RESEARCH

Open Access

Xai-driven knowledge distillation of large language models for efficient deployment on low-resource devices

Riccardo Cantini[1*], Alessio Orsino[1] and Domenico Talia[1]

*Correspondence:
rcantini@dimes.unical.it

[1] University of Calabria, Rende, Italy

Abstract

Large Language Models (LLMs) are characterized by their inherent memory inefficiency and compute-intensive nature, making them impractical to run on low-resource devices and hindering their applicability in edge AI contexts. To address this issue, Knowledge Distillation approaches have been adopted to transfer knowledge from a complex model, referred to as the teacher, to a more compact, computationally efficient one, known as the student. The aim is to retain the performance of the original model while substantially reducing computational requirements. However, traditional knowledge distillation methods may struggle to effectively transfer crucial explainable knowledge from an LLM teacher to the student, potentially leading to explanation inconsistencies and decreased performance. This paper presents *DiXtill*, a method based on a novel approach to distilling knowledge from LLMs into lightweight neural architectures. The main idea is to leverage local explanations provided by an eXplainable Artificial Intelligence (XAI) method to guide the cross-architecture distillation of a teacher LLM into a self-explainable student, specifically a bi-directional LSTM network.Experimental results show that our XAI-driven distillation method allows the teacher explanations to be effectively transferred to the student, resulting in better agreement compared to classical distillation methods,thus enhancing the student interpretability. Furthermore, it enables the student to achieve comparable performance to the teacher LLM while also delivering a significantly higher compression ratio and speedup compared to other techniques such as post-training quantization and pruning, which paves the way for more efficient and sustainable edge AI applications

Keywords: Knowledge distillation, eXplainable Artificial Intelligence, Low-resource devices, Edge AI, Large language models, Sustainable AI

Introduction

In recent years, Large Language Models (LLMs) have gained significant traction for their remarkable natural language understanding and generation capabilities [1–3]. However, LLMs are often characterized by their inherent memory-inefficient and compute-intensive nature. For instance, BERT (Bidirectional Encoder Representations from Transformers) encompasses 110 million parameters in its base version, while the GPT-3 contains

175 billion parameters, requiring at least 320 gigabytes of storage in half-precision (i.e., 16-bit floating point) [4].

The diffusion of low-resource devices, such as mobile and Internet-of-Things (IoT) ones, driven by novel edge AI paradigms, led to the increasing need for finding efficient ways to deploy and run complex deep learning models on such small and compact devices [5, 6]. In this context, different techniques for model compression such as pruning and quantization have emerged, as well as some approaches relying on Knowledge Distillation (KD) [7, 8]. In particular, knowledge distillation has been leveraged to transfer knowledge from a complex model, referred to as the teacher, to a more compact, computationally efficient one, known as the student. Despite the ability of such approaches to achieve comparable performance to the original model while significantly reducing computational demands, they may struggle to transfer the rationale behind the teacher's decision process to the student. Indeed, as the field of artificial intelligence continues to progress, there has been a growing emphasis on transparency and interpretability in deep learning systems. The demand for eXplainable Artificial Intelligence (XAI) has arisen as a response to the need for comprehensibility in the decision-making processes of complex predictive models.

Only a few efforts have been made in the literature to leverage recent advancements in XAI to drive the learning process of deep learning models [9, 10]. In this paper, we propose *DiXtill*, a novel approach to distilling knowledge from transformer-based LLMs into lightweight neural architectures, specifically a bi-directional Long Short-Term Memory network. The main idea behind *DiXtill* is to leverage local explanations to guide the distillation process of a teacher model into a self-explainable student. This XAI-driven distillation process aims at transferring the knowledge and the explanations from the teacher to the student model, thus enhancing both the performance and interpretability of the distilled student model.

Our method allows the student to achieve comparable performance to the teacher LLM, while also delivering significantly higher compression ratio and speedup compared to post-training quantization and pruning, thus easing the deployment and inference on low-resource devices. In addition, experimental results show how the teacher's explanations can be effectively transferred to the student model during the distillation process. In particular, we measured a strong agreement between the teacher's word attributions, computed by the Integrated Gradients method, and those of the self-explainable student learned during the distillation process.

The contribution of our work can be summarized as follows:

- We identify a limitation of traditional knowledge distillation methods in conveying explainable knowledge from a teacher LLM to the student.
- We propose to integrate local explanations within the distillation process of a teacher LLM into a self-explainable lightweight student, which enhances both the accuracy and interpretability of the distilled model.
- By employing a cross-architecture knowledge distillation process, a significantly smaller and more computationally efficient model can be attained compared to other compression techniques such as quantization and pruning, which instead retain the same complex architecture as the teacher model.

The remainder of the paper is organized as follows. "Related work" section discusses state of the art about knowledge distillation, deployment of LLM on resource-constrained devices, and XAI techniques. "Proposed method" section describes the proposed *DiX-till* method. "Experimental evaluation" section presents the experimental evaluation and the obtained results. Finally, "Conclusion" section concludes the paper.

Related work

Knowledge distillation

The literature on knowledge distillation encompasses a wide range of techniques, which can be broadly categorized into three classes [8, 11, 12]: (*i*) *offline*, where a pre-trained teacher model guides the student model training; (*ii*) *online*, where both models are updated simultaneously in a single end-to-end training process; and (*iii*) *self*-distillation, where either the same model serves as both the teacher and student [13], transferring knowledge from deeper to shallower layers, or the student is a separate model sharing the same architecture as the teacher.

The knowledge is transferred from the teacher model to the student by minimizing a loss function that encourages the student network to mimic the teacher network's label predictions. The commonly used loss function for knowledge distillation is a linear combination of the cross-entropy and the Kullback–Leibler (KL) divergence loss between the softened probability distributions of the teacher and student models. Specifically, a temperature scaling factor τ is generally used to smooth the probability distributions and reveal inter-class relationships learned by the teacher [8]. Hence, the loss function is mathematically expressed as follows:

$$\mathcal{L} = -(1-\alpha)\left[\sum_j \log p_j^S(1)\right] + \alpha\left[\sum_i p_i^T(\tau)\log\frac{p_i^T(\tau)}{p_i^S(\tau)}\cdot\tau^2\right] \tag{1}$$

where S is the student, T is the teacher, α is a hyperparameter for the convex combination, and $p^{\mathscr{F}}(\tau)$ is the probability distribution scaled with temperature τ, obtained as $p^{\mathscr{F}}(\tau) = \mathit{softmax}(z/\tau)$, where $z = \mathscr{F}(x)$ are the output logits computed by a given model $\mathscr{F}$. Recent research has also explored the impact of the KL loss function on logit matching and of such softening on generalization. Notably, as an alternative to the KL divergence, the Mean Squared Error (MSE) has been used in the literature [14, 15] to promote the matching between teacher and student logits z, formally: $\mathcal{L}_{MSE}(z_T, z_S) = ||z_T - z_S||_2^2$.

