Language in a Bottle: Language Model Guided Concept Bottlenecks for Interpretable Image Classification

Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin,
Chris Callison-Burch, Mark Yatskar
University of Pennsylvania
{yueyang1, artemisp, shzhou2, jindan, ccb, myatskar}@seas.upenn.edu

Abstract

*Concept Bottleneck Models (CBM) are inherently interpretable models that factor model decisions into human-readable concepts. They allow people to easily understand why a model is failing, a critical feature for high-stakes applications. CBMs require manually specified concepts and often under-perform their black box counterparts, preventing their broad adoption. We address these shortcomings and are first to show how to construct high-performance CBMs without manual specification of similar accuracy to black box models. Our approach, **La**nguage Guided **Bo**ttlenecks (LaBo), leverages a language model, GPT-3, to define a large space of possible bottlenecks. Given a problem domain, LaBo uses GPT-3 to produce factual sentences about categories to form candidate concepts. LaBo efficiently searches possible bottlenecks through a novel submodular utility that promotes the selection of discriminative and diverse information. Ultimately, GPT-3's sentential concepts can be aligned to images using CLIP, to form a bottleneck layer. Experiments demonstrate that LaBo is a highly effective prior for concepts important to visual recognition. In the evaluation with 11 diverse datasets, LaBo bottlenecks excel at few-shot classification: they are 11.7% more accurate than black box linear probes at 1 shot and comparable with more data. Overall, LaBo demonstrates that inherently interpretable models can be widely applied at similar, or better, performance than black box approaches.*

1. Introduction

As deep learning systems improve, their applicability to critical domains is hampered because of a lack of transparency. Efforts to address this have largely focused on post-hoc explanations [45, 51, 68]. Such explanations can be problematic because they may be incomplete or unfaithful with respect to the model's computations [46]. Models can also be designed to be inherently interpretable, but it is

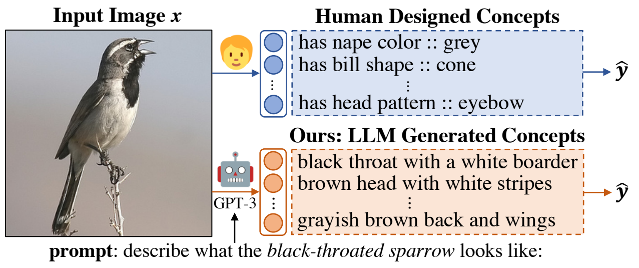

Input Image x
Human Designed Concepts
has nape color :: grey
has bill shape :: cone
has head pattern :: eyebow
$\hat{y}$
Ours: LLM Generated Concepts
black throat with a white boarder
brown head with white stripes
grayish brown back and wings
$\hat{y}$
GPT-3
prompt: describe what the black-throated sparrow looks like:

Figure 1. Our proposed high-performance Concept Bottleneck Model alleviates the need for human designed concepts by prompting large language models (LLMs) such as GPT-3 [4].

believed that such models will perform more poorly than their black box alternatives [15]. In this work, we provide evidence to the contrary. We show how to construct high-performance interpretable-by-design classifiers by combining a language model, GPT-3 [4], and a language-vision model, CLIP [42].

Our method builds on Concept Bottleneck Models (CBM) [24], which construct predictors through a linear combination of human-designed concepts. For example, as seen in Figure 1, a qualified person can design concepts, such as "nape color," as intermediate targets for a black box model before classifying a bird. CBMs provide abstractions that people can use to understand errors or intervene on, contributing to increased trust.

Application of CBMs is limited because they require costly attribute annotations by domain experts and often under-perform their black box counterparts. In contexts where CBM performance is competitive with black box alternatives, interpretability properties are sacrificed [33, 66]. To address both of these challenges, we propose to build systems that automatically construct CBMs.

Our **La**nguage Model Guided Concept **Bo**ttleneck Model (**LaBo**), Figure 2, allows for the automatic construction of high-performance CBMs for arbitrary classification problems without concept annotations. Large language models (LLMs) contain significant world knowledge [20, 40, 57],

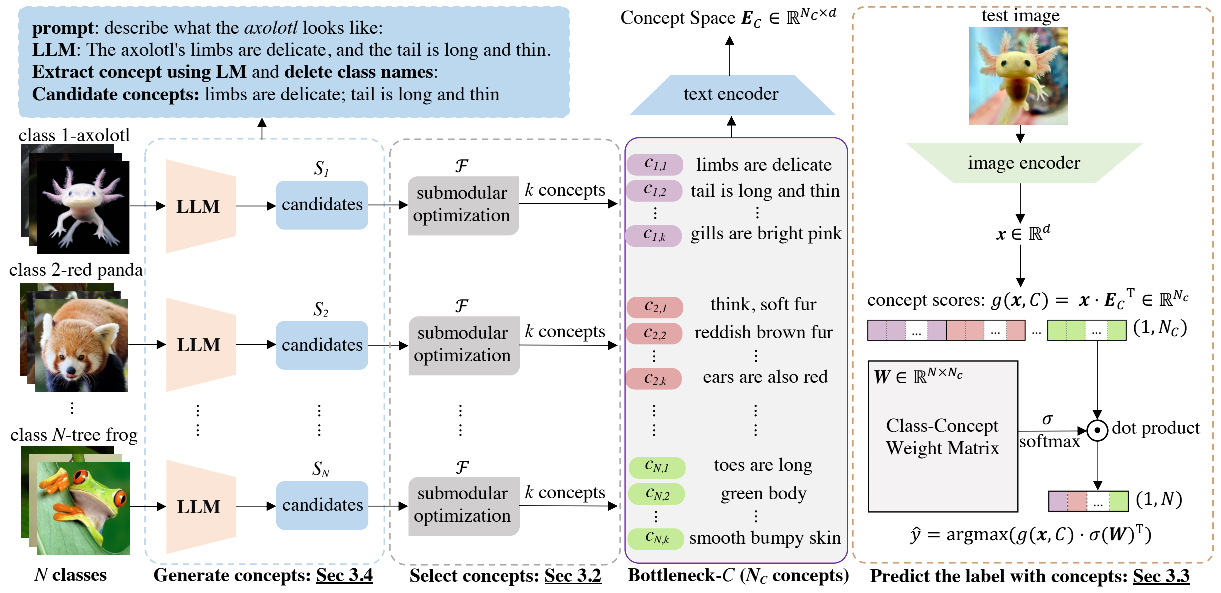

prompt: describe what the axolotl looks like:
LLM: The axolotl's limbs are delicate, and the tail is long and thin.
Extract concept using LM and delete class names:
Candidate concepts: limbs are delicate; tail is long and thin
class 1-axolotl
class 2-red panda
class N-tree frog
N classes
LLM
candidates
S1
S2
SN
F
submodular optimization
k concepts
Generate concepts: Sec 3.4
Select concepts: Sec 3.2
Concept Space $E_C \in \mathbb{R}^{N_C \times d}$
text encoder
limbs are delicate
tail is long and thin
gills are bright pink
think, soft fur
reddish brown fur
ears are also red
toes are long
green body
smooth bumpy skin
Bottleneck-C (N_C concepts)
test image
image encoder
$x \in \mathbb{R}^d$
concept scores: $g(x, C) = x \cdot E_C^{\mathsf{T}} \in \mathbb{R}^{N_c}$
$(1, N_C)$
$W \in \mathbb{R}^{N \times N_c}$
Class-Concept Weight Matrix
σ softmax
dot product
$(1, N)$
$\hat{y} = \text{argmax}(g(x, C) \cdot \sigma(W)^{\mathsf{T}})$
Predict the label with concepts: Sec 3.3

Figure 2. We present an overview of our **La**nguage-Model-Guided Concept **Bo**ttleneck Model (**LaBo**), which is an interpretable by design image classification system. First, we prompt the large language model (GPT-3) to generate candidate concepts (Sec 3.4). Second, we employ a submodular function to select concepts from all candidates to construct the bottleneck (Sec 3.2). Third, we apply a pretrained alignment model (CLIP) to obtain the embeddings of concepts and images, which is used to compute concept scores. Finally, we train a linear function in which the weight $\boldsymbol{W}$ denotes the concept-class association user to predict targets based on concept scores (Sec 3.3).

that can be elicited by inputting a string prefix and allowing LLMs to complete the string (prompting). For example, in Figure 1, GPT-3 is prompted about sparrows and completes with information such as "brown head with white stripes." LaBo leverages this by constructing bottlenecks where the concepts are such GPT-3 generated sentences. Since our concepts are sentences, we use CLIP to score their presence in an image and form a bottleneck layer out of these scores.

A key advantage of LaBo is the ability to control the selection of concepts in the bottleneck by generating candidates from the language model. We develop selection principles targeting both interpretability and classification accuracy. For example, we prefer smaller bottlenecks that include shorter sentences that do not include class names. Furthermore, to maximize performance, we prefer attributes that CLIP can easily recognize and are highly discriminative. To account for appearance variation, we select attributes that cover a variety of information and are not repetitive. We formulate these factors into a novel sub-modular criterion that allows us to select good bottlenecks efficiently [36].

We have evaluated LaBo-created bottlenecks on 11 diverse image classification tasks, spanning recognition of common objects [10, 25] to skin tumors [60]. fine-grained types [3, 31, 37, 63], textures [9], actions [55], skin tumors [60], and satellite photographed objects [7].[1] Our main finding is that LaBo is a highly effective prior for what concepts to look for, especially in low data regimes. In evaluations comparing with linear probes, LaBo outperforms by as much as 11.7% at 1-shot and marginally underperforms given larger data settings. Averaged over many dataset sizes, LaBo bottlenecks are 1.5% more accurate than linear probes. In comparison to modifications of CBMs that improve performance by circumventing the bottleneck [66], we achieve similar or better results without breaking the CBM abstraction. In extensive ablations, we study key trade-offs in bottleneck design and show our selection criteria are crucial, and highlight several other critical design choices.

Human evaluations indicate that our bottlenecks are largely understandable, visual, and factual. Finally, annotators find our GPT-3 sourced bottlenecks are more factual and groundable than those constructed from WordNet or Wikipedia sentences. Overall, our experiments demonstrate that automatically designed CBMs can be as effective as black box models while maintaining critical factors contributing to their interpretability.

2. Related Work

Broadly, interpretability methods fall into two categories: *post-hoc* and *by design*. While ours is an instance of the latter, **post-hoc methods** have the advantage of not imposing any model constraints. For example, *Gradient-weighted Class Activation Mapping* approaches [2, 18, 34, 51] trace network

[1]The only dataset specialization we perform is prompt tuning for GPT-3 when creating candidate attributes. This is largely done to overcome problems of word sense. For example, when naively prompted to produce knowledge about the flower "bird of paradise" GPT-3 yields information about birds instead of flowers. In general, specialization here was also minimal. See appendix for prompts.

gradients to identify the input areas that guide predictions. Similarly, *Explanation Generation* methods [16, 22, 38, 53] require models to produce explanations for visual tasks by conditioning their predictions on captioning models and [17, 39] incorporate visual evidence to ground explanations.

