ON COMPLETENESS-AWARE CONCEPT-BASED EXPLANATIONS IN DEEP NEURAL NETWORKS

A PREPRINT

Chih-Kuan Yeh[1], Been Kim[2], Sercan Ö. Arık[3], Chun-Liang Li[3], Tomas Pfister[3], and Pradeep Ravikumar[1]

[1]Machine Learning Department, Carnegie Mellon University
[2]Google Brain
[3]Google Cloud AI

ABSTRACT

Human explanations of high-level decisions are often expressed in terms of key concepts the decisions are based on. In this paper, we study such concept-based explainability for Deep Neural Networks (DNNs). First, we define the notion of completeness, which quantifies how sufficient is a particular set of concepts in explaining a model's prediction behavior. Next, we propose a concept discovery method that aims to infer a complete set of concepts that are additionally encouraged to be interpretable. Our concept discovery method aims to address the limitations of commonly-used methods such as PCA and TCAV. To define an importance score for each discovered concept, we adapt game-theoretic notions to aggregate over sets and propose *ConceptSHAP*. On a Synthetic dataset with ground-truth concept explanations, on a real-world dataset, and with a user study, we validate the effectiveness of our framework in finding concepts that are both complete in explaining the decisions, and interpretable.

1 Introduction

The lack of explainability of deep neural networks (DNNs) constitutes a bottleneck towards their full potential for real-world impact. Especially in high-stake decision cases such as in medicine, intuitive explanations, are highly valuable. They can help domain experts better understand rationals behind the model decisions, identify systematic failure cases, and potentially provide feedback to model builders for improvements.

The most commonly-used methods for DNNs explain each prediction by quantifying the importance of each input feature [Ribeiro et al., 2016, Lundberg and Lee, 2017]. One caveat with such explanations is that they typically focus on the local behavior for each data point, rather than globally explaining how the model reasons. Another caveat is that weighted input features are not necessarily the most intuitive explanations for human understanding, particularly when using low-level features such as raw pixel values. Human reasoning often comprise "concept-based thinking," extracting similarities from numerous examples and grouping them systematically based on their resemblance [Armstrong et al., 1983, Tenenbaum, 1999]. It is thus of interest to develop such "concept-based explanations" to characterize the global behavior of a DNN in a way understandable to humans by explaining how DNNs use concepts in arriving at particular decisions.

A few recent studies have focused on bringing such concept-based explainability to DNNs, largely based on the common implicit assumption that the concepts lie in low-dimensional subspaces of some intermediate DNN activations. Most of these approaches assume exogenous information of key concepts in the form of supervised training data [Kim et al., 2018, Zhou et al., 2018]. Given such training data, TCAV [Kim et al., 2018] trains linear concept classifiers to derive concept vectors, and uses how sensitive predictions are to these vectors (directional derivatives) to measure the importance of a concept with respect to a specific class. From concept vectors, Zhou et al. [2018] considers the decomposition of model predictions in terms of projections onto concept vectors. Instead of human-labeled concept training data, Ghorbani et al. [2019] employs k-means clustering of super-pixel segmentations of images, and uses the corresponding image clusters as training data for concepts. Bouchacourt and Denoyer [2019] proposes a Bayesian generative model involving concept vectors. One drawback of these concept extraction approaches does not take into

account *how much* each concept play a role in the prediction. This motivates the following key questions: Is there an unsupervised approach to extract concepts that are sufficiently predictive of a DNN's decisions? If so, how can we measure the sufficiency?

Note that selecting a set of concepts salient to a particular class does not guarantee that these concepts are sufficient in explaining the prediction. The notion of sufficiency is referred to as "completeness" of explanations, as in [Gilpin et al., 2018, Yang et al., 2019]. In this paper, we propose a completeness score for concept-based explanations. Our metric can be applied to a set of concept vectors that lie in a subspace of some intermediate DNN activations, which is a general assumption in previous work in this context [Kim et al., 2018, Zhou et al., 2018]. Intuitively speaking, a set of "complete" concepts can fully explain the prediction. By further assuming that for complete concept, the occurrence of concepts are a sufficient statistic for the prediction of the model, we may measure the "completion" of the concepts by the accuracy of the model by taking the concepts existence as input. For concept discovery we propose a novel algorithm, that optimizes a surrogate likelihood of the concept-based data generation process, motivated by topic modeling [Blei et al., 2003].

We further introduce an interpretability regularizer, to ensure that the discovered complete concepts are also coherent (distinct from other concepts) and semantically meaningful. Beyond concept discovery, we also propose a score, *ConceptSHAP*, for quantification of concept attributions as contextualized importance. We show that ConceptSHAP is the only scoring method that satisfies a key set of axioms involving the contribution of each concept to the completeness score [Shapley, 1988, Lundberg and Lee, 2017]. We also propose a class-specific version of ConceptSHAP that decomposes the ConceptSHAP with respect to each class in multi-class classification. This can be used to find class-specific concepts that contribute the most to a specific class. To verify the effectiveness of our *automated* completeness-aware concept discovery method, we create a Synthetic dataset with apriori-known ground truth concepts. We show that our approach outperforms all compared methods in correct retrieval of the concepts as well as in terms of its coherency via a user study. We also demonstrate how our concept discovery algorithm provides additional insights into the behavior of DNN models on a real-world dataset.

2 Related Work

Most of post-hoc interpretability methods fall under two categories: (i) feature-based explanation methods, that attribute the decision to important input features [Ribeiro et al., 2016, Lundberg and Lee, 2017, Smilkov et al., 2017, Chen et al., 2018], and (ii) sample-based explanation methods, that attribute the decision to previously observed samples [Koh and Liang, 2017, Yeh et al., 2018, Khanna et al., 2019, Arik and Pfister, 2019]. Recent work has also focused on *evaluations* of explanations, ranging from human-centric evaluations [Lundberg and Lee, 2017, Kim et al., 2018] to functionally-grounded evaluations [Samek et al., 2016, Kim et al., 2016, Ancona et al., 2017, Yeh et al., 2019, Yang et al., 2019]. Our work provides an evaluation of concept expalnations based on the completeness criteria.

