Delft University of Technology

From large language models to small logic programs
Building global explanations from disagreeing local post-hoc explainers

Agiollo, Andrea; Cavalcante Siebert, Luciano; Murukannaiah, Pradeep K.; Omicini, Andrea

DOI
10.1007/s10458-024-09663-8

Publication date
2024

Document Version
Final published version

Published in
Autonomous Agents and Multi-Agent Systems

Citation (APA)
Agiollo, A., Cavalcante Siebert, L., Murukannaiah, P. K., & Omicini, A. (2024). From large language models to small logic programs: Building global explanations from disagreeing local post-hoc explainers. *Autonomous Agents and Multi-Agent Systems*, *38*(2), Article 32. https://doi.org/10.1007/s10458-024-09663-8

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Autonomous Agents and Multi-Agent Systems (2024) 38:32
https://doi.org/10.1007/s10458-024-09663-8

From large language models to small logic programs: building global explanations from disagreeing local post-hoc explainers

Andrea Agiollo[1] · Luciano Cavalcante Siebert[2] · Pradeep K. Murukannaiah[2] · Andrea Omicini[1]

Accepted: 19 June 2024
© The Author(s) 2024

Abstract

The expressive power and effectiveness of *large language models* (LLMs) is going to increasingly push intelligent agents towards sub-symbolic models for natural language processing (NLP) tasks in human–agent interaction. However, LLMs are characterised by a performance vs. transparency trade-off that hinders their applicability to such sensitive scenarios. This is the main reason behind many approaches focusing on *local* post-hoc explanations, recently proposed by the XAI community in the NLP realm. However, to the best of our knowledge, a thorough comparison among available explainability techniques is currently missing, as well as approaches for constructing *global* post-hoc explanations leveraging the local information. This is why we propose a novel framework for comparing state-of-the-art local post-hoc explanation mechanisms and for extracting logic programs surrogating LLMs. Our experiments—over a wide variety of text classification tasks—show how most local post-hoc explainers are loosely correlated, highlighting substantial discrepancies in their results. By relying on the proposed novel framework, we also show how it is possible to extract faithful and efficient global explanations for the original LLM over multiple tasks, enabling explainable and resource-friendly AI techniques.

Keywords Natural language processing · Post-hoc explanations · Symbolic knowledge extraction · EXplainable AI · Resource-friendly AI

✉ Andrea Agiollo
andrea.agiollo@unibo.it

Luciano Cavalcante Siebert
L.CavalcanteSiebert@tudelft.nl

Pradeep K. Murukannaiah
P.K.Murukannaiah@tudelft.nl

Andrea Omicini
andrea.omicini@unibo.it

[1] Dipartimento di Informatica - Scienza e Ingegneria (DISI), Alma Mater Studiorum-Università di Bologna, Cesena, Italy

[2] Delft University of Technology, Delft, The Netherlands

Published online: 08 July 2024

1 Introduction

Large language models (LLMs) represent the de-facto solution for dealing with complex natural language processing (NLP) tasks such as sentiment analysis [1], question answering [2], and many others [3]. The ever-increasing popularity of such data-driven approaches is largely due to their uncanny performance improvements against human counterparts over tasks such as grammar acceptability of a sentence [4] and text translation [5]. In this context, the foreseeable future of intelligent agent systems is likely to be deeply intertwined with LLMs. Intelligent agents exploiting NLP-enabled process for human-agent interaction as well as for inter-agent communication within complex Multi-Agent Systems (MASs) are going to become more and more popular [6, 7]. Natural language explanations are fundamental for improving agents expressiveness and explaining agent actions, beliefs, and reasoning [8], as well as argumentation and negotiation processes [9, 10]. However, leaning on LLMs—and Neural Networks (NNs) for NLP tasks in general—brings about a novel layer of complexity, requiring full comprehensibility of sub-symbolic components. Agent's observable behaviour should be understandable at every step, which requires the explainability of the sub-symbolic mechanisms in charge of the interaction—both with humans and agent-to-agent. Substantial reliance on LLMs does not make explanation extraction easy, as the LLM decision process is far from being transparent, given the complexity of popular architectures such as BERT [11], GPT [12], and T5 [13]. While powerful and empirically reliable, those models suffer from a performance vs. transparency trade-off [14, 15].

LLMs are *black-box* models, as they rely on the optimisation of their numerical subsymbolical components, which are mostly unreadable by humans. Mechanisms are then needed that could make the reasoning process of LLM black-boxes somehow observable and understandable by humans. To this aim, a few different explainability approaches have been recently proposed, which mostly focus on *Local Post-hoc Explainer (LPE)* mechanisms. An LPE represents a popular solution to explain the reasoning process by highlighting how different portions of the input sample impact differently the produced output, by assigning a relevance score to each input component. These approaches apply to single instances of input sample—they are *local*—and to optimised LLM—they are *post-hoc*. While popular, such approaches do not give information about the general reasoning principle of the underlying LLM, as they cannot produce a global view. Moreover, despite a broad variety of LPE approaches, the state of the art lacks a fair comparison among them. A common trend for proposals of novel explanation mechanisms is to highlight its advantages through a set of tailored experiments. This hinders comparison fairness, making it very difficult to identify the best approach for explanations of NLP models, or even to determine whether a best approach exists.

This is why in the following we present a framework for comparing several well-known LPE mechanisms for text classification in NLP—first introduced in [16]. Aiming at obtaining comparison fairness, we rely on the aggregation of the local explanations obtained by each local post-hoc explainer into a set of global impact scores. The scores identify the set of concepts that best describe the underlying NLP model from the perspective of each LPE. The concepts, along with their aggregated impact scores, are then compared for each LPE against other LPE counterparts. The comparison between the aggregated global impact scores rather than the single explanations is justified by the locality of LPE approaches. Indeed, it is reasonable for local explanations of different LPEs to differ somehow, depending on the approach design, therefore making it complex to compare the quality of two LPEs over the same sample. However, it is also expected for the aggregated global impacts

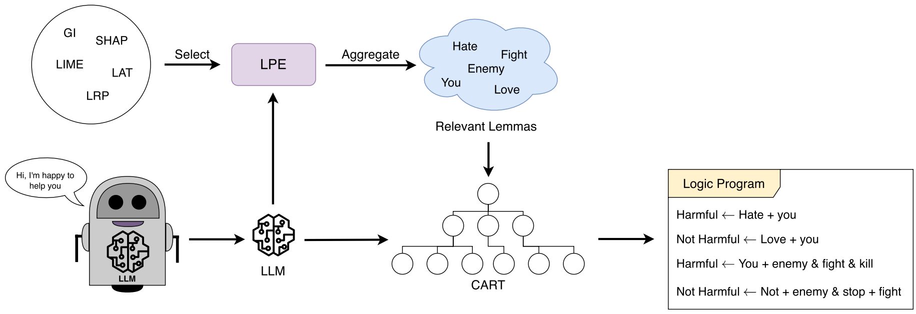


Fig. 1 Overview of the GELPE extraction process. The LLM belonging to a smart agent is examined by a single LPE mechanism, generating a set of relevant lemmas (see Eq. 16 for more details). The set of relevant lemmas, along with the LLM's predictions for a set of available inputs, are used to optimise the CART model to mimic the LLM behaviour. Thereafter, the CART model can be converted quickly into a logic program equivalent to the starting LLM.

to be aligned between different LPEs as they are applied to the same NN, which leverages the same set of relevant concepts for its inference. Therefore, when comparing the aggregated impact scores of different LPEs, we expect them to be correlated—at least up to a certain extent.

Subsequently, given the lack of available *global post-hoc explainers*, we consider enabling the extraction of global explanations from the output of a single LPE. Here, we refer to *global* as expressing the totality of knowledge acquired by the LLM at training time, thus being representative of the full reasoning principles of the LLM at hand. We extend the LPE comparison framework first introduced in [16] leveraging a a neuro-symbolic process [17–21] to build global explanations from LPEs. In this context, we aim at extracting the LLM knowledge under the form of a logic program equivalent to the sub-symbolic model at hand, similarly to [22, 23]. More in detail, the presented knowledge extraction framework—namely, Global Explanations from Local Post-hoc Explainers (GELPE)—relies on the output of a single LPE to identify the set of most relevant components in sentences, and optimise a transparent-by-design—such as Classification and Regression Tree (CART)—surrogate model to mimic the LLM predictions. Once the transparent model is optimised, an equivalent logic program is extracted from the model, allowing for the inspection of the global reasoning process of the LLM. Figure 1 summarises the GELPE's working process. While simple, this approach represents up to our knowledge the first mechanism for building global explanations of LLMs for text classification accounting for the available local explanations. As such, the proposed approach is likely to represent a desirable tool for the trustworthy and explainable AI community, as it allows opening LLM black-boxes while keeping the explanations complexity bounded. Identifying small and efficient surrogate programs over several tasks, the proposed framework enables the deployment of intelligent techniques over resource-constrained environments where LLMs represent a limited solution [24, 25].

We test the proposed framework over a large set of text classification domains, ranging from simple scenarios—e.g., spam text classification [26, 27]—to challenging tasks such as the Moral Foundation Twitter Corpus (MFTC) [28]. Possibly surprisingly, our experiments show how the explanations of different LPEs are far from being correlated, highlighting how explanation quality is highly dependent on the chosen eXplainable Artificial Intelligence (xAI) approach and the respective scenario at hand. There are huge discrepancies in the results of different state-of-the-art local explainers, each of which identifies a set of relevant concepts that largely differs from the others—at least in terms of relative impact scores. These results highlight the fragility of xAI approaches for NLP, caused mainly by the complexity of large NN models, their inclination to extreme fitting of data and the lack of sound techniques for comparing xAI mechanisms. Notably, the proposed experiments also highlights how GELPE enables the extraction of reliable surrogate logic programs from LLMs with high fidelity over a broad set of datasets. The extracted knowledge is not only faithful to the original model, but also quite simple, as the complexity of the logic program is kept bounded depending on the number of relevant lemmas selected. Throughout our experimental evaluation, we analyse the computation requirements of the proposed extraction process and the efficiency of the extracted logic program. Numerical results highlight the efficiency of the extracted surrogate model, improving over the original LLM in terms of required processing time and consumed energy. The results show how the proposed framework enables the deployment of intelligent solutions over resource-constrained environments via identifying transparent surrogate models. Also, we highlight that leveraging on LLMs to tackle a learning task in NLP does not always represent the best option, as alternative equivalent solutions that are simple, small and transparent can be available [29, 30].

Contributions: We summarise our contributions as follows:

- We present the first framework for comparing explanations obtained leveraging different LPEs over LLMs. The proposed scheme is designed to assert the correlation level of LPEs over a broad set of input sentences.
- We test the correlation performance of seven different LPEs over nine different NLP datasets, showcasing how state-of-the-art LPEs strongly disagree.
- We present GELPE, the first framework for extracting global explanations from the output of LPE processes, enabling the extraction of logic rules from LLMs.
- We study the performance of GELPE considering its fidelity with respect to LLMs, the complexity of the extracted rules and its achievable efficiency improvements, showcasing encouraging results.

Organization: Sect. 2 discusses the basic concepts of available explanation mechanisms in NLP, along with the required discussion between local and global explanations. Section 3 presents the methodology used in this paper for comparing LPE mechanisms and building global explanations from LPE's outputs. The experimental evaluation of our methodology is made available in Sect. 4, in which we first focus on the comparison between the available LPEs in Sect. 4.3, while Sect. 4.4 presents the knowledge extraction results. Consequently, Sect. 5 discusses the limitations of the proposed methodology, whereas Sect. 6 concludes the paper with some insight into the possible extensions of our work.

Glossary: Table 1 summarises notations used in the article.

2 Background: explanation mechanisms in NLP

The set of explanations extraction mechanisms available in the xAI community are often categorised along two main axis [31, 32]: (i) *local* against *global* explanations, and (ii) *self-explaining* against *post-hoc* approaches. In the former context, *local* identifies the

Table 1 Summary of glossary.

Acronym	Definition
NN	Neural Network
NLP	Natural Language Processing
xAI	eXplainable Artificial Intelligence
LPE	Local Post-hoc explainer
MFTC	Moral Foundation Twitter Corpus
LLM	Large Language Model
CART	Classification And Regression Tree
GELPE	Global Explanations from Local Post-hoc Explainers
SHAP	SHapley Additive exPlanations
BLM	Black Lives Matter
ALM	All Lives Matter
BLT	Baltimore protests
DAV	hate speech and offensive language
ELE	2016 presidential election
MT	MeToo movement
SND	hurricane Sandy
GS	Gradient Sensitivity analysis
GI	Gradient × Input
LRP	Layer-wise Relevance Propagation
LAT	Layer-wise Attention Tracing
LIME	Local Interpretable Model-agnostic Explanations

set of explainability approaches that given a single input produce an explanation of the reasoning process followed by the NN model to output its prediction for the given input [33]. In contrast, *global* explanations aim at expressing the reasoning process of the NN model as a whole [34, 35]. Given the complexity of the NN models leveraged for tackling most NLP tasks, it is worth noticing how there is a significant lack of *global* explainability systems, whereas a variety of *local* xAI approaches are available [36, 37].

About the latter aspect, we define *post-hoc* as those set of explainability approaches which apply to an already optimised black-box model for which it is required to obtain some sort of insight [38]. Therefore, a *post-hoc* approach requires additional operations to be performed after that the model outputs its predictions [39]. Conversely, inherently explainable—*self-explaining*—mechanisms aim at building a predictor having a transparent reasoning process by design—e.g., CART [40]. Therefore, a self-explaining approach can be seen as generating the explanation along with its prediction, using the information emitted by the model as a result of the prediction process [39].

In the context of *local post-hoc* explanation approaches, a popular solution in NLP is to explain the reasoning process by highlighting how different portions of the input sample impact differently the produced output, by assigning a relevance score to each input component. The relevance score is then highlighted by using some saliency map to ease the visualisation of the obtained explanation. Therefore, it is also common for local post-hoc explanations to be referred to as *saliency* approaches, as they aim at highlighting salient components.

3 Methodology

In this section, we present our methodology for comparing LPE mechanisms and building global explanations from LPE's outputs. We first overview the proposed approach in Sect. 3.1. Subsequently, the set of LPE mechanisms adopted in our experiments are presented in Sect. 3.2, and the aggregation approaches leveraged to obtain global impact scores from LPE outputs are described in Sect. 3.3. In Sect. 3.4 we present the metrics used to identify the correlation between LPEs. Finally, in Sect. 3.5 we propose GELPE as a novel methodology to build global explanations of LLMs on top of LPE's outputs.