Despite their advantages, there is no guarantee that post-hoc methods faithfully represent model reasoning [46]. In contrast, our work falls under **interpretable by design methods**, which constrain explanations to align with the model's reasoning. For example, *Prototype* methods [5, 35, 48, 54, 62] optimize a metric space that guides classification by computing distances to prototype representations of each class. While such methods identify important regions in the input for classification, they still require featurized region representations that obfuscate the semantic content of the region.

This work extends another family of interpretable by design methods known as *Concept Bottleneck Models* [24, 49]. Following early attempts in few shot learning [26] and attribute learning [47, 64], CBMs predict targets by linearly combining an intermediate layer of human-understandable attributes. Recently, Computational Derivation Learning (CompDL) [67] proposed a CBM architecture that applies a linear layer over CLIP scores between human expert designed concepts and images to predict targets in the context of an evaluation framework to measure how well CLIP grounds concepts. CBMs generally suffer from the need for costly class description annotations and lower performance compared to end-to-end counterparts. Post-hoc Concept Bottleneck (PCBM) [66] was proposed to fill these two gaps by leveraging information from a static knowledge base, such as ConceptNet [56], and adding a residual connection from image features to the final prediction to improve accuracy [66]. However, PCBMs cannot be expanded to larger-scale (e.g., ImageNet [10]) or domain-specific tasks (e.g., fine-grained [31]) because knowledge bases have limited coverage. In addition, they include a residual predictor, which effectively ensembles CBM with an end-to-end model, undermining interpretability.

We circumvent the need for external knowledge bases, which are often incomplete, and instead query LLMs to automate concept collection. We remove the need for direct mapping from image features to targets by fully automating the extraction and filtering of LLM knowledge, surpassing end-to-end models in few shot settings and achieving comparable performance in large data settings.

Our work capitalizes on improvements in **vision-language pretraining** from earlier BERT-based models [6, 29, 30, 58] to more scalable contrastive architectures [19, 27, 42, 65], which are very effective for few shot image classification [8, 59].

Our work can be viewed as interpretability-focused prompt tuning of CLIP [42]. Significant efforts have been devoted to **prompting** vision language models [11, 13, 28, 41, 44, 69, 70]. These focus on searching over text prompts to improve classification performance, and resemble earlier techniques in LLM prompt tuning [14, 50, 52].

3. Method

Figure 2 presents an overview of our method. Our model prompts a large language model, GPT-3 [4] to generate a set of candidate concepts for each class (Section 3.4). We employ submodular optimization to greedily select a subset of concepts for each class such that we maximize discriminability and diversity (Section 3.2). We then align the selected concepts to images using CLIP [42]. We apply a linear layer over the similarity scores of concepts and images to learn a weight matrix representing the importance of each concept in the final classification. This weight matrix is initialized using a language model prior from GPT-3 (Section 3.3).

3.1. Problem Formulation

Consider a training set of image-label pairs $\mathcal{D} = \{(i, y)\}$ where i is the image and $y \in \mathcal{Y}$, is a label from a set of N classes. Suppose we have a pretrained multimodal alignment model (e.g., CLIP [42]), which has an image encoder $\mathcal{I}$ and a text encoder $\mathcal{T}$. $\mathcal{I}$ and $\mathcal{T}$ can map images and text into the shared feature space, respectively. The dot product of the image and text features reflects the alignment score between the two modalities. We extract the features of all images in $\mathcal{D}$ as $\boldsymbol{x} = \mathcal{I}(i) \in \mathbb{R}^d$, and the dataset can be represented as $\mathcal{D} = \{(\boldsymbol{x}, y)\}$. Let S be the superset of candidate textual concepts generated from language models. We use a submodular function $\mathcal{F}$ to select a bottleneck, C, where $C \subseteq S$, made of N_C concepts, $C = \{c_1, c_2, ..., c_{N_C}\}$. We can construct a bottleneck embedding, $\boldsymbol{E}_C \in \mathbb{R}^{N_C \times d}$, and each row of $\boldsymbol{E}_C$ is the text feature $\mathcal{T}(c) \in \mathbb{R}^d$ of a concept c extracted by the text encoder $\mathcal{T}$.

Concept bottleneck models produce a prediction by composing two functions, $\hat{y} = f\left(g\left(\boldsymbol{x}, \boldsymbol{E}_C\right)\right)$, in which $g : \mathbb{R}^d \to \mathbb{R}^{N_C}$ maps the image feature to a score for every element of the bottleneck and $f : \mathbb{R}^{N_C} \to \mathcal{Y}$ makes the final prediction on the label space given the concept scores. In our setting, we find a bottleneck C and appropriate f by solving the following minimization problem:

$$\min_{f,C} \mathop{\mathbb{E}}_{(\boldsymbol{x},y)\sim\mathcal{D}} \left[\mathcal{L}\left(f\left(g\left(\boldsymbol{x}, \boldsymbol{E}_C\right)\right), y\right)\right] - \mathcal{F}\left(C, \mathcal{D}\right) \tag{1}$$

in which $\mathcal{L}(\hat{y}, y)$ is the cross-entropy loss on the label prediction and $\mathcal{F}(C, \mathcal{D})$ is the quality of the bottleneck as measured by the submodular function. In practice, we optimize sequentially: we first find a high scoring C under $\mathcal{F}$. Then, we use the dot product of image and concept embeddings as g. Finally, we find an f that minimizes $\mathcal{L}$. In the following sections, we will illustrate how we: construct the submodular function $\mathcal{F}$ to select a subset of concepts C from the candidates S (Section 3.2) and learn f (Section 3.3).

3.2. Submodular Concept Selection

We create a superset of candidate concepts, S, out of class-specific subsets. For every label $y \in \mathcal{Y}$, we construct S_y by prompting a language model to produce textual knowledge about y (Section 3.4). Instead of directly choosing N_C concepts from S, we select k concepts for each class, such that $N \times k = N_C$, to ensure each class has an equal number of relevant concepts in the bottleneck.

We employ submodular optimization [1] to select a subset $C_y \subseteq S_y$, $|C_y| = k$. Specifically, we need to design a score function $\mathcal{F} : 2^{|S_y|} \to \mathbb{R}$ to evaluate the utility of the subset. Submodular functions should satisfy the *diminishing returns* property.[2] If a submodular function is *monotone*,[3] a greedy algorithm [36] can be used to find a solution within a constant factor of the optimal one. We propose the following monotone submodular function[4] to select the subset C_y from the candidate set S_y:

$$\mathcal{F}(C_y) = \underbrace{\alpha \cdot \sum_{c \in C_y} D(c)}_{\text{dicriminability}} + \underbrace{\beta \cdot \sum_{c_1 \in S_y} \max_{c_2 \in C_y} \phi(c_1, c_2)}_{\text{coverage}}, \quad (2)$$

where $D(c)$ denotes the discriminability score of the concept c and $\phi(\cdot)$ is the intra-concept similarity. Generally, the first term tends to select more informative concepts, and the second term ensures the subset has good coverage of the candidate set. The hyperparameters α and β control the weights of the two sub-functions. Here we present how to compute these two scores:

Discriminability Score. We introduce a discriminability score to encourage the selection of concepts that are aligned with many images in class y, but few images in other classes.We first define the similarity score $Sim\,(y, c)$ between a class and concept by taking the mean of the dot product between the images and text features:

$$Sim\,(y, c) = \frac{1}{|\mathcal{X}_y|} \sum_{\boldsymbol{x} \in \mathcal{X}_y} \boldsymbol{x} \cdot \mathcal{T}(c)^\top, \quad (3)$$

where $\mathcal{X}_y$ is the set of training images labeled with y[5] and $\mathcal{T}$ is the text encoder. We define the normalized class association, which measures the conditional likelihood of aligning featurized images of a class given a concept's textual embedding, $\overline{Sim}(y|c) = Sim(y, c)/\sum_{y' \in \mathcal{Y}} Sim(y', c)$, and compute its negative entropy:

$$D(c) = \sum_{y' \in Y} \overline{Sim}\,(y'|c) \cdot \log\left(\overline{Sim}\,(y'|c)\right) \quad (4)$$

Maximizing $D(c)$ will result in the selection of concepts that have peaked $\overline{Sim}(y|c)$, indicating that a concept is strongly associated with only a few classes.

Coverage Score. The second term of equation 2 is a minimax facility location function that tries to minimize the maximum distance between each element in the subset and the candidate set. For distance, we use the cosine between the features of the two concepts extracted by the text encoder: $\phi(c_1, c_2) = \cos\left(\mathcal{T}(c_1), \mathcal{T}(c_2)\right)$. A high coverage score yields a diverse bottleneck that covers different possible appearances for a target class.

3.3. Optimize Class-concept Association

In this section, we explain how we compute g (the concept predictor) and learn f (the label predictor) of the bottleneck.

Predict the Concept Scores. The concept predictor g is not learned in our method because the alignment model we use can measure the correlation between image and text through dot product. We treat the dot product of input image feature $\boldsymbol{x}$ and the concept space $\boldsymbol{E}_C$ defined in Section 3.2 as g: $g\left(\boldsymbol{x}, \boldsymbol{E}_C\right) = \boldsymbol{x} \cdot \boldsymbol{E}_C^\top$, where $g\left(\boldsymbol{x}, \boldsymbol{E}_C\right) \in \mathbb{R}^{N_C}$, and each element is the score of image $\boldsymbol{x}$ on a concept.

Concept Weight Matrix. We learn a linear function for the label predictor f that maps from concept scores to the final prediction. Intuitively, these weights encode the affinity of the concept to the class, allowing the model to represent that classes depend differently on the same concept. To normalize the class-concept association distributed over the weight matrix, we regularize the matrix with the softmax activation function. Concretely, we learn a concept weight matrix $\boldsymbol{W} \in \mathbb{R}^{N \times N_C}$, that is used for prediction: $\hat{y} = \text{argmax}\left(g\left(\boldsymbol{x}, \boldsymbol{E}_C\right) \cdot \sigma\left(\boldsymbol{W}\right)^\top\right)$, in which $\sigma(\cdot)$ is the softmax activation which is applied along the concepts axis: $\boldsymbol{W}_{y,c} = e^{\boldsymbol{W}_{y,c}} / \sum_{y' \in \mathcal{Y}} e^{\boldsymbol{W}_{y',c}}$.

Initializing the Weight Matrix with Language Priors. Previous work trains the concept weight matrix freely from scratch, which is not feasible in low-resource scenarios where we don't have enough data to learn the weight effectively. To extend the application of CBM to few-shot image classification, we consider biasing the weights toward the initial association from the language model used to propose concepts. If a concept c was present in C_y, we initialize the elements of $\boldsymbol{W}$ corresponding to the weight between class y and concept c to a higher value before optimization: $\boldsymbol{W}_{y,c} = 1$, if $c \in C_y$, otherwise 0.