Deployment of LLMs on low-resource devices

To enable the deployment of LLMs on devices with limited computational resources, several techniques such as distillation, quantization, and pruning have been leveraged in the literature [16]. *Knowledge distillation* has been explored to reduce the size of LLMs and allow them to operate under constrained computational scenarios. One notable example is DistilBERT [17], which is a lightweight Transformer model obtained by distilling BERT base with 40% fewer parameters than the original BERT. Other attempts have been made to distill knowledge from large models to lightweight neural architectures, using a cross-architecture knowledge distillation process [18]. For example, Tang et al. [15] proposed a method to distill knowledge from BERT into a single-layer bi-directional LSTM, demonstrating comparable performance to the original network without external training data or additional input features. *Quantization* involves reducing the precision of model parameters (e.g., weights and activations from 32-bit floating-point values to 8-bit integers) to achieve smaller memory footprints and faster inference times. Generally, a deep neural model can be quantized using two main approaches: *Post-Training Quantization (PTQ)* and *Quantization-Aware Training (QAT)* [19]. The former quantizes the model parameters after training without modifying its underlying architecture, while the latter integrates quantization into the model's training process, allowing it to adapt to low-precision representations and ensuring higher accuracy compared to PTQ. However, since QAT methods cannot easily scale to large models like LLMs [20], common PTQ-based algorithms are generally used in the case of LLMs, including Activation-aware Weight Quantization (AWQ) [20], GPTQ [21], and dynamic quantization, which computes the range for each activation based on the data range observed at runtime. Finally, another common method for the compression of large models is *pruning*, which aims to reduce the size of the model by removing unnecessary network elements. This technique can be applied in an *unstructured* manner [4], by removing individual weights to build an irregular sparse structure, or in a *structured* one [22], by removing high-granularity components of the network, such as neurons, channels, or layers. While unstructured pruning significantly reduces model size, conventional hardware like GPUs struggle to leverage the unstructured sparse patterns to accelerate model inference [23]. Consequently, several structured pruning techniques have gained popularity in the landscape of LLM compression, such as attention head pruning [24], which involves removing individual attention heads without significant impacts on performance and without requiring model retraining.

Explainable artificial intelligence

Deep learning models pose a challenge in offering interpretable explanations for their predictions, hindering their practical utility in crucial domains such as healthcare and legal contexts [25]. To address this issue, eXplainable Artificial Intelligence (XAI) techniques have emerged, which can be distinguished between *local* explainers, which only explain the reasoning behind an individual prediction, and *global* explainers, which instead provide a rationale for the whole dataset [26]. XAI approaches can be further categorized into *post-hoc* and *interpretable-by-design* methods. Post-hoc methods aim at interpreting black-box models after training. Most post-hoc techniques are currently *model-agonistic* since they make any assumption about the structure of the deep learning model to be explained but treat it as a black-box model. In contrast, self-explainable models are inherently designed for interpretability during the prediction phase, providing *ante-hoc* explanations that faithfully represent the model's reasoning. However, these methods are not flexible and may not easily be integrated with other deep learning models [27, 28]. Among post-hoc methods, *LIME (Locally Interpretable Model-Agnostic Explainer)* [29] is a local and model-agnostic explanation technique that extracts feature importance scores by perturbing real samples and observing the corresponding changes in the model's predictions. Then a simple and interpretable model is built that approximates the original model's behavior in the neighborhood of the original samples.

Another common explanation method, grounded in cooperative game theory, is *SHAP (SHapley Additive exPlanation)* [30]. The method computes the contribution of each feature to the predicted outcome using the Shapley value, which is a measure that fairly distributes the credit among a set of players (i.e., the input features) contributing to a certain outcome (i.e., the model prediction). *Integrated Gradients (IG)* [31] is a model-specific explanation method that calculates feature attributions to the prediction by accumulating gradients along a path from a baseline instance to the specific instance of interest. By using integration, IG captures the sensitivity of the model's output to variations in each input feature, revealing influential features for a given prediction.

Besides using XAI methods to provide the user with useful insights into the rationale behind the model decision process, some attempts have been made in the field of *Explanation-Guided Learning* (EGL) to investigate how explanations can be leveraged to improve the learning process of deep models [32]. As an example, Nigisetty et al. [9] proposed xAI-GANs, a new class of Generative Adversarial Networks that incorporate local explanations of the classification made by the discriminator into the gradient descent process to provide richer corrective feedback to the generator. Zeng et al. [33] proposed a method to generate end-to-end attributional explanations for deep networks, leveraging attribution maps from an adversarially trained counterpart model to supervise the learned explanations. Another method was proposed by Alharbi et al. [10], consisting of leveraging convolutional autoencoders to transfer both the knowledge and explanations from a teacher to a student, represented by Convolutional Neural Networks at different scales.

Proposed method

The proposed method, namely *DiXtill*, provides a novel approach to distilling knowledge from LLMs into lightweight neural architectures, thus easing the deployment on resource-constrained devices. As depicted in Fig. 1, the main idea behind *DiXtill* is to leverage local explanations provided by an *Explainer* as a complement to the usual prediction-based supervision, to guide the distillation process of a *Teacher LLM* into a *Self-explainable student.*

The teacher is an LLM based on the Transformer [34] architecture, such as *BERT (Bidirectional Encoder Representations from Transformers)* [35] and *GPT (Generative Pre-trained Transformer)* [36]. The student model is a bi-directional Long Short-Term Memory (LSTM), which generates both a classification and an explanation via masked attention. In the following sections, the architecture of the self-explainable student network and the distillation process will be delineated in detail, describing how explanations are transferred from the teacher to enhance both the effectiveness and interpretability of the distilled student.

Self-explainable student model architecture

The student model consists of a bi-directional Long Short-Term Memory network enhanced with masked attention. The use of such a mechanism allows for improving classification performance and provides interpretability, by dynamically assigning weights to individual input words in proportion to their significance for the model's classification.