Our work is related to methods that learn semantically meaningful latent variables. Some use dimensionality reduction methods [Chan et al., 2015, Kingma and Welling, 2013], while others uncover higher level human-relatable concepts by dimensionality reduction (eg speech data [Chorowski et al., 2019] and language [Radford et al., 2017]). More recently Locatello et al. [2018] showed that meaningful latent dimensions cannot be acquired in a completely unsupervised setting, while implying that inductive biases is essential in discovering meaningful latent dimensions. Our work uses indirect supervision from the classifier of interest to discover semantically meaningful latent dimensions.

3 Defining Completeness of Concepts

Problem setting: Consider a set of n training examples $\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^n$, corresponding labels $y^1, y^2, ..., y^n$ and a given pre-trained DNN model that predicts the corresponding y from the input $\mathbf{x}$. We assume that the pre-trained DNN model can be decomposed into two functions: the first part $\Phi(\cdot)$ maps input $\mathbf{x}^i$ into an intermediate layer $\Phi(\mathbf{x}^i)$, and the second part $h(\cdot)$ maps the intermediate layer $\Phi(\mathbf{x}^i)$ to the output $h(\Phi(\mathbf{x}^i))$, which is a probability vector for each class. For DNNs that build up by processing parts of input at a time, such as those composed of convolutional layers, we can additionally assume that $\Phi(\mathbf{x}^i)$ is the concatenation of $[\phi(\mathbf{x}^i_1), ..., \phi(\mathbf{x}^i_T)]$, such that $\Phi(\cdot) \in \mathbb{R}^{(T \cdot d)}$, and $\phi(\cdot) \in \mathbb{R}^d$. Here, $\mathbf{x}^i_1, \mathbf{x}^i_2, ..., \mathbf{x}^i_T$ denote different, potentially overlapping parts of the input for $\mathbf{x}^i$, such as a segment of an image or a sub-sentence of a text. These parts for example, can be chosen to correspond to the receptive field of the neurons at the intermediate layer $\Phi(\cdot)$. We will use these $\mathbf{x}^i_t$s to relate discovered concepts. To exemplify such parts, consider the fifth convolution layer of a VGG-16 network with input shape 224×224 have the size $7 \times 7 \times 512$. If we treat this layer as $\Phi(\mathbf{x}^i)$, $\phi(\mathbf{x}^i_1)$ corresponds to the first 512 dimensions of the intermediate layer, and $\Phi(\mathbf{x}^i) = [\phi(\mathbf{x}^i_1), ..., \phi(\mathbf{x}^i_{49})]$. Here, each $\mathbf{x}^i_j$ corresponds to a 164×164 square in the input image (with effective stride 16), which is the receptive field of convolution layer 5 of VGG-16 [Araujo et al., 2019]. We note that when the receptive field of $\phi(\cdot)$ is equal to the entire

input size, as in multi-layer perceptrons, we may simply choose $T = 1$ so that $\mathbf{x}^i_{1:T} = \mathbf{x}^i$ and $\Phi(\mathbf{x}^i) = \phi(\mathbf{x}^i_1)$. Thus, our method can also be generally applied to any DNN with an arbitrary structure besides convolutional layers.

Suppose that there is a set of m concepts denoted by unit vectors[1] $\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_m$ that represent linear directions in the activation space $\phi(\cdot) \in \mathbb{R}^d$, given by a concept discovery algorithm. For each part of data point $\mathbf{x}_t$[2], let $\mathbf{z}_t \in \{0, 1\}^m$ be a binary vector where $\mathbf{z}_{t,j} = 1$ represents that $\mathbf{x}_t$ contains information of concept $\mathbf{c}_j$. Generally, we assume that the probability of whether $\mathbf{x}_t$ contains concept $\mathbf{c}_j$ as:

$$P(\mathbf{z}_{t,j} = 1|\mathbf{x}_t) \propto \gamma(\phi(\mathbf{x}_t), \mathbf{c}_j; \beta),$$

where $\gamma(\mathbf{v}_1, \mathbf{v}_2; \beta) = \mathbb{1}[\langle \mathbf{v}_1, \mathbf{v}_2 \rangle > \beta] \langle \mathbf{v}_1, \mathbf{v}_2 \rangle$. β is a threshold to ignore the probability of assigning to negligibly-likely concepts, with $\beta \geqslant 0$ to ensure a non-negative probability. The probability is a thresholded dot product between the concept vector and the embedding of an input part, where dot product is used to represent the similarity between intermediate representations of a DNN. We refer the probability $\mathbb{E}[\mathbf{z}_{t,j}|\mathbf{x}_t] = P(\mathbf{z}_{t,j} = 1|\mathbf{x}_t)$ as the *concept score*.

We construct the matrix $\mathbf{x}$ as $\mathbb{E}[\mathbf{z}_{1:T}] \in \mathbb{R}^{T \times m}$[3], for all concept scores of all parts $\mathbf{x}_{1:T}$. We define the completeness score of concepts as the ratio of the preserved predictability when the prediction is only made with given concepts:

Definition 3.1. Completeness Score: Given a prediction model $f(\mathbf{x}) = h(\phi(\mathbf{x}))$, a set of concept vectors $\mathbf{c}_1, ..., \mathbf{c}_m$, we define the completeness score $\eta(\mathbf{c}_1, ..., \mathbf{c}_m)$ as:

$$\frac{\mathbb{E}_{\mathbf{x},y\sim V}[\mathbb{1}[y = \arg\max_{y'} P(y'|\mathbb{E}[z_{1:T}], h)]] - R}{\mathbb{E}_{\mathbf{x},y\sim V}[\mathbb{1}[y = \arg\max_{y'} P(y'|\mathbf{x}_{1:T}, f)]] - R}, \tag{1}$$

where $\mathbb{E}_{\mathbf{x},y\sim V}[\mathbb{1}[y = \arg\max_{y'} P(y'|\mathbb{E}[z_{1:T}], h)]]$ is the accuracy by predicting the label just given the concept scores $\mathbb{E}[z_{1:T}]$, and R is the accuracy of random prediction to equate the lower bound of completeness score to 0. When the target y is multi-label, we may generalize the definition of completeness score by replacing the accuracy with the binary accuracy, which is the accuracy where each label is treated as a binary classification.

