Linearly-interpretable concept embedding models for text analysis

Machine Learning (2025) 114:224
https://doi.org/10.1007/s10994-025-06839-5

Linearly-interpretable concept embedding models for text analysis

Francesco De Santis[1] · Philippe Bich[2] · Gabriele Ciravegna[1,3] · Pietro Barbiero[4] · Danilo Giordano[1] · Tania Cerquitelli[1]

Received: 18 April 2025 / Revised: 5 July 2025 / Accepted: 9 July 2025

Abstract

Despite their success, Large-Language Models (LLMs) still face criticism due to their lack of interpretability. Traditional post-hoc interpretation methods, based on attention and gradient-based analysis, offer limited insights as they only approximate the model's decision-making processes and have been proved to be unreliable. For this reason, Concept-Bottleneck Models (CBMs) have been lately proposed in the textual field to provide interpretable predictions based on human-understandable concepts. However, CBMs still exhibit several limitations due to their architectural constraints limiting their expressivity, to the absence of task-interpretability when employing non-linear task predictors and for requiring extensive annotations that are impractical for real-world text data. In this paper, we address these challenges by proposing a novel Linearly Interpretable Concept Embedding Model (LICEM) going beyond the current accuracy-interpretability trade-off. LICEMs classification accuracy is better than existing interpretable models and matches black-box ones. We show that the explanations provided by our models are more intervenable and causally consistent with respect to existing solutions. Finally, we show that LICEMs can be trained without requiring any concept supervision, as concepts can be automatically predicted when using an LLM backbone.

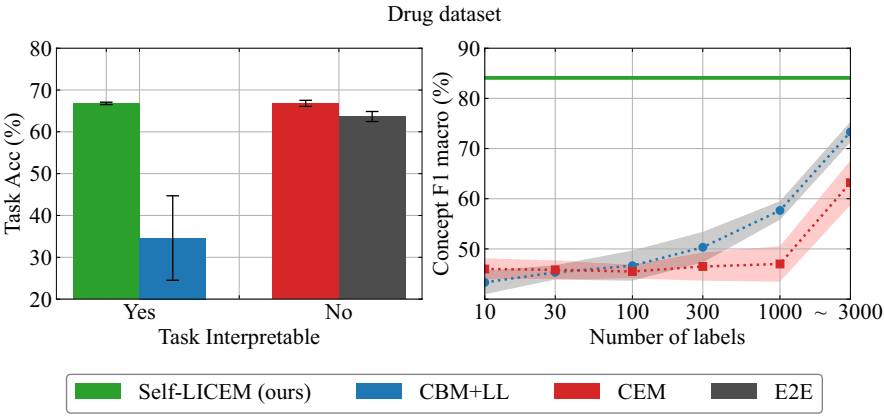
Editors: Maria Concepcion Bielza Lozoya, Ana Carolina Lorena, Tiago A. Almeida.

Francesco De Santis, Philippe Bich, and Gabriele Ciravegna have contributed equally to this work.

Extended author information available on the last page of the article

Published online: 04 September 2025

Graphical abstract



Keywords Concept-XAI · Text analysis · Linear concept attribution

1 Introduction

In recent years, Large-Language Models (LLMs) have revolutionized the way we approach text interpretation, generation, and classification (Kenton & Toutanova, 2019; Brown et al., 2020). Despite their success, LLMs' reliability is insufficient, due to the occurrence of hallucinations (Huang et al., 2023) and the inconsistency of self-provided explanations that often do not reflect the actual decision-making process (Ye & Durrett, 2022; Madsen et al., 2024). Traditional explainability methods rely mainly on the post-hoc analysis of the attention mechanisms (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019) and the output gradients (Chefer et al., 2021), both of which have shown limited interpretability as they are often unreliable (Adebayo et al., 2018; Taimeskhanov et al., 2024) and only show *where* the model looks, but not *what* it sees in a given input (Rudin, 2019; Poeta et al., 2023).

For this reason, Concept-Bottleneck Models (CBMs) (Koh et al., 2020) have been recently proposed in the textual field to improve the interpretability of LLM predictions (Tan et al., 2024, 2024). In CBMs, an intermediate layer outputs a set of human-understandable symbols, commonly referred to as concepts, before providing the final classification. While CBMs utilize a black-box module to predict concepts, they enhance the interpretability of end-to-end (E2E) deep neural networks by providing a transparent intermediate representation that allows users to interact with the model (Koh et al., 2020). With CBMs, users can check and modify the predicted concepts to extract counterfactual predictions. However, CBMs still present several limitations: (i) the concept bottleneck architecture prevents high classification accuracy, especially in real-world text scenarios where complete concept representations (Yeh et al., 2020) are difficult to obtain; (ii) when CBMs employ non-linear task predictors or provide predictions on top of concept embeddings Espinosa Zarlenga et al. (2022), they are not *task-interpretable*, i.e., the decision process from the concepts to the final classification is non-interpretable; (iii) concept annotation

in CBMs is expensive, and existing generative concept annotation approaches (Tan et al., 2024) require the use of multiple modules.

This paper tackles these challenges by proposing a novel Linearly-Interpretable Concept Embedding Model (LICEM). LICEM provides the final classification through an interpretable linear equation over concepts. Specifically, both the weights and the independent variables (concept predictions) of the linear equation are predicted for each individual sample. As illustrated in Fig. 1 (left), LICEM identifies a few relevant concepts in the text: an *Effective* drug with *Side Effects*. While the presence of *Side Effects* normally indicates a poor drug, LICEM adjusts the concept relevance according to the meaning of the specific input text. In this case, LICEM recognizes that the predicted side effect is marginally relevant because "most other medications result in [it]," thus assigning low concept importance to *Side Effects* while giving higher importance to *Effective*, leading to an overall positive review for the drug. In contrast, the CBM provides an incorrect prediction because it relies solely on the predicted concepts and their global importance for predicting a given task. In the reported example, CBM predicts a negative drug review solely due to the presence of *Side Effects*. Furthermore, when employing a non-linear task predictor, the CBM decision-making process is non-interpretable from the concept to the task, while LICEM explicitly reveals its reasoning.

In the experiments, we show that LICEM addresses all the above-mentioned CBM issues. In particular, we show that: (i) LICEM achieves higher accuracy than existing task-interpretable models (e.g. CBM+LL, a CBM with a linear layer on top) while matching or surpassing black-box methods (Fig. 1, middle); (ii) LICEM explanations are more intervenable and causally consistent with respect to existing solutions; (iii) LICEM can be trained without any concept annotation (Self-LICEM), as concepts can be automatically predicted by its LLM backbone, often providing higher concept accuracy than standard methods with full annotations (Fig. 1, right).

2 Related work

This section reviews prior work relevant to our approach, with a focus on LLM interpretability and concept-based models.

LLM interpretability: Recent studies have highlighted the unreliability of LLMs, as they often occur hallucinations (Ji et al., 2023), and when prompted for explanations, their

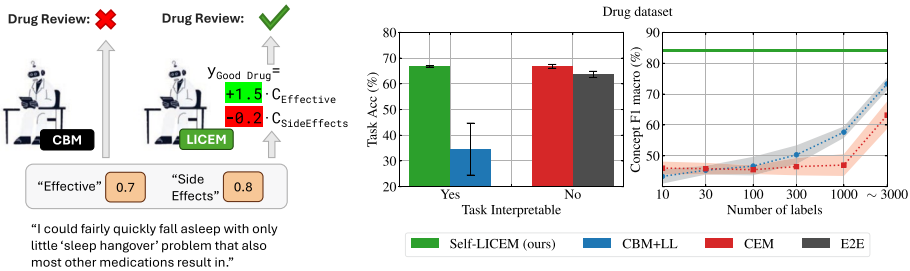


Fig. 1 Left, LICEM predicting the sentiment of a drug review (Gaber et al., 2018). LICEM provides accurate predictions and reveals its decision-making process. Middle, LICEM provides the best accuracy/interpretability trade-off. Right, models' concept F1 scores, when increasing the number of concept annotations. Self-LICEM achieves high scores without requiring concept labels

responses frequently do not reflect the actual decision-making process (Ye & Durrett, 2022; Madsen et al., 2024; Turpin et al., 2024). Although the attention mechanism in transformer models offers some interpretability, it has been criticized for its lack of clarity and consistency (Jain & Wallace, 2019; Wiegreffe & Pinter, 2019). To improve LLM explainability, various standard XAI techniques, such as LIME (Ribeiro et al., 2016) and Shapley values (Lundberg & Lee, 2017), along with newer methods (Kokalj et al., 2021; Heyen et al., 2024; Chefer et al., 2021, 2021), have been employed. However, these standard techniques have limitations (Kindermans et al., 2019; Ghorbani et al., 2019; Adebayo et al., 2018; Taimeskhanov et al., 2024), primarily because they explain predictions in terms of input features that often lack meaningful interpretations for non-experts (Poursabzi-Sangdeh et al., 2021). Consequently, researchers are now exploring interpretable-by-design models also in the textual domain (Rajagopal et al., 2021; Jain et al., 2022; Tan et al., 2024, 2024).

Concept-based models: Concept-based models (Alvarez Melis & Jaakkola, 2018; Koh et al., 2020; Ciravegna et al., 2023; Kim et al., 2023) are transparent and interactive models that utilize an intermediate layer to represent concepts. To increase the representation capability of the concept layer, Espinosa Zarlenga et al. (2022) proposed using concept embeddings. However, the interpretability of CEM task predictor is limited, as individual embedding dimensions lack clear meaning. In this work, we demonstrate how to create an interpretable task predictor over these embeddings. A recent neurosymbolic method (DCR, Barbiero et al. (2023)), based on fuzzy logic, also attempted to tackle this issue. While we extend CEM and DCR applicability to the textual domain, we show that LICEM achieves superior predictive performance than DCR and higher interpretability than both, as confirmed by a user study. Additionally, supervised concept-based models (Koh et al., 2020; Espinosa Zarlenga et al., 2022) often require extensive concept annotations, which are frequently unavailable, particularly in text. We enhance a recent generative approach (Yang et al., 2023; Oikarinen et al., 2023; Ludan et al., 2023) by using the same LLM for self-generated concept predictions and sample representations.

3 Background

Before introducing our proposed methodology, we outline the foundational methods that underpin our study. Specifically, we first describe CBM and CEM and then explain how Large Language Models (LLMs) can be leveraged to extract rich textual representations.

CBMs: As shown in Fig. 1 (left), CBMs (Koh et al., 2020; Tan et al., 2024) are transparent models that break the standard end-to-end learning paradigm into the training of two neural modules $f \circ g$. The concept encoder $g : X \to C$ maps raw features $x \in X \subset \mathbb{R}^d$ into m higher-level abstractions $c \in C \subset [0,1]^m$ (i.e., the concepts); the task encoder $f : C \to Y$ predicts n downstream classes based on the learned concepts $\hat{y} = f(g(x)), y \in Y \subset [0,1]^n$. Recalling the example in Fig. 1, a CBM decomposes the classification task into an inital prediction of drug-related attributes—the presence of *Side Effects* and its *Effectiveness*—which are subsequently used to evaluate the overall drug sentiment. CBMs are normally trained to minimize a composite cost function, considering concept and task learning: $\mathcal{L} = H(\hat{c}, c) + \lambda \cdot H(\hat{y}, y)$, where H denotes the standard cross-entropy function and $\lambda \in [0,1]$ is a coefficient used to prioritize concept learning relative to task learning. CBMs are considered more interpretable than DNNs because they employ a transparent intermediate representation, and they inherently deliver counterfactual predictions. However, the

expressivity of CBM is limited by the bottleneck representation created by the concept layer. This issue is particularly relevant when a single layer is used as task-predictor and when the concepts employed are incomplete (Yeh et al., 2020), i.e., they are not sufficient to uniquely distinguish the final classes, a frequent condition in Natural Language Processing (NLP) contexts. Figure 1 shows a case where this limitation is evident. While *Side Effects* are usually linked to negative sentiment, they are irrelevant in this example. Yet, the CBM still predicts a negative outcome, as it relies solely on the concept predictions and is unable to capture the broader context.

CEMs: Concept Embedding Models (CEMs) (Espinosa Zarlenga et al., 2022; Kim et al., 2023) address the limited expressivity of CBMs by generating a concept embedding to represent each concept. Initially, CEMs decompose the concept encoder into two functions $g = q \circ h$. The inner function $h : X \rightarrow H \subset \mathbb{R}^b$ provides a representation of an input sample, while $q : H \rightarrow \mathbf{C}$ maps this representation into m k-dimensional concept embeddings $\mathbf{c} \in \mathbf{C} \subset \mathbb{R}^{m,k}$. The concept prediction $\hat{c}_j$ is then given by a neural function over the concept embeddings $\hat{c}_j = s(\mathbf{c_j})$, where s is shared among the m concepts, while the task prediction is generated by a task function $f : \mathbf{C} \rightarrow Y$ $f(\mathbf{c})$ working on the concatenation of all concept embeddings. Hence, as shown in Fig. 1 (middle) the expressivity of CEM is much higher than CBM, as it is not constrained to represent concepts with single neurons. sOn the other side, the interpretability of the CEM task predictor $f(\mathbf{c})$ is very limited, as the individual dimensions of concept embeddings are not interpretable: while the concept $c_{\text{Side Effects}} = 0.7$ is interpretable, the single dimensions associated to the corresponding embedding $\mathbf{c}_{\text{Side Effects}} = [0.2, 2.5, \ldots, -1.7]$ are not semantically meaningful. CEM makes predictions on top the concept embeddings, thus, even when using a single linear layer, the task predictor is not interpretable. Ideally, we want a task predictor that combines the expressiveness of CEM with the interpretability of logistic regression applied directly to concept predictions. Also, to date, no adaption of CEM architectures to text scenarios has been proposed.

