Towards Achieving Concept Completeness for Unsupervised Textual Concept Bottleneck Models

Anonymous ACL submission

Abstract

Textual Concept Bottleneck Models (TBMs) are interpretable-by-design models for text classification that predict a set of salient concepts before making the final prediction. This paper proposes Complete Textual Concept Bottleneck Model (CT-CBM), a novel TCBM generator building concept labels in a fully unsupervised manner using a small language model, eliminating both the need for predefined human labeled concepts and LLM annotations. CT-CBM iteratively targets and adds important concepts in the bottleneck layer to create a complete concept basis and addresses downstream classification leakage through a parallel residual connection. CT-CBM achieves good results against competitors, offering a promising solution to enhance interpretability of NLP classifiers without sacrificing performance.

1 Introduction

The striking level of performance in natural language processing (NLP) achieved by black-box neural language models (Vaswani et al., 2017; Brown et al., 2020; Chowdhery et al., 2023) comes along with a lack of interpretability (Madsen et al., 2022). The field of eXplainable Artificial Intelligence (XAI) (Longo et al., 2024) intends to make the behavior of such models more interpretable. A common distinction of XAI is to define interpretability methods either (1) by applying post hoc explanation methods to interpret black box models, or (2) by constructing interpretable models by-design (Jacovi and Goldberg, 2020; Madsen et al., 2024).

One promising approach to designing models that are more interpretable is Concept Bottleneck Models (CBM) (Koh et al., 2020). CBM are models that first map the input representations to a set of human-interpretable high-level attributes, called *concepts*. The latter are then used to make the final prediction with a linear layer,

	C^3M	CB-LLM	CT-CBM (ours)
Need for predefined concepts	Yes	Yes	**No**
Classification leakage tackling	No	No	**Yes**
Concept base completenetess	No	No	**Yes**
Accurate concept detection	No	No	**Yes**
Black-box performance reached	**Yes**	**Yes**	**Yes**
Use of ChatGPT	Yes	Yes	**No**
Scalability	No	**Yes**	**Yes**

Table 1: Qualitative comparison of CT-CBM to competitors. Desired modalities are highlighted in bold.

improving the interpretability of black box models. While CBM have been widely used in computer vision (Yuksekgonul et al., 2023; Oikarinen et al., 2023; Shang et al., 2024; Zarlenga et al., 2022), they have been much less explored for NLP (Poeta et al., 2023a). Existing Textual Concept Bottleneck Models (TCBM) have limitations: (i) they mainly rely on the use of large language models (LLM) (Tan et al., 2024a,b; Sun et al., 2024; Ludan et al., 2023) whose computational cost is very high, (ii) they often require access to a set of predefined human-labeled concepts (Tan et al., 2024b,a; Ludan et al., 2023), (iii) their concept layers can be non complete, missing concepts relevant for the downstream classification and potentially leading to performance loss (Tan et al., 2024b; Sun et al., 2024; Tan et al., 2024a), (iv) they do not address classification leakage (Tan et al., 2024a,b; Sun et al., 2024), leading to the use of unintended information from the concept predictor, (v) they do not systematically guarantee the reliability of concept activations, actually challenging the interpretability faithfulness of CBM.

In this paper, we propose Complete Textual Concept Bottleneck Model (CT-CBM), a novel approach to transform any fine-tuned NLP classifier into an interpretable-by-design TCBM. As summarised in Table 1, the main contributions of CT-CBM are as follows:

1. Concept labels are computed in a fully unsupervised manner without the need of predefined labelled concepts, based only a small language model (SLM).
2. Concept completeness is achieved through iterative addition of important concepts in the concept layer.
3. Downstream classification leakage is addressed through a parallel residual connection.

This paper is organized as follows: Section 2 recalls some basic principles of XAI and related work. Section 3 describes in details the proposed CT-CBM. Section 4 discusses the conducted experiments, that show that CT-CBM systematically succeeds in reaching the performance of a fine-tuned black box neural NLP model, while accurately detecting the activations of the concepts contained in the complete concept bottleneck layer.

2 Background and Related Work

This section first recalls some principles of XAI methods used later in the papers and presents existing methods generating Concept Bottleneck Model for NLP.

2.1 XAI Background

Post Hoc Interpretability. Post hoc methods explain the behavior of a model after its training. These include post hoc attribution methods that attribute importance scores to inputs to explain the model outcome (Zhao et al., 2024). In particular, gradient-based approaches such as Integrated Gradients (Sundararajan et al., 2017) compute these scores by back-propagating the gradients through the model.

Post hoc concept-based approaches generate explanations at a higher level of abstraction, by focusing on human interpretable attributes, called *concepts*. TCAV (Kim et al., 2018) assesses the model's sensitivity to a concept by back-propagating the gradients with respect to a linear representation of a candidate concept in the activation space, called concept activation vector (CAV). In the original paper, TCAV relies on human-labeled concepts, whose annotation can be time-consuming and expensive.

Concept Bottleneck Models. Another way to improve interpretability consists in constructing so-called Concept Bottleneck Models (CBM) (Koh et al., 2020). These models sequentially (1) detect concepts and (2) linearly make the final prediction from concept activations, thereby significantly improving the understanding of the decision-making process.

CBM have some limitations, such as requiring a predefined set of human-labeled concepts, generating incomplete concept bottleneck layers (CBL), or doing downstream task leakage. Concept incompleteness can have consequences either on the model accuracy (under-complete concept base) or the intelligibility (over-complete concept base) of the provided explanations (Shang et al., 2024). Downstream task leakage (Havasi et al., 2022) occurs when the final prediction uses unintended additional information from the concept predictor scores. The concept predictor then no longer needs to detect faithfully the concepts to be accurate on the classification task, thus compromising the interpretability faithfulness of the CBM.

Among the vast literature on CBM (Poeta et al., 2023b), numerous variants have been proposed to address one by one the aforementioned limitations. Notably, Label-Free CBM (Oikarinen et al., 2023) prompts GPT-3 (Brown et al., 2020) to list the most important concepts for recognizing a specific class, freeing the approach from dependency on predefined labeled concepts. However, Label-Free CBM structurally depends on the parametric knowledge of GPT-3 and does not generate data driven concepts. In order to reach the accuracy of a black box NLP classifier while avoiding leakage, a non-interpretable connection parallel to the concept layer can be added to fit the residuals between the raw CBM outcome and the ground truth (Yuksekgonul et al., 2023; Havasi et al., 2022). However, adding such a residual connection decreases the CBM interpretability. Res-CBM (Shang et al., 2024) develops a method to derive new concepts from the residual layer to build a more complete CBL. Yet, it requires access to a set of candidate concepts to add to the CBL before probing the residual connection. Although these methods overcome some of the limitations inherent in CBM, their application has so far been restricted to computer vision.

2.2 Textual Concept Bottleneck Models

This section presents recent works on generating Concept Bottleneck Models for NLP, referred to as Textual Concept Bottleneck Models (TCBM).