3.4. Prepare the Candidates

To collect the candidates S to feed into our model, we prompt GPT-3 to generate relevant sentences by incorporating the class name in 5 templates shown in supplementary

[2] *diminishing returns* property means $\forall A \subseteq B \subseteq V \setminus v$, we have $\mathcal{F}(A + \{v\}) - \mathcal{F}(A) \geq \mathcal{F}(B + \{v\}) - \mathcal{F}(B)$.

[3] A submodular function is *monotone* if $\forall A \subseteq B, \mathcal{F}(A) \leq \mathcal{F}(B)$.

[4] Any linear combination of submodular functions is still submodular.

[5] In N-way-K-shot setting, $|\mathcal{X}_y| = K$.

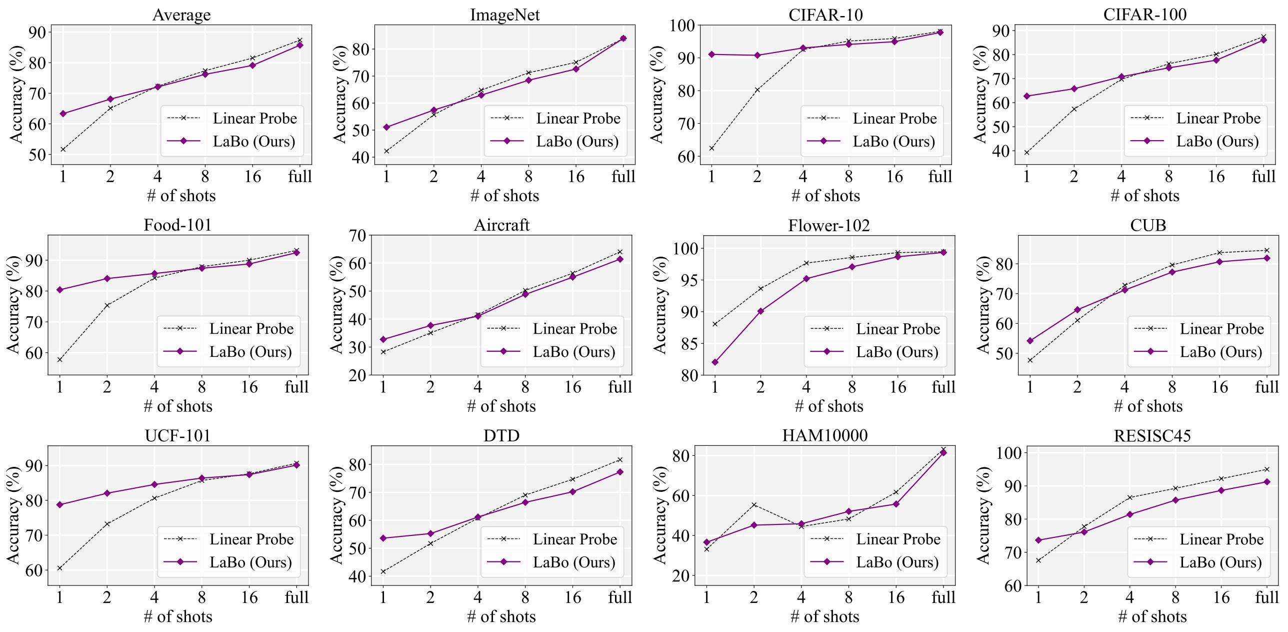


Figure 3. Test accuracy (%) comparison between LaBo and Linear Probe on 11 datasets. The x-axis represents the number of labeled images.

materials.[6] For example, as shown in the top-left of Figure 2, we prompt GPT-3 by asking "*describe what the **axolotl** looks like*", and the GPT-3 returns a sentence about the target class. We obtain 500 sentences for each class and automatically split these sentences into shorter concepts using a T5 model [43] fine-tuned on a small set of annotated sentence-concept pairs. We use string match to identify and remove class name tokens in each concept. (see supplementary)

4. Experimental Setup

We evaluate our method on a diverse set of 11 datasets (Section 4.1) and compare it to its end-to-end counterpart and other interpretable CBM methods (Section 4.2).

4.1. Dataset

We select a comprehensive benchmark of 11 image classification datasets spanning a diverse set of domains, including (1) Common objects: ImageNet [10], CIFAR-10 and CIFAR-100 [25]; (2) Fine-grained objects: Food-101 [3], FGVC-Aircraft [31], Flower-102 [37], CUB-200-2011 [63]; (3) Actions: UCF-101 [55]; (4) Textures: DTD [9]; (5) Skin tumors: HAM10000 [60] and (6) Satellite images: RESISC45 [7]. We use train/dev/test splits for all the datasets. Detailed statistics are presented in the supplementary material. We follow the few-shot evaluation protocol proposed by CLIP [42] with 1, 2, 4, 8, and 16 images randomly sampled from the training set for each class. We also evaluate in the fully-supervised setting where we train on all available images. For all experiments, we report the test accuracy.

4.2. Baselines

We compare our model, LaBo, with black-box linear probing and two interpretable methods.

Linear Probe Following previous evaluations on CBM [24, 66], linear probing serves as our primary baseline for comparison. We follow the implementation of CLIP [42] by training the scikit-learn's L-BFGS logistic regression with a hyperparameter sweep on the L2 regularization weight.

PCBM Post-hoc Concept Bottleneck Model [66] designs a residual modeling step that directly maps the original image embedding into the label space. PCBM treats the attributes of each class in ConceptNet [56] as concepts.

CompDL Compositional Derivation Learning [67] learns a linear layer over CLIP similarity scores between human-designed class descriptions and images to predict targets.

4.3. Implementation Details

We prompt GPT-3-text-davinci-002 to generate concepts. The CLIP model is adapted from OpenAI's public repo with ViT-L/14 as the default vision backbone. We only use CLIP-RN50 as the backbone when comparing with PCBM and ViiT-B/32 with CompDL for fair comparison. We implement the submodular function using the apricot package and set the default number of concepts selected for each class to 50. To train the linear function, we use the Pytorch-lightning library with Adam [23] optimizer. We tune the batch size,

[6]We use the same set of prompts for all datasets except UCF-101 since it is very different to describe an action.

Method	1	2	4	8	16	Full	Avg
Linear Probe	51.69	65.13	**72.33**	**77.38**	**81.53**	**87.38**	72.57
LaBo (Ours)	**63.35**	**68.10**	72.08	76.19	79.11	85.72	**74.09**

Table 1. Mean accuracy across all datasets, at different shots .

Method	w/ end-to-end	CIFAR-10	CIFAR-100
PCBM [66]	✗	84.5	56.0
LaBo (Ours)	✗	**87.9**	**69.1**
PCBM-h [66]	✓	87.6	69.9
Linear Probe	✓	88.8	70.1

Table 2. Test accuracy comparison between LaBo and Post-hoc Concept Bottleneck Model (PCBM) on CIFAR-10 and CIFAR-100. “w/ end-to-end” denotes whether the model employs an end-to-end residual predictor from image features to targets.

Method	w/ manual concepts	1	5	Full
CompDL [67]	✓	13.6	33.2	52.6
LaBo (Ours)	✗	**35.1**	**55.7**	**71.8**
Linear Probe	-	28.4	55.4	75.5

Table 3. LaBo and CompDL evaluated on CUB for 1/5/full shots.

n. of concepts	n. of shots					
per class (k)	1	2	4	8	16	Full
1	41.89	52.45	61.76	65.99	69.61	78.95
5	52.54	61.13	67.22	72.90	75.62	83.83
10	58.00	64.59	69.90	74.50	77.43	84.66
25	61.72	66.33	71.39	75.28	79.04	85.26
50	**63.03**	**67.79**	**71.88**	**76.08**	**79.10**	**85.71**

Table 4. Ablation results on bottleneck sizes. We vary the sizes of the bottlenecks and report the average performance on 11 datasets.

Selection	n. of shots					
Method	1	2	4	8	16	Full
RANDOM	59.24	64.71	70.42	74.07	78.29	85.06
SIMILARITY	54.59	61.42	67.17	72.66	77.32	84.88
COVERAGE	59.73	65.93	70.82	74.71	78.90	85.60
DISCRIM	60.99	66.49	70.93	74.81	77.90	85.31
SUBMODULAR	**63.03**	**67.79**	**71.88**	**76.08**	**79.10**	**85.71**

Table 5. Ablation results on concept selection methods. We report mean test accuracy on 11 datasets.

learning rate, and submodular weights on the development set. Model checkpoints with the highest validation accuracy are evaluated on the test set. We list the hyperparameters for all datasets and shots in the supplementary material.

5. Evaluation

5.1. Main Results

We compare LaBo’s performance with a linear probe and other interpretable baselines to evaluate if we can maintain black box accuracy without sacrificing interpretability.

Comparison with End-to-End Model. One of our goals is to close the performance gap between interpretable and black box models. Table 1 reports the mean test accuracy of LaBo and the linear probe on 11 datasets. LaBo significantly outperforms the end-to-end model when little data is available and continues to be competitive as the number of data increases. On average, LaBo surpasses the linear probe by 1.5%. Figure 3 provides analytic performance comparisons between LaBo and Linear Probe on each dataset.

In general, LaBo’s performance depends on the quality of knowledge extracted from GPT-3. For common categories, GPT-3 contains high-quality knowledge allowing substantial improvement over linear probes. For some fine-grained datasets, such as Flower-102, GPT-3’s knowledge is largely non-visual, as seen in Figure 7. In such cases, specialized language models could be used to improve LaBo.

Comparison with other Interpretable Methods. Table 2 compares LaBo’s performance with PCBM and Linear Probe. LaBo outperforms PCBM by 3.4% on CIFAR-10 and 13.1% on CIFAR-100. LaBo maintains comparable performance to PCBM with a residual predictor (PCBM-h), without circumventing the bottleneck. In Table 3, LaBo is more accurate than CompDL [67] without manually constructed concepts.

5.2. Ablation Study

We evaluate the importance of each of our model’s components on final performance. Specifically, we compare results with different concept selection methods, language and random weight initialization, and bottleneck sizes.

Concept Selection Methods. We compare our submodular function with four concept selection methods: (1) RANDOM: we randomly sample a subset of concepts from the candidates for each class; (2) SIMILARITY: we select the top concepts ranked by their similarity scores with the class calculated by equation 3; (3) COVERAGE: we only consider the coverage score for concept selection; (4) DISCRIM: we only consider the discriminability score for concept selection. As shown in Table 5, our submodular function, which jointly optimizes coverage and discriminability, achieves the best performance across different numbers of shots. We notice that using coverage or discriminability alone still outperforms using similarity between the class and random selection. Selection method plays an important role in all data settings, but its impact decreases with more supervision.

Initialization with Language Priors. We deactivate the LM initialization and use random initialization instead. Figure 4 shows that the LM prior is more important for low shot settings since there is less signal to guide concept importance.

Bottleneck Size. In Table 4, we compare performance for different bottleneck sizes ranging from 1 to 50 concepts selected by the submodular function. Larger bottlenecks are usually better, but with more data, similar performance is achievable with smaller bottlenecks.