LLM-based textual encoders: When considering transformer models, there exist several methods for implementing a text encoder $h(x)$. An immediate choice is to employ an encoder-only architecture, such as BERT (Kenton & Toutanova, 2019), and extracting the embedding associated to the [CLS] token. However, as recently shown in Jiang et al. (2023), one can also exploit the remarkable performance of existing decoder-only LLMs. The architecture of an LLM can be conceptualized as comprising two distinct components: the stacked decoder blocks which are responsible for generating a contextualized representation e, and a classification head that processes this representation to predict the next token. The stacked decoder blocks can be formalized as a function $h(\cdot) : D^l \rightarrow \mathbb{R}^b$, where D denotes the dictionary of tokens recognized by the LLM, and l represents the length of the context window. This function maps an input x to its vector representation $e = h(x)$. To facilitate the generation of sentence-representative embeddings, Jiang et al. (2023) proposed embedding the input within the prompt as *"this sentence: [sentence] means in one word:"* and substituting *"[sentence]"* with the specific text to encode.

4 Method

In this paper, we propose an interpretable concept-based model for text classification that leverages rich text and concept representations. Figure 2 illustrates the complete pipeline. A pretrained LLM is used as the text encoder to extract contextualized representations by

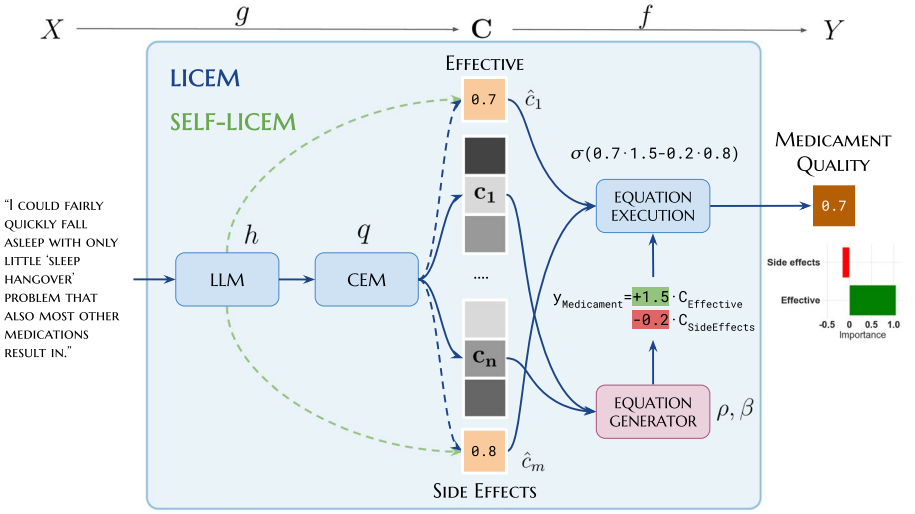


Fig. 2 LICEMs visualization. Using a pretrained LLM model, we (i) require it to provide an encoding of the input text following; (ii) prompt the LLM to generate the concepts predictions $\hat{c}$ (e.g., Side Effects = 0.8) in Self-LICEM, while in LICEM they are provided by a concept embedding layer; (iii) make the final prediction in an interpretable way by first predicting the equation weights w_{ij} (e.g., $w_{\text{Side Effects}} = -1.8$) for predicting the i-th class, then executing the resulting linear equation

employing the prompting strategy introduced in Jiang et al. (2023), thus generating the embedding $e = h(x)$ for the input text without requiring fine-tuning of the encoder. The encoded text is then passed through a concept embedding layer Espinosa Zarlenga et al. (2022), producing concept embeddings $\mathbf{c} = q(e)$ and corresponding concept predictions $\hat{c} = s(\mathbf{c})$. The proposed model (LICEM, Sect. 4.1) produces an interpretable task prediction by leveraging both the concept embeddings and predictions. Furthermore, using a pretrained LLMs as text encoder allows for the self-generation of concept predictions (Self-LICEM, Sect. 4.2).

4.1 Linearly-interpretable concept embedding model (LICEM)

To create an interpretable model, it is essential to utilize both an interpretable data representation and an interpretable function (Ribeiro et al., 2016; Rudin, 2019). To avoid generalization losses, state-of-the-art concept-based models lacks one of the characteristics: they either employ non-linear functions on top of concept predictions (losing functional interpretability), or any function on top of concept embeddings whose single dimensions are non-interpretable (losing data interpretability).

To address this issue, in this work, we propose to *neurally generate a linear equation* that can be *symbolically executed* over the concept predictions. In this way, the final classification is outputted as an interpretable aggregation of the most important concepts. More precisely, we predict the weights and the bias of a set of linear equations each one predicting one class, and in which the independent variables are the concepts scores $\hat{c}_j$ predicted by CEM concept encoder. The prediction of both parameters (weights and biases) is provided by two neural modules ρ, β working over CEM concept embeddings $\mathbf{c}$. The first neural module $\rho : \mathbf{C_j} \rightarrow \mathbb{R}^n$ predicts for a single concept j the weights for all classes

$\hat{w}_j = [\hat{w}_{1j}, \ldots, \hat{w}_{ij}] = \rho_i(\mathbf{c_j})$. The second neural module $\beta : \mathbf{C} \rightarrow \mathbb{R}$ predicts the bias term $\hat{b} = [b_1, \ldots, b_n] = \beta(\mathbf{c})$ over the concatenation of all concept embeddings $\mathbf{c} = [\mathbf{c_1}, \ldots, \mathbf{c_m}]$, which represents the general bias for each class. Overall, we can describe LICEM predictions as:

$$\text{LICEM}: \quad \hat{y}_i = \sigma\left(\sum_j \hat{w}_{ij}\hat{c}_j + \hat{b}_i\right) \tag{1}$$

where, as in common logistic regressions, $\hat{w}_{ij}$ is the weight for the j-th concept in predicting the i-th task, b_i is the bias for the i-th task, $\hat{w}_{ij} < 0$ indicates a negatively important concept, $\hat{w}_{ij} > 0$ a positively important one, and $\hat{w}_{ij} \sim 0$ a non-important concept. Notice that the bias term is optional, but it allows for positive predictions even when no concept is positively predicted. Indeed, when $\hat{c}_j = 0$ for all $j \in \{1, ..., m\}$, the prediction would be $\hat{y}_i = 0$ regardless of $\hat{w}_{ij}$. Also, σ represents a sigmoid activation function for binary classification tasks and a softmax for multi-class classification tasks.

Finally, to understand the contribution of a concept to the final prediction of a class, we propose considering the combined contribution $\hat{w}_{ij}\hat{c}_j$ and reporting them with a feature importance plot, as shown in the output of Fig. 2.

Training: LICEM is trained similarly to standard concept-based models, with a cross-entropy H loss over the predicted concepts and tasks, with the addition of a few regularizations. To improve the readability of LICEM equations, we add an L_1 regularization promoting sparse weights (i.e., $\hat{w}_j \neq 0$), thus equations composed of few terms. The analysis regarding the sparsity achieved by LICEM is shown in Appendix A.10. To prevent over-reliance on the bias term, we also add an L_2 regularization to encourage small bias values and minimize its influence on task prediction.

$$\mathcal{L}_{sup} = H(c, \hat{c}) + \lambda_y H(y, \hat{y}) + \lambda_w ||w||_1 + \lambda_b ||b||_2 \tag{2}$$

where we indicate the loss over the concept predictions as $\mathcal{L}_c = H(\hat{c}, c)$, the loss over the task predictions as $\mathcal{L}_t = H(\hat{y}, y)$, with $||w||$ and $||b||$ the regularization terms over the weights and biases and with λ_y, λ_w and λ_b the optimization weights for each term. In the rest of the paper, we will refer to this strategy as *supervised*.

4.2 Exploiting LLMs to avoid concept annotation: a self-generative approach

To alleviate human annotators from the burden of providing concept supervision, a few works are starting to exploit the knowledge already available in pre-trained LLMs, both in the image (Oikarinen et al., 2023; Yang et al., 2023) and in the textual domains (Tan et al., 2024). First, an LLM is asked to provide several attributes that describe each class. Each attribute is considered a concept for that class, possibly shared with other classes. For instance, a *parrot* may be described as having *bright feathers* and *medium size*. Then another LLM is required to predict whether the concept is present in the input samples. The LLM, in this case, is formally represented by the distribution p_θ, where θ denotes the parameters of a pre-trained LLM with classification head. When conditioned on a prompt t, the model generates the token"yes"if a specific concept is identified in the input text sequence x, and"no" otherwise. Thus, the predicted concept is sampled as $c' \sim p_\theta(c'|t, x)$. In Appendix A.1 we report some examples of prompts.

Generative approach: In Tan et al. (2024), these concept predictions c' are used as labels to train a textual concept encoder. Formally, $\mathcal{L}_{gen} = \mathcal{L}_{c'} + \lambda\mathcal{L}_t = H(c', \hat{c}) + \lambda H(y, \hat{y})$. We will refer to this strategy as *generative*, as a generative model provides concept annotations.

Self-generative approach: While the generative approach reduces human annotation efforts, it requires training an additional concept encoder to learn the LLM-provided labels. In this paper, since we already employ an LLM as a text encoder, we propose using the same LLM to directly make the concept predictions. More precisely, we prompt the LLM to provide both a representation e for each sample x and the concept predictions, i.e., $\hat{c} = c' \sim p_\theta(c'|t, x)$. This results in a modification of both CEM and LICEM as the concept predictions are self-generated by the same LLM, as shown in Fig. 2. We will refer to this approach as *self-generative*, as the same model directly provides the concept predictions. This method eliminates the need for concept annotations, but also reduces the number of parameters to train and improves concept performance if compared to the generative method. Indeed, the concept accuracy of the self-generative method represents an optimum for the generative one. In the former, the concepts c' provided by the LLM are directly used as concept predictions, while in the latter, they serve as training labels for an external text encoder, which aims to replicate c'. Self-LICEM is obtained by substituting the concept predictions $\hat{c}$ with c' from Eq. 1:

$$\text{Self-LICEM} \quad \hat{y}_i = \sigma\left(\sum_j \hat{w}_{ij} c'_j + \hat{b}_i\right). \tag{3}$$

The concept embedding encoder q and the neural modules ρ and β producing the interpretable linear equation are trained as in Eq. 2, but minimizing, this time, only the loss over the task:

$$\mathcal{L}_{selfgen} = H(y, \hat{y}) + \lambda_w ||w||_1 + \lambda_b ||b||_2, \tag{4}$$

This approach is not limited to LICEM; it can also be extended to CBM-based and CEM-based models. In these cases, the LLM provides the concept predictions (CBM) or the predictions and the embedding (CEM). In both cases, the optimization strategy involves minimizing only the cross-entropy on the task predictions $H(y, \hat{y})$, as shown in Eq. 4. This enables converting any pre-trained LLM into a concept-based model without the need for concept annotations.

5 Experiments

In this section, we want to answer the following research questions:

- **Generalization:** Does LICEM achieve superior performance in text analysis compared to other interpretable models, and is it on par with non-interpretable ones? How does the self-generative approach perform? (Sect. 5.2)
- **Concept efficiency:** How many concept supervisions are required to match Self-LICEM accuracy? Does the self-generative strategy outperform the generative one in concept accuracy? (Sect. 5.3)
- **Interpretability:** Can we effectively interact with LICEM? Are LICEM explanations clear and driven by most important concepts? (Sect. 5.4)

5.1 Setup

We test LICEM performance over different datasets (both with and without concept-supervisions), comparing against several models and for different metrics. For all experiments, we report the average and standard deviation across three repetitions. The models were trained on a dedicated server equipped with an AMD EPYC 7543 32-Core processor and one NVIDIA A100 GPU[1].

Dataset: We evaluated LICEM performance on three text classification datasets with available concept annotations: CEBaB (Abraham et al., 2022), MultiEmotions-IT (Sprugnoli et al., 2020), and Drug (Gaber et al., 2018). CEBaB is a dataset designed to study the causal effects of real-world concepts on NLP models. It includes short restaurant reviews annotated with sentiment ratings at both the overall review level (positive, neutral, and negative reviews) and for four dining experience aspects, which we use as concept labels: 'Good Food', 'Good Ambiance', 'Good Service', and 'Good Noise'. MultiEmotions-IT is a dataset designed for opinion polarity and emotion analysis, containing comments in Italian related to videos and advertisements posted on social media platforms. These comments have been manually annotated according to different aspects, from which we selected two dimensions: opinion polarity, describing the overall sentiment expressed by users (used as task label), and basic emotions. We selected 'Joy', 'Trust', 'Sadness', and 'Surprise' as the concept labels. The Drug dataset provides patient reviews on specific drugs. The reviews are annotated with the overall satisfaction of users (which we discretized to a binary representation) and drug experience annotations, namely 'Effectiveness' and 'Side Effects', which we used as concept labels. Additionally, we tested the generative and self-generative approaches on the Depression (Yates et al., 2017)[2], IMDb (Maas et al., 2011), TREC-50 (Li & Roth, 2002; Hovy et al., 2001)(Huanget al.,2025), Banking-77 (Casanueva et al., 2020) and CLINC-OOS (Larson et al., 2019). These datasets span a range of domains and allow for comprehensive assessment of classification performance. The number of classes varies significantly, from binary sentiment analysis in IMDb (2 classes) to fine-grained intent detection in CLINC-OOS (151 classes). This diversity ensures robust evaluation across different levels of classification complexity. Additional details regarding the datasets are reported in Appendix A.2.

