Towards Achieving Concept Completeness for Textual Concept Bottleneck Models

Milan Bhan[1,2*] Yann Choho[2*] Pierre Moreau[2]
Jean-Noël Vittaut[1] Nicolas Chesneau[2] Marie-Jeanne Lesot[1]
[1]Sorbonne Université, CNRS, LIP6, F-75005 Paris, France
[2]Ekimetrics, Paris, France
{milan.bhan, yann.choho, pierre.moreau, nicolas.chesneau}@ekimetrics.com
{jean-noel.vittaut, marie-jeanne.lesot}@lip6.fr

Abstract

Textual Concept Bottleneck Models (TCBMs) 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 and identifiable concepts in the bottleneck layer to create a complete concept basis. CT-CBM achieves striking results against competitors in terms of concept basis completeness and concept detection accuracy, offering a promising solution to reliably enhance interpretability of NLP classifiers.

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 in the second category 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*, in a Concept Bottleneck Layer (CBL). These concepts are then linearly projected to make the final prediction, 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., 2024; Sun et al., 2024; Ludan et al., 2023) whose computational cost is prohibitive, (ii) they often require access to a set of predefined human-labeled concepts (Tan et al., 2024; Ludan et al., 2023), (iii) the concept base of the CBL can be over complete, making the CBM predictions based on too many concepts and difficult to understand (Tan et al., 2024; Sun et al., 2024) (iv) they do not systematically guarantee the reliability of the CBL concept detection, making the corresponding explanations unfaithful.

	C3M	CB-LLM	CT-CBM (ours)
Need for predefined concepts	Yes	**No**	**No**
Use of LLM	Yes	Yes	**No**
Scalability	No	**Yes**	**Yes**
Black-box performance reached	**Yes**	**Yes**	**Yes**
Concept base completeness	No	No	**Yes**
Accurate concept detection	No	No	**Yes**

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

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 accurately detecting concept from a complete concept basis. As summarized in Table 1, the main contributions

*These authors contributed equally to this work.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 2007–2024
November 4-9, 2025

of CT-CBM are as follows:

1. We present a computationally affordable method to perform concept discovery and annotation in a fully unsupervised manner, solely based on a small language model.

2. We propose a new method to target relevant concepts to be added in the CBL, based on local concept importance and identifiability.

3. Concept completeness, estimated as the smallest concept base to cover the dataset while enabling the TCBM to approximately replicate the performance of the initial black-box model, is achieved through iterative addition of concepts in the CBL.

This way, the CT-CBM method we propose offers both (1) an affordable concept annotation method to construct a concept bank and (2) a method to generate a TCBM derived from any initial concept bank, with an accurate CBL constructed upon a complete concept base.

The paper is organized as follows: Section 2 recalls some basic principles of XAI and related work. Section 3 describes the proposed CT-CBM. Section 4 presents the conducted experiments, that show that CT-CBM systematically succeeds in reaching the performance of its competitors in terms of downstream task accuracy and detects significantly more precisely the concepts present in its concept layer, while containing fewer concepts than its competitors.

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 in general, and for NLP.

2.1 XAI Background

Post Hoc Interpretability. Post hoc methods explain the behavior of a model after its training. They first include attribution methods that compute importance scores to identify the input dimensions that are mostly responsible for the obtained results, e.g. Integrated Gradients (Sundararajan et al., 2017).

Second, post hoc concept-based approaches generate explanations at a higher level of abstraction, by focusing on human interpretable attributes, called *concepts*. Given a concept exemplified by some user defined examples, TCAV (Kim et al., 2018) assesses the model's sensitivity to the latter by back-propagating the gradients with respect to a linear representation of the considered concept, called concept activation vector (CAV).

Concept Bottleneck Models. Another way to improve the interpretability of AI systems consists in constructing so-called interpretable-by-design Concept Bottleneck Models (CBM) (Koh et al., 2020). They sequentially detect concepts in a Concept Bottleneck Layer (CBL) and linearly make the final prediction from the latter, thereby significantly improving the understanding of the decision-making process. CBM face several limitations: (1) they require predefined human-labeled concepts, (2) their CBL are often incomplete, leading to either reduced model accuracy (under-complete concept base) or unintelligible explanations (over-complete concept base) (Shang et al., 2024), (3) they are vulnerable to downstream task leakage, where prediction models inadvertently use unintended signals from concept predictor scores rather than the actual concepts, compromising both the faithful detection of concepts and the overall interpretability of CBM (Havasi et al., 2022), (4) they do not ensure accurate concept prediction in the CBL.

Among the extensive CBM literature (Poeta et al., 2023b), several approaches have been proposed to address specific limitations. Label-Free CBM (Oikarinen et al., 2023) employs GPT-3 (Brown et al., 2020) to identify key concepts for class recognition, eliminating the need for predefined concepts. Other approaches (Yuksekgonul et al., 2023; Havasi et al., 2022) add a non-interpretable parallel residual connection to match black-box NLP classifier accuracy and mitigate leakage by processing unintended information through this residual layer, though reducing the interpretability of the CBM. Res-CBM (Shang et al., 2024) derives new concepts from the residual layer to create more complete CBLs, but still requires predefined concept candidates. While these methods overcome or mitigate certain CBM limitations, their application has primarily 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).

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.

C^{3}M (Tan et al., 2024) supplements human-labeled concepts with ChatGPT-generated concepts, achieving performance comparable to black-box NLP classifiers. However, this approach builds a TCBM with an over-complete concept basis, and its dependence on ChatGPT and human labeling limits its reproducibility and scalability.