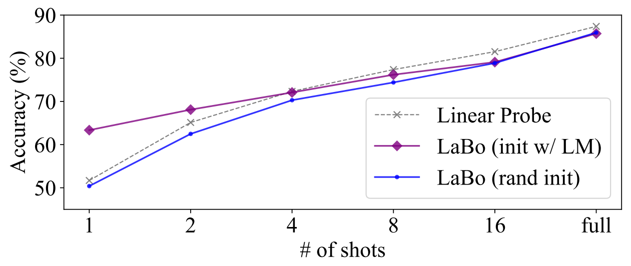


Figure 4. Language Prior vs. Random Weight initialization average over all datasets.

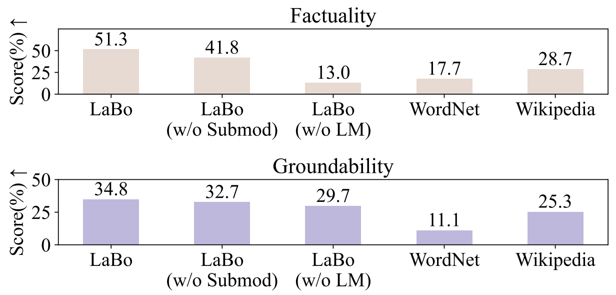


Figure 5. Human evaluation on *Factuality* and *Groundability* for different bottlenecks on ImageNet. "w/o Submod" denotes without submodular function, i.e., random concept selection. "w/o LM" denotes no language prior weight initialization.

5.3. Human Evaluation

It is important for interpretability that the vision-language alignment model correctly grounds concepts to images. For example, if a concept "usually round" ranks both circles and stripes highly, the name of the attribute does not faithfully represent the computation. In addition, it is important that the automatically generated concept bottlenecks factually correspond to the class they describe. To this end, we introduce two metrics to evaluate the quality of our concept bottleneck items: (1) *Factuality* measures how accurate the concepts are in describing their designated class by requiring annotators to judge whether they describe ground truth images, and (2) *Groundability* measures how consistent the vision-language model grounding of the concepts to images with human interpretations by requiring annotators to judge their applicability on the top-10 images ranked by CLIP alignment scores.

Setup. Both metrics are computed by asking annotators to select images that describe a highly ranked concept in our bottlenecks. Formally, the two metrics are represented by:

$$Factuality(c) = \frac{\text{number of images selected}}{k \text{ ground truth images of the class}}$$

$$Groundability(c) = \frac{\text{number of images selected}}{\text{top-}k \text{ aligned images of the concept}}$$

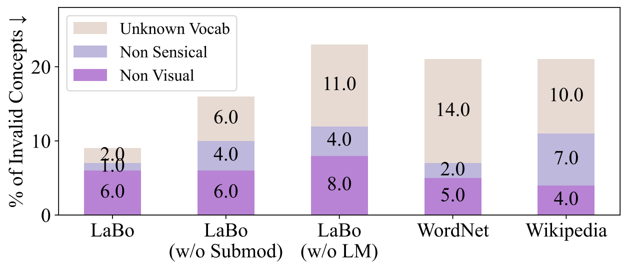


Figure 6. Percentage of invalid concepts identified by humans for different bottlenecks on ImageNet. **Lower** percentage is better.

Metrics	LaBo	w/o Submod	w/o LM
Factuality (%) ↑	**24.0**	22.8	14.1
Groundability (%) ↑	14.1	**22.5**	20.2
Non Visual (%) ↓	**4.8**	5.6	5.8
Non Sensical (%) ↓	**8.0**	9.6	8.7
Unknown Vocab (%) ↓	**10.2**	10.5	10.7

Table 6. Average human evaluation results of LaBo on 11 datasets. We also evaluate LaBo by removing the submodular function (w/o Submod) and language model priors (w/o LM).

where we set $k = 10$.[7] In addition to the two main metrics, we ask the annotator to select whether the concept is non-visual, nonsensical, or contains unknown vocabulary. We randomly sample 20 classes for each dataset and evaluate the top 5 concepts (ranked by the weights of the linear function) for each class, 100 concepts per dataset. We release our human evaluation task on Amazon Mechanical Turk and collect three annotations for each concept. More details on the task and the results can be found in the supplement.

Baselines. We evaluate the bottlenecks under full supervision and compare them with two main baselines: (1) LaBo (w/o Submod), which randomly selects the concepts instead of using the submodular function, and (2) LaBo (w/o LM), which initializes the concept weight matrix randomly without leveraging the priors of the language model. For ImageNet, we add two additional baselines using human-written text: (1) WordNet [12] definitions and (2) Wikipedia sentences [21]. We adopt the same preprocessing pipeline as LaBo to extract concepts from human-written resources and utilize the submodular function to select the bottlenecks.

Results. Figure 5 shows the evaluation on ImageNet, and we observe that LaBo has significantly higher *Factuality* and *Groundability* than human-written text. We further observe that removing components from our system (submodular and LM Prior) hurt both human evaluation metrics, indicating their collective importance in our system. In addition, Figure 6 shows that LaBo has significantly fewer invalid concepts

[7]With the only exception of *Factuality* for Flower-102 where we set $k = 8$, i.e., the minimum number of images for a specific class in the development set.

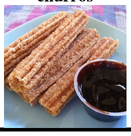
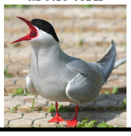
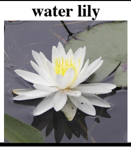
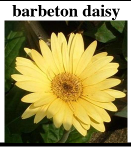
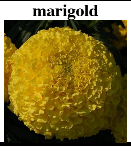
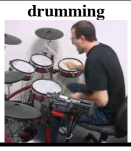
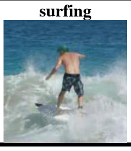
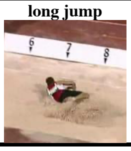
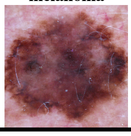
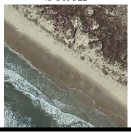
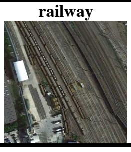
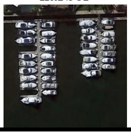
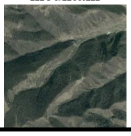
	Class Name	Top-3 Concepts	Class Name	Top-3 Concepts	Class Name	Top-3 Concepts	Class Name	Top-3 Concepts
ImageNet	**badger**	1.short legs and long body make it an excellent digger 2. black-and-white striped fur 3. coat is very shaggy	**ant**	1. black and red stinger 2. small, black insect with six legs 3. long, slender antennae that it uses to smell and touch	**hammer**	1.long, thin tool with a wooden handle 2. great tool for pounding object 3. used to pound on surfaces	**water buffalo**	1. large head with short, curved horns 2. heaviest living species of bovid 3. huge, dark-colored animal
Food101	**ramen**	1. garnished with green onions, nori, and other toppings 2. most grocery stores 3. various toppings	**hummus**	1. chickpeas, tahini, olive oil, garlic, lemon juice 2. made from cooked, mashed chickpeas 3. roasted red peppers	**beef tartar**	1. center of the tartare is still pink 2. small, round, flat cake of minced beef 3. stunning, vibrant red color	**churros** 	1. rolled in a cinnamon sugar mixture 2. origin in spain 3. spiraling outwards
CUB	**eared grebe**	1. black and white plumage that is striking in the sunlight 2. black body with a long, slender neck 3. red and black bill	**horned lark**	1. black line running through yellow face 2. head is black with a white horn on each side 3. black horn on each side of their head	**white pelican**	1. long neck and bill make it look like a giant swan 2. large, white bird with black wingtips 3. bill is huge and yellow	**arctic tern** 	1. breeds in greenland, iceland, and northern russia 2. black and white markings 3. small white bird with a black cap
Flower	**water lily** 	1. depicted in artworks of ponds and waterfall 2. member of the nymphaeaceae family 3. lily pads float	**barbeton daisy** 	1. scientific name for the flower is taraxacum officinal 2. named after the city of barberton 3. member of the daisy family	**marigold** 	1. central disc with smaller florets 2. have a slightly furry texture 3. bold and vibrant color palette	**tiger lily**	1. long, protruding stamen 2. orange with black spots and stripes 3. scientific name is lilium columbianum
UCF-101	**archery**	1. grip bow tightly in their left hand 2. focused and concentrated on their task 3. keep bow and arrows in safe and dry place when not in use	**drumming** 	1. blur as they fly over the drums 2. sitting on a stool in front of a drum set 3. position the drumstick so it is resting on your index finger	**surfing** 	1.deep blue color 2. tans contrast with the white of their boards 3. sending a spray of water into the air	**long jump** 	1. get out of sandpit 2. series of movements starting from a standing position 3. person tucks their knees up to chest
HAM100000	**dermatofibroma**	1. generally not painful 2. red, brown, or purple in color 3. thin white halo around them	**melanoma** 	1. dark brown or black in color 2. large and dark 3. flesh-colored, brown, or black	**melanocytic nevi**	1. color is tan 2. dark brown or black color 3. small, round, and slightly raised	**benign lesions**	1. color ranges from light brown to black 2. rough or scaly texture 3. darker in color, such as brown or black
RESISC45	**beach** 	1. waves crashing onto the shore 2. few rocks poking out 3. waves are gentle	**railway** 	1. connected by steel rails 2. tramline that is 3 feet wide and runs along the length of the court 3. faint, twinkling line	**harbor** 	1. boats of all colors moored in the scene 2. boats of all sizes 3. well-lit and well-marked	**mountain** 	1. sides are covered in trees 2. three main peaks 3. trees and vegetation on its slopes

Figure 7. Several example bottlenecks generated by LaBo. The top-3 concepts, ranked by their weights in the linear function, for randomly selected classes, paired with a random image from the class, across 6 datasets.

than other baselines. Table 6 summarizes the average human evaluation results over the 11 datasets[8]. On average, we observe a trade-off between *Factuality* and *Groundability*. Increasing coverage and discriminability leads to more variable and specific concepts that CLIP finds more difficult to ground. This could be due to challenges in capturing composite concepts [32, 67]. For individual analysis of the datasets, refer to the supplementary material. Finally, Figure 7 shows several CBMs we constructed. Across many types of tasks, the bottlenecks are largely coherent, factual, and groundable by CLIP.

[8]The low resolution of CIFAR images partially affects those metrics since annotators have greater difficulty in completing the task (see supplementary material)

6. Conclusion and Limitation

Overall, LaBo demonstrates that accuracy and interpretability of vision systems may be less at odds than previously believed. Leveraging LLMs was crucial, as they encode important visual knowledge. In the future, our approach can easily be enriched with new factors that capture different priors on bottleneck construction. The limits of knowledge in GPT-3 are not known, but likely there are domains where prompting generates few useful facts. Even in contexts where GPT-3 can generate useful information, our method depends on CLIP being able to recognize those aspects in images. Future work could focus on dynamically prompting GPT-3 to make this coupling more robust.

