SELECTIVE CONCEPT BOTTLENECK MODELS WITHOUT PREDEFINED CONCEPTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Concept-based models like Concept Bottleneck Models (CBMs) have garnered significant interest for improving model interpretability by first predicting human-understandable concepts before mapping them to the output classes. Early approaches required costly concept annotations. To alleviate such, recent methods utilized large language models to automatically generate class-specific concept descriptions and learned mappings from a pretrained black-box model's raw features to these concepts using vision-language models. However, these approaches assume prior knowledge of which concepts the black-box model has learned. In this work, we discover the concepts encoded by the model through unsupervised concept discovery techniques instead. We further propose an input-dependent concept selection mechanism that dynamically retains a sparse set of relevant concepts for each input, enhancing both sparsity and interpretability. Our approach not only improves downstream performance but also needs significantly fewer concepts for accurate classification. Lastly, we show how large vision-language models can guide the editing of our models' weights to correct errors.

1 INTRODUCTION

Deep neural networks have achieved tremendous success in a variety of tasks on various input modalities. However, they are *black-box* models, making it difficult for humans to understand and comprehend their decisions. Thus, there has been considerable recent interest in developing *interpretable* models. One popular framework is Concept Bottleneck Models (CBMs) (Koh et al., 2020), i.e., models that first predict human-understandable concepts and then use these concepts to predict the classes (Lampert et al., 2009; Kumar et al., 2009). Initial CBMs are trained in an end-to-end fashion through supervision on *both* the concepts and classes. However, the need for human-annotated concepts during model training requires the time-consuming and expensive collection of such.

To address this limitation of initial CBMs, recent work (Yuksekgonul et al., 2023; Oikarinen et al., 2023; Menon & Vondrick, 2023; Marcinkevičs et al., 2024) has proposed converting pretrained black-box models into CBMs in a *post-hoc* fashion. To avoid the need for annotations, they leveraged large language models (e.g., GPT-3 (Brown et al., 2020)) to generate class-specific language descriptions and learned a mapping from the black-box model's uninterpretable features to these concepts using vision-language models (e.g., CLIP (Radford et al., 2021)). However, this raises a crucial question:

How can we know ***a priori*** *which concepts a pretrained black-box model has learned?*

Instead of defining the concepts in advance, we propose to discover concepts that accurately decompose the features learned by the black-box model (1st contribution). To do so, we draw from the rich literature on unsupervised concept discovery (Ghorbani et al., 2019; Zhang et al., 2021; Zou et al., 2023; Fel et al., 2023b; Vielhaben et al., 2023; Fel et al., 2023a; Huben et al., 2024; Stein et al., 2024). We chose CRAFT (Fel et al., 2023b) for our experiments because it has been shown to yield human-understandable concepts (Fel et al., 2023a), but other techniques are also possible. CRAFT employs non-negative matrix factorization (Lee & Seung, 1999) to decompose each feature activation into a sparse linear combination of concept vectors. The set of shared concept vectors forms a dictionary matrix. After learning this dictionary matrix, we compute the alignment between the raw bottleneck features and the concept vectors to measure a concept's presence or absence.

Subsequently, we train an interpretable linear classifier on the concepts' alignment scores, linking the alignment scores to the predictions. Previous work (Yuksekgonul et al., 2023; Oikarinen et al., 2023) has shown that a sparsity penalty on the linear classifier's weights ensures that each class relies on only a sparse set of concepts. However, they did not examine the per-sample number of concepts that affect the classification across *all* classes. That is, while individual classes rely on sparse sets of concepts, the overall model depends on substantially more. Empirically, we found that typically over 90% of the available concepts–up to ca. 4200 concepts (see Table 2)–affect the classification per input. As a result, it complicates the interpretation of the model's decision-making process.

To address these challenges, we propose an *input-dependent concept selection mechanism* that ensures that only a sparse set of concepts relevant for the classification of an individual input sample is dynamically retained (2nd contribution). We simply apply an activation function before the sparse linear classifier, and enforce sparsity on its output or directly use its sparsity parameter. In our experiments, TopK (Makhzani & Frey, 2014) performed best. This mechanism allows the concepts that are retained or removed (i.e., zeroed out) to vary between inputs, making it input-dependent. It also preserves the interpretability of CBMs, as the predictions remain linear w.r.t. the retained concepts.

In summary, our contributions are as follows:

- We propose a new type of CBM called Unsupervised Concept Bottleneck Models (UCBMs)[1]; see Figure 1 for an overview. UCBMs convert pretrained, black-box models into a CBM by *discovering and using the concepts that it has learned.*
- We propose a novel input-dependent concept selection mechanism that *dynamically retains a sparse set of concepts* relevant to classification. For example, as few as ca. 1.4% of the available concepts are used per input (Table 2).
- We show that UCBMs improve *performance* while having a substantially higher degree of *sparsity* compared to previous work (Tables 1 and 2).
- We show that UCBMs are *interpretable* qualitatively as well as through a user study (Section 3.2), and demonstrate that large-vision-language models can help us to *intervene* on UCBMs' weights to fix errors (Section 3.3).

2 Unsupervised Concept Bottleneck Models with input-dependent concept selection

In this section, we introduce Unsupervised Concept Bottleneck Models (UCBMs), a novel CBM that uses concepts that are automatically discovered and most accurately decompose features learned by a black-box model (Section 2.1), dynamically only retains the concepts most relevant to classification, and finally classifies the input with a sparse linear model (Section 2.2). Figure 1 provides an overview of UCBMs, and the above steps are described in detail below.

Notations. Let $f : \mathcal{X} \rightarrow \mathbb{R}^p$ be a pretrained, black-box model's feature extractor that maps from an input space $\mathcal{X} \subseteq \mathbb{R}^d$ to the bottleneck feature space of a size of p. Further, let $\mathbf{X} \in \mathbb{R}^{N \times d}$ be the input data matrix where the ith row is the input $\mathbf{x}_i \in \mathbf{X}$ and let $\mathbf{A} = f(\mathbf{X}) \in \mathbb{R}^{N \times p}$ be the bottleneck feature activations. Lastly, let $\mathcal{Y}$ denote the class label space.

2.1 Discovery of concepts learned by the black-box model

Previous post-hoc CBMs have either used human-annotated concepts (Yuksekgonul et al., 2023; Marcinkevičs et al., 2024) or aligned the black-box model's features with precomputed text features from vision-language models, using natural language descriptions, such as those generated by a large language model (Yuksekgonul et al., 2023; Oikarinen et al., 2023; Menon & Vondrick, 2023; Marcinkevičs et al., 2024). Importantly, both approaches rely on a *predefined set of concepts* –either through concept annotations or language descriptions thereof– implicitly assuming which concepts the black-box model has learned. However, these concepts are typically unknown in advance.

[1] Code is available at https://anonymous.4open.science/r/ucbm.

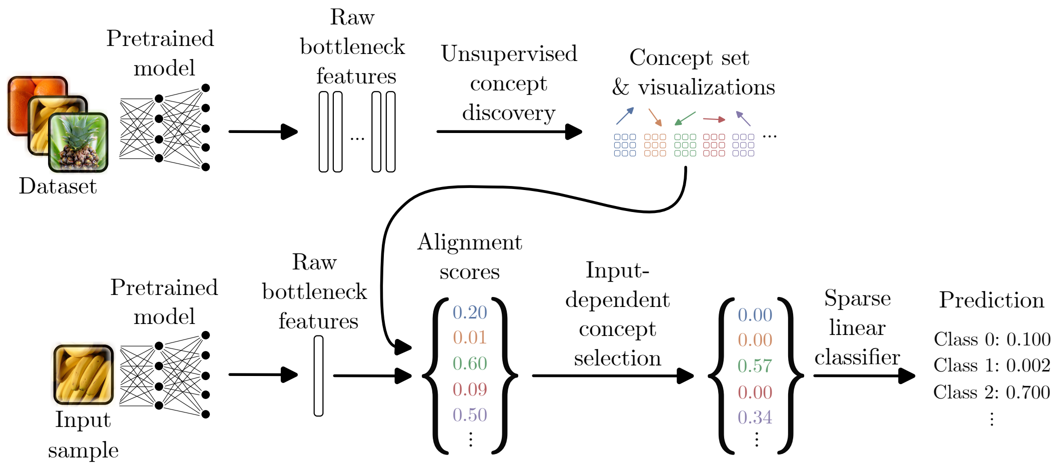


Figure 1: **Overview of Unsupervised Concept Bottleneck Models (UCBMs).** Top: We first extract concepts from raw bottleneck features of a pretrained black-box model using an unsupervised concept discovery method. Bottom: We compute the alignment between the bottleneck's features and previously discovered concepts (middle). Finally, we train an interpretable classifier consisting of an input-dependent concept selection mechanism and sparse linear classifier (middle to right).

Discovering the concepts that the black-box model has learned. To address this, we propose using unsupervised concept discovery techniques for UCBMs. These enable us to discover the concepts that the black-box model has learned, and do not require defining the concepts in advance.

Formally, the goal of unsupervised concept discovery is to extract a small set of interpretable concepts $\mathbf{c}$ that most faithfully reconstruct the feature activations $\mathbf{A}$. Assuming linearity of concepts, as per the superposition hypothesis (Kim et al., 2018; Elhage et al., 2022), unsupervised discovery methods can be understood as an instance of a dictionary learning problem (Dumitrescu & Irofti, 2018), as shown by Fel et al. (2023a):

$$(\mathbf{U}^*, \mathbf{C}^*) = \underset{\mathbf{U},\mathbf{C}}{\arg\min}\, ||\mathbf{A} - \mathbf{U}\mathbf{C}||_F^2 \qquad , \tag{1}$$

where $\mathbf{U} \in \mathbb{R}^{N \times |\mathbf{C}|}$ (sparse coefficient matrix) represents the activations $\mathbf{A} = f(\mathbf{X}) \in \mathbb{R}^{N \times p}$ w.r.t. a new basis spanned by the set of $|\mathbf{C}|$ concept activation vectors $\mathbf{C} \in \mathbb{R}^{|\mathbf{C}| \times p}$ (dictionary matrix), and $|| \cdot ||_F$ denotes the Frobenius norm. Intuitively, we learn a sparse linear decomposition of the feature activations for each input in Equation 1, where we weigh the shared concepts vectors by the input-specific sparse coefficients. Fel et al. (2023a) showed that previous methods, such as K-Means (Ghorbani et al., 2019), PCA (Zhang et al., 2021; Zou et al., 2023), non-negative matrix factorization (Lee & Seung, 1999; Zhang et al., 2021; Fel et al., 2023b), or sparse autoencoders (Makhzani & Frey, 2014; Huben et al., 2024), only differ in their constraints on $\mathbf{U}, \mathbf{C}$ in Equation 1.

In this work, we chose non-negative matrix factorization (i.e., CRAFT (Fel et al., 2023b)) for UCBMs, as it has been shown to discover human-understandable concepts (Fel et al., 2023a). However, we emphasize that UCBMs will benefit from future unsupervised concept discovery methods.

2.2 Learning the classifier with input-dependent concept selection

In the previous subsection, we discovered concept vectors $\mathbf{c}_j$ that most accurately decompose the uninterpretable features of a black-box model. Next, we compute the alignment scores between each concept vector and the model's features, denoted as $\text{sim}_{\mathbf{C}}(\mathbf{x}_i) \in [-1, 1]^{|\mathbf{C}|}$, where $\text{sim}_{\mathbf{C}}(\mathbf{x}_i)_j := \frac{\langle \mathbf{a}_i, \mathbf{c}_j \rangle}{||\mathbf{a}_i||_2 \cdot ||\mathbf{c}_j||_2}$ is the cosine similarity between the feature activations $\mathbf{a}_i = f(\mathbf{x}_i)$ of input $\mathbf{x}_i$ and concept $\mathbf{c}_j \in \mathbf{C}$. Then, we dynamically select the most relevant concepts and subsequently classify the input with a sparse linear model (Wong et al., 2021). Both are described in detail below.

Sparse linear classifier. Following Yuksekgonul et al. (2023); Oikarinen et al. (2023), we learn a sparse linear classifier by enforcing sparsity on its weight matrix (Wong et al., 2021):

$$\min_{\mathbf{W},\mathbf{b}} \sum_{i=1}^{N} \mathcal{L}(\mathbf{W}\text{sim}_{\mathbf{C}}(\mathbf{x}_i) + \mathbf{b}, y_i) + \lambda_w \underbrace{R_\alpha(\mathbf{W})}_{\mathcal{L}^{\mathbf{W}}_{\text{sparsity}}} \quad , \tag{2}$$

where $\mathbf{W} \in \mathbb{R}^{|\mathcal{Y}|\times|\mathbf{C}|}$ are the weights, $\mathbf{b} \in \mathbb{R}^{|\mathcal{Y}|}$ is the bias, $y_i \in \mathcal{Y}$ is the target class for input $\mathbf{x}_i$, $\mathcal{L}$ represents the task-specific loss function (cross-entropy loss throughout this work), λ_w controls the regularization strength on $\mathbf{W}$, and $R_\alpha(\mathbf{W}) := (1-\alpha)\frac{1}{2}||\mathbf{W}||_F + \alpha||\mathbf{W}||_{1,1}$ denotes the elastic net regularization (Zou & Hastie, 2005). Note that $\text{sim}_{\mathbf{C}}(\mathbf{x}_i)$ is normalized and frozen during optimization. Importantly, the sparsity aims to make the linear model's classifications sparse and Yuksekgonul et al. (2023) & Oikarinen et al. (2023) have shown that *an individual class* indeed relies on only a sparse set of concepts.