Baselines: We compare LICEM against several baselines, including black-box and concept-based models, both task-interpretable and non-interpretable approaches. For all models, we use a non fine-tuned Mixtral 8x7B (Jiang et al., 2024) encoder $h(x)$, following the encoding strategy proposed in Jiang et al. (2023). In Appendix A.3 we also report all results based on a fine-tuned BERT encoder (Kenton & Toutanova, 2019) as backbone. The results show that the decoder-only LLM achieves similar performance without fine-tuning the whole LLM. Besides, it enables the self-generative approach: in Appendix A.4 we report a comparison of the concept annotation performance when using different LLMs. For black-box models (E2E), we evaluate an end-to-end model directly classifying the task with a Mixtral encoder $h(x)$ and few layers as classification head (MLP), and the same Mixtral used in Zero-shot and Few-shot prompting. CBM+LL and CBM+MLP are the two CBMs originally proposed in (Koh et al., 2020) and recently adapted to text in (Tan et al., 2024). They employ a concept bottleneck layer followed, the first one, by an interpretable

[1] Our code is available at https://github.com/francescoTheSantis.

[2] For this dataset, we used the cleaned version available on Kaggle.

linear layer, while the second by a non-interpretable multi-layer perceptron. CBM+DT and CBM+XG are respectively two CBM variants proposed in (Barbiero et al., 2023), using a decision tree and a XGBoost classifier (Chen & Guestrin, 20176) on top of the concept bottleneck layer, respectively. CBM+DT is task-interpretable, as one can extract a decision rule based on concepts, whereas the second variant CBM+XG is non task- interpretable. As described in Sect. 3, CEM (Espinosa Zarlenga et al., 2022) employs embeddings to represent concepts and enhance CBM generalization performance, but at the cost of losing task interpretability. Finally, DCR (Barbiero et al., 2023) is a neuro-symbolic approach designed to improve the interpretability of CEM. It generates propositional rules executed by a fuzzy system on top of concept predictions. We adapt CEM and DCR to work in the text analysis scenario, and we compare their performance against the proposed model. For the training details regarding each model, please refer to Appendix A.2.

Metrics: We evaluate LICEM using various metrics. To assess **generalization** performance, we compute the task accuracy and the macro-averaged concept F1 score (as concept classes are highly imbalanced); for self-generative models, the macro-averaged F1 score evaluates the concept predictions directly provided by the LLM (Sect. 4.2). To measure **efficiency**, we examine the concept F1 score of all models when increasing the number of concept annotations. For **interpretability**, we evaluate the effectiveness of concept interventions in LICEM to enhance classification accuracy (Koh et al., 2020) and we compute the area under the accuracy gain curve for each model-dataset combination, calculated using the composite trapezoidal rule; secondly, we measure the Causal-Concept Effect (CaCE) (Goyal et al., 2019), which assesses the causal relevance of concepts for task predictions; thirdly, we qualitatively report some of the equations generated by LICEM to demonstrate their clarity; lastly, we also conducted a user study to evaluate how easily LICEM's explanations can be understood from a human perspective, providing insight into their interpretability in real-world settings.

5.2 LICEM generalization

LICEM matches black-box task performance and outperforms all task-interpretable models (Table 1).

LICEM consistently outperforms competing models across a diverse set of datasets, achieving either the highest or statistically equivalent task accuracy. In the supervised setting, LICEM is the top-performing task-interpretable model, showing a $7-19\%$ improvement over CBM variants and a $1-2\%$ gain over DCR. This performance gap widens in the generative setting, where evaluation involves more challenging tasks (e.g., classification over 151 classes in CLINC-OOS). Here, CBM-based models perform significantly worse, and while DCR outperforms CBMs, it still falls short of LICEM, with a notable 55% accuracy gap on CLINC-OOS. In the self-supervised setting, overall model performance improves, including that of CBM variants and DCR. LICEM continues to lead, outperforming DCR on 6 out of 8 datasets and matching its performance on IMDb. We attribute LICEM's advantage over DCR to its simpler and more direct classification mechanism: LICEM predicts the parameters of a linear function, whereas DCR constructs and optimizes complex fuzzy logic rules. The linear approach requires only a weighted sum, making both training and inference more efficient.

Self-generative approach increases concept-based models applicability and improves task performance (Fig. 3). To better analyze whether a certain annotation type improves the task accuracy of a model, in Fig. 3 we report the concept-based model average task

Table 1 Task accuracy (%) of the compared models

Type	Method	C.S	T.I	CEBaB	Multiemo-It	Drug	Depression	IMDb	TREC-50	CLINC-OOS	Banking-77
Mixtral (e2e)	MLP	✗	✗	**88.80** ± 0.75	80.01 ± 0.63	**63.66** ± 1.20	**97.18** ± 0.03	86.92 ± 0.48	**86.33** ± 0.23	**85.14** ± 0.19	**91.81** ± 0.24
	Zero-Shot	✗	✗	86.80 ± 0.31	80.06 ± 0.66	60.81 ± 0.28	73.77 ± 0.23	95.05 ± 0.19	69.67 ± 0.61	82.22 ± 0.43	78.34 ± 0.11
	Few-Shot	✗	✗	84.79 ± 0.67	**84.17** ± 0.67	62.16 ± 0.27	76.38 ± 0.08	**94.67** ± 0.00	72.13 ± 0.31	84.22 ± 0.59	78.23 ± 0.11
Sup.	CBM+MLP	✓	✗	78.41 ± 9.30	45.43 ± 8.20	45.42 ± 4.90	–	–	–	–	–
	CBM+XG	✓	✗	83.01 ± 0.10	69.01 ± 0.02	55.00 ± 0.13	–	–	–	–	–
	CEM	✓	✗	**89.60** ± 0.49	**83.33** ± 0.47	**66.81** ± 0.40	–	–	–	–	–
	CBM+LL	✓	✓	71.43 ± 9.71	42.67 ± 7.01	34.60 ± 10.10	–	–	–	–	–
	CBM+DT	✓	✓	77.20 ± 0.40	65.00 ± 0.02	47.20 ± 0.40	–	–	–	–	–
	DCR	✓	✓	88.05 ± 0.53	82.01 ± 0.71	65.40 ± 0.80	–	–	–	–	–
	LICEM	✓	✓	**89.89** ± 0.77	**83.47** ± 0.49	**66.80** ± 0.29	–	–	–	–	–
Gen.	CBM+MLP	✗	✗	76.19 ± 3.40	70.28 ± 0.65	42.43 ± 1.10	84.72 ± 1.72	84.76 ± 0.66	49.13 ± 0.23	14.30 ± 0.70	13.77 ± 0.65
	CBM+XG	✗	✗	74.43 ± 0.40	69.80 ± 0.09	53.85 ± 0.30	86.87 ± 0.03	69.71 ± 1.06	38.47 ± 5.31	5.50 ± 7.67	3.16 ± 0.97
	CEM	✗	✗	**89.97** ± 0.66	**82.41** ± 0.11	**63.80** ± 0.38	**97.06** ± 0.11	**85.97** ± 0.11	**77.67** ± 1.29	**69.66** ± 0.35	**86.31** ± 0.44
	CBM+LL	✗	✓	62.07 ± 0.22	68.66 ± 4.20	33.14 ± 2.10	50.25 ± 0.39	82.67 ± 0.66	46.33 ± 0.12	13.87 ± 0.33	10.48 ± 0.07
	CBM+DT	✗	✓	78.03 ± 0.23	65.32 ± 0.39	40.06 ± 1.30	83.10 ± 0.13	72.00 ± 2.84	22.07 ± 18.61	1.48 ± 0.26	4.08 ± 0.84
	DCR	✗	✓	88.97 ± 0.18	**80.82** ± 0.54	63.74 ± 1.16	**95.35** ± 0.21	84.32 ± 0.29	51.20 ± 1.91	17.72 ± 0.95	39.03 ± 2.52
	LICEM	✗	✓	**90.64** ± 0.38	**81.85** ± 0.71	**66.15** ± 0.44	**95.79** ± 1.40	**85.46** ± 0.40	**77.47** ± 0.70	**72.98** ± 0.58	**86.66** ± 0.20
Self gen. (ours)	CBM+MLP	✗	✗	82.71 ± 0.01	75.42 ± 4.42	47.59 ± 1.37	82.31 ± 0.04	86.10 ± 0.00	46.80 ± 0.00	12.31 ± 0.07	7.68 ± 0.10
	CBM+XG	✗	✗	82.70 ± 1.23	79.09± ≤0.01	53.28± ≤0.01	82.28± ≤0.01	72.57 ± 0.00	38.40 ± 5.54	14.89 ± 0.00	3.28 ± 1.41
	CEM	✗	✗	**89.14** ± 0.38	**84.06** ± 0.09	**65.20** ± 0.73	**97.16** ± 0.08	**88.19** ± 0.99	**82.47** ± 1.15	**76.41** ± 0.40	**88.99** ± 0.23
	CBM+LL	✗	✓	82.71 ± 0.01	77.15 ± 0.96	47.35 ± 0.29	82.12 ± 0.15	84.95 ± 1.98	45.67 ± 0.12	12.40 ± 0.31	7.88 ± 0.10
	CBM+DT	✗	✓	83.95± ≤0.01	78.44± ≤0.01	53.28± ≤0.01	82.28± ≤0.01	86.10 ± 0.00	46.80 ± 0.00	12.53 ± 0.00	8.34 ± 0.00
	DCR	✗	✓	87.72 ± 0.66	83.47 ± 0.43	63.29 ± 0.36	97.11 ± 0.03	**88.83** ± 0.40	**82.20** ± 0.69	74.90 ± 0.63	88.55 ± 0.37
	LICEM	✗	✓	**89.56** ± 0.29	**84.49** ± 0.25	**65.89** ± 0.39	**97.23** ± 0.02	**88.44** ± 0.22	80.00 ± 1.06	**76.90** ± 0.40	**89.29** ± 0.25

We report in **bold** the best result among the same type of models (e.g., supervised, interpretable ones) considering models equally best if their standard deviations overlap. We use ✓to indicate models requiring concept supervision (C. S.) or having a task-interpretable predictor (T.I.). We highlight in light gray the models we propose in this work. The Self-Generative and Generative approaches extend the scalability of concept-based models to datasets without concept annotations, where supervised models cannot be applied (−)

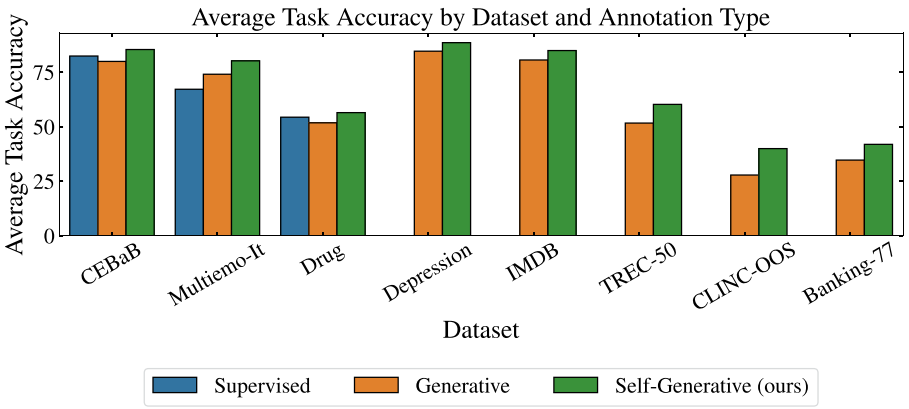


Fig. 3 Average task accuracy of concept-based models grouped by annotation type over the four datasets. The self-generative approach extends concept-based models applicability to scenarios without annotations (e.g., Depression) while improving their generalization

accuracies by annotation type. We can notice that the task accuracy of the self-generative approach is generally higher than both generative and supervised. This result is interesting because self-supervised models have fewer parameters to train than the corresponding generative ones. This behaviour is likely due to the higher concept accuracy of the self-generative approach (see Fig. 4) which also affects the resulting predictions.

We reported the performance of all concept-based baselines (not only Self-LICEM) when trained along the self generative approach. This was to showcase that any concept-based model can operate on a pretrained LLM without needing concept annotations.

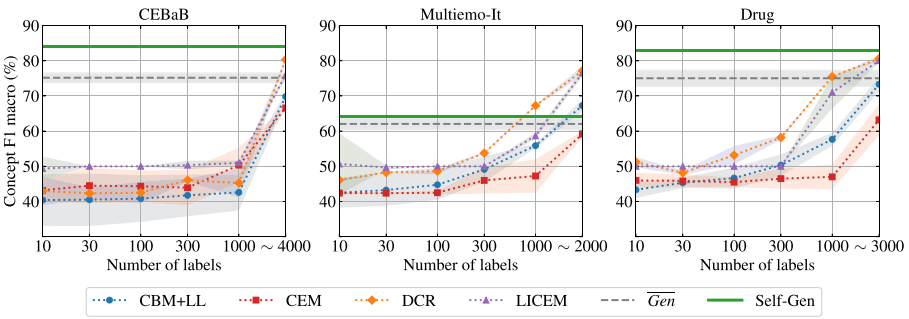


Fig. 4 Concepts prediction performance vs number of concept labels used during training. To increase plot readability, we only included the CBM+LL and the average F1 score for the generative approaches ($\overline{\text{Gen}}$). Self-generative and Gen. approaches are reported with a straight line, as they do not require concept annotation

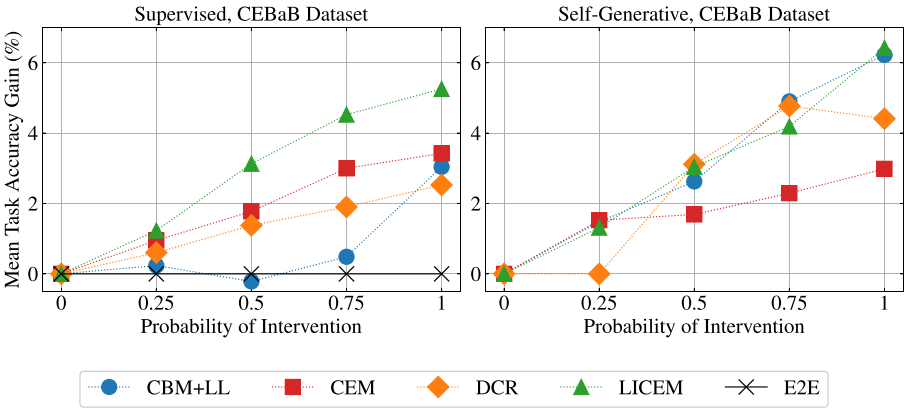


Fig. 5 Concept interventions on the CEBaB dataset for (left) supervised approaches and (right) self-supervised ones

5.3 LICEMs concept efficiency

Self-generative approach strongly reduces the human annotation effort (Fig. 4). To assess the efficiency in terms of concept labels required to properly train the different models, in Fig. 4 we report the concept prediction performance when increasing the number of concept labels used for training. Self-generative and generative approaches are reported with a straight line since they do not require any concept supervision[3]. Generative and self-generative models achieve a concept macro-averaged F1 score that is higher or close to that of supervised models when using all available annotations, and significantly higher otherwise. When considering the CEBaB and Drug datasets, supervised models do not surpass self-Gen even when using all concept annotations, with the latter achieving the highest concept accuracy. Likely, the amount of concept annotations required to match the accuracy of the self-generative approach exceeds what is available in these datasets.