To calculate $P(y|\mathbb{E}[z_{1:T}], h)$, we first learn a projection $l : \mathbb{R}^{T \cdot m} \to \mathbb{R}^{T \cdot d}$, which maps the concept space back to the activation space, and then pass the reconstructed activation space back through the original DNN. The final calculation of $P(y|\mathbb{E}[z_{1:T}], h)$ is given by:

$$P(y|\mathbb{E}[z_{1:T}], h) = P(y|h(\widehat{l}(\mathbb{E}[z_{1:T}]))), \tag{2}$$

where $\widehat{l}$ is set to maximize completeness score (1) (or the differential surrogate of completeness score which replaces the accuracy by the cross entropy loss). l could be a DNN, or a simple linear projection. The interpretation of the completeness score is that we assume the concept score for "complete concepts" is a sufficient statistic of the model, such that $P(y|\mathbb{E}[\mathbf{z}_{1:T}|\mathbf{x}_{1:T}], h) = P(y|\mathbf{x})$. By measuring the accuracy made by the concept score, we are effectively measuring how "complete" the concepts are.

Below is an illustrative example on why we need the completeness score:

Example 3.1. Consider a simplified scenario where we have the input $\mathbf{x} \in \mathbb{R}^m$, and the intermediate layer Φ is the identical function. In this case, the m concepts $\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_m$ are naturally the one-hot encoding of each feature in $\mathbf{x}$. Assume that the concepts $\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_m$ follow independent Bernoulli distribution with $p = 0.5$, and the model we attempt to explain is $f(\mathbf{x}) = \mathbf{c}_1$ XOR $\mathbf{c}_2$... XOR $\mathbf{c}_m$. The ground truth concepts that are sufficient to the prediction of the model should then be $\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_m$. However, if we have the information of $\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_{m-1}$ but do not have information regarding $\mathbf{c}_m$, we may have at most 0.5 probability to predict the output of the model, which is the same as random. In this case, $\eta(\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_{m-1}) = 0$. On the other hand, given $\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_m$, $\eta(\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_m) = 1$.

The completeness score offers a way to assess the 'sufficiency' of the discovered concepts to "explain" reasoning behind a model's decision. Not only the completeness score is useful in evaluating a proposed concept discovery method, but it can also shed light on how much of what the DNN learned may not be 'understandable' to humans. For example, if the completeness score is very high, but discovered concepts aren't making cohesive sense to users, this may mean that the DNN is basing its decisions on other concepts that are potentially hard to explain.

[1] We also apply additional normalization to $\phi(\cdot)$ so it has unit norm, and use $\phi(\cdot)$ for the normalized embedding for simplicity.

[2] We omit i for notational simplicity

[3] We omit the dependency on $\mathbf{x}_{1:T}$ for simplicity.

4 Discovering Completeness-aware Interpretable Concepts

4.1 Limitations of existing methods

Our goal is to discover a set of maximally-complete concepts under the definition 4.2, where each concept is also interpretable and semantically-meaningful to humans. We first discuss the limitations of recent notable work related to concept discovery and then explain how we address them.

PCA: We show that under strict conditions, the PCA vectors applied on an intermediate layer where the principle components are used as concept vectors, maximizes the completeness score.

Proposition 4.1. *When h is an isometry function that maps from $(\Phi(\cdot), \|\cdot\|_F) \rightarrow (f(\cdot), \|\cdot\|_F)$, and additionally $f(\mathbf{x}_i) = \mathbb{1}[y_i],\ \forall(\mathbf{x}_i, y_i) \in V$ (i.e. the loss is minimized, $\mathbb{1}[y_i]$ is the one hot vector of class y_i), and also assume $T = 1$, $\mathbb{E}[z] = \langle \phi(\mathbf{x}), \mathbf{c} \rangle$, and l is a linear function, the first m PCA vectors maximizes the L2 surrogate of η.*

The proof is in Appendix. We note that the assumptions for this proposition are extremely stringent, and may not hold in general. When the isometry and other assumptions do not hold, PCA no longer maximizes the completeness score as the lowest reconstruction in the intermediate layer do not imply the highest prediction accuracy in the output. In fact, DNNs are shown to be very sensitive to small perturbations in the input [Narodytska and Kasiviswanathan, 2017] – they can yield very different outputs although the difference in the input is small (and often perceptually hard to recognize to humans). Thus, even though the reconstruction loss between two inputs are low at an intermediate layer, subsequent deep nonlinear processing may cause them to diverge significantly. The principal components are also not trained to be semantically meaningful, but to greedily minimize the reconstruction error (or maximize the projected variance). Even though completeness score and PCA share the idea of minimizing the reconstruction loss via dimensionality reduction, the lack of human interpretability of the principle components is a major bottleneck for PCA.

TCAV & ACE: TCAV and ACE are concept discovery methods that use training data for specific concepts and use trained linear concept classifier to derive concept vectors. They additionally quantify the saliency of a concept to a class, which they term the TCAV score, in terms of the similarity of the loss gradients to the concept vectors, which implicitly assumes a first-order relationship between the concepts and the model outputs. With regards to the training data for the concept classifiers, TCAV relies on human-defined labels, while ACE uses automatically derived image clusters by k-means clustering of super-pixel segmentations. There are two main caveats to these approaches. The first is that while they may retrieve an important set of concepts, there is no guarantee on how 'complete' the concepts are in explain the model – e.g., one may have 10 concepts with high TCAV scores, but they may still be very insufficient in understanding the predictions. Besides, human-suggested exogenous concept data might even encode confirmation bias. The second caveat is that their saliency scores may fail to capture concepts that have non-linear relationships with the output due to first-order assumption. The concepts in Example 3.1 might not be retrieved by the TCAV score since XOR is not a linear relationship. Overall, our completeness score adds a valuable criterion to determine whether existing concept discovery methods are sufficient to explain the model completely, which complements previous works in concept discovery.

4.2 Our method

We propose a novel algorithm to obtain concepts that are *complete* to the model. We consider the case where each data point $\mathbf{x}^i$ has parts $\mathbf{x}^i_{1:T}$, as described above. We also assume that input data has spatial dependency, which can help learning coherent concepts. Thus, we encourage the closeness between the each concept and its nearest neighbors, which are parts of the data. It is aimed that the concepts would obtain consistent nearest neighbors that only occur in parts of the input, e.g. head of a lion or the grass in the background so that the concepts are pertained to certain spacial regions. By encouraging the closeness between the each concept and its nearest neighbors, we aim to obtain consistent nearest neighbors that may make sense to human. Lastly, we introduce an optimization of the variational lower bound of the log likelihood, which encourages the *completeness* of the discovered concepts.

