TGML Lab | SUT | Winter 1403

An Exploration of Concept-Based Interpretability Methods

Comparison of 5 Concept-Based Papers from NeurIPS, ICLR, ACL

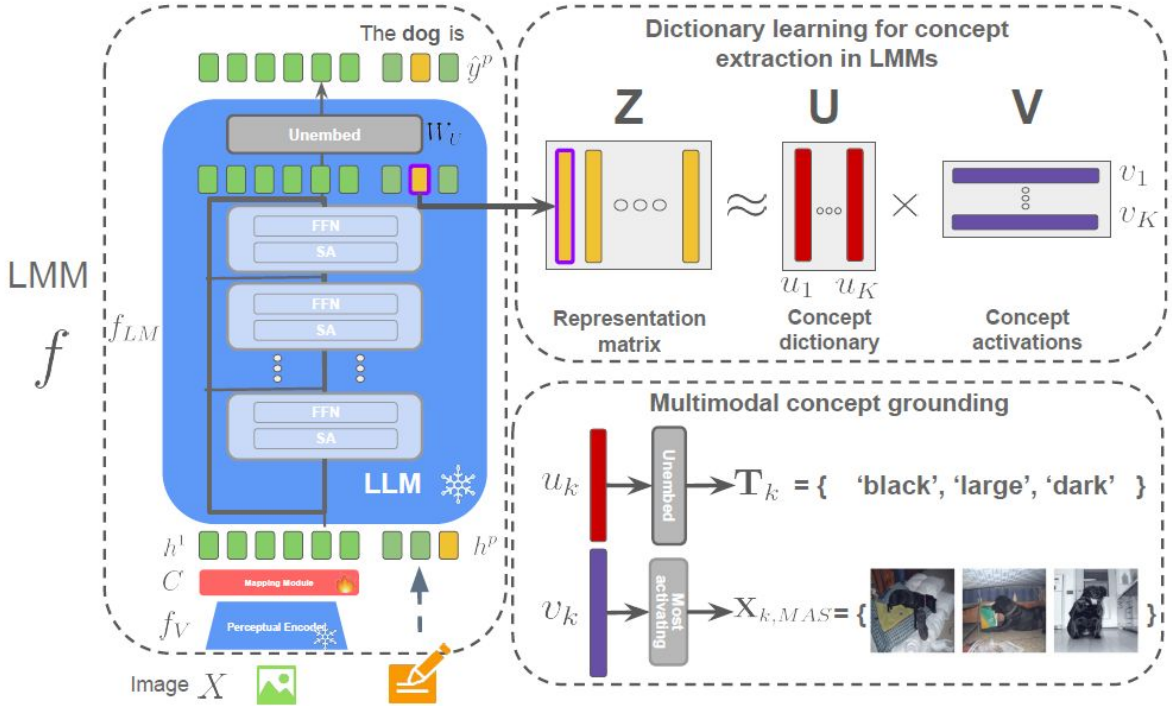
Presentation by: **Maryam Rezaee**

01

NeurIPS 2024

A Concept-Based Explainability Framework for Large Multimodal Models (CoX-LMM)

1.1 Overview


The dog is
LMM
Unembed
FFN
SA
LLM
Mapping Module
Perceptual Encoder
Image
Dictionary learning for concept extraction in LMMs
Z
U
V
Representation matrix
Concept dictionary
Concept activations
Multimodal concept grounding
Unembed
'black', 'large', 'dark'
Most activating

1.2 Preliminary

Model Architecture

- Contains: Visual encoder f_V; Trainable connector C; LLM f_{LM}
- Internal representation of token at layer l and position p inside f_{LM} is denoted as $h^p_{(l)}$, with $h^p_{(0)} = h^p$
- Input sequence for f_{LM} is concatenation of:
  - N_V visual tokens provided by the visual encoder f_V operating on an image X, followed by the connector C
  - linearly embedded textual tokens previously predicted by f_{LM}

$$\hat{y}^p = f_{LM}(h^1, h^2, \ldots, h^{N_V}, \ldots, h^p)$$

Where: $h^1, \ldots, h^{N_V} = C(f_V(X))$, h^{N_V+1} is [SOS] and $h^p = \mathrm{Emb}(\hat{y}^{p-1})$

1.2 Preliminary

Training

- f_V and f_{LM} are frozen, C is trained

Transformer Representations View

- Residual stream view of decoder-only transformers:

$$h^p_{(l+1)} = h^p_{(l)} + a^p_{(l)} + m^p_{(l)}$$

1.3 Method

Overview

- Given a pretrained LMM f and a token of interest $t \in Y$:
  - Select a subset of images X from dataset S, relevant for target token t.
  - Extract representations by processing samples in X through f and collect extracted representations of dimension B in a matrix $Z \in R^{B \times M}$, where M is number of samples in X.
  - Linearly decompose $Z \approx UV$, which includes a dictionary of learnt concepts $U \in R^{B \times K}$ of size K and coefficient/activation matrix $V \in R^{K \times M}$.
  - Semantically ground the learnt “multimodal concepts”, contained in dictionary U in both visual and textual modalities.

1.3 Method

Overview

- The main objective is to understand internal representations of an LMM and not to interpret the output of the model, which can be accomplished by combining this pipeline with some concept importance estimation method.

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

  https://proceedings.neurips.cc/paper_files/paper/2023/hash/abf3682c9cf9245a0294a4bebe4544ff-Abstract-Conference.html

1.3 Method

Representation Extraction

- Extract samples:

$$\mathbf{X} = \{X_i \mid t \in f(X_i) \text{ and } t \in y_i \text{ and } (X_i, y_i) \in \mathcal{S}\}$$

- Given any $X \in \boldsymbol{X}$, extract the residual stream representation $h^p_{(L)}$ from a deep layer L, at the first position in the predicted caption $p > N_V$, such that $\hat{y}^p = t$.
- Stacked representations $z_j \in R^B$ of each sample $X_j \in \boldsymbol{X}$ as columns of the matrix $Z = [z_1, ..., z_M] \in R^{B \times M}$

1.3 Method

Representation Decomposition

- The representation matrix $Z \approx UV$, is decomposed as product of two low-rank matrices $U \in R^{B \times K}$, $V \in R^{K \times M}$
- The columns of $U = [u_1, ..., u_K]$ are the basis vectors aka concept vectors/concepts. The rows of V or columns of $V^T = [v_1, ..., v_K]$, $v_i \in R^M$ denote the activations of u_i for each sample.
- The optimization problem to decompose Z via Semi-NMF:

$$\mathbf{U}^*, \mathbf{V}^* = \arg\min_{\mathbf{U},\mathbf{V}} ||\mathbf{Z} - \mathbf{UV}||_F^2 + \lambda||\mathbf{V}||_1 \quad s.t.\ \mathbf{V} \geq 0, \text{ and } ||u_k||_2 \leq 1\ \forall k \in \{1, ..., K\}$$

1.3 Method

Representation Decomposition ***(cont.)***

- **How to derive activations for each image?** Given a sample X, z_X is projected on U^* to compute $v(X)$:

$$v(X) = \arg\min_{v \geq 0} ||z_X - \mathbf{U}^* v||_2^2 + \lambda ||v||_1$$

- **How to find textual equivalent of concepts?** Since the concept vectors are defined in the token representation space of f_{LM}, we can leverage the insights from "Lens" based methods that attempt to understand LLM representations using the model's unembedding layer W_U.
- **Method?** Map using $W_U u_k$ and extract tokens with highest probability, then filter for english, non-stop-word words with at least 3 characters. The final set of words is denoted as T_k.

1.3 Method

Using Concept Dictionary for Interpretation

- To find images from **X** that maximally activate concept u_k (MAS):

$$\mathbf{X}_{k,MAS} = \underset{\hat{X}\subset\mathbf{X},|\hat{X}|=N_{MAS}}{\operatorname{argmax}} \sum_{X\in\hat{X}} |v_k(X)|$$

- To find most activating concepts for image X:

$$\tilde{u}(X) = \{u_k \in U^* | k \in \arg \max_{k\in\{1,\ldots,K\},|S|=r} |v_k(X)|\}$$

This step could be further combined with concept importance estimation techniques to interpret the model's prediction for token *t*.

1.4 Results

Token	Simple	PCA	KMeans	Semi-NMF
Dog	0.371	**0.004**	0.501	<u>0.086</u>
Bus	0.622	**0.002**	0.487	<u>0.177</u>
Train	0.619	**0.015**	0.367	<u>0.107</u>
Cat	0.452	**0.000**	0.500	<u>0.146</u>

Table 2: Overlap between learnt concepts. Lower is better. Best score in **bold**, second best <u>underlined</u>.