The main limitation with the approach discussed above is that it fails to produce *globally sparse* classifications. Specifically, most concepts contribute to the classification of any given input, meaning that even if a concept has no impact on one class (e.g., the predicted one), it may still influence others. We consider a concept to be actively contributing if it has a non-zero impact on the output (see Equation 7 for details). When we computed the number of such concepts in above approach, we found that nearly all of them affect the classifications (Table 2). This is because the cosine similarities between the black-box model's activations and concepts are generally non-zero.[2] This lack of (global) sparsity limits interpretability and makes it challenging to comprehend a prediction.

Input-dependent concept selection mechanism. To ensure that only few concepts affect classification per input without significant performance sacrifices, we propose a simple yet effective *input-dependent concept selection mechanism*. Specifically, we introduce a concept selector $\pi : \mathbb{R}^{|\mathbf{C}|} \to \mathbb{R}^{|\mathbf{C}|}$, which takes the alignment scores $\text{sim}_{\mathbf{C}}(\mathbf{x}_i)$ as input and outputs a sparse set of non-zero (i.e., active) concepts and zeroes out the others. We enforce sparsity through a penalty term on concept selector's output: $\mathcal{L}^{\pi}_{\text{sparsity}} = ||\pi(\cdot)||_0$. Intuitively, the sparsity penalty $\mathcal{L}^{\pi}_{\text{sparsity}}$ drives the concept selector π to only retain a sparse set of concepts which are important for classifying the input $\mathbf{x}_i$, as signaled by the task-specific loss $\mathcal{L}$.

We considered three candidates for the implementation of the input-dependent concept selection mechanism (please refer to Appendix B for further technical details):

- **ReLU:** We define the concept selector using the ReLU activation function as:

  $$\pi(\mathbf{x}_i) := \max(0, \text{sim}_{\mathbf{C}}(\mathbf{x}_i) - \mathbf{o}) \text{ with trainable offset parameter } \mathbf{o} \in \mathbb{R}_+^{|\mathbf{C}|} \quad . \tag{3}$$

  We apply elastic net regularization on the selector's output: $\mathcal{L}^{\pi}_{\text{sparsity}} = R_\alpha(\pi(\mathbf{x}_i))$.

- **JumpReLU:** We use JumpReLU activation function (Erichson et al., 2019) for concept selection with trainable offset parameter $\mathbf{o} \in \mathbb{R}_+^{|\mathbf{C}|}$ and the Heaviside step function H. We define the concept selector as:

  $$\pi(\mathbf{x}_i) := \text{sim}_{\mathbf{C}}(\mathbf{x}_i) \cdot H(\text{sim}_{\mathbf{C}}(\mathbf{x}_i) - \mathbf{o}) = \begin{cases} 0, & \text{sim}_{\mathbf{C}}(\mathbf{x}_i) \le \mathbf{o} \\ \text{sim}_{\mathbf{C}}(\mathbf{x}_i), & \text{sim}_{\mathbf{C}}(\mathbf{x}_i) > \mathbf{o} \end{cases} \quad . \tag{4}$$

  Following Rajamanoharan et al. (2024), we compute the gradients of the *expected* loss using straight-through-estimators (Bengio et al., 2013). We use the following sparsity penalty $\mathcal{L}^{\pi}_{\text{sparsity}} = \sum_j^{|\mathbf{C}|} H(\text{sim}_{\mathbf{C}}(\mathbf{x}_i)_j - \mathbf{o}_j)$. Note that $\mathcal{L}^{\pi}_{\text{sparsity}}$ directly optimizes L0.

- **TopK:** The TopK activation function (Makhzani & Frey, 2014) only keeps the $k \ll |\mathbf{C}|$ concepts with the largest alignment scores and zeroes out the remaining concepts:

  $$\pi(\mathbf{x}_i) := \text{TopK}(\text{sim}_{\mathbf{C}}(\mathbf{x}_i) - \mathbf{o}) \text{ with trainable offset parameter } \mathbf{o} \in \mathbb{R}_+^{|\mathbf{C}|} \quad . \tag{5}$$

  Note that the sparsity can be directly controlled by k and, thus, $\mathcal{L}^{\pi}_{\text{sparsity}} = 0$.

[2] While the classifier could technically "turn off" a concept $\mathbf{c}_j$ by setting its associated column vector to the null vector ($\mathbf{W}_{:,j} = 0$), this would effectively reduce the number of concepts and degrades performance, e.g., see Figure 3. Consequently, the sparse linear classifier is unlikely to learn many of such null vectors.

Final interpretable classifier. We obtain the final interpretable classifier by plugging Equation 3, 4, or 5 into Equation 2 together with the respective implementation of $\mathcal{L}^{\pi}_{\text{sparsity}}$:

$$\min_{\mathbf{W},\mathbf{b},\mathbf{o}} \sum_{i=1}^{N} \mathcal{L}(\mathbf{W}\pi(\mathbf{x}_i) + \mathbf{b}, y_i) + \lambda_w \mathcal{L}^{\mathbf{W}}_{\text{sparsity}} + \lambda_\pi \mathcal{L}^{\pi}_{\text{sparsity}} \quad , \tag{6}$$

where λ_π (or k for TopK) controls the regularization strength of $\mathcal{L}^{\pi}_{\text{sparsity}}$. Appendix B provides a detailed overview for all variants. It is important to note that the selection of concepts is learned in an unsupervised manner, and that the prediction remains linear w.r.t. the *active* concepts ($\pi(\mathbf{x}_i) \neq 0$).

Concept dropout. During initial experiments, we found that models became overly reliant on a single concept. To reduce this reliance, we added a dropout layer (Srivastava et al., 2014) after concept selection. As dropout is applied per concept, it encourages the model to spread its classification decisions across more concepts. Interestingly, we found that this could also improve performance.

3 EXPERIMENTS

We evaluated UCBM on diverse image classification tasks and compared it to relevant baselines. We show that UCBMs outperform prior work and narrow the gap to their black-box counterparts, while relying on substantially fewer concepts globally in their classification (Section 3.1). Then, we demonstrate the interpretability qualitatively as well as through a user study (Section 3.2). Lastly, we showcase how large vision-language models can be leveraged to improve our UCBMs by informing weight editing in order to fix model errors (Section 3.3).

Datasets & backbone black-box models. Following previous work, we evaluated UCBMs on ImageNet (Deng et al., 2009) with a pretrained ResNet-50 V2 (He et al., 2016), CUB (Wah et al., 2011) with ResNet-18 pretrained on CUB[3], and Places-365 (Zhou et al., 2017) with ResNet-18 pretrained on Places-365[4]. These datasets cover a diverse set of tasks from standard image classification (ImageNet), fine-grained classification (CUB), to scene recognition (Places-365).

Implementation details. We trained our UCBMs with Adam (Kingma, 2014) and cosine annealing learning rate scheduling (Loshchilov & Hutter, 2017) for 20 epochs. We used a learning rate of 0.001 on ImageNet and Places-365, and 0.01 on CUB; except for the JumpReLU for which we set it to 0.08 on CUB. We set $\alpha = 0.99$ for the elastic net regularization for all variants. We tuned the other hyperparameters (λ_π or k, λ_w, and dropout rate) to yield a good trade-off between performance, sparsity, and fair comparability. Refer to Appendix C for the hyperparameters and to Figure 4 and Appendices D and E for the effect of them.

Experimental setup. Since the number of concepts $|\mathbf{C}|$ substantially influence downstream performance, we set $|\mathbf{C}|$ proportional to the number of classes with various (expansion) factors $\{0.5, 1, 3, 5\}$. All models were trained on a single NVIDIA RTX 2080 GPU and a full training run took from few minutes to a maximum of 1–2 days depending on dataset size and number of concepts $|\mathbf{C}|$. We report top-1 accuracy on the standard holdout sets throughout our experiments.

Baselines. We compared our UCBMs to Post-hoc CBM (Yuksekgonul et al., 2023) and Label-free CBM (Oikarinen et al., 2023), as they are the most related to our work. Note that Post-hoc CBM requires concept annotations and is therefore not applicable on ImageNet and Places-365.

Quality of the discovered concepts. Before we evaluated UCBMs, we verified that the discovered concepts behave faithfully. For this, we analyzed the change in cosine similarities between feature activations and concepts after the removal of relevant image parts of a certain concept; see Figure 2 and Appendix A. For example, as we remove the saw blade (concept 1985), the cosine similarity of the aforementioned concept decreases from ca. 0.5 to around 0.25 (Figure 2). We also verified that concepts are semantically consistent and human-understandable, as seen in the top activating crops throughout this paper. Please refer to, e.g., Fel et al. (2023a) for an extensive analysis.

[3]Provided at https://github.com/osmr/imgclsmob.
[4]Provided at https://github.com/Trustworthy-ML-Lab/Label-free-CBM.

Figure 2: **The discovered concepts exhibit faithful behavior.** Removing the saw blade (right) from the original image (left) shrinks the alignment score of the respective concept 1985 (blue). Concepts are represented by their most activating crops. Additional results are provided in Appendix A.

Table 1: **UCBMs outperform the baselines and reduce the gap to the original, black-box model.** We report mean top-1 accuracy with standard deviation across three training runs (we kept the discovered concepts fixed). UCBMs used $|\mathbf{C}| = 3000$ (ImageNet), 200 (CUB), and 1825 (Places-365) concepts. Post-hoc CBM used 112 (CUB) concepts and Label-free CBM used 4521 (ImageNet), 212 (CUB), and 2008 (Places-365) concepts after its concept removal step. For fair comparison, we also varied the number of concepts $|\mathbf{C}|$ for UCBMs and Figure 3 shows that UCBMs *Pareto-dominate* the baselines. Lastly, UCBM w/o concept selection slightly outperforms the variants with concept selection, which *trade-off* performance for increased sparsity (see Table 2 and Appendix D).

		Top-1 test accuracy		
Approach	Sparse?	ImageNet	CUB	Places-365
Original, black-box model	✗	80.9	76.7	53.69
Post-hoc CBM (Yuksekgonul et al., 2023)	(✓)	n/a	58.80*	n/a
Label-free CBM (Oikarinen et al., 2023)	(✓)	78.09	74.38	50.67
UCBM w/o concept selection	(✓)	**79.80** ± **0.027**	**75.15** ± **0.037**	**52.41** ± **0.028**
UCBM with ReLU concept selector	✓	79.07 ± 0.029	74.61 ± 0.128	50.86 ± 0.021
UCBM with JumpReLU concept selector	✓	*79.49* ± 0.016	74.57 ± 0.290	*51.24* ± *0.019*
UCBM with TopK concept selector	✓	79.32 ± 0.009	*74.96* ± *0.083*	51.20 ± 0.050

*: reported by Yuksekgonul et al. (2023).

3.1 Performance and sparsity results

How do UCBMs perform? Table 1 shows that UCBMs outperform the baseline methods across all datasets. Thereby, they also close the performance gap to the original, black-box model. We also find that UCBMs without concept selection achieve better performance than UCBMs with concept selection. Note that this is expected, as the concept selection variants trade-off performance for more sparsity (Table 2). We further investigate this trade-off in Appendix D.

We found that performance is strongly influenced by the total number of concepts $|\mathbf{C}|$ used. In Figure 3, we varied the number of concepts to assess this and, as expected, observe that increasing $|\mathbf{C}|$ improves performance. Notably, our UCBMs Pareto-dominate the baselines, confirming that their superiority from Table 1 is not due to the total number of concepts chosen.

How sparse are UCBMs' decisions? To assess sparsity, we computed the mean number of concepts that actively influence the classification decision per input. We considered concepts with non-zero contribution as actively influencing the classification. Specifically, concept $\mathbf{c}_j$ is considered active for the classification of input $\mathbf{x}_i$ if

$$\pi(\mathbf{x}_i)_j \neq 0 \ \wedge \ \exists \, y_i \in \{1, ..., |\mathcal{Y}|\} \text{ such that } \mathbf{W}_{y_i,j} \neq 0 \quad . \tag{7}$$

Table 2 shows that UCBMs with concept selection use significantly fewer concepts globally for classification than Label-free CBM or UCBM without concept selection. For example, on ImageNet, UCBM with TopK concept selector uses an average of 42.0 concepts per input (1.4% of the available concepts), while Label-free CBM and UCBM without concept selection use averages of 4238.0 (93.74%) or 3000.0 (100%), respectively. We find similar differences for CUB and Places-365.