CB-LLM (Sun et al., 2024, 2025) also uses ChatGPT to generate concept candidates that are then scored using a sentence embedding model to create numerical concept representations, making the approach more affordable than C^{3}M (Tan et al., 2024). While matching black-box classifier performance, it also neglects CBL completeness, potentially leading to unreliable concept detection and unintelligible explanations.

3 Proposed approach: CT-CBM

This section describes our proposed Complete Textual Concept Bottleneck Model (CT-CBM). As shown in Table 1, CT-CBM uses only small language models (SLM) for concept discovery, eliminating the need for predefined human-labeled concepts, and builds a TCBM with a complete concept bottleneck layer containing properly detected and relevant concepts. The code is available online on a public repository*

3.1 CT-CBM Overview

We consider a corpus of text-label pairs $\mathcal{T} = \mathcal{X} \times \mathcal{Y} = \{(x, y)\}$ where $x \in \mathcal{X}$ denotes the text and $y \in \mathcal{Y}$ the label. We respectively denote $\mathcal{X}_{train}, \mathcal{X}_{dev}$ and $\mathcal{X}_{test}$ the training, development and test sets related to the given set of texts $\mathcal{X}$. $f : \mathcal{X} \to \mathbb{R}^d$ is the backbone of a language model classifier fine-tuned on $\mathcal{T}$, where d denotes the dimension of f embedding space.

As shown in Figure 1, CT-CBM is a 4-step method that follows an iterative process, beginning with a concept base covering the text corpus of interest and progressively incorporating the most relevant concepts into the CBL until meeting a completeness stopping criterion, as detailed in Sections 3.2 to 3.5:

1. Concept Bank Construction. We generate candidate concepts by prompting an auto-regressive SLM to identify micro-concepts (topics) within $\mathcal{T}$. These micro-concepts are then clustered to form a set of high-level macro-concept candidates $\mathcal{C}$.

2. Concept Scoring and CBL Initialization. Each candidate concept in $\mathcal{C}$ receives a score based on its importance for classification and its identifiability in f embedding space. These scores determine which concepts will be incorporated into the concept bottleneck layer. The TCBM CBL is initialized (1) to ensure that nearly every text in the corpus of interest activates at least one concept and (2) to foster concept diversity in the CBL.

3. TCBM Training. Given a set of selected concepts, we train two TCBM variants: a *simple* version using only the explicit concepts and a *residual* one with an additional parallel residual connection capturing additional non interpretable information.

4. Stopping Criterion. The training process stops when the performance of the *simple* TCBM achieves comparable performance to the *residual* TCBM, indicating that the explicit concepts from the CBL alone provide a complete basis for the classification task without requiring residual information.

3.2 Concept Bank Construction

In case of absence of labeled concepts, the first step aims at automatically constructing a set of concept candidates $\mathcal{C}$ without human annotation for potential inclusion in the CBL of the TCBM.

Micro Concept Bank Creation. We use a scalable and computationally affordable small language model (9B parameters following Lu et al. (2024)) to annotate each text with topic-level "micro concepts" that represent higher-level abstractions than tokens. This creates a micro concept bank $\widetilde{\mathcal{C}}$ from the text corpus. We give more information about the prompt used to generate micro concepts in Appendix A.3.1.

Macro Concept Bank Creation. We cluster the micro concepts into p macro concepts with $p \ll |\widetilde{\mathcal{C}}|$, sequentially using sentence embeddings, UMAP (McInnes et al., 2018), and HDBSCAN (McInnes et al., 2017). This addresses semantic redundancy (e.g., grouping "demoniac

*https://github.com/yann-Choho/CT-CBM

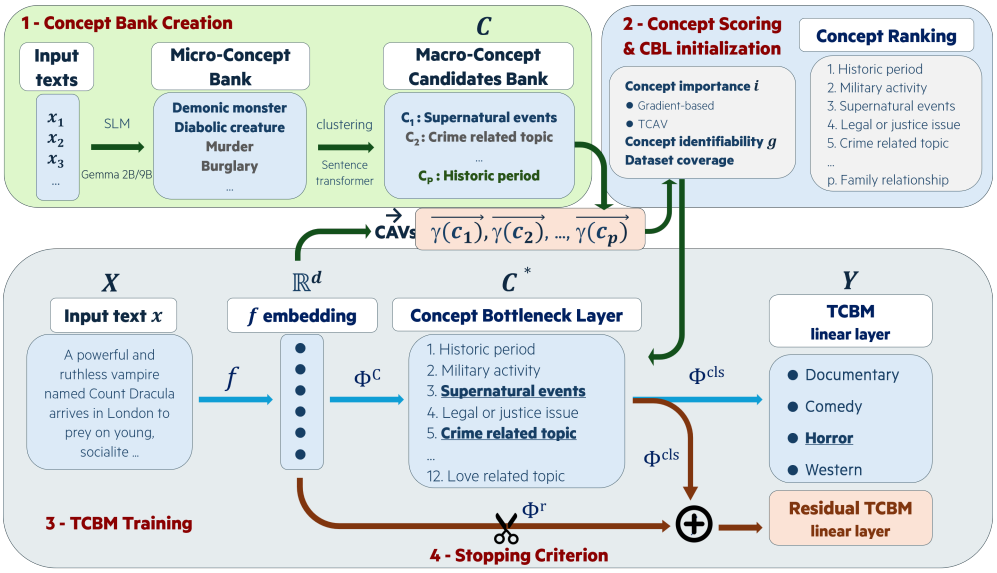

Figure 1: CT-CBM overview illustrated with film synopsis classification. CT-CBM is a 4-step approach to build a TCBM from a f black box NLP classifier. (1) A concept bank is created from the text corpus of interest. (2) Concepts are scored given their importance to explain f predictions and their identifiability score, and the CBL is initialized. (3) The TCBM is trained through 3 layers: Φ^{C}, Φ^{cls} and Φ^{r}. (4) The TCBM training stops when the performance of the TCBM with Φ^{r} is reached without the latter; Φ^{r} is finally removed.

monster" and "diabolic creature" into "supernatural entities"). The final corpus is formalized as $\mathcal{T}_M = \{(x, y, \mathbf{c})\}$, where $\mathbf{c} \in \{0,1\}^p$ is a vector of absence or presence of the p found macro concepts.