$\texttt{C}^3\texttt{M}$ (Tan et al., 2024b) enriches a set of predefined human-labeled concepts with additional concepts obtained from ChatGPT. While it approximately reaches the performance of an unrestricted black-box NLP classifier, it is trained without addressing the completeness of the CBL and the downstream classification leakage. Besides, its relying on ChatGPT and human-labeled concepts prevents reproducibility and scalability.

`CB-LLM` (Sun et al., 2024, 2025) also uses ChatGPT to generate a set of concepts that are scored with a sentence embedding model (Reimers and Gurevych, 2019) to perform concept labeling. This way, concept are represented with numerical values, unlike $\texttt{C}^3\texttt{M}$. The concepts are then added to the CBL to train the TCBM. While `CB-LLM` approximately reaches the performance of a black-box NLP classifier, downstream classification leakage is not addressed. Moreover, it overlooks the completeness of its CBL, possibly resulting in an excessive number of concepts in the bottleneck layer and unintelligible explanations.

`TBM` (Ludan et al., 2023) iteratively discovers concepts by leveraging GPT-4 (Achiam et al., 2023) and focusing on examples misclassified by a separately trained linear layer. `TBM` is not strictly a CBM, since concept detection is performed with GPT-4 during inference, making also the approach non scalable and computationally expensive.

3 Proposed approach: `CT-CBM`

This section describes the architecture of the proposed Complete Textual Concept Bottleneck Model (`CT-CBM`). As summarized in Table 1, `CT-CBM` addresses classification leakage and only uses SLM to build TCBM with a complete concept bottleneck basis without the need for a predefined human-labeled set of concepts.

3.1 `CT-CBM` Overview

We consider a corpus of text-label pairs $\mathcal{T} = \{(x, y)\}$ where $x \in \mathcal{X}$ denotes the text and $y \in \mathcal{Y}$ the label. $f : \mathcal{X} \rightarrow \mathbb{R}^d$ is the backbone of a language model classifier fine-tuned on $\mathcal{T}$, with d the dimension of the f embedding space. The TCBM is constructed by iteratively adding concepts into the CBL until a stopping criterion, which validates concept completeness, is met. Concepts are added progressively based on their importance scores, starting with the highest. As shown in Figure 1, `CT-CBM` is a 4-step framework detailed in the following subsections:

1. Concept Bank Construction. A set of concept candidates is generated from $\mathcal{X}$. `CT-CBM` prompts an auto-regressive SLM to enrich $\mathcal{T}$ with micro concepts defined as topics. Micro concepts are clustered to construct a set of meaningful high level macro concept candidates.

2. Concept Scoring. The importance of the candidate concepts is assessed, in order to select the one to be added in the concept bottleneck layer.

3. TCBM Training. Given a set of concepts, we train a *simple* TCBM and a *residual* TCBM with an additional parallel residual connection.

4. Stopping Criterion. Training stops when the importance of the residual connection is low and stable, indicating a complete concept bottleneck basis. The residual layer is finally removed to obtain a *simple* TCBM.

3.2 Concept Bank Construction

The first step aims at constructing a set of concept candidates $\mathcal{C}$ without human annotation and in an automated manner for potential inclusion in the final CBL of the TCBM.

Micro Concept Bank Creation. We first prompt an auto-regressive language model to annotate each input text with several topics that we call micro concepts, representing features at a higher level of abstraction than simple tokens. We then define the micro concept bank $\widetilde{\mathcal{C}}$ as the set of micro concepts (topics) associated with the text corpus $\mathcal{X}$ by the auto-regressive language model.

To be scalable and computationally affordable, `CT-CBM` uses an SLM, defined as having less than 9B parameters (Lu et al., 2024), to perform the micro concept annotation. This choice is justified by the ability of recent SLM to follow precisely instructions at lower cost than LLM (Riviere et al., 2024). We give more information about the prompt used to generate micro concepts in Appendix A.3.1.

Macro Concept Bank Creation. Secondly, we construct a set of macro concepts $\mathcal{C}$ by decomposing the set of micro concepts $\widetilde{\mathcal{C}}$ into p clusters, with $p \ll |\widetilde{\mathcal{C}}|$. This choice is justified by the variability of the micro concepts generated by the SLM whose labels may differ but whose semantics may be very similar (e.g.

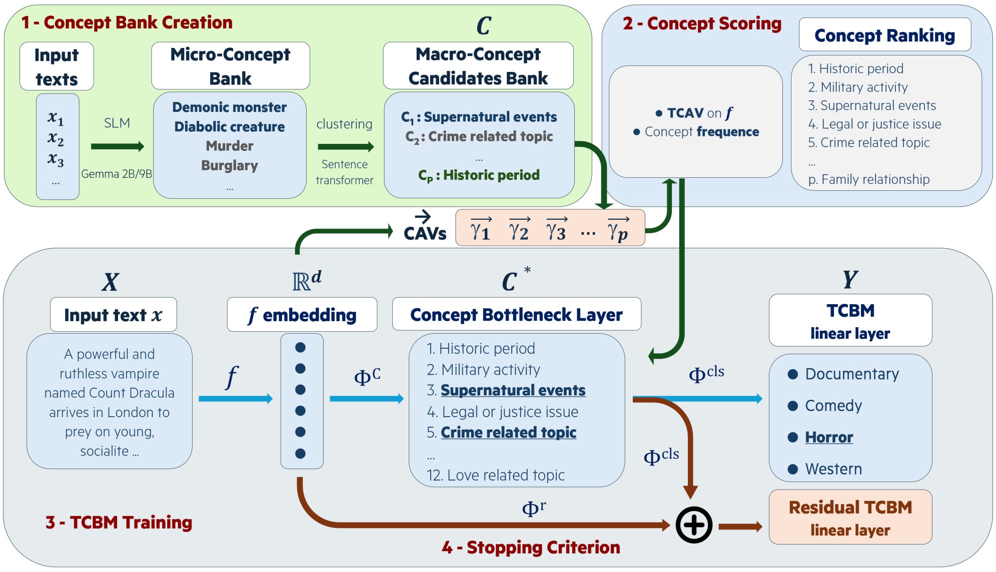

Figure 1: CT-CBM overview illustrated in the example of film synopsis classification. CT-CBM is a 4-step approach to build a TCBM from a f black box NLP classifier. (1) A concept bank is created from the text corpus of interest. (2) Concepts are scored with respect to their importance to explain f predictions. (3) The TCBM is trained through 3 layers: Φ^C, Φ^{cls} and Φ^r. (4) The TCBM training stops when the importance of Φ^r stops decreasing.

demoniac monster vs. *diabolic creature* clustered as *supernatural entities*). We sequentially use a sentence embedding model, UMAP (McInnes et al., 2018) and HDBSCAN (McInnes et al., 2017) to perform macro concept clustering. By deriving candidate concepts from the analysis of the entire text corpus, CT-CBM concept bank creation stage is *data-driven*, unlike its competitors which directly ask ChatGPT to provide the most promising concepts in general.