3.1 Overview

Measuring different LPE approaches over single local explanations is a complex task. This is why we first consider measuring how much LPEs correlate with each other over a set of fixed samples. The underlying assumption of our framework is that various LPE techniques aim at explaining the same NN model used for prediction. Therefore, while explanations may differ over local samples, one could reasonably assume that reliable LPEs when applied over a vast set of samples—sentences or set of sentences—should converge to similar (correlated) results. Indeed, the underlying LLM considers being relevant for its inference always the same set of concepts—lemmas. A lack of correlation between different LPE mechanisms would hint to the existence of a conflict among the set of concepts that each explanation mechanism considers as relevant for the LLM—thus making at least one, if not all, of the explanations unreliable.

We first analyse the correlation between a set of LPEs over the same pool of samples, and define ϵ_{NN} as a LPE technique applied to a NN model at hand. Being local, ϵ_{NN} is applied to the single input sample $\mathbf{x}_i$, producing as output one impact score for each component (token) of the input sample l_k. Throughout the remainder of the paper, we consider l_k to be the lemmas corresponding to the input components. Mathematically, we define the output impact score for a single token or its corresponding lemma as $j\big(l_k, \epsilon_{NN}(\mathbf{x}_i)\big)$. Depending on the given ϵ_{NN}, the corresponding impact score j may be associated with a single label, making j a scalar value, or with a set of labels, making j a vector—one scalar value for each label. To enable the comparison between different LPE, we define the aggregated impact scores of a LPE mechanism over a NN model and a set of samples $\mathcal{S}$ as $\epsilon_{NN}(\mathcal{S})$. In our framework we obtain $\epsilon_{NN}(\mathcal{S})$ aggregating $\epsilon_{NN}(\mathbf{x}_i)$ for each $\mathbf{x}_i \in \mathcal{S}$ using an aggregation operation $\mathcal{A}$—mathematically:

$$\epsilon_{NN}(\mathcal{S}) = \mathcal{A}\Big(\big\{\epsilon_{NN}(\mathbf{x}_i)\, \textit{for each}\, \mathbf{x}_i \in \mathcal{S}\big\}\Big). \tag{1}$$

By defining a correlation metric $\mathcal{C}$, we obtain from Eq. 1 the following for describing the correlation between two LPE techniques:

$$\begin{aligned}\mathcal{C}\big(\epsilon_{NN}(\mathcal{S}), \epsilon'_{NN}(\mathcal{S})\big) = \mathcal{C}\Big(&\mathcal{A}\Big(\big\{\epsilon_{NN}(\mathbf{x}_i)\, \textit{for each}\, \mathbf{x}_i \in \mathcal{S}\big\}\Big),\\ &\mathcal{A}\Big(\big\{\epsilon'_{NN}(\mathbf{x}_i)\, \textit{for each}\, \mathbf{x}_i \in \mathcal{S}\big\}\Big)\Big)\end{aligned} \tag{2}$$

where ϵ_{NN} and ϵ'_{NN} are two LPE techniques applied to the same NN model.

The aggregated explanations $\epsilon_{NN}(\mathcal{S})$ obtained from LPE's outputs can also be leveraged as a starting point for building transparent surrogate models of the original LLM, as they highlight the impact of each lemma or token in the LLM decision process. Constructing a transparent surrogate model allows for extracting explanations of the *global* reasoning process of the black-box LLM, enabling knowledge extraction, model debugging, and interaction with a human user or other intelligent agents. To this extent, we here propose GELPE as a novel framework for constructing a logic program—represented as a set of sequential propositional rules—that mimics the LLM behaviour starting from a set of locally relevant lemmas $\epsilon_{NN}(\mathcal{S})$, extracted using a single LPE. More in detail, GELPE relies on transparent-by-design models such as CART optimised over the LLM outputs, rather than the dataset considered.

We rely on CART models as they represent one of the easiest and most reliable approaches to identify human-readable rules—under the form of trees—from complex structured data. In summary, the optimisation of CART models involves selecting input variables and split points on those variables until a suitable tree is constructed. The selection of which input variables and split points to use is performed using a greedy algorithm aiming at minimising a given cost function. Finally, the tree construction process ends using a predefined stopping criterion, such as a minimum number of training instances assigned to each leaf node of the tree. The set of tree-structured rules extracted using CART can be easily translated into a list of sequential, human-readable expressions that contain logic expressions over the input variables, by extracting one rule for each leaf used in the CART model. Therefore, CART represents a very popular solution for extracting explanations from fuzzy data or black-box classifiers, trying to mimic their outputs. However, a thorough background on CART models is out of the scope of this paper and we refer interested readers to [40].

Since CART relies on structured—usually tabular—data to perform optimisation and inference, we convert the input sentences into a binary format, expressing the presence or absence of relevant lemmas and their combinations. The binarised input is used to optimise the underlying CART model, from which it is possible to extract the equivalent logic program $\mathcal{P}$. Mathematically, we represent the knowledge extraction procedure as:

$$\mathcal{P} = \mathcal{H}\big\{(bin_{\epsilon_{NN}(\mathcal{S})}(\mathbf{x}_i),\ NN(\mathbf{x}_i))\ \forall\ \mathbf{x}_i \in \mathcal{S}\big\}, \tag{3}$$

where $\mathcal{H}$ identifies the transparent-by-design models used to extract the explanations logic program $\mathcal{P}$, $bin_{\epsilon_{NN}(\mathcal{S})}$ represents the binarization process used to convert the sentence $\mathbf{x}_i$ into a corresponding binary vector of lemmas occurences and $NN(\mathbf{x}_i)$ identifies the output of the LLM when fed with input sentence $\mathbf{x}_i$.

3.2 Local post-hoc explanations

In our framework, we consider seven different LPE approaches for extracting local explanations $j\big(l_k, \epsilon_{NN}(\mathbf{x}_i)\big)$ from an input sentence $\mathbf{x}_i$ and the trained LLM—identified as NN. The seven LPEs are selected in order to represent as faithfully as possible the state-of-the-art of xAI approaches in NLP. Subsequently, we briefly describe each of the seven selected LPEs. However, a detailed analysis of these LPEs is out of the scope of this paper and we refer interested readers to [33, 39, 41].

3.2.1 Gradient sensitivity analysis (GS)

The Gradient Sensitivity Analysis (GS) probably represents the simplest approach for assigning relevance scores to input components. GS relies on computing gradients over inputs components as $\frac{\delta f_{\tau_m}(\mathbf{x}_i)}{\delta \mathbf{x}_{i,k}}$, which represents the derivative of the output with respect to the the k^{th} component of $\mathbf{x}_i$. Following this approach local impact scores of an input component can be thus defined as:

$$j\big(l_k, \epsilon_{NN}(\mathbf{x}_i)\big) = \frac{\delta f_{\tau_m}(\mathbf{x}_i)}{\delta \mathbf{x}_{i,k}}, \tag{4}$$

where $f_{\tau_m}(\mathbf{x}_i)$ represents the predicted probability distribution of an input sequence $\mathbf{x}_i$ over a target class τ_m. While simple, GS has been shown to be an effective approach for understanding approximate input components relevance. However, this approach suffers from a variety of drawbacks, mainly linked with its inability to define negative contributions of input components for a specific prediction—i.e., negative impact scores.

3.2.2 Gradient × input (GI)

Aiming at addressing few of the limitations affecting GS, the Gradient × Input (GI) approach defines the relevance scores assignment as GS multiplied—element-wise—with $\mathbf{x}_{i,k}$ [42]. Therefore, mathematically speaking, GI impact scores are defined as:

$$j\big(l_k, \epsilon_{NN}(\mathbf{x}_i)\big) = \mathbf{x}_{i,k} \cdot \frac{\delta f_{\tau_m}(\mathbf{x}_i)}{\delta \mathbf{x}_{i,k}}, \tag{5}$$

where notation follows the one of Eq. 4. Being very similar to GS, GI also inherits most of its limitations.

3.2.3 Layer-wise relevance propagation (LRP)

Building on top of gradient-based relevance scores mechanisms—such as GS and GI—, Layer-wise Relevance Propagation (LRP) proposes a novel mechanism relying on conservation of relevance scores across the layers of the NN at hand. Indeed, LRP relies on the following assumptions: (i) NN can be decomposed into several layers of computation; (ii) there exists a relevance score $R_d^{(l)}$ for each dimension $\mathbf{z}_d^{(l)}$ of the vector $\mathbf{z}^{(l)}$ obtained as the output of the l^{th} layer of the NN; and (iii) the total relevance scores across dimensions should propagate through all layers of the NN model, mathematically:

$$f(\mathbf{x}) = \sum_{d \in L} R_d^{(L)} = \sum_{d \in L-1} R_d^{(L-1)} = \cdots = \sum_{d \in 1} R_d^{(1)}, \tag{6}$$

where, $f(\mathbf{x})$ represents the predicted probability distribution of an input sequence $\mathbf{x}$, and L the number of layers of the NN at hand. Moreover, LRP defines a propagation rule for obtaining $R_d^{(l)}$ from $R^{(l+1)}$. However, the derivation of the propagation rule is out of the scope of this paper, thus we refer interested readers to [43, 44]. In our experiments we consider as impact scores the relevance scores of the input layer, namely $j\big(l_k, \epsilon_{NN}(\mathbf{x}_i)\big) = R_d^{(1)}$.

3.2.4 Layer-wise attention tracing (LAT)

Since LLMs rely heavily on self-attention mechanisms [45], recent efforts propose to identify input components relevance scores analysing solely the relevance scores of attentions heads of LLM models, introducing Layer-wise Attention Tracing (LAT) [46, 47]. Building on top of LRP, LAT proposes to redistribute the inner relevance scores $R^{(l)}$ across dimensions using solely self-attention weights. Therefore, LAT defines a custom redistribution rule as:

$$R_i^{(l)} = \sum_k \sum_h \mathbf{a}^{(h)} R_{k,h}^{(l+1)}, \tag{7}$$

where, h corresponds to the attention head index, while $\mathbf{a}^{(h)}$ are the corresponding learnt weights of the attention head and k is such that i is input for neuron k. Similarly to LRP, we here consider as impact scores the relevance scores of the input layer, namely $j\big(l_k, \epsilon_{NN}(\mathbf{x}_i)\big) = R^{(1)}$.

3.2.5 Integrated gradient (HESS)

Motivated by the shortcomings of previously proposed gradient-based relevance score attribution mechanisms—such as GS and GI—, Sundararajan et al. [48] propose a novel Integrated Gradient approach. The proposed approach aims at explaining the input sample components relevance by integrating the gradient along some trajectory of the input space, which links some baseline value $\mathbf{x}'_i$ to the sample under examination $\mathbf{x}_i$. Therefore, the relevance score of the k^{th} input component of the input sample $\mathbf{x}_i$ is obtained following

$$j\big(l_k, \epsilon_{NN}(\mathbf{x}_i)\big) = \left(\mathbf{x}_{i,k} - \mathbf{x}'_{i,k}\right) \cdot \int_{a=0}^{1} \frac{\delta f(\mathbf{x}'_i + t \cdot (\mathbf{x}_i - \mathbf{x}'_i))}{\delta \mathbf{x}_{i,k}} \, dt, \tag{8}$$

where $\mathbf{x}_{i,k}$ represents the k^{th} component of the input sample $\mathbf{x}_i$. By integrating the gradient along an input space trajectory, the authors aim at addressing the locality issue of gradient information. In our experiments we refer to the Integrated Gradient approach as HESS, as for its implementation we rely on the integrated hessian library available for hugging face models.[1]

3.2.6 SHapley additive exPlanations (SHAP)

SHapley Additive exPlanations (SHAP) relies on Shapley values to identify the contribution of each component of the input sample toward the final prediction distribution. The Shapley value concept derives from game theory, where it represents a solution for a cooperative game, found assigning a distribution of a total surplus generated by the players coalition. SHAP computes the impact of an input component as its marginal contribution toward a label τ_m, computed deleting the component from the input and evaluating the output discrepancy. Firstly defined for explaining simple NN models [36], in our experiments

[1] https://github.com/suinleelab/path_explain.

we leverage the extension of SHAP supporting transformer models such as BERT [49], available in the SHAP python library.[2]

3.2.7 Local interpretable model-agnostic explanations (LIME)

Similarly to SHAP, Local Interpretable Model-agnostic Explanations (LIME) relies on input sample perturbation to identify its relevant components. Here, the predictions of the NN at hand are explained via learning an explainable surrogate model [37]. In detail, in order to obtain its explanations LIME constructs a set of samples from the perturbation of the input observation under examination. The constructed samples are considered to be close to the observation to be explained from a geometric perspective, thus considering small perturbation of the input. The explainable surrogate model is then trained over the constructed set of samples, obtaining the corresponding local explanation. Given an input sentence, we here consider obtaining its perturbed version via words—or tokens—removal and words substitution. In our experiments, we rely on the already available LIME python library.[3]

3.3 Aggregating local explanations

Once local explanations of the NN model are obtained for each input sentence, we aggregate them to obtain a global list of concept impact scores. Before aggregating the local impact scores, we convert the words composing local explanations into their corresponding lemmas-i.e., concepts-to avoid issues when aggregating different words expressing the same concept-e.g., hate and hateful. As no bullet-proof solution exists for the aggregation of different impact scores, we adopt four different approaches in our experiments, namely:

Sum A simple summation operation is leveraged to obtain the aggregated score for each lemma. While simple this aggregation approach is effective when dealing with additive impact scores such as SHAP values. However, it suffers from lemma frequency issues, as it tends to overestimate frequent lemmas with average low impact scores. Global impact scores are here defined as $J(l_k, \epsilon_{NN}) = \sum_{i=1}^{N} j\big(l_k, \epsilon_{NN}(\mathbf{x}_i)\big)$. Therefore, we define $\mathcal{A}$ as

$$\mathcal{A}\big(\{\epsilon_{NN}(\mathbf{x}_i)\,for\ each\ \mathbf{x}_i \in \mathcal{S}\}\big) = \left\{ \sum_{i=1}^{N} j\big(l_k, \epsilon_{NN}(\mathbf{x}_i)\big)\,for\ each\ l_k \in \mathcal{S} \right\}. \tag{9}$$

Absolute sum Here we sum the absolute values of the local impact scores—rather than their true values—to increase the awareness of global impact scores towards lemmas having both high positive and high negative impact over some sentences. Mathematically, we obtain aggregated scores as $J(l_k, \epsilon_{NN}) = \sum_{i=1}^{N} |j\big(l_k, \epsilon_{NN}(\mathbf{x}_i)\big)|$.