We first assume that there is a probabilistic graphical model for the data generation process of $(\mathbf{x}, y)$, where $\mathbf{z}_t \rightarrow \mathbf{x}_t$ and $\mathbf{z}_{1:T} \rightarrow y$, such that each part of the data is generated by the concept assignment $\mathbf{z}_t$, and the overall concept assignment $\mathbf{z}_{1:T}$ determines the label y. Our goal is to find the underlying concepts $\mathbf{c}_{1:m}$ such that they follow the generation process while being coherent and complete. We show that by optimizing a variational lower bound of the log likelihood of the data, with an additional interpretability regularizer, we can achieve our goals.

Variational lower bound: To discover a set of interpretable and complete concepts, we first consider the variational lower bound of the log probability:

$$\begin{aligned}\log P(\mathbf{x}_{1:T}, y) &= \int \log P(\mathbf{x}_{1:T}, y, \mathbf{z}_{1:T}) dz \\ &= \int \log \frac{P(\mathbf{x}_{1:T}, y, \mathbf{z}_{1:T}) q(z)}{q(z)} dz \\ &\leqslant \sum_{t=1}^{T} \mathbb{E}_q[\log P(\mathbf{x}_t, \mathbf{z}_t)] + \mathbb{E}_q[\log P(y|\mathbf{z}_{1:T})] + H(q),\end{aligned} \tag{3}$$

where the expectation is taken with respect to a variational distribution, where we choose a factorized distribution $q(\mathbf{z}_{1:T}|\kappa_{1:T}) = \prod_{t=1}^{T} q(\mathbf{z}_t|\kappa_t)$, where each κ_t parametrizes a categorical distribution over K elements so that $\mathbb{E}_q[z_t] = \kappa_t$. We note that such a factorized distribution is generally adopted in previous works in topic modelling [Blei et al., 2003, Mcauliffe and Blei, 2008], where the independence assumption may not hold in practice. Correspondingly, we obtain the entropy term and $\mathbb{E}_q[\log P(\mathbf{x}_t, \mathbf{z}_t)]$ as:

$$H(q) = -\sum_{t=1}^{T} \sum_{k=1}^{m} \kappa_{t,k} \log(\kappa_{t,k}).$$

$$\mathbb{E}_q[\log P(\mathbf{x}_t, \mathbf{z}_t)] \;=\; \sum_{k=1}^{m} \kappa_{t,k} \log P(\mathbf{x}_t, \mathbf{z}_{t,k} = 1).$$

To estimate $\mathbb{E}_q[\log P(y|\mathbf{z}_{1:T})]$, we use

$$\mathbb{E}_q[\log P(y|\mathbf{z}_{1:T})] = \log P(y|\mathbb{E}_q[\mathbf{z}_{1:T}], h),$$

which is an approximation by bringing the expectation into the log to avoid additional sampling.

Learning concepts: To optimize the variational lower bound, we optimize $\kappa_{t,k}$ with respect to the first two terms in (3), and plug the resulting $\kappa_{t,k}$ into the third term in (3). The benefit of this optimization is that we get consistent $\kappa_{t,k}$ in both training and inference time (regardless whether the label y is given). By the first order condition we obtain $\kappa_{t,k}^{\text{new}} \propto P(\mathbf{x}_t, \mathbf{z}_{t,k} = 1)$, and thus $\kappa_{t,k}^{\text{new}} = P(\mathbf{z}_{t,k} = 1|\mathbf{x}_t) = \mathbb{E}[\mathbf{z}_{t,k}|\mathbf{x}_t]$.

The third term in (3) is then $\log P(y|\mathbb{E}[\mathbf{z}_{1:T}|\mathbf{x}_t], h)$. Cross entropy is often seen as a differential loss to optimize the accuracy, and by optimizing $\log P(y|\mathbb{E}[\mathbf{z}_{1:T}|\mathbf{x}_t], h)$ we are effectively optimizing the completeness score in Definition 4.2. An alternative interpretation on optimizing $P(y|\mathbb{E}[\mathbf{z}_{1:T}|\mathbf{x}_t], h)$ is that we assume $\mathbb{E}[\mathbf{z}_{1:T}|\mathbf{x}_{1:T}]$ is a sufficient statistic of the model, so that $P(y|\mathbb{E}[\mathbf{z}_{1:T}|\mathbf{x}_{1:T}], h) = P(y|\mathbf{x})$, which optimizes the log likelihood of the data.

By plugging in (2) into $\log P(y|\mathbb{E}[\mathbf{z}_{1:T}|\mathbf{x}_{1:T}], h)$, the optimization of (3) is then reduced to

$$\arg\max_{\mathbf{c}_{1:m}, l} \log P(y|h(l(\mathbb{E}[\mathbf{z}_{1:T}|\mathbf{x}_{1:T}]))), \tag{4}$$

which optimizes the surrogate loss of the completeness score of the concepts. We also design a regularizer to encourage the spacial dependency (and thus coherency) of concepts. Intuitively, we require that the top-K nearest neighbor training input patches of each concept to be sufficiently close to the concept, and different concepts are as different as possible.This formulation encourages the top-K nearest neighbors of the concepts would be coherent. K is a hyperparameter that is usually chosen based on domain knowledge of the desired frequency of concepts. In our results, we fix K to be half of the average class size in our experiments. When using batch update, we find that picking $k = \frac{1}{2} \times$ batchsize $\times$ average class ratio works well in our experiments, where average class ratio $=$ average instance of each class/total number of instances.

Now we introduce the regularizer terms. This term tries to maximize $\Phi(\mathbf{x}_t^i) \cdot \mathbf{c}_k$ while minimizing $\mathbf{c}_j \cdot \mathbf{c}_k$. $\Phi(\mathbf{x}_t^i) \cdot \mathbf{c}_k$ is the similarity between the t^{th} patch of the i^{th} example and $\mathbf{c}_j \cdot \mathbf{c}_k$ is the similarity between the j^{th} concept vector and the k^{th} concept vector. By averaging over all concepts, the final regularization term is

$$R(\mathbf{c}) = \lambda_1 \frac{\sum_{k=1}^{m} \sum_{\mathbf{x}_a^b \subseteq T_{\mathbf{c}_k}} \Phi(\mathbf{x}_a^b) \cdot \mathbf{c}_k}{mK} - \lambda_2 \frac{\sum_{j \neq k} \mathbf{c}_j \cdot \mathbf{c}_k}{m(m-1)},$$

where $T_{\mathbf{c}_k}$ is the set of top-K nearest neighbors of $\mathbf{c}_k$.