1.4 Results

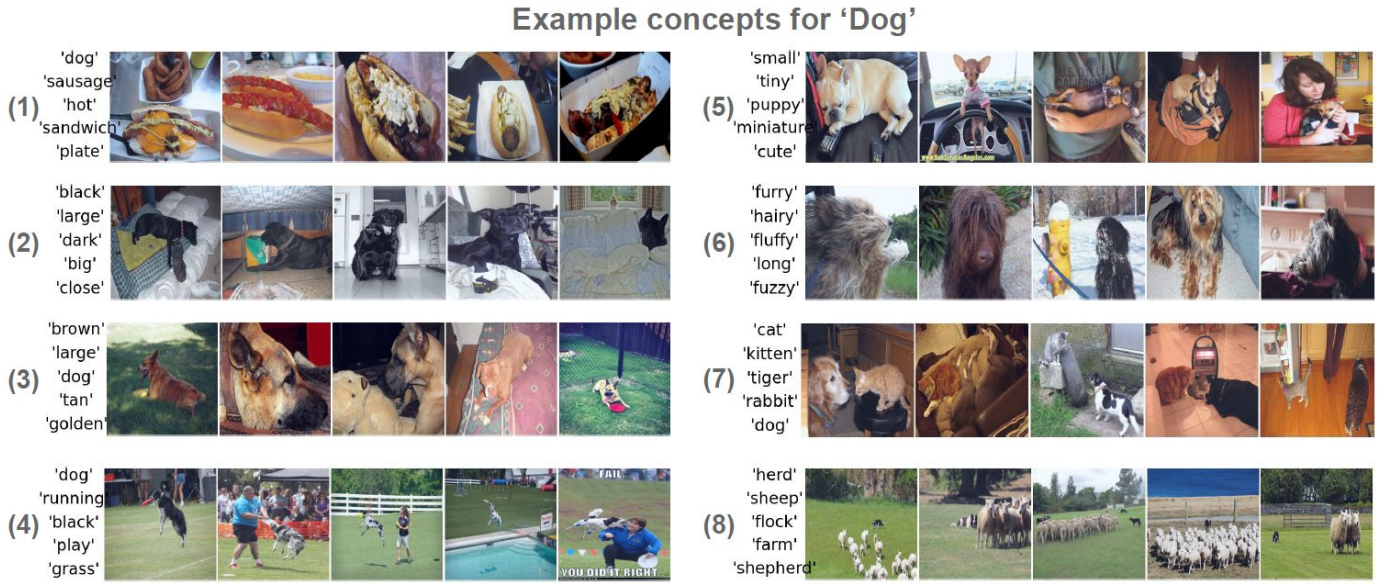


Figure 4: Visual/textual grounding for 8 out of 20 concepts for ’Dog’ token (layer 31). For each concept we illustrate the set of 5 most activating samples and 5 most probable decoded words.

1.4 Results

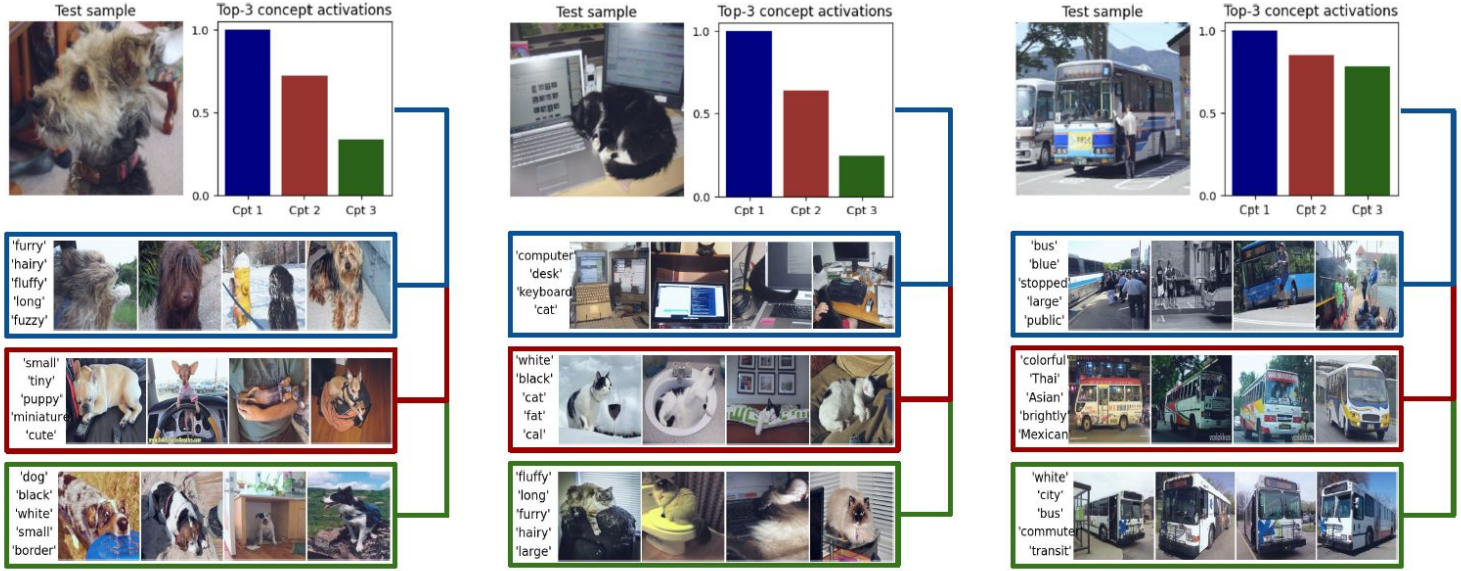


Figure 5: Local interpretations for test samples for different tokens (‘Dog’, ‘Cat’, ‘Bus’) with Semi-NMF (layer 31). Visual/text grounding for three highest concept activations (normalized) is shown.

1.4 Results

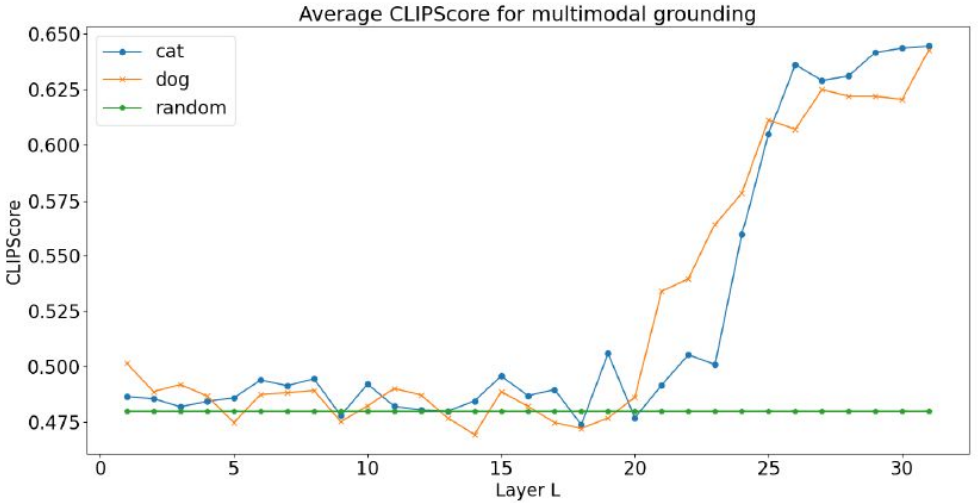


Figure 6: Mean CLIPScore between visual/text grounding $\mathbf{X}_{k,MAS}, \mathbf{T}_k$ for all concepts (Semi-NMF), across different layers L. Results are for tokens 'Dog' and 'Cat'.

02

ICML 2023

Probabilistic Concept Bottleneck Models

2.1 Overview

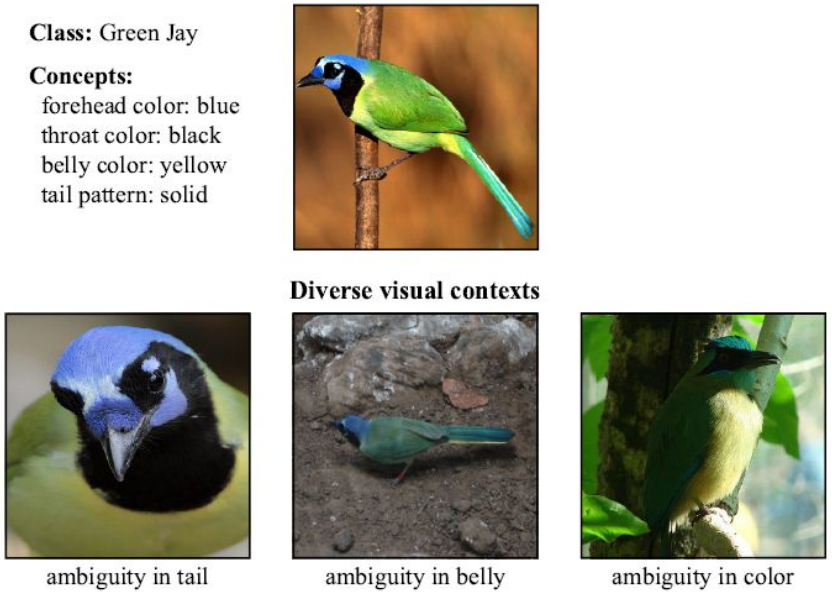


Figure 1. Examples of ambiguous cases in the existence of the concept. Images have diverse visual contexts, where partial concepts may become invisible and unclear.