Macro Concept Labeling. Each macro concept c receives a textual label $l(c)$ by prompting the SLM to identify the superclass of the m micro concepts closest to the macro concept's centroid, with $m = 15$ by default. More details about the prompt used perform macro concept labeling are provided in Appendix A.3.2.

Thus, unlike its competitors, CT-CBM offers to perform concept discovery without using ChatGPT and without relying on human annotations.

3.3 Concept Scoring and CBL Initialization

Given any labeled concept candidate bank $\mathcal{C}$ (obtained whether through CT-CBM concept annotation as in Section 3.2, or alternatives), our objective is to determine which subset $\mathcal{C}^* \subset \mathcal{C}$ should be included in the CBL. CT-CBM scores each concept c based on linear representations from f embedding and determined by combining two key components: (1) a concept importance measure $i(c)$ and (2) a linear identifiability score $g(c)$. The CBL of the TCBM is then initialized based on these scores and by favoring diversity in the concepts.

Concept Activation Vectors Computation. 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 concept c a linear representation from f embedding space, called Concept Activation Vector (CAV) $\overrightarrow{\gamma(c)}$. Among the different ways to compute a CAV (Wu et al., 2025), we consider the mean difference of embeddings (Rimsky et al., 2024) that has been shown to lead to the best compromise is terms of concept detection accuracy and computational cost (Marks and Tegmark, 2024):

$$\overrightarrow{\gamma(c)} = \frac{1}{|\mathcal{X}_{tr}^{c+}|} \sum_{x \in \mathcal{X}_{tr}^{c+}} f(x) - \frac{1}{|\mathcal{X}_{tr}^{c-}|} \sum_{x \in \mathcal{X}_{tr}^{c-}} f(x)$$

where $\mathcal{X}_{tr}^{c+}$ and $\mathcal{X}_{tr}^{c-}$ respectively represent the sets of texts from the training set $\mathcal{X}_{train}$ where the concept c is present or absent.

Concept Importance. CT-CBM identifies concepts with high discriminative power by computing their importance. We propose a novel approach, called Concept Integrated Gradients (CIG), to estimate the latter, based on Integrated gradients (Sundararajan et al., 2017). For each input text $x \in \mathcal{X}$ and concept c,

we calculate the local concept importance using the absolute dot product: $|\langle\overrightarrow{\gamma(c)}, \texttt{IG}(f(x))\rangle|$, where $\texttt{IG}(f(x))$ is the neuron importance vector derived by applying `Integrated Gradients` to f final layer with respect to the ground truth label. This way, the local concept importance is defined as the absolute value of the projection of the f embedding local neuron importance of input x onto the concept direction $\overrightarrow{\gamma(c)}$. The final importance score $i(c)$ is the average of the local concept importance values across $\mathcal{X}_{train}$. `CT-CBM` also compute concept importance by appling `TCAV` as in Nejadgholi et al. (2022). Both approaches identify concepts that play a significant role in f decision-making process, making them valuable candidates for inclusion in the CBL. Further implementation details for both `CIG` and `TCAV` are provided in Appendix A.4.2.

Concept Linear Identifiability. `CT-CBM` identifies concepts that can effectively be detected in f embedding space by computing a linear identifiability score for each concept c. For each input text $x \in \mathcal{X}_{dev}$, we predict concept presence based on previously computed CAVs using a simple linear rule:

$$\widehat{c}(x) = \begin{cases} 1 & \text{if } \langle\overrightarrow{\gamma(c)}, f(x)\rangle > M_c \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $\langle\overrightarrow{\gamma(c)}, f(x)\rangle$ is the projection of the f embedding of input x onto the concept direction $\overrightarrow{\gamma(c)}$. The threshold M_c is the median projection value across $\mathcal{X}_{dev}$. We then compute the linear identifiability score $g(c)$ as the F1 score between the predictions $\widehat{c}(x)$ and ground truth labels on $\mathcal{X}_{dev}$. This identifiability score quantifies how accurately each concept can be linearly detected in f embedding space. We define the final concept score as $i(c) \times g(c)$, ensuring that selected concepts are both discriminative for the task and reliably detectable from f representations.

CBL Initialization. We initialize the CBL before the first TCBM training with a minimal set of concepts that covers at least 99% of $\mathcal{X}_{train}$, ensuring nearly every text activates at least one concept. To achieve this efficiently, we (1) cluster concepts based on their co-occurrence patterns in texts and using HDBSCAN, (2) sort concepts by their previously calculated scores and (3) select concepts iteratively by cluster in descending score order. This approach balances concept relevance with diversity, avoiding redundancy by prioritizing concepts from different clusters. Subsequently, during iterative TCBM training, a concept per cluster is added to the CBL.

These concept clusters should not be confused with the micro-concept clusters introduced in Section 3.2. During the TCBM initialization stage, concept clusters are groups of concepts that activate on the same texts, whereas micro-concept clusters are groups of concepts with similar semantics. In Section 4.2, we compare the clustering-based selection method against a simpler approach that selects concepts solely based on their scores.