[2] https://github.com/slundberg/shap.

[3] https://github.com/marcotcr/lime.

$$\mathcal{A}\left(\left\{\epsilon_{NN}(\mathbf{x}_i)\,for\ each\ \mathbf{x}_i \in \mathcal{S}\right\}\right) = \left\{\sum_{i=1}^{N} |j\left(l_k, \epsilon_{NN}\left(\mathbf{x}_i\right)\right)|\,for\ each\ l_k \in \mathcal{S}\right\}. \tag{10}$$

Average Similar to the sum operation, here we obtain aggregated scores averaging local impact scores, thus avoiding possible overshooting issues arising when dealing with very frequent lemmas. Mathematically, we define $J(l_k, \epsilon_{NN}) = \frac{1}{N} \cdot \sum_{i=1}^{N} j\left(l_k, \epsilon_{NN}\left(\mathbf{x}_i\right)\right)$.

$$\mathcal{A}\left(\left\{\epsilon_{NN}(\mathbf{x}_i)\,for\ each\ \mathbf{x}_i \in \mathcal{S}\right\}\right) = \left\{\frac{1}{N} \cdot \sum_{i=1}^{N} j\left(l_k, \epsilon_{NN}\left(\mathbf{x}_i\right)\right)\,for\ each\ l_k \in \mathcal{S}\right\}. \tag{11}$$

Absolute average Similarly to absolute sum, here we average absolute values of local impact scores for better-managing lemmas with a skewed impact as well as tackling frequency issues. Global impact scores are here defined as $J(l_k, \epsilon_{NN}) = \frac{1}{N} \cdot \sum_{i=1}^{N} |j\left(l_k, \epsilon_{NN}\left(\mathbf{x}_i\right)\right)|$.

$$\mathcal{A}\left(\left\{\epsilon_{NN}(\mathbf{x}_i)\,for\ each\ \mathbf{x}_i \in \mathcal{S}\right\}\right) = \left\{\frac{1}{N} \cdot \sum_{i=1}^{N} |j\left(l_k, \epsilon_{NN}\left(\mathbf{x}_i\right)\right)|\,for\ each\ l_k \in \mathcal{S}\right\}. \tag{12}$$

Since the selection of the aggregation mechanism may influence the correlation between different LPEs, in our experiments we analyse LPEs correlation over the same aggregation scheme. Moreover, we also analyse how aggregation impacts the impact scores correlation over the same LPE, highlighting how leveraging the absolute value of impact score is highly similar to adopting its true value—see Sect. 4.3.2.

3.4 Comparing explanations

Each aggregated global explanation J depends on a corresponding label τ_m since LPEs produce either a scalar impact value for a single τ_m or a vector of impact scores for each τ_m. Therefore, recalling Sect. 3.3, we can define the set of aggregated global scores depending on the label they refer to as following:

$$\mathcal{J}_{\tau_m}\left(\epsilon_{NN}, \mathcal{S}\right) = \left\{J\left(l_k, \epsilon_{NN}\right)|\tau_m\,for\ each\ l_k \in \mathcal{S}\right\}. \tag{13}$$

$\mathcal{J}_{\tau_m}\left(\epsilon_{NN}, \mathcal{S}\right)$ represents a distribution of impact scores over the set of lemmas—i.e., concepts—available in the samples set for a specific label. To compare the distributions of impact scores extracted using two LPEs—i.e., $\mathcal{J}_{\tau_m}\left(\epsilon_{NN}, \mathcal{S}\right)$ and $\mathcal{J}_{\tau_m}\left(\epsilon'_{NN}, \mathcal{S}\right)$—we use Pearson correlation, which is defined as the ratio between the covariance of two variables and the product of their standard deviations, and it measures their level of linear correlation. The selected correlation metric is applied to the normalised impact scores. Indeed, different LPEs produce impact scores that may differ relevantly in terms of their magnitude. Normalising the impact scores, we map impact scores to a fixed interval, allowing for a direct comparison of $\mathcal{J}_{\tau_m}$ over different ϵ_{NN}. Mathematically, we refer to the normalised global impact scores as $||\mathcal{J}_{\tau_m}||$. Therefore, we define the correlation score between two sets of global impact scores for a single label as:

$$\begin{aligned}\rho\left(||\mathcal{J}_{\tau_m}\left(\epsilon_{NN}, \mathcal{S}\right)||, ||\mathcal{J}_{\tau_m}\left(\epsilon'_{NN}, \mathcal{S}\right)||\right) = \rho\big(&||\left\{J\left(l_k, \epsilon_{NN}\right)|\tau_m\,for\ each\ l_k \in \mathcal{S}\right\}||,\\ &||\left\{J\left(l_k, \epsilon'_{NN}\right)|\tau_m\,for\ each\ l_k \in \mathcal{S}\right\}||\big)\end{aligned} \tag{14}$$

where ρ refers to the Pearson correlation used to compare couples of $\mathcal{J}_{\tau_m}\left(\epsilon_{NN}, \mathcal{S}\right)$. Throughout our analysis we experimented with similar correlation metrics, such as Spearman correlation and simple vector distance—similarly to [50]—, obtaining similar results. Therefore, to avoid redundancy we here show only the Pearson correlation results. Throughout our experiments, we consider a simple *min-max* normalisation process, scaling the scores to the range [0, 1].

As we aim at obtaining a measure of similarity between LPEs applied over the same set of samples, we can average the correlation scores ρ obtained for each label τ_m over the set of labels $\mathcal{T}$. Therefore, we mathematically define the correlation score of two LPEs, putting together Eqs. 13, 2 and 14 as:

$$\mathcal{C}\left(\epsilon_{NN}(\mathcal{S}), \epsilon'_{NN}(\mathcal{S})\right) = \frac{1}{M} \cdot \sum_{m=1}^{M} \rho\left(\|\mathcal{J}_{\tau_m}\left(\epsilon_{NN}, \mathcal{S}\right)\|, \|\mathcal{J}_{\tau_m}\left(\epsilon'_{NN}, \mathcal{S}\right)\|\right) \tag{15}$$

where M is the total number of labels, belonging to $\mathcal{T}$.

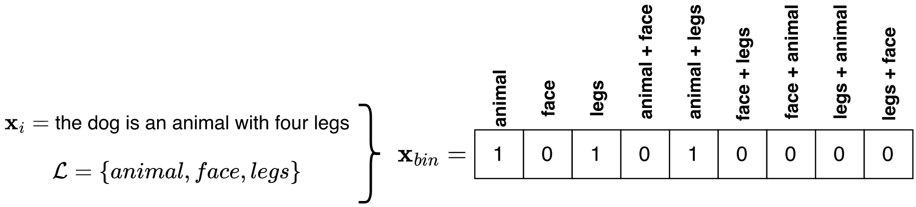
3.5 GELPE: global explanations from LPEs

Although useful, local explanations are limited, as they do not highlight the general reasoning principle of the underlying model, but rather focus solely on relevant input components for a specific prediction. Aiming at overcoming such limitations, we here present GELPE as the first—up to our knowledge—framework for extracting global explanations from LPEs. Relying on LPE outputs, GELPE allows for the adoption of reliable local extraction mecanisms, while extending their impact to the global reasoning process of the black-box model. Figure 1 presents an overview of GELPE's working process.

The aggregated explanations $\epsilon_{NN}(\mathcal{S})$ obtained from a single LPE's output are leveraged as a starting point for building a transparent surrogate model of the original LLM. GELPE relies on transparent-by-design models such as CART optimised over the LLM outputs, rather than the dataset considered. As described in Eq. 3, during the optimisation process of the CART model, input sentences are converted into a binary format, expressing the presence or absence of relevant lemmas and their combinations. In order to convert a sentence $\mathbf{x}_i$ into its binary format, we consider the $\mathcal{K}$ most valuable lemmas for each class identified during the aggregation process presented in Sect. 3.3. The $\mathcal{K}$ most valuable lemmas are the ones with the highest aggregated impact scores over a set of sample sentences for a single LPE mechanism. To avoid relying only on keywords, and accounting instead for more complex constructs, we also consider the set of skipgrams built from the combination of the single $\mathcal{K}$ most valuable lemmas. In this context, skipgrams define co-occurences of relevant lemmas over a span of limited tokens sequences [51]. With such a procedure we build a set of valuable lemmas and sequences $\mathcal{L}$ defined as:

$$\mathcal{L} = \{(L_i), (L_i, L_j), (L_i, L_j, L_k), \dots \ \forall\, i, j, k \in \mathcal{K}\}, \tag{16}$$

where L_i represents the lemma in the i^{th} position of the sorted lemmas list—in terms of relevance—, and $(L_i, \dots, L_j)$ represent the concatenation of two or more lemmas. Once the set of most relevant lemmas and corresponding sequences $\mathcal{L}$ are available, we can define the binarized version of an input sentence as the binary vector that identify the presence or absence of each lemma and sequence in the considered sentence. Mathematically, the binarisation function can be defined as the following:



$\mathbf{x}_i$ = the dog is an animal with four legs

$\mathcal{L} = \{animal, face, legs\}$

$\mathbf{x}_{bin} =$

animal	face	legs	animal + face	animal + legs	face + legs	face + animal	legs + animal	legs + face
1	0	1	0	1	0	0	0	0

Fig. 2 Sentence binarization approach in GELPE.

$$\mathbf{x}_{bin} = bin_{\epsilon_{NN}(\mathcal{S})}(\mathbf{x}_i) = \mathbb{1}(x_{i,j} \in \mathcal{L}) \,||\, \mathbb{1}(skip(x_{i,j-n}, \dots, x_{i,j}) \in \mathcal{L}) \quad \forall j \in \mathbf{x}_i, \tag{17}$$

where $x_{i,j}$ represent the components—i.e., tokens or lemmas—of the input sentence $\mathbf{x}_i$, $skip(x_{i,j-n}, \dots, x_{i,j})$ the corresponding skipgrams built from the last n input components, and $\mathbb{1}$ represents the indicator function, being equal to 1 if the lemma/skipgram belongs to $\mathcal{L}$ and 0 otherwise. Finally, || represents the concatenation operation between vectors. As an example, consider the input sentence *the dog is an animal with four legs* and the set of most relevant lemmas extracted by a given LPE to be $\mathcal{L} = \{animal, face, legs\}$. Then the corresponding binarised version of the input sentence is shown in Fig. 2, where the + symbol is used to identify the concatenation of two relevant lemmas inside a sentence—i.e., *lemma1* + *lemma2* can be interpreted as *lemma1 followed by lemma2*.

The binarised input is used to optimise the underlying CART model, from which it is possible to extract the equivalent logic program $\mathcal{P}$—see Eq. 3. $\mathcal{P}$ is extracted by identifying one rule for each leaf used in the CART model optimised over the LLM outputs. The obtained logic program $\mathcal{P}$ represents an explanation of the black-box LLM in the form of a set of sequential propositional rules containing lemmas, sequences of lemmas, and negations thereof. Extracted rules are sequential, meaning that each propositional rule applies if and only if the previous ones were not valid. As GELPE relies on the CART model, the extracted rules can only identify the presence or absence of a specific set of keywords and sequences, which represents a limitation of such approach. However, varying the value of $\mathcal{K}$ and the length and expressiveness of the skipgram construction process, the GELPE extraction procedure can be tuned to consider sequences of lemmas as complex as it is needed to fit well the LLM reasoning process. To keep the complexity of the extraction process under control, throughout our experiments we consider relying at most on (2,5)-skipgrams—i.e., building sequences of lemmas of length at most two which are contained over the span of five input tokens. An example of the GELPE extracted knowledge, along with the analysis of its correctness is made available in Sect. 4.4.3.

4 Experiments

In this section we present the setup and results of our experiments. More in detail, we first analyse the set of datasets used in our experimental evaluation in Sect. 4.1, along with the model training details and its obtained performance in Sect. 4.2. We then focus on the comparison between the available LPEs, showing the correlation between their explanations in

Table 2 Size of the considered datasets.

	SMS	YOUTUBE	TREC	ALM	BLM	BLT	ELE	MT	SND
Number of samples	5574	2403	4965	4424	5257	5593	5358	4891	4591

Sect. 4.3. Section 4.4 presents the knowledge extraction results, analysing the performance of the knowledge extractor model, along with the complexity of the extracted knowledge. Finally, we analyse the efficiency of the knowledge extraction model, showcasing the improvements in terms of time and energy consumption over the LLM counterpart. The source code of our framework and experiments is publicly available.[4]

4.1 Datasets

In our experiments, we aim at analysing the correlation among different LPEs and the feasibility of global knowledge extraction from LLM over a large set of scenarios. Therefore, we consider an heterogeneous set of datasets targetting text classification tasks, ranging from easy to complex setups. More in detail, we consider targetting the SMS [26] and YOUTUBE [27] spam classification datasets as easy setups, having two highly separable classes. Here, each sample represents a text—either obtained from text messages or from comment posted in the comments section of youtube videos—manually labeled as spam or legitimate (ham). Although available, the metadata information—such as the author's name and publication date—is not used. As a slightly more complex setup, we consider the TREC [52] dataset, containing 4,965 labeled questions. In this context, each sample represents a question belonging to one of six classes—i.e., *Abbreviation, Entity, Description, Human, Location, Numeric-value*—to be semantically classified. Finally, as a complex setup we select the MFTC datasets as the target classification task. The MFTC dataset is composed of 35,108 tweets—sentences—, which can be considered as a collection of different datasets. Each split of MFTC corresponds to a different context. Here, tweets corresponding to the dataset samples are collected following a certain event or target. As an example, tweets belonging to the Black Lives Matter (BLM) split were collected during the period of Black Lives Matter protests in the US. The list of all MFTC subjects considered in our experiments is the following: (i) All Lives Matter (ALM), (ii) Black Lives Matter (BLM), (iii) Baltimore protests (BLT), (iv) 2016 presidential election (ELE), (v) MeToo movement (MT), (vi) hurricane Sandy (SND). Each tweet in MFTC is labelled, following the same moral theory, with one or more of the following 11 moral values: (i) care/harm, (ii) fairness/cheating, (iii) loyalty/betrayal, (iv) authority/subversion, (v) purity/degradation, (vi) non-moral. Ten of the 11 available moral values are obtained as a moral concept and its opposite expression—e.g., fairness refers to the act of supporting fairness and equality, while cheating refers to the act of refraining from exploiting others. Given morality subjectivity, each tweet is labelled by multiple annotators, and the final moral labels are obtained via majority voting.