The self-generative concept accuracy exceeds that of the generative approach (Fig. 4). The concepts prediction performance of the generative approach tends to be lower than that of the self-generative approach, with a reduction ranging from 2 to 7% in F1 macro score. This is because the concepts predicted by generative models are approximations of the self-generated concepts c' used in the self-generative approach. These self-generated concepts serve as the labels for training the concept encoders in the generative learning process. For a different visualization, we report the concept accuracy when provided with all samples for all models across the three dataset with concept annotation also in Table 6, Appendix A.5.

[3] Generative approaches results are reported with variance because the concepts are still learnt and thus the performance vary across models. For the self-generative approach, instead, the result does not vary because the concepts are predicted equally by the LLM for all models since we set the LLM's temperature to zero, which results in a deterministic annotation.

Table 2 Average area under the accuracy gain curve for each model-dataset combination, calculated using the composite trapezoidal rule

	Models	CEBaB	Multiemo	Drug
	E2E	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000
Superv.	CEM	1.858 ± 0.302	**10.111 ± 0.403**	2.474 ± 0.356
	CBM+LL	0.506 ± 0.522	0.973 ± 0.368	1.081 ± 0.765
	DCR	1.286 ± 0.465	4.546 ± 0.233	2.213 ± 0.262
	LICEM	**2.878 ± 0.363**	5.416 ± 0.408	**3.135 ± 0.251**
Self Gen.	CEM	1.749 ± 0.232	2.132 ± 0.111	1.725 ± 0.182
	CBM+LL	2.878 ± 0.082	4.826 ± 0.083	2.479 ± 0.281
	DCR	2.522 ± 0.019	3.299 ± 0.036	**4.090 ± 0.048**
	LICEM	**3.029 ± 0.47**	**5.416 ± 0.115**	3.136 ± 0.083

Average area under the accuracy gain curve for each model-dataset combination, calculated using the composite trapezoidal rule. The best-performing model, trained using either the supervised or self-generative approach, is highlighted in **bold**. Despite typically starting with higher accuracy, LICEM consistently improves its performance through interactions

Table 3 Causal Concept Effect (CaCE) for different methods. A high (absolute) value implies a strong responsiveness of a model to modifications to a certain concept

	Concept	CBM+LL	CEM	DCR	LICEM	SELF-LICEM
CeBAB	Good Food	-0.02 ± 0.01	0.29 ± 0.03	0.33 ±0.04	0.62 ± 0.02	0.63 ± 0.01
	Good Amb.	0.01 ± 0.05	0.08 ± 0.01	0.02 ±0.01	0.18 ± 0.03	0.20 ± 0.04
	Good Service	0.01 ± 0.04	0.13 ± 0.01	0.20 ±0.08	0.37 ± 0.01	0.35 ± 0.02
	Good Noise	-0.01 ± 0.10	-0.05 ± 0.01	-0.02 ±0.01	0.15 ± 0.02	0.15 ± 0.03
Multiemo	Joy	0.04 ± 0.06	0.18 ± 0.01	0.16 ± 0.07	0.28 ± 0.01	0.27 ± 0.01
	Trust	0.02 ± 0.10	0.60 ± 0.04	0.47 ± 0.15	0.62 ± 0.03	0.63 ± 0.01
	Sadness	-0.04 ± 0.05	-0.06 ± 0.01	-0.04 ± 0.02	-0.04 ± 0.01	-0.10 ± 0.02
	Surprise	-0.01 ± 0.06	0.03 ± 0.01	0.06 ± 0.05	-0.02 ± 0.01	0.01 ± 0.01
Drug	Effectiveness	0.02 ±0.10	0.43 ±0.02	0.28 ±0.02	0.45 ±0.04	0.46 ± 0.02
	Side Effects	-0.07 ±0.14	-0.52 ±0.01	-0.25 ±0.02	-0.55 ±0.06	-0.55 ± 0.03

5.4 LICEM interpretability

LICEM is responsive to concept interventions (Fig. 5, Table 2). A fundamental interpretability property of CBMs is their *intervenability*, i.e., the possibility to modify the concept predictions in order to correct the model or assess potential counterfactual predictions. As commonly done in CBM literature Koh et al. (2020); Espinosa Zarlenga et al. (2022); Kim et al. (2023) we simulate this scenario by randomly replacing the concept predictions with the concept labels with increasing probability. Figure 5 shows the test task accuracy gain with increasing intervention probability on the CEBaB dataset, demonstrating LICEM's responsiveness and significant performance improvement. A similar behaviour can also be observed for CBMs, even though they were starting from a lower task accuracy and a higher increase could have also been expected. For comparison, we also report the E2E model with a flat line, since it does not offer this possibility. Concept intervention figures for all datasets are reported in Appendix A.6 showing similar results. In Table 2, we summarize the performance of every model across all datasets, reporting the area under the accuracy gain curve for each model-dataset combination, calculated using the composite

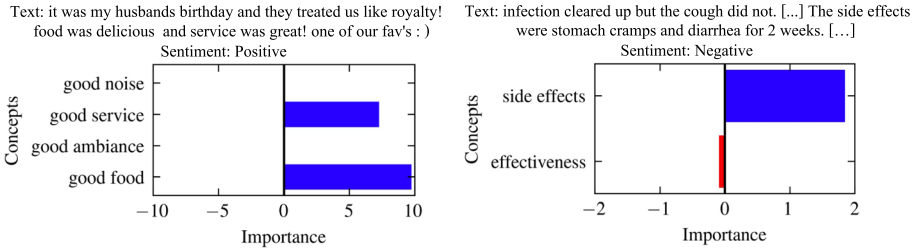


Fig. 6 Explanations generated by LICEM for the predicted classes on two datasets: CEBaB (left) and Drug (right). The x-axis shows the importance scores, computed as $\hat{w}_{pj}\hat{c}_j$, where p indicates the index of the class predicted by LICEM, and j represents the concept index. The y-axis displays the concept names. Above the LICEM explanations, the original text and the corresponding LICEM predictions are shown. For these samples, LICEM's predictions aligned with the ground-truth values

trapezoidal rule. The results demonstrate that, although LICEM typically starting with higher accuracy, it consistently improves its task accuracy through interactions. In four out of six scenarios, LICEM stands out as the most responsive model. When considering only task-interpretable models, LICEM is the most responsive in five cases, being surpassed by DCR only on the Drug dataset using the self-generative approach.

LICEM predictions are caused by most important concepts (Table 3). In order to globally explain the task prediction of the compared models, we assess the effect to *do-interventions* over concepts (Pearl et al., 2016), by computing the Causal Concept Effect (CaCE) (Goyal et al., 2019). CaCE measures the impact of modifying input samples on model predictions. For concept-based models, interventions can be made at the concept level (Dominici et al., 2024). In the evaluated dataset, several concepts are globally relevant for the classification task (positively or negatively), thus we expect models to exhibit high absolute CaCE values for those values. In Table 3, we present the results across all annotated datasets. For the CEBaB dataset, the concepts of 'Food' and 'Service' emerge as the most crucial, while 'Joy' and 'Trust' hold more importance in the Multiemo dataset. On average, both LICEM's CaCE values are higher or on par with those of CEM, and consistently surpass those of DCR and CBM. For models using concept-embedding models (CEM, DCR), these results indicate that LICEM correctly relies more on the concept scores rather than on the concept embeddings for prediction. Thus, LICEM should be less affected by concept leakage issues (Marconato et al., 2022), and thus results more interpretable. In contrast, the low CaCE values for CBM+LL indicate a poor understanding of the task and confirms the underfitting issues outlined in Sect. 5.2, likely due to the concept bottleneck representation.

LICEM predictions can be directly interpreted (Fig. 6). To disclose the decision-making process underlying the prediction of a LICEM, we plot the logits $\hat{w}_{pj}\hat{c}_{pj}$ calculated by the model for the predicted class p and the various concepts j. This visualization enables users to assess the importance of concept j in the classification of the predicted class. Figure 6 illustrates the explanations generated by LICEM for two samples: one from the dataset CEBaB (left) and one from Drug (right). In the CEBaB example, *good noise* and *good ambiance* did not influence the *Positive* sentiment prediction due to their absence ($\hat{c}_{pj} = 0$). In contrast, *good service* and *good food* positively impacted the prediction, with *good food* being most significant, reflecting its importance in restaurant evaluations. The Drug explanation depicts a review where the medication provided minor relief from the condition

(*infection cleared up but the cough did not*) but caused severe *side effects* (*stomach cramps and diarrhea for 2 weeks*). LICEM predicts the sentiment of the review as *negative* and identifies the presence of both *side effects* and *effectiveness*. Since the explanation shown corresponds to the negative sentiment class, it highlights that the weight associated with *side effects* is positive and substantially larger in magnitude than the weight associated with *effectiveness*. Additional explanations are reported in Appendix A.7.

LICEM Explanations Align Better with Human Intuition (Fig. 7).

To evaluate the interpretability of LICEM explanations, we conducted a user study comprising 7 questions and involving 46 participants, consisting of both machine learning experts and non-experts (see Fig. 20). We compared LICEM's explanations against those generated by DCR, the strongest task-interpretable baseline. In the first task, participants are asked to choose the most plausible explanation (Rajagopal et al., 2021) from three options: the LICEM explanation, the DCR explanation, or neither. The explanations are extracted by randomly sampling the CEBaB dataset. In the second task, we evaluated the usefulness of explanations by measuring the participant's ability to infer the prediction of the model from the explanation provided (Fel et al., 2023). Examples of the two types of questions are shown in Figs. 21 and 22. Additional explanations are shown in Appendix A.7.

The left graph of Fig. 7 presents the results related to explanation plausibility. It is evident that the LICEM explanation is consistently considered more plausible over the rule-based DCR explanation by both expert and non-expert users. Contrary to our expectations, LICEM was especially favored by expert users, with nearly 80% of them appreciating its explanations. The graph on the right of Fig. 7 illustrates the accuracy achieved by the users when tasked with selecting a class label based on a given explanation. Both groups of users demonstrated good accuracy when making classifications using the LICEM explanations.

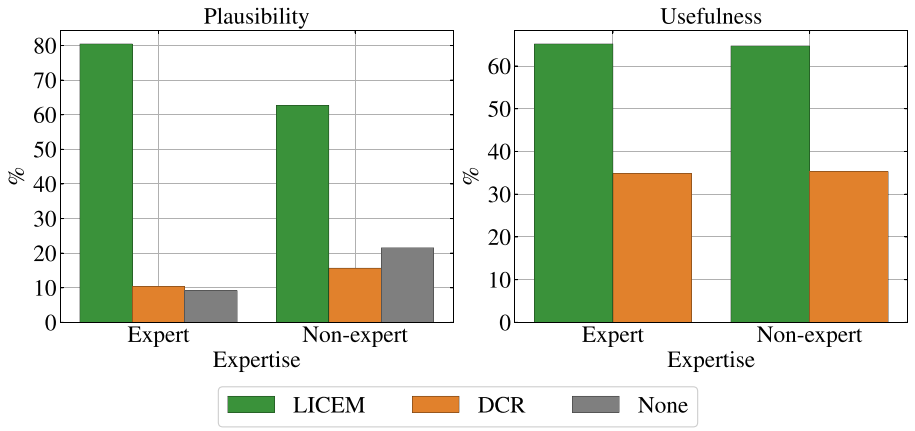


Fig. 7 Averaged survey results for the two user groups. Left: explanation plausibility; Right: prediction accuracy based on explanations

6 Conclusion

In this paper, we propose LICEM, a novel linearly interpretable concept-based model for text analysis. The experimental results show this model matches black-box models performance, is interpretable and can be trained without concept supervision (Self-LICEM). Besides a technological impact, we believe this work can also positively impact the society by enhancing LLM transparency and interpretability, thus facilitating their employment in several fields such as Healthcare, Finance, Legal Systems, and Policy Compliance. As an example, in the latter case LICEM could be used to classify social media contents. LICEM would not only identify harmful posts but would also explain its decisions using concepts (e.g., hate speech, harassment) and how they contribute to the final prediction, helping moderators act quickly and transparently.