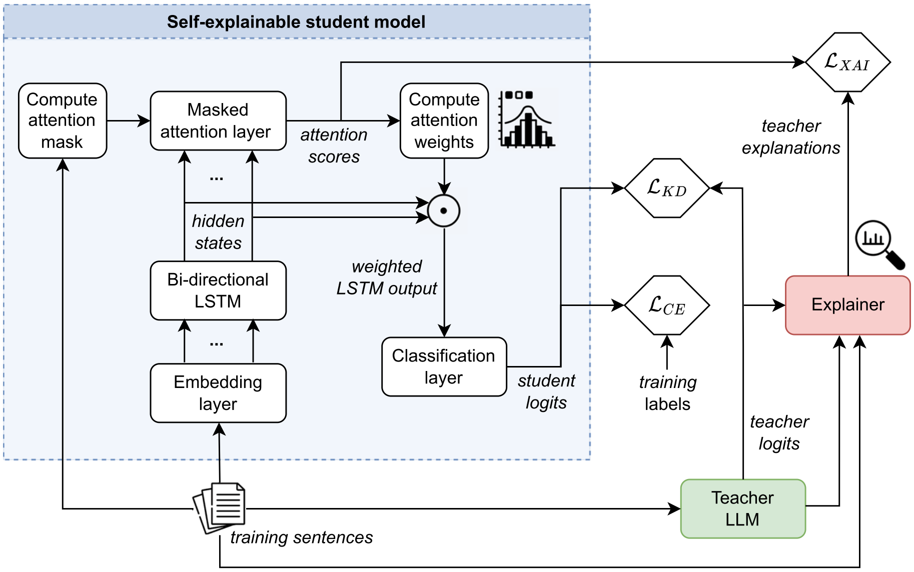


Fig. 1 Distillation process in *DiXtill*. The *Explainer*, the *Teacher LLM*, and the *Student* are indicated in red, green, and blue colors, respectively

The student model is composed of three main trainable modules, i.e., the *Embedding layer*, the *Bi-LSTM network*, and the *Masked attention layer*. Given an input sentence x, composed of a sequence of words $w_1, w_2, \ldots, w_k$, the embedding layer learns a continuous vector representation of x, i.e., a sequence of k d-dimensional vectors $E = e_1, e_2, \ldots, e_k$, where $e_i \in \Re^d$ is the embedding of w_i, and $E \in \Re^{d \times k}$ is the embedding matrix. Subsequently, the matrix is inputted into the bi-directional Long Short-Term Memory (bi-LSTM) layer, which learns a sequence of hidden states denoted as $h_1, h_2, \ldots, h_k$. Such representations are obtained by concatenating the left-to-right (i.e., $\overrightarrow{h_i}$) and right-to-left (i.e., $\overleftarrow{h_i}$) components, which consist of u-dimensional vectors, thus generating a hidden states matrix $H \in \Re^{2u \times k}$. Afterward, a weight is computed for each hidden state in H by the masked attention layer. Specifically, a score vector σ is calculated to determine the unnormalized importance of each of the k elements in the sequence for the model's classification. This vector is obtained using a Bahdanau-like attention mechanism [37], implemented by a parameterized feed-forward neural network, in which a trainable matrix $U \in \Re^{2u \times 2u}$ is employed to perform a linear projection of H, which is subsequently fed into a tanh layer. Following this, a learnable vector $v \in \Re^{2u}$ is utilized to calculate the ultimate vector $\sigma \in \Re^k$, formally: $\sigma = v^T tanh(U \cdot H)$.

An attention mask is incorporated to prevent the model from attending to padding tokens. Specifically, for each sequence, a mask vector $\mu \in \Re^k$ is computed such that $\mu_i = 0$ if w_i corresponds to the *PAD* token, $\mu_i = 1$ otherwise. This vector is then used to mask the attention scores σ by computing the Hadamard product $\sigma = \sigma \odot \mu$. Then, the μ vector is adjusted to transform values corresponding to a mask of 0 into highly negative numbers, whereas values associated with a mask of 1 remain unaltered. This ensures that when computing the α weights through the softmax distribution, elements masked with 0 receive exponentially small values, preventing the model from attending to those masked elements. Overall, the masking process is performed as follows: $\sigma_i = \sigma_i \cdot \mu_i \quad \longrightarrow \quad \mu_i = (\mu_i - 1) \cdot 10^4 \quad \longrightarrow \quad \tilde{\sigma_i} = \sigma_i + \mu_i$.

At this point, the attention weights $\alpha \in \Re^k$ are computed by applying a softmax function to the unnormalized adjusted scores $\tilde{\sigma}$, converting them into a distribution. Finally, a weighted average of the bi-LSTM hidden states H is determined, resulting in the vector $\hat{h} \in \Re^d$, which is fed to a linear layer to compute the output logits z:

$$z = W_{out} \cdot \hat{h} + b_{out} \ , \quad \text{where: } \ \alpha_i = \frac{e^{\tilde{\sigma_i}}}{\sum_{j=1}^{k} e^{\tilde{\sigma_j}}} \ , \quad \hat{h}_i = \sum_{j=1}^{k} H_{ij} \cdot \alpha_j \tag{2}$$

Here, $W_{out} \in \Re^{m \times d}$ and $b_{out} \in \Re^m$ are trainable weights, where m is the number of classes. The model predicts the class c for the input sequence x as $c = \underset{m}{\operatorname{argmax}}\,(z)$, along with an explanation $\mathcal{E}(x)$. The explanation is built starting from the vector of masked attention scores σ as a set of pairs $(w_i : \sigma_i)$, denoting the influence of each word w_i on the model classification of x into class c.

Incorporating explanations into the distillation process

As mentioned before, the key idea behind the XAI-driven approach introduced by *DiXtill* involves utilizing local explanations to guide the distillation of a teacher LLM into a compact self-explainable bi-LSTM enhanced with masked attention, to improve both the effectiveness and interoperability of the distilled student.

In *DiXtill*, a post-hoc explanation technique is leveraged to compute an explanation $\mathcal{E}^T(x)$ of the teacher predictions for each training instance x. Such an explanation is required to be a list of $(w_i : \sigma_i^T)$, in which σ_i^T are the word attributions specifying to what extent each word $w_i \in x$ influences the teacher prediction for each particular instance. Therefore, the proposed method integrates well with the most popular post-hoc XAI approaches in the literature, such as SHAP, LIME, and Integrated Gradients (IG). Specifically, in *DiXtill* we used Integrated Gradients for computing teacher explanations for the training data. We chose IG due to its ease of implementation, theoretical justifications, and computational efficiency when compared to alternative approaches such as LIME or SHAP, which allow it for scaling to large networks such as those of LLMs, and feature spaces. In addition, IG is specifically designed to work well with a variety of deep networks, while methods such as LIME and SHAP may provide explanations that are inconsistent and unstable [38]. The IG method highlights feature importance by computing the gradient of the model output prediction with respect to its input features. Specifically, let $x \in \Re^d$ and $x' \in \Re^d$ be the input instance and a baseline, respectively; in the case of text data, the baseline may be a zero d-dimensional vector. The integrated gradient for an input x and a baseline x' is obtained by cumulating the gradients computed on all points lying on the straight path connecting the input and the baseline. Formally, the IG along the i^{th} dimension for a particular instance x is defined as follows:

$$(x_i - x_i') \times \int_{\alpha=0}^{1} \frac{\partial \mathscr{F}\,(x' + \alpha \times (x - x'))}{\partial x_i} d\alpha \tag{3}$$

Here, $\mathscr{F} : \Re^d \rightarrow [0, 1]$ is a function representing the model to be explained, and $\frac{\partial \mathscr{F}\,(x)}{\partial x_i}$ is the gradient of $\mathscr{F}\,(x)$ along the i^{th} feature.