As the size of each dataset represents a relevant component to take into account, Table 2 reports the number of sentences belonging to each dataset. Throughout our experiments we use 70% of the samples belonging to the dataset as the training set, in which LLMs are

[4] https://github.com/AndAgio/SKE_NLP.

Table 3 BERT performance over considered datasets.

	SMS	YOUTUBE	TREC	ALM	BLM	BLT	ELE	MT	SND
F_1 score	98.71%	95.81%	97.18%	63.04%	82.59%	64.51%	63.14%	52.16%	56.85%

trained, and both local and global explanations are fitted. The remaining 30% of samples is kept for testing the LLM performance as well as the quality of both local and global explanations.

4.2 Model training

The SMS, YOUTUBE, and TREC datasets represent standard multi-class single-label classification tasks. Therefore, we tackle the classification task over those datasets using a standard cross entropy loss [53]. Meanwhile, tackling MFTC we follow state-of-the-art approaches for dealing with morality classification task [54, 55]. Thus, we treat the morality classification problem as a multi-class multi-label classification task, using a *binary* cross entropy loss [53]. Differently from recent approaches, we here do not rely on the *sequential training* paradigm for the MFTC datasets, but rather train each model solely on the MFTC split at hand. Indeed, in our experiments, we do not aim at obtaining strong transferability between domains, but rather we focus on analysing LPEs behaviour.

For all datasets we leverage BERT as the LLM to be optimised [11], and define one NN model for each dataset, optimising its parameters over the 70% of samples, leaving the remaining 30% for testing purposes. We leverage the pre-trained *bert-base-uncased* model—available in the Hugging Face python library[5]—as the starting point of our training process. Each model is trained using the standard Stochastic Gradient Descent (SGD) optimization procedure for 3 epochs, a learning rate of 5×10^{-5}, a batch size of 16 and a maximum sequence length of 64. We keep track of the macro F1-score for each model to identify its performance over the test samples. Table 3 shows the performance of the trained BERT model.

4.3 Local post-hoc explainers comparison

We analyse the extent to which different LPEs are aligned in their process of identifying impactful concepts for the underlying NN model. With this aim, we train a BERT model over a specific dataset (following the approach described in Sect. 4.2) and compute the pairwise correlation $\mathcal{C}\big(\epsilon_{NN}(\mathcal{S}), \epsilon'_{NN}(\mathcal{S})\big)$ (as described in Sect. 3) for each pair of LPEs in the selected set. To avoid issues caused by model overfitting over the training set, which would render explanations unreliable, we apply each ϵ_{NN} over the test set of the selected dataset.

[5] https://github.com/huggingface.

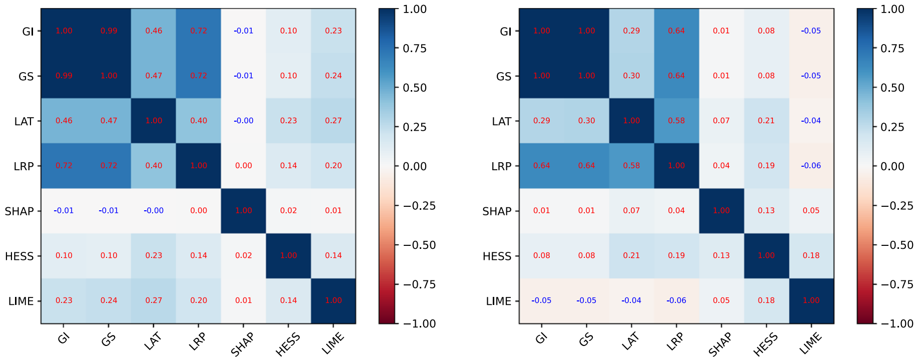


Fig. 3 $\mathcal{C}\left(\epsilon_{NN}(\mathcal{S}), \epsilon'_{NN}(\mathcal{S})\right)$ using average aggregation as $\mathcal{A}$ over the SMS (left) and YOUTUBE (right) dataset.

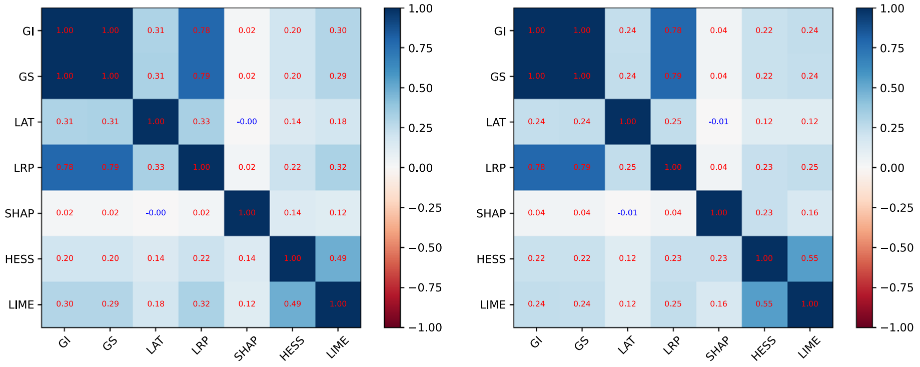
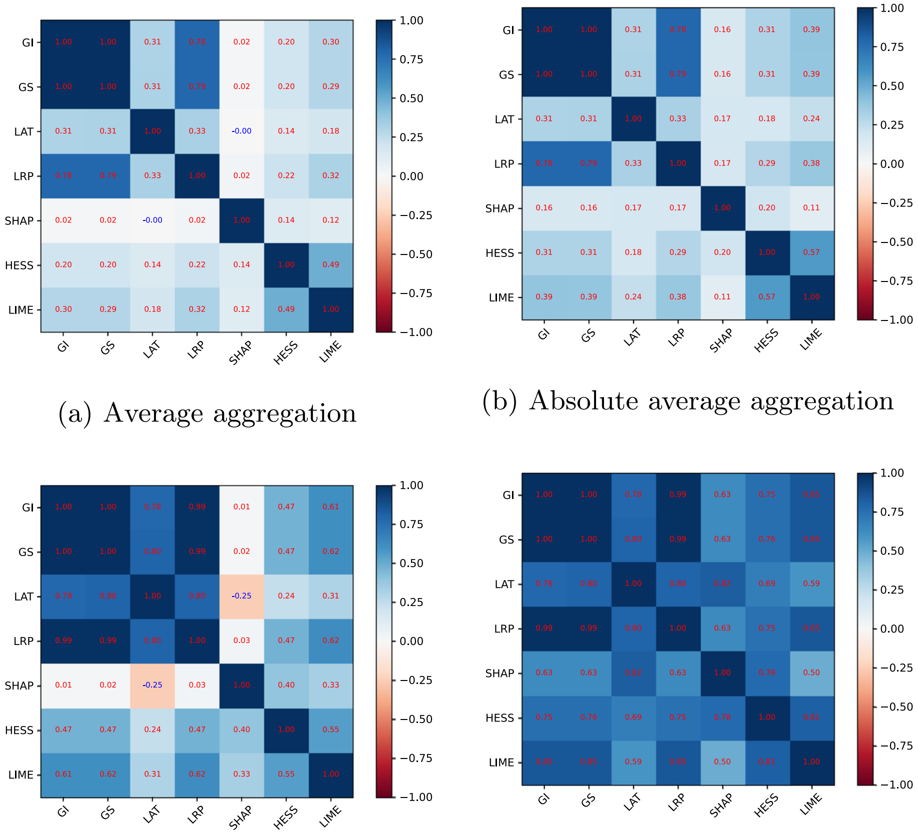


Fig. 4 $\mathcal{C}\left(\epsilon_{NN}(\mathcal{S}), \epsilon'_{NN}(\mathcal{S})\right)$ using average aggregation as $\mathcal{A}$ over the ALM (left) and BLM (right) dataset.

4.3.1 Local post-hoc explainers disagreement

Using the pairwise correlation values we construct the correlation matrices shown in Figs. 3 and 4, which highlight how there exist a very weak correlation score between most LPEs over different datasets. Here, it is interesting to notice how few specific couples or clusters of LPEs exist which highly correlate with each other. For example, GS, GI, and LRP show moderate-to-high correlation score, mainly due to their reliance on computing the gradient of the prediction to identify impactful concepts. However, this is not the case for all LPE couples relying on similar approaches. For example, GI and gradient integration—HESS in the matrices—show little to no correlation, although they both are gradient-based approach for producing local explanations. Similarly, SHAP and LIME show no correlation even if they both rely on input perturbation and are considered the state-of-the-art.

Figures 3 and 4 highlight how the vast majority of LPE pairs show very-small-to-no correlation at all, exposing how the selected approaches actually disagree. Interestingly enough, disagreement between LPEs holds true for every dataset studied in our analyses, no matter the complexity or simplicity of the learning task and the samples considered.



(a) Average aggregation

(b) Absolute average aggregation

(c) Sum aggregation

(d) Absolute sum aggregation

Fig. 5 $\mathcal{C}\big(\epsilon_{NN}(\mathcal{S}), \epsilon'_{NN}(\mathcal{S})\big)$ using different aggregations over the ALM dataset.

This finding represents a fundamental result of our study, as it demonstrates how no accordance exists between LPEs even when they are applied to the same model and dataset, even on very simple classification tasks such as the one represented by the SMS dataset. The reason behind the large discrepancies among LPE might be various, but mostly bear down to the following:

- Few of the LPEs considered in the literature do not represent reliable solutions for identifying the reasoning principles of LLMs.
- Each of the uncorrelated LPEs highlights a different set or subset of reasoning principles of the underlying model.

Therefore, our results show how complex it is to identify a set of fair and reliable metrics to spot the best LPE or even reliable LPEs, as they seem to gather uncorrelated explanations. Similar results to the ones shown in Figs. 3 and 4 are obtained for all datasets and are made available at https://github.com/AndAgio/GELPE.

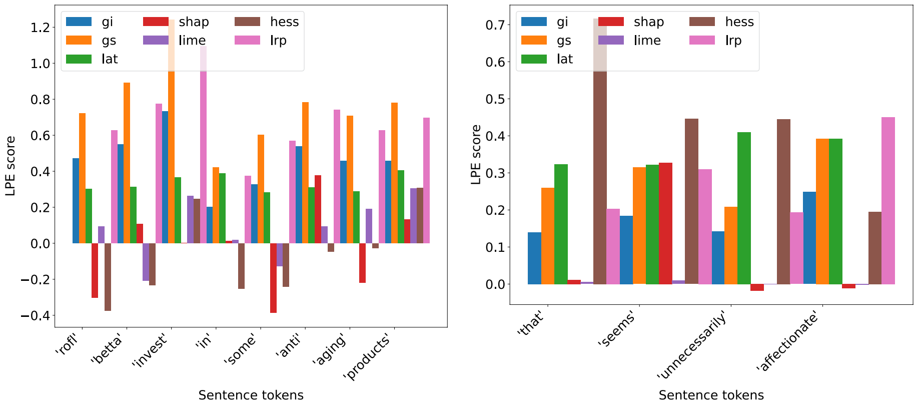


Fig. 6 Example of LPEs influence scores over the sentence achieving the lowest (left) and highest (right) correlation of LPEs in the SMS dataset.

4.3.2 Aggregation affects correlation

Since our LPE correlation metric is dependent on $\mathcal{A}$, we here analyse how the selection of different aggregation strategies impacts the correlation between LPEs. To understand the impact of $\mathcal{A}$ on $\mathcal{C}$, we plot the correlation matrices for a single dataset, varying the aggregation approach, thus obtaining the four correlation matrices shown in Fig. 5.

From Figureds 5c, d one could notice the strong correlation between different LPEs. This seems to be in contrast with the results found in Sect. 4.3. However, the reason behind the strong correlation achieved when relying on summation aggregation is not caused by the actual correlation between explanations, but rather on the susceptibility of summation to tokens frequency. Indeed, since the summation aggregation approaches do not take into account the occurrence frequency of lemmas in $\mathcal{S}$, they tend to overestimate the relevance of popular concepts. Intuitively, using this aggregations, a rather impactless lemma appearing 5000 times would obtain a global impact higher than a very impactful lemma appearing only 10 times. These results highlight the importance of relying on average based aggregation approaches when considering to construct global explanations from the LPE outputs.

Figure 5 also points out how leveraging the absolute value of LPEs incurs in higher correlation scores. The reason behind this is to be found in the impact scores distributions. While true local impact scores are distributed over the set of real numbers $\mathbb{R}$, computing the absolute value of local impacts j shifts their distribution to $\mathbb{R}^+$, shrinking possible differences between positive and negative scores. Moreover, LPE outputs rely much more heavily on scoring positive contributions using positive impact scores, and typically give less focus to negative impact scores. Therefore, the output of LPEs is generally unbalanced towards positive impact scores, making negative impact scores mostly negligible.

4.3.3 LPEs visualization examples

The results obtained over various LPEs when considering several input sentences identify a large discrepancy between the available LPE approaches. To better visualize the quarrel between LPEs, we here consider to visualize the output of LPE explanations over few of the sentences belonging to the considered datasets in Figs. 6 and reffig:lpespsalmspssingle

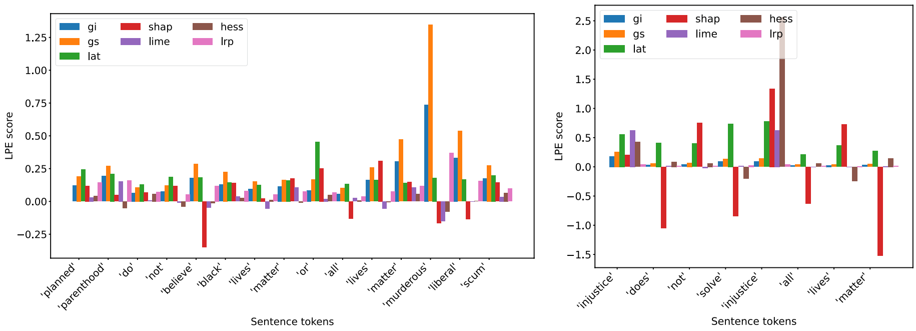


Fig. 7 Example of LPEs influence scores over the sentence achieving the lowest (left) and highest (right) correlation of LPEs in the ALM dataset.

spssentence. More in detail, we plot the LPEs relevance scores for each token over the sentences of the dataset used in our experiments. Generally speaking, higher scores identify the most relevant tokens for the LLM prediction, while low scores identify non relevant tokens. Negative scores are assigned to the tokens that negatively influence the prediction for a specific class, thus identifying the tokens that should drift the prediction towards a different class.