The text corpus is finally formally defined as $\mathcal{T}_M = \{(x, y, c)\}$, where $c = [c_1, ..., c_p] \in \{0, 1\}^p$ is a vector of absence or presence of the p macro concepts found in $\mathcal{X}$. As detailed below, each concept c_k is associated with a numerical representation $\overrightarrow{\gamma_k}$ and a textual label l_k.

Concept Activations Vectors Computing. The "Linear Representation Hypothesis" states that high-level concepts are represented linearly in the embedding space of language models (Elhage et al., 2022; Park et al., 2024). Motivated by this hypothesis, we assign to each macro concept a linear representation from f embedding space, called Concept Activation Vector (CAV). These CAVs are later used to build the CBL.

For each concept c_k, we define its CAV $\overrightarrow{\gamma_k}$ as the mean difference (MD) of embeddings (Rimsky et al., 2024):

$$\overrightarrow{\gamma_k} = \frac{1}{|\mathcal{X}_k^+|} \sum_{x \in \mathcal{X}_k^+} f(x) - \frac{1}{|\mathcal{X}_k^-|} \sum_{x \in \mathcal{X}_k^-} f(x) \quad (1)$$

where $\mathcal{X}_k^+$ and $\mathcal{X}_k^-$ respectively represent the corpora of texts where c_k is present or absent. Among the different ways to compute a CAV (Wu et al., 2025), MD leads to the best compromise is terms of concept detection accuracy and computational cost (Marks and Tegmark, 2024).

Macro Concept Labeling. Finally, the macro concepts are assigned to a textual label l_k using the SLM used for micro concept annotation. Due to the potential various micro concepts related to a macro concept, we sample the 15 micro concepts closest to the macro concept centroid and prompt the SLM to define the superclass of these micro concepts. More details about the prompt used to do macro concept labeling are given in Appendix A.3.2.

3.3 Concept Scoring

The objective of the concept scoring step is to find $\mathcal{C}^* \subset \mathcal{C}$ to be included in the final CBL. CT-CBM assigns a fixed importance score to each concept throughout the iterative TCBM generation. CT-CBM implements 2 ways of computing score importance.

Firstly, based on the previously computed CAVs, we apply TCAV as in Nejadgholi et al. (2022)

on the last layer of f to compute the fraction of inputs positively influenced by each concept summed over all target classes. This way, we know which concepts are most important for f to make its predictions. Secondly, motivated by the fact that the most represented concepts in the training dataset are often among the most detectable ones in the latent representations of language models (Templeton et al., 2024), `CT-CBM` also implements the concept `frequency` as a simple way of targeting concepts.

The concepts are then sorted in descending order of importance and iteratively added to the CBL.

3.4 TCBM Training

Given a subset of concepts $\mathrm{C} \subset \mathcal{C}$, we introduce the protocol followed by `CT-CBM` to train a TCBM. To avoid classification leakage, we propose to guide the evolution of the TCBM training by adding a residual connection parallel to the CBL as done with CBM in computer vision (Shang et al., 2024). Generating a *simple* TCBM consists in training the two following layers: $\Phi^{\mathrm{C}} : \mathbb{R}^d \rightarrow \mathbb{R}^{|\mathrm{C}|}$ and $\Phi^{\mathrm{cls}} : \mathbb{R}^{|\mathrm{C}|} \rightarrow \mathcal{Y}$, where Φ^{C} is the layer detecting concepts from f embedding and Φ^{cls} is the sparse linear concept-based classification layer. The *simple* TCBM is then defined by $\Phi^{\mathrm{cls}} \circ \Phi^{\mathrm{C}} \circ f$. A *residual* TCBM contains an additional non interpretable residual layer $\Phi^r : \mathbb{R}^d \rightarrow \mathcal{Y}$ using unknown residual concepts to enhance the downstream classification accuracy of the TCBM and avoid leakage. This way, the *residual* TCBM is defined as $((\Phi^{\mathrm{cls}} \circ \Phi^{\mathrm{C}}) + \Phi^r) \circ f$.

`CT-CBM` constructs Φ^{C} based on supervised learning, minimising the following loss function:

$$\mathcal{L}_{TCBM} = \lambda \mathcal{L}(\Phi^{\mathrm{C}}(f(x)), \mathtt{c}) \qquad (2)$$
$$+\mathcal{L}(\Phi^{\mathrm{cls}}(\Phi^{\mathrm{C}}(f(x)) + \Phi^r(f(x)), y)$$

where $\mathcal{L}$ is the cross-entropy loss, λ is a hyperparameter and $\mathtt{c}$ is the vector of absence or presence of the concepts included in C. Φ^{r} is trained with a Ridge penalty constraint as in Yuksekgonul et al. (2023) and Φ^{cls} is trained with an elastic net penalty constraint to foster sparsity. The supervised training can be done *jointly* (concept detection and downstream classification performed at the same time) or *sequentially* (concept detection learned firstly and classification training performed afterwards).

`CT-CBM` also implements a TCBM building method based on the CAVs projection in the concept layer. We present this `projection` methodology and give more details about training strategies, Ridge and elastic net hyperparameters in Appendix A.4.

3.5 Stopping Criterion

`CT-CBM` stops adding concepts in the CBL when the concept basis is deemed complete. The residual connection is then removed in order to obtain a TCBM without non-interpretable layer.

Since the residual layer uses unknown residual concepts, we measure its importance in the classification decision process as a proxy of concept completeness. A low residual connection importance indicates that most of the necessary concepts have been added to the CBL, pinpointing a complete concept base. Thus, we define the stopping criterion based on the importance of Φ^r in the TCBM decision making process. For a given input text $x \in \mathcal{X}$ and a given target class $k \in \mathcal{Y}$, we formally define the importance of Φ^r as follows:

$$\mathtt{I}_r^k(x) = \frac{|\langle w_k, f(x)\rangle|}{|\langle w_k, f(x)\rangle| + \langle |a_k|, |\Phi^{\mathrm{C}}(f(x))|\rangle} \qquad (3)$$

where w_k and a_k are respectively the weights of Φ^r and Φ^{cls} associated to class k: $\mathtt{I}_r^k(x)$ measures the importance of the residual connection relative to the sum of the importance of each concept in the decision process related to class k. We finally compute the global importance of Φ^r denoted as $\mathtt{I}_r$ by averaging $\mathtt{I}_r^k(x)$ over all target classes k and all inputs x.

`CT-CBM` training stops when the moving average of $\mathtt{I}_r$ of order 4 over iterations stops decreasing. We then select the iteration that minimizes the importance of Φ^r on the training set. The residual layer Φ^r is finally removed to end with a fully interpretable TCBM. We show an example of the evolution of the residual connection importance over the `CT-CBM` training in Appendix A.8

4 Experimental Settings

This section presents the experimental study conducted across four datasets and two NLP classifiers of different sizes, first comparing `CT-CBM` to several competitors. Next, we assess the impact of the method used to target important concepts. Then we illustrate several TCBM applications, such as the better understanding of counterfactual explanations and adversarial attacks (Lyu et al., 2024). Finally, we show how global explanations can be derived from a TCBM.