Once explanations for the teacher are computed, these can be leveraged to guide the distillation process. In particular, *DiXtill* introduces an extension of the classical distillation loss, by adding an XAI-based term, i.e., $\mathcal{L}_{XAI}$, which promotes the alignment between the explanation provided by the teacher and the student for each training instance x by using a cosine distance loss. Let $\mathcal{E}^T(x)$ and $\mathcal{E}^S(x)$ be the explanation of the teacher and the student prediction for an input instance x, respectively. Word attributions can be extracted from the explanations, which are the σ^T vector for the teacher, statically computed by IG, and the σ^S vector for the student, dynamically computed by the student itself, and learned via backpropagation. The $\mathcal{L}_{XAI}$ loss term is defined as follows:

$$\mathcal{L}_{XAI} = \frac{1}{2}\Big(1 - \frac{\sigma^T \cdot \sigma^S}{||\sigma^T||\,||\sigma^S||}\Big) \tag{4}$$

Therefore, the overall loss is defined as: $\mathcal{L} = (1-\alpha)\mathcal{L}_{CE} + \alpha(\mathcal{L}_{KD} + \mathcal{L}_{XAI})$, where:

- $\mathcal{L}_{CE}$ is the standard cross-entropy loss of the student with respect to the real hard training labels y weighted by a factor $(1-\alpha)$.
- $\mathcal{L}_{KD}$ is the distillation loss term, represented by a normalized version of the Kullback–Leibler divergence scaled by a factor α. It is computed between the softmax predictions of the teacher and the student models, softened with temperature τ, and it is defined as: $1 - \exp\big(-KL(p^T(\tau), p^S(\tau)\big)$.
- $\mathcal{L}_{XAI}$ is the aforementioned XAI-based loss term, scaled by a factor α, forcing the student to align the learned explanations (i.e, the unnormalized attention weights σ^S) to those of the teacher computed via the IG method.

Experimental evaluation

In this section, we present the experimental evaluation we carried out to assess the effectiveness of *DiXtill*. The experiments were conducted using FinBERT [39], a BERT model pre-trained on financial communication text. Specifically, the FinBERT model was fine-tuned on the Twitter Financial News dataset, which is an English-language dataset containing an annotated corpus of finance-related tweets for sentiment analysis. The dataset contains 9 938 training instances and 2 486 instances for testing purposes. The model determines the financial sentiment of given tweets, which can be classified as *bearish*, *bullish*, or *neutral*.

As concerns the student model configuration, we used a Glove word embedding layer, which produces 50-dimensional vector representations, and 50 hidden LSTM units with a sequence length equal to 150. In addition, each model was trained for 15 epochs using the SGD optimization algorithm with a momentum of 0.9 and a learning rate of 0.01. Moreover, as suggested by Hinton et al. [8], we used a value of 0.9 for the α hyperparameter, which assigns a considerably higher weight to the distillation loss $\mathcal{L}_{KD}$ and the attributions alignment term $\mathcal{L}_{XAI}$ introduced by *DiXtill*, compared to the standard cross-entropy loss term $\mathcal{L}_{CE}$. Finally, the temperature value τ was set to 5 for all distillation techniques employing temperature scaling.

In the following sections, we demonstrate how our XAI-driven distillation method outperforms state-of-the-art techniques in terms of performance. Then, we discuss how our method compares to other compression techniques, ensuring a better balance between performance and model size. Finally, we evaluate the consistency of the self-computed explanations of *DiXtill* and those related to the other distillation methods, showcasing how the proposed method can ensure a high level of faithfulness and interpretability.

Comparison with knowledge distillation methods

Here we compare the performance of *DiXtill* against different models, including (*i*) the baseline student model trained from scratch without knowledge distillation, using a standard cross-entropy loss; (*ii*) two distilled student models, obtained with KL knowledge distillation [8] and matching logits with MSE [14, 15], respectively; (*iii*) the teacher LLM, used as the upper bound for classification performance.

For each model, we evaluated the accuracy, macro F1 score, Matthews correlation, and multiclass ROC AUC score, using the One-vs-Rest (OvR) approach. Results are reported in Table 1 and Fig. 2.

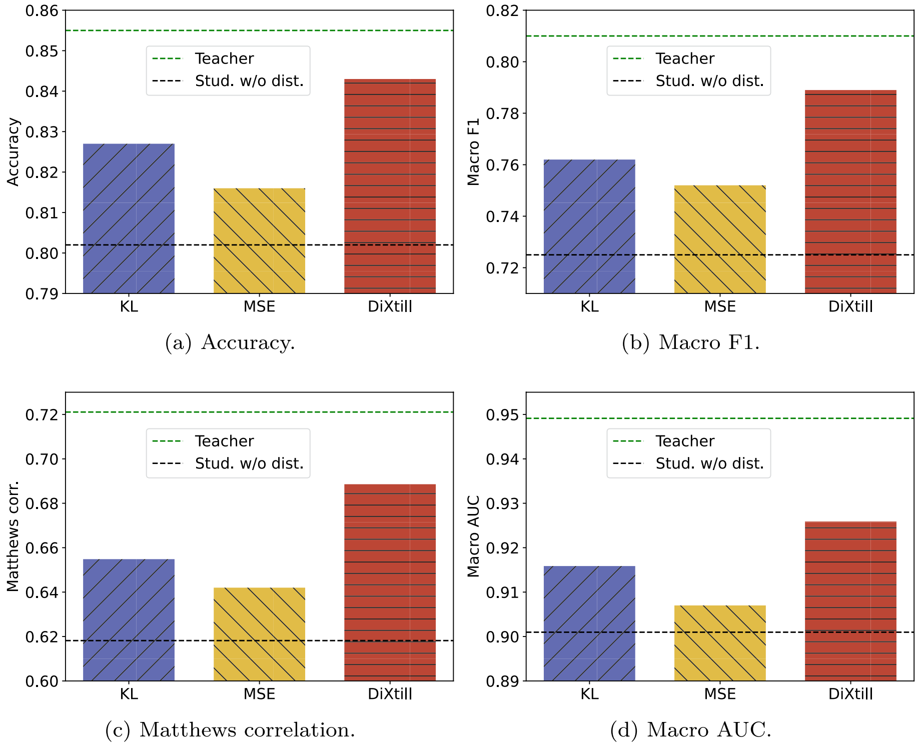
Achieved results show that the teacher (i.e., the reference model) provides an upper bound for performance, reaching an accuracy of 85.5% and a macro F1 score of 81%. When employing traditional knowledge distillation with the KL divergence loss, there is a noticeable decrease in accuracy to 82.7% and in macro F1 score to 76%. Similar performances are achieved by using the MSE-based knowledge distillation, which further reduces the accuracy to 81.6% and macro F1 to 75.2%. As a baseline, training a student model from scratch without any distillation results in the lowest accuracy of 80.2%. These results highlight the challenges of distilling an LLM into a small bi-LSTM, as well as learning from scratch without any teacher guidance. Conversely, the performance achieved by using *DiXtill*, which shows an accuracy of 84.3% and a macro F1 score of 78.9%, indicates that incorporating local explanations during distillation enables competitive performance comparable to those of the teacher, with a remarkable reduction of the number of parameters, decreasing from 0.11 billion to less than a million.