Figure 6 shows the LPEs scores over two sentences of the SMS dataset. The left-side plot is obtained for a sentence where LPEs are far from being correlated, thus highlighting the quarrel between LPEs and confirming the findings of Sect. 4.3.1. The difference in LPEs influence scores is evident across most tokens, with each LPE considering as the most relevant tokens several different candidates—e.g., SHAP focuses on *anti*, GS focuses on *invest*, LRP focuses on *in*, etc. On the other hand, the right-side plot is obtained for a sentence where LPEs are slightly correlated, thus showing somewhat an agreement between most LPEs. However, even considering sentences where LPEs generally agree, it is possible to notice how few approaches are far from being perfectly adherent to the majority of LPEs. For example, SHAP and LIME assign an almost zero influence score to all tokens, while other LPEs tend to produce non-negligible scores. Similar results are obtained for the ALM dataset and shown in Fig. 7. However, for the ALM dataset, the disagreement among LPEs is evident even when selecting the sentence achieving the highest LPEs correlation (right-side plot). Similar results to the ones shown in Figs. 6 and 7 are obtained for all sentences in each dataset considered, and made available at https://github.com/AndAgio/GELPE.

4.4 Knowledge extraction

We here analyse if and to what extent it is possible to extract a knowledge base representing the trained LLM from each LPE, and how much these are aligned in their process of explaining the underlying NN model. With this aim, we rely on the GELPE global explainer construction process presented in Sect. 3.5, extracting a set of rules representing the LLM decision process for each dataset at hand. As the building process is dependent on the number of most impactful lemmas, we consider varying the hyperparameter $\mathcal{K}$ to select the top-$\mathcal{K}$ relevant lemmas for each class. After the relevant lemmas are selected from a

Table 4 Fidelity of the extracted knowledge w.r.t. to the original BERT model over the SMS dataset.

LPEs	$\mathcal{K}$				
	50 (%)	100 (%)	150 (%)	200 (%)	250 (%)
GI	**87.00**	**87.60**	**90.20**†	**91.80**†	**91.60**†
GS	**87.40**†	**87.80**†	**89.80**	**90.40**	**91.60**†
LAT	87.40†	87.40	89.00	89.60	91.00
LRP	86.60	86.40	86.60	87.80	90.80
SHAP	86.40	86.60	86.60	86.40	86.40
HESS	86.20	86.40	86.80	86.80	86.40
LIME	86.20	86.20	86.20	86.60	86.80

†Identifies the best LPE over a single $\mathcal{K}$ value, while the bold row(s) identify the overall best LPE

Table 5 Fidelity of the extracted knowledge w.r.t. to the original BERT model over the YOUTUBE dataset.

LPEs	$\mathcal{K}$				
	50 (%)	100 (%)	150 (%)	200 (%)	250 (%)
GI	69.20	72.40	72.40	76.00	76.80
GS	69.20	72.40	72.40	78.00	76.40
LAT	66.00	64.40	70.80	80.00	84.40
LRP	65.20	65.20	68.80	70.00	70.00
SHAP	43.20	75.20	80.80	80.80	80.40
HESS	82.40	87.60	86.40	88.80	87.20
LIME	**88.00**†	**92.00**†	**94.00**†	**93.20**†	**92.80**†

†Identifies the best LPE over a single $\mathcal{K}$ value, while the bold row(s) identify the overall best LPE

given LPE, we construct the skipgrams of relevant lemmas as the set of skipgrams occurring in the training set that are composed from relevant lemmas only. Skipgrams are considered to extend the capabilities of the extraction process to consider sequences of relevant concepts rather than blindly focusing only on single tokens. Once the relevant lemmas and skipgrams are available, we consider converting the samples of the training set into binary vectors describing the presence or absence of each lemma and skipgram. We optimise the CART model on the binary vectors representing the training samples and extract the corresponding knowledge from the tree as a set of ordered propositional rules. The extracted rules are sequential, meaning that one rule applies if and only if the previous rules were not successful in identifying the relevant prediction. To avoid incurring in an unbearable number of propositional clauses—that would hinder the utility of the knowledge extraction process—we limit the depth of the CART model to be:

$$depth = \mu \cdot \frac{\Lambda}{\mathcal{K} * |\mathcal{Y}|}, \tag{18}$$

where Λ represents the number of total relevant lemmas and skipgrams identified from the LPE, $|\mathcal{Y}|$ represents the number of classes of the classification task at hand, and μ represents an hyperparameter that we set to $\mu = 5$ empirically. Throughout the remainder of

Table 6 Fidelity of the extracted knowledge w.r.t. to the original BERT model over the BLT dataset.

LPEs	$\mathcal{K}$				
	50 (%)	100 (%)	150 (%)	200 (%)	250 (%)
GI	92.64	92.70	93.18	93.12	92.76
GS	93.60	92.28	93.18	93.24	92.82
LAT	90.19	91.74	92.28	92.34	92.46
LRP	92.28	93.12	93.00	93.48	92.88
SHAP	95.69	94.14	94.14	94.14	94.14
HESS	93.72	93.84	93.48	93.48	93.60
LIME	**95.27**†	**95.27**†	**95.09**†	**95.09**†	**95.09**†

†Identifies the best LPE over a single $\mathcal{K}$ value, while the bold row(s) identify the overall best LPE

Table 7 Fidelity of the extracted knowledge w.r.t. to the original BERT model over the ELE dataset.

LPEs	$\mathcal{K}$				
	50 (%)	100 (%)	150 (%)	200 (%)	250 (%)
GI	68.51	72.00	74.67	75.92	76.23
GS	**68.95**	**73.93**	**74.74**†	**74.67**	**76.79**†
LAT	58.93	61.36	66.77	67.45	69.14
LRP	72.81	74.74†	75.36	75.48	75.79
SHAP	67.64	68.70	69.51	70.50	70.50
HESS	68.33	73.80	74.05	74.11	74.11
LIME	**73.61**†	**73.93**	**74.49**	**76.60**†	**76.73**

† identifies the best LPE over a single $\mathcal{K}$ value, while the bold row(s) identify the overall best LPE

Table 8 Fidelity of the extracted knowledge w.r.t. to the original BERT model over the SND dataset.

LPEs	$\mathcal{K}$				
	50 (%)	100 (%)	150 (%)	200 (%)	250 (%)
GI	45.39	58.39	60.20	59.84	61.73
GS	**46.48**	**57.59**	**59.84**	**61.00**†	**62.60**†
LAT	38.63	45.53	49.82	49.89	57.30
LRP	40.02	49.67	59.98	59.62	61.00
SHAP	57.01	57.23	57.23	57.23	57.23
HESS	60.49	60.13†	58.75	58.53	58.61
LIME	**61.15**†	**60.06**	**60.28**†	**60.20**	**60.13**

†Identifies the best LPE over a single $\mathcal{K}$ value, while the bold row(s) identify the overall best LPE

this paper, we consider leveraging the average operation as the aggregation function $\mathcal{A}$, as it represents the least biased aggregation process. However, we also experiment with other aggregation functions, such as sum, absolute sum, and absolute average, obtaining similar results. Therefore, in order to avoid redundancy we here show only the average aggregation results.

4.4.1 Knowledge fidelity

To asses the performance of the proposed knowledge extraction process from LPEs, we measure the fidelity of the predictions obtained using the propositional rules against the corresponding LLM predictions. The fidelity metric measures the percentage of instances in which the propositional rules predictions and model predictions are equivalent, thus measuring the accuracy of the knowledge extraction process. Since, GELPE relies on the output of a single LPE mechanism to produce the logic program equivalent to the LLM at hand, we compare the fidelity performance of GELPE over all the LPEs presented in Sect. 3.2. Tables 4 and 5 present the fidelity of the GELPE extraction process over the SMS and YOUTUBE datasets. In those simple scenarios, the proposed approach extracts a set of accurate rules, representing with high fidelity the decision process of the underlying LLM. Using GELPE, we enable the extraction of simple and easy to understand rules from the complex black-box model.

Over more complex datasets, the performance of the extracted knowledge using GELPE varies depending on the dataset at hand. Table 6 shows the fidelity of GELPE over the BLT dataset, where the explanation model achieves up to 95.09% fidelity. Meanwhile, Tables 7 and 8 presents the fidelity results over the ELE and SND datasets respectively, where the proposed GELPE extraction seems to struggle to achieve high fidelity values. This is due to the underlying complexity of the dataset at hand. For some tasks—e.g. YOUTUBE, BLT—, considering the most relevant lemmas and their skipgram combinations is sufficient, while others—e.g. ELE, SND—require a more complex understanding of the inner sentence constructs.

As expected, increasing the number of relevant lemmas $\mathcal{K}$ considered to optimise GELPE results in higher fidelity, as the underlying CART model takes into account a broader set of meaningful features. However, increasing $\mathcal{K}$ over a certain threshold results in an unbearable rules complexity and in smaller fidelity gains. The increment on rule complexity also hiders the understandability of the extracted explanation, representing a fundamental concept to take into account. This phenomenon is clearly shown in Tables 7 and 8, where the fidelity grows up to 20% when $\mathcal{K}$ ranges from 50 to 250.

Interestingly, the disagreement between different LPEs seems to affect also the performance of the obtained global explainer model. Fidelity results highlight that GELPE explanations obtained from highly correlated LPEs such as GI and GS achieve comparable performance level. Meanwhile, propositional rules obtained from uncorrelated LPEs result in different fidelity level. While expected, such a behaviour represents a useful finding as it allows for the identification of more reliable LPEs, as the ones that results in a higher level of fidelity—e.g., LIME in most scenarios.

4.4.2 Knowledge complexity

The ideal extraction process is required to output a set of sequential propositional rules that is as faithful as possible w.r.t. the underlying LLM. However, the dimensionality of the extracted program should be kept small to limit the complexity burden of the analysis process. An overly complex knowledge base would not be useful for analysing the

Table 9 Complexity of the extracted knowledge over the YOUTUBE dataset.

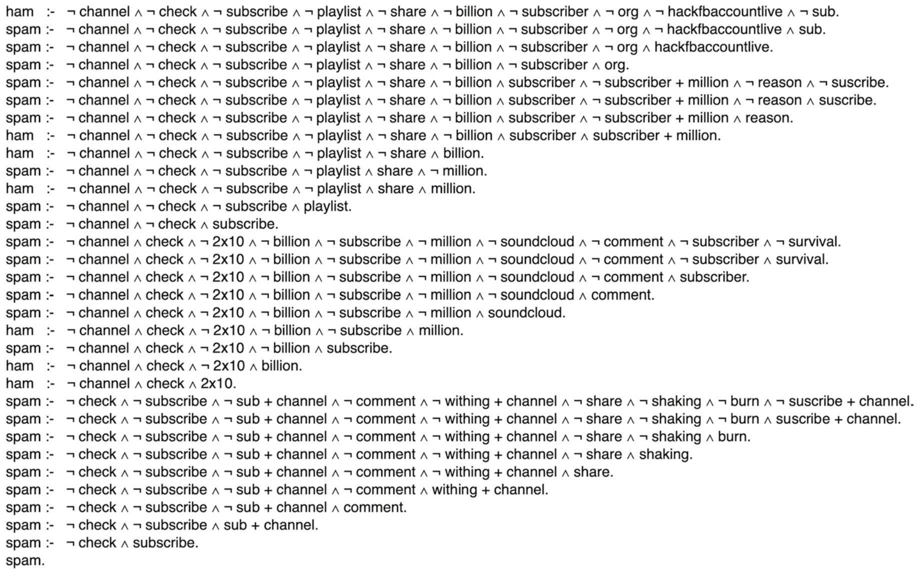
LPEs	$\mathcal{K}$				
	50	100	150	200	250
GI	$L = 30$	$L = 41$	$L = 40$	$L = 40$	$L = 66$
	$C = 6.73$	$C = 7.71$	$C = 7.65$	$C = 7.05$	$C = 10.38$
GS	$L = 30$	$L = 41$	$L = 42$	$L = 37$	$L = 73$
	$C = 6.63$	$C = 7.71$	$C = 7.79$	$C = 7.51$	$C = 10.59$
LAT	$L = 20$	$L = 43$	$L = 116$	$L = 75$	$L = 64$
	$C = 5.65$	$C = 6.42$	$C = 10.28$	$C = 8.84$	$C = 11.23$
LRP	$L = 37$	$L = 46$	$L = 36$	$L = 32$	$L = 48$
	$C = 6.51$	$C = 7.15$	$C = 7.17$	$C = 7.09$	$C = 7.81$
SHAP	**L = 14**	**L = 25**	**L = 34**	**L = 36**	**L = 33**
	C = 5.21	**C = 7.04**	**C = 7.62**	**C = 7.78**	**C = 7.61**
HESS	$L = 48$	$L = 53$	$L = 55$	$L = 39$	$L = 52$
	$C = 9.67$	$C = 10.36$	$C = 12.05$	$C = 11.67$	$C = 11.88$
LIME	$L = 32$	$L = 56$	$L = 57$	$L = 60$	$L = 68$
	$C = 6.72$	$C = 9.29$	$C = 9.42$	$C = 11.30$	$C = 13.01$

L represents the length of the obtained explanation—i.e., the number of clauses—, while *C* represents the cumbersomeness—i.e., the average number of atoms in each clause. The bold row(s) identify the overall simplest LPE

Table 10 Complexity of the extracted knowledge over the ELE dataset.