2.1 Overview

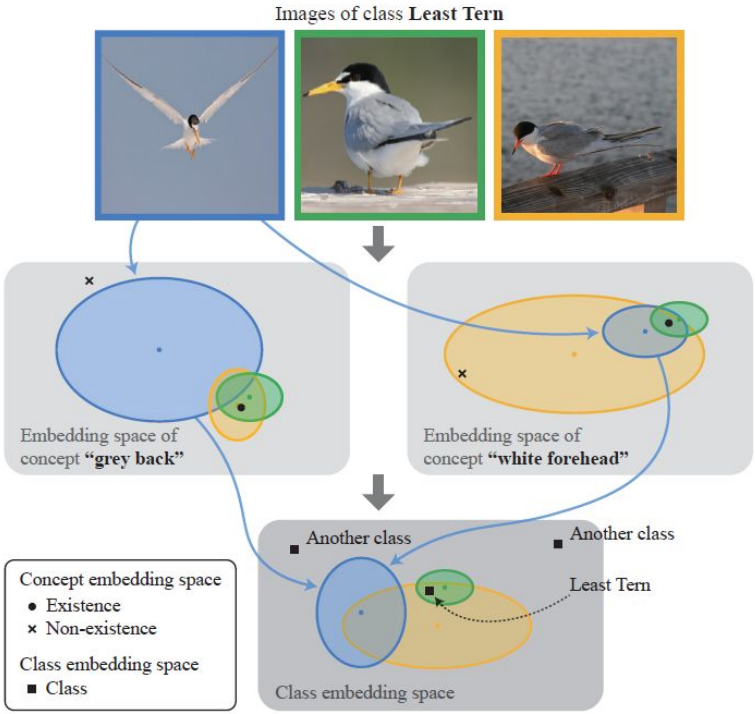


Figure 2. Probabilistic embeddings in ProbCBM. Images are

2.1 Overview

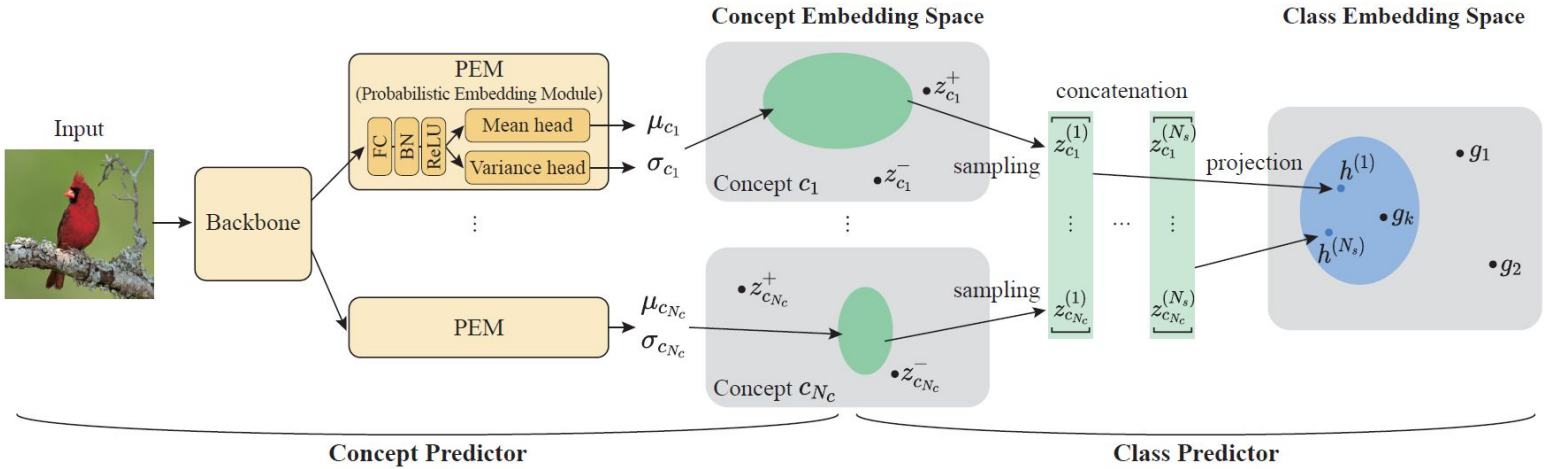


Figure 3. Prediction flow of ProbCBM. Ellipses represent probabilistic embeddings of a given input.

2.2 Preliminary

Background

- CBM (Koh et al., 2020) consists of two predictors: a concept predictor and a class predictor.
- Given an input *x*, the concept predictor *g* maps it to a concept space to predict a set of the existing concepts *C* ($\hat{C} = g(x)$). The class predictor *f* estimates the class y from $\hat{C}$, i.e., the concepts predicted by the concept predictor ($\hat{y} = f(\hat{C})$).
- To learn the mapping *g*, CBM requires the dataset {x(i), C(i), y(i)} while the conventional classifier that directly maps *x* to *y* require the dataset {x(i), y(i)}

2.3 Method

Overview

- ProbCBM: an interpretable model that exploits probabilistic embedding in prediction. It provides concept prediction and concept uncertainty as explanations.

- It adds an interpretable layer into the architecture!

2.3 Method

Probabilistic Concept Embedding

- Given an input x, the concept predictor makes probabilistic concept embedding for each concept c ∈ C, which is formulated as a normal distribution with a mean vector and a diagonal covariance matrix:

$$p(z_c|x) \sim \mathcal{N}(\mu_c, \text{diag}(\sigma_c))$$

- μc and σc are predicted by a probabilistic embedding module (PEM) from a feature map extracted from the backbone. Self-attention-based mean and variance head submodules are used in PEM. To reduce the feature dimension, an FC layer was added followed by batch norm and ReLU activation before the mean and variance head modules.

2.3 Method

Concept Prediction

- With N_s representations sampled from $p(z_c|x)$, the probability of the existence of concept c (c = 1) is obtained via Monte-Carlo estimation:

$$p(c=1|x) \approx \frac{1}{N_s} \sum_{n=1}^{N_s} p(c=1|z_c^{(n)}),$$

$$p(c=1|z_c^{(n)}) = s\left(a\left(||z_c^{(n)} - z_c^-||_2 - ||z_c^{(n)} - z_c^+||_2\right)\right),$$

- Trainable anchors z^+_c and z^-_c represent existence and non-existence of concept *c*. For a sampled representation $z^{(n)}_{c'}$ the probability that the concept exists is the Euclidean distance with z^+_c and z^-_c.

2.3 Method

Class Embedding

- To generate a class embedding from conceps, concatenate sampled concept representations from all concepts, then project with FC:

$$h^{(n)} = \mathbf{w}^T \left(\left[z_{c_1}^{(n)}, z_{c_2}^{(n)}, \ldots, z_{c_{N_c}}^{(n)} \right] \right) + \mathbf{b}$$

Class Prediction

- Classification probability is obtained via Monte-Carlo estimation, based on Euclidean distance between the class embedding and a trainable anchor point for class *k*:

$$p(y_k = 1|x) \approx \frac{1}{N_s} \sum_{n}^{N_s} \frac{\exp(-d\|h^{(n)} - g_k\|_2)}{\sum_{k'} \exp(-d\|h^{(n)} - g_{k'}\|_2)}$$

2.3 Method

Training

- Add a KL divergence loss between the predicted concept embedding distributions and the standard normal distribution:

$$\mathcal{L}_{\text{concept}} = \mathcal{L}_{\text{BCE}} + \lambda_{\text{KL}}\mathcal{L}_{\text{KL}} \qquad \mathcal{L}_{\text{KL}}(c) = \text{KL}\left(\mathcal{N}(\mu_c, \text{diag}(\sigma_c)) || \mathcal{N}(0, I)\right)$$

- Train the concept predictor and class predictor separately. First, the concept predictor is trained with $L_{concept}$. Then, the class predictor is trained with L_{class} using the concept embeddings predicted by the concept predictor.

Inference

- Done via MC sampling or using μc as zc without sampling.

2.4 Results

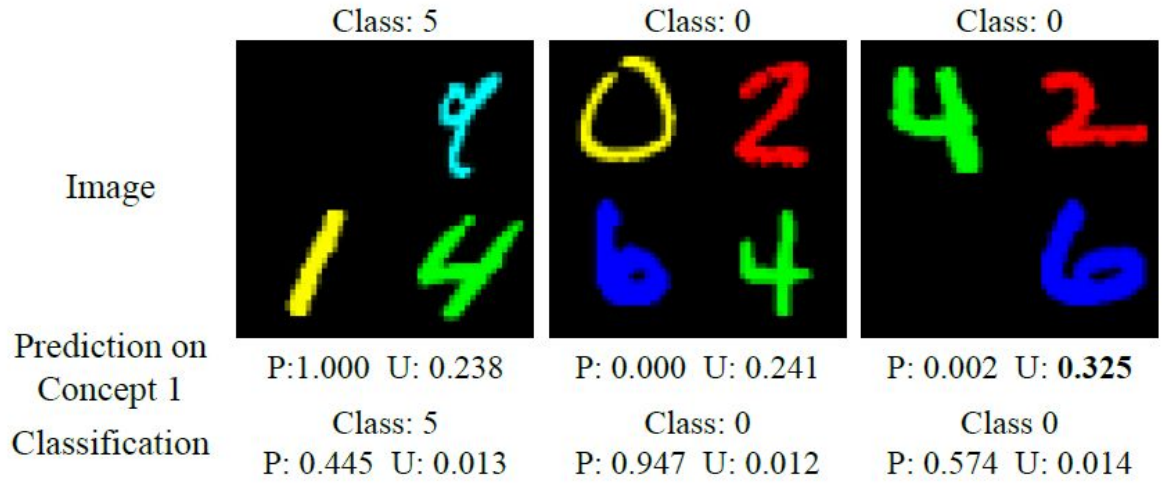


Figure 4. Examples of ProbCBM's prediction in diverse visual contexts. P and U represent the probability and uncertainty, respectively. The highest value of concept uncertainty is bolded.

2.4 Results

Table 1. Prediction accuracy for the CUB and AwA2 datasets. Results include mean values with standard deviation from three experiment repetitions. "w/o prob." denotes ProbCBM with deterministic embeddings. "w/o sampling" denotes inference without sampling.

DATA	METHOD	CONCEPT	CLASS
CUB	IMAGE: 224 × 224		
	CBM	0.950±0.001	0.670±0.006
	PROBCBM	0.949±0.001	0.680±0.004
	W/O PROB.	0.950±0.001	0.677±0.004
	W/O SAMPLING	0.949±0.001	0.679±0.003
	IMAGE: 299 × 299		
	CBM	0.956±0.001	0.708±0.006
	CEM	0.954±0.001	0.759±0.002
	PROBCBM	0.956±0.001	0.718±0.005
AWA2	CBM	0.975±0.001	0.877±0.004
	PROBCBM	0.975±0.000	0.880±0.002
	W/O PROB.	0.975±0.000	0.880±0.002
	W/O SAMPLING	0.975±0.000	0.880±0.001

2.4 Results

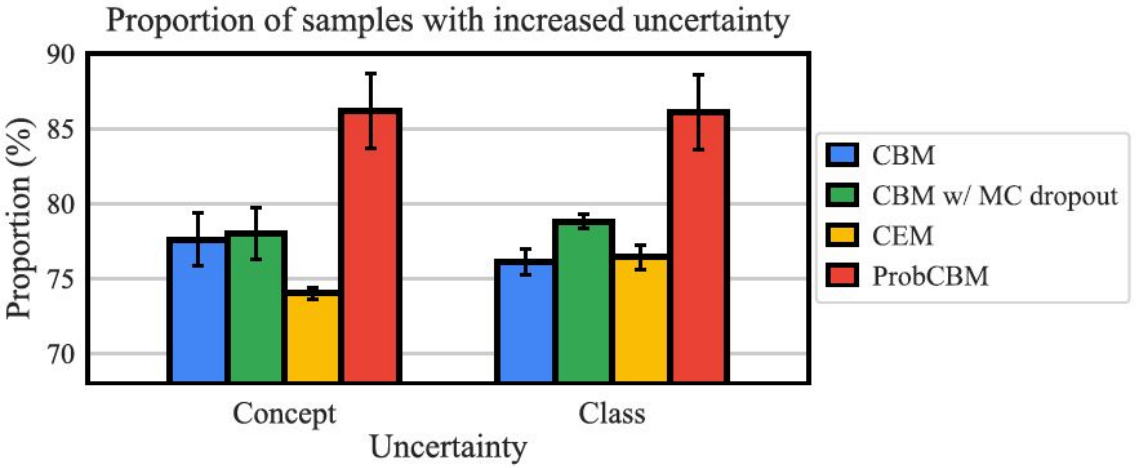


Figure 6. Proportion (%) of samples whose uncertainty increases after occlusion for the CUB dataset. The experiments are conducted with an image size of 299 × 299. Results include mean values with standard deviation from three experiment repetitions.

