terms of the accuracy of the models. Analyzing a wider set of explanations, also *BERT* and *ALBERT* models overfit some tokens, such as HTML strings of web pages, that are often related to misclassified inputs in *Science/Technology*. Regarding *Cola*, the explanations from *id* 7 to 13 show that the models generally learned to classify grammatically correct sentences. The explanations of the *Acceptable* class label usually contain most of the input text, while the explanations of the *Unacceptable* class labels tend to highlight small portions of the input text containing errors. This behavior is reasonable because a sentence is correct if all its tokens are correct, while it is incorrect if it contains some wrong tokens.

id	MLWE feature	L_o	L_f	nPIR
	BERT-Ag News			
1.a	uk gives blessing to open source . with most organizations that planned to move already moved to microsoft server 2003 , os migration has dropped to the bottom ranks after making its	S/T	W	**0.983**
1.b	uk gives blessing to open source . with most organizations that planned to move already moved to microsoft server 2003 , os migration has dropped to the bottom ranks after making its	S/T	S/T	0.000
2.a	radcliffe to run in new york marathon . london (reuters) - world marathon record holder paula radcliffe believes she has put her failure at the athens olympics behind her after announcing on tuesday that she will run in the new york marathon on november 7 .	S	W	**0.893**
2.b	radcliffe to run in new york marathon . london (reuters) - world marathon record holder paula radcliffe believes she has put her failure at the athens olympics behind her after announcing on tuesday that she will run in the new york marathon on november 7 .	S	S	0.006
	ALBERT-Ag News			
3.a	eu seeks joint asylum policy. eu ministers meeting in luxembourg plan moves to integrate their asylum and immigration procedures.	W	S/T	**0.709**
3.b	eu seeks joint asylum policy. eu ministers meeting in luxembourg plan moves to integrate their asylum and immigration procedures.	W	W	-0.023
4.a	job numbers give candidates room to debate. washington - employers stepped up hiring in august, expanding payrolls by 144,000 and lowering the unemployment rate to 5.4 percent.	B	W	**0.912**
4.b	job numbers give candidates room to debate. washington - employers stepped up hiring in august, expanding payrolls by 144,000 and lowering the unemployment rate to 5.4 percent.	B	B	0.008
	ULMfit-Ag News			
5.a	nato to send staff to iraq . nato will send military trainers to iraq before the end of the year in response to appeals by iraqi leaders for speedy action , us ambassador to nato nicholas burns said today .	W	S/T	**0.706**
5.b	nato to send staff to iraq . nato will send military trainers to iraq before the end of the year in response to appeals by iraqi leaders for speedy action , us ambassador to nato nicholas burns said today .	W	W	0.001
6.a	court seen lifting yukos block – lawyers . london (reuters) - a u.s . bankruptcy court is likely to revoke its temporary ban on the sale of russian oil group yukos 's main production unit, lawyers said on friday	B	W	**0.993**
6.b	court seen lifting yukos block – lawyers . london (reuters) - a u.s . bankruptcy court is likely to revoke its temporary ban on the sale of russian oil group yukos 's main production unit, lawyers said on friday	B	B	0.043
	BERT-Cola			
7.a	many people said they were sick who weren' t .	U	A	**0.985**
7.b	many people said they were sick who weren' t .	U	U	0.200
8.a	charlie will leave town if his mother - in - law doesn' t .	A	U	**0.995**
8.b	charlie will leave town if his mother - in - law doesn' t .	A	A	0.452
9.a	snow white poisoned .	U	A	**0.754**
9.b	snow white poisoned .	U	U	0.014
	ALBERT-Cola			
10.a	mary runs not the marathon.	U	A	**0.819**
10.b	mary runs not the marathon.	U	U	0.267
11.a	both workers will wear carnations.	A	U	**0.744**
11.b	both workers will wear carnations.	A	A	0.033
	ULMfit-Cola			
12.a	you could give a headache to a tylenol .	U	A	**0.930**
12.b	you could give a headache to a tylenol .	U	U	0.119
13.a	paul breathed on mary .	U	A	**0.999**
13.b	paul breathed on mary .	U	U	0.320

Table 10: Features extracted by MLWE on different models and different tasks (highlighted in cyan). For *Ag News*, the labels are *Sport* (S), *World* (W), *Business* (B), *Science/Technology* (S/T). For *Cola*, the labels are *Unacceptable* (U) and *Acceptable* (A). L_o is the original predicted label, L_f is the label predicted after the perturbation on the feature, nPIR is the score obtained by the feature with respect to the original predicted label (nPIR(L_o)). For each input i, there are two features, one highly influential ($i.a$) and one neutral or less influential ($i.b$).