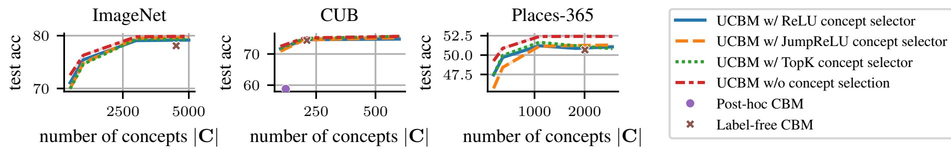
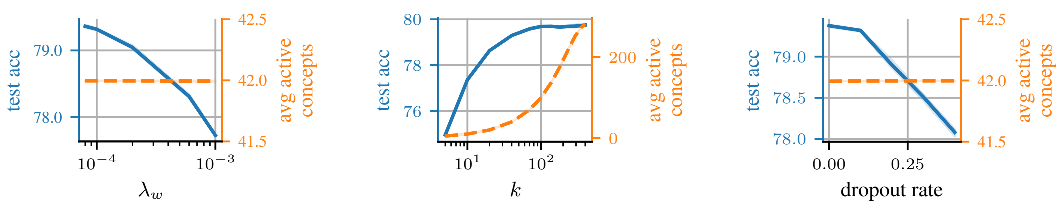


Figure 3: **UCBMs Pareto-dominate the baselines.** We varied the number of available concepts $|\mathbf{C}|$. As expected, we found that the more available concepts, the better the downstream performance. Importantly, our UCBMs Pareto-dominate the baseline methods.

Table 2: **The concept selection mechanism leads to substantially fewer concepts being used in the classification.** We report the mean number of active concepts with standard deviation according to Equation 7. Parentheses show their percentage relative to the total number of concepts $|\mathbf{C}|$. Label-free CBM and UCBM without input-dependent concept selection use substantially more concepts than our UCBM variants with concept selection.

	Mean number of active concepts (according to Equation 7)		
Approach	ImageNet	CUB	Places-365
Label-free CBM (Oikarinen et al., 2023)	4238.0 ± 0.19 (93.7%)	211.9 ± 0.05 (100%)	1820.0 ± 0.12 (90.6%)
UCBM w/o concept selection	3000.0 ± 0.0 (100%)	200.0 ± 0.0 (100%)	1825.0 ± 0.0 (100%)
UCBM with ReLU concept selector	47.8 ± 0.02 (1.6%)	61.0 ± 0.3 (30.5%)	162.4 ± 0.12 (8.9%)
UCBM with JumpReLU concept selector	42.8 ± 0.07 (1.4%)	62.3 ± 1.13 (31.2%)	166.2 ± 0.94 (9.1%)
UCBM with TopK concept selector	42.0 ± 0.00 (1.4%)	64.2 ± 0.00 (32.1%)	162.0 ± 0.00 (8.9%)



(a) Performance vs. λ_w. (b) Performance vs. k. (c) Performance vs. dropout.

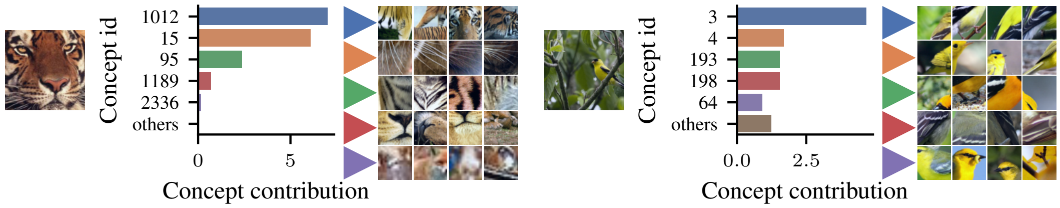
Figure 4: **Sensitivity analysis over λ_w (a), k (b), and dropout (c) on ImageNet.** Larger λ_w and smaller k lead to worse performance. Smaller k leads to higher sparsity. For dropout, there is no clear relation (esp. on the other datasets). Results for the other datasets are provided in Appendix E.

Sensitivity analysis. We varied λ_w (Figure 4a), k (Figure 4b), and dropout rate (Figure 4c) to analyze their impact on performance and sparsity. We find that only k controls sparsity (Equation 7) in TopK, whereas for the other concept selectors, all hyperparameters affect sparsity (see Appendix E). We consider this is as an advantage of TopK, as it disentangles the influence of the hyperparameters. This is discussed in more detail in Appendix E. For performance, we find that larger λ_w and smaller k lead to worse performance. For dropout rate, there typically seems to be a sweet spot.

3.2 Interpretability of UCBM

Explainable sample-wise decisions. Figure 5 shows qualitative examples of the most contributing concepts with their contribution strength (contribution of concept $\mathbf{c}_j$ to class y_i: $|\mathbf{W}_{y_i,j}\pi(\mathbf{x}_i)_j|$). We find that the most contributing concepts are relevant to both the input and prediction, while also being diverse. For example, UCBM with TopK concept selector focuses on concepts such as 'tiger striped fur', 'whiskers' or 'big cats' snouts' for the tiger in Figure 5a, or the 'bright yellow plumage' of the American goldfinch in Figure 5b.

Figure 6 compares the explanation of our UCBM with TopK concept selector and Label-free CBM (more examples in Appendix F). We find that UCBM relies on fewer concepts, that are present in the image and relevant to the predicted class. In contrast, Label-free CBM often relies on concepts that are correlated with the predicted class but absent in the image. This is especially pronounced for misclassifications (Figures 17f to 17i in Appendix F).



(a) 'tiger' (ImageNet), conf.: 93.68%. (b) 'American goldfinch' (CUB), conf: 92.51%.

Figure 5: **Decisions of UCBM with TopK concept selector rely on few reasonable and diverse concepts.** Results on ImageNet (a) and CUB (b). Additional examples are provided in Appendix F.

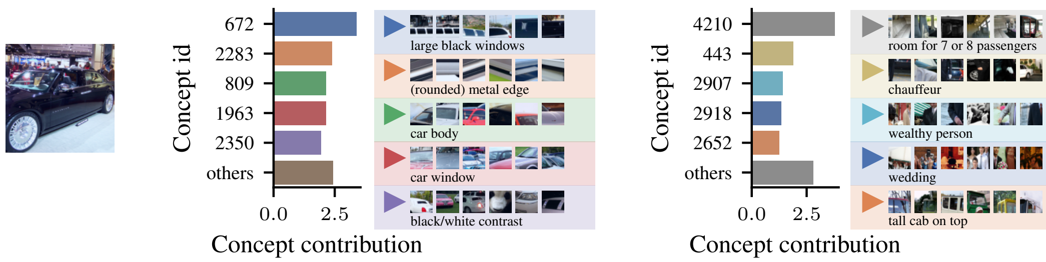


Figure 6: **The decision of UCBM with TopK concept selector (left) is more comprehensible than that of Label-free CBM (right).** Our approach relies on concepts that are present in the image and relevant to the prediction, whereas Label-free CBM tends to use concepts that are not even present, which is particularly pronounced for misclassifications. Appendix F provides additional examples.

User study on explainable sample-wise decisions. To corroborate the qualitative results from above, we conducted a user study to assess the interpretability of UCBM with TopK concept selector compared to Label-free CBM. Specifically, we evaluated the comprehensibility of their explanations. Note that the approaches present their concepts differently: UCBM and Label-free CBM use visual or textual concept representations, respectively. Thus, for fair comparison, we labeled concepts or retrieved images using SigLIP SoViT-400m (Zhai et al., 2023; Alabdulmohsin et al., 2023). Further details on the user study design are provided in Appendix G.

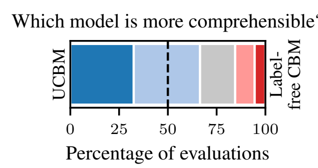


Figure 7: **Users strongly prefer UCBM.** From clearly UCBM (blue) to clearly Label-free CBM (red).

Figure 7 shows that users strongly preferred UCBM over Label-free CBM, corroborating the qualitative results shown in Figures 5 and 6 and Appendix F. Further analysis is provided in Appendix G.

Explainable class-level decision rules. To derive class-level decision rules, we computed the average contribution of each concept for a class. Figure 8 shows the top-3 concepts for two classes. We find that UCBM with TopK concept selector focuses on reasonable, human-understandable concepts relevant to each class. For example, Figure 8a shows that UCBM bases its classification of pineapples on the typical 'pineapple's texture' or its 'leaves'.

3.3 Case study: Correcting errors using a large vision-language model

In this subsection, we show how a large vision-language model (GPT-4o (Achiam et al., 2023)) can guide us to correct errors in UCBMs (specifically, a UCBM with TopK concept selector trained on ImageNet). We prompted the model asking it to adjust the weights of the sparse linear classifier $\mathbf{W}$ in UCBMs (Equation 6) to correct an error without affecting the classification of other inputs. The prompt included the misclassified input image, the top-5 concepts, and their contributions for both the misclassified and correct class. For an example of the prompt, see Appendix J. During initial experiments, we found that the suggested changes, $\Delta\mathbf{W}$, were sometimes too strong, leading to errors of previously correct inputs. To address this, we ran a grid search on the training set of ImageNet to find optimal weighing factors $\beta_i \in [0, 1]$ for each proposed change $\Delta\mathbf{W}_i$.

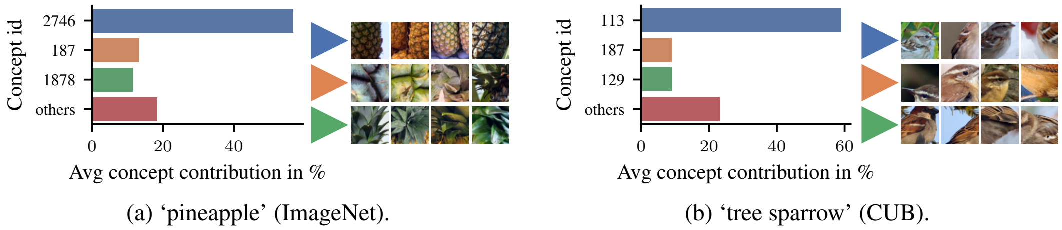
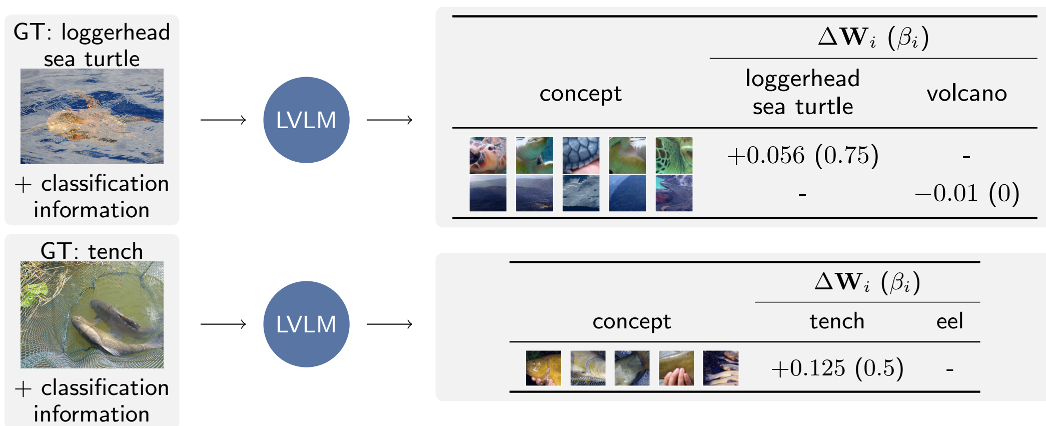


Figure 8: **UCBM with TopK concept selector uses concepts that are relevant to the classes** (represented by the most activating crops). Results for ImageNet (a) and CUB (b). Additional examples are provided in Appendix H.



example	prediction, before → after edit	test accuracy, before → after edit
top	volcano → loggerhead sea turtle	79.322 → 79.322
bottom	eel → tench	79.322 → 79.328

Figure 9: **UCBMs are intervenable.** We used a large vision-language model to help us to correct errors by guiding the edits of the weights of UCBM with TopK concept selector.

Figure 9 shows two examples that were correctly classified after applying the weight adjustments proposed by the large vision-language model. This demonstrates the intervenability of UCBMs and illustrates the potential use case of large vision-language models to automatically identify and correct the traceable causes of errors of UCBMs (or other concept-based models).

4 Related works

Concept-based models. Concept Bottleneck Models (CBMs) (Koh et al., 2020) are trained to directly leverage concepts in their classifications (Lampert et al., 2009; Kumar et al., 2009). Many works highlighted (and partially addressed) the limitations of them (Margeloiu et al., 2021; Mahinpei et al., 2021; Havasi et al., 2022; Marconato et al., 2022; Raman et al., 2024). Other work improved the performance-interpretability trade-off (Espinosa Zarlenga et al., 2022; Yang et al., 2023) or extended them beyond image classification (Ismail et al., 2023; Zarlenga et al., 2023).