2.4 Results

(a) CUB

Target class: American Goldfinch

Concept Prediction	P	U	P	U	P	U
wing color: white	1.000	0.178	0.988	0.258	0.967	**0.301**
upper tail color: black	1.000	0.119	0.986	**0.311**	0.995	0.257
breast color: yellow	0.999	0.164	0.998	0.170	0.306	**0.264**
belly color: yellow	1.000	0.093	1.000	0.145	0.016	**0.303**
Class Prediction	American Goldfinch P: 0.972 U: 2.436		American Goldfinch P: 0.954 U: 3.019		American Goldfinch P: 0.791 U: 3.663	

(b) AwA2

Target class: Deer

Concept Prediction	P	U	P	U
furry	0.999	**0.517**	0.999	0.465
spots	1.000	0.033	0.996	**0.317**
horns	1.000	0.129	0.999	**0.437**
forest	1.000	0.059	0.998	**0.342**
Class Prediction	Deer P: 1.000 U: 0.188		Deer P: 0.999 U: 0.233	

Figure 7. Examples of ProbCBM's prediction in diverse visual contexts for the (a) CUB and (b) AwA2 datasets. P and U represent the probability and uncertainty, respectively. The highest value of uncertainty in each concept is bolded.

2.4 Results

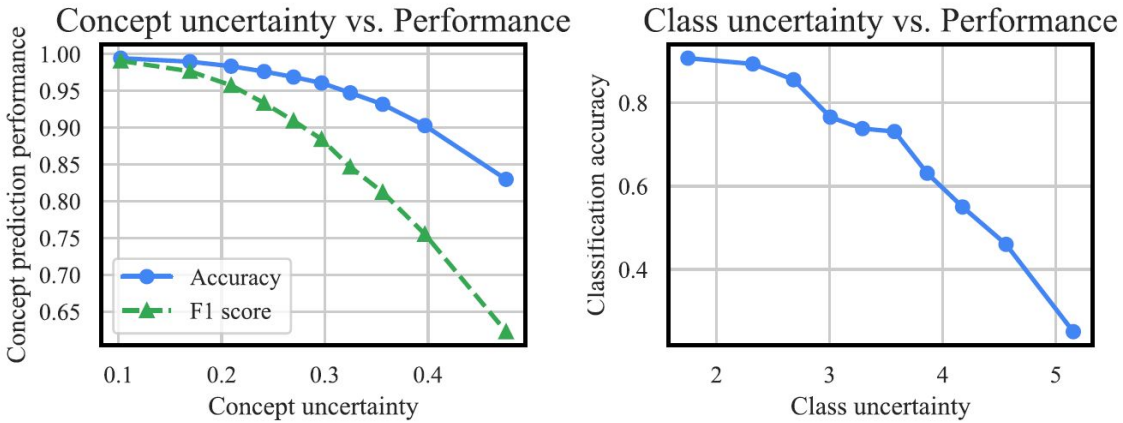


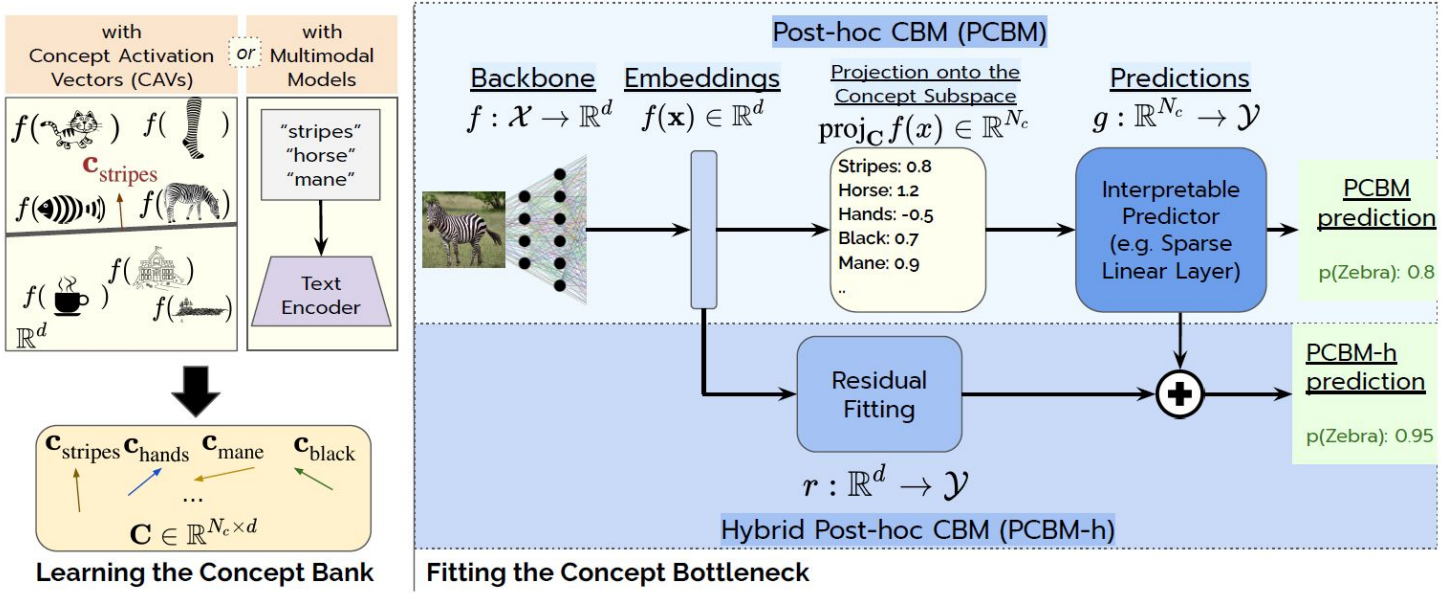
Figure 10. Correlation between performance and uncertainty for the CUB dataset. A point represents one group of samples with similar uncertainties and the x-axis indicates the median uncertainty for each group.

03

ICLR 2023

Post-Hoc Concept Bottleneck Models

3.1 Overview


with Concept Activation Vectors (CAVs)
or
with Multimodal Models
$\mathbf{c}_{\text{stripes}}$
"stripes"
"horse"
"mane"
Text Encoder
$\mathbb{R}^d$
$\mathbf{c}_{\text{stripes}}$ $\mathbf{c}_{\text{hands}}$ $\mathbf{c}_{\text{mane}}$ $\mathbf{c}_{\text{black}}$
$\mathbf{C} \in \mathbb{R}^{N_c \times d}$
Learning the Concept Bank
Post-hoc CBM (PCBM)
Backbone
$f : \mathcal{X} \to \mathbb{R}^d$
Embeddings
$f(\mathbf{x}) \in \mathbb{R}^d$
Projection onto the Concept Subspace
$\text{proj}_{\mathbf{C}} f(x) \in \mathbb{R}^{N_c}$
Predictions
$g : \mathbb{R}^{N_c} \to \mathcal{Y}$
Stripes: 0.8
Horse: 1.2
Hands: -0.5
Black: 0.7
Mane: 0.9
..
Interpretable Predictor (e.g. Sparse Linear Layer)
PCBM prediction
p(Zebra): 0.8
Residual Fitting
PCBM-h prediction
p(Zebra): 0.95
$r : \mathbb{R}^d \to \mathcal{Y}$
Hybrid Post-hoc CBM (PCBM-h)
Fitting the Concept Bottleneck

3.2 Preliminary

Model Architecture

- Any pretrained backbone model defined as $f : \mathcal{X} \to \mathbb{R}^d$
  d is the size of embedding space and *X* is the input space

- **Examples?** *f* can be:
  - the image encoder of CLIP
  - the model layers up to the penultimate layer of a ResNet

3.3 Method

Learning the Concept Subspace (C)

- First define a concept library $I = \{i_1, i_2, ..., i_{Nc}\}$

  Concepts can be selected by a domain expert or learned automatically from the data (Ghorbani et al., 2019; Yeh et al., 2020)

- For each concept *i*, collect embeddings for:
  - positive examples $P_i = \{f(x_{p1}), ..., f(x_{pNp})\}$
  - negative examples $N_i = \{f(x_{n1}), ..., f(x_{pNn})\}$

  Unlike normal CBMc, these samples can be different from the data used to train the backbone model

3.3 Method

Learning the Concept Subspace (C) ***(cont.)***

- Train a linear SVM using P_i and N_i to learn the corresponding CAV (the vector normal to the linear classification boundary)
- Denote the CAV for concept *i* as c_i
- Denote $C \in R^{Nc \times d}$ as matrix of CAVs (rows are c_i)
- Given an input, project the embedding of the input onto the subspace spanned by CAVs (the concept subspace):

$$f_{\boldsymbol{C}}(x) = \text{proj}_{\boldsymbol{C}} f(x) \in \mathbb{R}^{N_c} \qquad f_{\boldsymbol{C}}^{(i)}(x) = \frac{\langle f(x), \boldsymbol{c}_i \rangle}{||\boldsymbol{c}_i||_2^2} \in \mathbb{R}$$

3.3 Method

Leveraging Multimodal Models to Learn Concepts

- No need for annotation and labeled data!
- Multimodal models such as CLIP have a text encoder f_{text} along with an image encoder, which maps a description to the shared embedding space.
- Therefore use:

$$\boldsymbol{c}_{\text{stripes}}^{\text{text}} = f_{\text{text}}(\text{“stripes”})$$

- Collect the text embeddings and construct our multimodal concept bank C^{text} as the concept subspace.