Feature ID	nPIR	Tokens Ratio	FIS
Feature 1	0.50	$10/50 = 0.20$	0.620
Feature 2	0.99	$15/50 = 0.30$	0.874
Feature 3	1.00	$25/50 = 0.50$	0.800

Table 11: Most informative local explanation example.

6.6 MLWE Effectiveness

For a given input text, while the number of tokens of each feature extracted by the part-of-speech (PoS) and sentence-based (SEN) approaches is fixed, the MLWE feature extraction figures out by itself the right number and which tokens to assign to each feature. For an *effective explanation*, we want that the most influential feature extracted by the MLWE maximizes the nPIR while minimizing the number of tokens (i.e., it selects only the tokens contributing to a high nPIR).

For instance, Table 11 reports a sample (partial) result where the MLWE is applied to an input text with 50 total tokens: 3 possible clustering results are discussed (note that the discussion is limited to 3 for simplicity, but K_{max} should be used for full results, as described in Section 4.4).

- The *most influential feature* in the first clustering is *Feature 1* with $nPIR = 0.50$ and it consists of 10 tokens.
- The *most influential feature* in the second clustering is *Feature 2* with $nPIR = 0.99$ and it consists of 15 tokens.
- The *most influential feature* in the third clustering is *Feature 3* with $nPIR = 1.00$ and it consists of 25 tokens.

The *feature informative score FIS*, as explained in Section 4.4, is computed accordingly to the following formula:

$$\begin{aligned} FIS(\kappa) &= \max\Big(\big(\alpha(nPIR_\kappa) + \beta(1 - \kappa_{tk}/n_{tk})\big), 0\Big) \\ &= \max\Big(\big(0.60(nPIR_\kappa) + 0.40(1 - \kappa_{tk}/n_{tk})\big), 0\Big) \end{aligned} \tag{9}$$

Then, the final *most informative local-explanation* selected by T-EBAnO is *Feature 2* because it provides a high nPIR with a limited number of tokens. *Feature 1* has a smaller number of tokens, but its lower nPIR leads to a lower FIS. *Feature 3*, on the contrary, has a higher nPIR but includes much more tokens, hence having a lower FIS too.

Thus, MLWE can be viewed as a heuristic that, exploiting the inner information of the model, figures out exactly the group of tokens that influenced mostly the original input prediction in a reasonable amount of time. Indeed, in theory, the best possible solution (i.e., the smallest amount of tokens that mostly influenced the prediction) could be found by exploring all the *n-combinations* of tokens for each n in the range $[2, n_{tk}]$ (where n_{tk} is the number of tokens in the input text) and taking the one that maximizes a performance metric such as the FIS score. For instance, if an input text contains 100 tokens, it would be necessary to explore and evaluate all the 2-combinations, 3-combinations up to 100-combinations of 100 input tokens, making the problem unfeasible, especially for long texts.

Therefore, to evaluate the effectiveness of the MLWE, we compare its performance with a *Random* feature extraction method. The *Random* feature extraction creates several features, each composed of a group of n_r random tokens, with different sizes (i.e., number of tokens) in the range $n_r \in [1, n_r\_max]$ where n_r_max is set to 80% of the total tokens of the input text n_{tk}. Specifically, for each n_r value, it creates 5 random features, each composed of a different group of n_r random tokens selected from the input text. For instance, if an input text has 100 tokens, the *Random* feature extraction creates 5 features composed of 1 random token, 5 features each composed of a group of 2 random tokens, up until 5 features each composed of a group of 80 random tokens. We chose to create 5 features for each random feature size n_r value in the specified range because, with these settings, the *Random* feature extraction creates at least 5 times more features than *MLWE*. Consequently, it has a clear advantage in the comparison at the cost of more computational power. Thus, we want to see if, selecting a random subset of all the possible solutions (i.e., the random features of different sizes are a subset of all the possible combinations of tokens), where the cost of extracting and evaluating the influence of these random features is much higher with respect to the MLWE (i.e., higher computation time), the most influential features founded by the MLWE are more effective in terms of influence and compactness (i.e., nPIR and tokens ratio).

We experimented on *BERT-IMDB* and *BERT-Ag News* since BERT-Cola contains very short sentences, which was not meaningful for our goals. We produced the local explanations with both the MLWE and the *Random* feature extraction from 512 input documents for each task. For *BERT-IMDB*, about 230 thousand features have been produced with $nPIR\_mean = 0.1$ (about 460 for each input). Instead, for *BERT-Ag News*, about 91 thousand features have been produced with $nPIR\_mean = 0.07$ (about 185 for each input). This shows that simply removing some random groups of tokens to obtain a high nPIR value would be insufficient, hence the need to carefully and smartly select the tokens. However, as expected, due to the large number of features extracted from each input text, some *Random* features obtain a high nPIR score by chance. For each input text, we selected the *most informative local-explanation* extracted with the *Random* feature extraction method exploiting the same formula used by the *MLWE* (equation 9).