References

[1] Francis Bach. Convex analysis and optimization with submodular functions: a tutorial. *arXiv preprint arXiv:1010.4207*, 2010. 4

[2] 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*, pages 6541–6549, 2017. 2

[3] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014. 2, 5

[4] 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 H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. 1, 3

[5] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32, 2019. 3

[6] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, 2020. 3

[7] Gong Cheng, Junwei Han, and Xiaoqiang Lu. Remote sensing image scene classification: Benchmark and state of the art. *Proceedings of the IEEE*, 105(10):1865–1883, 2017. 2, 5

[8] Arkabandhu Chowdhury, Mingchao Jiang, Swarat Chaudhuri, and Chris Jermaine. Few-shot image classification: Just use a library of pre-trained feature extractors and a simple classifier. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9445–9454, 2021. 3

[9] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014. 2, 5

[10] 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*, pages 248–255. Ieee, 2009. 2, 3, 5

[11] Sinuo Deng, Lifang Wu, Ge Shi, Lehao Xing, and Meng Jian. Learning to compose diversified prompts for image emotion classification. *arXiv preprint arXiv:2201.10963*, 2022. 3

[12] Christiane Fellbaum. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer, 2010. 7

[13] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. Clip-adapter: Better vision-language models with feature adapters. *arXiv preprint arXiv:2110.04544*, 2021. 3

[14] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *ACL/IJCNLP (1)*, 2021. 3

[15] David Gunning and David Aha. Darpa's explainable artificial intelligence (xai) program. *AI magazine*, 40(2):44–58, 2019. 1

[16] Lisa Anne Hendricks, Zeynep Akata, Marcus Rohrbach, Jeff Donahue, Bernt Schiele, and Trevor Darrell. Generating visual explanations. In *European conference on computer vision*, pages 3–19. Springer, 2016. 3

[17] Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. Grounding visual explanations. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 264–279, 2018. 3

[18] Evan Hernandez, Sarah Schwettmann, David Bau, Teona Bagashvili, Antonio Torralba, and Jacob Andreas. Natural language descriptions of deep visual features. In *International Conference on Learning Representations*, 2021. 2

[19] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021. 3

[20] Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020. 1

[21] Jihyung Kil and Wei-Lun Chao. Revisiting document representations for large-scale zero-shot learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3117–3128, Online, June 2021. Association for Computational Linguistics. 7

[22] Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John Canny, and Zeynep Akata. Textual explanations for self-driving vehicles. In *Proceedings of the European conference on computer vision (ECCV)*, pages 563–578, 2018. 3

[23] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[24] 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*, pages 5338–5348. PMLR, 2020. 1, 3, 5

[25] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 2, 5

[26] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE transactions on pattern analysis and machine intelligence*, 36(3):453–465, 2013. 3

[27] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified

vision-language understanding and generation. *arXiv preprint arXiv:2201.12086*, 2022. 3

[28] Jiangmeng Li, Wenyi Mo, Wenwen Qiang, Bing Su, and Changwen Zheng. Supporting vision-language model inference with causality-pruning knowledge prompt. *arXiv preprint arXiv:2205.11100*, 2022. 3

[29] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019. 3

[30] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019. 3

[31] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 2, 3, 5

[32] Massimiliano Mancini, Muhammad Ferjad Naeem, Yongqin Xian, and Zeynep Akata. Open world compositional zero-shot learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5222–5230, 2021. 8

[33] Andrei Margeloiu, Matthew Ashman, Umang Bhatt, Yanzhi Chen, Mateja Jamnik, and Adrian Weller. Do concept bottleneck models learn as intended? *arXiv preprint arXiv:2105.04289*, 2021. 1

[34] Jesse Mu and Jacob Andreas. Compositional explanations of neurons. *Advances in Neural Information Processing Systems*, 33:17153–17163, 2020. 2

[35] Meike Nauta, Ron van Bree, and Christin Seifert. Neural prototype trees for interpretable fine-grained image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14933–14943, 2021. 3

[36] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978. 2, 4

[37] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008. 2, 5

[38] Kosuke Nishida, Kyosuke Nishida, and Shuichi Nishioka. Improving few-shot image classification using machine-and user-generated natural language descriptions. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1421–1430, 2022. 3

[39] Dong Huk Park, Lisa Anne Hendricks, Zeynep Akata, Anna Rohrbach, Bernt Schiele, Trevor Darrell, and Marcus Rohrbach. Multimodal explanations: Justifying decisions and pointing to the evidence. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8779–8788, 2018. 3

[40] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. 1

[41] Sarah Pratt, Rosanne Liu, and Ali Farhadi. What does a platypus look like? generating customized prompts for zero-shot image classification. *arXiv preprint arXiv:2209.03320*, 2022. 3

[42] 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*, pages 8748–8763. PMLR, 2021. 1, 3, 5, 16

[43] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020. 5, 13

[44] Yongming Rao, Wenliang Zhao, Guangyi Chen, Yansong Tang, Zheng Zhu, Guan Huang, Jie Zhou, and Jiwen Lu. Denseclip: Language-guided dense prediction with context-aware prompting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18082–18091, 2022. 3

[45] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. " why should i trust you?" explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016. 1

[46] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019. 1, 3

[47] Olga Russakovsky and Li Fei-Fei. Attribute learning in large-scale datasets. In *European Conference on Computer Vision*, pages 1–14. Springer, 2010. 3

[48] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *International Conference on Learning Representations*, 2018. 3

[49] Yoshihide Sawada and Keigo Nakamura. Concept bottleneck model with additional unsupervised concepts. *IEEE Access*, 10:41758–41765, 2022. 3

[50] Timo Schick and Hinrich Schütze. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, 2021. 3

[51] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 1, 2

[52] Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, 2020. 3

[53] Chandan Singh, John X Morris, Jyoti Aneja, Alexander M Rush, and Jianfeng Gao. Explaining patterns in data with language models via interpretable autoprompting. *arXiv preprint arXiv:2210.01848*, 2022. 3

[54] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017. 3

[55] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012. 2, 5

[56] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*, 2017. 3, 5

[57] Mujeen Sung, Jinhyuk Lee, Sean Yi, Minji Jeon, Sungdong Kim, and Jaewoo Kang. Can language models be biomedical knowledge bases? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4723–4734, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. 1

[58] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019. 3

[59] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In *European Conference on Computer Vision*, pages 266–282. Springer, 2020. 3

[60] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5(1):1–9, 2018. 2, 5

[61] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 13

[62] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016. 3

[63] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 2, 5

[64] Wenjia Xu, Yongqin Xian, Jiuniu Wang, Bernt Schiele, and Zeynep Akata. Attribute prototype network for zero-shot learning. *Advances in Neural Information Processing Systems*, 33:21969–21980, 2020. 3

[65] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021. 3

[66] Mert Yuksekgonul, Maggie Wang, and James Zou. Post-hoc concept bottleneck models. In *ICLR 2022 Workshop on PAIR^{2}Struct*, 2022. 1, 2, 3, 5, 6

[67] Tian Yun, Usha Bhalla, Ellie Pavlick, and Chen Sun. Do vision-language pretrained models learn primitive concepts? *arXiv preprint arXiv:2203.17271*, 2022. 3, 5, 6, 8

[68] Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2021. 1

[69] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16816–16825, 2022. 3

[70] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348, 2022. 3, 12, 14

A. Dataset Statistics

Table 7 depicts detailed statistics for all datasets. For each dataset, we provide in parentheses a one-word description of the type of classes it contains, which we refer to as *super class* of a dataset. We use the same train/dev/test splits of Food-101, Aircraft, Flower-102, UCF-101, and DTD provided by CoOp [70]. For CUB, we randomly sample 10 training images for each category as the development set. For CIFAR-10 and CIFAR-100, we randomly split 10% of the training data as the dev set. For HAM10000, we adopt 80/10/10 splits on the images of each class. For ImageNet, we only evaluate the dev set.

Name	**n. of class**	**n. of Images**		
		Train	Dev	Test
Food-101 (food)	101	50,500	20,200	30,300
FGVC-Aircraft (aircraft)	102	3,334	3,333	3,333
Flower-102 (flower)	102	4,093	1,633	2,463
CUB-200-2011 (bird)	200	3,994	2,000	5,794
UCF-101 (action)	101	7,639	1,898	3,783
DTD (texture)	47	2,820	1,128	1,692
HAM10000 (lesion)	7	8,010	1,000	1,005
RESISC45 (scene)	45	3,150	3,150	25,200
CIFAR-10 (object)	10	45,000	5,000	10,000
CIFAR-100 (object)	100	45,000	5,000	10,000
ImageNet (object)	1,000	1,281,167	50,000	-

Table 7. Detailed statistics of the 11 datasets. The text in parentheses that follows the dataset name corresponds to the super class name, which is used to remove class names in concepts.

B. Implementation Details

B.1. Linear Probe

Following CLIP's implementation of Linear Probe, we use the encoded images, before their projection to the vision-text embedding space, as input to the classifier. We use sklearn's L-BFGS implementation of logistic regression with 1,000 maximum iterations. To determine the best performing values for the L2 regularization strength C, we perform binary search on the validation set initialized with $[1e^6, 1e^4, 1e^2, 1, 1e^{-2}, 1e^{-4}, 1e^{-6}]$. After determining the left and right bounds of C, we iteratively halve the interval with 8 steps to get the final hyperparameter value. We compare our Linear Probe results on ImageNet with CoOp. To perform a fair comparison, we select CLIP-RN50 as the vision encoder and perform 3 random runs to select the few shot images. As shown in Table 8, we marginally outperform CoOp in all data settings.

B.2. Prompt

Table 9 presents the prompts used to query GPT-3. We design 5 general prompts and 5 additional prompts for UCF-101. The general prompts are used for all datasets, with a slight modification: we add the super-class name that describes the type of data present in more fine-grained datasets. For example, when prompting for Flower-102, we add the super class name *flower* after each class name. In this way we reduce ambiguity problems: e.g., for the class *bishop of llandaff*, without the super class name, GPT-3 returns results for *bishop* instead of the *flower*. While this approach reduces ambiguities, it does not completely eliminate them. For example, we found that GPT-3 generates sentences about the *mouse* (device), but in fact, the class *mouse* on ImageNet refers to the animal. Future work can explore better prompting methods, such as providing a detailed definition for each class or designing customized prompts for each dataset.

# of shots	**1**	**2**	**4**	**8**	**16**
CoOp	22.07	31.95	41.29	49.55	55.87
Ours	**22.26**	**32.28**	**41.57**	**49.80**	**55.92**

Table 8. Compare linear probe performance on ImageNet with CoOp. All experiments are based on CLIP-RN50 and we report the average score of 3 random runs.

General Prompt Template
1. describe what the `[CLASS NAME]` looks like:
2. describe the appearance of the `[CLASS NAME]`:
3. describe the color of the `[CLASS NAME]`:
4. describe the pattern of the `[CLASS NAME]`:
5. describe the shape of the `[CLASS NAME]`:
UCF-101 Prompt Template
1. describe what the `[CLASS NAME]` looks like:
2. describe the appearance of the `[CLASS NAME]`:
3. describe how to perform the `[CLASS NAME]`:
4. describe a person performing the `[CLASS NAME]`:
5. describe what can you see when a person is performing the `[CLASS NAME]`:

Table 9. The prompt templates used to generate the raw sentences from GPT-3. The UCF-101 has a different set of prompts, while the other datasets share the same set of general templates.