  For a given classification task, we can use ConceptNet to obtain concepts relevant to these classes. ConceptNet is an open knowledge graph, where we can find concepts that have particular relations to a query concept.

3.3 Method

Learning the Interpretable Predictor

- An interpretable sparse predictor that maps the concept subspace to the model prediction: $g : \mathbb{R}^{N_c} \rightarrow \mathcal{Y}$
- To learn the PCBM, we solve:

$$\min_{g} \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}} [\mathcal{L}(g(f_{\boldsymbol{C}}(x)), y)] + \frac{\lambda}{N_c K}\Omega(g)$$

$$g(f_{\boldsymbol{C}}(x)) = \boldsymbol{w}^T f_{\boldsymbol{C}}(x) + b \qquad \Omega(g) = \alpha||\boldsymbol{w}||_1 + (1-\alpha)||\boldsymbol{w}||_2^2$$

However, even when we have a rich concept subspace, concepts may not be enough to solve a task of interest.

3.3 Method

Recovering Original Model Performance with Residual Modeling

- We introduce Hybrid Post-Hoc CBMs (PCBM-h).
- After fixing the concept bottleneck and the interpretable predictor, re-introduce the embeddings to 'fit the residuals':

$$\min_r \mathbb{E}_{(x,y)\sim\mathcal{D}}[\mathcal{L}(g(f_{\boldsymbol{C}}(x)) + r(f(x)), y)]$$

$$r : \mathbb{R}^d \to \mathcal{Y} \text{ is the residual predictor } \quad r(f(x)) = w_r^T f(x) + b_r$$

- *r* can be discarded to observe normal performance.

3.4 Results

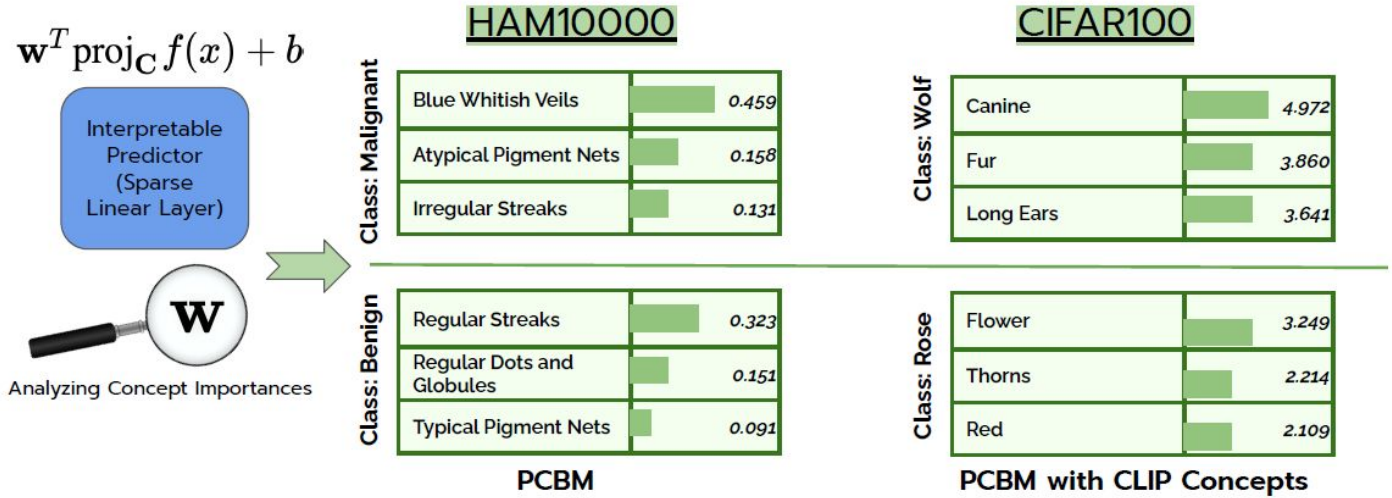


Figure 2: **Explaining Post-hoc CBMs.** We report the top 3 largest weights in the linear layer for the shown classes. For instance, *Blue Whitish Veils*, *Atypical Pigment Networks*, and *Irregular Streaks* have large weights for classifying whether a skin lesion is malignant. These are consistent with dermatologists' findings (Menzies et al., 1996).

3.4 Results

Table 1: **PCBMs achieve comparable performance to the original model**. We report performance over different scenarios for the original model and PCBMs with concept datasets. In CIFAR100, PCBM performs poorly since the concept bank is not expressive enough to solve a finer-grained task; however, PCBM-h recovers the original model's accuracy. Strikingly, PCBMs match the performance of the original model in HAM10000 and ISIC, *with as few as 8 human-interpretable concepts*. Original CBMs cannot be trained on CIFAR/HAM10000/ISIC/COCO-Stuff, as they do not have concept labels in the training dataset. The mean and standard errors are reported over 10 random seeds. We report AUROC for HAM10000 and ISIC, mAP for COCO-Stuff, and accuracy for CIFAR and CUB.

	CIFAR10	CIFAR100	COCO-Stuff	CUB	HAM10000	ISIC
Original Model	0.888	0.701	0.770	0.612	0.963	0.821
PCBM	0.777 ± 0.003	0.520 ± 0.005	0.741 ± 0.002	0.588 ± 0.008	0.947 ± 0.001	0.736 ± 0.012
PCBM-h	0.871 ± 0.001	0.680 ± 0.001	0.768 ± 0.01	0.610 ± 0.010	0.962 ± 0.002	0.801 ± 0.056

3.4 Results

Table 2: **Concept Bottlenecks with CLIP concepts**. When a concept bank is not available or is insufficient, we can use natural language descriptions of concepts with CLIP to implement CBMs.

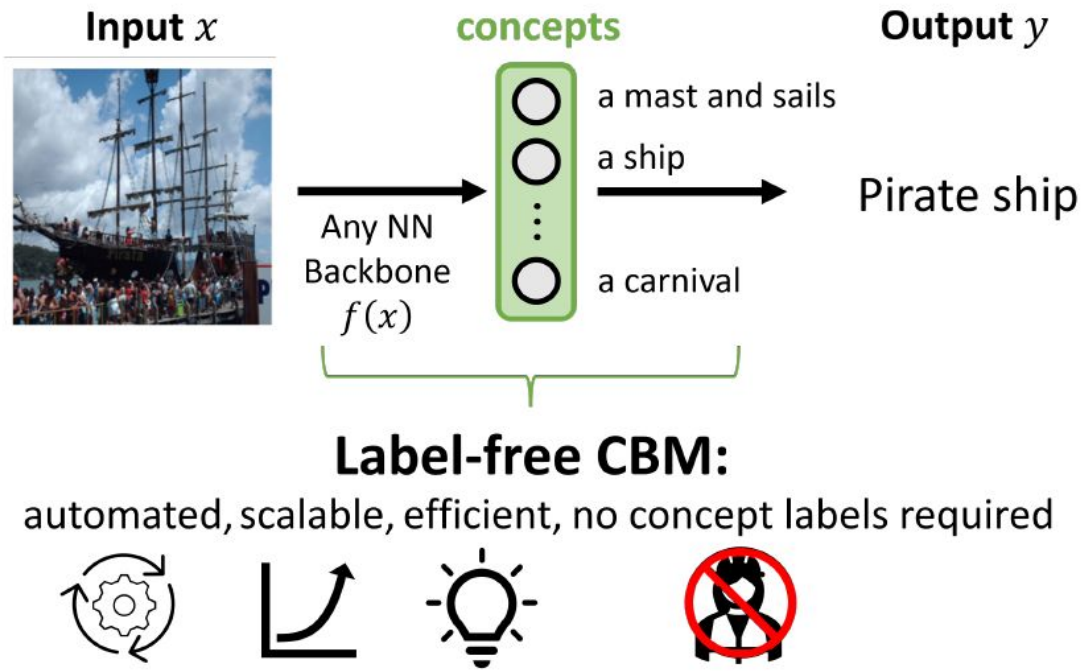
	CIFAR10	**CIFAR100**	**COCO-Stuff**
Original Model (CLIP)	0.888	0.701	0.770
PCBM & labeled concepts	0.777 ± 0.003	0.520 ± 0.005	0.741 ± 0.002
PCBM & CLIP concepts	0.833 ± 0.003	0.600 ± 0.003	0.755 ± 0.001
PCBM-h & CLIP concepts	0.874 ± 0.001	0.691 ± 0.006	0.769 ± 0.001