To understand the effectiveness of the MLWE, we compared the percentage of selected tokens ratio (i.e., the number of tokens of the feature with respect to the total number of tokens in the input) belonging to the very high influential features ($nPIR \geq 0.9$) extracted by *MLWE* and *Random* on the two tasks.

Figure 12 shows the CDF (Cumulative Distribution Function) of the very high influential features with respect to the percentage of tokens. The chart shows that T-EBAnO with *MLWE* finds very high influential features selecting fewer tokens with respect to the *Random* feature extraction method. Indeed, looking at the CDF, the 75% of very high influential features (i.e., $nPIR \geq 0.90$) found by *MLWE* (blu lines) on *IMDB* and *Ag News* contains, respectively, less than 35% and 50% of tokens. On the other hand, the 75% of very high influential features found by *Random* (orange lines) on *IMDB* and *Ag News* contains less than or equal to 55% and 70% of tokens. MLWE is then more effective in selecting a lower number of more influential tokens.

We also compared the execution time of the *MLWE* and the *Random* feature extraction. For the *IMDB* task, the *MLWE* feature extraction method is about 6

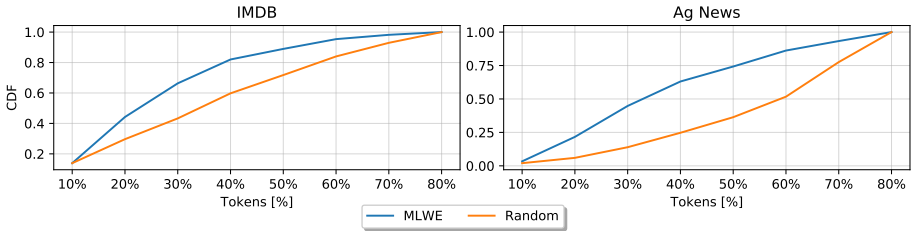


Fig. 12: CDF of tokens percentage ratio (i.e., percentage of feature tokens over total input tokens) for very high influential features (i.e., features with $nPIR \geq 0.9$) extracted with T-EBAnO-*MLWE* and *Random*, separately for *BERT-IMDB* and *BERT-Ag News*.

times faster than the *Random* approach, with 35 seconds per input versus 215 seconds per input, on average. For the *Ag News* task, the *MLWE* feature extraction method is about 4.5 times faster than the *Random* approach, with 10 seconds per input versus 46 seconds per input, on average. By exploiting the inner knowledge learned by the model, the *MLWE* feature extraction method provides higher *effectiveness* and *efficiency* with respect to searching random features. The *MLWE* approach finds high influential features containing a lower percentage of tokens and, at the same time, reduces the execution time.

6.7 Hyperparameters evaluation

We evaluated the impact on the most informative local explanation produced by the *Multi-layer Word Embedding* feature extraction (MLWE) by changing the hyperparameters α and β of the *Feature Informative Score* (FIS) computation. We recall that α and β sum up to one and weights the influence (nPIR) and the compactness (1 - tokens ratio), respectively, in the *Feature Importance Score* (FIS) computation (equation 9). The *tokens ratio* is computed as the number of tokens in the feature over the total number of tokens. The objective of equation 9 in the unsupervised clustering analysis is to maximize the influence (nPIR) and minimize the number of tokens inside the most influential cluster of each k division. Thus, the most influential clusters founded will change based on these hyperparameters.

We used the BERT model fine-tuned for topic classification on *Ag News* dataset (Use cases 3 in Table 4) for this purpose. We randomly selected 512 input texts from the dataset, and we produced the local explanations with T-EBAnO. Table 12 shows the mean *nPIR* and *tokens ratio* for different α and β values for the *most informative* local explanations extracted by T-EBAnO (i.e., with max FIS score). Specifically, for each local explanation of each input text, we selected the most informative explanation, and we averaged the tokens ratio and the influence (nPIR) of the most influential features over the entire dataset.