B.3. T5 concept extractor

The raw outputs of language models are long sentences and sometimes contain class names that need to be removed from the bottlenecks for the sake of interpretability. For example, GPT-3 generates a sentence "*The hen is brown and has a white chest.*" for the class *hen*, which could be decomposed to two concepts: "*brown*" and "*white chest*". We annotate a random sample of 100 sentence-concepts pairs from each of the following datasets: Food-101, CIFAR-100, Aircraft, Flower, and ImageNet. In total, we collect 500 sentences. An example annotation is depicted below:

> The 737-400 has a long and slender fuselage with tapered wings and a small tail. (737-400)
> long and slender fuselage; tapered wings; small tail

The class name is concatenated with the raw sentence, and the concepts are separated by semicolons. We train a T5-large model [43] using the Huggingface API. We add a task prefix - "*extract concepts from sentence:* " for each example. We train the model with Adam optimizer for 5 epochs, setting the batch size to 8 and learning rate to $1e^{-5}$.

B.4. Remove Class Name

After extracting the short concepts using T5, some still contain class names. To ensure there are no class names in the bottleneck, we design two heuristics: (1) If we find the class name in the concept using string match, we replace it with the super class name[9], e.g., the concept "*leaves of the **orange dahlia** are long and narrow*" for the class *orange dahlia* in Flower-102 is modified as "*leaves of the **flower** are long and narrow*". (2) For class names with multiple tokens, the tokens are not always in the same order as the class name. In this case, if a concept with all tokens for the class name present, we remove it. For instance, the concept "*a **cake** made of **carrot***" for the class *carrot cake* will be deleted. The two heuristics are applied to each concept by considering all class names in the dataset.

B.5. Hyperparameters

We apply grid search with 5 runs to find the best weights for the submodular function for different datasets and shots. We determine the learning rate and batch size by monitoring the validation accuracy with wandb. Table 16 lists all the hyperparameters of our best-performing models.

B.6. Other Details

GPT-3 Generation. Generating 500 sentences for one class takes around 5 minutes by calling the OpenAI APIs. The price of GPT-3-Davinci is $ 0.02 / 1k tokens, and it costs about $ 0.2 for each class.

Running Time. Because we use CLIP with frozen weights, we only need to extract the image features once and reuse them in the rest experiments. Since we only fit a single linear layer, our training time is low. For example, training the full ImageNet for one epoch on an NVIDIA RTX A6000 takes less than 1 minute.

Full Results. The full numerical results are shown in Table 14. Both validation and test accuracy are provided.

C. Additional Analysis

C.1. Activation Function

We ablate the impact of the softmax activation by removing it or replacing it with other activation functions such as ReLU and sigmoid. As shown in Table 10, not using an activation function significantly hurts performance, while using other activation functions performs poorly compared to softmax.

Activation	1	2	4	8	16	Full
-	52.66	58.01	63.02	68.93	73.52	81.32
`relu`	50.40	53.53	56.61	59.82	61.75	68.01
`sigmoid`	52.15	57.86	62.59	69.08	73.43	81.42
`softmax`	**63.03**	**67.79**	**71.88**	**76.08**	**79.10**	**85.71**

Table 10. Compare different activation functions. We report the mean accuracy across the 11 datasets.

C.2. Language Model Size vs. Performace

We experiment with different sizes of GPT-3: Curie, Babbage, and Ada (sorted from larger to smaller). Figure 11 compares the different GPT-3 variants on ImageNet, showing that larger language models result in better performance, especially in few show settings. However, there is only a marginal difference in performance when enough data is available.

GPT-3 type	1	2	4	8	16	Full
Davinci (175B)	**51.09**	**57.43**	**62.94**	**68.45**	**72.60**	83.97
Curie (13B)	45.75	53.89	60.36	66.96	71.65	**84.00**
Babbage (6.7B)	44.61	52.91	60.22	67.06	71.66	83.86
Ada (2.7B)	43.12	53.26	60.99	67.90	72.42	83.96

Table 11. The performance of LaBo on ImageNet using different sizes of GPT-3 to generate concepts. The number in the parenthesis is the number of parameters of the corresponding language model.

C.3. Performance of Human-Written Text

Table 12 compares the performance of LaBo between using GPT-3 generated concepts and human-designed concepts sourced from WordNet and Wikipedia. We observe that GPT-3 generated concepts outperform human-written ones in 1-shot experiments, while there is less than 1% drop in performance on average in larger data settings. In addition, our human evaluation on Imagenet (see Figure 5 and 6 in Section 5.3) shows that humans judge the quality of GPT-3 generated concepts to be better than that of human-designed concepts.

We visualize the embeddings of concepts and class names using t-SNE [61] to identify the reason behind the perceived higher quality of GPT-3 concepts. We encode the 1,000 class names of ImageNet using the CLIP text encoder along with the top-1 concept of each class (1,000 concepts in total) from each bottleneck (LaBo, WordNet, and Wikipedia). Figure 8 reflects that, compared to GPT-3, the embeddings of WordNet and Wikipedia concepts have a higher overlap with the embeddings of class names. In other words, Wikipedia and WordNet concepts are more likely to replicate the text features of class names rather than describe the class. This

[9]The super class name depends on the datasets. For example, the super class name for the Flower-102 dataset is *flower* (see Table 7).

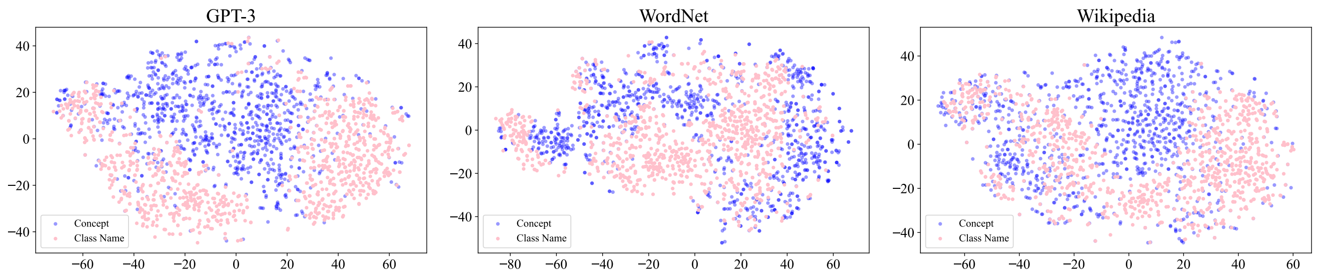


Figure 8. t-SNE visualization of the embeddings of concepts (blue) and class names (pink) on ImageNet. For the three bottlenecks constructed from GPT-3, WordNet and Wikipedia, we visualize the top-1 concept of each class ranked by the weights of the linear function.

Concept Source	1	2	4	8	16	Full
GPT-3	**51.09**	57.43	62.94	68.45	72.60	83.97
Wikipedia	48.76	56.73	63.00	68.96	73.07	**84.07**
WordNet	49.37	**57.84**	**64.10**	**69.92**	**73.35**	83.93

Table 12. The performance of LaBo on ImageNet using different sources of concepts to construct the bottlenecks.

Method	w/ cls	Aircraft	Food	Flower	DTD	UCF
LP	-	39.42	76.99	95.89	68.74	80.04
LaBo	✗	37.29	76.04	92.37	64.78	80.07
CoOp [70]	✓	33.22	**78.45**	**94.97**	**65.37**	78.66
LaBo†	✓	**37.53**	77.83	93.18	**65.37**	**80.10**

Table 13. Compare LaBo with prompt tuning methods on 5 datasets (16 shots). w/ cls stands for using class names in the context. LaBo† is our method without removing the class names in the concepts. All methods use CLIP-ViT-B/32 as the vision backbone.

explains why the human-written text has higher accuracy but is less interpretable.

C.4. Comparison with the Prompt Tuning Method

Table 13 compares the performance between LaBo and CoOp [70], which employs a soft prompt tuning method (not interpretable) on five datasets. Even though LaBo does not use class names, its performance is similar to that of CoOp. Adding class names to LaBo leads to performance gains, such that it outperforms CoOp on Aircraft and UCF-101.

D. Human Evaluation

We introduce two qualitative metrics to evaluate the automatically generated concept bottlenecks to highlight areas of possible improvement. We introduce two metrics that evaluate the bottleneck items along two dimensions: *Factuality* and *Groundability* (see Section 5.3).

Annotator Statistics. Both metrics rely on human annotations, which we collect on Amazon Mechanical Turk. To ensure confidence in the results, we collect 3 annotations per concept. Annotators are paid on average \$14.5 per hour, and the total cost of the annotation was \$2,100. Our rate was computed by estimating the time it takes to complete the task by 4 different control annotators.[10] In total, our task was completed by a diverse set of 477 annotators. The average pairwise annotator agreement for all annotated data without any pre-processing is 69.83%.

Interface. Figure 11 displays the annotation interface. Given a concept phrase, annotators are prompted to select from 12 images, 10 of which correspond to the ground truth target corresponding to the concept, and 2 control images randomly sampled from other classes. The user interface was accompanied by a set of instructions presented in Figure 12.

Invalid Annotations. In reporting *Factuality* and *Groundability*, we disregard annotations that select any of the control images unless all annotators failed the control for a particular concept. In total, we disregard 18% of annotations for this reason. In reporting invalid concepts (non-visual, non-sensical, or unknown vocabulary), we consider all annotations but consider a bottleneck invalid if at least 2 out of 3 annotators agree.

Analytic Results. Table 15 displays analytic results of *Factuality* and *Groundability* for all datasets. Figure 10 presents the invalid concept distribution for all datasets separately. It is worth noting the high percentage of non-visual concepts in CIFAR-10 and CIFAR-100 compared to other datasets. We hypothesize that this reflects the annotators' inability to see the images clearly due to the low resolution (see Figure 9) rather than the lack of visual content in the concept. For example, the concepts "small and black" and "blue nose and tail" were annotated as non-visual for CIFAR-10, and the concepts "color of trees and grass" and "two large pincers on its front legs" for CIFAR-100.

E. Qualitative Examples

Figure 9 shows the additional qualitative examples for the rest 4 datasets (CIFAR-10, CIFAR-100, DTD and Aircraft).

[10]Our focus group was graduate students. Since this is not representative of the average population, we doubled the time estimate.