In addition, we investigated the performance of *DiXtill* at different temperature values, ranging from 1 to 5, as reported in Table 2. Specifically, as τ increases, the softmax function generates a softer probability distribution, facilitating the transfer of richer information from the teacher to the student model during the distillation process.

Table 1 Classification performance comparison of different knowledge distillation methods

Method	Accuracy	Macro F1	Matthews Corr.	Macro AUC
Student w/o distillation	0.802	0.725	0.618	0.901
Distillation with KL	0.827	0.762	0.655	0.916
Distillation with MSE	0.816	0.752	0.642	0.907
DiXtill	**0.843**	**0.789**	**0.689**	**0.926**
Teacher	0.855	0.810	0.721	0.949

Results achieved by *DiXtill* are highlighted in bold. In addition, the performance of the baseline student (trained without distillation) and the teacher models are reported



(a) Accuracy.

(b) Macro F1.

(c) Matthews correlation.

(d) Macro AUC.

Fig. 2 Classification performance comparison with other knowledge distillation methods. Dotted lines indicate the upper and lower bounds achieved by the teacher (in green) and the student without distillation (in black), respectively

Table 2 Performance achieved by *DiXtill* at different temperature values

Metric	**Temperature**				
	$\tau = 1$	$\tau = 2$	$\tau = 3$	$\tau = 4$	$\tau = 5$
Accuracy	0.834	0.835	0.838	0.840	**0.843**
Macro F1	0.775	0.778	0.782	0.779	**0.789**
Matthews corr.	0.671	0.678	0.677	0.679	**0.688**
Macro AUC	0.924	0.923	0.925	0.926	**0.926**

Bolded values indicate the best temperature

Comparison with quantization and pruning methods

In this section, we present the results of the comparison between *DiXtill* and two alternative methods for model compression applied to the pre-trained FinBERT model, namely *dynamic post-training quantization* (PTQ), and *attention head pruning* (AHP). Regarding quantization, the weights of the teacher model are quantized using static int8, while activations are dynamically quantized (per batch) to int8. For pruning we followed the methodology outlined in [24], which involves pruning multiple heads from the entire model within each layer. Specifically, we analyzed the model performance by selectively masking each attention head. Heads whose removal resulted in the most significant drop in classification performance were retained, while those whose masking did not affect performance were deemed redundant and pruned accordingly [24]. For the sake of simplicity, we initially discuss the

Table 3 Comparison of compression ratio and speedup between *DiXtill*, PTQ, and AHP models, relative to the teacher model

Method	**Size** **($\mathcal{C}_{\text{ratio}}$)**	**Inference time** **(*Speedup*)**
AHP-6	365 MB (↑ 1.20×)	0.28 s (↑ 2.18×)
PTQ	182.5 MB (↑ 2.40×)	0.40 s (↑ 1.52×)
DiXtill	**3.45 MB** (↑ **127**×)	**0.07 s** (↑ **8.7**×)
Teacher	439 MB	0.61 s

Results achieved by *DiXtill* are highlighted in bold

Table 4 Comparison of classification performance between *DiXtill*, PTQ, and AHP models

Method	**Accuracy** **($\mathcal{P}^{\text{Acc}}_{\text{drop}}$)**	**Macro F1** **($\mathcal{P}^{\text{F1}}_{\text{drop}}$)**	**Matthews corr.** **($\mathcal{P}^{\text{Matt.}}_{\text{drop}}$)**	**Macro AUC** **($\mathcal{P}^{\text{AUC}}_{\text{drop}}$)**
AHP-6	0.832 (↓ $2.6e^{-2}$)	0.743 (↓ $8.2e^{-}2$)	0.652 (↓ $9.5e^{-2}$)	0.938 (↓ $1.2e^{-2}$)
PTQ	0.852 (↓ $3.5e^{-3}$)	0.808 (↓ $2.5e^{-3}$)	0.719 (↓ $2.8e^{-3}$)	0.948 (↓ $1.1e^{-3}$)
DiXtill	0.843 (↓ $1.4e^{-2}$)	0.789 (↓ $2.6e^{-2}$)	0.689 (↓ $4.5e^{-2}$)	0.926 (↓ $2.4e^{-2}$)
Teacher	0.855	0.810	0.721	0.949

performance of a pruned model in which 6 out of 12 attention heads per layer are retained, denoted as AHP-6.

To fairly compare *PTQ* and *AHP* to *DiXtill*, in Table 3 we measure the model size and the compression ratio $\mathcal{C}_{ratio} = \frac{\mathcal{S}_r}{\mathcal{S}_c}$, which is defined as the ratio of the reference model size r to the size of the compressed model c, where a higher compression ratio means that the model is more compact and efficient. In addition, we provide the inference time computed for a batch of 32 test samples and the speedup relative to the teacher model, determined as the ratio between the inference time of the reference model and that of the compressed one.

We compared *DiXtill* to the other compression methods in terms of classification performance, by also computing for each classification metric the performance drop relative to the teacher model. Particularly, we define the performance drop for a given metric m as $\mathcal{P}^m_{drop} = 1 - \frac{\mathcal{P}^m_c}{\mathcal{P}^m_r}$, where $\mathcal{P}^m_c$ is the performance of the compressed model c concerning the metric m and $\mathcal{P}^m_r$ is the performance of the reference model r (e.g., the teacher) for the same metric. Results achieved are reported in Table 4.