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. We guide the evolution of the TCBM training by adding a residual connection parallel to the CBL. The residual connection serves two purposes: it acts as an indicator of concept base completeness (see Section 3.5) while simultaneously reducing downstream classification leakage. Generating a *simple* TCBM consists in training two layers, $\Phi^{\mathrm{C}} : \mathbb{R}^d \rightarrow \mathbb{R}^{|\mathrm{C}|}$ and $\Phi^{\mathrm{cls}} : \mathbb{R}^{|\mathrm{C}|} \rightarrow \mathcal{Y}$. 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 as $\Phi^{\mathrm{cls}} \circ \Phi^{\mathrm{C}} \circ f$. 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 mitigate 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:

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

where $\mathcal{L}$ is the cross-entropy loss, λ a hyperparameter and c the vector of absence or presence of the concepts included in C, i.e. the restriction to C of $\mathbf{c}$ defined on $\mathcal{C}$ in Section 3.2. 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 first and classification training performed afterwards).

3.5 Stopping Criterion

CT-CBM employs a performance-based criterion to determine when sufficient concepts have been incorporated into the CBL. At each iteration, we evaluate the two variants of our TCBM, the simple and the residual ones. Indeed, the residual layer Φ^r captures information not yet represented by the current concept base. We then approximate concept base completeness through a quantitative performance comparison: when the TCBM without Φ^r achieves at least $(1-\epsilon)\%$ times the performance of the TCBM with Φ^r, we conclude that the CBL adequately captures the essential information for classification. This criterion is reasonable because the concept base is initially constructed to cover nearly all texts in $\mathcal{X}$. Therefore, when the TCBM without Φ^r approximates the performance with Φ^r, it suggests that the current concepts capture the necessary information for the classification task, with minimal reliance on unexplained residual concepts. Once this performance threshold is met, the training process terminates, and we remove the residual layer Φ^r from the model. This final step yields a pure TCBM without any non-interpretable components, ensuring that all classification decisions are solely based on the human-understandable CBL.

4 Experimental Settings

This section presents the experimental study conducted across 5 datasets and 3 NLP classifiers of different sizes. We compare CT-CBM to several competitors, and we run an ablation study to assess the impact of (1) the concept clustering during the CBL initialization, (2) the method used to compute concept importance and (3) the concept identifiability score. Then we illustrate TCBM applications, namely concept intervention during inference, better understanding of counterfactual explanations and adversarial attacks (Lyu et al., 2024) and providing global explanations.

4.1 Experimental Protocol

Datasets and models. CT-CBM is tested on five multi-class text classification datasets: AG News (Gulli, 2005), DBpedia (Lehmann et al., 2015), Movie Genre[*] and critical domain datasets such as the legal dataset Ledgar (Tuggener et al., 2020) ans Medical Abstracts (Schopf et al., 2022). We apply CT-CBM on three fine-tuned NLP classifiers of different sizes: BERT (Devlin et al., 2019), DeBERTa-large (He et al., 2020) and Gemma-2-2B. More information about the used language models are provided in Appendix A.5.

CT-CBM and Competitors. We run CT-CBM with the 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 either CIG and TCAV. The stopping criterion parameter ϵ is set a 0.05. We compare CT-CBM to both C^{3}M (Tan et al., 2024) and CB-LLM (Sun et al., 2025). Our aim is to enrich a simple classifier to generate concepts prior to a final prediction. 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.

Since CT-CBM and C^{3}M work with binary represented concepts, their training is done either based on CT-CBM or C^{3}M concept annotation. To ensure comparability and address the ChatGPT annotation non scalability of C^{3}M (complexity proportional to size of the dataset $\times$ number of targeted concepts), we run C^{3}M with Gemma-2-9B as concept annotator. For each approach, concept annotation is done on $\mathcal{X}_{train}$, $\mathcal{X}_{dev}$ and $\mathcal{X}_{test}$, to enable to evaluate concept detection (see next paragraph). Each method is trained with an early stopping strategy applied to $\mathcal{X}_{dev}$. The hyperparameters of the experiments are detailed in Appendices A.4.5 and A.6.

Evaluation Criteria. We propose an evaluation with 3 metrics: (1) final classification task accuracy (**%ACC**), (2) concept detection accuracy (**%c**) measured by F1 score to address concept label imbalance, (3) number of concepts (**#c**) in the CBL. Due to computational constraints, C^{3}M concept detection evaluation uses Gemma-2-9B instead of ChatGPT and is limited to 4000 texts. CB-LLM concept evaluation is done by discretizing its concept prediction since the CB-LLM framework represents concepts with numerical values.

4.2 Results

Global Results. Table 2 shows the experimental results obtained by CT-CBM and its competitors on BERT and DeBERTa, where C^{3}M and CT-CBM are trained based on concepts obtained with the C^{3}M

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

Model backbone (size)		BERT-base (110M)				DeBERTa-large (395M)			
Dataset	**Method**	Black box	C^{3}M	CB-LLM	CT-CBM (ours)	Black box	C^{3}M	CB-LLM	CT-CBM (ours)
AGNews	%ACC ↑	91.0	90.1	90.0	**90.6**	92.0	91.5	90.1	**91.6**
	%c ↑	-	68.8	56.0	**74.5**	-	80.2	69.2	**88.7**
	#c ↓	-	41	41	**12**	-	41	41	**13**
DBpedia	%ACC ↑	99.4	**99.5**	99.3	**99.5**	99.4	**99.5**	99.4	**99.5**
	%c ↑	-	79.5	56.0	**92.9**	-	86.0	42.1	**99.3**
	#c ↓	-	63	63	**18**	-	63	63	**19**
Ledgar	%ACC ↑	95.6	**97.0**	95.1	96.1	95.9	**97.0**	97.2	96.8
	%c ↑	-	64.7	47.7	**80.2**	-	75.8	63.4	**86.8**
	#c ↓	-	78	78	**13**	-	78	78	**12**
Medical Abstract	%ACC ↑	62.7	56.7	**57.9**	56.7	62.6	**60.0**	59.1	**60.0**
	%c ↑	-	53.5	29.2	**62.3**	-	57.4	25.2	**76.5**
	#c ↓	-	57	57	**15**	-	57	57	**16**
Movie Genre	%ACC ↑	91.7	91.6	91.4	**92.7**	93.8	**93.8**	90.9	93.5
	%c ↑	-	68.7	29.8	**84.5**	-	77.1	52.2	**84.3**
	#c ↓	-	68	68	**14**	-	68	68	**13**