04

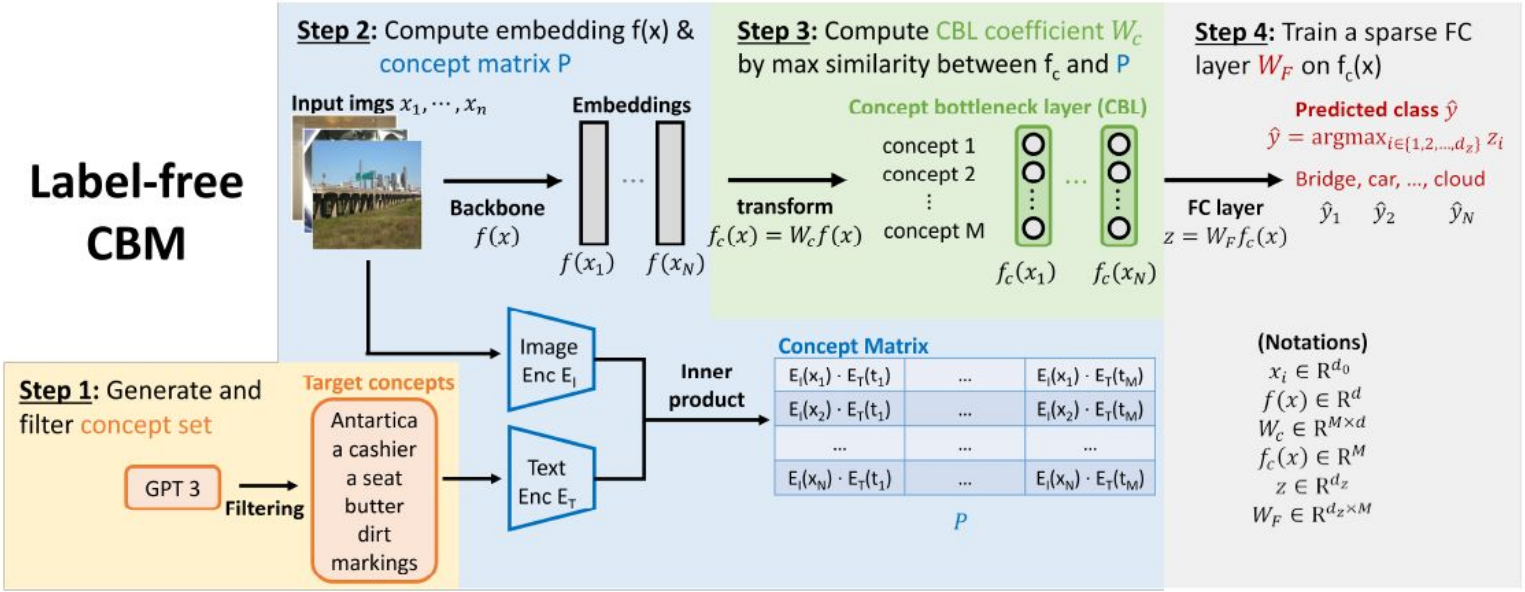
ICLR 2023

Label-Free Concept Bottleneck Models

4.1 Overview


Input x
concepts
Output y
a mast and sails
a ship
a carnival
Any NN
Backbone
$f(x)$
Pirate ship
Label-free CBM:
automated, scalable, efficient, no concept labels required

4.1 Overview


Label-free CBM
Step 1: Generate and filter concept set
GPT 3
Filtering
Target concepts
Antartica
a cashier
a seat
butter
dirt
markings
Step 2: Compute embedding f(x) & concept matrix P
Input imgs $x_1, \cdots, x_n$
Backbone $f(x)$
Embeddings
$f(x_1)$ $f(x_N)$
Image Enc E_I
Text Enc E_T
Inner product
Concept Matrix
$E_I(x_1) \cdot E_T(t_1)$ ... $E_I(x_1) \cdot E_T(t_M)$
$E_I(x_2) \cdot E_T(t_1)$ ... $E_I(x_2) \cdot E_T(t_M)$
... ...
$E_I(x_N) \cdot E_T(t_1)$ ... $E_I(x_N) \cdot E_T(t_M)$
P
Step 3: Compute CBL coefficient W_c by max similarity between f_c and P
transform $f_c(x) = W_c f(x)$
Concept bottleneck layer (CBL)
concept 1
concept 2
concept M
$f_c(x_1)$ $f_c(x_N)$
Step 4: Train a sparse FC layer W_F on $f_c(x)$
FC layer $z = W_F f_c(x)$
Predicted class $\hat{y}$
$\hat{y} = \text{argmax}_{i \in \{1,2,\ldots,d_z\}} z_i$
Bridge, car, ..., cloud
$\hat{y}_1$ $\hat{y}_2$ $\hat{y}_N$
(Notations)
$x_i \in \mathrm{R}^{d_0}$
$f(x) \in \mathrm{R}^{d}$
$W_c \in \mathrm{R}^{M \times d}$
$f_c(x) \in \mathrm{R}^{M}$
$z \in \mathrm{R}^{d_z}$
$W_F \in \mathrm{R}^{d_z \times M}$

4.2 Preliminary

Background

- CBMs create a layer before the last fully connected layer where each neuron corresponds to a human interpretable concept.
- Post-Hoc CBM only retrains the last FC layer to solve this.
- The problems of requiring annotated concept data for TCAV remains and their use of CLIP model can only be applied to if the NN backbone is the CLIP image encoder.
- The performance of PCBMs is uninterpretable due to res con.

4.2 Preliminary

	(I) Flexibility		(II) Interpretability		(III) Performance	
Method:	Without labeled concept data	Any network architecture	Sparse final layer	All features interpretable	Preserves accuracy	Extends to ImageNet scale
CBM	No	**Yes**	No	**Yes**	No	No
IBD	No	**Yes**	No	No	**Yes**	No
P-CBM	No	**Yes**	Yes	**Yes**	No	No
P-CBM (CLIP)	**Yes**	No	**Yes**	**Yes**	No	Maybe
P-CBM-h	No	**Yes**	**Yes**	No	**Yes**	No
P-CBM-h (CLIP)	**Yes**	No	**Yes**	No	**Yes**	Maybe
Label-free CBM **(This work)**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**	**Yes**

Table 1: Comparison of our method against existing methods for creating Concept Bottleneck models, CBM (Koh et al., 2020), IBD (Zhou et al., 2018) and 4 versions of P-CBM (Yuksekgonul et al., 2022), where '-h' indicates the hybrid model that uses uninterpretable residual term, '(CLIP)' means models using CLIP concepts. We used maybe to indicate models that could in theory extend to ImageNet but have not been tested.

4.3 Method

Step 1.A: Initial Concept Set Creation

- Ask GPT-3 to:
  - List the most important features for recognizing something as a {class}:
  - List the things most commonly seen around a {class}:
  - Give superclasses for the word {class}:

We found using GPT-3 to generate initial concepts to perform better than using the knowledge-graph ConceptNet which was used in PCBM. Reasons for this and a comparison between the two are presented in Appendix A.6.

4.3 Method

Step 1.B: Concept Set Filtering

- Filters to improve quality and reduce the size of concept set:
  - Concept length
  - Remove concepts too similar to classes
  - Remove concepts too similar to each other
  - Remove concepts not present in training data
  - Remove concepts we can't project accurately

    (last item is performed after step 3)

4.3 Method

Step 2: Compute Embeddings

- Given concepts $C = \{t_1, ..., t_M\}$, training dataset $D = \{x_1, ..., x_N\}$, calculate and save the CLIP concept activation matrix P using the CLIP image and text encoders:

$$P_{i,j} = E_I(x_i) \cdot E_T(t_j)$$

- *This is independent of the backbone model, thus we need to learn a projection from the backbone model's feature space into a space where axis directions correspond to interpretable concepts.*

4.3 Method

Step 3: Learn Projection to Create a Concept Bottleneck Layer

- Initialize projection weights W_c as a random $M \times d_0$ matrix where d_0 is the dimensionality of backbone features f(x).
- Define $f_c(x) = W_c f(x)$, where $f_c(x_i) \in R^M$.
- Use k to denote a neuron of interest in the projection layer, and its activation pattern is q_k where $q_k = [f_{c,k}(x1), \ldots, f_{c,k}(x_N)]^\top$, with $q_k \in R^N$ and $f_{c,k}(x) = [f_c(x)]_k$.
- Now, how do we train each neuron? Use CLIP-Dissect.

4.3 Method

Step 3: Learn Projection to Create a CBL ***(cont.)***

- To make the neurons in the CBL interpretable, we need to enforce the projected neurons to activate in correlation with the target concept, which we do by optimizing W_c to maximize the CLIP-Dissect similarity between the neuron's activation pattern and the target concept.
- To improve this similarity, optimize:

$$L(W_c) = \sum_{i=1}^{M} -\text{sim}(t_i, q_i) := \sum_{i=1}^{M} -\frac{\bar{q_i}^3 \cdot \bar{P_{:,i}}^3}{||\bar{q_i}^3||_2 ||\bar{P_{:,i}}^3||_2}$$

4.3 Method

Step 4: Learning the Sparse Final Layer

- Learn the predictor with the fully connected layer $W_F \in R^{dz \times M}$ where *dz* is the number of output classes.
- Keep W_F sparse, since sparse layers have been demonstrated to be more interpretable.
- Solve elastic net objective:

$$\min_{W_F, b_F} \sum_{i=1}^{N} L_{ce}(W_F f_c(x_i) + b_F, y_i) + \lambda R_\alpha(W_F)$$

$$R_\alpha(W_F) = (1-\alpha)\tfrac{1}{2}||W_F||_F + \alpha||W_F||_{1,1}$$

4.4 Results

Model	Sparse final layer	Dataset				
		CIFAR10	CIFAR100	CUB200	Places365	ImageNet
Standard	No	88.80%	70.10%	76.70%	48.56%	76.13%
Standard (sparse)	Yes	82.96%	58.34%	**75.96%**	38.46%	**74.35%**
P-CBM*	Yes	70.50%	43.20%	59.60%	N/A	N/A
P-CBM (CLIP)*	Yes	84.50%	56.00%	N/A	N/A	N/A
Label-free CBM **(Ours)**	Yes	**86.40%** ± 0.06%	**65.13%** ± 0.12%	74.31% ± 0.29%	**43.68%** ± 0.10%	71.95% ± 0.05%

Table 2: Accuracy comparison, best performing sparse model bolded. We can see our method outperforms Posthoc-CBM and performs similarly to a sparse standard model. The results for our method are mean and standard deviation over three training runs. *Indicates reported accuracy.