Model backbone (size)		BERT-base (110M)						DeBERTa-large (395M)				
Dataset	**Method**	Black-box	C^3M		CB-LLM	CT-CBM (ours)		Black-box	C^3M		CT-CBM (ours)	
	Concept Annotation		-	CT-CBM		-	C^3M		-	CT-CBM	-	C^3M
AG News	%ACC ↑	91.0	**91.5**	91.1	90.0	90.6	91.1	92.0	91.8	91.2	91.4	**92.5**
	%c ↑	-	**76.2**	54.8	56.0	62.8	70.2	-	81.5	55.0	58.9	**84.5**
	#c ↓	-	41	100	41	**11**	12	-	41	100	**12**	16
	%D ↑	-	76.5	78.5	76.5	**80.8**	81.2	-	76.5	78.5	**87.2**	85.1
DBpedia	%ACC ↑	99.4	**99.5**	**99.5**	99.3	99.3	99.4	99.4	**99.5**	99.4	99.2	**99.5**
	%c ↑	-	56.1	59.0	56.0	**60.8**	52.9	-	68.1	62.2	64.0	**70.5**
	#c ↓	-	63	100	63	**10**	16	-	63	100	**9**	12
	%D ↑	-	82.2	80.3	82.2	**85.1**	84.5	-	82.2	80.3	**85.0**	83.5
Medical Abstract	%ACC ↑	62.7	**61.4**	60.5	57.9	57.5	59.2	62.6	64.7	62.6	58.6	60.1
	%c ↑	-	50.6	50.2	25.2	51.1	**54.0**	-	54.7	51.6	58.5	**61.2**
	#c ↓	-	57	100	57	13	**10**	-	57	100	12	**9**
	%D ↑	-	77.2	76.4	77.2	**79.2**	77.1	-	**77.2**	76.4	77.1	76.9
Movie Genre	%ACC ↑	91.7	92.0	**92.6**	91.4	90.6	91.4	93.8	93.3	92.7	92.2	91.9
	%c ↑	-	70.4	45.3	29.8	52.5	**72.0**	-	**77.3**	51.5	55.2	62.1
	#c ↓	-	68	100	68	**11**	12	-	68	100	**10**	14
	%D ↑	-	78.1	78.8	78.1	**81.6**	80.2	-	78.1	78.8	**81.3**	80.5

Table 2: CT-CBM and competitors evaluation on four test sets and two NLP classifiers. Except the black box baseline, the best results are highlighted in bold and the second best ones are underlined.

4.1 Experimental Protocol

Datasets and models. CT-CBM is tested on four multi-class text classification datasets: AG News (Gulli, 2005), DBpedia (Lehmann et al., 2015), Movie Genre[1] and the more challenging classification dataset Medical Abstracts (Schopf et al., 2022). We apply CT-CBM on two fine-tuned NLP classifiers of different sizes: BERT (Devlin et al., 2019) and DeBERTa-large (He et al., 2020). More information about the content of the classification datasets and the language models are provided in Appendix A.5.

CT-CBM and Competitors. We run the proposed CT-CBM with a Gemma-2-9B SLM to generate concept candidates. The CBL is constructed by *joint* training and important concepts are targeted and added in the bottleneck layer with TCAV and frequency. We compare CT-CBM to both C^3M (Tan et al., 2024b) and CB-LLM (Sun et al., 2025). Given that TBM (Ludan et al., 2023) does not enhance an NLP classifier but rather performs concept detection with GPT4 during inference, we do not include it in the comparative study. We only show the results of CB-LLM for BERT, as we were unable to make TCBM converge for DeBERTa, despite using a grid search on the hyperparameters. We also run CT-CBM based on the concept annotation from C^3M and, conversely, we utilize C^3M based on CT-CBM concept annotation. To ensure comparability and address the ChatGPT annotation non scalability of C^3M (complexity proportional to size of the dataset $\times$ number of targeted concepts), we run C^3M with Gemma-2-9B as concept annotator. CB-LLM concept evaluation is done by discretizing its concept prediction since concept detection learning is done with numeric values. The hyperparameters of the experiments are detailed in Appendices A.4.4 and A.6.

Evaluation Criteria. We propose a 4-metric evaluation, with the first metric being the final classification task accuracy (**%ACC**). We evaluate concept detection accuracy (**%c**) based on the F1 score related to concept detection due to the strong imbalance in concept labels. Given the prohibitive computational cost of C^3M despite the use of Gemma-2-9B instead of ChatGPT for concept annotation, the C^3M evaluation of concept detection accuracy is only carried out on 1000 texts. We also report the number of concepts (**#c**) in the CBL and the concept diversity (**D**) in the TCBM. The latter is motivated by the extensive XAI literature emphasising the significance of diversity in the components of an explanation (Laugel et al., 2023; Mothilal et al., 2020). Moreover, greater diversity in the case of TCBM tends to reduce concept redundancy, therefore promoting the objective of a more complete concept base. We formally measure diversity as $D = 1 - \frac{2}{k(k-1)} \sum_{i=1}^{k} \sum_{j=i+1}^{k} \frac{\langle \mathbf{e_i} \cdot \mathbf{e_j} \rangle}{\|\mathbf{e_i}\| \|\mathbf{e_j}\|}$ where $\mathbf{e_i}$ is the embedding representation from a sentence embedding model of the textual label l_k of concept c_k.

[1] https://www.kaggle.com/competitions/movie-genre-classification/overview

Metric	Concept scoring	AG News	DBPedia	Medical Abstract	Movie Genre
%ACC ↑	TCAV	**90.6**	99.3	**57.5**	90.6
	frequency	88.5	**99.4**	57.6	**91.7**
%c ↑	TCAV	62.8	60.8	**52.5**	**52.5**
	frequency	**63.3**	**61.0**	49.7	51.2
#c ↓	TCAV	**11**	**10**	**13**	**11**
	frequency	19	13	14	19

Table 3: Downstream task accuracy, concept accuracy and number of concepts of CT-CBM applied to BERT per concept scoring method.

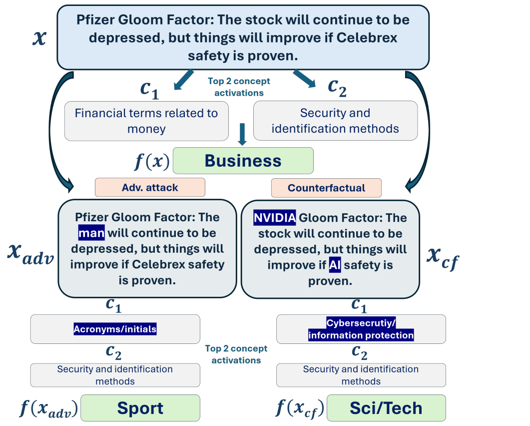

Figure 2: Example of an adversarial attack (x_{adv}) and a counterfactual explanation (x_{cf}) obtained from CT-CBM and the AGnews dataset. TCBM enable to understand the label change by explaining it in terms of concept change.