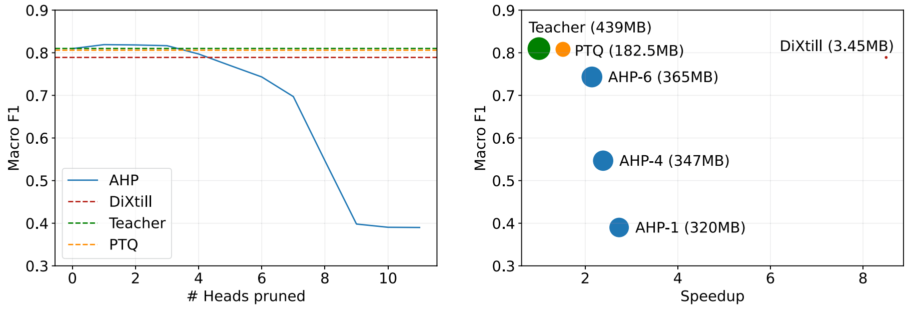
The dynamically int8 quantized model maintains a high accuracy of 85.2%, comparable to that of the original 32-floating point model. This suggests that the quantization process, which reduces the precision of model weights and activations, has a minimal impact on the overall performance. However, int8 quantization only reduces the model size from 439 MB to 182.5 MB, resulting in a compression ratio of 2.40×, while maintaining inference times similar to those of the teacher model, enabling a modest speedup of 1.52×. On the other hand, attention head pruning, while demonstrating a marked reduction in inference time by a factor of 2.18×, falls short in achieving a significant compression ratio compared to the original teacher model. Notably, when 50% of attention heads are pruned, the resultant model size remains prohibitively large for deployment on edge devices, yielding a mere compression ratio of 1.20×. Furthermore, this method exhibits the most substantial decline in overall classification performance compared to the other compression techniques.

In contrast, *DiXtill* allows for distilling knowledge from a large BERT-like model to a very lightweight Bi-LSTM network, with a size of only 3.45 MB, achieving a remarkable compression ratio of 127×. Our method also reduces inference time by an order of magnitude with a speedup of 8.7×, proving effective in alleviating computational burdens during inference compared to the teacher model and other compression methods.

It is worth pointing out that to achieve higher compression ratios for the pruned model, more attention heads can be removed. However, this approach results in a marked degradation of classification performance, as illustrated in Fig. 3a. This figure depicts the evolution of the macro F1 score as the number of pruned heads increases from 0 (i.e., no pruning is performed) to 11 (i.e., only the most important head within each layer is retained from the original model), in comparison to the performance of the teacher and other compressed models. Notably, performance drop becomes unacceptable when pruning more than 6 attention heads, with a significant decrease observed, reaching a macro F1 score of 0.4 when employing only 1 head. In line with the results of [24], in some cases removing an attention head may result in increased performance (e.g., pruning 1, 2, or 3 heads within each layer in our experiments). Furthermore, a cross-domain analysis of classification performance and speedup, shown in Fig. 3b, reveals that *DiXtill* consistently outperforms other methods in terms of reduced model size, classification performance, and speedup.

Evaluating agreement between teacher and student explanations

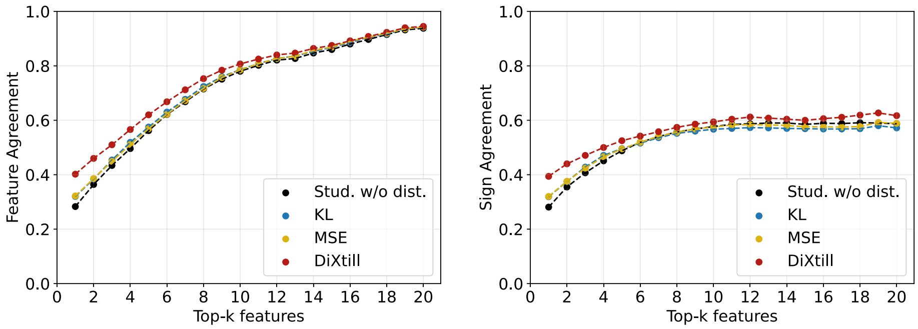
In this section, we evaluate the consistency of the self-computed explanations of *DiXtill* and those related to the other distillation methods outlined in "Comparison with knowledge distillation methods" section, computed through IG. Specifically, given $\mathcal{F}$ as the set of features (i.e., words) of the sample x to be explained, we measured the pairwise agreement between the explanations obtained for the teacher T (i.e., $\mathcal{E}^T(x)$) and the different distilled students S (i.e., $\mathcal{E}^S(x)$). We used the following metrics [40]:



(a) Evolution of macro F1 as the number of pruned heads increases, compared to the performance of the teacher and compressed models obtained using *DiXtill* and PTQ.

(b) Cross-domain comparison (macro F1 vs. speedup) between *DiXtill* and other compression methods. The size of the circles indicates the model size achieved with each method.

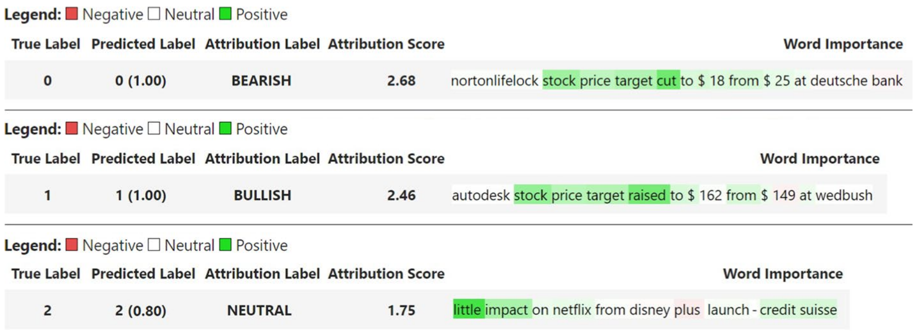
Fig. 3 Comparison of *DiXtill*, PTQ, and AHP models at varying numbers of heads pruned, in terms of macro F1, compression ratio, and speedup



(a) Feature Agreement.

(b) Sign Agreement.