The most related methods to our work convert a pretrained black-box model into a CBM post-hoc (Yuksekgonul et al., 2023; Oikarinen et al., 2023; Menon & Vondrick, 2023; Marcinkevičs et al., 2024). These approaches alleviate the need for costly concept annotations by leveraging language models, like GPT-3 (Brown et al., 2020), to automatically generate class-specific descriptions and vision-language models, like CLIP (Radford et al., 2021), to learn a mapping from a black-box model's uninterpretable features to these concepts. In contrast to these, we do not presume which concepts the black-box model has learned, but find the ones that most accurately decompose the black-box model's features in an unsupervised manner. Concurrently, akin to our first contribution, Rao et al. (2024) discovered concepts with sparse autoencoders. In contrast to the aforementioned works, we also introduced a novel input-dependent concept selection mechanism that dynamically retains only a sparse set of concepts for each input.

Concept discovery. Early work searched for neuron-aligned concepts (Bau et al., 2017; Olah et al., 2017), while later works, inspired by the superposition hypothesis (Kim et al., 2018; Elhage et al., 2022), went beyond this to (linear) vector (Kim et al., 2018; Zhou et al., 2018; Ghorbani et al., 2019; Zhang et al., 2021; Zou et al., 2023; Fel et al., 2023b; Huben et al., 2024; Stein et al., 2024) or linear subspace (Vielhaben et al., 2023) concept representations. Early work needed costly annotated datasets to find concepts through supervision. Later work overcame this bottleneck by formulating concept discovery as a dictionary learning problem (Fel et al., 2023a).

Model editing. Model editing aims to modify a model's weights to remove a bias or correct errors. Previous work edited knowledge in large language models (Zhu et al., 2020; Meng et al., 2022), generative image models (Bau et al., 2020; Gandikota et al., 2023), or modified a classifier's prediction rules (Santurkar et al., 2021; Oikarinen et al., 2023). These works relied on, e.g., human intervention or hypernetworks, whereas we leverage large vision-language models to inform model editing.

5 Limitations & future work

The main limitation (or advantage) of our approach is that discovered concepts are only represented visually, not textually. While images may be more informative, texts aid faster and easier interpretability. To obtain textual descriptions of concepts, we could manually label concepts. However, this does not scale to large amounts of concepts. Thus, we also experimented with automatic concept labeling through large vision-language models (GPT-4o (Achiam et al., 2023)), see Appendix I for details. While we found it to yield overall good concept descriptions, we also found many instances with poor descriptions; especially for non-object-centric or more abstract concepts.

Another limitation of our approach is that we only extract concepts from the bottleneck layer of black-box models. We conjecture that the use of concepts throughout the feature hierarchy of these models may be beneficial for concept-based models in terms of performance and/or interpretability, as such a hierarchy is also learned by these models (Zeiler & Fergus, 2014). For instance, an early layer could find concepts for 'windows', 'car body', or 'wheels', while a later layer assembles them to a 'car' concept (Olah et al., 2020).

6 Conclusion

We presented UCBMs, which convert pretrained black-box models into interpretable concept-based models by discovering the concepts that the model has learned through unsupervised concept discovery. We further introduced a novel input-dependent concept selection that only retains the concepts most relevant for classifications. Our experiments show that UCBMs outperform previous methods, while being substantially more sparse globally. Finally, we qualitatively and quantitatively validated the interpretability of UCBMs, and showcased how large vision-language models can guide the editing of UCBMs to correct its errors.

REFERENCES

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

Ibrahim M Alabdulmohsin, Xiaohua Zhai, Alexander Kolesnikov, and Lucas Beyer. Getting vit in shape: Scaling laws for compute-optimal model design. *Advances in Neural Information Processing Systems*, 2023.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6541–6549, 2017.

David Bau, Steven Liu, Tongzhou Wang, Jun-Yan Zhu, and Antonio Torralba. Rewriting a deep generative model. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pp. 351–369. Springer, 2020.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv*, 2013.

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

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Bogdan Dumitrescu and Paul Irofti. *Dictionary learning algorithms and applications*. Springer, 2018.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *arXiv*, 2022.

N. Benjamin Erichson, Zhewei Yao, and Michael W. Mahoney. Jumprelu: A retrofit defense strategy for adversarial attacks. *arXiv*, 2019.

Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Zohreh Shams, Frederic Precioso, Stefano Melacci, Adrian Weller, et al. Concept embedding models: Beyond the accuracy-explainability trade-off. *Advances in Neural Information Processing Systems*, 35:21400–21413, 2022.

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, 2023a.

Thomas Fel, Agustin Picard, Louis Bethune, Thibaut Boissin, David Vigouroux, Julien Colin, Rémi Cadène, and Thomas Serre. Craft: Concept recursive activation factorization for explainability. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023b.

Rohit Gandikota, Joanna Materzynska, Jaden Fiotto-Kaufman, and David Bau. Erasing concepts from diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2426–2436, 2023.

Amirata Ghorbani, James Wexler, James Y Zou, and Been Kim. Towards automatic concept-based explanations. *Advances in neural information processing systems*, 32, 2019.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Aya Abdelsalam Ismail, Julius Adebayo, Hector Corrada Bravo, Stephen Ra, and Kyunghyun Cho. Concept bottleneck generative models. In *The Twelfth International Conference on Learning Representations*, 2023.

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

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv*, 2014.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept Bottleneck Models. In *International conference on machine learning*. PMLR, 2020.

Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. Attribute and simile classifiers for face verification. In *2009 IEEE 12th international conference on computer vision*, pp. 365–372. IEEE, 2009.

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 951–958. IEEE, 2009.

Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 1999.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Skq89Scxx.

Anita Mahinpei, Justin Clark, Isaac Lage, Finale Doshi-Velez, and Weiwei Pan. Promises and pitfalls of black-box concept learning models. *arXiv*, 2021.

Alireza Makhzani and Brendan Frey. K-sparse autoencoders. In *International Conference on Learning Representations*, 2014.

Ričards Marcinkevičs, Sonia Laguna, Moritz Vandenhirtz, and Julia E Vogt. Beyond concept bottleneck models: How to make black boxes intervenable? *arXiv*, 2024.

Emanuele Marconato, Andrea Passerini, and Stefano Teso. Glancenets: Interpretable, leak-proof concept-based models. *Advances in Neural Information Processing Systems*, 35:21212–21227, 2022.

Andrei Margeloiu, Matthew Ashman, Umang Bhatt, Yanzhi Chen, Mateja Jamnik, and Adrian Weller. Do concept bottleneck models learn as intended? *arXiv*, 2021.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 2022.

Sachit Menon and Carl Vondrick. Visual classification via description from large language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=jlAjNL8z5cs.

Tuomas Oikarinen, Subhro Das, Lam M. Nguyen, and Tsui-Wei Weng. Label-free Concept Bottleneck Models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=FlCg47MNvBA.

Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2(11):e7, 2017.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 5(3):e00024–001, 2020.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. Jumping Ahead: Improving Reconstruction Fidelity with JumpReLU Sparse Autoencoders. *arXiv*, 2024.

Naveen Raman, Mateo Espinosa Zarlenga, Juyeon Heo, and Mateja Jamnik. Do concept bottleneck models obey locality? *arXiv*, 2024.

Sukrut Rao, Sweta Mahajan, Moritz Böhle, and Bernt Schiele. Discover-then-name: Task-agnostic concept bottlenecks via automated concept discovery. *arXiv*, 2024.

Shibani Santurkar, Dimitris Tsipras, Mahalaxmi Elango, David Bau, Antonio Torralba, and Aleksander Madry. Editing a classifier by rewriting its prediction rules. *Advances in Neural Information Processing Systems*, 34:23359–23373, 2021.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Adam Stein, Aaditya Naik, Yinjun Wu, Mayur Naik, and Eric Wong. Towards compositionality in concept learning. In *Forty-first International Conference on Machine Learning*, 2024.

Johanna Vielhaben, Stefan Bluecher, and Nils Strodthoff. Multi-dimensional concept discovery (MCD): A unifying framework with completeness guarantees. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=KxBQPz7HKh.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset, 2011.

Eric Wong, Shibani Santurkar, and Aleksander Madry. Leveraging sparse linear layers for debuggable deep networks. In *International Conference on Machine Learning*, pp. 11205–11216. PMLR, 2021.

Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19187–19197, 2023.

Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc Concept Bottleneck Models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=nA5AZ8CEyow.

Mateo Espinosa Zarlenga, Zohreh Shams, Michael Edward Nelson, Been Kim, and Mateja Jamnik. TabCBM: Concept-based interpretable neural networks for tabular data. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=TIsrnWpjQ0.

Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13*, pp. 818–833. Springer, 2014.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

Ruihan Zhang, Prashan Madumal, Tim Miller, Krista A Ehinger, and Benjamin IP Rubinstein. Invertible concept-based explanations for cnn models with non-negative concept activation vectors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 11682–11690, 2021.

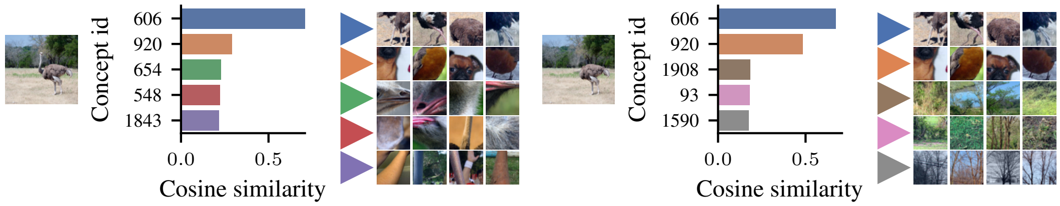
Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.

Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 119–134, 2018.

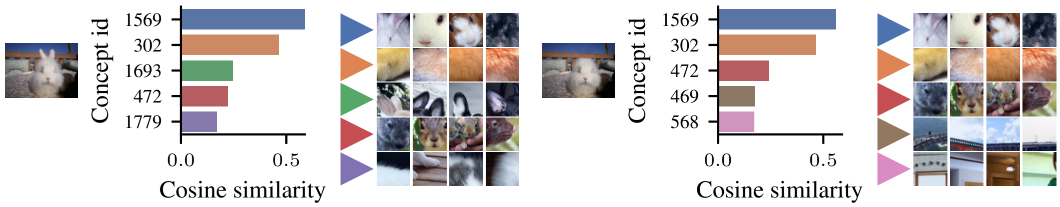
Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models. *arXiv*, 2020.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency, 2023.

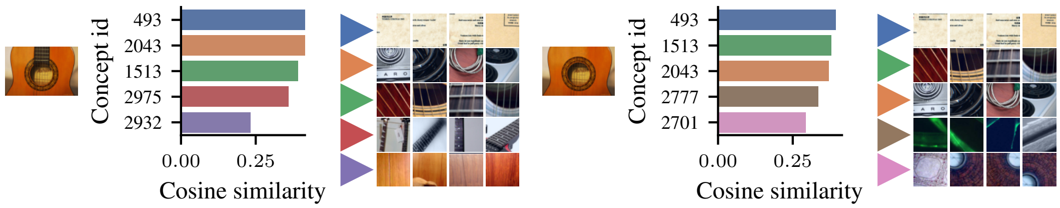
Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(2):301–320, 2005.



(a) Removing the head and neck of an ostrich makes concepts 654 (green), 549 (red), and 1843 (purple) disappear from the top-5 cosine similarities.



(b) Removing the ears of an angora rabbit makes concept 1693 (green) disappear from the top-5 cosine similarities.



(c) Removing the neck of an acoustic guitar makes concept 2975 (red) disappear from the top-5 cosine similarities.

Figure 10: **Concepts discovered in an unsupervised manner exhibit faithful behavior.** Concepts are represented by their most activating image crops. From the original image (left), we manually removed image parts (right) using an image manipulation tool and computed the concept-activation cosine similarities for an ostrich (a), an angora rabbit (b), and an acoustic guitar (c). We find that cosine similarity scores reduce, as we remove an image part where that concept or these concepts were previously present.

A Additional results for the faithfulness of discovered concepts

Figure 10 provides additional results for the faithfulness of the discovered concepts. In Figure 10a removing the head and neck of the ostrich in the input image makes concepts 654 (green), 549 (red), and 1843 (purple) disappear from the top-5 cosine similarities. Since concepts 654, 549 and 1843 represent parts of an ostrich's head or neck, this demonstrates the faithfulness of the discovered concepts. Figures 10b and 10c show similar behavior for a rabbit's ears and guitar's neck, respectively

B Further details on the interpretable classifiers

Table 3 provides the full overview over the interpretable classifiers for all of our UCBM variants from Section 2.2. Below, we provide further details for the JumpReLU and TopK concept selectors.