On the one hand, with smaller α values (0.2 and 0.3), and respectively high β values (0.8 and 0.7), the most informative features founded by the MLWE have a low mean influence (mean nPIR 0.31 and 0.45 respectively). But the most informative clusters are very compact, being composed of only 11% and 15%. However, even if tiny clusters increase the comprehensibility of the explanation, they lack

	Hyperparameter Values						
	$\alpha = 0.2$ $\beta = 0.8$	$\alpha = 0.3$ $\beta = 0.7$	$\alpha = 0.4$ $\beta = 0.6$	$\alpha = 0.5$ $\beta = 0.5$	$\alpha = \mathbf{0.6}$ $\beta = \mathbf{0.4}$	$\alpha = 0.7$ $\beta = 0.3$	$\alpha = 0.8$ $\beta = 0.2$
$\overline{\textbf{nPIR}}$	0.31	0.45	0.54	0.59	0.61	0.64	0.64
$\overline{\textbf{TokensRatio}}$	11%	15%	21%	24%	25%	29%	31%

Table 12: Mean influence ($\overline{nPIR}$) and mean selected tokens ratio ($\overline{TokensRatio}$) in the most influential explanations with different α and β values.

completeness because they select only a partial set of the relevant tokens mostly used by the model for the original prediction.

On the other hand, greater α values (0.7 and 0.8), and respectively high β values (0.3 and 0.2), obtain larger nPIR mean values (close to 0.64) with the pain of larger clusters found. However, the increase in the nPIR mean is too small compared with the cost of the increasing size with respect to values of $\alpha \in [0.4, 0.5, 0.6]$ and relative $\beta \in [0.6, 0.5, 0.4]$ values. Indeed, they achieve a mean nPIR of 0.54, 0.59, and 0.61, with a mean tokens ratio of 21%, 24%, and 25%, respectively. Finally, the couple $\alpha = 0.3$ and $\beta = 0.7$ values obtained an already good mean nPIR value of 0.45 with a very small percentage of tokens highlighted, equal to 15%.

In conclusion, in this paper, we used $\alpha = 0.6$ and $\beta = 0.4$ as default values because they allow us to reach a good trade-off between the number of highlighted tokens and their influence. Indeed, even if a clear best value does not emerge from this experiment, setting $\alpha = 0.4$ and $\beta = 0.6$ seems good to obtain small clusters with a strong influence. However, other possible values could be useful in different scenarios. Thus, the final user can change this parameter accordingly to specific needs.

6.8 nPIR correlation with human judgement

To assess the quality of the explanations, which are selected by T-EBAnO based on their nPIR value, we evaluate the correlation between the nPIR value and human judgment. Te human validation is performed by interviewing both expert and non-expert users with a survey[2]. The survey contains local explanations extracted by T-EBAnO, and their nPIR value is compared with the relevance assigned by the users. More precisely, we selected 12 input texts from *Ag News* with BERT and 8 input texts from the *Toxic Comment* use case with the LSTM model. In such use cases, input texts are shorter and then more suitable for a survey. For the purpose of this survey, we selected only correctly classified examples. For each input text, we randomly picked one highly influential feature and one neutral feature extracted by T-EBAnO. Those features are then presented to the user, who is requested to select one option among *"Very Relevant"*, *"Relevant"*, and *"Not Relevant"* for each feature. The main scope of the survey is to measure and evaluate the correlation between the influence index (nPIR) and human judgment. However, we also indirectly validate the quality and readability of the explanations

[2] The link to the online survey is available in the T-EBAnO GitHub repository.

produced by T-EBAnO: for correctly classified examples, if the proposed explanations are effective and human-readable, then the user should understand which features are important (*"Relevant"* or *"Very Relevant"*) and which are neutral (*"Not Relevant"*).

Figure 13 shows the introductory example of the survey. In the first box (input text), the user can read the original input text, the predicted label, and the probabilities of such prediction computed by the NLP model. Then, two explanations are presented for each input text, with the feature words highlighted in light blue. In total, at the time of writing, we collected 4320 user evaluations from 108 participants (each evaluating 2 explanations from 20 input texts), with 76% being expert machine learning users, and 18% being also expert users of Natural Language Processing with deep learning (as anonymously self-declared by themselves in the survey). Participants have been invited among researchers and students of PhD and Master courses in Computer Science.

To evaluate the correlation between nPIR and the human judgment, we assigned to each question (that corresponds to an explanation/feature extracted by T-EBAnO) a *manual score* of 0 if the user selected *"Not Relevant"*, 0.5 for *"Relevant"*, and 1 for *"Very Relevant"*.

Figure 13 shows, for each of the 40 explanations (2 for each of the 20 input texts), the nPIR assigned by T-EBAnO (the blue bars), and the mean relevance assigned by the 108 users (the red bars), according to the *manual scores* (the data are presented in descending order of *Human Score*). The chart shows an explicit correlation between the nPIR assigned by T-EBAnO for both influential and neutral features, for both tasks, topic detection and toxic comment classification. This also implies that T-EBAnO produces effective and human-readable explanations for the final users.