Putting together, the final optimization objective in training is

$$\arg\max_{\mathbf{c}_{1:m}, l} \log P(y|h(l(\mathbb{E}[\mathbf{z}_{1:T}|\mathbf{x}_{1:T}]))) + R(\mathbf{c}), \tag{5}$$

for which we use stochastic gradient descent to optimize.

4.3 ConceptSHAP: How Important is Each Concept?

Given a set of concept vectors $C_S = \{\mathbf{c}_1, \mathbf{c}_2, ...\mathbf{c}_m\}$ with a high completeness score, we would like to evaluate the importance of each individual concept, specifically, by quantifying how much each individual concept contributes to the final completeness score. Let $\mathbf{s}_i$ denote the importance score for concept $\mathbf{c}_i$, such that $\mathbf{s}_i$ quantifies how much of the completeness score $\eta(C_S)$ is contributed by $\mathbf{c}_i$. Motivated by its successful applications in quantifying attributes in what-if scenarios for complex systems, we adapt Shapley values [Shapley, 1988], to fairly assign the importance of each concept (which we call ConceptSHAP):

Definition 4.1. Given a set of concepts $C_S = \{\mathbf{c}_1, \mathbf{c}_2, ...\mathbf{c}_m\}$ and some completeness score η, we define the ConceptSHAP $\mathbf{s}_i$ for concept $\mathbf{c}_i$ as

$$\mathbf{s}_i(\eta) = \sum_{S \subseteq C_s \backslash \mathbf{c}_i} \frac{(m - |S| - 1)!|S|!}{m!} [\eta(S \cup \{\mathbf{c}_i\}) - \eta(S)],$$

The main benefit of using Shapley for importance attribution is that it uniquely satisfies a set of desired axioms, listed in the following proposition:

Proposition 4.2. *Given a set of concepts $C_S = \{\mathbf{c}_1, \mathbf{c}_2, ...\mathbf{c}_m\}$ and a completeness score η, and some importance score $\mathbf{s}_i$ for each concept $\mathbf{c}_i$ that depends on the completeness score η. $\mathbf{s}_i$ defined by conceptSHAP is the unique importance assignment that satisfy the following four axioms:*

- *Efficiency: The sum of all importance value should sum up to the total completeness score, $\sum_{i=1}^{m} \mathbf{s}_i(\eta) = \eta(C_S)$.*
- *Symmetry: For two concept that are equivalent s.t. $\eta(u \cup \{\mathbf{c}_i\}) = \eta(u \cup \{\mathbf{c}_j\})$ for every subset $u \subseteq C_S \backslash \{\mathbf{c}_i, \mathbf{c}_j\}$, $\mathbf{s}_i(\eta) = \mathbf{s}_j(\eta)$.*
- *Dummy: If $\eta(u \cup \{\mathbf{c}_i\}) = \eta(u)$ for every subset $u \subseteq C_S \backslash \{\mathbf{c}_i\}$, then $\mathbf{s}_i(\eta) = 0$.*
- *Additivity: If η and η' have importance value $\mathbf{s}(\eta)$ and $\mathbf{s}(\eta')$ respectively, then the importance value of the sum of two completeness score should be equal to the sum of the two importance values, i.e, $\mathbf{s}_i(\eta + \eta') = \mathbf{s}_i(\eta) + \mathbf{s}_i(\eta')$ for all i.*

The proof and the interpretation for these concepts are well discussed in [Shapley, 1988, Lundberg and Lee, 2017, Fujimoto et al., 2006].

Per-class saliency of concepts: So far, conceptSHAP measures the global contribution (i.e., contribution to completeness when all classes are considered). However, per-class saliency, how much concepts contribute to prediction of a particular class, might be informative in many cases. To obtain the concept importance score for each class, we first define the completeness score with respect to the class by considering data points that only belongs to it, which is formalized as:

Definition 4.2. Given a prediction model $f(\mathbf{x}) = h(\phi(\mathbf{x}))$, a set of concept vectors $\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_m$ that lie in the feature subspace in $\phi(\cdot)$. We then define the completeness score $\eta_j(\mathbf{c}_1, ..., \mathbf{c}_m)$ for class j as:

$$\frac{\mathbb{E}_{\mathbf{x}, y \sim V_j}[\mathbb{1}[y = \arg\max_{y'} P(y' | \mathbb{E}[z_{1:T}], h)]] - R_j}{\mathbb{E}_{\mathbf{x}, y \sim V}[\mathbb{1}[y = \arg\max_{y'} P(y' | \mathbf{x}_{1:T}, f)]] - R}, \tag{6}$$

where V_j is the set of validation data where ground truth label is j and R_j is the random accuracy for data in class j. Given the completeness score for a specific class, we define the ConceptSHAP for concept i with respect to class j as:

Definition 4.3. Given a prediction model $f(\mathbf{x})$, a set of concept vectors $\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_m$ that lie in the feature subspace in $\phi(\cdot)$. We can define the ConceptSHAP for concept i with respect to class j as: $\mathbf{s}_{i,j}(\eta) = \mathbf{s}_i(\eta_j)$.

For each class j, we may select the concepts with the highest conceptSHAP score with respect to class j. We note that $\sum_j \frac{|V_j|}{|V|} \eta_j = \eta$ and thus with the additivity axiom, $\sum_j \frac{|V_j|}{|V|} \mathbf{s}_{i,j}(\eta_j) = \mathbf{s}_i(\eta)$.

Note that one can generalize the above per class importance definition using other feature based attribution methods besides Shapley values, without the guarantees of satisfying the proposed axioms.

5 Experiments

In this section, we demonstrate our method both on a synthetic dataset, where we have ground truth concept importance, and a real-world image dataset.