JumpReLU concept selector. The JumpReLU activation function (Erichson et al., 2019) is defined as follows:

$$\text{JumpReLU}_{\mathbf{o}}(\mathbf{x}) = \mathbf{x} \cdot H(\mathbf{x} - \mathbf{o}) = \begin{cases} 0, & \mathbf{x} \leq \mathbf{o} \\ \mathbf{x}, & \mathbf{x} > \mathbf{o} \end{cases} \quad , \tag{8}$$

where H is the Heaviside step function. Note that we cannot directly train our offset parameter $\mathbf{o}$. Thus, following Rajamanoharan et al. (2024), we used straight-through-estimators (Bengio et al.,

Table 3: **Overview of interpretable classifiers.** In the equations below, let $\tilde{s}(\mathbf{x}_i) := \text{sim}_{\mathbf{C}}(\mathbf{x}_i)$ denote the normalized cosine similarity between activations $\mathbf{a}_i = f(\mathbf{x}_i)$ for input $\mathbf{x}_i$ and the concepts $\mathbf{C}$, $\mathbf{W} \in \mathbb{R}^{|\mathcal{Y}|\times|\mathbf{C}|}$ and $\mathbf{b} \in \mathbb{R}^{|\mathcal{Y}|}$ are the weights and bias of the linear classifier, $\mathbf{o} \in \mathbb{R}_+^{|\mathbf{C}|}$ is a trainable offset parameter, $y_i \in \mathcal{Y}$ denotes the target class of input $\mathbf{x}_i$ for a total of $|\mathcal{Y}|$ classes, $\mathcal{L}$ denotes the task-specific loss function (in our case cross-entropy loss throughout this work), R_α is the elastic net regularization penalty (Zou & Hastie, 2005), λ_w, λ_π govern the regularization strengths, H denotes the Heaviside step function, and TopK denotes the TopK activation function (Makhzani & Frey, 2014). Note that $\tilde{s}(\mathbf{x}_i)$ is frozen during optimization. Further, note that the TopK concept selector does not need a sparsity penalty since sparsity can be controlled directly using the parameter k.

name	concept selector π	interpretable classifier
ReLU	$\pi(\mathbf{x}_i) := \max(0, \tilde{s}(\mathbf{x}_i) - \mathbf{o})$	$\min_{\mathbf{W},\mathbf{b},\mathbf{o}} \sum_{i=1}^{N} \mathcal{L}(\mathbf{W}\pi(\mathbf{x}_i) + \mathbf{b}, y_i) + \lambda_w R_\alpha(\mathbf{W}) + \lambda_\pi R_\alpha(\pi(\mathbf{x}_i))$
JumpReLU	$\pi(\mathbf{x}_i) := \tilde{s}(\mathbf{x}_i) \cdot H(\tilde{s}(\mathbf{x}_i) - \mathbf{o})$	$\min_{\mathbf{W},\mathbf{b},\mathbf{o}} \sum_{i=1}^{N} \mathcal{L}(\mathbf{W}\pi(\mathbf{x}_i) + \mathbf{b}, y_i) + \lambda_w R_\alpha(\mathbf{W}) + \lambda_\pi \sum_{j}^{\lvert\mathbf{C}\rvert} H(\tilde{s}_j(\mathbf{x}_i) - \mathbf{o}_j)$
TopK	$\pi(\mathbf{x}_i) := \text{TopK}(\tilde{s}(\mathbf{x}_i) - \mathbf{o})$	$\min_{\mathbf{W},\mathbf{b},\mathbf{o}} \sum_{i=1}^{N} \mathcal{L}(\mathbf{W}\pi(\mathbf{x}_i) + \mathbf{b}, y_i) + \lambda_w R_\alpha(\mathbf{W})$

Table 4: **Hyperparameter settings for all UCBMs variants on ImageNet — CUB — Places-365.**

	λ_π	k	λ_w	dropout rate
UCBM w/o concept selection	n/a	n/a	1e-4 — 8e-4 — 4e-4	0.1 — 0.2 — 0.2
UCBM with ReLU concept selector	2e-5 — 1e-4 — 2e-5	n/a	1e-4 — 8e-4 — 4e-4	0.1 — 0.2 — 0.2
UCBM with JumpReLU concept selector	1e-5 — 4e-7 — 4e-7	n/a	1e-4 — 8e-4 — 4e-4	0.1 — 0.2 — 0.2
UCBM with TopK concept selector	n/a	42 — 66 — 162	1e-4 — 8e-4 — 4e-4	0.1 — 0.2 — 0.2

2013) to make $\mathbf{o}$ trainable. Specifically, we adopted the pseudo-derivates from Rajamanoharan et al. (2024):

$$\frac{\tilde{\partial}}{\tilde{\partial}\mathbf{o}}\text{JumpReLU}_{\mathbf{o}}(\mathbf{x}) := -\frac{0}{\epsilon}K(\frac{\mathbf{x}-\mathbf{o}}{\epsilon}) \tag{9}$$

and

$$\frac{\tilde{\partial}}{\tilde{\partial}\mathbf{o}}H(\mathbf{x}-\mathbf{o}) := -\frac{1}{\epsilon}K(\frac{\mathbf{x}-\mathbf{o}}{\epsilon}) \quad , \tag{10}$$

where $\tilde{\partial}$ denotes the pseudo-derivative, K is a kernel (following Rajamanoharan et al. (2024) we used the rectangle function: $\text{rect}(\mathbf{x}) := H(\mathbf{x} + \frac{1}{2}) - H(\mathbf{x} - \frac{1}{2})$), and ϵ can be seen as the KDE bandwidth.

TopK concept selector. The TopK activation function (Makhzani & Frey, 2014) is defined as follows:

$$\text{TopK}_k(\mathbf{x})_i = \begin{cases} \mathbf{x}_i & \text{if } \mathbf{x}_i \in \text{top-k}(\mathbf{x}), \\ 0 & \text{otherwise,} \end{cases} \quad . \tag{11}$$

Note that we can directly control the sparsity through the hyperparameter k and the TopK concept selector becomes equivalent to the identity function as $k = |\mathbf{C}|$.

Why do we add a trainable offset parameter o? We introduce the additional trainable offset parameter $\mathbf{o} \in \mathbb{R}_+^{|\mathbf{C}|}$ to allow the classifier to adapt to different ranges of alignment scores for each concept. The reasons for this is that the distribution of alignment scores can vary between concepts. For example, for one concept, the alignment scores may be more uniformly distributed, indicating a more ambiguous presence of the concept. For another concept, the alignment scores might follow a bimodal distribution, indicating two distinct modes that indicate the object is present or absent. The offset parameter helps the classifier in such cases to account for such different distributions.

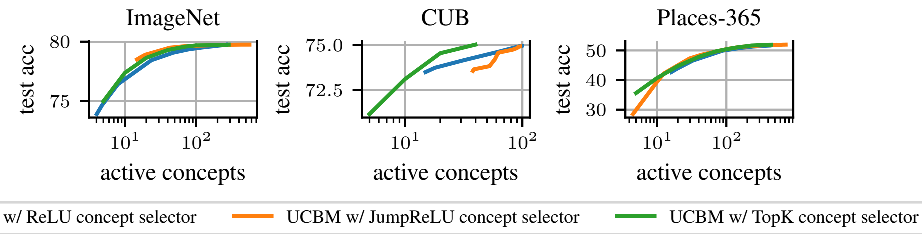


Figure 11: **Trade-off curves between sparsity and performance of the three UCBM variants.** We plot the mean number of active concepts per input according to Equation 7 as we decrease k (for TopK) or increase λ_π (for the others). Note that we only plot the Pareto-optimal points.

Table 5: **UCBM with TopK concept selector requires less concepts to explain a prediction**. We report the mean and the standard deviation of the number of concepts that are required to explain 95% of the prediction (see Equation 12 for more details).

	#concepts to explain 95% of the prediction (Equation 12)		
Approach	ImageNet	CUB	Places-365
UCBM w/o concept selection	9.51 ± 0.016	6.95 ± 0.05	46.12 ± 0.075
UCBM with ReLU concept selector	4.93 ± 0.002	5.93 ± 0.051	17.59 ± 0.031
UCBM with JumpReLU concept selector	6.27 ± 0.007	5.75 ± 0.099	28.53 ± 0.044
UCBM with TopK concept selector	6.15 ± 0.011	6.54 ± 0.041	28.22 ± 0.023

C Hyperparameter settings

Table 4 provides the hyperparameters (λ_π, k, λ_w, dropout rate) for all our UCBMs variants. We chose those hyperparameters such that they yielded a good trade-off between performance, sparsity, and fair comparability (see Figure 4 and Appendices D and E). It is important to note that we first optimized λ_π for the ReLU and JumpReLU concept selectors and then set k accordingly, as we found that its relationship to sparsity (c.f., Equation 7) is straightforward.

D Trade-off between performance and sparsity

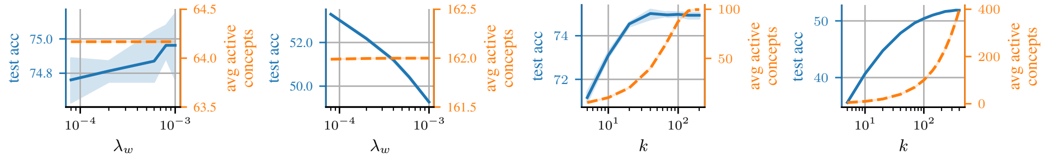
The hyperparameter k for UCBM with TopK concept selector, or λ_π for UCBM with ReLU or JumpReLU concept selector, governs the model's sparsity (c.f., Equation 7). It is important to note that this also affects performance–more sparse models typically have degraded performance. Figure 11 illustrates this trade-off. We find that each concept selector enables 'smooth' control over this trade-off. This allows practitioners to set these hyperparameters according to their desired balance between sparsity (and better interpretability) and performance, based on the requirements of their application.

Beyond the sparsity measurements and discussion for Table 2, we computed how many concepts the models need to explain their prediction of a class. For this, we computed the mean number of concepts that are required to explain 95% of a model's prediction per sample:

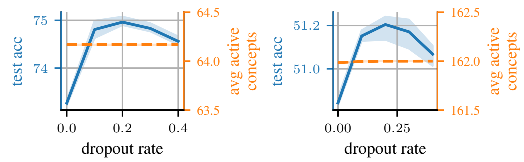
$$\frac{1}{N}\sum_{i=1}^{N} C'_i \text{ , where } \min_{C'_i \subseteq \{1,\ldots,|\mathbf{C}|\}} |C'_i| \text{ s.t. } \frac{\sum_{c\in C'_i} |\mathbf{W}_{\tilde{y}_i,c}\pi(\mathbf{x}_i)_c|}{\sum_{c\in\{1,\ldots,|\mathbf{C}|\}} |\mathbf{W}_{\tilde{y}_i,c}\pi(\mathbf{x}_i)_c|} \geq 95\% \quad , \tag{12}$$

where $\tilde{y}_i$ denotes the model's prediction of input $\mathbf{x}_i$.

Table 5 shows that UCBMs with concept selector rely on fewer concepts than UCBM without concept selection. Note that relying on fewer concepts makes it easier for users to comprehend a prediction since they do not need to inspect a lot of concepts.

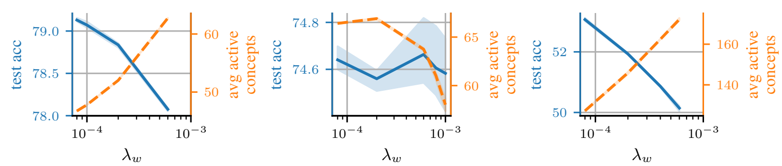


(a) Performance vs. λ_w. (b) Performance vs. k.

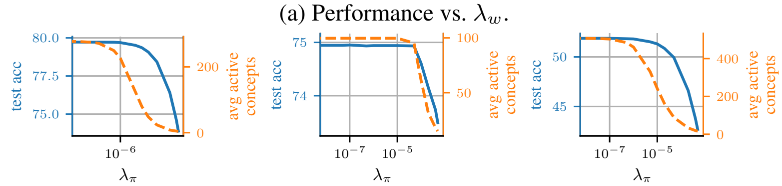


(c) Performance vs. dropout.

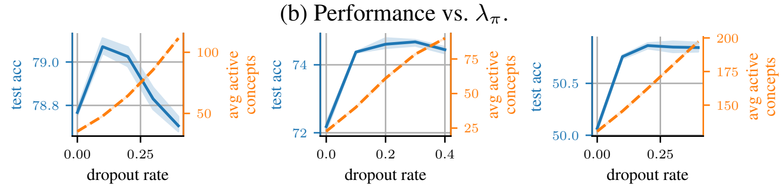
Figure 12: **Sensitivity analysis for UCBM with TopK concept selector over λ_w (a), k (b), and the dropout rate (c) for CUB (left) and Places-365 (right).**



(a) Performance vs. λ_w.



(b) Performance vs. λ_π.



(c) Performance vs. dropout.