Fig. 4 Evaluation of explanation agreement between the teacher and different student models obtained with *DiXtill* and other distillation methods



Legend: 🟥 Negative □ Neutral 🟩 Positive

True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
0	0 (1.00)	BEARISH	2.68	nortonlifelock stock price target cut to $ 18 from $ 25 at deutsche bank

Legend: 🟥 Negative □ Neutral 🟩 Positive

True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
1	1 (1.00)	BULLISH	2.46	autodesk stock price target raised to $ 162 from $ 149 at wedbush

Legend: 🟥 Negative □ Neutral 🟩 Positive

True Label	Predicted Label	Attribution Label	Attribution Score	Word Importance
2	2 (0.80)	NEUTRAL	1.75	little impact on netflix from disney plus launch - credit suisse

Fig. 5 Example of explanations produced by *DiXtill* for a *bearish*, *bullish*, and *neutral* test instance, plotted with the Captum library [41]

- *Feature agreement*. It is computed as the fraction of common features between the sets of top-k attributions of $\mathcal{E}^T(x)$ and $\mathcal{E}^S(x)$ of a given student model S. Formally: $\frac{1}{k}|\{f \in \mathcal{F} \mid f \in \text{top}_f(\mathcal{E}^T(x), k) \wedge f \in \text{top}_f(\mathcal{E}^S(x), k)\}|$.
- *Sign agreement*. It measures to what extent $\mathcal{E}^T(x)$ and $\mathcal{E}^S(x)$ of a given student model S agree by also considering the feature attribution signs for the top-k features. Therefore, this is a stricter measure, computed as: $\frac{1}{k}|\{f \in \mathcal{F} \mid f \in \text{top}_f(\mathcal{E}^T(x), k) \wedge f \in \text{top}_f(\mathcal{E}^S(x), k) \wedge sgn(\mathcal{E}^T(x), f) = sgn(\mathcal{E}^S(x), f)\}|$.

The results depicted in Fig. 4 demonstrate the superior agreement achieved by *DiXtill* compared to other distillation techniques.

Lastly, for the sake of completeness, some examples of explanations learned by the distilled student are reported in Fig. 5, showing word attributions for the different classes considered.

As can be seen in Fig. 5, the student model effectively captures the patterns linking the provided test instances to the correct class, also providing a high-quality explanation of its predictions. In particular:

- In the first example, the text is correctly classified as *bearish*, with the model focusing on words like *cut*, *stock*, and *price*. Indeed, a bearish sentiment in the financial domain reflects a pessimistic outlook among investors, anticipating a decline in asset prices and an overall market downturn.

- In the second example, the model correctly identifies the provided text as *bullish*, focusing on words like *raised*, *stock*, and *price*. Notably, a bullish sentiment indicates a positive perspective among investors, foreseeing an increase in asset values and overall market expansion.
- The last example is correctly classified as *neutral* by the model, as it does not convey any explicit bearish or bullish tendency. In this case, the model focuses on the words *little*, *impact*, *credit*, and *suisse*, where Credit Suisse is a global financial services company based in Switzerland.

Discussion

Our research showcased the efficacy of the proposed XAI-driven distillation method in transferring explainable knowledge from an LLM to a lightweight self-explainable student network. This allows the student to achieve comparable classification performance to the teacher LLM, outperforming other traditional distillation methods, such as KL distillation and matching logits with MSE. Moreover, by integrating teacher explanations into the distillation process, *DiXtill* allows for achieving a higher level of interpretability and faithfulness of the distilled student model, whose explanations, learned via backpropagation, show a stronger agreement with those of the teacher, compared to traditional distillation techniques. In terms of computational and memory efficiency, employing a cross-architecture knowledge distillation approach enables the use of a substantially smaller and more compact student network, which facilitates deployment and inference on resource-constrained devices. Conversely, common compression techniques, such as post-training quantization and attention head pruning, retain the same complex neural architecture as the teacher model, resulting in lower compression ratios and speedup compared to *DiXtill*. As a consequence, despite ensuring a high level of accuracy, such techniques may fail to produce models compact enough for deployment on resource-constrained devices, such as IoT ones. Furthermore, the compression achieved by quantization techniques is constrained by the representation range of weights and activations, while, as noticed in our experiments, performance achieved through attention head pruning deteriorates dramatically as more heads are removed beyond a certain threshold.

Conclusion

Despite their remarkable performance in natural language understanding and generation tasks, Large Language Models are inherently memory- and compute-intensive, which hinders their deployment on resource-constrained devices. To tackle this issue, compression techniques such as quantization and pruning have emerged as promising solutions, alongside approaches based on knowledge distillation.

In this paper, we propose *DiXtill*, a novel approach to distilling explainable knowledge from an LLM into a lightweight, self-explainable neural architecture, leveraging local explanations as a complement to the usual prediction-based supervision. In particular, an additional loss term is introduced to quantify the degree of misalignment between attribution-like explanations of the LLM predictions, obtained with the IG method, and the attention-based explanations of the student.

Our experiments, involving the distillation of a pre-trained BERT-like LLM into an attention-enhanced bi-LSTM student model, reveal that our approach enables the student to achieve comparable performance to the teacher while also showing higher interpretability compared to traditional distillation. Furthermore, it allows for delivering a significantly higher compression ratio and speedup compared to other compression techniques such as post-training quantization and attention head pruning. This facilitates deployment and inference on resource-constrained devices, enabling more efficient and sustainable edge AI applications.

As a future direction, we will investigate how to address potential negative transfer issues, such as the presence of biases in the pre-trained model's explanations, which may negatively impact student performance. Furthermore, we will consider the distillation of different LLMs, as well as more lightweight student architectures to enable deployment on even more constrained devices such as microcontrollers.

Acknowledgements

This work has been partially supported by the "FAIR - Future Artificial Intelligence Research" project - CUP H23C22000860006.

Author contributions

All the authors contributed to the structuring of this paper, providing critical feedback and helping shape the research, analysis, and manuscript.

Funding

We acknowledge financial support from "FAIR - Future Artificial Intelligence Research" project - CUP H23C22000860006.

Availability of data and materials

Not applicable.

Declarations

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no Conflict of interest.