Table 2: CT-CBM and competitors evaluation on 5 test sets and 2 NLP classifiers. Except the black box baseline, the best results (resp. second best) are in bold (resp. underlined). CT-CBM and C^{3}M training are based on C^{3}M annotation.

Concept annotation		CT-CBM		CT-CBM + C3M	
Dataset	**Method**	C^{3}M	CT-CBM (ours)	C^{3}M	CT-CBM (ours)
AGNews	%ACC↑	91.1	**91.1**	90.3	**91.1**
	%c↑	**54.8**	52.1	56.2	**80.8**
	#c↓	100	**12**	141	**12**
DBpedia	%ACC↑	**99.5**	**99.5**	**99.5**	**99.5**
	%c↑	52.0	**81.1**	58.9	**91.3**
	#c ↓	100	**18**	163	**18**
Movie Genre	%ACC↑	91.3	**92.6**	91.5	**91.6**
	%c↑	45.3	**50.4**	54.5	**82.6**
	#c↓	100	**14**	168	**16**

Table 3: CT-CBM and C^{3}M evaluation on three test sets on BERT. Evaluation is done either based on CT-CBM concept annotation or the union of CT-CBM and C^{3}M concept annotations. The best result are in bold.

protocol. Overall, CT-CBM achieves a performance very similar to that of the original black box models and its competitors in terms of downstream task accuracy (%ACC). More importantly, CT-CBM achieves by far the best results both in terms of CBL size (up to 54 fewer concepts in #c) and concept detection accuracy (%c improvements of 5.7 to 33.1 points) for all datasets and models. These results are achieved by ensuring a good compromise between concept conciseness and expressiveness, since CT-CBM constructs a TCBM based on a complete concept base properly covering the text corpus. CT-CBM performs well both on datasets from general domains (AGnews, DBpedia and Movie Genre) and more technical critical domains (Ledgar and Medical Abstract). We give additional results in Appendix A.9, Table 6 showing that CT-CBM works well when applied to the Gemma-2-2B classifier on AGNews, DBpedia and the Movie Genre dataset.

Varying the Concept Annotation Method. Table 3 shows the experimental results obtained by appling CT-CBM and C^{3}M on a BERT base model. In this table CT-CBM and C^{3}M are either trained based on CT-CBM annotation or in a case where concept annotations from both methods are available. Except for the AGnews dataset, CT-CBM always over-performs C^{3}M in terms of concept detection accuracy, with drastically less concepts in its CBL. These results show that CT-CBM is robust to the concept base it is given as input, almost always leading to the best results. While CT-CBM annotation is computationally less costly than C^{3}M (number of texts vs number of texts $\times$ number of concepts), it results in concepts that are more difficult to classify (e.g. 52.1%c vs 74.5%c for BERT/Agnews). This way, the concept annotation method has to be set considering computational budget constraints.

We give additional results comparing CT-CBM and C^{3}M based on CT-CBM annotation when applied to DeBERTa in Appendix A.9, Table 7. Table 9 also contains results based on gemma-2-2b as concept annotator model, showing that the latter performs on pair with gemma-2-9B and outperforms C^{3}M with a concept annotation based on gemma-2-9B. This result highlights the robustness of CT-CBM and its scalability.

Ablation Study. Table 4 shows the experimental results of the CT-CBM ablation study on BERT based on C^{3}M annotation. We vary concept clustering

Dataset	CT-CBM version	%ACC ↑	%c ↑	#c ↓
AGnews	CC-CIG-I	90.6	74.5	**12**
	$\overline{\text{CC}}$-CIG-I	90.4	**83.9**	19
	CC-CIG-$\bar{\text{I}}$	**91.2**	72.5	14
	CC-TCAV-I	90.6	70.1	**12**
DBpedia	CC-CIG-I	**99.5**	**92.9**	**18**
	$\overline{\text{CC}}$-CIG-I	99.4	91.1	26
	CC-CIG-$\bar{\text{I}}$	99.3	91.4	**18**
	CC-TCAV-I	**99.5**	92.5	**18**
Movie Genre	CC-CIG-I	**92.7**	**84.5**	**14**
	$\overline{\text{CC}}$-CIG-I	92.1	76.2	24
	CC-CIG-$\bar{\text{I}}$	91.2	76.4	13
	CC-TCAV-I	92.0	75.1	**14**

Table 4: CT-CBM ablation study of concept clustering (CC), concept importance (either CIG or TCAV) and concept identifiability score (I). $\overline{\text{CC}}$ and $\bar{\text{I}}$ respectively stand for the cases without concept clustering and without concept identifiability computation.

(CC), local concept importance (CIG, TCAV) and identifiability scoring (I). For each dataset, the CT-CBM basic settings (CC-CIG-I) give the best compromise in terms of downstream task accuracy, concept detection and CBL size. Not clustering concept ($\overline{\text{CC}}$) when initializing the concept base results in many more concepts in the CBL (e.g. 26 vs 18 for DBpedia). Not using the identifiability score leads in average to less accurate concept detection (e.g. 76.4 vs 84.5 for Movie Genre). Using CIG gives slightly better results than using TCAV. These results validate the interest of (1) concept clustering to efficiently cover the text corpus and (2) CIG and identifiability scoring for accurate concept detection.