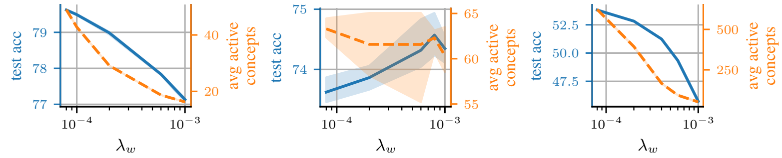
Figure 13: **Sensitivity analysis for UCBM with ReLU concept selector over λ_w (a), λ_π (b), and the dropout rate (c) for ImageNet (left), CUB (middle), and Places-365 (right).**

E ADDITIONAL SENSITIVITY ANALYSIS RESULTS

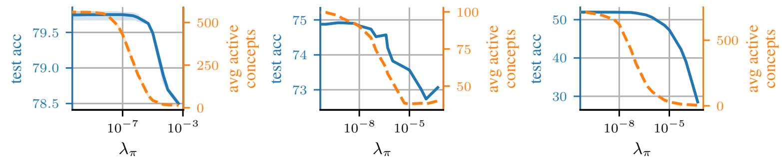
Figure 12 provides the results for the sensitivity analysis for UCBM with TopK concept selector on CUB and Places-365. Figures 13 and 14 provide the results for UCBM with ReLU or JumpReLU concept selector, respectively.

As also discussed in Appendix D, the hyperparameters k (for TopK) or λ_π (for ReLU and JumpReLU) control the trade-off between performance and sparsity. Regarding the other hyperparameters, λ_w and dropout rate, it is important to observe that they have less influence on the sparsity for the TopK concept selector than for the other concept selectors. We consider this as an advantage of the TopK concept selector, as it reduces the interaction between hyperparameters. This makes hyperparameter tuning simpler and simplifies the interpretation: k governs the average number of active concepts per sample, λ_w governs the number of concepts used per class, and the dropout rate influences whether the classifier relies on a broader or narrower set of concepts.

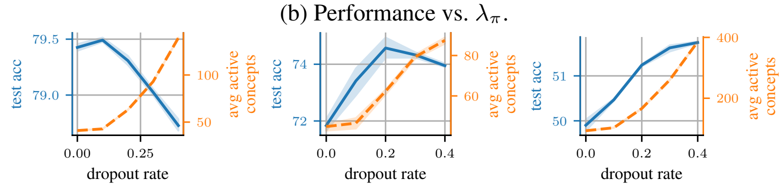
For λ_w, we find that increasing it typically leads to worse performance and a smaller average number of active concepts per sample. Interestingly, for the UCBMs with ReLU concept selector trained on



(a) Performance vs. λ_w.

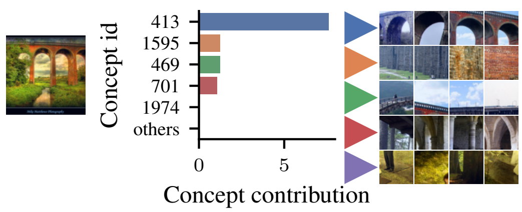


(b) Performance vs. λ_π.

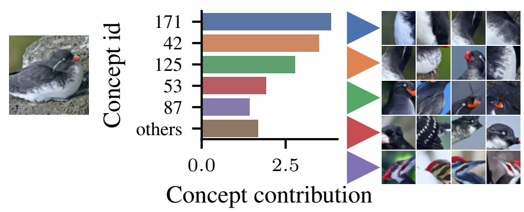


(c) Performance vs. dropout.

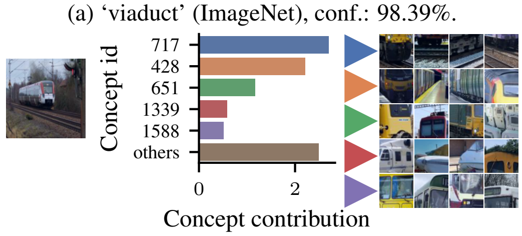
Figure 14: **Sensitivity analysis for UCBM with JumpReLU concept selector over λ_w (a), λ_π (b), and the dropout rate (c) for ImageNet (left), CUB (middle), and Places-365 (right).**



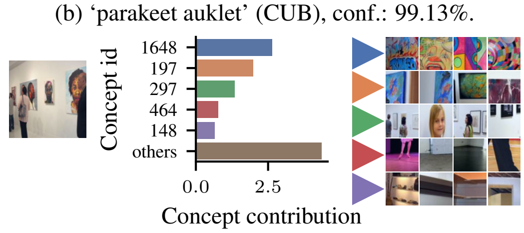
(a) 'viaduct' (ImageNet), conf.: 98.39%.



(b) 'parakeet auklet' (CUB), conf.: 99.13%.



(c) 'railroad track' (Places-365), conf.: 76.15%.



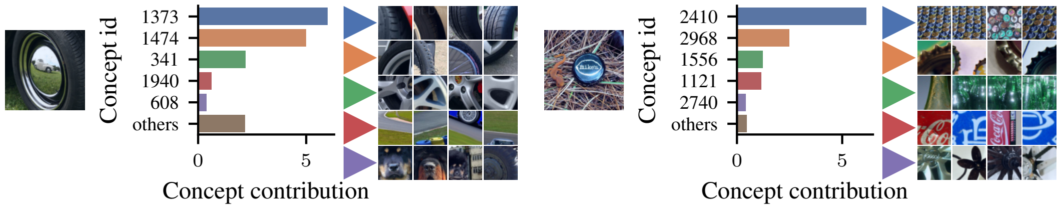
(d) 'art gallery' (Places-365), conf.:72.55%.

Figure 15: **Explainable decisions by UCBM** with TopK concept selector on ImageNet (a), CUB (b), and Places-365 (c, d) classes. The model's prediction are comprehensible and typically rely on only few concepts.

ImageNet and Places-365, we observe the opposite behavior. For the dropout rate, a higher dropout rate results in more active concepts per sample, though its relationship with performance is less straightforward.

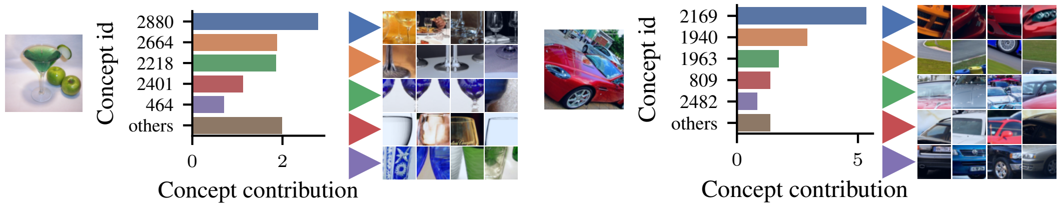
F Additional Examples of Explainable Decisions

Additional examples for sample-wise explanations. Figure 15 provides more examples of explainable decision of UCBM with TopK concept selector on ImageNet, CUB, and Places-365. We typically find that our method relies on a small set of concepts that are present in the images, human-comprehensible and class-relevant. For instance, for the viaduct in Figure 15a, UCBM uses class-relevant concepts (e.g., 'arches', 'stones', or 'walkway'). For the 'railroad track' in Figure 15c, it



(a) GT: 'station wagon', pred.: 'car wheel'.

(b) GT: 'eft', pred.: 'bottle cap'.



(c) GT: 'granny smith apple', pred.: 'goblet'.

(d) GT: 'car wheel', pred.: 'sports car'.

Figure 16: **The most contributing concepts explain the misclassifications on ImageNet of UCBM with TopK concept selector.** a: The image shows a station wagon mirrored in a car wheel. Most of the top-5 concepts are related to car wheels, which explains that the model only focuses on the car wheel itself instead of the mirrored station wagon. This clearly explains why the model predicts 'car wheel' instead of 'station wagon'. b: The image shows an eft next to a bottle cap. The concepts show that the model used concepts related to bottle caps, which is the object at the center of the image. c: The image shows two granny smith apples next to a goblet that was predicted by the model. The concepts reveal that the model focuses on concepts related to the goblet at the center of the image. d: The image shows a sports car, including one of its front wheels. The most important concept is related to sports cars. The other concepts also focus more on general car concepts than on the wheels.

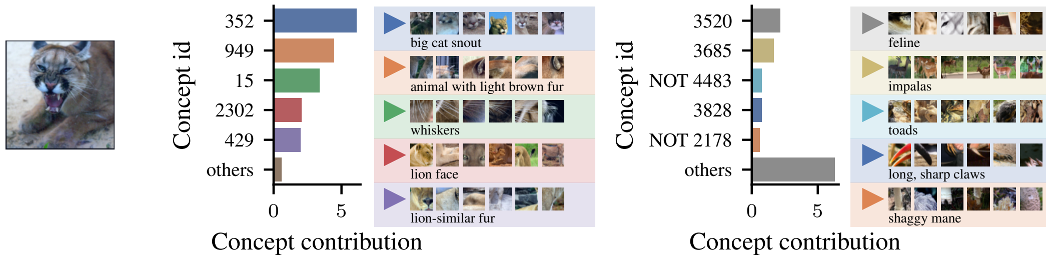
uses concepts such as 'tracks' or 'train'. Interestingly, it also uses the concept 'large window' that is also related to, e.g., buses. This indicates that UCBMs first assess if concepts are present or absent and then based on that evidence predict the class that is most likely given that.

Understanding misclassifications of UCBMs. Figure 16 shows that we can comprehend why UCBMs made a misclassification. For example, Figure 16a shows that the UCBM incorrectly predicted 'car wheel' instead of 'station wagon'. However, the image shows such station wagon mirrored in a car wheel. Looking at the most contribution concepts reveals that UCBM focused on concepts that are related to the car wheel, as it is the most salient in the image.

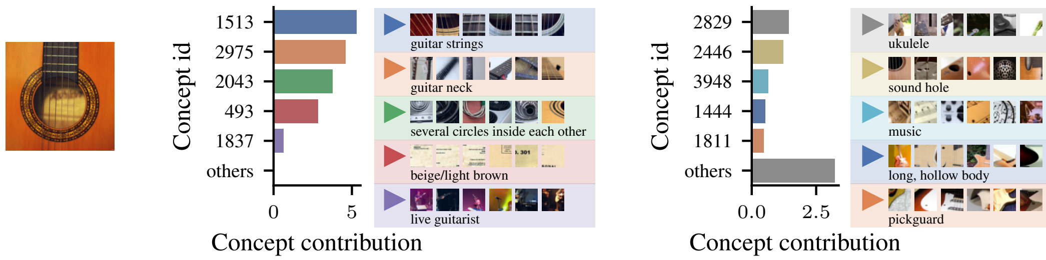
Additional examples for the comparison of UCBM to Label-free CBM. Figure 17 compares the explanations of UCBM with TopK concept selector and Label-free CBM (Oikarinen et al., 2023). We find that our approach provides more comprehensible explanations:[5] UCBM relies on intuitive concepts that are present in the image and relevant to the prediction. In contrast, Label-free CBM tends to rely on concepts that are correlated to the prediction but may not be present in the image, e.g., the concepts 'graduation markings' or 'diploma' for the prediction 'graduation cap' in Figure 17d.[6] Note that such reliance on prediction-class correlated but absent concepts is particularly pronounced for misclassifications (Figures 17f to 17i). For example, Label-free CBM relies on the concepts 'garden', 'plants', or 'rainforest' for an image that depicts an restaurant from the street (without any greenery). We believe relying on such non-visible concepts is not helpful to understand the decision of a concept-based model.

[5]These qualitative findings are further corroborated in the user study in Section 3.2 and Appendix G.

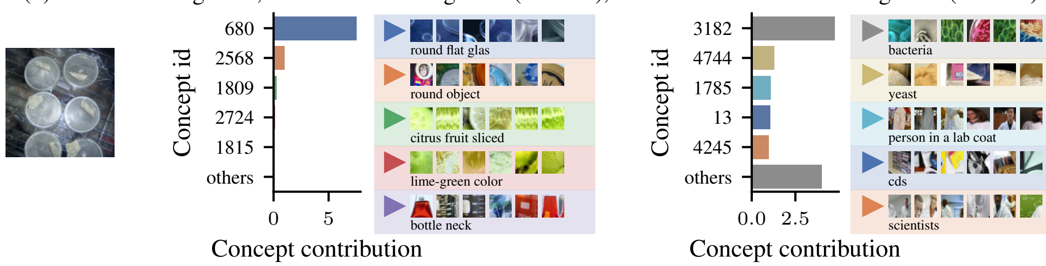
[6]We suspect the reason for this are shortcomings of CLIP's embeddings. For instance, the concepts '*graduated* cylinder' is unrelated to the prediction of 'graduation cap' in Figure 17d. However, the word 'graduated' is related to 'graduation'. Indeed, when we compute the cosine similarity of text features (we considered the following: 'graduated cylinder', 'graduation ceremony', 'graduation markings', 'graduation', 'university', 'dog', 'house'), we found that concepts related graduation have higher similarities with the graduated cylinder than the unrelated concepts. We leave further investigations for future work.