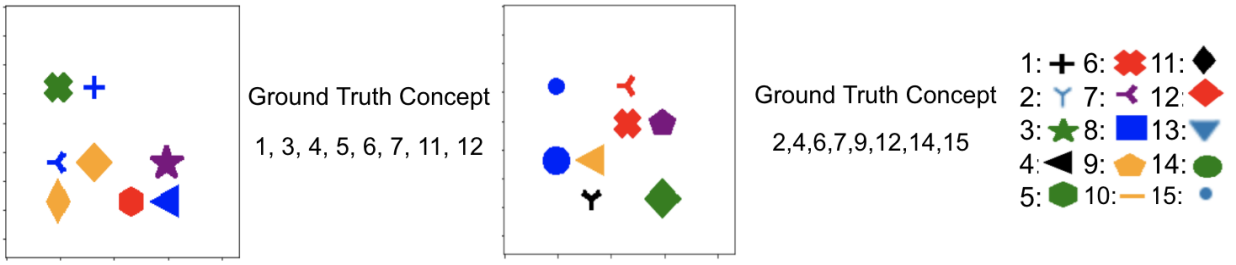

Figure 1: Two random training images and the corresponding ground truth concepts, along with the legend of ground truth concept shapes – each object shape in the image corresponds to a ground truth concept (with random color and location), and the label depends solely on ground truth shape 1 to 5.

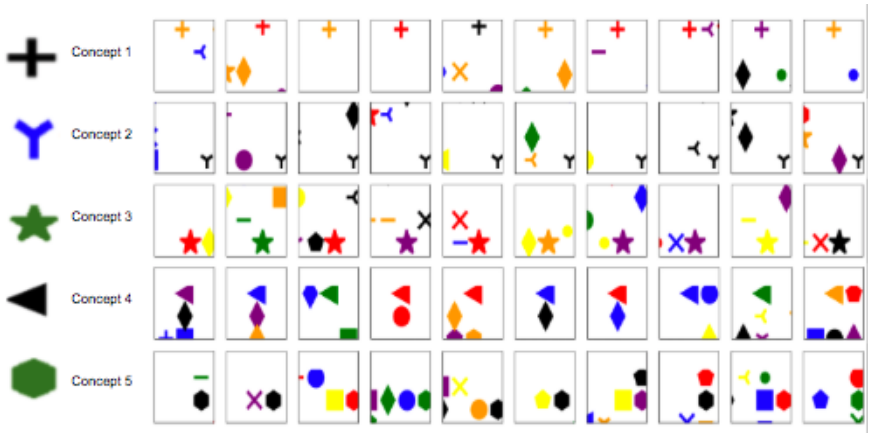

Figure 2: Top nearest neighbors (each neighbor corresponds to a part of the full image) for each discovered concepts. The ground truth concepts are in the left most column.

5.1 Synthetic data with ground truth concepts

Setting: We construct a synthetic image dataset with known and complete concepts, to evaluate how accurately the proposed concept discovery algorithm can extract the ground truth concepts. In this dataset, each image contains at most 15 shapes (shown in Fig. 1), and only 5 of them are relevant for the ground truth class, by construction. For each sample $\mathbf{x}^i$, $\mathbf{z}^i_j$ is a binary variable which represents whether $\mathbf{x}^i$ contains shape j. $\mathbf{z}^i_{1:15}$ is a 15-dimensional binary variable with elements independently sampled from Bernoulli distribution with $p = 0.5$. We construct a 15-dimensional multi-label target for each sample, where the target of sample i, y^i is a function that depends only on $\mathbf{z}^i_{1:5}$, which represents whether the first 5 shape exists in $\mathbf{x}^i$. For example, $y_1 =\sim (\mathbf{z_1} \cdot \mathbf{z_3}) + \mathbf{z_4}, y_2 = \mathbf{z_2} + \mathbf{z_3} + \mathbf{z_4}, y_3 = \mathbf{z_2} \cdot \mathbf{z_3} + \mathbf{z_4} \cdot \mathbf{z_5}$, where $\sim$ denotes logical Not (details are in Appendix). We construct 48k training samples and 12k evaluation samples to train a convolutional neural network with 5 layers, which achieves 0.999 accuracy. We take the last convolution layer as the feature layer $\phi(\mathbf{x})$.

Evaluation metrics: Given the existence of each ground truth shape $\mathbf{z}^i_{1:5}$ in each sample $\mathbf{x}^i$, we can evaluate how closely the discovered concept vectors $\mathbf{c}_{1:m}$ align with the actual ground truth shapes 1 to 5. Our evaluation assumes that if $\mathbf{c}_k$ corresponds to some shape v, then the parts of input that contain the shape v and the parts of input that does not contain ground truth shape v can be linearly separated by $\mathbf{c}_k$. That is, $\mathbf{c}_k \cdot \mathbf{x}_a > \mathbf{c}_k \cdot \mathbf{x}_b$ or $\mathbf{c}_k \cdot \mathbf{x}_a > \mathbf{c}_k \cdot \mathbf{x}_b$ for all $\mathbf{x}_a$ that contains shape v and all $\mathbf{x}_b$ that does not contain shape v. Without loss of generality, we assume $\mathbf{c}_k \cdot \mathbf{x}_a > \mathbf{c}_k \cdot \mathbf{x}_b$ if $\mathbf{x}_a$ contains shape v and $\mathbf{x}^d_c$ does not contain shape v for notation simplicity, and check $\mathbf{c}_k$ and $-\mathbf{c}_k$ for each discovered concepts. Following this assumption, $\max_{t=1}^T \mathbf{c}_k \cdot \mathbf{x}^i_t > \max_{t=1}^T \mathbf{c}_k \cdot \mathbf{x}^j_t$ for all i, j such that $\mathbf{z}^i_v = 1$ and $\mathbf{z}^j_v = 0$, since at least one part of $\mathbf{x}^b_t$ should contain the ground truth shape v. Therefore, to evaluate how well $\mathbf{c}_k$ corresponds to shape v, we measure the accuracy of using $\mathbb{1}[\max_{t=1}^T \mathbf{c}_k \cdot \mathbf{x}^i_t > \text{const}]$ to classify $\mathbf{z}^i_v$. More formally, we define the matching score between concept $\mathbf{c}_k$ to the shape v as:

$$\text{Match}(\mathbf{c}_k, \mathbf{z}_v) = \mathbb{E}_{\mathbf{x}^i \sim V}[\mathbb{1}[\max_{t \subseteq [1,T]} \mathbf{c}_k \cdot \mathbf{x}^i_t > e] = \mathbf{z}^i_v],$$