We also measured the inter-annotator agreement between the 108 annotators (survey participants) by using each explanation as input (for a total of 40 annotations). Then, we obtained only two possible labels by aggregating the *"Relevant"* and the *"Very Relevant"* into the same label. We exploited the Krippendorff's alpha coefficient[3] [24] to measure the inter-annotator reliability agreement. We obtained a Krippendorff's alpha coefficient of 0.65, denoting a good agreement between the 108 participants and the 40 explanations.

Additionally, we asked the participants (i) if the task of the survey was clear, as a self-evaluation check: 44% answered 5 (max value), and 41% chose 4 out of 5; (ii) if the explanations proposed by T-EBAnO were easy to understand, the answers from top (5) to bottom (1) were distributed as follows: 34%, 45%, 18% 3%, and 0%.

6.9 Effectiveness evaluation with respect to model-agnostic techniques

In this section, we compare the effectiveness of the T-EBAnO-MLWE explanations with two model-agnostic explainability techniques. Comparing explainability methodologies is still an open issue in the research community, as a definitive definition of *good explanation* is missing. However, explanations should have im-

[3] It has been exploited the implementation in: https://github.com/LightTag/simpledorff

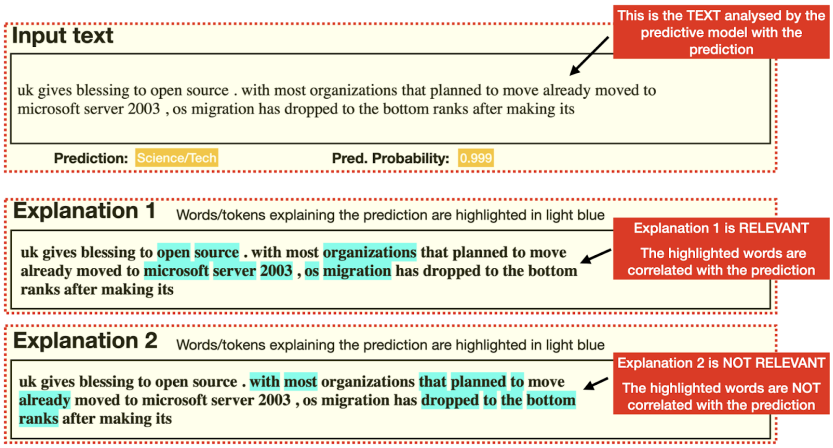


Fig. 13: Survey's introduction example.

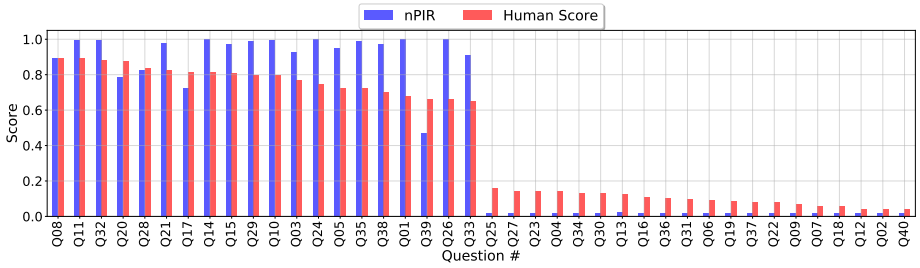


Fig. 14: Comparison between nPIR assigned by T-EBAnO (blue bars) and mean human scores (red bars). Questions are ordered by descending mean human score.

portant properties such as: *Fidelity*, *Comprehensibility*, *Complexity*, *Effectiveness*, *Trustworthiness*, etc.

We performed an experiment to evaluate the *Fidelity* and the *Effectiveness* of the explanations proposed by T-EBAnO with respect to two state-of-the-art model-agnostic techniques, *LIME*[4] [40] and *SHAP*[5] [32].

To measure the *Fidelity* and *Effectiveness* of the proposed explanations, we removed the words/tokens highlighted as important by the different methodologies, and we measured the change in probability caused by this deletion. Basically, we are asking the following questions:

1. The important words/tokens highlighted by the explainability techniques are effectively the ones used by the model to perform the original prediction?
2. How the model prediction changes if the highlighted words/tokens are not present in the original text?

[4] The LIME parameter number of permutations has been set to 5.000.
[5] The Partition Explainer version of SHAP has been used.

If removing the words/tokens highlighted by the explanation does not correlate with a reduction in the probability of the original class label, then the selected words are not among the important features used by the model to produce the original label. On the contrary, the larger the probability changes, the more the model relied on those words/tokens to predict the original label.