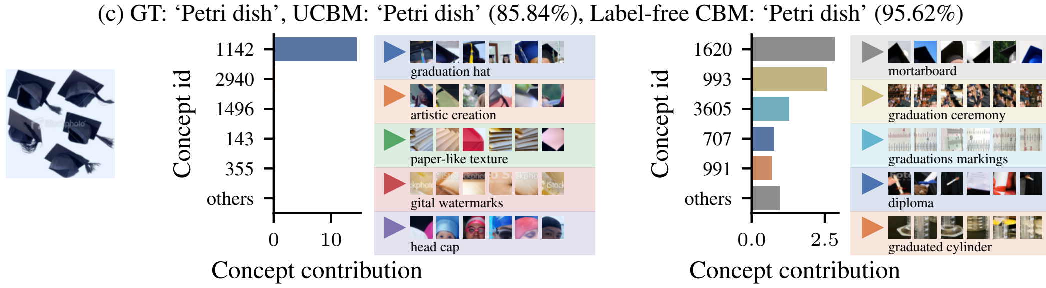
(a) GT: ‘cougar’, UCBM: ‘cougar’ (99.99%), Label-free CBM: ‘cougar’ (95.97%)



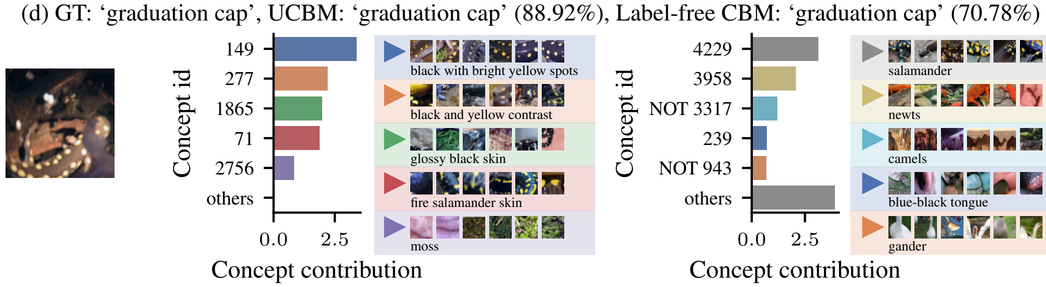
(b) GT: ‘acoustic guitar’, UCBM: ‘acoustic guitar’ (99.93%), Label-free CBM: ‘acoustic guitar’ (42.35%)



(c) GT: ‘Petri dish’, UCBM: ‘Petri dish’ (85.84%), Label-free CBM: ‘Petri dish’ (95.62%)

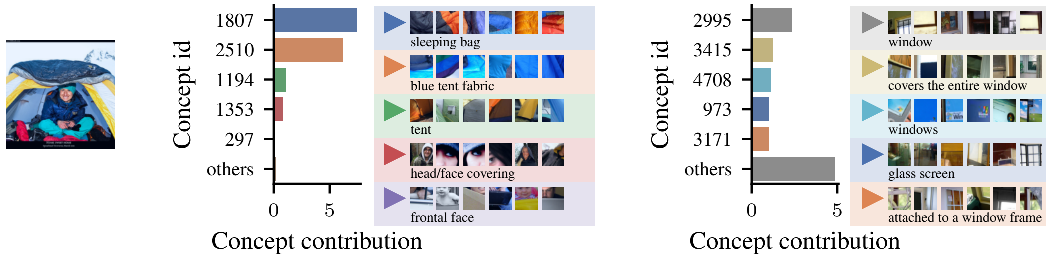


(d) GT: ‘graduation cap’, UCBM: ‘graduation cap’ (88.92%), Label-free CBM: ‘graduation cap’ (70.78%)

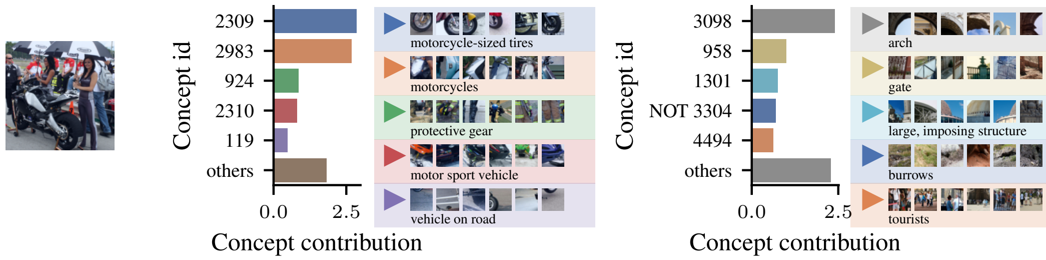


(e) GT: ‘spotted salamander’, UCBM: ‘spotted salamander’ (90.35%), Label-free CBM: ‘spotted salamander’ (92.77%)

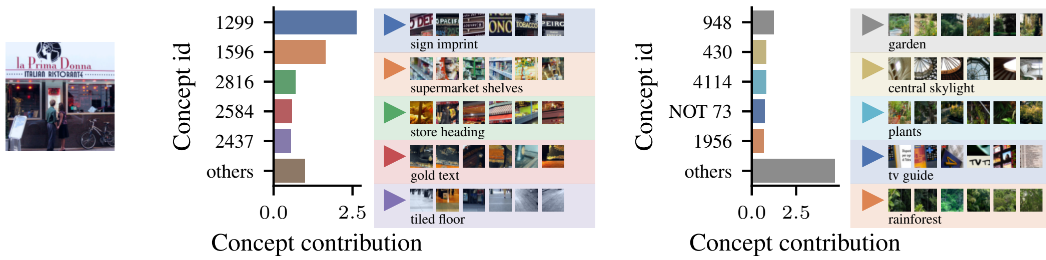
Besides that, we find that a significant part of the concept contributions of the decisions of Label-free CBM is also attributed to other concepts (bar ‘others’ in the plots). In contrast, UCBMs typically rely on fewer concepts. The benefit of this is that users have to only consider a small set of concepts in practice, making the interpretability of UCBMs’ explanations easier to comprehend.



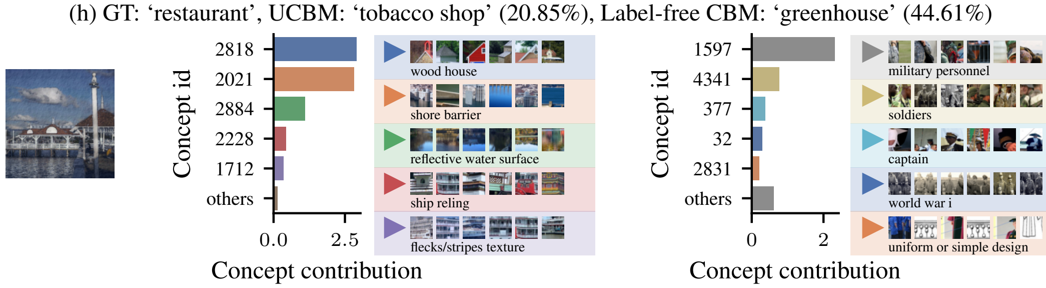
(f) GT: ‘tent’, UCBM: ‘sleeping bag’ (97.70%), Label-free CBM: ‘window screen’ (59.72%)



(g) GT: ‘umbrella’, UCBM: ‘vespa’ (54.38%), Label-free CBM: ‘triumphal arch’ (23.74%)



(h) GT: ‘restaurant’, UCBM: ‘tobacco shop’ (20.85%), Label-free CBM: ‘greenhouse’ (44.61%)



(i) GT: ‘pier’, UCBM: ‘boathouse’ (32.64%), Label-free CBM: ‘military uniform’ (6.46%)

Figure 17: **Comparison of explainable decisions of UCBM with TopK concept selector (left) vs Label-Free CBM (right)**. Subfigures a-e and f-i show correct or incorrect predictions of both CBMs, respectively. Our UCBM with TopK concept selector provides more comprehensible explanations, while Label-free CBM often relies on concepts that are not even visible in the image (this is especially pronounced for misclassifications). We suspect one reason for this are the shortcomings of CLIP’s text features that are used in Label-free CBM.

G Further details on the user study

In the user study, we studied whether users consider the explanations of the decisions of UCBM to be comprehensible. To do so, we compared the explanations of UCBM with TopK concept selector with Label-free CBM (Oikarinen et al., 2023). Both were trained on ImageNet.

Task. We asked users to assess which model provides a more comprehensible explanation from a scale from ‘Model A clearly more’ to ‘Model B clearly more’. Further, we asked for the reasons why they think one model is more comprehensible than the other.

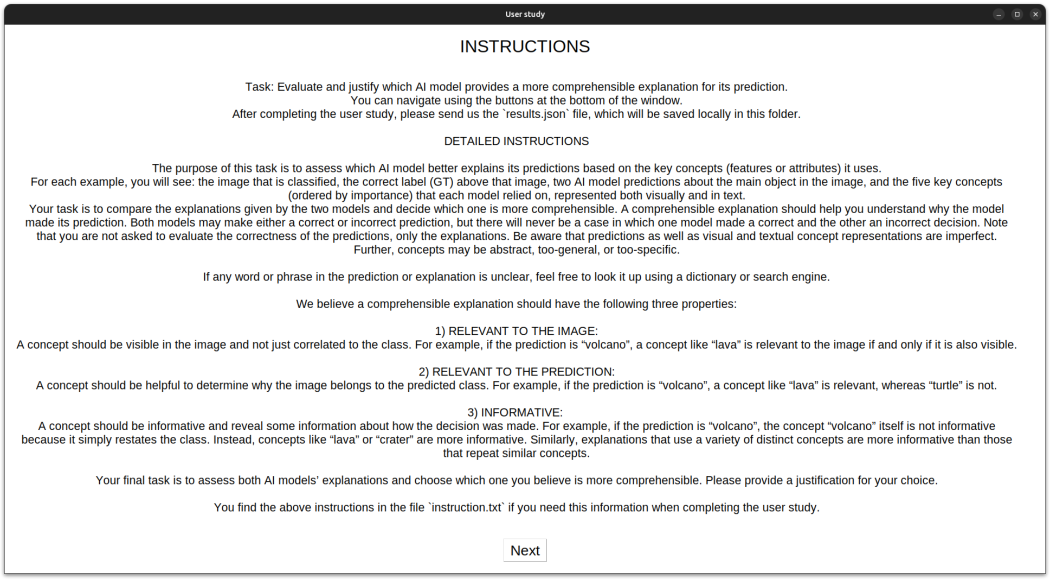


Figure 18: **Instruction text.**

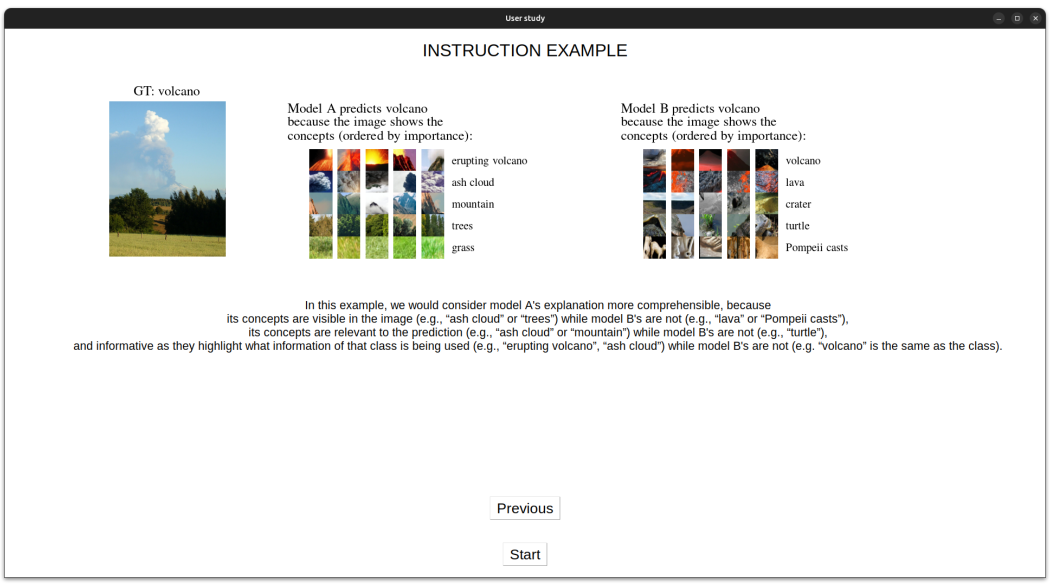


Figure 19: **Instruction example.**

User study data. We showed users local explanations based on which concepts contributed the most to the decision of each model, akin to Figures 6, 8, 15 and 16. Importantly, 20% of samples showed misclassifications of *both* models (for the other 80% both model predicted correctly).[7] We include misclassifications to also understand how comprehensible models are under errors. We believe this is an important aspect to study, as users will also interact with models that make errors in practice. For sake of this user study, we simplified the explanations by removing the concept contributions and only showed the names and top-activating image crops of the five most contributing concepts and a corresponding concept description.

Note that UCBM and Label-free CBM represent their concepts differently: UCBMs show visual representations, whereas Label-free CBM shows concept descriptions. To ensure fair comparison,

[7] No sample for which one model was correct and the other was incorrect was shown in the user study.