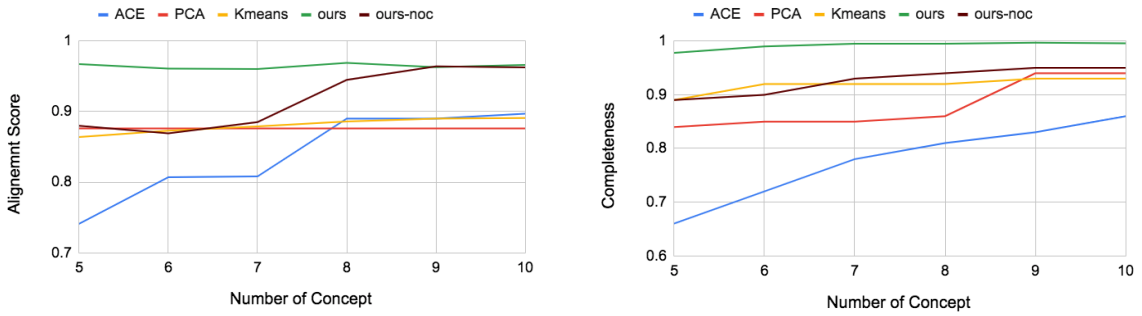

Figure 3: Alignment (left) and completeness (right) scores versus different number of discovered concepts m for all concept discovery methods in the synthetic dataset. Our proposed concept discovery method outperforms other methods. Ours-noc refers to our method without the completeness score objective as an ablation study.

Table 1: The average number of correctly answered concepts by users based on 10 nearest neighbor patches.

ACE	ACE-SP	PCA	k-means	**Ours**
3.0	2.75	4.0	3.75	**5.0**

where e is some constant. We then evaluate how well the set of discovered concepts $\mathbf{c}_{1:m}$ aligns with shapes 1 to 5:

$$\text{Alignment}(\mathbf{c}_{1:m}, \mathbf{z}_{1:5}) = \max_{P \in [1,m]^m} \frac{1}{5} \sum_{j=1}^{5} \text{Match}(\mathbf{c}_{P[j]}, \mathbf{z}_j),$$

which measures the best average matching accuracy by assigning the best concept vector to differentiate each shape. For each concept vector $\mathbf{c}_j$, we test $\mathbf{c}_j$ and $-\mathbf{c}_j$ and choose the direction that leads to the highest alignment score.

Results: We compare our methods to ACE, k-means, PCA, and ours-noc. For k-means clustering and PCA, we take the embedding of the patch as input to be consistent to our method. For ACE, we implement a version which replaces the superpixels by patches and another version that takes superpixels as input, which we call ACE and ACE-SP respectively. We do not calculate the completeness score and Alignment score of ACE-SP since the method do not operate on patches and thus is unfair to compare with others[4]. Ours-noc means our method without the completeness score objective in (5), as a form of comparison. We show results for the alignment and completeness scores when 5 to 10 concepts are discovered (i.e. m is set from 5 to 10). Since there are 5 ground truth shapes, a desired concept discovery should be able to discover all when $m = 5$. Our concept discovery method consistently achieves higher alignment and completeness scores compared to other concept discovery methods in Fig. 3.

We also see the benefit of including the completeness score in (5) by comparing ours to ours-noc when m is 5-8. The completeness score objective is helpful in retrieving the key concepts of interest, especially when the number of retrieved concept is smaller. Fig.2 shows the top-10 nearest neighbors for each concept $\mathbf{c}_k$ of our concept discovery method based on the dot product score $\langle \mathbf{c}_k, \Phi(\mathbf{x}_a)^b \rangle$. Note that top-10 nearest neighbors for other concept discovery methods are in Appendix. All nearest neighbors contains the specific shape that corresponds to the ground-truth shapes 1 to 5. For example, all nearest neighbors of concept 1 contains the ground truth shape 1, which is a cross as listed in Figure 1.

Human evaluations: We conduct a user-study with 20 users to evaluate the nearest neighbor samples of a few concept discovery methods. At each question, a user sees 10 nearest neighbor images of each discovered concept vector (as shown in the right of Fig. 2), and is asked to choose the most common and coherent shape out of the 15 shapes based on the 10 nearest neighbors. We evaluate the results for our method, k-means clustering, PCA, ACE, and ACE-SP when 5 concepts are retrieved (i.e. $m = 5$). Each user is tested on two randomly chosen methods, and thus each method is tested on 8 users. We report the average number of correct answers for each method in Table 1, and our method outperforms other methods (on the average number of the correct answers are reported). We put an example question of the user-study in Appendix.

[4]This in fact, leads to much lower scores.

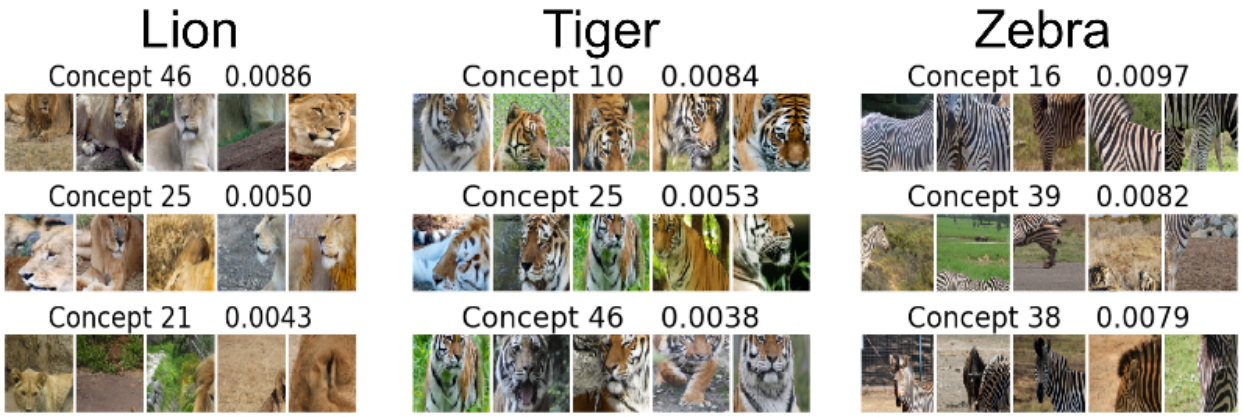

Figure 4: Concept examples with the samples that are the nearest to concept vectors in the activation space. The top concepts for each class are based on per-class ConceptSHAP score, which is listed above the images.