$\mathcal{K}$	LPEs				
	50	100	150	200	250
GI	$L = 430$	$L = 363$	$L = 353$	$L = 273$	$L = 296$
	$C = 19.78$	$C = 19.68$	$C = 18.21$	$C = 15.76$	$C = 15.29$
GS	$L = 422$	$L = 335$	$L = 350$	$L = 269$	$L = 369$
	$C = 19.58$	$C = 19.54$	$C = 18.27$	$C = 15.61$	$C = 17.94$
LAT	$L = 639$	$L = 487$	$L = 373$	$L = 379$	$L = 360$
	$C = 20.52$	$C = 19.93$	$C = 19.11$	$C = 20.00$	$C = 20.47$
LRP	$L = 390$	$L = 433$	$L = 391$	$L = 375$	$L = 364$
	$C = 19.85$	$C = 22.08$	$C = 18.45$	$C = 18.19$	$C = 17.93$
SHAP	**L = 16**	**L = 15**	**L = 16**	**L = 16**	**L = 16**
	C = 4.06	**C = 3.93**	**C = 4.06**	**C = 4.06**	**C = 4.06**
HESS	$L = 17$	$L = 64$	$L = 71$	$L = 71$	$L = 72$
	$C = 4.12$	$C = 7.84$	$C = 8.04$	$C = 8.03$	$C = 8.08$
LIME	$L = 64$	$L = 68$	$L = 71$	$L = 130$	$L = 131$
	$C = 7.75$	$C = 7.94$	$C = 8.06$	$C = 10.72$	$C = 10.76$

L represents the length of the obtained explanation—i.e., the number of clauses—, while *C* represents the cumbersomeness—i.e., the average number of atoms in each clause. The bold row(s) identify the overall simplest LPE



```
ham  :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ ¬ share ∧ ¬ billion ∧ ¬ subscriber ∧ ¬ org ∧ ¬ hackfbaccountlive ∧ ¬ sub.
spam :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ ¬ share ∧ ¬ billion ∧ ¬ subscriber ∧ ¬ org ∧ ¬ hackfbaccountlive ∧ sub.
spam :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ ¬ share ∧ ¬ billion ∧ ¬ subscriber ∧ ¬ org ∧ hackfbaccountlive.
spam :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ ¬ share ∧ ¬ billion ∧ ¬ subscriber ∧ org.
spam :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ ¬ share ∧ ¬ billion ∧ subscriber ∧ ¬ subscriber + million ∧ ¬ reason ∧ ¬ suscribe.
spam :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ ¬ share ∧ ¬ billion ∧ subscriber ∧ ¬ subscriber + million ∧ ¬ reason ∧ suscribe.
spam :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ ¬ share ∧ ¬ billion ∧ subscriber ∧ ¬ subscriber + million ∧ reason.
ham  :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ ¬ share ∧ ¬ billion ∧ subscriber ∧ subscriber + million.
ham  :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ ¬ share ∧ billion.
spam :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ share ∧ ¬ million.
ham  :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ ¬ playlist ∧ share ∧ million.
spam :-  ¬ channel ∧ ¬ check ∧ ¬ subscribe ∧ playlist.
spam :-  ¬ channel ∧ ¬ check ∧ subscribe.
spam :-  ¬ channel ∧ check ∧ ¬ 2x10 ∧ ¬ billion ∧ ¬ subscribe ∧ ¬ million ∧ ¬ soundcloud ∧ ¬ comment ∧ ¬ subscriber ∧ ¬ survival.
spam :-  ¬ channel ∧ check ∧ ¬ 2x10 ∧ ¬ billion ∧ ¬ subscribe ∧ ¬ million ∧ ¬ soundcloud ∧ ¬ comment ∧ ¬ subscriber ∧ survival.
spam :-  ¬ channel ∧ check ∧ ¬ 2x10 ∧ ¬ billion ∧ ¬ subscribe ∧ ¬ million ∧ ¬ soundcloud ∧ ¬ comment ∧ subscriber.
spam :-  ¬ channel ∧ check ∧ ¬ 2x10 ∧ ¬ billion ∧ ¬ subscribe ∧ ¬ million ∧ ¬ soundcloud ∧ comment.
spam :-  ¬ channel ∧ check ∧ ¬ 2x10 ∧ ¬ billion ∧ ¬ subscribe ∧ ¬ million ∧ soundcloud.
ham  :-  ¬ channel ∧ check ∧ ¬ 2x10 ∧ ¬ billion ∧ ¬ subscribe ∧ million.
spam :-  ¬ channel ∧ check ∧ ¬ 2x10 ∧ ¬ billion ∧ subscribe.
ham  :-  ¬ channel ∧ check ∧ ¬ 2x10 ∧ billion.
ham  :-  ¬ channel ∧ check ∧ 2x10.
spam :-  ¬ check ∧ ¬ subscribe ∧ ¬ sub + channel ∧ ¬ comment ∧ ¬ withing + channel ∧ ¬ share ∧ ¬ shaking ∧ ¬ burn ∧ ¬ suscribe + channel.
spam :-  ¬ check ∧ ¬ subscribe ∧ ¬ sub + channel ∧ ¬ comment ∧ ¬ withing + channel ∧ ¬ share ∧ ¬ shaking ∧ ¬ burn ∧ suscribe + channel.
spam :-  ¬ check ∧ ¬ subscribe ∧ ¬ sub + channel ∧ ¬ comment ∧ ¬ withing + channel ∧ ¬ share ∧ ¬ shaking ∧ burn.
spam :-  ¬ check ∧ ¬ subscribe ∧ ¬ sub + channel ∧ ¬ comment ∧ ¬ withing + channel ∧ ¬ share ∧ shaking.
spam :-  ¬ check ∧ ¬ subscribe ∧ ¬ sub + channel ∧ ¬ comment ∧ ¬ withing + channel ∧ share.
spam :-  ¬ check ∧ ¬ subscribe ∧ ¬ sub + channel ∧ ¬ comment ∧ withing + channel.
spam :-  ¬ check ∧ ¬ subscribe ∧ ¬ sub + channel ∧ comment.
spam :-  ¬ check ∧ ¬ subscribe ∧ sub + channel.
spam :-  ¬ check ∧ subscribe.
spam.
```

Fig. 8 Logic program $\mathcal{P}$ obtained from the GELPE extraction process when leveraging LIME as LPE and $\mathcal{K} = 50$ on the YOUTUBE dataset.

inner working principle of the explained LLM, as it would be mostly impossible to be processed by a human interpreter. To assess the complexity of the extracted knowledge, we consider tracking the length of the logic program and its cumbersomeness. In this context, the length L represents the number of clauses in the obtained explanation, while the cumbersomeness C represents the average number of atoms in each clause. L and C represent two fundamental parameters for describing the complexity of the extracted logic program. Lengthier programs are more complex to read and may result in the reader getting lost. On the other hand, a higher cumbersomeness translates directly into longer rules, which are by default more complex to understand, as human users are generally more susceptible to complex multi-variable reasoning. Moreover, longer rules are generally more specific, as they require linking multiple input variables—and possibly their interactions—to a specific output label. Therefore, when long rules are extracted it possibly means that the LLM signals a specific behavior over a specific input. This phenomenon can translate directly into the identification of bias issues, overfitting problems and much more.

For each dataset considered we keep track of L and C and analyse their variability over each LPE and $\mathcal{K}$ value. Tables 9 and 10 show the complexity of the GELPE output over the YOUTUBE and ELE dataset respectively. The results highlight the relevant difference in terms of required complexity to extract reliable explanations when dealing with simple or complex classification tasks. Both L and C are kept small for each LPE and $\mathcal{K}$ combination over the YOUTUBE dataset, while still being able to reach high fidelity (see Table 5). Meanwhile, the ELE moral classification task requires to consider higher values of L and C in order to achieve a satisfactory level of fidelity (see Table 7).

Table 9 also highlights a dependency between the complexity of the extracted explanations and the parameter $\mathcal{K}$. In the vast majority of cases, the higher $\mathcal{K}$ produces a more complex global explanation program, usually characterized by a higher number of clauses L and a larger number of atoms for each clause C. This is expected, since a higher value

```
non-moral   :-  ¬ solidarity ∧ ¬ justice ∧ ¬ obey ∧ ¬ injustice ∧ ¬ compassion.
care        :-  ¬ solidarity ∧ ¬ justice ∧ ¬ obey ∧ ¬ injustice ∧ compassion.
cheating    :-  ¬ solidarity ∧ ¬ justice ∧ ¬ obey ∧ injustice ∧ ¬ compassion.
care        :-  ¬ solidarity ∧ ¬ justice ∧ ¬ obey ∧ injustice ∧ compassion.
authority   :-  ¬ solidarity ∧ ¬ justice ∧ obey ∧ ¬ rape ∧ ¬ corrupt.
cheating    :-  ¬ solidarity ∧ ¬ justice ∧ obey ∧ ¬ rape ∧ corrupt.
harm        :-  ¬ solidarity ∧ ¬ justice ∧ obey ∧ rape.
fairness    :-  ¬ solidarity ∧ justice ∧ ¬ injustice ∧ ¬ disobedience ∧ ¬ murder.
harm        :-  ¬ solidarity ∧ justice ∧ ¬ injustice ∧ ¬ disobedience ∧ murder.
subversion  :-  ¬ solidarity ∧ justice ∧ ¬ injustice ∧ disobedience.
cheating    :-  ¬ solidarity ∧ justice ∧ injustice ∧ ¬ standing + injustice ∧ ¬ racist + injustice.
cheating    :-  ¬ solidarity ∧ justice ∧ injustice ∧ ¬ standing + injustice ∧ racist + injustice.
fairness    :-  ¬ solidarity ∧ justice ∧ injustice ∧ standing + injustice.
loyalty     :-  ¬ injustice ∧ ¬ disobedience ∧ ¬ pseudoscience ∧ ¬ justice.
loyalty     :-  ¬ injustice ∧ ¬ disobedience ∧ ¬ pseudoscience ∧ justice.
harm        :-  ¬ injustice ∧ ¬ disobedience ∧ pseudoscience.
subversion  :-  ¬ injustice ∧ disobedience.
cheating    :-  ¬ solidarity + injustice.
loyalty.
```

Fig. 9 Logic program $\mathcal{P}$ obtained from the GELPE extraction process when leveraging SHAP as LPE and $\mathcal{K} = 100$ on the BLM dataset.

of $\mathcal{K}$ identifies a broader set of relevant lemmas considered during the optimization of the CART explainer, thus increasing the number of features available to construct propositional clauses. However, it is interesting to notice how the almost-linear dependency on $\mathcal{K}$ affects more C than L, since L can be bounded during the CART optimization process via pruning. The increased complexity of the obtained explanation represents a fundamental aspect to take into account when considering leveraging GELPE, as we need for the explanations to be bounded in complexity for them to be human-readable. The limitation of the CART depth—see Eq. 18—represents an helping tool from this perspective, as it allows to keep the complexity of the explainer under control in complex setup, such as the ELE dataset. This phenomenon can be seen in Table 10, where the complexity of the extracted explanations remains stable over $\mathcal{K}$. However, depth limitation is not drawback free, as it hinders the achievement of high fidelity values.

4.4.3 Knowledge visualisation

We visualise the logic programs obtained from the knowledge extraction process to analyse their correctness and understandability. Figure 8 shows the logic program $\mathcal{P}$ obtained from the GELPE extraction process when leveraging LIME as LPE and $\mathcal{K} = 50$ on the YOUTUBE dataset. The extracted knowledge is characterised by a manageable complexity, having a small number of relatively short clauses. In this context, the summation symbol + is used to identify the concatenation of two relevant lemmas inside a sentence—*lemma1* + *lemma2* can be interpreted as *lemma1 followed by lemma2*. Moreover, we remind that the extracted rules are sequential, meaning that each propositional rule applies if an only if the previous rules did not. For example, in Fig. 8 the last rule, specifing that the message is spam, is valid only if all the previous 31 rules did not match a class output. Interestingly, the extracted knowledge also shows some relevant properties, such as the identification of spam comments as those containing certain

Table 11 Resource efficiency comparison of BERT against GELPE for each dataset.

Dataset Model	SMS	YOUTUBE	TREC	ALM	BLM	BLT	ELE	MT	SND
BERT$_{GPU}$	$\bar{t} = 0.017s$ $\overline{E} = 2.841J$	$\bar{t} = 0.009s$ $\overline{E} = 2.350J$	$\bar{t} = 0.008s$ $\overline{E} = 0.987J$	$\bar{t} = 0.006s$ $\overline{E} = 1.181J$	$\bar{t} = 0.006s$ $\overline{E} = 1.209J$	$\bar{t} = 0.006s$ $\overline{E} = 1.481J$	$\bar{t} = 0.006s$ $\overline{E} = 1.196J$	$\bar{t} = 0.007s$ $\overline{E} = 1.961J$	$\bar{t} = 0.006s$ $\overline{E} = 1.148J$
BERT$_{CPU}$	$\bar{t} = 0.047s$ $\overline{E} = 5.008J$	$\bar{t} = 0.066s$ $\overline{E} = 7.893J$	$\bar{t} = 0.023s$ $\overline{E} = 2.421J$	$\bar{t} = 0.027s$ $\overline{E} = 2.940J$	$\bar{t} = 0.028s$ $\overline{E} = 3.037J$	$\bar{t} = 0.029s$ $\overline{E} = 3.141J$	$\bar{t} = 0.026s$ $\overline{E} = 2.906J$	$\bar{t} = 0.049s$ $\overline{E} = 5.576J$	$\bar{t} = 0.026s$ $\overline{E} = 2.719J$
SHAP$_{50}$	$\bar{t} = 0.009s$ $\overline{E} = 0.574J$	$\bar{t} = 0.004s$ $\overline{E} = 0.223J$	$\bar{t} = 0.004s$ $\overline{E} = 0.269J$	$\bar{t} = 0.008s$ $\overline{E} = 0.492J$	$\bar{t} = 0.011s$ $\overline{E} = 0.595J$	$\bar{t} = 0.005s$ $\overline{E} = 0.307J$	$\bar{t} = 0.006s$ $\overline{E} = 0.383J$	$\bar{t} = 0.008s$ $\overline{E} = 0.456J$	$\bar{t} = 0.008s$ $\overline{E} = 0.490J$
LIME$_{50}$	$\bar{t} = 0.004s$ $\overline{E} = 0.208J$	$\bar{t} = 0.004s$ $\overline{E} = 0.260J$	$\bar{t} = 0.010s$ $\overline{E} = 0.592J$	$\bar{t} = 0.021s$ $\overline{E} = 1.189J$	$\bar{t} = 0.026s$ $\overline{E} = 1.455J$	$\bar{t} = 0.005s$ $\overline{E} = 0.283J$	$\bar{t} = 0.015s$ $\overline{E} = 0.891J$	$\bar{t} = 0.020s$ $\overline{E} = 1.121J$	$\bar{t} = 0.013s$ $\overline{E} = 0.777J$
SHAP$_{250}$	$\bar{t} = 0.010s$ $\overline{E} = 0.556J$	$\bar{t} = 0.012s$ $\overline{E} = 0.694J$	$\bar{t} = 0.019s$ $\overline{E} = 1.115J$	$\bar{t} = 0.032s$ $\overline{E} = 1.736J$	$\bar{t} = 0.035s$ $\overline{E} = 1.968J$	$\bar{t} = 0.018s$ $\overline{E} = 1.015J$	$\bar{t} = 0.025s$ $\overline{E} = 1.403J$	$\bar{t} = 0.026s$ $\overline{E} = 1.432J$	$\bar{t} = 0.034s$ $\overline{E} = 1.859J$
LIME$_{250}$	$\bar{t} = 0.016s$ $\overline{E} = 0.866J$	$\bar{t} = 0.034s$ $\overline{E} = 1.789J$	$\bar{t} = 0.084s$ $\overline{E} = 4.696J$	$\bar{t} = 0.144s$ $\overline{E} = 7.990J$	$\bar{t} = 0.235s$ $\overline{E} = 13.047J$	$\bar{t} = 0.024s$ $\overline{E} = 1.388J$	$\bar{t} = 0.087s$ $\overline{E} = 4.808J$	$\bar{t} = 0.106s$ $\overline{E} = 5.797J$	$\bar{t} = 0.078s$ $\overline{E} = 4.364J$

For each dataset, we highlight in blue the most energy efficient model, in brown the least energy efficient one, in green the quickest model and in red the slowest one

hyperlinks (*org* lemma), subscription related lemmas (*sub* and *subscribe*), as well as grammatical errors (*suscribe* rather than subscribe and *withing* rather than within).