4.2 Results

Global Results. Table 2 shows the experimental results obtained from CT-CBM and its competitors on BERT and DeBERTa and the same training sets. C^{3}M with CT-CBM concept annotation consists in training the TCBM with the C^{3}M training methodology based on the dataset enriched with concepts dervied from the CT-CBM methodology. Conversely, CT-CBM with C^{3}M concept annotation applies the CT-CBM training approach subsequent to the generation of concepts following C^{3}M.

Overall, CT-CBM achieves a performance very similar to that of the original black box models, even surpassing it for AG news and DBpedia when the concept annotation was previously done with C^{3}M. For the majority of datasets and models, CT-CBM achieves the best results in terms of both quantity and diversity of concepts in the CBL. The accuracy of the classification task is comparable between CT-CBM C^{3}M and CB-LLM, except for the Medical Abstract dataset, where our approach performs marginally lower.

Influence of Concept Annotation on Concept Detection Accuracy. Concept detection accuracies of CT-CBM and C^{3}M are highly dependent on the method used to do concept annotation upstream. On average, the C^{3}M annotation leads to higher concept accuracy as compared to the CT-CBM annotation. Annotation with C^{3}M is extremely costly, with a complexity equal to the product of the number of concepts in the database and the size of the dataset, whereas annotation with CT-CBM has a complexity proportional to the size of the dataset of interest. For a given annotation method, our CT-CBM achieves on average a higher concept accuracy than C^{3}M. CB-LLM has overall a lower concept accuracy than its competitors.

Consequently, CT-CBM appears to offer a balancd compromise between downstream task accuracy, concept detection performance and concept diversity, while structurally avoiding classification leakage in a fully unsupervised manner. However, a heavier but higher-quality annotation method such as C^{3}M does improve results, at the expense of scalability.

Impact of the Concept Scoring Method. Table 3 shows the results of the evaluation of the impact of the concept scoring method of CT-CBM on its performance. It turns out that TCAV and the frequency approach give similar results in terms of downstream task and concept detection accuracy. However, TCAV converges much faster than frequency with less concepts in its CBL, which underlines the benefits of using TCAV to target important concepts to be added to the CBL. However, the frequency-based targeting concepts remains a noteworthy compromise in terms of its implementation simplicity and computational cost.

4.3 Practical Applications of TCBM

Better Understanding Adversarial Attacks and Counterfactual Explanations. A common application of CBMs is to allow domain experts to modify predicted concepts at test time to improve the accuracy of final task predictions (Steinmann et al., 2024). We propose here another application of TCBM by showing how TCBM can provide a better understanding of adversarial attacks effectiveness and counterfactual explanation expressivity. To this end, we apply TextAttack (Morris et al., 2020) and use Claude 3.5 Sonnet to generate adversarial attacks and counterfactual explanations to a TCBM trained

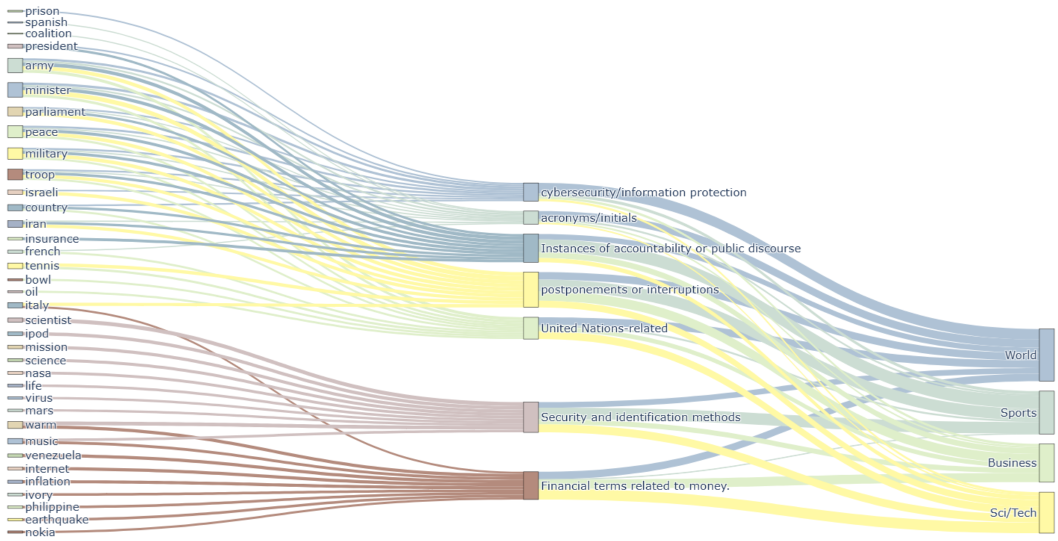

Figure 3: Global explanation of a TCBM trained on the AGnews dataset with `CT-CBM`. The 7 most important concepts of the bottleneck layer and the 8 most important tokens to explain each concept are shown.

via the proposed `CT-CBM` on the AGnews dataset, thereby achieveing a switch in the outcome of its prediction.

Figure 2 gives a salient example of how TCBM can provide an understanding of adversarial attacks and counterfactual explanations at the concept level. The adversarial attack succeeds in flipping the label of the TCBM from *Business* to *Sport* by performing the following change at token level: **stock** $\rightarrow$ **man**. The change in concepts **Financial terms related to money** $\rightarrow$ **Acronyms/initials** clearly highlights the model's poor understanding of the token change induced by the adversarial attack, offering avenues for troubleshooting the model. In the same way, the label flipping from *Business* to *Sci/Tech* induced by the counterfactual token changes **Pfizer** $\rightarrow$ **NVIDIA** and **Celebrex** $\rightarrow$ **AI** can be understood at a higher level of abstraction than the token level with the **Financial terms related to money** $\rightarrow$ **Security and identification methods**. This way, we believe that focusing on the conceptual level can significantly improve the understanding of a counterfactual explanation.

Global TCBM Interpretability. We illustrate how to interpret TCBM at a global scale by representing the relationship between tokens, concepts and target classes. Regarding the relationship between important tokens and concepts, we propose to apply an attribution method, here `Integrated gradients` (Sundararajan et al., 2017) to explain TCBM concept activations for each concept at a global scale. The resulting local feature importance scores are then averaged by concept. Finally, the weights of the Φ^{cls} layers are directly used to represent the intensity of the relationship between concepts and target labels. Figure 3 gives an example of a global explanation of the proposed `CT-CBM` trained on DBPedia. The "Financial terms related to money" concept is important to predict that a press article is related to "Science and Technology". Moreover, "internet" is identified as important to activate the "Financial terms related to money" concept.