To make a fair comparison, we created features composed of the same percentage of the most important tokens identified by the different methodologies. LIME assigns an importance score to each token. However, it requires defining the percentage of the most important tokens for the importance score. Therefore, we set this parameter so that the number of the most important tokens for the class of interest is almost equal to the mean number of tokens highlighted by the T-EBAnO explanations. Instead, SHAP assigns an importance score to each token of the input text (*Shapley Values* [44]). Thus, we selected the most important ones with the same percentage of T-EBAnO. In this way, we selected subsets with similar cardinality and importance.

We chose as experimental use cases (i) a BERT model fine-tuned for sentiment analysis with IMDB and (ii) a BERT model fine-tuned for topic classification on Ag News Subset. We did not use the same models trained in Table 4 due to compatibility issues. We trained two new models exploiting the *HuggingFace*[6] library. The fine-tuned models reached 93% and 95% accuracy on the validation set for IMDB and Ag News Subset, respectively. The experiments were performed on a single node of the SmartData BigData cluster at Polito[7]. The node contains two Intel Xeon Gold 6140 CPUs with 2.30 GHz frequency and 384 GB of RAM. However, for the experiment, we limited the process to using only one CPU with a maximum of 120 GB of RAM (without exploiting GPUs).

For the IMDB case, T-EBAnO-MLWE highlights on average about 20% of tokens, so we also removed the top-20% of tokens selected by LIME and SHAP. We evaluated the probability difference before and after removing the highlighted tokens for each methodology. Removing the most influential tokens highlighted by T-EBAnO causes a mean decrease of probability around 71%, The same removal for LIME causes a 48% probability drop on average, and for SHAP the mean probability decrease is 59%. We also compared the mean execution time to produce an explanation. The IMDB dataset contains relatively long texts and, on average, T-EBAnO took 38 seconds, while LIME 304 and SHAP 484 seconds.

For the second use case, on the Ag News dataset, T-EBAnO-MLWE highlights, on average, about 30% of tokens. We removed the top-30% of tokens selected by LIME and SHAP. This time, removing the most important tokens yields a mean decrease of probability around 75% for T-EBAnO, 60% for LIME, and 61% for SHAP. The mean execution time to produce each explanation is lower because this dataset contains shorter sentences. Specifically, T-EBAnO takes on average 4 seconds, LIME 239 seconds, and SHAP 16 seconds.

We notice that not only T-EBAnO is much faster than the other two methodologies (approximately from 1 to 2 orders of magnitude), but also the explanations provided are more *faithful* and *effective*, by highlighting as important tokens the ones that were the most impacting for the prediction of the model under analysis.

[6] https://huggingface.co/

[7] https://smartdata.polito.it/computing-facilities/

7 Conclusion and future research directions

This paper proposed T-EBAnO, a new engine able to provide both prediction-local and model-global interpretable explanations in the context of NLP analytics tasks that exploit deep learning models. T-EBAnO's experimental assessment includes different NLP classification tasks, i.e., sentiment analysis task, comment toxicity, topic classification, and sentence acceptability, performed through state-of-the-art techniques: fine-tuned models like BERT, ALBERT, and ULMFit and a custom LSTM model.

Results showed that T-EBAnO can (i) identify specific features of the textual input data that are predominantly influencing the model's predictions, (ii) highlight such features to the end-user, and (iii) quantify their impact through effective indexes. The proposed explanations enable end-users to decide whether a specific local prediction made by a deep learning model is reliable and to evaluate the general behavior of the global model across predictions. Besides being useful to general-purpose end users, explanations provided by T-EBAnO are especially useful for data scientists, artificial intelligence and machine learning experts in need of understanding the behavior of their models since the extracted features, both textual and numeric, are an efficient way to harness the complex knowledge learned by the models themselves.

Future research directions include: (a) investigating new strategies for the perturbation of the input features, such as new kinds of substitution perturbations, exploiting task-specific or expert-driven directives; (b) integrating T-EBAnO in a real-life setting to measure the effectiveness of the proposed textual explanations by real-world human evaluation; (c) extending T-EBAnO to address new data analytics activities, such as guiding data scientists in applying fine-tuned deep-learning models, explaining concept drifts, and providing insights on Adversarial Attack countermeasures; (d) extending the proposed methodology and influence index (i.e., nPIR) to new NLP tasks such as Question Answering and Named Entity Recognition. (e) designing an XAI comparison methodology tailored to the NLP domain containing both objective and subjective comparison criteria and applying it to compare T-EBAnO with several XAI methodologies.