4.4 Results

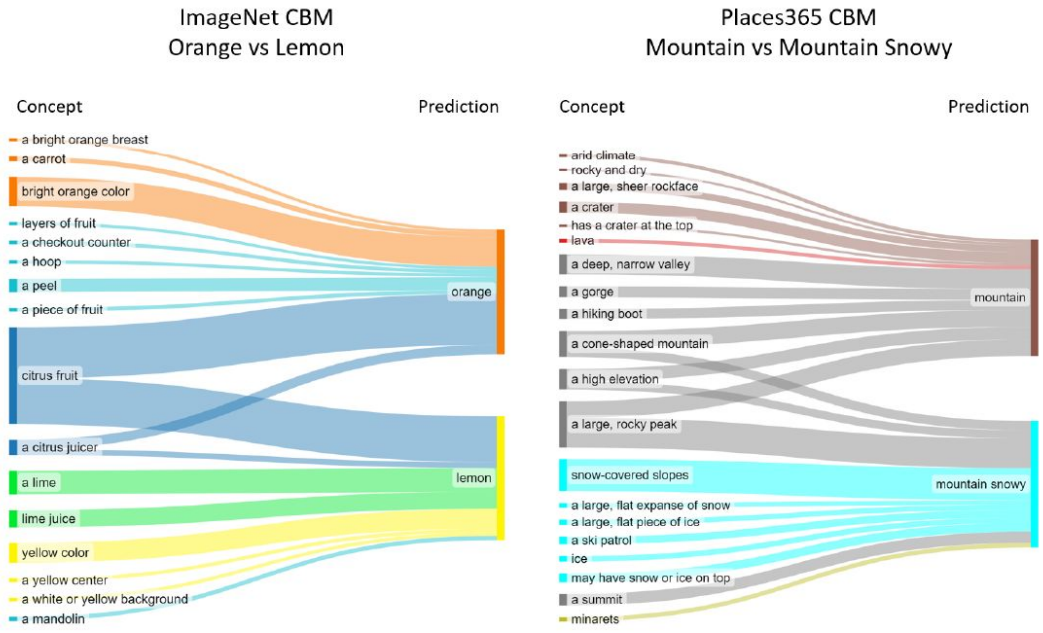


Figure 3: Visualization of the final layer weights of our Label-free CBM. Showcasing how our models differentiate between two similar classes.

05

ACL 2024

Explaining Language Model Predictions with High-Impact Concepts

5.1 Overview

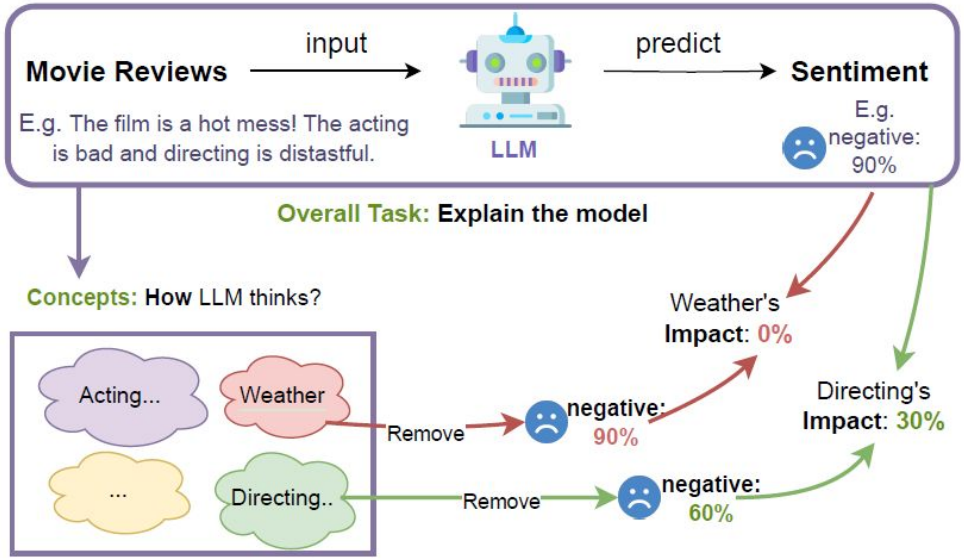


Figure 1: Illustration of concept-based explanations that result in high impact (green line) or not (red line) when explaining the LLMs in a sentiment classification task.

5.1 Overview

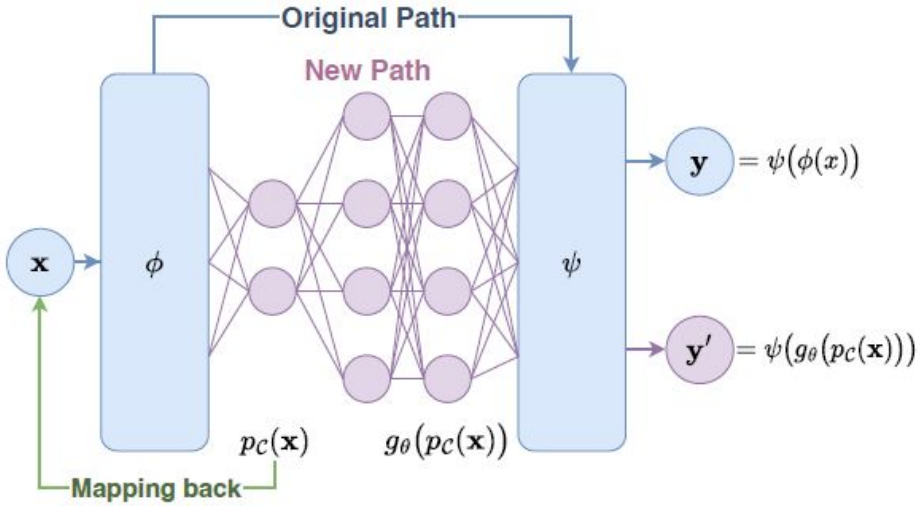


Figure 2: The overall concept generation process of a concept bottleneck model.

5.1 Overview

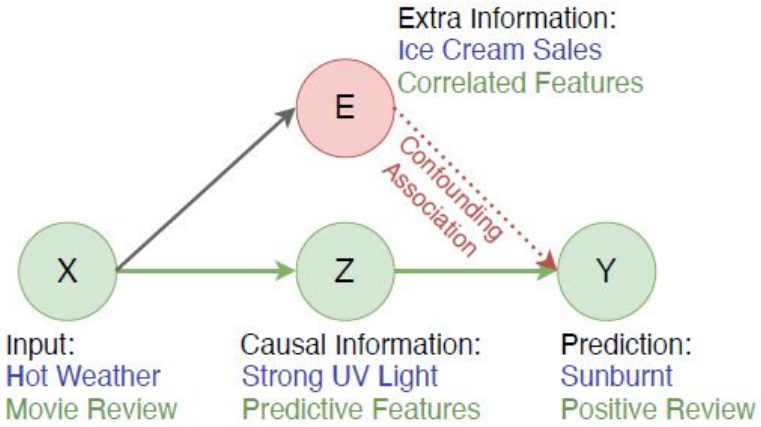


Figure 3: Illustration of the causal graph indicating the confounding association in explanation models. Blue is a real-life example. Green is the correspondence in a movie review classification task.

5.2 Preliminary

Background

- Traditional CBMs that do not consider impact may falsely use the correlational feature to explain causation.
- Previous causality evaluations focused on assessing the causal effect via local (i.e., instance-level) change and removal intervention, leading to potential bias.

- **Alternative?** High-Impact Concepts (HI-concept)
  - Concept explanation framework with causal impact optimization.
  - Can be used on a pretrained whitebox LLM.
  - Employs a global (i.e., model-level) accuracy change metric and insertion operation to diagnose the causality measurement.

5.2 Preliminary

CBM Losses

- **Reconstruction loss:** To recover the original model's predictions after inserting a step in the middle of two sections of model (hidden layers and final prediction).

$$\begin{aligned}\mathcal{L}_{\text{rec}}(\theta, \mathcal{C}) &= \text{CE}\Big(\psi\big(\phi(\mathbf{x})\big), \psi\big(g_\theta(p_{\mathcal{C}}(\mathbf{x}))\big)\Big) \\ &= -\sum_{b \in \mathcal{B}} \psi\big(\phi(\mathbf{x})\big)_b \log\left(\psi(g_\theta(p_{\mathcal{C}}(\mathbf{x})))_b\right)\end{aligned}$$

- **Regularization loss:** To make each concept vector correspond to actual examples and concepts to be distinct from eachother.

$$\begin{aligned}\mathcal{L}_{\text{reg}}(\mathcal{C}) = &-\lambda_1 \frac{\sum_{i=1}^{n} \sum_{\mathbf{x}_t \in \mathcal{T}_{\mathbf{c}_i}} \mathbf{c}_i^\top \phi(\mathbf{x}_t)}{nN} \\ &+ \lambda_2 \frac{\sum_{i_1 \neq i_2} \mathbf{c}_{i_1}^\top \mathbf{c}_{i_2}}{n(n-1)}.\end{aligned}$$

5.2 Preliminary

Causal Analysis

- To define impact in our method, we utilize two important definitions in causal analysis: Individual Treatment Effect (ITE) and Average Treatment Effect (ATE), which measure the effect of interventions in randomized experiments.

$$\begin{aligned}\mathrm{ITE}(x) :=\ & \mathbb{E}[y|\mathbf{X} = x, \mathrm{do}(T = 1)] \\ & - \mathbb{E}[y|\mathbf{X} = x, \mathrm{do}(T = 0)]; \\ \mathrm{ATE} :=\ & \mathbb{E}[\mathrm{ITE}(x)].\end{aligned}$$

5.3 Method

Defining Impact

- To separate correlation from causation, enforce explanations to be predictive by considering their "impact".
- A concept c_i is a cause if its removal changes the outcome, and corresponds to a direction in latent space. As f is fixed, its prediction process is deterministic and reproducible, allowing us to conduct experiments.
- Thus, we define the do-operation of ITE as concept removal and define impact of the concept as:

$$\mathrm{I}(\mathbf{c}_i, \mathbf{x}) = \mathbb{E}[y|\mathbf{X} = \mathbf{x}, \mathbf{c}_i = \mathbf{0}] - \mathbb{E}[y|\mathbf{X} = \mathbf{x}, \mathbf{c}_i = \mathbf{c}_i]$$

5.3 Method

Optimizing for Impact

- To incorporate consideration for impact into the concept discovery process, we introduce two new losses to the original framework.
- **Auto-encoding loss:** Reconstructs the hidden representations.