Dataset	Method	Dev						Test					
		1	2	4	8	16	Full	1	2	4	8	16	Full
Food-101	Linear Prob	58.04	75.24	84.16	**87.48**	**89.87**	**93.11**	57.75	75.34	84.21	**87.90**	**90.02**	**93.17**
	LaBo (Ours)	**80.32**	**84.15**	**85.76**	87.07	88.74	92.53	**80.41**	**84.05**	**85.68**	87.39	88.77	92.45
Aircraft	Linear Prob	27.63	34.86	41.40	**49.72**	**57.91**	**62.89**	28.26	35.07	**41.55**	**50.26**	**56.38**	**64.03**
	LaBo (Ours)	**33.12**	**35.97**	**42.90**	49.08	56.41	61.96	**32.73**	**37.71**	41.04	48.81	54.97	61.42
Flower-102	Linear Prob	**89.20**	**94.06**	**97.00**	**98.40**	**98.91**	**99.11**	**88.06**	**93.65**	**97.67**	**98.56**	**99.32**	**99.45**
	LaBo (Ours)	82.24	88.18	94.92	96.20	98.16	98.65	82.05	90.09	95.21	97.08	98.66	99.35
CUB	Linear Prob	48.55	60.40	**72.50**	**78.25**	**83.35**	**83.60**	47.69	61.06	**72.82**	**79.60**	**83.74**	**84.54**
	LaBo (Ours)	**55.20**	**64.80**	72.45	76.55	79.90	81.00	**54.19**	**64.60**	71.21	77.22	80.69	81.90
UCF-101	Linear Prob	65.54	76.34	85.83	90.25	**93.63**	**98.63**	60.56	73.22	80.62	85.70	**87.63**	**90.67**
	LaBo (Ours)	**80.72**	**83.77**	**88.46**	**90.73**	93.05	97.68	**78.75**	**82.05**	**84.56**	**86.39**	87.39	90.11
DTD	Linear Prob	43.62	53.19	60.55	**68.79**	**74.47**	**80.50**	41.67	51.71	60.76	**69.03**	**74.70**	**81.68**
	LaBo (Ours)	**55.59**	**56.47**	**62.15**	68.44	70.92	76.86	**53.61**	**55.26**	**61.17**	66.43	70.21	77.30
HAM10000	Linear Prob	32.30	**55.40**	45.40	50.90	**63.10**	**84.40**	33.13	**55.32**	44.48	48.26	**61.69**	**83.18**
	LaBo (Ours)	**34.90**	46.40	**45.80**	**54.40**	58.20	81.40	**36.62**	45.17	**45.87**	**52.04**	55.72	81.39
RESISC45	Linear Prob	68.62	**79.10**	**86.72**	**89.89**	**92.49**	**95.24**	67.57	**77.75**	**86.50**	**89.27**	**92.17**	**94.98**
	LaBo (Ours)	**73.02**	76.03	81.37	85.05	88.86	91.65	**73.66**	76.11	81.40	85.71	88.63	91.22
CIFAR-10	Linear Prob	62.36	80.32	92.94	**95.36**	**96.06**	**98.16**	62.44	80.27	92.54	**95.14**	**95.90**	**98.10**
	LaBo (Ours)	**91.24**	**91.04**	**92.98**	94.40	95.06	97.90	**91.06**	**90.79**	**93.03**	94.11	94.93	97.75
CIFAR-100	Linear Prob	39.66	57.84	70.06	**76.52**	**80.34**	**87.70**	39.26	57.35	69.73	**76.22**	**80.16**	**87.48**
	LaBo (Ours)	**62.84**	**66.56**	**71.78**	75.30	78.08	86.82	**62.73**	**65.80**	**70.82**	74.49	77.67	86.04
ImageNet	Linear Prob	42.25	55.71	**64.80**	**71.23**	**75.08**	83.90	-	-	-	-	-	-
	LaBo (Ours)	**51.09**	**57.43**	62.94	68.45	72.60	**83.97**	-	-	-	-	-	-
Average	Linear Prob	52.53	65.68	72.85	**77.89**	**82.29**	**87.93**	51.69	65.13	**72.33**	**77.38**	**81.53**	**87.38**
	LaBo (Ours)	**63.66**	**68.25**	**72.86**	76.88	80.00	86.40	**63.35**	**68.10**	72.08	76.19	79.11	85.72

Table 14. Full results of Linear Prob and LaBo on the development and test sets of 11 datasets.

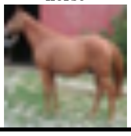
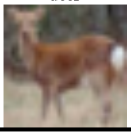
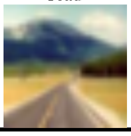
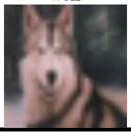
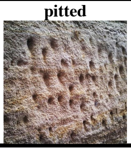
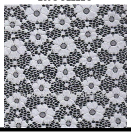
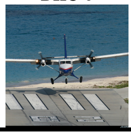
	Class Name	Top-3 Concepts	Class Name	Top-3 Concepts	Class Name	Top-3 Concepts	Class Name	Top-3 Concepts
CIFAR-10	**badger**	1. blue nose and tail 2. versatile vehicle 3. amazing	**horse** 	1. tail is long and flowing 2. large bed in the back for carrying cargo 3.soft muzzle	**deer** 	1. peaceful creature 2. muzzle is long and narrow 3. fur is soft and thick	**frog**	1. popular pet because it is easy to care for 2. two short, sharp horns on its head 3. croak
CIFAR-100	**beaver**	1. sensitive whiskers on its face 2. eats leaves, bark, and twigs 3. large, stocky rodent with a thick, brown coat of fur	**house**	1. windows are evenly spaced 2. a lot of windows and doors 3. bookshelf and comfortable object	**road** 	1. color of freshly tarred driveway 2. bordered on each side by a grassy shoulder 3.lead to a distant horizon	**wolf** 	1. thick and gray fur 2. often seen running and playing with its pack mates 3. light brown coat with a black nose and dark eyes
DTD	**wrinkled**	1. intersect and criscross each other 2. looks like a dry, crumpled paper 3. looks like a piece of cloth that has been crumpled up	**spiralled**	1. consistent width throughout the spiral 2. tight, spiralling curls 3. clockwise or counterclockwise	**pitted** 	1. always smooth 2. these depressions may be evenly spaced or clustered together 3. these holes are evenly spaced	**lacelike** 	1. complex 2. arranged in a symmetrical fashion 3. a lot of small holes that make it look like a net
Aircraft	**737-200**	1. professional color 2. first 737 to be equipped with winglets 3. equipped with an apu	**DHC-6** 	1. stol aircraft with a fixed tricycle landing gear 2. floats for operation on water 3. twin-engined stol utility aircraft	**Gulfstream IV**	1. spacious cabin and large windows 2. "t-tail" configuration 3. first flown in 1985	**DR-400**	1. entered via a sidehinged canopy 2. enclosed cockpit 3. drives a threebladed

Figure 9. Additional qualitative examples for CIFAR-10, CIFAR-100, DTD and Aircraft.

	Food	**Aircraft**	**HAM10K**	**RESISC**	**Flower**	**CUB**	**UCF**	**DTD**	**CIFAR10**	**CIFAR100**
Factuality ↑	P@10	P@10	P@10	P@10	P@8	P@10	P@10	P@10	P@10	P@10
LaBo	**33.07**	**11.57**	15.05	14.80	11.48	**27.97**	**37.78**	23.90	14.70	22.48
w/o submod	27.08	8.10	9.57	**16.40**	**18.58**	23.12	37.22	**25.27**	**20.70**	**22.72**
w/o LM	21.63	8.97	**19.71**	12.15	9.98	12.17	20.43	14.83	6.87	14.97
Groundability↑	P@10	P@10	P@10	P@10	P@8	P@10	P@10	P@10	P@10	P@10
LaBo	10.98	8.48	18.83	13.87	9.53	15.63	8.08	8.90	5.70	19.83
w/o submod	**21.52**	**13.67**	17.22	**17.90**	**21.52**	23.07	**29.93**	20.02	**23.10**	21.78
w/o LM	20.58	12.00	**20.00**	14.38	17.93	**25.02**	27.96	**20.31**	7.15	**27.04**

Table 15. Analytic Factuality and Groundability for all datasets ecept Imagenet (see Fig. 5)

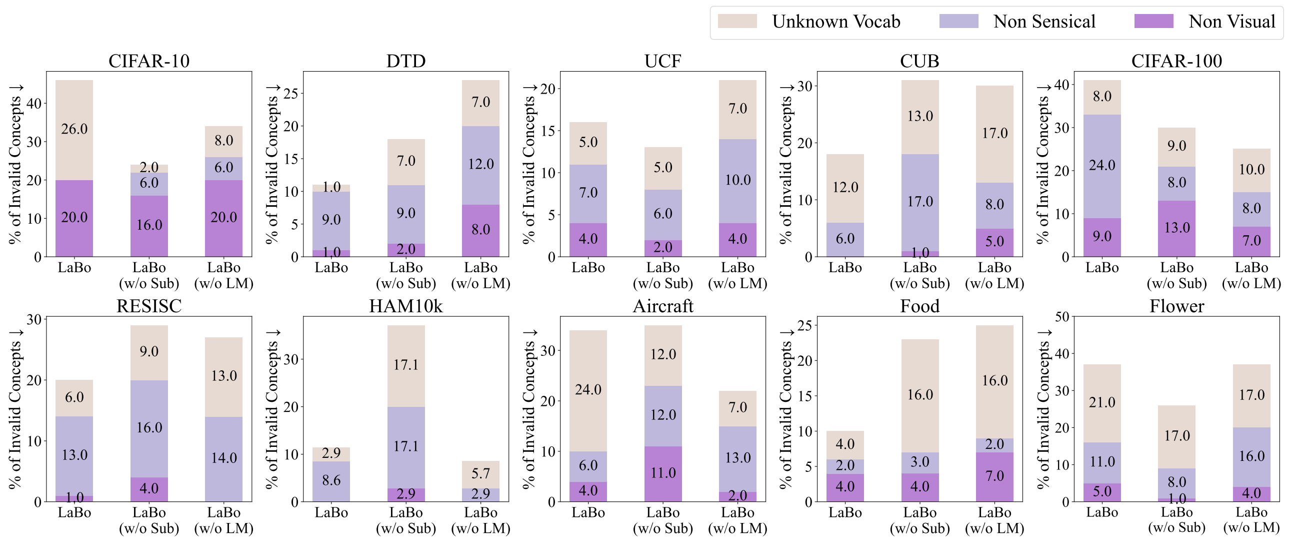
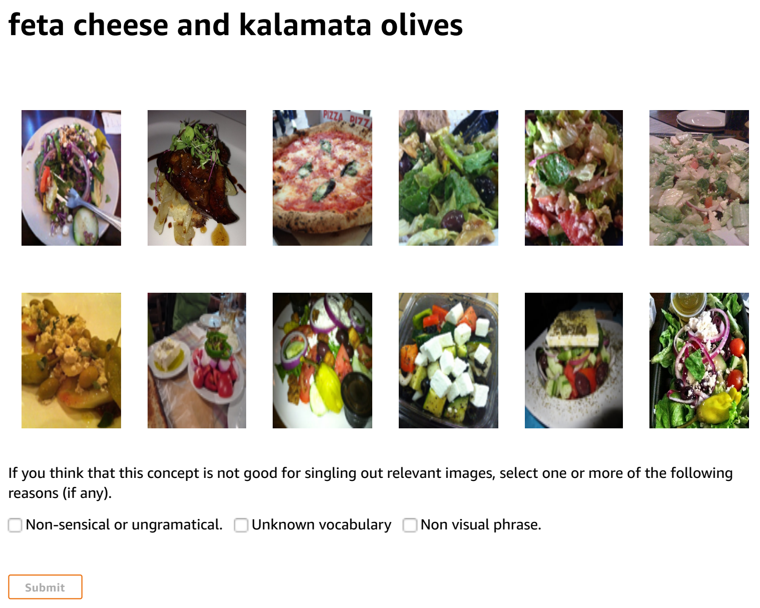


Figure 10. Percentage of invalid concepts identified by humans for different bottlenecks for all 10 datasets except ImageNet (see Figure 6). **Lower** percentage is better.