Limitations. While LICEM improves interpretability in text classification, it still face a few challenges. First, in scenarios with a high number of concepts or fine-grained classes, the resulting linear explanations may induce cognitive overload, reducing their practical usefulness. Second, while its linear form enhances transparency, it may fail to represent complex interactions between concepts that influence task outcomes. Third, LICEM's self-generative approach hinges on the quality of concept extraction by the underlying LLM, which can vary across domains and be sensitive to prompt phrasing. Finally, even with regularization, model predictions can be biased in inputs lacking clear concept signals, especially under ambiguity or domain drift.

Future work. In this analysis, we focused on binary or ternary sentiment analysis for the ease of identifying concepts, and to texts composed of a few sentences. In future work, we will extend our analysis to other NLP tasks and to longer texts, to ensure the scalability of this approach. Furthermore, we plan to extend the capability of this model to work in language modelling tasks, similarly to Ismail et al. (2023) employing CBMs to solve generative tasks in computer vision. We leave these investigations for future research.

7 Supplementary information

This article includes a technical appendix with supplementary details on the methodology and experimental campaign, which are helpful but not essential to the main content.

Appendix A

A.1: Prompts for annotation

Here we report the prompts used to instruct *Mistral 7B* and *Mixtral 8x7B* to perform the annotations on the four different datasets used in this work. We adopted the in-context instruction learning prompting strategy (Ye et al., 2023).

CEBaB

```
In a dataset of restaurant reviews there are 4 possible concepts: Good Food,
Good Ambiance, Good Service and Good Noise. Given a certain review, you have
to detect if those concepts are present or not in the review.

Answer format: Good Food:score,Good Ambiance:score, Good Service:score,
Good Noise:score.

Do not add any text other than that specified by the answer format.
The score should be equal to 1 if the concept is present or zero otherwise,
no other values are accepted.

The following are examples:

Review: "The food was delicious and the service fantastic".
Answer: Good Food:1, Good Ambiance:0, Good Service:1, Good Noise:0

Review: "The staff was very rough but the restaurant decorations were great.
Other than that there was a very
relaxing background music".
Answer: Good Food:0, Good Ambiance:1, Good Service:0, Good Noise:1

Now it's your turn:

Review: <review>
Answer:
```

Drug

```
In a dataset of drug reviews there are 2 possible concepts:

- Effectiveness: 1 if the drug was highly effective and 0 if it was
marginally or not effective,
- Side effects: 1 if the drug gave side effects and 0 otherwise.

Given a certain review, you have to detect if those concepts are present or
not in the review.

Answer format: Effectveness:score, Side effects:score.

Do not add any text other than that specified by the answer format.
The score should be equal to 1 if the concept is present or zero otherwise,
no other values are accepted.

The following are examples:

Review: "The medicine worked wonders for me. However, I did experience some
side effects. Despite this, I still found it easy to use and incredibly
effective".
Answer: Effectiveness:1, Side effects:1

Review: "Not only it did fail to alleviate my symptoms, but it also led to
unpleasant side effects".
Answer: Effectiveness:0, Side effects:1

Now it's your turn:

Review: <review>
Answer:
```

Multiemo-it

```
In a dataset containing comments in Italian, you need to identify the
following concepts:

-Joy: the user who wrote the comment expresses joy,
-Trust: the user who wrote the comment expresses trust,
-Sadness: the user who wrote the comment expresses sadness,
-Surprise: the user who wrote the comment is surprised.

Response format: Joy:score, Trust:score, Sadness:score, Surprise:score.

The score must be equal to 1 if the concept is present and 0 otherwise;
other values are not accepted.

The following is an example:
Comment: "Mi piace la rivisitazione di questa canzone, dolce, raffinata,
elegante, bellissima!"
Answer: Joy:1, Trust:1, Sadness:0, Surprise:1

Now it's your turn:
Comment: <comment>
Answer:
```

Depression

```
You have to identify the presence or absence of 6 concepts in a given text.
The concepts to be identified are:

- Self-Deprecation: the text exhibits self-critical or self-deprecating
language, expressing feelings
of guilt, shame, or inadequacy.
- Loss of Interest: diminished pleasure or motivation in the writer's
descriptions of hobbies or pursuits.
- Hopelessness: the writer express feelings of futility or a lack of
optimism about their prospects.
- Sleep Disturbances: the writer mentions insomnia, oversleeping, or
disrupted sleep as part of their
experience.
- Appetite Changes: there are references to changes in eating habits.
- Fatigue: there are references to exhaustion or lethargy.

Answer format: Self-Deprecation:score, Loss of Interest:score,
Hopelessness:score, Sleep Disturbances:score, Appetite Changes:score,
Fatigue:score.

The score has to be 1 if the concept is detected and 0 otherwise. Do not add
any other text besides the one specified in the answer format.

Text: <text>
Answer:
```

IMDb

```
In a dataset of film reviews (IMDb), there are 4 possible concepts:

- Good Acting,
- Good Storyline,
- Good Emotional Arousal,
- Good Cinematography.

Given a certain review, you have to detect if those concepts are
present or not in the review.

Answer format:
Good Acting: score, Good Storyline: score, Good Emotional Arousal: score,
Good Cinematography: score.

Do not add any text other than that specified by the answer format.
The score should be equal to 1 if the concept is present and zero if ,
no other values are accepted.

The following are examples:

Review: "The performances were outstanding, especially the lead actor.
The story dragged in the middle though."
Answer: Good Acting: 1, Good Storyline: 0, Good Emotional Arousal: 0,
Good Cinematography: 0

Review: "This film moved me to tears. The plot was very touching,
and the visual effects were just stunning."
Answer: Good Acting: 0, Good Storyline: 1, Good Emotional Arousal: 1,
Good Cinematography: 1

Now it's your turn:

Review: <review>
Answer:
```

TREC-50

```
In a dataset of questions, there are 6 possible concepts:

- Definition Request,
- Person Entity,
- Location Reference,
- Numeric Expectation,
- Abbreviation or Acronym,
- Object Reference.

Given a certain question, you have to detect if those concepts
are present or not in the question.

Answer format:
Definition Request: score, Person Entity: score,
Location Reference: score, Numeric Expectation: score,
Abbreviation or Acronym: score, Object Reference: score.

Do not add any text other than that specified by the answer format.
The score should be equal to 1 if the concept is present
or zero otherwise, no other values are accepted.

The following are examples:

Question: "What is the capital of France?"
Answer: Definition Request: 0, Person Entity: 0,
Location Reference: 1, Numeric Expectation: 0,
Abbreviation or Acronym: 0, Object Reference: 0

Question: "Who discovered penicillin?"
Answer: Definition Request: 0, Person Entity: 1,
Location Reference: 0, Numeric Expectation: 0,
Abbreviation or Acronym: 0, Object Reference: 0

Now it's your turn:

Question: <review>
Answer:
```

CLINC-OOS

```
You are given a user query to a task-oriented dialog system.
The system supports multiple domains and intents,
but some queries may be out-of-scope (OOS),
meaning they do not fall into any supported intent.

Your task is to detect the presence or absence of the following
concepts in the query. For each concept,
answer with a score of 1 if the concept is present, or 0 if it is absent.
Do not add any text other than the answer format.

Concepts:
- Domain Mention: Does the query explicitly mention
or imply a supported domain or topic?
- Intent Specific Keywords: Does the query contain keywords
or phrases related to any specific intent?
- Action Request: Does the query ask to perform an action or service?
- Out-of-Scope Indicators: Does the query contain terms
or topics unrelated to any supported domain or intent,
indicating it is out-of-scope?

Answer format:
Domain Mention: score, Intent Specific Keywords: score,
Action Request: score, Out-of-Scope Indicators: score

Examples:

Query: "Can you help me book a flight to New York?"
Answer: Domain Mention: 1, Intent Specific Keywords: 1,
Action Request: 1, Out-of-Scope Indicators: 0

Query: "What's the capital of France?"
Answer: Domain Mention: 0, Intent Specific Keywords: 0,
Action Request: 0, Out-of-Scope Indicators: 1

Now it's your turn:

Query: <review>
Answer:
```

Banking-77

```
In a dataset of user queries related to banking and financial services,
there are 4 possible concepts:

- Transaction Mention
- Issue/Problem Description
- Account Reference
- Request for Help or Clarification

Given a user query, you have to detect if each of these concepts
is present or not in the query.

Answer format:
Transaction Mention: score, Issue/Problem Description: score,
Account Reference: score, Request for Help or Clarification: score.

Do not add any text other than that specified by the answer format.
The score should be 1 if the concept is present or 0 otherwise.
No other values are accepted.

The following are examples:

Query: "A card payment on my account is shown as pending."
Answer: Transaction Mention: 1, Issue/Problem Description: 1,
Account Reference: 1, Request for Help or Clarification: 0

Query: "I can't seem to make a standard bank transfer.
I have tried at least five times already but none of them are going through.
Please tell me what is wrong?"
Answer: Transaction Mention: 1, Issue/Problem Description: 1,
Account Reference: 0, Request for Help or Clarification: 1

Now it's your turn:

Query: <review>
Answer:
```

A.2: Experimental details

For the E2E, CBMs, CEM, DCR and LICEM models, the training process involved utilizing an AdamW optimizer (Loshchilov & Hutter, 2017). The λ_y coefficient (2) was set to 0.5 to emphasize concept learning over task loss while $\lambda_w = 10^{-6}$ and $\lambda_b = 10^{-6}$. Moreover, a scheduler was implemented with a gamma of 0.1 and a step size of 10 epochs was employed during the training process, spanning 100 epochs when using BERT as the backbone and 50 epochs when utilizing Mixtral 8x7B. After every hidden layer we have used a ReLU activation function. Here are further insights into the methodologies' architectures, with the number of output neurons indicated within brackets.

- E2E: layer 1 (100), layer 2 (number of classes);
- CEM: concept embedding size of 768, layer 1 (10), layer 2 (number of classes);
- CBMs, concept prediction: layer 1 (10), layer 2 (number of concepts);
  - LL, task prediciton: layer (number of classes);
  - MLP, task prediction: layer 1 ($3 \cdot$ number of concepts), layer 2 (number of classes).

- DCR: the temperature parameter is set to 0.1.

The text's embedding size varies depending on the chosen backbone. When employing BERT, it remains at 768, whereas adopting the LLM approach (Jiang et al., 2023) it increases to 4096. For Dtree and XGBoost, we employed the default hyperparameter settings. The DTree model was implemented using the sklearn library, while the XGBoost model was implemented using the xgboost library.[4] We conducted five experiments for each methodology. The training time for the different experiments averages around 10 min using the setup specified in Section 5.1.

Since concept-annotated datasets were discussed in Section 5.1, we focus here on the remaining datasets. **Depression** contains Reddit posts from users and the goal is to classify a post as depressed or not. As concept annotations are unavailable, we used an LLM (Jiang et al., 2024) to identify six relevant concepts: *Self-deprecation*, *Loss of Interest*, *Hopelessness*, *Sleep Disturbances*, *Appetite Changes*, and *Fatigue*. **IMDb** includes 50,000 movie reviews labeled as positive or negative for sentiment analysis. The LLM identified: *Acting*, *Storyline*, *Emotional Arousal*, and *Cinematography*. **TREC-50** comprises open-domain questions classified into 50 fine-grained types. Relevant concepts identified: *Definition Request*, *Person Entity*, *Location Reference*, *Numeric Expectation*, *Abbreviation or Acronym*, and *Object Reference*. **Banking-77** features real-world banking queries labeled with 77 intent categories. The LLM-generated concepts are: *Transaction Mention*, *Issue/Problem Description*, *Account Reference*, and *Request for Help or Clarification*. **CLINC-OOS** contains 151 in-domain intents across ten topics and one out-of-scope class, used for open-domain intent classification. The associated concepts are *Domain Mention*, *Intent-Specific Keywords*, *Action Request*, and *Out-of-Scope Indicators*. All datasets included predefined training and test splits. Additionally, **IMDb** and **CEBaB** provided validation sets. For the remaining datasets, we split the training such that $\frac{1}{8}$ of the data is used as validation.

A.3: Encoder comparison

This section presents all the results obtained using a fine-tuned BERT backbone as the encoder $h(x)$. In the remainder of the paper, we consistently reported results when utilizing Mixtral 8x7B (Jiang et al., 2024) as the backbone model. The total number of trainable parameters remains relatively modest: approximately 100K when using Mixtral as the backbone (without fine-tuning), and around 10 M when using BERT as the backbone. In this section, we provide the performance of all models in terms of task accuracy (see Table 4) and of concept macro-averaged F1 score (refer to Table 5) when employing BERT as the backbone (Kenton & Toutanova, 2019), which is an encoder-only model.

Both tables show that there is no great difference with respect to Tables 1, 6, with BERT providing slightly lower performance on Multiemo-It and on the Drug dataset. This result shows that the proposed approach can be applied also to other architectures. We chose to employ Mixtral in the remainder of the paper since it can be also effectively used to provide concept annotations (thus enabling the Self-LICEM strategy), therefore having a single model for both encoding the sample and predicting the concept predictions.

[4] The xgboost library we used can be found at https://github.com/dmlc/xgboost.

A.4: LLM-based concept annotation vs class-level annotation

This section presents a comparison between the usage of two different LLMs, Mistral 7B (Jiang et al., 2023) and Mixtral 8x7B (Jiang et al., 2024), as concept annotators. In Fig. 8 we report the results in terms of macro-averaged F1 score (as concept classes are highly imbalanced) on the three datasets for which human concept annotation is available. We also report, as a baseline, a global (class-level) annotation strategy, providing to all samples belonging to a given class the same concept annotation. In this case, we label the positive class with positive concepts and negated negative concepts (e.g. for all samples of the class *Good Drug* we use 'Efficient' and 'Not Side Effects'). We can observe that between the two LLMs there is not a significant difference in performance, with Mixtral 8x7B providing on average slightly better results. Comparing against the baseline, instead, we can observe that there is a great improvement in CEBaB and in the Drug dataset, while in Multiemo-It the improvement is more modest.