References

1. Explaining classifier decisions linguistically for stimulating and improving operators labeling behavior. Information Sciences **420**, 16 – 36 (2017)
2. Adadi, A., Berrada, M.: Peeking inside the black-box: A survey on explainable artificial intelligence (xai). IEEE Access **6**, 52138–52160 (2018). DOI 10.1109/ACCESS.2018.2870052
3. Alvarez-Melis, D., Jaakkola, T.S.: A causal framework for explaining the predictions of black-box sequence-to-sequence models. arXiv preprint arXiv:1707.01943 (2017)
4. Banzhaf, J.: Weighted voting doesn't work: A mathematical analysis. Rutgers Law Review **19**(2), 317–343 (1965)
5. Basiri, M.E., Nemati, S., Abdar, M., Cambria, E., Acharya, U.R.: Abcdm: An attention-based bidirectional cnn-rnn deep model for sentiment analysis. Future Generation Computer Systems **115**, 279 – 294 (2021). DOI 10.1016/j.future.2020.08.005
6. Bolukbasi, T., Chang, K.W., Zou, J., Saligrama, V., Kalai, A.: Man is to computer programmer as woman is to homemaker? debiasing word embeddings (2016)
7. Borkan, D., Dixon, L., Sorensen, J., Thain, N., Vasserman, L.: Nuanced metrics for measuring unintended bias with real data for text classification. CoRR **abs/1903.04561** (2019)

8. Chakraborty, M., Biswas, S.K., Purkayastha, B.: Rule extraction from neural network trained using deep belief network and back propagation. Knowledge and Information Systems **62**(9), 3753–3781 (2020). DOI 10.1007/s10115-020-01473-0. URL https://doi.org/10.1007/s10115-020-01473-0
9. Chen, J., Jordan, M.: Ls-tree: Model interpretation when the data are linguistic. Proceedings of the AAAI Conference on Artificial Intelligence **34**(04), 3454–3461 (2020). DOI 10.1609/aaai.v34i04.5749. URL https://ojs.aaai.org/index.php/AAAI/article/view/5749
10. Datta, A., Sen, S., Zick, Y.: Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In: 2016 IEEE Symposium on Security and Privacy (SP), pp. 598–617 (2016). DOI 10.1109/SP.2016.42
11. Deeks, A.: The judicial demand for explainable artificial intelligence. Columbia Law Review **119**(7), 1829–1850 (2019)
12. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. CoRR **abs/1810.04805** (2018)
13. Du, M., Liu, N., Yang, F., Hu, X.: Learning credible dnns via incorporating prior knowledge and model local explanation. Knowledge and Information Systems (2020). DOI 10.1007/s10115-020-01517-5. URL https://doi.org/10.1007/s10115-020-01517-5
14. Ethayarajh, K.: How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings. ArXiv **abs/1909.00512** (2019)
15. Ethayarajh, K.: How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings (2019)
16. Fong, R.C., Vedaldi, A.: Interpretable explanations of black boxes by meaningful perturbation. 2017 IEEE International Conference on Computer Vision (ICCV) (2017). DOI 10.1109/iccv.2017.371
17. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Y.W. Teh, M. Titterington (eds.) Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, *Proceedings of Machine Learning Research*, vol. 9, pp. 249–256. JMLR Workshop and Conference Proceedings, Chia Laguna Resort, Sardinia, Italy (2010)
18. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM Comput. Surv. **51**(5), 93:1–93:42 (2018). DOI 10.1145/3236009
19. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**, 1735–80 (1997). DOI 10.1162/neco.1997.9.8.1735
20. Howard, J., Ruder, S.: Universal language model fine-tuning for text classification (2018)
21. Jia, Y., Bailey, J., Ramamohanarao, K., Leckie, C., Ma, X.: Exploiting patterns to explain individual predictions. Knowledge and Information Systems **62**(3), 927–950 (2020). DOI 10.1007/s10115-019-01368-9. URL https://doi.org/10.1007/s10115-019-01368-9
22. Karlsson, I., Rebane, J., Papapetrou, P., Gionis, A.: Locally and globally explainable time series tweaking. Knowledge and Information Systems **62**(5), 1671–1700 (2020). DOI 10.1007/s10115-019-01389-4. URL https://doi.org/10.1007/s10115-019-01389-4
23. Khodabandehloo, E., Riboni, D., Alimohammadi, A.: Healthxai: Collaborative and explainable ai for supporting early diagnosis of cognitive decline. Future Generation Computer Systems (2020). DOI 10.1016/j.future.2020.10.030
24. krippendorff, k.: Computing krippendorff's alpha-reliability (2011)
25. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: A lite bert for self-supervised learning of language representations (2020)
26. Lei, T., Barzilay, R., Jaakkola, T.: Rationalizing neural predictions (2016)
27. Lepri, B., Staiano, J., Sangokoya, D., Letouzé, E., Oliver, N.: The Tyranny of Data? The Bright and Dark Sides of Data-Driven Decision-Making for Social Good, pp. 3–24. Springer International Publishing, Cham (2017)
28. Lertvittayakumjorn, P., Toni, F.: Human-grounded evaluations of explanation methods for text classification. ArXiv **abs/1908.11355** (2019)
29. Li, J., Monroe, W., Jurafsky, D.: Understanding neural networks through representation erasure (2016)
30. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized BERT pretraining approach. CoRR **abs/1907.11692** (2019). URL http://arxiv.org/abs/1907.11692
31. Lloyd, S.: Least squares quantization in pcm. IEEE Transactions on Information Theory **28**(2), 129–137 (1982). DOI 10.1109/TIT.1982.1056489

32. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (eds.) Advances in Neural Information Processing Systems 30, pp. 4765–4774. Curran Associates, Inc. (2017)
33. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pp. 142–150. Association for Computational Linguistics, Portland, Oregon, USA (2011)
34. Mathews, S.M.: Explainable artificial intelligence applications in nlp, biomedical, and malware classification: A literature review. In: K. Arai, R. Bhatia, S. Kapoor (eds.) Intelligent Computing, pp. 1269–1292. Springer International Publishing, Cham (2019)
35. Murdoch, W.J., Szlam, A.: Automatic rule extraction from long short term memory networks (2017)
36. Naseem, U., Razzak, I., Musial, K., Imran, M.: Transformer based deep intelligent contextual embedding for twitter sentiment analysis. Future Generation Computer Systems **113**, 58 – 69 (2020). DOI 10.1016/j.future.2020.06.050
37. Pastor, E., Baralis, E.: Explaining black box models by means of local rules. In: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC '19, pp. 510–517. ACM, New York, NY, USA (2019). DOI 10.1145/3297280.3297328
38. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543 (2014)
39. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100,000+ questions for machine comprehension of text (2016)
40. Ribeiro, M.T., Singh, S., Guestrin, C.: Why should i trust you? explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1135–1144 (2016)
41. Samek, W., Montavon, G., Vedaldi, A., Hansen, L., Muller, K.R.: Explainable AI: Interpreting, Explaining and Visualizing Deep Learning (2019). DOI 10.1007/978-3-030-28954-6
42. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. CoRR **abs/1910.01108** (2019). URL http://arxiv.org/abs/1910.01108
43. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. International Journal of Computer Vision (2019). DOI 10.1007/s11263-019-01228-7
44. Shapley, L.S.: A value for n-person games. Contributions to the Theory of Games **2**(28), 307–317 (1953)
45. Shrikumar, A., Greenside, P., Kundaje, A.: Learning important features through propagating activation differences. In: D. Precup, Y.W. Teh (eds.) Proceedings of the 34th International Conference on Machine Learning, *Proceedings of Machine Learning Research*, vol. 70, pp. 3145–3153. PMLR (2017). URL https://proceedings.mlr.press/v70/shrikumar17a.html
46. Trifonov, V., Ganea, O.E., Potapenko, A., Hofmann, T.: Learning and evaluating sparse interpretable sentence embeddings. In: Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pp. 200–210. Association for Computational Linguistics, Brussels, Belgium (2018). DOI 10.18653/v1/W18-5422
47. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. CoRR **abs/1706.03762** (2017)
48. Ventura, F., Cerquitelli, T., Giacalone, F.: Black-box model explained through an assessment of its interpretable features. In: New Trends in Databases and Information Systems - ADBIS 2018 Short Papers and Workshops, AI*QA, BIGPMED, CSACDB, M2U, BigDataMAPS, ISTREND, DC, Budapest, Hungary, September, 2-5, 2018, Proceedings, pp. 138–149 (2018). DOI 10.1007/978-3-030-00063-9_15
49. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: Glue: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461 (2018)
50. Warstadt, A., Singh, A., Bowman, S.R.: Neural network acceptability judgments. arXiv preprint arXiv:1805.12471 (2018)
51. Zhang, X., Zhao, J., LeCun, Y.: Character-level convolutional networks for text classification (2015)
52. Zheng, X., Wang, M., Chen, C., Wang, Y., Cheng, Z.: Explore: Explainable item-tag corecommendation. Information Sciences **474**, 170 – 186 (2019)

53. Zhou, Q., Liu, X., Wang, Q.: Interpretable duplicate question detection models based on attention mechanism. Information Sciences (2020)