Received: 31 January 2024 Accepted: 16 April 2024
Published online: 04 May 2024

References

1. Brown T, et al. Language models are few-shot learners. Adv Neural Inf Process Syst. 2020;33:1877–901.
2. Chang Y, Wang X, Wang J, Wu Y, Yang, L, Zhu K, Chen H, Yi X, Wang C, Wang Y. et al A survey on evaluation of large language models. ACM Trans Intell Syst Technol 2023.
3. bchapter Cantini R, Cosentino C, Kilanioti I, Marozzo F, Talia D. Unmasking covid-19 false information on twitter: A topic-based approach with bert. In: International Conference on Discovery Science, Springer, 2023; pp. 126–140
4. bchapter Frantar E, Alistarh D. Sparsegpt: Massive language models can be accurately pruned inone-shot. In: International Conference on Machine Learning, PMLR, 2023; pp. 10323–10337
5. bchapter Marozzo F, Orsino A, Talia D, Trunfio P. Edge computing solutions for distributed machine learning. In: 2022 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), IEEE, 2022; pp. 1–8
6. Belcastro L, Cantini R, Marozzo F, Orsino A, Talia D, Trunfio P. Programming big data analysis: principles and solutions. J Big Data. 2022;9(1):4.
7. Ba J, Caruana R. Do deep nets really need to be deep? Adv Neural Inf Process Syst 27; 2014;
8. Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 2015

9. Saxena D, Cao J. Generative adversarial networks (GANS) challenges, solutions, and future directions. ACM Comput Surv (CSUR). 2021;54(3):1–42.
10. bchapter Alharbi R, Vu MN, Thai MT. Learning interpretation with explainable knowledge distillation. In: 2021 IEEE International Conference on Big Data (Big Data), pp. 705–714, 2021
11. Gou J, Yu B, Maybank SJ, Tao D. Knowledge distillation: a survey. Int J Comput Vision. 2021;129:1789–819.
12. Chen T, Kornblith S, Swersky K, Norouzi M, Hinton GE. Big self-supervised models are strong semi-supervised learners. Adv Neural Inf Process Syst. 2020;33:22243–55.
13. bchapter Zhang L, Song J, Gao A, Chen J, Bao C, Ma, K. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 3713–3722
14. Kim T, Oh J, Kim N, Cho S, Yun S.-Y. Comparing kullback-leibler divergence and mean squared error loss in knowledge distillation. arXiv preprint, 2021. arXiv:2105.08919
15. Tang R, Lu Y, Liu L, Mou L, Vechtomova O, Lin J. Distilling task-specific knowledge from bert into simple neural networks. arXiv preprint, 2019. arXiv:1903.12136
16. Zhu X, Li J, Liu Y, Ma C, Wang W. A survey on model compression for large language models. arXiv preprint, 2023. arXiv:2308.07633
17. Sanh V, Debut L, Chaumond J, Wolf T. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv preprint, 2019. arXiv:1910.01108
18. bchapter Liu, Y, Cao J, Li B, Hu W, Ding J, Li L. Cross-architecture knowledge distillation. In: Proceedings of the Asian Conference on Computer Vision, 2022, pp. 3396–3411
19. bchapter Jacob B, Kligys S, Chen B, Zhu M, Tang M, Howard A, Adam H, Kalenichenko D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018; pp. 2704–2713
20. Lin J, Tang J, Tang H, Yang S, Dang X, Han S. Awq: Activation-aware weight quantization for llm compression and acceleration. arXiv preprint, 2023. arXiv:2306.00978
21. Frantar E, Ashkboos S, Hoefler T, Alistarh D. Gptq: Accurate post-training quantization for generative pre-trained transformers. arXiv preprint, 2022. arXiv:2210.17323
22. Wang Z, Wohlwend J, Lei T. Structured pruning of large language models. arXiv preprint, 2019. arXiv:1910.04732
23. Kwon W, Kim S, Mahoney MW, Hassoun J, Keutzer K, Gholami A. A fast post-training pruning framework for transformers. Adv Neural Inf Process Syst. 2022;35:24101–16.
24. Michel P, Levy O, Neubig G. Are sixteen heads really better than one? Adv Neural Inf Process Syst 32; 2019;
25. Du M, Liu N, Hu X. Techniques for interpretable machine learning. Commun ACM. 2019;63(1):68–77.
26. Ali S, Abuhmed T, El-Sappagh S, Muhammad K, Alonso-Moral JM, Confalonieri R, Guidotti R, Del Ser J, Díaz-Rodríguez N, Herrera F. Explainable artificial intelligence (XAI): What we know and what is left to attain trustworthy artificial intelligence. Inf Fusion. 2023;99: 101805.
27. Rajani NF, McCann B, Xiong C, Socher R. Explain yourself! leveraging language models for commonsense reasoning. preprint, 2019. arXiv:1906.02361
28. Kumar P, Raman B. A bert based dual-channel explainable text emotion recognition system. Neural Netw. 2022;150:392–407.
29. bchapter Ribeiro MT, 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, San Francisco, CA, USA, August 13-17, 2016, 2016; pp. 1135–1144
30. Lundberg SM, Lee S.-I. A unified approach to interpreting model predictions. Adv Neural Inf Process Syst 30:2017
31. bchapter Sundararajan, M, Taly A, Yan Q. Axiomatic attribution for deep networks. In: International Conference on Machine Learning, PMLR, 2017; pp. 3319–3328
32. Gao Y, Gu S, Jiang J, Hong SR, Yu D, Zhao, L. Going beyond XAI: a systematic survey for explanation-guided learning. ACM Comput Surv 2022.
33. bchapter Zeng G, Kowsar Y, Erfani S, Bailey J. Generating deep networks explanations with robust attribution alignment. In: Asian Conference on Machine Learning, PMLR, 2021; pp. 753–768
34. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser, Ł, Polosukhin I. Attention is all you need. Adv Neural Inf Process Syst 30: 2017
35. Devlin J, Chang M-W, Lee K, Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint, 2018. arXiv:1810.04805
36. Radford A, Narasimhan K, Salimans T, Sutskever I, et al. Improving language understanding by generative pre-training 2018;
37. Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. arXiv preprint, 2014. arXiv:1409.0473
38. bchapter Ghorbani A, Abid A, Zou J. Interpretation of neural networks is fragile. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019; pp. 3681–3688
39. Yang Y, UY MCS, Huang A. FinBERT: a pretrained language model for financial communications 2020;<arxivurl>2006.08097</arxivurl>
40. Krishna S, Han T, Gu A, Pombra J, Jabbari S, Wu S, Lakkaraju H. The disagreement problem in explainable machine learning: a practitioner's perspective. arXiv preprint, 2022. arXiv:2202.01602

41. Kokhlikyan N, Miglani V, Martin M, Wang E, Alsallakh B, Reynolds J, Melnikov A, Kliushkina N, Araya C, Yan S, et al. Captum: A unified and generic model interpretability library for pytorch. arXiv preprint, 2022. arXiv:2009.07896

Publisher's Note

Riccardo Cantini is a researcher in computer engineering at the University of Calabria.

Alessio Orsino is a Ph.D. student ofcomputer engineering at the University of Calabria.

Domenico Talia is a professor of computer engineering at the Universityof Calabria and an adjunct professor at Fuzhou University.