A.5: Concepts prediction performance

In this section we report in Table 6 the averaged F1 macro to measure the concepts prediction performance of all models when provided with all the available concept annotations for all the different experiments conducted, generative approach included, when using Mixtral 8x7B as a backbone. The results shown in Fig. 4 are here confirmed. We again see that Self-supervised strategy is a very good approach since without human effort it provides better concept macro-averaged F1 score in CEBaB and Drug. Only on Multiemo-It the performance are significantly lower. This result may be due to the fact that the latter dataset is in Italian while the other datasets are in English, a language for which the LLMs have certainly seen more training samples.

A.6: Concept interventions

As introduced in Section 5.4, LICEM is sensible to concept interventions. This characteristic is very important since it implies that a human can interact with the model, providing counterfactual predictions when prompted with different concept predictions. In Fig. 9, 10, 11 we simulate this situation by correcting mispredicted concepts with the correct concept predictions and check whether the task prediction has been also modified. More in details, we report the improvement in task accuracy when increasing the probability to correct the concepts, demonstrating LICEM's responsiveness and significant performance improvement. A similar behaviour can also be observed for CBMs, even though they were starting from a lower task accuracy and a higher increase could have also been expected. For comparison, we also report the E2E model with a flat line, since it does not offer this possibility. As noted in (Espinosa Zarlenga et al., 2022), CEMs (which are not task interpretable) may not respond well to concept interventions, especially without conducting them during training. Thus, we trained all CEM-based models with a 0.5 intervention probability during the forward pass.

Table 4 This table presents the performance in terms of task accuracy (%) of different models utilizing BERT as backbone. We report in **bold** the best result among the same type of models (e.g., supervised, interpretable ones) considering models equally best if their standard deviations overlap. We use ✓to indicate models requiring concept supervision (C. Sup.) or having an interpretable task predictor (T. Inter.). We highlight in light gray the models we propose in this work. We do not report supervised model results for depression (−) since it does not provide concept annotations

Type	Method	C. S	T. I	CEBaB	Multiemo-It	Drug	Depression
E2E	MLP	✗	✗	**90.68** ± 0.47	**75.67** ± 0.47	**59.33** ± 0.56	**97.80** ± 0.23
SUP.	CBM+MLP	✓	✗	78.01 ± 6.51	54.10 ± 4.51	36.67 ± 6.24	–
	CBM+XG	✓	✗	80.00 ± 0.34	69.02 ± 0.64	51.00 ± 0.28	–
	CEM	✓	✗	**90.67** ± 0.47	**77.00** ± 0.82	**58.33** ± 1.70	–
	CBM+LL	✓	✓	61.00 ± 12.02	49.67 ± 5.46	34.33 ± 7.38	–
	CBM+DT	✓	✓	75.67 ± 0.47	65.02 ± 0.34	46.23 ± 0.78	–
	DCR	✓	✓	86.55 ± 0.58	74.01 ± 0.24	**59.75** ± 0.45	–
	LICEM (ours)	✓	✓	**87.89** ± 0.38	**75.31** ± 0.15	**60.14** ± 0.44	–
GEN.	CBM+MLP	✗	✗	73.93 ± 5.67	44.19 ± 2.07	35.16 ± 4.3	83.20 ± 2.18
	CBM+XG	✗	✗	83.29 ± 0.43	69.85 ± 1.55	34.94 ± 0.91	87.00 ± 1.01
	CEM	✗	✗	**85.88** ± 0.95	**73.15** ± 0.67	**56.95** ± 0.36	**96.12** ± 0.50
	CBM+LL	✗	✓	58.81 ± 7.16	58.35 ± 1.59	36.84 ± 11.52	51.48 ± 2.16
	CBM+DT	✗	✓	79.28 ± 0.52	62.61 ± 2.08	34.17 ± 0.11	80.55 ± 0.03
	DCR	✗	✓	**85.63** ± 0.81	70.02 ± 2.70	57.46 ± 0.02	95.98 ± 0.27
	LICEM (ours)	✗	✓	**86.22** ± 0.66	**74.45** ± 0.57	**60.23** ± 0.58	**96.87** ± 0.20

A.7: Additional visualizations of explanations

This appendix provides additional LICEM explanations. Although the main text includes a representative set of explanations to support the core findings, the materials presented here offer a broader view of the model's interpretability across different datasets. From Figs. 12, 13, 14, 15, 16, 17, 18, and 19, the concepts on the y-axis are ordered in ascending order. This means that the most important concepts-those with the highest importance-are displayed at the bottom of the y-axis, with decreasing importance as you move upward.

A.8: User-study characterization

In this section, we provide further details regarding the conducted survey. A total of 46 participants with varying levels of experience in machine learning, from complete beginners to experts, were recruited (see Fig. 20). The gender distribution was nearly balanced, with 40% identifying as female and 60% as male. The majority of participants, 91.3%, were within the 20 – 40 age range, while only 8.7% were aged over 40.

A.9: Critical difference diagram

The Critical Difference (CD) diagram was computed by considering all the models across all datasets for both the generative and self-generative approaches. The supervised approach was not included, as the depression dataset was unavailable (no concept

Table 5 This table presents the performance in terms of concept prediction of the models that utilize BERT as backbone

Type	Method	CEBaB	Multiemo-It	Drug
E2E	MLP	**79.92** ± 1.77	**63.25** ± 1.09	**79.01** ± 2.9
SUP.	CBM+MLP	75.17 ± 3.11	**64.08** ± 1.22	74.26 ± 0.9
	CEM	**79.97** ± 1.29	**64.42** ± 1.21	**77.32** ± 1.2
	CBM+XG	**79.92** ± 1.77	**63.25** ± 1.09	**79.01** ± 0.9
	CBM+LL	74.25 ± 4.55	62.08 ± 0.88	73.11 ± 1.7
	CBM+DT	79.92 ± 1.77	63.25 ± 1.09	79.01 ± 2.9
	DCR	82.06 ± 0.40	64.29 ± 0.42	80.10 ± 0.2
	LICEM (ours)	**82.93** ± 0.13	**65.61** ± 0.69	**81.59** ± 0.42
GEN.	CBM+MLP	75.05 ± 8.31	49.59 ± 10.01	43.58 ± 14.99
	CEM	**81.08** ± 0.44	**58.30** ± 1.79	**80.99** ± 0.42
	CBM+XG	79.24 ± 1.21	**60.79** ± 0.71	64.72 ± 0.45
	CBM+LL	**78.75** ± 0.59	**61.72** ± 0.24	66.72 ± 19.48
	CBM+DT	**79.24** ± 1.21	**60.79** ± 0.70	64.72 ± 0.45
	DCR	**80.25** ± 1.02	59.11 ± 0.84	**81.47** ± 0.49
	LICEM (ours)	**77.79** ± 2.49	58.87 ± 0.66	**81.18** ± 0.33
SELF GEN.	–	**84.08** ± 0.00	**64.27** ± 0.00	**83.00** ± 0.00

Concept prediction (%) of the compared models for datasets equipped with concept annotations is measured using the macro-averaged F1 score. We report in **bold** the best result among the same type of models (e.g., supervised, interpretable ones) considering models equally best if their standard deviations overlap. We highlight in light gray the models we propose in this work. The methods using the self-generative have the same macro-averaged F1 score, therefore we use – to represent all methods

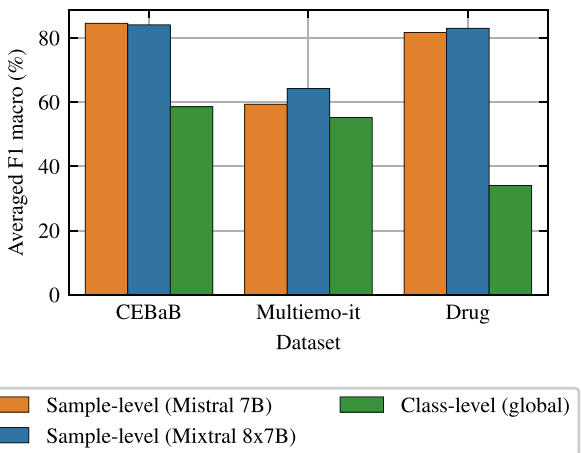


Fig. 8 Comparison among concept annotation methods where the annotation quality is measured in terms of macro-averaged F1 score. On average, Mixtral 8x7B yields the best results

supervision). However, even in the supervised scenario, LICEM and CEM remain the top-performing models.

The results of the CD diagram (Figure 23) indicate that LICEM is the top performer, consistently achieving first or second place across all datasets. However, the performance

Table 6 This table presents the performance in terms of concept prediction of the models that utilize Mixtral 8 × 7B as backbone

Type	Method	CEBaB	Multiemo-It	Drug
E2E	MLP	**75.92** ± 0.77	**74.25** ± 1.02	**78.50** ± 0.23
SUP.	CBM+MLP	65.17 ± 2.35	61.75 ± 1.02	65.33 ± 2.46
	CEM	**78.83** ± 0.85	**77.12** ± 1.38	**80.79** ± 0.47
	CBM+XG	75.92 ± 0.77	74.25 ± 1.02	78.50 ± 0.23
	CBM+LL	64.25 ± 2.56	59.12 ± 2.13	64.83 ± 1.20
	CBM+DT	**75.92** ± 0.77	74.25 ± 1.02	78.50 ± 0.23
	DCR	**78.45** ± 1.92	**75.67** ± 1.43	79.96 ± 0.43
	LICEM (ours)	75.45 ± 0.93	**76.36** ± 0.39	**80.83** ± 0.36
GEN.	CBM+MLP	71.87 ± 0.14	52.60 ± 14.32	55.68 ± 19.84
	CEM	**74.70** ± 0.98	**63.61** ± 0.44	**79.45** ± 0.41
	CBM+XG	**75.02** ± 0.57	61.69 ± 0.44	**79.15** ± 0.30
	CBM+LL	72.15 ± 0.59	**63.72** ± 0.84	66.72 ± 19.48
	CBM+DT	**75.02** ± 0.57	61.69 ± 0.44	**79.04** ± 0.30
	DCR	**75.62** ± 2.59	62.79 ± 0.44	**79.04** ± 0.33
	LICEM (ours)	**74.44** ± 0.25	**63.75** ± 0.36	**79.05** ± 0.58
SELF GEN.	–	**84.08** ± 0.00	**64.27** ± 0.00	**83.00** ± 0.00

Concept prediction (%) of the compared models for datasets equipped with concept annotations is measured using the macro-averaged F1 score. We report in **bold** the best result among the same type of models (e.g., supervised, interpretable ones) considering models equally best if their standard deviations overlap. Self-supervised methods are reported with the same concept accuracy with zero standard deviation, since the concept predictions are provided by an LLM with temperature set to zero. The methods using the self-generative have the same macro-averaged F1 score, therefore we use – to represent all methods

difference between LICEM and CEM is not statistically significant, nor is there a notable distinction between the CBM-like models (CBM+MLP, CBM+XG, and CBM+LL). DCR ranks in the middle tier with an average position of 3.0, demonstrating strong performance, although it does not outperform LICEM or CEM. Both CBM+XG and CBM+MLP also rank in the middle, while CBM+LL consistently ranks the lowest across all datasets.

A.10: Concept sparsity

In this section, we provide a quantitative analysis of how closely the predicted concepts sparsity of each model matches the intrinsic sparsity of the concepts in the data. For each sample, we compute the absolute difference between the number of predicted active concepts and the number of active concepts in the ground truth:

$$\delta_i = \left| \sum_j \hat{c}_{ij} - \sum_j c_{ij} \right|,$$

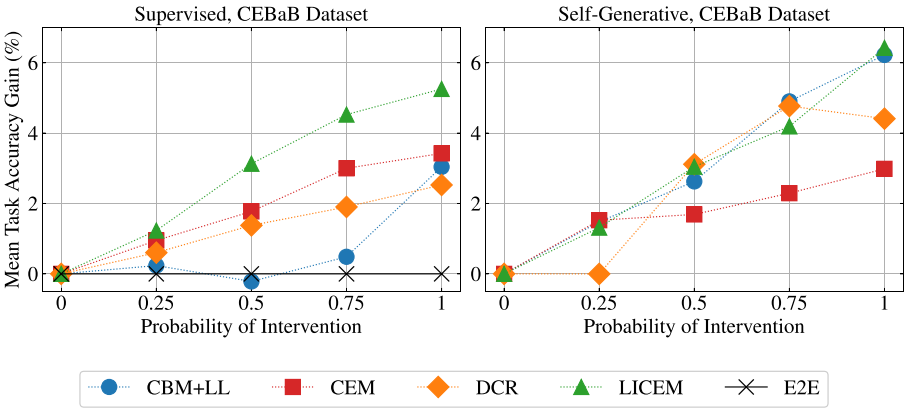


Fig. 9 Concept interventions on the CEBaB dataset for (left) supervised approaches and (right) self-supervised ones

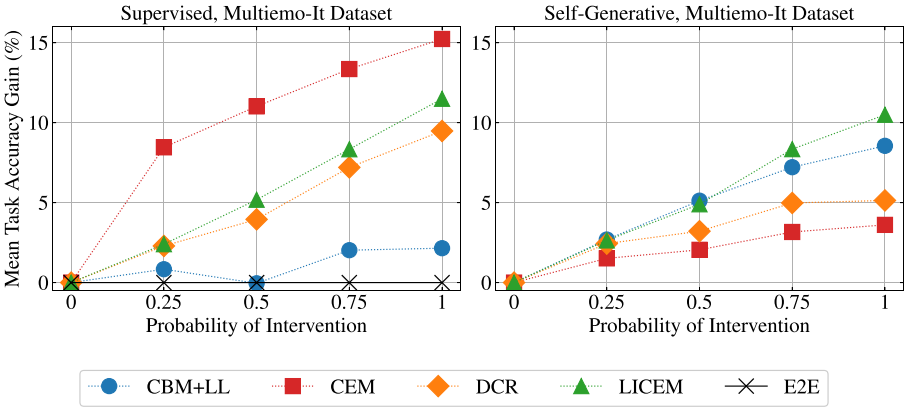


Fig. 10 Concept interventions on the Multiemo-it dataset for (left) supervised approaches and (right) self-supervised ones

where $\hat{c}_{ij} \in \{0, 1\}$ denotes whether concept j was predicted as active in sample i, and $c_{ij} \in \{0, 1\}$ is the corresponding ground-truth label. The average deviation across all samples provides a direct measure of this alignment, with lower values indicating better correspondence. Table 7 reports the results across all datasets.