5 Conclusion

We introduced `CT-CBM`, a novel unsupervised approach to transform a fine-tuned NLP classifier into a TCBM. `CT-CBM` automatically generates, scores and targets concepts to build a complete CBL. `CT-CBM` demonstrates a good level of performance as compared to its competitors for the two involved classification tasks, concept detection and class prediction, especially regarding the former. Moreover, `CT-CBM` leads to a more diverse concept basis, reducing the risk of redundant explanations at concept level. We have highlighted several advantages of TCBM, such as increased adversarial attacks and counterfactual intelligibility and the ability to produce global explanations.

6 Limitations

Datasets and models. This work tested CT-CBM on 4 datasets and 2 language models. It would be interesting to include other models in the study, such as recent decoder-only architectures e.g. Gemma-2B.

Concept Interactions. We have not considered possible relationships between concepts. This could highlight a better understanding of the impact of concepts on the classes to be predicted. We see this as a promising way of improving our approach.

Concept Importance. There are other approaches for assessing the importance of a concept in explaining the behavior of a model (Fel et al., 2023; Crabbé and van der Schaar, 2022), especially when concepts do not necessarily appear to be represented linearly in the latent spaces of models. Using these approaches would enable CT-CBM to better target important concepts to be added to the CBL.

Text generation. Recent work has proposed having generative models generate explanations before answering the question in the same way as TCBM (Bhan et al., 2024; Sun et al., 2025). For the time being, our work has focused on text classification.

Ethics Considerations

Since NLP training data can be biased, there is a risk of generating harmful concepts to be added in the CBL. One using CT-CBM to enhance a NLP classifier must be aware of these biases in order to stand back and analyze the produced concepts and the manipulated texts. Moreover, the use of Gemma-9B for concept annotation is computationally costly and consumes energy, potentially emitting greenhouse gases. CT-CBM must be used with caution.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Milan Bhan, Jean-Noël Vittaut, Nicolas Chesneau, and Marie-Jeanne Lesot. 2024. Self-AMPLIFY: Improving small language models with self post hoc explanations. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10974–10991, Miami, Florida, USA. Association for Computational Linguistics.

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

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. PaLM: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113.

Jonathan Crabbé and Mihaela van der Schaar. 2022. Concept activation regions: A generalized framework for concept-based explanations. *Advances in Neural Information Processing Systems*, 35:2590–2607.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. 2023. A Survey on In-context Learning. ArXiv:2301.00234 [cs].

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2022/toy_model/index.html.

Thomas Fel, Victor Boutin, Louis Béthune, Rémi Cadène, Mazda Moayeri, Léo Andéol, Mathieu Chalvidal, and Thomas Serre. 2023. A holistic approach to unifying automatic concept extraction and concept importance estimation. *Advances in Neural Information Processing Systems*, 36:54805–54818.

Antonio Gulli. 2005. Ag's corpus of news articles. *Dipartimento di Informatica, University of Pisa, Nov.*

Marton Havasi, Sonali Parbhoo, and Finale Doshi-Velez. 2022. Addressing leakage in concept bottleneck models. *Advances in Neural Information Processing Systems*, 35:23386–23397.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*.

Alon Jacovi and Yoav Goldberg. 2020. Towards faithfully interpretable nlp systems: How should we define and evaluate faithfulness? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. 2018. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. 2020. Concept bottleneck models. In *International conference on machine learning*, pages 5338–5348. PMLR.

Thibault Laugel, Adulam Jeyasothy, Marie-Jeanne Lesot, Christophe Marsala, and Marcin Detyniecki. 2023. Achieving diversity in counterfactual explanations: a review and discussion. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, pages 1859–1869.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Luca Longo, Mario Brcic, Federico Cabitza, Jaesik Choi, Roberto Confalonieri, Javier Del Ser, Riccardo Guidotti, Yoichi Hayashi, Francisco Herrera, Andreas Holzinger, et al. 2024. Explainable artificial intelligence (xai) 2.0: A manifesto of open challenges and interdisciplinary research directions. *Information Fusion*, 106:102301.

Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D Lane, and Mengwei Xu. 2024. Small language models: Survey, measurements, and insights. *arXiv preprint arXiv:2409.15790*.

Josh Magnus Ludan, Qing Lyu, Yue Yang, Liam Dugan, Mark Yatskar, and Chris Callison-Burch. 2023. Interpretable-by-design text classification with iteratively generated concept bottleneck. *arXiv preprint arXiv:2310.19660*.

Qing Lyu, Marianna Apidianaki, and Chris Callison-Burch. 2024. Towards faithful model explanation in nlp: A survey. *Computational Linguistics*, pages 1–67.

Andreas Madsen, Himabindu Lakkaraju, Siva Reddy, and Sarath Chandar. 2024. Interpretability needs a new paradigm. *arXiv preprint arXiv:2405.05386*.

Andreas Madsen, Siva Reddy, and Sarath Chandar. 2022. Post-hoc interpretability for neural nlp: A survey. *ACM Computing Surveys*, 55(8):1–42.

Samuel Marks and Max Tegmark. 2024. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets.

Leland McInnes, John Healy, and Steve Astels. 2017. hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11):205.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. 2018. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861.

Vivek Miglani, Aobo Yang, Aram Markosyan, Diego Garcia-Olano, and Narine Kokhlikyan. 2023. Using Captum to Explain Generative Language Models. In *Proc. of the 3rd Workshop for Natural Language Processing Open Source Software (NLP-OSS 2023)*, pages 165–173. Association for Computational Linguistics.

John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126.

Ramaravind Kommiya Mothilal, Amit Sharma, and Chenhao Tan. 2020. Explaining Machine Learning Classifiers through Diverse Counterfactual Explanations. In *Proc. of the 2020 Conf. on Fairness, Accountability, and Transparency*, pages 607–617. ArXiv:1905.07697 [cs, stat].

Isar Nejadgholi, Kathleen Fraser, and Svetlana Kiritchenko. 2022. Improving generalizability in implicitly abusive language detection with concept activation vectors. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5517–5529, Dublin, Ireland. Association for Computational Linguistics.

Tuomas Oikarinen, Subhro Das, Lam M Nguyen, and Tsui-Wei Weng. 2023. Label-free concept bottleneck models. In *Proc. of the 11th Int. Conf. on Learning Representations, ICLR*.

Kiho Park, Yo Joong Choe, and Victor Veitch. 2024. The linear representation hypothesis and the geometry of large language models. In *Forty-first International Conference on Machine Learning*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

Eleonora Poeta, Gabriele Ciravegna, Eliana Pastor, Tania Cerquitelli, and Elena Baralis. 2023a. Concept-based explainable artificial intelligence: A survey. *arXiv preprint arXiv:2312.12936*.

Eleonora Poeta, Gabriele Ciravegna, Eliana Pastor, Tania Cerquitelli, and Elena Baralis. 2023b. Concept-based explainable artificial intelligence: A survey. *arXiv preprint arXiv:2312.12936*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proc. of the 2019 Conf. on Empirical Methods in Natural Language Processing and the 9th Int. Joint Conf. on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992. Association for Computational Linguistics.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522, Bangkok, Thailand. Association for Computational Linguistics.

Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. 2024. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*.

Tim Schopf, Daniel Braun, and Florian Matthes. 2022. Evaluating unsupervised text classification: zero-shot and similarity-based approaches. In *Proceedings of the 2022 6th International Conference on Natural Language Processing and Information Retrieval*, pages 6–15.

Chenming Shang, Shiji Zhou, Hengyuan Zhang, Xinzhe Ni, Yujiu Yang, and Yuwang Wang. 2024. Incremental residual concept bottleneck models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11030–11040.

David Steinmann, Wolfgang Stammer, Felix Friedrich, and Kristian Kersting. 2024. Learning to intervene on concept bottlenecks. In *Proc. of the 41st Int. Conf. on Machine Learning, ICML*.

Chung-En Sun, Tuomas Oikarinen, Berk Ustun, and Tsui-Wei Weng. 2025. Concept bottleneck large language models. In *Proc. of the 13th Int. Conf. on Learning Representations, ICLR23*.

Chung-En Sun, Tuomas Oikarinen, and Tsui-Wei Weng. 2024. Crafting large language models for enhanced interpretability. *Proceedings of the Mechanistic Interpretability Workshop@ICML.*

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. 2017. Axiomatic attribution for deep networks. In *Proc. of the 34th Int. Conf. on Machine Learning, ICML*, volume 70 of *ICML'17*, pages 3319–3328. JMLR.org.

Zhen Tan, Tianlong Chen, Zhenyu Zhang, and Huan Liu. 2024a. Sparsity-guided holistic explanation for llms with interpretable inference-time intervention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 21619–21627.

Zhen Tan, Lu Cheng, Song Wang, Bo Yuan, Jundong Li, and Huan Liu. 2024b. Interpreting pretrained language models via concept bottlenecks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 56–74. Springer.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. 2024. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proc. of the Conf. on Empirical Methods in Natural language Processing: system demonstrations, EMNLP*, pages 38–45.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D Manning, and Christopher Potts. 2025. Axbench: Steering llms? even simple baselines outperform sparse autoencoders. *arXiv preprint arXiv:2501.17148*.

Mert Yuksekgonul, Maggie Wang, and James Zou. 2023. Post-hoc concept bottleneck models. In *The Eleventh International Conference on Learning Representations*.

Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Frederic Precioso, Stefano Melacci, Adrian Weller, Pietro Lio, et al. 2022. Concept embedding models. In *NeurIPS 2022-36th Conference on Neural Information Processing Systems*.

Haiyan Zhao, Hanjie Chen, Fan Yang, Ninghao Liu, Huiqi Deng, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, and Mengnan Du. 2024. Explainability for Large Language Models: A Survey. *ACM Transactions on Intelligent Systems and Technology*.

A Appendix

A.1 Scientific Libraries

We used several open-source libraries in this work: pytorch (Paszke et al., 2019), HuggingFace transformers (Wolf et al., 2020) sklearn (Pedregosa et al., 2011) and Captum (Miglani et al., 2023).

A.2 Autoregressive language models implementation details

Language Models. The library used to import the pretrained autoregressive language models is Hugging-Face. In particular, the backbone version of Gemma-2-9B is `gemma-2-9B-it`.

Gemma-2 instruction special tokens. The special tokens to use Gemma in instruction mode were the following:

- `Gemma-2:`
  - `user_token='<start_of_turn>user'`
  - `assistant_token='<start_of_turn>model'`
  - `stop_token='<eos>'`

Text generation. Text generation was performed using the native functions of the Hugging Face library: `generate`. The `generate` function has been used with the following parameters:

- `max_new_tokens = 50`
- `do_sample = True`
- `num_beams = 2`
- `no_repeat_ngram_size = 2`
- `early_stopping = True`
- `temperature = 1`

A.3 Prompting format

Here we provide some details of different prompts used to give instructions to Gemma-2-9B for micro concept annotation and macro concept labeling. We mainly leverage the In-context Learning (ICL) (Dong et al., 2023) capabilities of Gemma-2-9B.

A.3.1 Preprompt for micro concept generation

user
You are presented with several parts of speech. Identify only the main topics in this text. Respond with topic in list format like the examples in a very concise way using as few words as possible. Example: 'As cities expand and populations grow, there is a growing tension between development and the need to preserve historical landmarks. Citizens and authorities often clash over the balance between progress and cultural heritage.'
assistant
Topics: ['urban development', 'cultural heritage', 'conflict']<eos>
user
'Recent breakthroughs in neuroscience are shedding light on the complexities of human cognition. Researchers are particularly excited about the potential to better understand decision-making processes and emotional regulation in the brain.'
assistant
Topics: ['neuroscience', 'human cognition', 'decision-making', 'emotional regulation']<eos>

A.3.2 Preprompt for macro concept labeling

user
You are presented with several parts of speech. Summarise what these parts of speech have in common in a very concise way using as few words as possible. Example: ["piano", "guitar", "saxophone", "violin", "cheyenne", "drum"]
assistant
Summarization: 'musical instrument'<eos>
user
["football", "basketball", "baseball", "tennis", "badmington", "soccer"]
assistant
Summarization: 'sport'<eos>
user
["lion", "tiger", "cat", "pumas", "panther", "leopard"]

assistant
Summarization: 'feline-type animal'<eos>

A.4 TCBM implementation details

A.4.1 Micro concept clustering settings

In order to perform micro concept clustering to build macro concepts, we use the `umap` library to perform dimension reduction with UMAP with `n_components = 5`. Text embeddings are initially obtained with the `all-mpnet-base-v2` backbone from the `sentence_transformers` library. Finally, clustering is performed with `HDBSCAN` with the basic settings from the `hdbscan` library.

A.4.2 TCBM training strategies

`CT-CBM` implements two strategies for TCBM training: *joint* and *sequential*. The *sequential* strategy first predicts concepts from input texts and then uses these predicted concepts to make the final target prediction. In this approach, the output of the concept prediction stage is directly used as input for the target prediction stage. This way, the concept loss $\mathcal{L}(\Phi^{\text{C}}(f(x)), \text{c})$ is firstly minimized before minimizing the target one $\mathcal{L}(\Phi^{\text{cls}}(\Phi^{\text{C}}(f(x)) + \Phi^{r}(f(x)), y)$. On the other hand, the joint strategy predicts concepts and the final target simultaneously. It optimizes both concept prediction and target prediction losses during training. This enables the model to consider the relationship between concept and target predictions. This way, the loss of Equation 2 is directly optimized. In our experiments, TCBMs are trained *jointly* and the f parameters are frozen during the TCBM training.