We also compare with a random baseline on 10 runs in Appendix A.9, Table 8, highlighting that TCAV and CIG generate better results in terms of concept accuracy than randomly selected concepts. Additionally, we show in Table 10 that CIG computation time is significantly faster than TCAV, reinforcing the interest of using CIG to compute concept importance in our framework.

4.3 Practical Applications of TCBM

TCBM Intervention. A common application of CBM is to make domain experts modify concept activations at test time to improve the final task accuracy (Steinmann et al., 2024). We show in Table 5 how CT-CBM concept intervention during inference improves the accuracy, from 92.7% to 94.0% on the Movie Genre dataset and 90.6% to 91.8% on AGNews.

Number of interventions	CT-CBM Performance	
	AGnews	Movie Genre
0	90.6	92.7
1	91.3	93.6
2	91.6	93.8
3	91.7	**94.0**
4	**91.8**	**94.0**

Table 5: Performance of CT-CBM applied to BERT with respect to the number of interventions during inference on AGnews and the Movie Genre dataset.

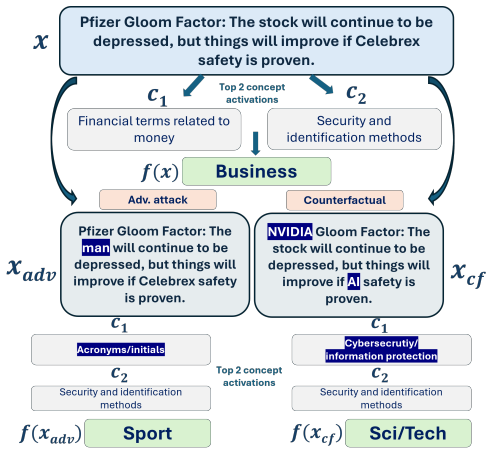

Figure 2: Example of an adversarial attack (left, x_{adv}) and a counterfactual explanation (right, x_{cf}) obtained from CT-CBM on AGnews. TCBM enables to understand the label change in terms of concept change.

Better Understanding Adversarial Attacks and Counterfactual Explanations. We propose to use TCBM for analyzing adversarial attacks and counterfactual explanations by focusing on the concept modifications enabling the prediction switch. As shown in Figure 2, we use TextAttack (Morris et al., 2020) and Claude 3.5 Sonnet, to generate examples that successfully switch the TCBM predictions on AGnews.

An adversarial attack flips a prediction from "Business" to "Sport" by changing the token "stock" to "man". TCBM highlight that this change is interpreted as concept shift from "Financial terms related to money" to "Acronyms/initials", revealing a concept misunderstanding. Similarly, counterfactual changes from the tokens "Pfizer" to "NVIDIA" and "Celebrex" to "AI" flip the label from "Business" to "Sci/Tech", highlighting a concept shift from "Financial terms" to "Security and identification methods". This concept-level analysis provides deeper understanding than token-level examination alone.

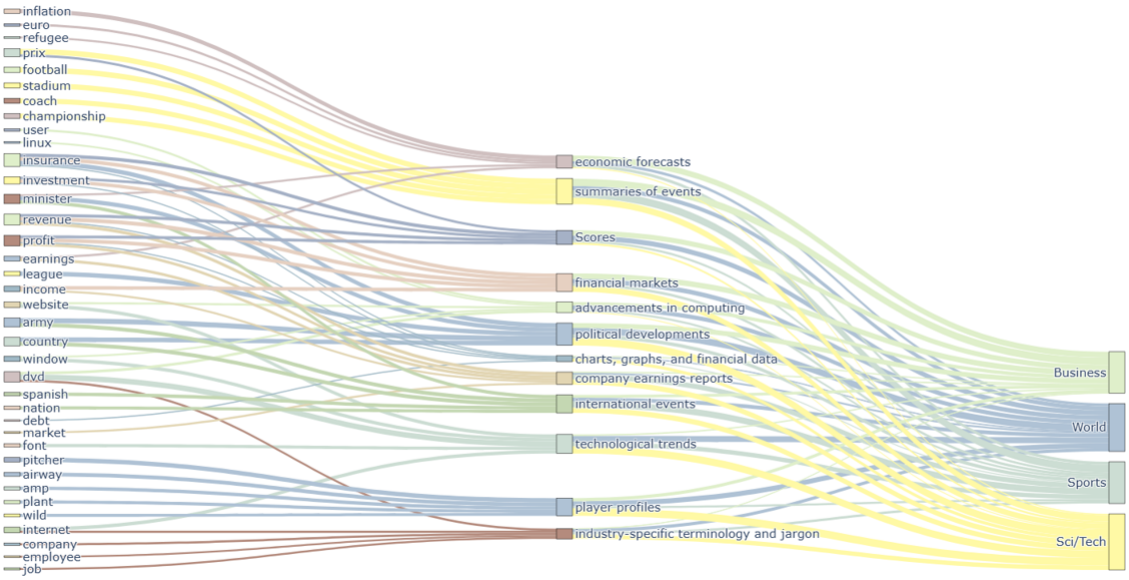

Figure 3: Global explanation of a TCBM trained on the AGnews dataset with the CT-CBM method.

Global TCBM Interpretability. We finally propose to exploit TCBM at a global scale, by representing the relationship between tokens, concepts and target classes. To identify important tokens for concept activation, we apply Integrated Gradients (Sundararajan et al., 2017) and average the resulting importance scores by concept. The weights of the Φ^{cls} layers allow to directly represent concept-to-label relationships. This principle is illustrated with Figure 3.