Figure 9 shows the extracted knowledge when GELPE is used with SHAP and $\mathcal{K} = 100$ over the BLM dataset. Here, it is also possible to notice relevant concepts being extracted from the LLM decision process. For example, the proposed extraction process allows to identify that the combination of keywords *obey* and *rape* result in the text being considered as harmful, as well as the keyword *murder*. Meanwhile, the sequence *standing* + *injustice* along with the *justice* keyword identify that the sentiment is fairness. Finally, since the extracted rules are sequential, the *loyalty* fact at the end of the program serves as the default prediction whenever none of the extracted rules applies. These results highlight the goodness of the proposed GELPE framework than enables the extraction of meaningful logic rules from the LLM reasoning principle with high fidelity.

4.4.4 Resource effeciency

The proposed GELPE framework allows for the extraction of sequential propositional rules from LLM starting from LPEs outputs. In an ideal scenario, the logic program obtained as a result of the GELPE process contains a handful of simple—i.e., short—clauses. The execution of such simple program—surrogate of the original LLM model—requires few computational power, as it does not rely on complex operations such as convolutions that require GPUs or hardware-specific solutions. However, the complexity of the GELPE output can grow quickly depending on the set of considered lemmas and skipgrams, thus hindering its efficiency. Therefore, it is fundamental to assess the ability of the proposed GELPE framework to produce a resource-friendly surrogate model of the original LLM. To this end, we consider measuring the time and energy efficiency of the original LLM model against few of the logic programs obtained using GELPE. More in detail, we consider running the original BERT model both in a GPU enabled scenario—using a Tesla V100S-PCIE with 32GB of RAM—and a CPU only scenario—using an Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz. We rely on the pyJoules[6] library for measuring the energy consumption and latency of both LLM and logic program executions. pyJoules is a software toolkit relying on (i) the Intel "Running Average Power Limit" (RAPL)[7] technology

[6] https://github.com/powerapi-ng/pyJoules.

[7] https://powerapi.org/reference/formulas/rapl/.

to estimate power consumption of the CPU, RAM and integrated GPU devices; and on (ii) the Nvidia "Nvidia Management Library"[8] technology to measure energy consumption of Nvidia GPU devices. Therefore, pyJoules represents a reliable solution to measure the energy footprint of a host machine during the execution of a piece of Python code. We consider comparing the BERT efficiency performance against the most faithful logic program—i.e., the one obtained with LIME as LPE—and against the simplest one—i.e., the one obtained with SHAP as LPE. For each LPE, we consider two setups, having the lowest and highest value of $\mathcal{K}$—i.e., $\mathcal{K} = 50$ and $\mathcal{K} = 250$, respectively. The logic programs obtained from GELPE from each LPE are run using only the CPU device. We keep track of the average time $\bar{t}$ required to infer the prediction over a single sample and the corresponding average energy consumed $\overline{E}$. Table 11 shows the obtained results over all datasets.

The obtained results highlight how over simple setups such as SMS and YOUTUBE, the surrogate model obtained using GELPE always outperforms the BERT counterpart. This is due to the small task complexity, enabling the proposed framework to extract a small set of simple clauses to mimic the model behaviour. Indeed, the efficiency of the logic program obtained is proportional to the complexity of the clauses to be analysed to achieve a prediction. Meanwhile, over more complex setups, such as the ELE dataset, in which GELPE outputs a large set of long clauses, it is possible to outperform the BERT counterpart only when considering a small value of $\mathcal{K}$. However, noticeably it is always possible to find a surrogate logic model obtained via GELPE representing a more efficient solution than running the LLM model over the CPU. These results highlight the advantage of leveraging a simple rule-based approach over sub-symbolic models when hardware acceleration is not available. As such, the proposed model represents a feasible solution for those scenarios where the deployment setup is composed of resource-constrained devices, such as embedded devices and micro-controllers. In this scenarios, running the original LLM would not be acceptable, due to latency and memory issues, while GELPE's output results in a resource efficient transparent program that is easily deployable. Therefore, the obtained results show that the GELPE surrogate model does not represent just an explainable and transparent twin of the LLM original model, but also an efficient one.

5 Discussion and limitations

Fidelity vs. efficiency trade-off The set of experiments proposed in Sect. 4 highlights how it is possible to identify a relevant logic program surrogate of the original LLM achieving high fidelity and efficiency for some scenarios. However, generally speaking there exists an intrinsic trade-off between the achievable fidelity of the surrogate logic program and its resource-efficiency improvements. Indeed, Table 11 highlights how resource efficiency gains are usually achievable whenever small logic programs are enforced using a small set of relevant lemmas—i.e., small $\mathcal{K}$ values. However, these small programs do not attain the best achievable fidelity. Consider for example the YOUTUBE dataset, where GELPE relying on LIME with $\mathcal{K} = 50$ achieves 88% fidelity, against the best fidelity of 94% achieved with $\mathcal{K} = 150$. On the other hand, logic programs extracted using large set of relevant lemmas—e.g., $\mathcal{K} = 250$—usually achieve higher fidelity, while being less effective in reducing the resource consumption. Therefore, it is possible to identify the fidelity vs. efficiency

[8] https://developer.nvidia.com/nvidia-management-library-nvml.

trade-off as one of the limitations of the proposed approach. However, while this trade-off exists, it is fundamental to note that it is relevant only whenever hardware acceleration—e.g., using GPUs—is available. Indeed, even the largest logic programs—which are expected to be the most faithful—extracted with GELPE performs similarly—from the resource-efficiency perspective—to the LLM at hand whenever it runs on CPU only (see Table 11). Moreover, the application of knowledge extraction mechanisms is generally considered in those scenarios where model opacity is a no-go. Therefore, we consider the trade-off between the achievable fidelity and the resource-efficiency to not apply in those scenarios where available hardware is limited or whenever transparency represents the most important feature, thus rendering the trade-off less relevant.

BERT and other LLMs Throughout our investigation we consider BERT as the target LLM architecture. Indeed, BERT represent the first large NN model—comprising 340 millions weights—which targets NLP and that is trained on large corpus of data collected from the web, namely the BooksCorpus dataset and a dump of the English Wikipedia of the time. We rely on BERT as it allows for the quick implementation of all LPE approaches available in the state-of-the-art. Indeed, few of these approaches require access to the inner mechanisms building the NN model to produce their explanations, thus being not applicable to closed source models such as the GPT family. The full focus on BERT represents a limitation of the proposed work, as the behaviour of BERT may differ significantly from other LLMs. Therefore, we consider the analysis of the application of our methodology to several different LLMs—wherever possible—as a future extension of this work. Moreover, we note that larger models—such as GPT or Llama—might exhibit some *emergent* properties not appearing in the adopted BERT model [56]. The emergent properties may somehow cause different results to be achieved employing the same methodology proposed in this paper, thus requiring further investigation. Indeed, emergent properties clash with model interpretability, thus making larger LLMs even more complex to analyse and inspect using available LPEs. Therefore, it is reasonable to expect an even larger lack of correlation amongst available LPEs over larger LLMs caused by the inherent fuzzy nature of their emergent properties which is difficult to analyse from a single or a few examples.

On the LLM reasoning principles From the very first proposal of LLMs, the research community has largely explored and speculated on their ability to reason over complex concepts. However, the definition of the LLMs reasoning capabilities and their limitations represents an open research question in the literature. Indeed, there is no definitive proof on the extent to which LLMs can process complex concepts incorporating human-like logical reasoning behaviour. Therefore, we here feel the need to stress that in this paper, whenever we refer to the *reasoning principles* of LLMs we consider the process by which the model elaborates the textual information given, without assuming any human-like reasoning capability from the LLM. Accordingly, the explanations extracted using GELPE mimic the information elaboration process of the LLM, rather than conjecturing the LLM's logical reasoning capabilities. Therefore, the reasoning process carried out in the logic program may be profoundly different to the reasoning capabilities of LLMs and rather represent the logic grounding of how information is elaborated sub-symbolically by the LLM.

6 Conclusions and future work

As intelligent agents are going to increasingly rely on LLMs for smooth interaction with humans and other agents, a fundamental issue for intelligent MASs is to open the LLM black-boxes, enabling explanation of their inner reasoning principles. However, xAI techniques for NLP still suffer several issues, linked with the heterogeneity of available local explanation techniques and the lack of robust global explanation processes. Inspired by these limitations, we propose a novel approach for enabling a fair comparison among state-of-the-art local post-hoc explanation mechanisms, aiming at identifying the extent to which their extracted explanations correlate. We rely on a novel framework for extracting and comparing global impact scores from local explanations obtained from LPEs, and apply such a framework over several text classification datasets, ranging from simple to complex tasks. Our experiments show how most LPEs explanations are far from being mutually correlated when LPEs are applied over a large set of input samples. These results highlight what we called the "quarrel" among state-of-the-art local explainers, highlighting the current fragility of xAI approaches for NLP. The disagreement is apparently caused by each of them focusing on a different set or subset of relevant concepts, or imposing a different distribution on top of them. Furthermore, we propose a novel approach to construct global explanations—under the form of logic programs—of the original LLM starting from the LPE outputs. We test the global explanation extraction approach—namely GELPE—over a broad set of scenarios, highlighting its fidelity against the sub-symbolic model and the simplicity of the extracted knowledge. Moreover, we analyse the efficiency of the extracted logic programs, showing how it is possible to extract a logic program that is equivalent to the original LLM and is faster and less energy wasteful in scenarios where hardware acceleration is not available. Therefore, our experiments show how the extraction process can be leveraged to enable the deployment of NLP applications to resource-constrained environments, such as embedded devices and microcontrollers. These findings also highlights how—for some learning tasks—leveraging LLMs might represents an over complication, as it is possible to achieve similar performance using simple and small logic programs.

Future work is likely to include the application of the proposed methodology to a broad range of state-of-the-art LLMs, starting from Llama and other available open-source architectures, aiming at showing if—and to what extent—the findings of this paper apply to models different from BERT. Similarly, we intend to extend the analysis on the trade-off between the achievable fidelity and efficiency of the surrogate logic programs extracted using GELPE. Finally, although the proposed framework is applied to the NLP realm, it represents a useful starting point for analysing the relevance of LPEs in different domains, such as computer vision [57, 58], graph processing [59–61] and many more.

Acknowledgements This work was partially supported by (i) PNRR—M4C2—Investimento 1.3, Partenariato Esteso PE00000013—"FAIR—Future Artificial Intelligence Research"—Spoke 8 "Pervasive AI", funded by the European Commission under the NextGenerationEU programme; (ii) the "ENGINES — ENGineering INtElligent Systems around intelligent agent technologies" project funded by the Italian MUR program "PRIN 2022" under grant number 20229ZXBZM; and (iii) the CHIST-ERA IV project "*Expectation*"—CHIST-ERA-19-XAI-005—, co-funded by EU and the Italian MUR (Ministry for University and Research).

Funding Open access funding provided by Alma Mater Studiorum - Università di Bologna within the CRUI-CARE Agreement.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

References

1. Zhang, L., Wang, S., & Liu, B. (2018). Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*. https://doi.org/10.1002/widm.1253
2. Hao, T., Li, X., He, Y., Wang, F. L., & Qu, Y. (2022). Recent progress in leveraging deep learning methods for question answering. *Neural Computing and Applications, 34*(4), 2765–2783. https://doi.org/10.1007/s00521-021-06748-3
3. Otter, D. W., Medina, J. R., & Kalita, J. K. (2021). A survey of the usages of deep learning for natural language processing. *IEEE Transactions on Neural Networks and Learning Systems, 32*(2), 604–624. https://doi.org/10.1109/TNNLS.2020.2979670
4. Warstadt, A., Singh, A., & Bowman, S. R. (2019). Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics, 7*, 625–641. https://doi.org/10.1162/tacl_a_00290
5. Stahlberg, F. (2020). Neural machine translation: A review. *Journal of Artificial Intelligence Research, 69*, 343–418. https://doi.org/10.1613/jair.1.12007
6. Lazaridou, A., & Baroni, M. (2020). *Emergent multi-agent communication in the deep learning era*. CoRR arXiv:2006.02419
7. Kocaballi, A. B., Berkovsky, S., Quiroz, J. C., Laranjo, L., Tong, H. L., Rezazadegan, D., Briatore, A., & Coiera, E. (2019). The personalization of conversational agents in health care: Systematic review. *Journal of Medical Internet Research, 21*(11), 15360. https://doi.org/10.2196/15360
8. Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., Sermanet, P., Jackson, T., Brown, N., Luu, L., Levine, S., Hausman, K., & Ichter, B. (2022). Inner monologue: Embodied reasoning through planning with language models. In K. Liu, D. Kulic, J. Ichnowski (Eds.), *Conference on robot learning (CoRL 2022). Proceedings of machine learning research* (vol. 205, pp. 1769–1782). PMLR. https://proceedings.mlr.press/v205/huang23c/huang23c.pdf
9. Cheng, M., Wei, W., & Hsieh, C. (2019). Evaluating and enhancing the robustness of dialogue systems: A case study on a negotiation agent. In J. Burstein, C. Doran, T. Solorio (Eds.), *2019 conference of the North American chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2019)* (vol. 1 (Long and Short Papers), pp. 3325–3335). Association for Computational Linguistics. https://doi.org/10.18653/V1/N19-1336
10. Glaese, A., McAleese, N., Trebacz, M., Aslanides, J., Firoiu, V., Ewalds, T., Rauh, M., Weidinger, L., Chadwick, M.J., Thacker, P., Campbell-Gillingham, L., Uesato, J., Huang, P., Comanescu, R., Yang, F., See, A., Dathathri, S., Greig, R., Chen, C., Fritz, D., Elias, J.S., Green, R., Mokrá, S., Fernando, N., Wu, B., Foley, R., Young, S., Gabriel, I., Isaac, W., Mellor, J., Hassabis, D., Kavukcuoglu, K., Hendricks, L.A., & Irving, G. (2022). *Improving alignment of dialogue agents via targeted human judgements*. CoRR arXiv:2209.14375
11. Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the Association for Computational Linguistics: Human Language Technologies* (vol. 1 (Long and Short Papers), pp. 4171–4186). Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1423
12. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., & Askell, A. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems, 33*, 1877–1901.

13. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research, 21*(1), 5485–5551.
14. Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency* (pp. 610–623). https://doi.org/10.1145/3442188.3445922
15. Zini, J. E., & Awad, M. (2022). On the explainability of natural language processing deep models. *ACM Computing Surveys, 55*(5), 1–31. https://doi.org/10.1145/3529755
16. Agiollo, A., Siebert, L. C., Murukannaiah, P. K., & Omicini, A. (2023). The quarrel of local post-hoc explainers for moral values classification in natural language processing. In *Explainable and transparent AI and multi-agent systems. Lecture notes in computer science* (Chapter 6, vol. 14127, pp. 97–115). Springer. https://doi.org/10.1007/978-3-031-40878-6_6
17. Ciatto, G., Sabbatini, F., Agiollo, A., Magnini, M., & Omicini, A. (2024). Symbolic knowledge extraction and injection with sub-symbolic predictors: A systematic literature review. *ACM Computing Surveys, 56*(6), 161–116135. https://doi.org/10.1145/3645103
18. Kautz, H. A. (2022). The third AI summer: AAAI Robert S. Engelmore Memorial Lecture. *AI Magazine, 43*(1), 93–104. https://doi.org/10.1609/AIMAG.V43I1.19122
19. Agiollo, A., Rafanelli, A., Magnini, M., Ciatto, G., & Omicini, A. (2023). Symbolic knowledge injection meets intelligent agents: QoS metrics and experiments. *Autonomous Agents and Multi-Agent Systems, 37*(2), 27–12730. https://doi.org/10.1007/s10458-023-09609-6
20. Agiollo, A., & Omicini, A. (2023). Measuring trustworthiness in neuro-symbolic integration. In *Proceedings of the 18th conference on computer science and intelligence systems. Annals of computer sciences and information systems* (vol. 35, pp. 1–10). https://doi.org/10.15439/2023F6019
21. Agiollo, A., Rafanelli, A., & Omicini, A. (2022). Towards quality-of-service metrics for symbolic knowledge injection. In *WOA 2022—23rd Workshop "From Objects to Agents"*. CEUR workshop proceedings (vol. 3261, pp. 30–47). Sun SITE Central Europe, RWTH Aachen University. http://ceur-ws.org/Vol-3261/paper3.pdf
22. Calegari, R., & Federico, S. (2023). *The PSyKE technology for trustworthy artificial intelligence*. In *XXI international conference of the Italian Association for Artificial Intelligence, AIxIA 2022, Udine, Italy, November 28–December 2, 2022, Proceedings* (vol. 13796, pp. 3–16). https://doi.org/10.1007/978-3-031-27181-6_1
23. Sabbatini, F., Ciatto, G., Calegari, R., & Omicini, A. (2022). Symbolic knowledge extraction from opaque ML predictors in PSyKE: Platform design & experiments. *Intelligenza Artificiale, 16*(1), 27–48. https://doi.org/10.3233/IA-210120
24. Sarkar, S., Babar, M. F., Hassan, M. M., Hasan, M., & Santu, S. K. K. (2023). *Exploring challenges of deploying BERT-based NLP models in resource-constrained embedded devices*. CoRR arXiv:2304.11520
25. Agiollo, A., & Omicini, A. (2021). Load classification: A case study for applying neural networks in hyper-constrained embedded devices. *Applied Sciences*. https://doi.org/10.3390/app112411957. Special Issue "Artificial Intelligence and Data Engineering in Engineering Applications"
26. Almeida, T. A., Hidalgo, J. M. G., & Yamakami, A. (2011). Contributions to the study of SMS spam filtering: new collection and results. In M. R. B. Hardy, & F. W. Tompa (Eds.), *Proceedings of the 2011 ACM symposium on document engineering* (pp. 259–262). ACM. https://doi.org/10.1145/2034691.2034742
27. Alberto, T. C., Lochter, J. V., & Almeida, T. A. (2015). TubeSpam: Comment spam filtering on YouTube. In T. Li, L. A. Kurgan, V. Palade, R. Goebel, A. Holzinger, K. Verspoor, & M. A. Wani, (Eds.), 14th IEEE international conference on machine learning and applications (ICMLA 2015) (pp. 138–143). IEEE. https://doi.org/10.1109/ICMLA.2015.37
28. Hoover, J., Portillo-Wightman, G., Yeh, L., Havaldar, S., Davani, A. M., Lin, Y., Kennedy, B., Atari, M., Kamel, Z., & Mendlen, M. (2020). Moral foundations Twitter corpus: A collection of 35k tweets annotated for moral sentiment. *Social Psychological and Personality Science, 11*(8), 1057–1071. DOI: https://doi.org/10.1177/194855061987662
29. Bayhaqy, A., Sfenrianto, S., Nainggolan, K., & Kaburuan, E. R. (2018). Sentiment analysis about e-commerce from tweets using decision tree, k-nearest neighbor, and Naïve Bayes. In *2018 International Conference on Orange Technologies (ICOT)* (pp. 1–6). https://doi.org/10.1109/ICOT.2018.8705796
30. Singh, J., & Tripathi, P. (2021). Sentiment analysis of twitter data by making use of svm, random forest and decision tree algorithm. In *2021 10th IEEE international conference on communication systems and network technologies (CSNT)* (pp. 193–198). https://doi.org/10.1109/CSNT51715.2021.9509679

31. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2018). A survey of methods for explaining black box models. *ACM Computing Surveys (CSUR), 51*(5), 93–19342. https://doi.org/10.1145/3236009
32. Adadi, A., & Berrada, M. (2018). Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access, 6*, 52138–52160. https://doi.org/10.1109/ACCESS.2018.2870052
33. Luo, S., Ivison, H., Han, S. C., & Poon, J. (2021). *Local interpretations for explainable natural language processing: A survey*. CoRR arXiv:2103.11072
34. Hailesilassie, T. (2016). Rule extraction algorithm for deep neural networks: A review. *International Journal of Computer Science and Information Security, 14*(7), 376–381.
35. Ibrahim, M., Louie, M., Modarres, C., & Paisley, J. (2019). Global explanations of neural networks: Mapping the landscape of predictions. In *Proceedings of the 2019 AAAI/ACM conference on AI, ethics, and society* (pp. 279–287).
36. Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems, 30*, 66.
37. Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). 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). https://doi.org/10.18653/v1/N16-3020
38. Madsen, A., Reddy, S., & Chandar, S. (2022). Post-hoc interpretability for neural NLP: A survey. *ACM Computing Surveys, 55*(8), 1–42. https://doi.org/10.1145/3546577
39. Danilevsky, M., Qian, K., Aharonov, R., Katsis, Y., Kawas, B., & Sen, P. (2020). A survey of the state of explainable AI for natural language processing. In *Proceedings of the 1st conference of the Asia-Pacific chapter of the Association for Computational Linguistics and the 10th international joint conference on natural language processing* (pp. 447–459). Association for Computational Linguistics.
40. Loh, W.-Y. (2014). Fifty years of classification and regression trees. *International Statistical Review, 82*(3), 329–348. https://doi.org/10.1111/insr.12016
41. Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., & Müller, K.-R. (2021). Explaining deep neural networks and beyond: A review of methods and applications. *Proceedings of the IEEE, 109*(3), 247–278. https://doi.org/10.1109/JPROC.2021.3060483
42. Kindermans, P.-J., Hooker, S., Adebayo, J., Alber, M., Schütt, K.T., Dähne, S., Erhan, D., & Kim, B. (2019). The (un)reliability of saliency methods. In *Explainable AI: Interpreting, explaining and visualizing deep learning* (pp. 267–280). Springer. https://doi.org/10.1007/978-3-030-28954-6_14
43. Ali, A., Schnake, T., Eberle, O., Montavon, G., Müller, K.-R., & Wolf, L. (2022). XAI for transformers: Better explanations through conservative propagation. In *International conference on machine learning* (pp. 435–451). PMLR.
44. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., & Samek, W. (2015). On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE, 10*(7), 0130140.
45. Tay, Y., Bahri, D., Metzler, D., Juan, D.-C., Zhao, Z., & Zheng, C. (2021). Synthesizer: Rethinking self-attention for transformer models. In *Proceedings of the 38th international conference on machine learning. Proceedings of machine learning research* (vol. 139, pp. 10183–10192). PMLR.
46. Abnar, S., & Zuidema, W. (2020). Quantifying attention flow in transformers. In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics* (pp. 4190–4197). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.385
47. Wu, Z., Nguyen, T.-S., & Ong, D. C. (2020). Structured self-attention weights encode semantics in sentiment analysis. In *Proceedings of the third blackbox NLP workshop on analyzing and interpreting neural networks for NLP* (pp. 255–264). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.blackboxnlp-1.24
48. Sundararajan, M., Taly, A., & Yan, Q. (2017). Axiomatic attribution for deep networks. In *International conference on machine learning*. Proceedings of machine learning research (vol. 70, pp. 3319–3328). PMLR. http://proceedings.mlr.press/v70/sundararajan17a/sundararajan17a.pdf
49. Kokalj, E., Škrlj, B., Lavrač, N., Pollak, S., & Robnik-Šikonja, M. (2021). BERT meets shapley: Extending SHAP explanations to transformer-based classifiers. In *Proceedings of the EACL hackashop on news media content analysis and automated report generation* (pp. 16–21).
50. Liscio, E., Araque, O., Gatti, L., Constantinescu, I., Jonker, C. M., Kalimeri, K., & Murukannaiah, P. K. (2023). What does a text classifier learn about morality? An explainable method for cross-domain comparison of moral rhetoric. In A. Rogers, J. Boyd-Graber, & N. Okazaki (Eds.) *Proceedings of the 61st annual meeting of the Association for Computational Linguistics, vol. 1: Long Papers* (pp. 14113–14132). Association for Computational Linguistics. https://doi.org/10.18653/v1/2023.acl-long.789
51. Nguyen, T. H., & Grishman, R. (2016). Modeling skip-grams for event detection with convolutional neural networks. In J. Su, X. Carreras, & K. Duh (Eds.), 2016 Conference on empirical methods in

natural language processing (EMNLP 2016) (pp. 886–891). The Association for Computational Linguistics. https://doi.org/10.18653/V1/D16-1085

52. Li, X., & Roth, D. (2002). Learning question classifiers. In *19th International conference on computational linguistics (COLING 2002), Taipei, Taiwan*. https://aclanthology.org/C02-1150
53. Zhang, Z., & Sabuncu, M. (2018). Generalized cross entropy loss for training deep neural networks with noisy labels. In *Advances in neural information processing systems* (vol. 31). Curran Associates, Inc..
54. Kiesel, J., Alshomary, M., Handke, N., Cai, X., Wachsmuth, H., & Stein, B. (2022). Identifying the human values behind arguments. In *Proceedings of the 60th annual meeting of the Association for Computational Linguistics* (vol. 1: Long Papers, pp. 4459–4471). https://doi.org/10.18653/v1/2022.acl-long.306
55. Alshomary, M., Baff, R. E., Gurcke, T., & Wachsmuth, H. (2022). The moral debater: A study on the computational generation of morally framed arguments. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics* (vol. 1: Long Papers, pp. 8782–8797). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.acl-long.601
56. Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., & Chi, E. H. (2022). Emergent abilities of large language models. *Transactions on Machine Learning Research, 2022*, 66.
57. Buhrmester, V., Münch, D., & Arens, M. (2021). Analysis of explainers of black box deep neural networks for computer vision: A survey. *Machine Learning & Knowledge Extraction, 3*(4), 966–989. https://doi.org/10.3390/make3040048
58. Agiollo, A., Ciatto, G., & Omicini, A. (2021). *Shallow2Deep*: Restraining neural networks opacity through neural architecture search. In *Explainable and transparent AI and multi-agent systems. Third international workshop, EXTRAAMAS 2021. Lecture notes in computer science* (vol. 12688, pp. 63–82). Springer. https://doi.org/10.1007/978-3-030-82017-6_5
59. Jaume, G., Pati, P., Bozorgtabar, B., Foncubierta, A., Anniciello, A.M., Feroce, F., Rau, T., Thiran, J., Gabrani, M., & Goksel, O. (2021). Quantifying explainers of graph neural networks in computational pathology. In *IEEE conference on computer vision and pattern recognition, CVPR 2021, Virtual, June 19–25, 2021*, (pp. 8106–8116). Computer Vision Foundation/IEEE. https://doi.org/10.1109/CVPR46437.2021.00801
60. Agiollo, A., & Omicini, A. (2022). GNN2GNN: Graph neural networks to generate neural networks. In J. Cussens, & K. Zhang (Eds.) Uncertainty in artificial intelligence. Proceedings of machine learning research (vol. 180, pp. 32–42). ML Research Press. https://proceedings.mlr.press/v180/agiollo22a.html
61. Agiollo, A., Bardhi, E., Conti, M., Lazzeretti, R., Losiouk, E., & Omicini, A. (2023). GNN4IFA: Interest flooding attack detection with graph neural networks. In *2023 IEEE 8th European symposium on security and privacy (EuroS &P)* (pp. 615–630). IEEE Computer Society. https://doi.org/10.1109/EuroSP57164.2023.00043

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.