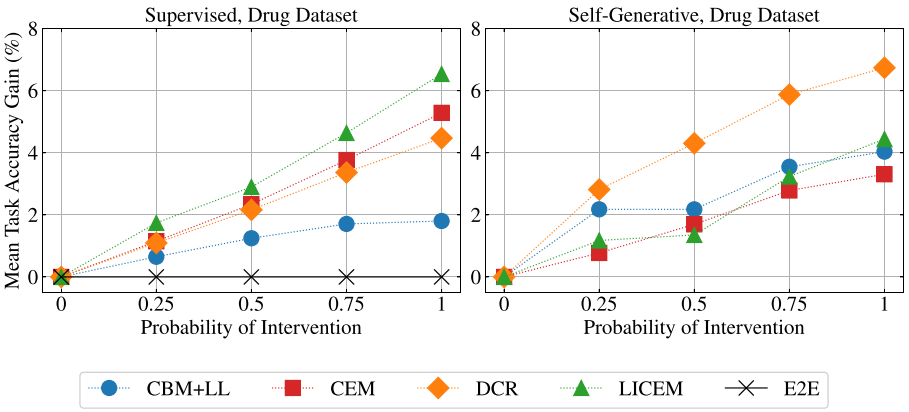


Fig. 11 Concept interventions on the Drug dataset for (left) supervised approaches and (right) self-supervised ones

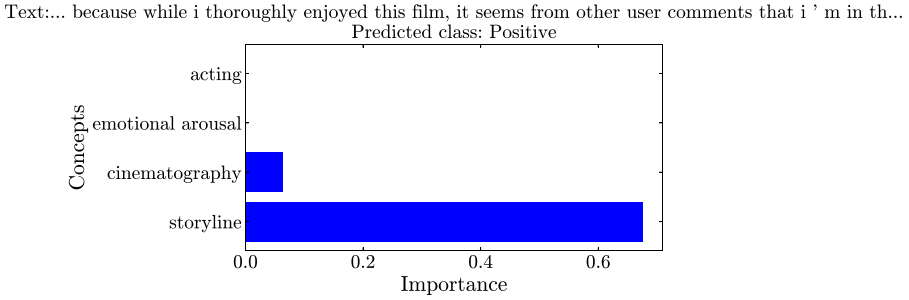


Fig. 12 Explanation 1 from the IMDB dataset

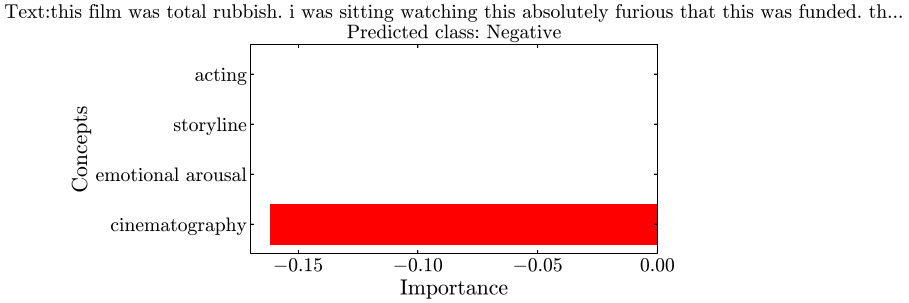


Fig. 13 Explanation 2 from the IMDB dataset

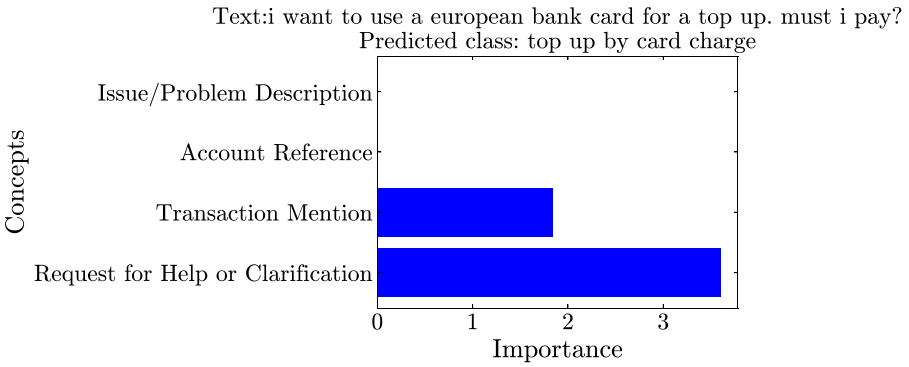


Fig. 14 Explanation 1 from the Banking-77 dataset

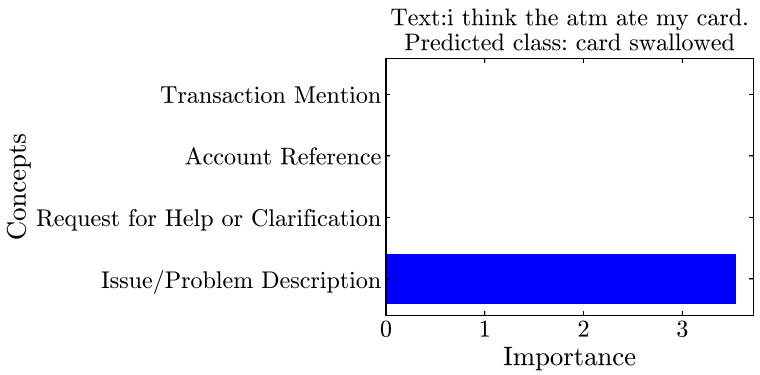


Fig. 15 Explanation 2 from the Banking-77 dataset

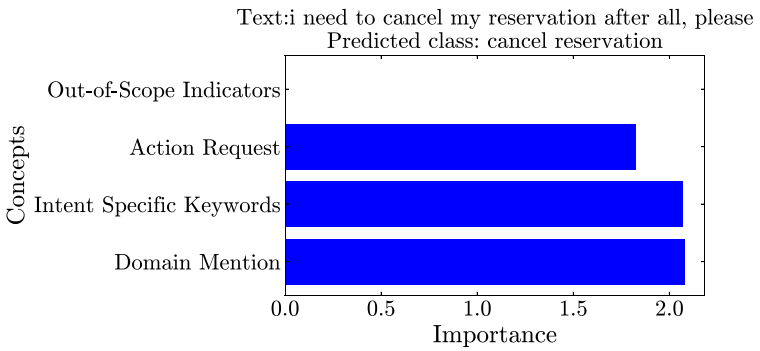


Fig. 16 Explanation 1 from the CLINC-OOS dataset

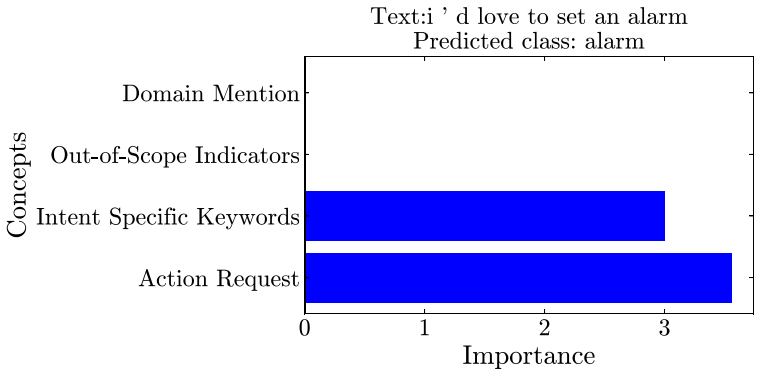


Fig. 17 Explanation 2 from the CLINC-OOS dataset

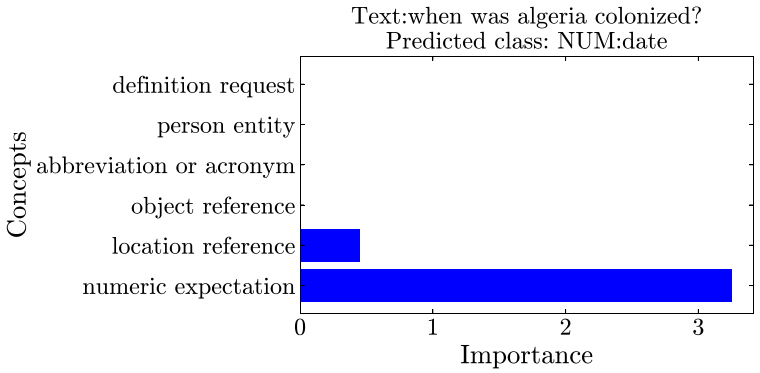


Fig. 18 Explanation 1 from the TREC-50 dataset

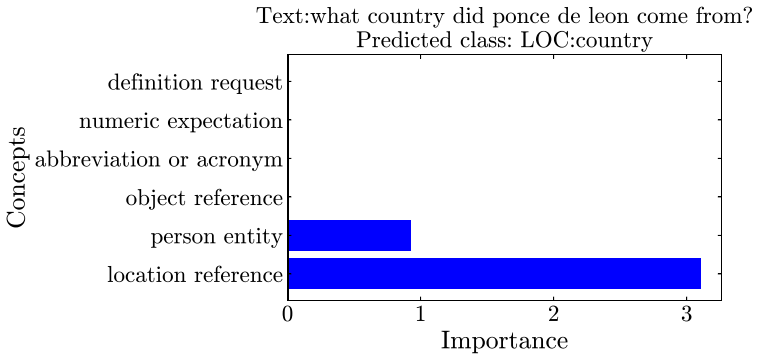


Fig. 19 Explanation 2 from the TREC-50 dataset

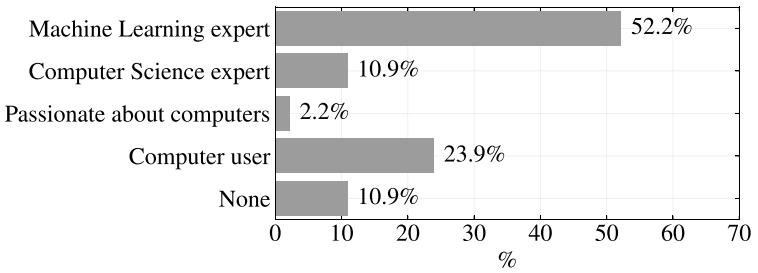


Fig. 20 Distribution of users by expertise level

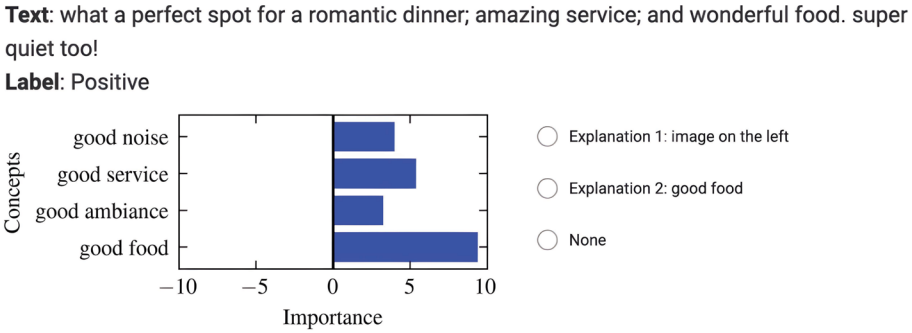


Fig. 21 Example of explanation plausibility question, CEBaB dataset

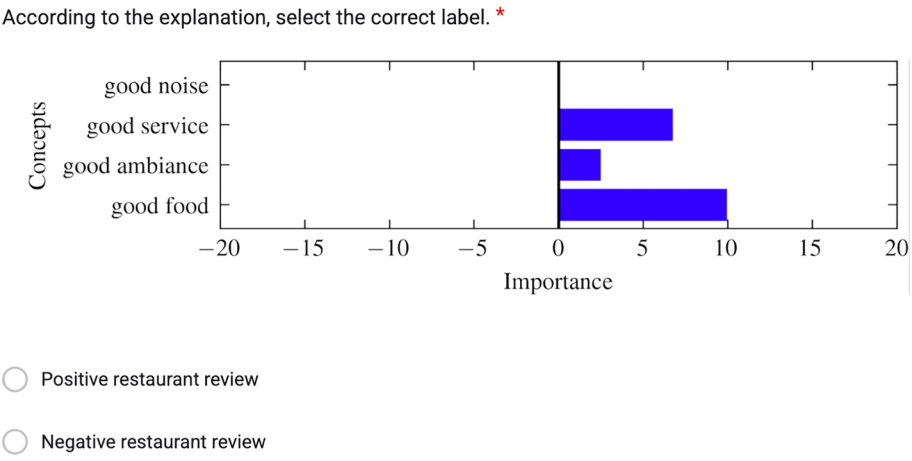


Fig. 22 Example of label prediction given LICEM explanation, CEBaB dataset

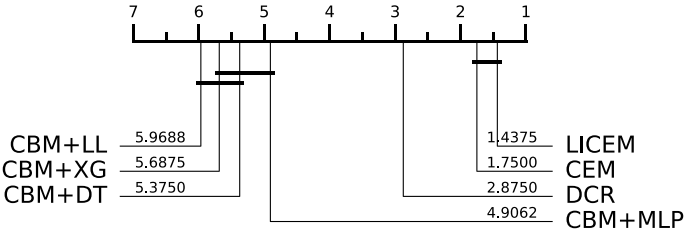


Fig. 23 Results for the CD diagram considering all the models in both the *supervised*, *generative*, and *self-generative* scenarios for all the datasets

Table 7 Average deviation between predicted and ground-truth sparsity

Method	DRUG	CEBaB	MULTIEMO
CBM + LL	0.48 ± 0.07	0.64 ± 0.09	0.66 ± 0.05
CBM + MLP	0.30 ± 0.03	0.58 ± 0.04	0.71 ± 0.06
CEM	0.21 ± 0.02	0.42 ± 0.02	0.73 ± 0.03
DCR	0.21 ± 0.02	**0.29** ± 0.02	0.71 ± 0.04
LICEM	**0.18** ± 0.01	0.39 ± 0.01	**0.58** ± 0.02

Lower values indicate closer alignment with the intrinsic sparsity of the data

On average, LICEM achieves the lowest deviation from the ground-truth sparsity on the DRUG and MULTIEMO datasets, indicating that its explanations most faithfully reflect the true number of active concepts. DCR also exhibits consistently low deviation across datasets. By contrast, the CBM variants tend to produce higher deviations, particularly on CEBaB and MULTIEMO. These results complement the concept accuracy findings (Tables 5 and 6) and underscore the importance of jointly evaluating sparsity and alignment with ground-truth concepts.