5 Conclusion

We introduced CT-CBM, a novel approach to transform a fine-tuned NLP classifier into a Textual Concept Bottleneck Model. CT-CBM automatically generates, scores and targets concepts to build a complete Concept Bottleneck Layer. CT-CBM replicates the same downstream classification performance than its competitors in normal and critical domains dataset, while generating a complete concept base with drastically less concepts, leading to significantly more accurate concept detection. Moreover, we highlighted several advantages of TCBM, such as intervening on concepts to improve performance, increasing the intelligibility of adversarial attacks and counterfactuals and producing global explanations.

6 Limitations

Datasets and models. This work tested CT-CBM on five datasets and three language models. It would be interesting to include other models in the study.

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). Using these approaches would enable CT-CBM to better target important concepts to be added to the CBL.

Text Generation. Recent work has focused on text generation to 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 Statement

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, and 1 others. 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, and 12 others. 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, and 1 others. 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 preprint*. 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, and 1 others. 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.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, and 1 others. 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, and 1 others. 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.

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, and 1 others. 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, and 1 others. 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*.

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.

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

Don Tuggener, Pius Von Däniken, Thomas Peetz, and Mark Cieliebak. 2020. Ledgar: a large-scale multi-label corpus for text classification of legal provisions in contracts. In *Proceedings of the twelfth language resources and evaluation conference*, pages 1235–1241.

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, and 1 others. 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, and 1 others. 2022. Concept embedding models. In *NeurIPS 2022-36th Conference on Neural Information Processing Systems*.

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 Concept Importance Implementation

The `TCAV` to compute concept-based explanations is done as in Nejadgholi et al. (2022) by focusing on the final layer of f, based on the previously computed CAV $\overrightarrow{\gamma(c)}$. The importance score $i(c)$ is calculated by aggregating the fraction of inputs influenced by each concept with respect to TCAV across all ground truth target classes. Formally, `TCAV` is locally computed as $\langle\overrightarrow{\gamma(c)}, \nabla f_{cls,k}(f(\mathbf{x}))\rangle$ with $f_{cls,k}$ the classification layer of the initial model, coming after the f backbone, related to the ground truth class sk. In the same way, `CIG` can be locally formally defined as $|\langle\overrightarrow{\gamma(c)}, \texttt{IG}(f(x))\rangle|$, where $\texttt{IG}(f(x))$ is defined along dimension i as $\texttt{IG}_i(f(x)) = (f_i(x) - f_i(x')) \times \int_0^1 (\nabla_i f_{cls,k}(f_i(x') + \alpha \times (f_i(x) - f_i(x'))d\alpha)$, with x' a baseline point defined as text with padding only and $f_i(x)$ and $f_i(x')$ the i-th neuron of $f(x)$ and $f(x')$.

A.4.3 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.4.4 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:

$$R(W) = \lambda_R \|W\|_2^2 \tag{3}$$

with $W \in \mathbb{R}^{d \times p}$ the weight matrix of the Φ^r layer, λ_R an hyperparameter and $\|\cdot\|_2^2$ the L_2 norm.

The elastic net penalization EN can be written as follows:

$$EN(A) = \lambda_{EN} \left(\alpha \|A\|_1 + (1-\alpha)\|A\|_2^2\right) \tag{4}$$

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.4.5 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.4.6 TCBM construction with CAV projection

The `projection` approach to build Φ^{C} consists in projecting the CAVs into the concept space. We formally define $\Phi^{\mathrm{c}_k}(f(x)) = \frac{\langle f(x), \overrightarrow{\gamma(c)} \rangle}{||f(x)||.||\overrightarrow{\gamma(c)}||}$ as the linear projection of the embedding of x from f on the concept space associated to concept c. 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 and the final layer. Finally, Φ^{cls} and Φ^{r} are trained to perform the classification by minimizing the following loss function:

$$\mathcal{L}_{TCBM} = \mathcal{L}(\Phi^{\mathrm{cls}}(\Phi^{\mathrm{C}}(f(x)) + \Phi^{r}(f(x)), y) \quad (5)$$

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 AGnews, DBpedia, Movie Genre and Medical Abstracts are respectively 4000, 6000, 4000 and 5000. The size of the test sets for AGnews, 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., 2024) 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` is 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 Examples of `CT-CBM` Macro Concept Compositions

Figures 3 to 8 illustrates examples of micro-concepts clusters, thus illustrating several macro concepts with their assigned label.

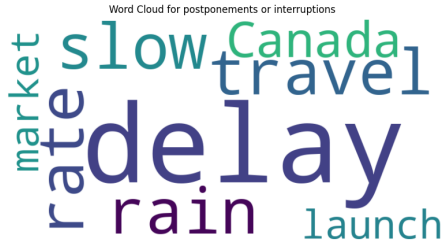

Figure 4: Cloud of micro concepts composing the macro concept "Postponements or interputions" from the AGnews dataset.

Figure 5: Cloud of micro concepts composing the macro concept "Instances of accountability or public discourse" from the AGnews dataset.

Figure 6: Cloud of micro concepts composing the macro concept "Acronyms and initials" from the AGnews dataset.

Figure 7: Cloud of micro concepts composing the macro concept "Cybersecurity and information protection" from the AGnews dataset.

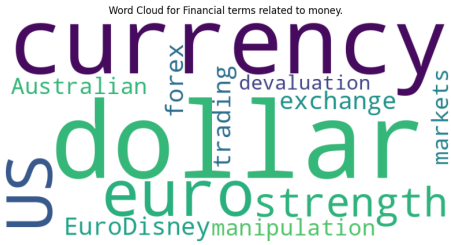

Figure 8: Cloud of micro concepts composing the macro concept "Financial terms related to money" from the AGnews dataset.