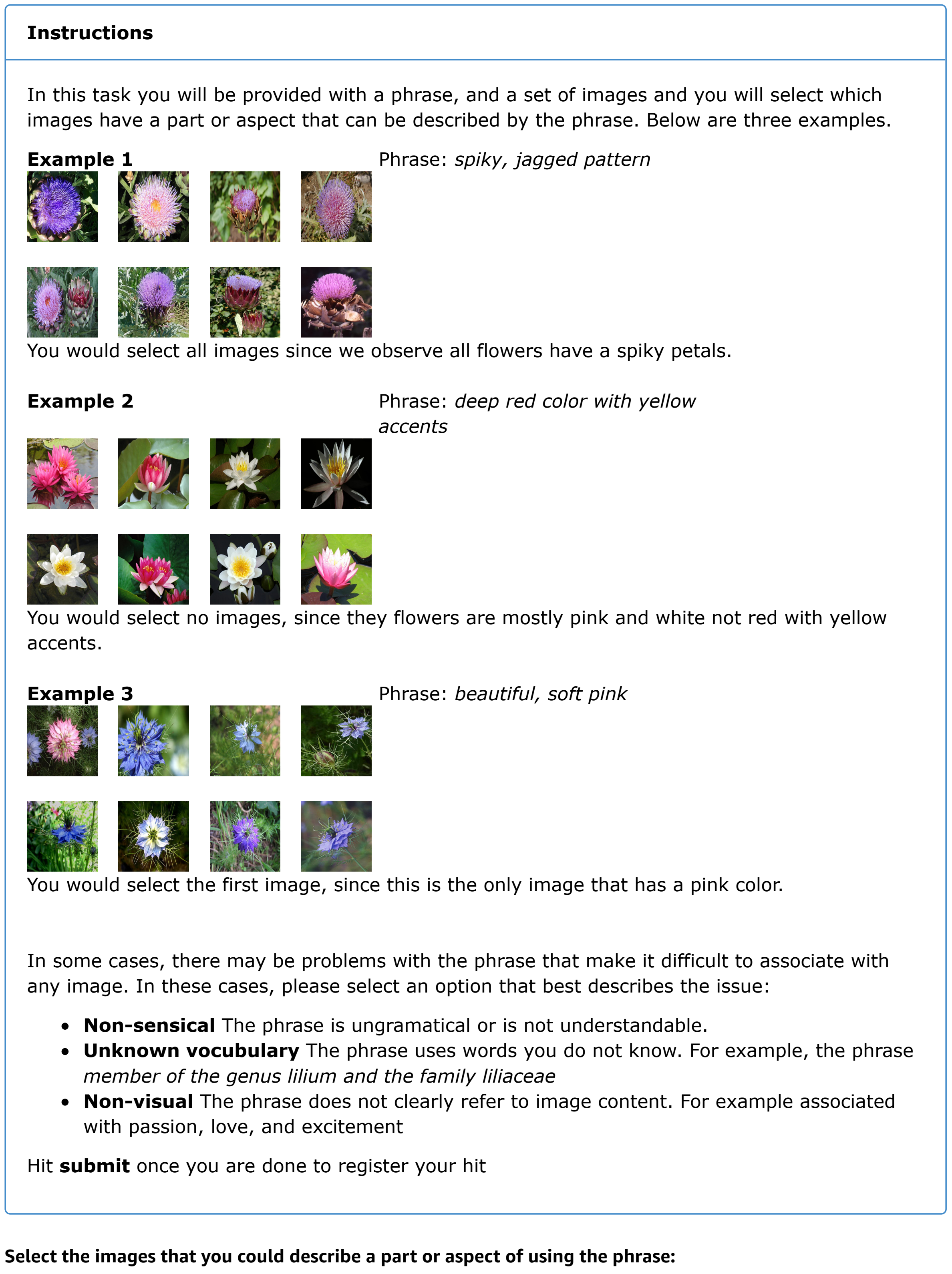
feta cheese and kalamata olives

If you think that this concept is not good for singling out relevant images, select one or more of the following reasons (if any).

☐ Non-sensical or ungramatical. ☐ Unknown vocabulary ☐ Non visual phrase.

Submit

Figure 11. Sample user interface for measuring *Factuality*. We provide 10 ground truth images with 2 control images randomly positioned. Annotators are required to select the images that can be described by the phrase. The user interface for *Groundability* is identical, but the images presented are the top-10 images in the dataset sorted by CLIP [42] similarity score.



Instructions

In this task you will be provided with a phrase, and a set of images and you will select which images have a part or aspect that can be described by the phrase. Below are three examples.

Example 1 Phrase: *spiky, jagged pattern*

You would select all images since we observe all flowers have a spiky petals.

Example 2 Phrase: *deep red color with yellow accents*

You would select no images, since they flowers are mostly pink and white not red with yellow accents.

Example 3 Phrase: *beautiful, soft pink*

You would select the first image, since this is the only image that has a pink color.

In some cases, there may be problems with the phrase that make it difficult to associate with any image. In these cases, please select an option that best describes the issue:

- **Non-sensical** The phrase is ungramatical or is not understandable.
- **Unknown vocubulary** The phrase uses words you do not know. For example, the phrase *member of the genus lilium and the family liliaceae*
- **Non-visual** The phrase does not clearly refer to image content. For example associated with passion, love, and excitement

Hit **submit** once you are done to register your hit

Select the images that you could describe a part or aspect of using the phrase:

Figure 12. Instructions provided to annotators to compute *Factuality* and *Groundability*.

	n. of shots	Bottleneck Size	Discriminability (α)	Coverage (β)	Learning Rate	Batch Size
Food-101	1	5,050	$1e^7$	0.5	$1e^{-5}$	16
	2	5,050	$1e^7$	1	$1e^{-4}$	32
	4	5,050	$1e^7$	1	$1e^{-4}$	64
	8	5,050	$1e^7$	1	$1e^{-4}$	128
	16	5,050	$1e^7$	1	$1e^{-4}$	256
	Full	5,050	$1e^7$	5	$1e^{-5}$	1024
Aircraft	1	5,100	$1e^7$	0.5	$5e^{-5}$	16
	2	5,100	$1e^7$	1	$5e^{-5}$	32
	4	5,100	$1e^7$	0.1	$5e^{-5}$	64
	8	5,100	$1e^7$	0	$5e^{-5}$	128
	16	5,100	$1e^7$	1	$5e^{-5}$	256
	Full	5,100	$1e^7$	0.5	$5e^{-5}$	256
Flower-102	1	2,050	$1e^7$	10	$1e^{-5}$	16
	2	2,050	$1e^7$	100	$1e^{-5}$	32
	4	2,050	$1e^7$	10	$1e^{-5}$	64
	8	2,050	$1e^7$	10	$1e^{-5}$	128
	16	2,050	$1e^7$	1	$1e^{-5}$	256
	Full	2,050	$1e^7$	1	$1e^{-5}$	256
CUB	1	2,000	$1e^7$	0	$5e^{-5}$	32
	2	2,000	$1e^7$	0	$5e^{-5}$	64
	4	2,000	$1e^7$	0.1	$5e^{-5}$	128
	8	2,000	$1e^7$	0	$5e^{-5}$	256
	16	2,000	$1e^7$	1	$5e^{-5}$	512
	Full	2,000	$1e^7$	0.1	$5e^{-5}$	512
UCF-101	1	5,050	$1e^7$	1	$1e^{-5}$	8
	2	5,050	$1e^7$	1	$1e^{-5}$	16
	4	5,050	$1e^7$	100	$1e^{-5}$	32
	8	5,050	$1e^7$	10	$1e^{-5}$	64
	16	5,050	$1e^7$	100	$1e^{-5}$	128
	Full	5,050	$1e^7$	10	$1e^{-5}$	256
DTD	1	2,350	$1e^7$	10	$1e^{-5}$	8
	2	2,350	$1e^7$	10	$1e^{-5}$	16
	4	2,350	$1e^7$	5	$1e^{-5}$	32
	8	2,350	$1e^7$	1	$1e^{-5}$	64
	16	2,350	$1e^7$	2.5	$5e^{-5}$	256
	Full	2,350	$1e^7$	7.5	$1e^{-4}$	512
HAM10000	1	350	$1e^7$	0.1	$1e^{-3}$	4
	2	350	$1e^7$	0.1	$1e^{-3}$	4
	4	350	$1e^7$	1	$1e^{-4}$	8
	8	350	$1e^7$	10	$1e^{-3}$	8
	16	350	$1e^7$	15	$1e^{-3}$	16
	Full	350	$1e^7$	0.1	$5e^{-4}$	256
RESISC45	1	2,250	$1e^7$	5	$5e^{-5}$	8
	2	2,250	$1e^7$	5	$5e^{-5}$	16
	4	2,250	$1e^7$	10	$5e^{-5}$	32
	8	2,250	$1e^7$	15	$5e^{-5}$	64
	16	2,250	$1e^7$	15	$5e^{-5}$	128
	Full	2,250	$1e^7$	15	$5e^{-5}$	256
CIFAR-10	1	500	$1e^7$	1	$1e^{-4}$	2
	2	500	$1e^7$	5	$5e^{-4}$	4
	4	500	$1e^7$	5	$1e^{-4}$	8
	8	500	$1e^7$	1	$1e^{-4}$	16
	16	500	$1e^7$	10	$1e^{-4}$	32
	Full	500	$1e^7$	5	$1e^{-4}$	512
CIFAR-100	1	5,000	$1e^7$	7.5	$1e^{-5}$	16
	2	5,000	$1e^7$	2.5	$1e^{-5}$	32
	4	5,000	$1e^7$	7.5	$1e^{-5}$	64
	8	5,000	$1e^7$	7.5	$1e^{-5}$	128
	16	5,000	$1e^7$	5	$1e^{-5}$	256
	Full	5,000	$1e^7$	0	$1e^{-5}$	512
ImageNet	1	50,000	$1e^8$	0	$1e^{-5}$	128
	2	50,000	$1e^8$	0	$1e^{-5}$	256
	4	50,000	$1e^8$	0	$1e^{-5}$	256
	8	50,000	$1e^8$	0	$1e^{-5}$	512
	16	50,000	$1e^8$	0	$1e^{-5}$	1024
	Full	50,000	$1e^8$	0	$1e^{-5}$	2048

Table 16. All hyperparameters used for the main experiments which are tuned on the development set.