Author contributions D.S.F. and Bich P. contributed in producing the code and wrote the experimental section of the manuscript. C.G. wrote the remaining sections of the paper. Barbiero P. contributed in curating the methodology section. G.D. and C.T. contributed in curating the related works and the appendix of the manuscript. All authors reviewed the manuscript.

Funding Open access funding provided by Politecnico di Torino within the CRUI-CARE Agreement.

Data Availability No datasets were generated or analysed during the current study.

Declarations

Conflict of interest The authors declare no Conflict of interest.

References

Abraham, E. D., D'Oosterlinck, K., Feder, A., Gat, Y., Geiger, A., Potts, C., Reichart, R., & Wu, Z. (2022). Cebab: Estimating the causal effects of real-world concepts on NLP model behavior. *Advances in Neural Information Processing Systems, 35*, 17582–17596.

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., & Kim, B. (2018). Sanity checks for saliency maps. Advances in Neural Information Processing Systems, 31, 9525–9536.

Alvarez Melis, D., & Jaakkola, T. (2018). Towards robust interpretability with self-explaining neural networks. *Advances in Neural Information Processing Systems, 31, 7786–7795*.

Barbiero, P., Ciravegna, G., Giannini, F., Espinosa Zarlenga, M., Magister, L. C., Tonda, A., Lio, P., Precioso, F., Jamnik, M., & Marra, G. (2023). Interpretable neural-symbolic concept reasoning. In Proceedings of the 40th international conference on machine learning. Proceedings of machine learning research (vol. 202, pp. 1801–1825).

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. Advances in neural information processing systems, 33, 1877-1901.

Casanueva, I., Temcinas, T., Gerz, D., Henderson, M., & Vulic, I. (2020). Efficient intent detection with dual sentence encoders. In *Proceedings of the 2nd workshop on NLP for ConvAI—ACL 2020* . Retrieved from https://github.com/PolyAI-LDN/task-specific-datasets. https://arxiv.org/abs/2003.04807

Chefer, H., Gur, S., & Wolf, L. (2021). Generic attention-model explainability for interpreting bi-modal and encoder-decoder transformers. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 397–406).

Chefer, H., Gur, S., & Wolf, L. (2021). Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 782–791).

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 785–794).

Ciravegna, G., Barbiero, P., Giannini, F., Gori, M., Lió, P., Maggini, M., & Melacci, S. (2023). Logic explained networks. *Artificial Intelligence, 314*, 103822.

Dominici, G., Barbiero, P., Zarlenga, M.E., Termine, A., Gjoreski, M., & Langheinrich, M. (2024). Causal concept embedding models: Beyond causal opacity in deep learning. arXiv preprint arXiv:2405.16507

Espinosa Zarlenga, M., Barbiero, P., Ciravegna, G., Marra, G., Giannini, F., Diligenti, M., Shams, Z., Precioso, F., Melacci, S., Weller, A., Lió, P., & Jamnik, M. (2022). Concept embedding models: Beyond the accuracy-explainability trade-off. *Advances in Neural Information Processing Systems, 35*, 21400–21413.

Fel, T., Picard, A., Bethune, L., Boissin, T., Vigouroux, D., Colin, J., Cadène, R., & Serre, T. (2023). Craft: Concept recursive activation factorization for explainability. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 2711–2721).

Ghorbani, A., Abid, A., & Zou, J. (2019). Interpretation of neural networks is fragile. *Proceedings of the AAAI Conference on Artificial Intelligence, 33*, 3681–3688.

Goyal, Y., Feder, A., Shalit, U., & Kim, B. (2019). Explaining classifiers with causal concept effect (CACE). arXiv preprint arXiv:1907.07165

Gräßer, F., Kallumadi, S., Malberg, H., & Zaunseder, S. (2018). Aspect-based sentiment analysis of drug reviews applying cross-domain and cross-data learning. In *Proceedings of the 2018 international conference on digital health* (pp. 121–125).

Heyen, H., Widdicombe, A., Siegel, N.Y., Treleaven, P.C., & Perez-Ortiz, M. (2024). The effect of model size on LLM post-hoc explainability via lime. In *ICLR 2024 workshop on secure and trustworthy large language models*

Hovy, E., Gerber, L., Hermjakob, U., Lin, C.-Y., & Ravichandran, D. (2001). Toward semantics-based answer pinpointing. In *Proceedings of the first international conference on human language technology research*. Retrieved from https://www.aclweb.org/anthology/H01-1069

Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., ... & Liu, T. (2025). A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. ACM Transactions on Information Systems, 43(2), 1-55.

Ismail, A. A., Adebayo, J., Bravo, H.C., Ra, S., & Cho, K. (2023). Concept bottleneck generative models. In *The twelfth international conference on learning representations*.

Jain, S., & Wallace, B.C. (2019) Attention is not explanation. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: Human language technologies* (Vol. 1, pp. 3543–3556).

Jain, R., Ciravegna, G., Barbiero, P., Giannini, F., Buffelli, D., & Lio, P. (2022). Extending logic explained networks to text classification. In *Proceedings of the 2022 conference on empirical methods in natural language processing* (pp. 8838–8857).

Jiang, T., Huang, S., Luan, Z., Wang, D., & Zhuang, F. (2023). Scaling sentence embeddings with large language models

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., ... & Sayed, W. E. (2024). Mixtral of experts. arXiv preprint arXiv:2401.04088.

Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., & Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys, 55*(12), 1–38.

Kenton, J. D. M. -W. C., & Toutanova, L. K. (2019) Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT* (vol. 1, p. 2). Minneapolis.

Kim, E., Jung, D., Park, S., Kim, S., & Yoon, S. (2023). Probabilistic concept bottleneck models. In *Proceedings of the 40th international conference on machine learning*. ICML'23

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).

Koh, P. W., Nguyen, T., Tang, Y. S., Mussmann, S., Pierson, E., Kim, B., & Liang, P. (2020). Concept bottleneck models. *International conference on machine learning* (pp. 5338–5348). PMLR.

Kokalj, E., Škrlj, B., Lavrač, N., Pollak, S., & Robnik-Šikonja, M. (2021). BERT meets shapley: Extending SHAP explanations to transformer-based classifiers. In H. Toivonen, and M. Boggia (Eds.), *Proceedings of the EACL Hackashop on news media content analysis and automated report generation* (pp. 16–21).

Larson, S., Mahendran, A., Peper, J.J., Clarke, C., Lee, A., Hill, P., Kummerfeld, J. K., Leach, K., Laurenzano, M. A., Tang, L., & Mars, J. (2019). An evaluation dataset for intent classification and out-of-scope prediction. *Proceedings conference on natural language processing (EMNLP-IJCNLP)*. Retrieved from https://www.aclweb.org/anthology/D19-1131

Li, X., & Roth, D. (2002). Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics* . https://www.aclweb.org/anthology/C02-1150

Loshchilov, I., & Hutter, F. (2017). Fixing weight decay regularization in Adam. CoRR **abs/1711.05101** 1711.05101

Ludan, J.M., Lyu, Q., Yang, Y., Dugan, L., Yatskar, M., & Callison-Burch, C. (2023). Interpretable-by-design text classification with iteratively generated concept bottleneck. arXiv preprint arXiv:2310.19660

Lundberg, S.M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems, 30, 4768–4777*.

Maas, A., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., & Potts, C. (2011). Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies* (pp. 142–150).

Madsen, A., Chandar, S., & Reddy, S. (2024). Are self-explanations from large language models faithful? In *Findings of the Association for Computational Linguistics ACL 2024* (pp. 295–337).

Marconato, E., Passerini, A., & Teso, S. (2022). Glancenets: Interpretable, leak-proof concept-based models. *Advances in Neural Information Processing Systems, 35*, 21212–21227.

Oikarinen, T., Das, S., Nguyen, L.M., & Weng, T.-W. (2023). Label-free concept bottleneck models. In *The eleventh international conference on learning representations*.

Pearl, J., Glymour, M., & Jewell, N. P. (2016). Causal inference in statistics: A primer.

Poeta, E., Ciravegna, G., Pastor, E., Cerquitelli, T., & Baralis, E. (2023). Concept-based explainable artificial intelligence: A survey. arXiv preprint arXiv:2312.12936

Poursabzi-Sangdeh, F., Goldstein, D. G., Hofman, J. M., Wortman Vaughan, J. W., & Wallach, H. (2021). Manipulating and measuring model interpretability. In *Proceedings of the 2021 CHI conference on human factors in computing systems* (pp. 1–52).

Rajagopal, D., Balachandran, V., Hovy, E.H., & Tsvetkov, Y. (2021) Selfexplain: A self-explaining architecture for neural text classifiers. In *Proceedings of the 2021 conference on empirical methods in natural language processing* (pp. 836–850).

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).

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence, 1*(5), 206–215.

Sprugnoli, R. (2020). Multiemotions-it: a new dataset for opinion polarity and emotion analysis for italian. In Proceedings of the seventh Italian conference on computational linguistics (CLiC-it 2020) (pp. 402-408). Accademia University Press.

Taimeskhanov, M., Sicre, R., & Garreau, D. (2024). Cam-based methods can see through walls. In *Joint European conference on machine learning and knowledge discovery in databases* (pp. 332–348). Springer

Tan, Z., Chen, T., & Zhang, Z., & Liu, H. (2024). Sparsity-guided holistic explanation for LLMS with interpretable inference-time intervention. *Proceedings of the AAAI Conference on Artificial Intelligence, 38*, 21619–21627.

Tan, Z., Cheng, L., Wang, S., Yuan, B., Li, J., & Liu, H. (2024). Interpreting pretrained language models via concept bottlenecks. In *Pacific-Asia conference on knowledge discovery and data mining* (pp. 56–74). Springer.

Turpin, M., Michael, J., Perez, E., & Bowman, S. (2024). Language models don't always say what they think: unfaithful explanations in chain-of-thought prompting. *Advances in Neural Information Processing Systems, 36, 3275*.

Wiegreffe, S., & Pinter, Y. (2019) Attention is not not explanation. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)* (pp. 11–20).

Yang, Y., Panagopoulou, A., Zhou, S., Jin, D., Callison-Burch, C., & Yatskar, M. (2023). Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 19187–19197).

Yates, A., Cohan, A., & Goharian, N. (2017). Depression and self-harm risk assessment in online forums. In *Proceedings of the 2017 conference on empirical methods in natural language processing* (pp. 2968–2978).

Ye, S., Hwang, H., Yang, S., Yun, H., Kim, Y., & Seo, M. (2023). Investigating the effectiveness of task-agnostic prefix prompt for instruction following.

Ye, X., & Durrett, G. (2022). The unreliability of explanations in few-shot prompting for textual reasoning. *Advances in Neural Information Processing Systems, 35*, 30378–30392.

Yeh, C.-K., Kim, B., Arik, S., Li, C.-L., Pfister, T., & Ravikumar, P. (2020). On completeness-aware concept-based explanations in deep neural networks. *Advances in Neural Information Processing Systems, 33*, 20554–20565.

Authors and Affiliations

Francesco De Santis[1] · Philippe Bich[2] · Gabriele Ciravegna[1,3] · Pietro Barbiero[4] · Danilo Giordano[1] · Tania Cerquitelli[1]

✉ Francesco De Santis
francesco.desantis@polito.it

Philippe Bich
philippe.bich@polito.it

Gabriele Ciravegna
gabriele.ciravegna@polito.it

Pietro Barbiero
pietro.barbiero@usi.ch

Danilo Giordano
danilo.giordano@polito.it

Tania Cerquitelli
tania.cerquitelli@polito.it

[1] Dipartimento di Automatica e Informatica, Politecnico di Torino, Corso Duca degli Abruzzi, Torino 10129, Italy

[2] Dipartimento di Elettronica e Telecomunicazioni, Politecnico di Torino, Corso Duca degli Abruzzi, Torino 10129, Italy

[3] CENTAI Institute, Corso Inghilterra, 3, Torino 10138, Italy

[4] Faculty of Informatics, Universita'della Svizzera Italiana, Via Giuseppe Buffi, Lugano 6900, Switzerland