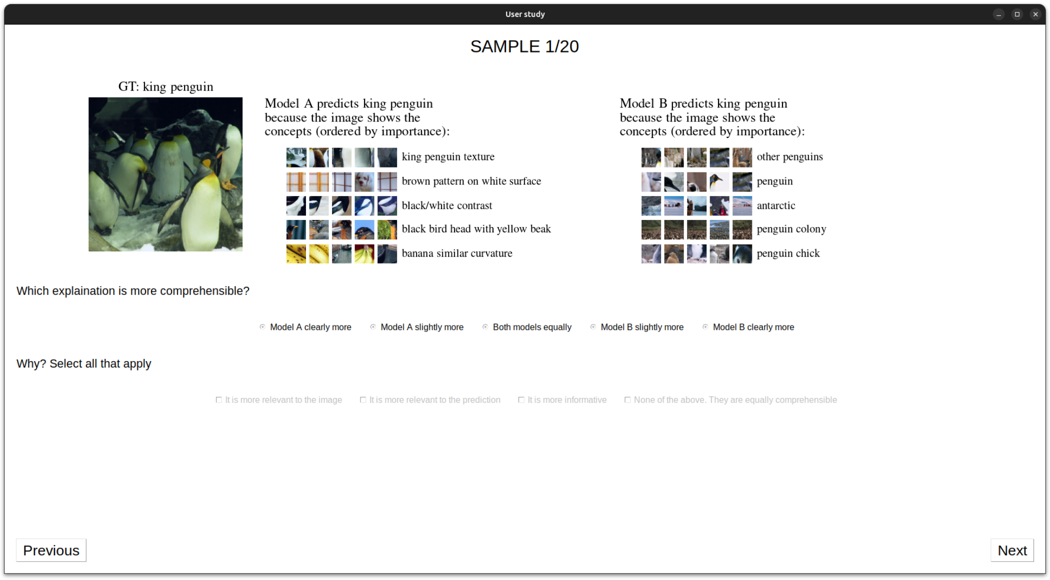
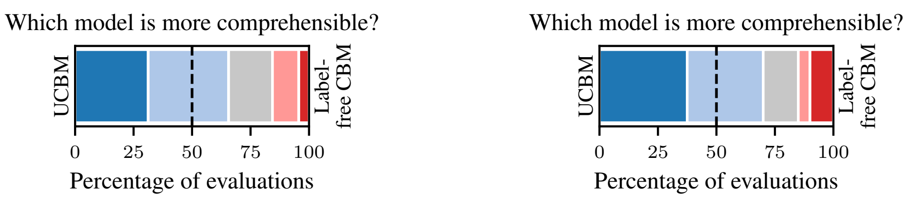


Figure 20: **User study sample.**



(a) Correct predictions only.

(b) Incorrect predictions only.

Figure 21: **Users strongly preferred UCBM with TopK concept selector over Label-free CBM for correct as well as incorrect predictions.**

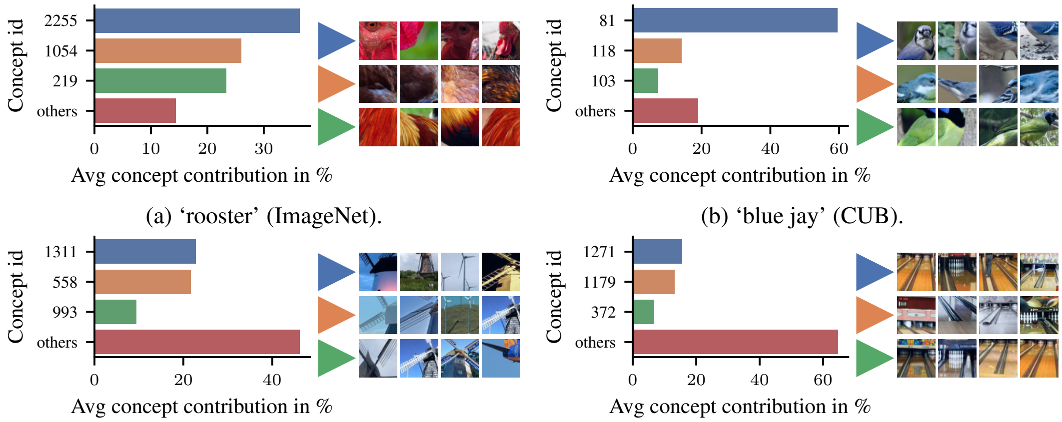
we labeled the most activating image crops of UCBM's concepts and retrieved images using SigLIP SoViT-400m (Zhai et al., 2023; Alabdulmohsin et al., 2023) for Label-free CBM's concepts.

Setup. We implemented the user study in a lightweight Python GUI so that users could run the study locally on their machine. Users were provided with the task description (Figure 18) and an example (Figure 19). After the instruction, users interacted with our user study interface (Figure 20).

We asked ten users to rate a total of 200 samples (20 per user). Users participated voluntarily and without payment. They have strong background in machine learning and related fields. However, none of them is working on concept-based models or had seen explanations of UCBM before.

Further analysis. Complementary to the results presented in Section 3.2, we conducted further analysis on the results of the user study. Figure 21 shows that users strongly preferred our UCBM with Topk concept selector over Label-free CBM in ca. 65-70% of evaluations (Label-free CBMs are only preferred in ca. 15%). Users' preference was similar for correct or incorrect predictions.

Users based their preference decisions mostly on relevance to the prediction (selected in 66.5% of the evaluations). However, relevance to the image (55%) and informativeness (55%) closely followed it.



(a) ‘rooster’ (ImageNet). (b) ‘blue jay’ (CUB).

(c) ‘windmill’ (Places-365). (d) ‘bowling alley’ (Places-365).

Figure 22: **Visualization of decision rules learned by UCBM with TopK concept selector** on ImageNet (a), CUB (b) and Places-365 (c, d).

H ADDITIONAL EXAMPLES OF EXPLAINABLE DECISION RULES

Figure 22 provides more examples of explainable decision rules of UCBM. The examples show that UCBM uses reasonable human-interpretable concepts to build the score of a specific class.

I CONCEPT LABELING WITH A LARGE VISION-LANGUAGE MODEL

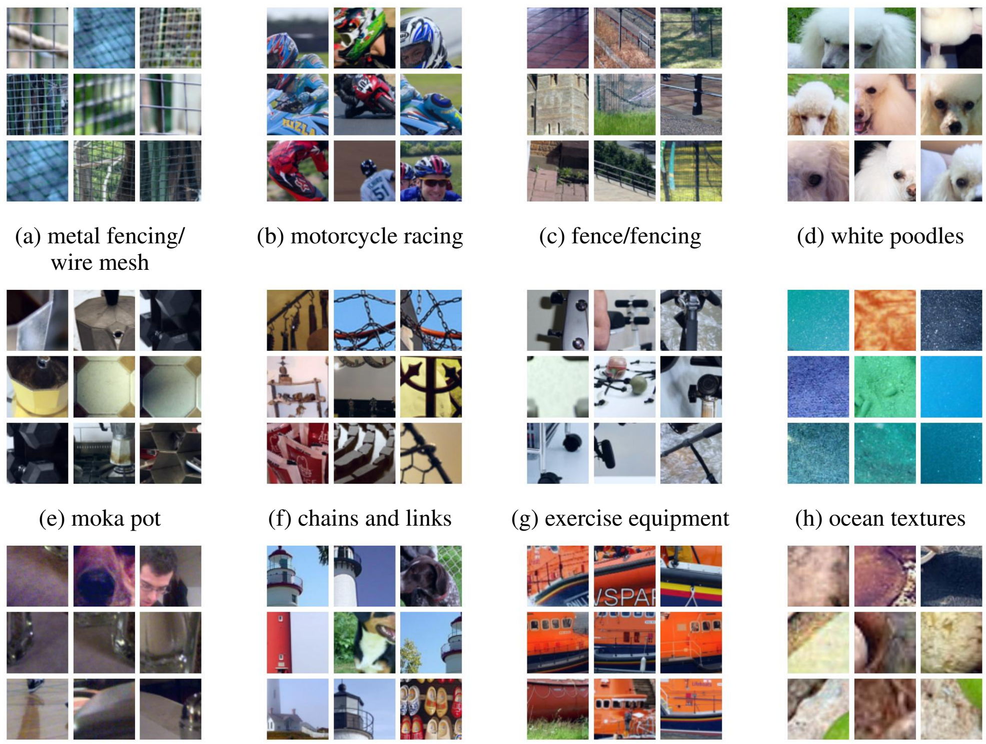
As an alternative to providing the top-activating image crops and manual concept labelling, we also experimented with large vision-language models (GPT-4o (Achiam et al., 2023)) to automatically label concepts. We prompted it with the top-9 image crops and task description:

The nine pictures within the image are matching a specific concept.
Can you describe the concept with very few words (ca. 1–3)?

Figure 23 shows the outputted concept labels for twelve, diverse concepts. Overall, we found that concept labels are mostly matching to the top image crops, e.g., Figures 23a, 23d, 23e and 23k. However, there are also instances that may not be correctly labelled. For example, the large vision-language model outputs “motorcycle racing” for the image crops in Figure 23b. While this matches well with most of the image crops, it does not for the baseball player (bottom middle) and cyclist (bottom right). We suspect that the concept is representing a more general concept for “safety equipment” instead. For another example, in Figure 23h, the large vision-language model labelled the concept as “ocean textures”. However, the image crops more likely resemble a starry sky rather than some ocean textures due to the point structure.

J EXAMPLE PROMPT TO THE LARGE VISION-LANGUAGE MODEL

Figure 24 shows an example prompt to the large vision-language model for the misclassification from the lower, left subfigure in Figure 9. Figure 25 shows the corresponding output from the large vision-language model.



(a) metal fencing/ wire mesh

(b) motorcycle racing

(c) fence/fencing

(d) white poodles

(e) moka pot

(f) chains and links

(g) exercise equipment

(h) ocean textures

(i) restaurant table/ dining experience

(j) lighthouses

(k) lifeboat

(l) tree bark/ wood textures

Figure 23: **Labeling of concepts using large vision-language models.** The subfigures' captions are the labeling/descriptions that the large vision-language model (GPT-4o (Achiam et al., 2023)) assigned to the provided concept visualizations.

The image from above leads to the following output if we are putting it into a trained concept bottleneck model.

Ground truth: tench
Prediction: eel

Top-5 model prediction sorted by the logits:
Class eel: 7.71
Class tench: 7.05
Class silver salmon: 6.76
Class snoek fish: 6.08
Class sturgeon: 5.93

Largest five concept contributions (by absolute value) for ground truth class tench, sorted by size (and their weight in the final linear layer):
Concept 2088: 5.27 (0.34)
Concept 2987: 1.48 (0.20)
Concept 162: 0.17 (0.01)
Concept 2177: 0.01 (0.00)
Concept 2479: 0.01 (0.00)

Largest five concept contributions (by absolute value) for predicted class eel sorted by size (and their weight in the final linear layer):
Concept 333: 2.94 (0.34)
Concept 1367: 2.26 (0.20)
Concept 760: 0.72 (0.11)
Concept 2088: 0.68 (0.04)
Concept 2987: 0.51 (0.07)

Furthermore, the layer from the concept similarities to the prediction is a linear layer. Which weights (concept and class) should be adjusted by how much in order to get a correct classification for this image, while not changing any other classifications? Please give an answer in case of 'increase/decrease weight of class i and concept j by x'.

Figure 24: **Input prompt to the large vision-language model to edit the linear classifier's weights to correct an error.**

To get the model to correctly classify the image as "tench" instead of "eel," we'll need to adjust the weights in the final linear layer between the relevant concepts and the classes. Here's the approach to identifying which weights should be adjusted and by how much:

Step-by-step Approach:

1. Identify Misclassification and Concept Contributions:
   - The model currently predicts "eel" with a logit of 7.71, while "tench" has a logit of 7.05.
   - To change the classification to "tench," we need the logit for "tench" to be higher than the logit for "eel."
2. Calculate Difference Needed:
   - Currently, the difference between "eel" and "tench" logits is: $7.71 - 7.05 = 0.66$.
   - To correctly classify the image as "tench," the logit for "tench" must exceed "eel." Therefore, we need to adjust the weights such that the logit for "tench" increases by at least 0.66.
3. Determine Which Concept Weights to Adjust:
   - For "tench," the largest contributing concept is 2088 with a contribution value of 5.27 and a weight of 0.34.
   - For "eel," concept 2088 is also present with a smaller contribution (0.68) and a lower weight (0.04).
   - Adjusting concept 2088's weight for "tench" might be the most straightforward way to change the output because it has a significant contribution value.
4. Calculate Weight Adjustments:
   - To increase the "tench" logit by 0.66, we need to find an increase in the weight of concept 2088 for "tench":

     $$\texttt{Increase in logit} = \texttt{concept contribution} \times \texttt{weight increase}.$$

     Here, the concept contribution for 2088 is 5.27. So, we need to find the weight increase (Δw) such that:

     $$5.27 \times \Delta w = 0.66.$$

     Solving for Δw:

     $$\Delta w = \frac{0.66}{5.27} \approx 0.125$$
5. Adjust the Weight:
   - Increase the weight between "tench" and concept 2088 by approximately 0.125.

Final Answer:

- Increase the weight of class tench and concept 2088 by 0.125.

This targeted adjustment will help correct the classification for this specific image without affecting other classifications significantly, as it focuses on the concept that most strongly contributes to the "tench" logit.

Figure 25: **Output of the large vision-language model to edit the linear classifier's weights to correct the error.**