$$\mathcal{L}_{\text{enc}}(\theta, \mathcal{C}) = \text{MSE}\Big(\phi(\mathbf{x}), g_\theta(p_\mathcal{C}(\mathbf{x}))\Big) = \frac{1}{d}||\phi(\mathbf{x}) - g_\theta(p_\mathcal{C}(\mathbf{x})||_2^2.$$

- **Causality loss:** Disentangles concept directions that have a greater impact by optimizing the following loss:

$$\mathcal{L}_{\text{cau}}(\theta, \mathcal{C}) = -\sum_{\mathbf{c}_i \in \mathcal{S}} \sum_{\mathbf{x}_j \in \mathcal{D}} \Big| \psi\Big(g_\theta(p_\mathcal{C}(\mathbf{x}_j)|\mathbf{c}_i = \mathbf{0})\Big) - \psi\Big(g_\theta(p_\mathcal{C}(\mathbf{x}_j)|\mathbf{c}_i = \mathbf{c}_i)\Big)\Big| \approx -|I_{\text{avg}}(\mathcal{C})|.$$

5.3 Method

Final Method Steps

- Extract concepts from the model's hidden representations.
  - Run the dataset to collect activations of a layer from all samples.
- Discover human-understandable concepts.
  - Assume there is a lower-dimensional concept space $p_C(x)$ that can approximate the model's hidden space $\phi(x)$.
  - Initialize a random set of concept vectors C in the hidden space.
  - Optimize c_i using the losses:

$$\mathcal{L}(\theta, \mathcal{C}) = \mathcal{L}_{\text{rec}}(\theta, \mathcal{C}) + \mathcal{L}_{\text{reg}}(\mathcal{C}) + \lambda_e \mathcal{L}_{\text{enc}}(\theta, \mathcal{C}) + \lambda_c \mathcal{L}_{\text{cau}}(\theta, \mathcal{C})$$

- Apply concepts to explain the model.
  - Project activations of a sample *x* onto concept space using $C^T\phi(x)$.
  - This $P_c(x)$ is prob of concept importance, use to highlight words.

5.3 Method

Evaluating Impact of Concepts

- Recovering Accuracy Change (ΔAcc) is introduced to measure whether the model's ability to faithfully recover predictions after concept omission is disrupted:

$$\Delta\mathrm{Acc}(\mathcal{C}) = \frac{1}{|\mathcal{C}|}\sum_{\mathbf{c}_i \in \mathcal{C}} |\mathrm{Acc}(\mathcal{C}) - \mathrm{Acc}(\mathcal{C} \setminus \{\mathbf{c}_i\})|$$

- We follow previous work to use Causal Concept Effect (CACE) to evaluate the causal effect of the set of concepts C:

$$\mathrm{CACE}(\mathbf{c}_i) := \sum_{\mathbf{x}_j \in \mathcal{D}_{\text{test}}} |\psi\big(g_\theta(p_{\mathcal{C}}(\mathbf{x}_j))\big) - \psi\big(g_\theta(p_{\mathcal{C}\setminus\{i\}}(\mathbf{x}_j))\big)|;$$

$$\mathrm{CACE}(\mathcal{C}) = \frac{1}{|\mathcal{C}|}\sum_{\mathbf{c}_i \in \mathcal{C}} \mathrm{CACE}(\mathbf{c}_i)$$

5.4 Results

p_{cor}	Cls.Acc	Method	Acc	CACE	ΔAcc
0.50	95.4%	ConceptSHAP	97.6%	0.070	6.1%
		HI-concept	**98.4%**	**0.102**	**9.4%** (+3.3%)
0.65	99.0%	ConceptSHAP	**99.7%**	0.038	3.5%
		HI-concept	99.3%	**0.084**	**6.8%** (+3.4%))
0.75	96.1%	ConceptSHAP	98.3%	0.069	6.0%
		HI-concept	**98.9%**	**0.123**	**12.2%** (+6.2%)

Table 1: Faithfulness (Acc) and Causality (CACE, ΔAcc) evaluation on the toy dataset. Cls.Acc denotes the original classification model's accuracy.

5.4 Results

Dataset	Model	Method	Faithfulness					Causality	
			Acc	Precision	Recall	F1	Completeness	CACE	ΔAcc
IMDB	Transformer	β-TCVAE (Chen et al., 2018)	43.53%	50.23	50.03	33.08	27.36	0.037	1.50%
		K-means (Likas et al., 2003)	83.64%	84.74	85.05	83.63	61.87	0.047	2.59%
		PCA (F.R.S., 1901)	85.18%	85.56	86.20	85.15	**62.36**	0.001	0.01%
		ConceptSHAP (Yeh et al., 2020)	84.36%	85.04	85.56	84.34	62.05	0.031	1.30%
		HI-concept	**88.78%**	**90.07**	**87.50**	**88.24**	58.10	**0.150**	**11.06%**
	BERT	β-TCVAE (Chen et al., 2018)	93.86%	94.31	93.43	93.68	10.71	0.057	4.05%
		K-means (Likas et al., 2003)	**98.69%**	96.16	96.23	96.19	15.69	0.037	0.97%
		PCA (F.R.S., 1901)	96.68%	**96.65**	**96.68**	**96.67**	15.33	0.002	0.02%
		ConceptSHAP (Yeh et al., 2020)	95.84%	95.78	95.96	95.83	17.16	0.050	0.06%
		HI-concept	92.97%	93.25	93.34	92.97	**21.04**	**0.099**	**8.99%**
	Llama	β-TCVAE (Chen et al., 2018)	20.56%	33.41	33.36	13.30	-14.29	0.001	0.15%
		K-means (Likas et al., 2003)	15.31%	5.10	33.33	8.85	-21.82	0.019	0.00%
		PCA (F.R.S., 1901)	**95.15%**	**67.97**	**77.66**	**69.80**	**64.19**	0.001	0.03%
		ConceptSHAP (Yeh et al., 2020)	18.83%	42.83	34.95	14.88	-1.78	0.005	1.60%
		HI-concept	87.87%	53.27	68.60	55.29	59.83	**0.042**	**28.69%**
AG-News	Transformer	β-TCVAE (Chen et al., 2018)	98.91%	98.94	98.94	98.93	**66.73**	0.049	6.62%
		K-means (Likas et al., 2003)	98.16%	98.32	98.11	98.18	65.99	0.044	0.07%
		PCA (F.R.S., 1901)	**99.99%**	**99.99**	**99.99**	**99.99**	66.66	0.000	0.03%
		ConceptSHAP (Yeh et al., 2020)	73.01%	59.36	74.34	64.88	47.07	0.000	0.00%
		HI-concept	99.50%	99.50	99.51	99.50	66.70	**0.046**	**7.12%**
	BERT	β-TCVAE (Chen et al., 2018)	92.30%	94.93	91.89	92.91	57.25	0.044	5.32%
		K-means (Likas et al., 2003)	86.83%	92.74	85.42	87.53	52.62	0.028	7.15%
		PCA (F.R.S., 1901)	99.79%	99.82	99.77	99.79	61.04	0.001	0.01%
		ConceptSHAP (Yeh et al., 2020)	93.46%	93.70	94.62	93.66	**62.69**	0.025	4.44%
		HI-concept	**99.90%**	**99.89**	**99.90**	**99.89**	61.12	**0.058**	**10.54%**
	Llama	β-TCVAE (Chen et al., 2018)	1.27%	0.25	20.00	0.50	-23.89	0.000	0.01%
		K-means (Likas et al., 2003)	37.00%	7.40	20.00	10.80	1.09	0.007	0.02%
		PCA (F.R.S., 1901)	**85.41%**	**65.78**	**67.98**	**66.73**	**51.46**	0.000	0.03%
		ConceptSHAP (Yeh et al., 2020)	17.01%	35.37	35.20	15.87	-7.73	0.002	2.83%
		HI-concept	81.52%	48.59	55.99	51.53	43.07	**0.039**	**27.79%**

Table 2: Faithfulness (Acc, Precision, Recall, F1, Completeness) and causality (CACE, ΔAcc) evaluation of different text classification methods. The best result is bolded, and the second-best result is underlined.

5.4 Results

Method	CACE	Keywords
CS	0.134	apple, NASA, Microsoft, new, sun, red, super, game
CS	0.000	one, two, gt, new, cl, lt, first, world, mo, last
HI-C	0.130	us, bush, u, eu, new, peoples, china, high, gt, world
HI-C	0.003	us, update, new, mo, two, first, knicks, last, one, hen

Table 3: Generated concepts with Average Impact (CACE) from AG-News dataset, BERT model. CS is ConceptSHAP, HI-C is *HI-concept*. Each line is one concept, represented by keywords, which are ordered by descending importance.

5.4 Results

Method	Visualization
ConceptSHAP	dream team leads spain 44 - 42 at halftime athens, greece - as expected, the u.s. men's basketball team had its hands full in a quarterfinal game against spain on thursday...
HI-concept	dream team leads spain 44 - 42 at halftime athens, greece - as expected, the u.s. men's basketball team had its hands full in a quarterfinal game against spain on thursday ...

Figure 4: Qualitative comparison from AG-News: "World" news misclassified as "Sports" by BERT.

5.4 Results

	Accuracy	Confidence	Time Spent
Plain	72.5%	3.2	10.7
ConceptSHAP	68.5%	2.7	10.6
Polyjuice	73.5%	2.6	7.6
HI-concept	**80.5%**	**3.5**	**9.3**

Table 4: Human study for explainability evaluation.

5.4 Results

Method	Acc	CACE	ΔAcc
Without Auto-Encoding Loss	93.46%	0.028	6.11%
Without Prediction Loss	68.00%	0.035	**17.41%**
Without Regularizer Loss	95.76%	0.041	6.23%
Without Causality Loss	**99.92%**	0.029	2.95%
HI-concept	99.90%	**0.058**	10.54%

Table 5: Ablation on BERT for IMDB with faithfulness (Acc) and impact (CACE, ΔAcc) evaluation.

THANKS!

Any questions?

Maryam Rezaee

TGML Lab | Sharif University of Technology

Under the supervision of

Dr. Fatemeh SeyyedSalehi