Figure 9: Cloud of micro concepts composing the macro concept "United-Nations-related" from the AGnews dataset.

A.9 Additional Experimental Results

In this section we show additional results to highlight the performance of `CT-CBM`. Table 7

shows the performance of CT-CBM and C^{3}M when applied to both BERT and DeBERTa on AGnews, DBpedia and Movie Genre, based on the CT-CBM concept annotation. Table 8 shows the ablation study of CT-CBM on AGnews, DBpedia and Movie Genre on BERT. Concept clustring (CC), local concept importance (CIG, TCAV and random) and concept identifiability. Results for random are shown by computing the average and the standard deviation on 10 TCBM trainings.

Dataset	Method	Black-box Gemma-2-2B	CT-CBM Gemma-2-2B
AGNews	%ACC ↑	85.7	91.1
	%c ↑	-	83.9
	#c ↓	-	12
DBpedia	%ACC ↑	95.3	99.2
	%c ↑	-	99.5
	#c ↓	-	19
Movie Genre	%ACC ↑	84.7	89.5
	%c ↑	-	74.7
	#c ↓	-	12

Table 6: CT-CBM performance when applied to the Gemma-2-2B classifier on AGNews, DBPedia and the Movie Genre dataset.

Model backbone (size)		BERT-base (110M)		DeBERTa-large (395M)	
Dataset	**Method**	C^3M	CT-CBM (ours)	C^3M	CT-CBM (ours)
AGNews	%ACC ↑	91.1	**91.1**	**92.1**	91.2
	%c ↑	**54.8**	52.1	55.0	**55.4**
	#c ↓	100	**12**	100	**13**
	%D ↑	78.5	**79.3**	78.5	**79.1**
DBpedia	%ACC ↑	**99.5**	**99.5**	**99.5**	**99.4**
	%c ↑	52.0	**81.1**	53.0	**76.0**
	#c ↓	100	**18**	100	**19**
	%D ↑	80.3	**77.5**	**80.3**	76.9
Movie Genre	%ACC ↑	91.3	**92.6**	92.4	**92.7**
	%c ↑	45.3	**50.4**	51.5	**52.1**
	#c ↓	100	**14**	100	**13**
	%D ↑	78.8	**78.9**	**78.8**	**78.1**

Table 7: CT-CBM and C^3M evaluation on three test sets and two NLP classifiers based on the CT-CBM concept annotation. The best results are highlighted in bold.

Dataset	CT-CBM version	%ACC ↑	%c ↑	#c ↓
AGnews	**CC-CIG-I**	90.6	74.5	**12**
	$\overline{\text{CC}}$-CIG-I	90.4	**83.9**	19
	CC-CIG-$\bar{\text{I}}$	**91.2**	72.5	14
	CC-TCAV-I	90.6	70.1	**12**
	CC-random-I	90.5$\pm$0.9	73.9$\pm$4.3	13.5$\pm$2.1
DBpedia	**CC-CIG-I**	**99.5**	**92.9**	**18**
	$\overline{\text{CC}}$-CIG-I	99.4	91.1	26
	CC-CIG-$\bar{\text{I}}$	99.3	91.4	**18**
	CC-TCAV-I	**99.5**	92.5	**18**
	CC-random-I	**99.5**($\pm$**0.3**)	78.6$\pm$11.6	18.9$\pm$1.0
Movie Genre	**CC-CIG-I**	**92.7**	**84.5**	**14**
	$\overline{\text{CC}}$-CIG-I	92.1	76.2	24
	CC-CIG-$\bar{\text{I}}$	91.2	76.4	13
	CC-TCAV-I	92.0	75.1	**14**
	CC-random-I	91.7$\pm$0.3	71.9$\pm$2.8	14.1$\pm$2.0

Table 8: CT-CBM ablation study of concept clustering (CC), concept importance (either CIG, TCAV or random) and concept identifiability score (I). The random importance score is a score randomly generated, run 10 times to compute the average and the standard deviation. $\overline{\text{CC}}$ and $\bar{\text{I}}$ respectively stand for the cases without concept clustering and without concept identifiability computation during concept scoring.

Concept annotation		CT-CBM			C^{3}M		
Annotator model		gemma-2-9B		gemma-2-2B	gemma-2-9B		gemma-2-2B
Dataset	Training method	C^{3}M	CT-CBM	CT-CBM	C^{3}M	CT-CBM	CT-CBM
DBpedia	%ACC ↑	**99.5**	**99.5**	**99.5**	**99.5**	**99.5**	**99.5**
	%c ↑	52.0	**81.1**	69.0	79.5	92.9	**94.1**
	#c ↓	100	**18**	20	63	**18**	20
Ledgar	%ACC ↑	95.1	-	**96.5**	**97.0**	96.1	95.9
	%c ↑	49.9	-	**59.9**	64.7	80.2	**80.8**
	#c ↓	94	-	**19**	78	13	**12**

Table 9: CT-CBM results either obtained through C^{3}M or CT-CBM annotation with gemma-2-2B and gemma-2-9B annotator models.

Dataset	TCAV	CIG
AGNews	3.70	**0.71**
DBpedia	1.92	**0.68**
Ledgar	4.12	**0.73**
Medical Abstract	4.40	**0.70**
Movie Genre	4.78	**0.71**

Table 10: CT-CBM computation time (seconds) either based on TCAV or CIG. Our contribution (CIG) is significantly faster (approximately 5 times) than TCAV to compute concept importance.