A.4.3 Implementation of Φ^r and Φ^{cls}

Φ^r and Φ^{cls} are respectively trained with Ridge and elastic net penalties during the TCBM training. The Ridge penalization R can be written as follows:

$$R(W) = \lambda_R \|W\|_2^2 \tag{4}$$

with $W \in \mathbb{R}^{d \times k}$ the weight matrix of the Φ^r layer, λ_R an hyperparameter and $\|\cdot\|_2^2$ the L_2 norm. On the other hand, the elastic net penalization EN can be written as follows:

$$EN(A) = \lambda_{EN}\left(\alpha\|A\|_1 + (1-\alpha)\|A\|_2^2\right) \tag{5}$$

with $A \in \mathbb{R}^{|\text{C}| \times k}$ the weight matrix of the Φ^{cls} layer, λ_{EN} and α two hyperparameters and $\|\cdot\|_1$ the L_1 norm.

A.4.4 Other TCBM training hyperparameters

In our experiments, language model and TCBM training is done with the following hyperparameters:

- `batch_size` = 8
- `num_epochs` = 15
- `max_len` = 128 for AGNews and DBPedia, 256 for Movie Genre and 512 for Medical Abstracts.
- `learning rate` = 0.001
- `optimizer` = `Adam`
- $\lambda_R = 0.01$
- $\lambda_{EN} = 0.5$
- $\alpha = 0.01$
- $\lambda = 0.5$

A.4.5 TCBM construction with CAV projection

The `projection` approach to build Φ^{C} consists in projecting the CAVs into the concept space. We formally define $\Phi^{\text{c}_k}(f(x)) = \frac{\langle f(x), \overrightarrow{\gamma_k} \rangle}{||f(x)|| \cdot ||\overrightarrow{\gamma_k}||}$ as the linear projection of the embedding of x from f on the concept space associated to concept c_k. This way, the concept embedding projection consists in computing the cosine similarity between the CAV and f output. Φ^{C} is then constructed by concatenating linear projections corresponding to each concept c_i and the final layer. Finally, Φ^{cls} and Φ^{r} are trained to perform the classification by minimizing the following loss function:

$$\mathcal{L}_{TCBM} = \mathcal{L}(\Phi^{\text{cls}}(\Phi^{\text{C}}(f(x)) + \Phi^{r}(f(x)), y) \tag{6}$$

where $\mathcal{L}$ is the cross-entropy loss, Φ^r is trained with a Ridge penalty constraint and Φ^{cls} is trained with an elastic net penalty constraint.

A.5 Language model classifiers and classification datasets details

Language model classifiers. The library used to import the pretrained language models is HuggingFace. In particular, the backbone version of BERT is `bert-base-uncased` and the one of DeBERTa is `deberta-large`.

Classification datasets. The size of the training sets for AG News, DBpedia, Movie Genre and Medical Abstracts are respectively 4000, 6000, 4000 and 5000. The size of the test sets for AG News, DBpedia, Movie Genre and Medical Abstracts are respectively 23778, 30000, 7600 and 2888. `C`^{3}`M` concept evaluation is done on 1000 randomly selected rows on each dataset.

A.6 Competitors implementation details

In our experiments, `C`^{3}`M` (Tan et al., 2024b) training is done with the following hyperparameters:

- `batch_size` = 8
- `num_epochs` = 15
- `max_len` = 128 for AGNews and DBPedia, 256 for Movie Genre and 512 for Medical Abstracts.
- `learning rate` = 0.001
- `optimizer` = `Adam`
- $\lambda_R = 0.01$
- $\lambda_{EN} = 0.5$
- $\alpha = 0.01$
- $\lambda = 0.5$

The training of `CB-LLM` is a two-stage process: (1) CBL training and (2) classification layer training. The CBL training is done with the following hyperparameters:

- `batch_size` = 16
- `num_epochs` = 4
- `max_len` = 128 for AGNews and DBPedia, 256 for Movie Genre and 512 for Medical Abstracts.
- `learning rate` = 0.001
- `optimizer` = `Adam`
- `loss_function` = *cos cubed* as in Oikarinen et al. (2023) and Sun et al. (2025)

The training of the classification layer of `CB-LLM` is done with the following hyperparameters:

- `batch_size` = 64
- `num_epochs` = 50
- `max_len` = 128 for AGNews and DBPedia, 256 for Movie Genre and 512 for Medical Abstracts.
- `learning rate` = 0.001
- `optimizer` = `Adam`

A.7 Post hoc attribution explanation methods

Captum library. Post hoc attribution has been computed using the Captum (Miglani et al., 2023) library. In particular, `Integrated gradients` has been computed with respect to language models' embedding layer with Captum's default settings. The embedding layers of BERT and DeBERTa are specified as follows: `model.model.embed_tokens`.

A.8 `CT-CBM` output examples

A.8.1 Residual connection importance evolution

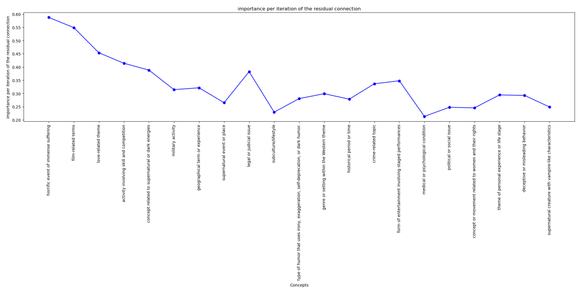

Figure 4: Residual connection importance evolution during `CT-CBM` BERT training for the movie genre dataset.

A.8.2 Examples of macro concept compositions

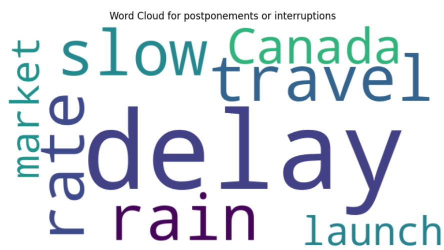

Figure 5: Cloud of micro concepts composing the macro concept "Postponements or interpuptions" from the AGNews dataset.

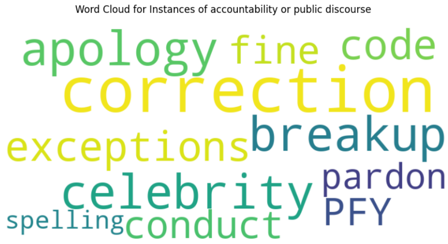

Figure 6: Cloud of micro concepts composing the macro concept "Instances of accountability or public discourse" from the AGNews dataset.

Figure 7: Cloud of micro concepts composing the macro concept "Acronyms and initials" from the AGNews dataset.

Figure 8: Cloud of micro concepts composing the macro concept "Cybersecurity and information protection" from the AGNews dataset.

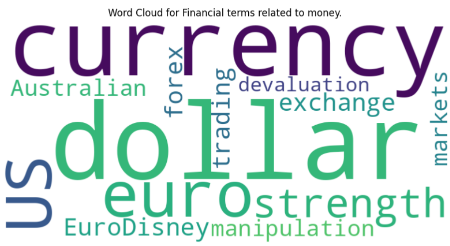

Figure 9: Cloud of micro concepts composing the macro concept "Financial terms related to money" from the AGNews dataset.

Figure 10: Cloud of micro concepts composing the macro concept "United-Nations-related" from the AGNews dataset.