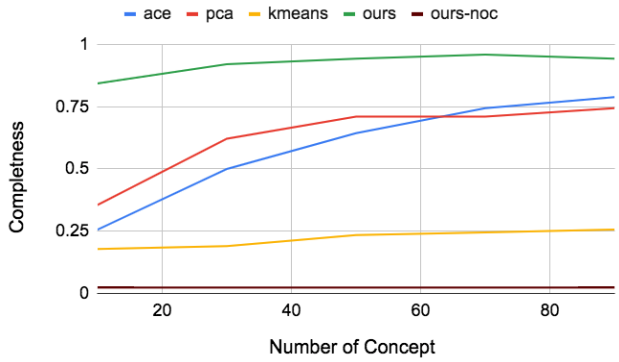

Figure 5: The completeness score for different number of concepts retrieved on AwA dataset.

5.2 Image classification

Setting and metrics: We next perform experiments on Animals with Attribute (AwA) [Lampert et al., 2009] that contains 50 animal classes. We use 26905 images as training data and 2965 images as evaluation data. We use a Inception-V3 model pre-trained on Imagenet [Szegedy et al., 2016] to achieve 0.9 test accuracy. We then apply our concept discovery algorithm to obtain 70 concepts ($m = 70$). We conduct ad-hoc duplicate concept removal, by removing one concept vector if there are two vectors where the dot product is over 0.95. This gives us a total of 53 concepts. We then calculate the ConceptSHAP and per class saliency score for each concept and each class. For each class, the top concepts based on the conceptSHAP are the most important concepts to classify this class, as shown in Fig.4. While ConceptSHAP is useful in capturing the sufficiency of concepts for prediction, sometimes we may want to show examples. We propose to measure the quality of the nearest neighbors explanations by the average dot product between the nearest-neighbor patches that belongs to the class and the concept vector. In other words, the quality of the nearest neighbors explanations is simply the first term in $R(\mathbf{c})$, which we call $R_1(\mathbf{c}) = \sum_{k=1}^{m} \sum_{\mathbf{x}_a^b \subseteq T_{\mathbf{c}_k}} \langle \Phi(\mathbf{x}_a^b), \mathbf{c}_k \rangle$, where the top-K set is limited to image patches in the class of interest. When the nearest neighbor set contains patches of the same original image, we only show the patch with the highest similarity to the concept to increase the diversity,

Results: We show the top concepts (ranked by concept SHAP value) of 3 randomly chosen classes whose $R_1(\mathbf{c})$ is above 0.8 in Fig.4 (full results are in Appendix). Interestingly, we find that there exists overlapping concepts when explaining different classes. For example, concept 46 is important for both the class Lion and Tiger, whose nearest neighbors shows visually similar face shape of tigers and lions, with rectangular nose and similar mouth. Similarly concept 25 is also important both the class tiger and lion, which shows the triangular head and side faces of tiger and lions. While tiger and lion class share concepts, the most salient concept with respect to conceptSHAP for tiger is concept 10 - striped-like pattern - a distinguishable feature from lion. Similarly, the most salient concept for zebra is

their unique stripe patterns. Figure 5 shows that our method achieves the highest completeness of all methods for all 50 classes.

Failure cases and limitations: One of the most significant limitations of our method is that while we are able to infer many concepts by human eyes, the DNN might be using concepts that we cannot quite put a name on. For example, concept 39 contains segment of images with backgrounds that contains mountain, ocean, and rocks. A possible interpretation is that it focuses on ripple-like texture that occurs in both ocean and grass, that it is difficult to grasp the concise definition for. As future work, it would be interesting to explore combining our method with a language model to automatically 'name' each discovered concepts. Another limitation of our method is that certain class do not have salient concepts that satisfy $R_1(\mathbf{c}) > 0.8$, which happens in 4 out of the 50 classes. This introduces additional challenge of communicating many concepts with distributed importance.

6 Conclusions

Concept-based explanations are crucial to understand how DNNs make decisions. In this paper, we study concept-based explainability in a systematic framework. First, we define the notion of completeness, which quantifies how sufficient a particular set of concepts is in explaining the model's behavior. Next, we study additional constraints to ensure the interpretability of discovered concept. Through experiments on synthetic and real-world image data, we demonstrate that our method is effective in finding concepts that are complete and interpretable. Although our work focuses on post-hoc explainability of pre-trained DNNs, joint training with our proposed objective function is possible to train inherently-interpretable DNNs. An interesting future direction is exploring the benefits joint learning of the concepts along with the model, for better interpretability.

References

Marco Ancona, Enea Ceolini, Cengiz Öztireli, and Markus Gross. Towards better understanding of gradient-based attribution methods for deep neural networks. *arXiv preprint arXiv:1711.06104*, 2017.

André Araujo, Wade Norris, and Jack Sim. Computing receptive fields of convolutional neural networks. *Distill*, 4(11): e21, 2019.

Sercan Ömer Arik and Tomas Pfister. Attention-based prototypical learning towards interpretable, confident and robust deep neural networks. *arXiv:1902.06292*, 2019.

Sharon Lee Armstrong, Lila R. Gleitman, and Henry Gleitman. What some concepts might not be. *Cognition*, 13(3): 263 – 308, 1983.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *JMLR*, 3(Jan):993–1022, 2003.

Diane Bouchacourt and Ludovic Denoyer. Educe: Explaining model decisions through unsupervised concepts extraction. *arXiv preprint arXiv:1905.11852*, 2019.

Tsung-Han Chan, Kui Jia, Shenghua Gao, Jiwen Lu, Zinan Zeng, and Yi Ma. Pcanet: A simple deep learning baseline for image classification? *IEEE transactions on image processing*, 24(12):5017–5032, 2015.

Jianbo Chen, Le Song, Martin J. Wainwright, and Michael I. Jordan. L-shapley and c-shapley: Efficient model interpretation for structured data. *arXiv:1808.02610*, 2018.

Jan Chorowski, Ron J Weiss, Samy Bengio, and Aäron van den Oord. Unsupervised speech representation learning using wavenet autoencoders. *arXiv preprint arXiv:1901.08810*, 2019.

Katsushige Fujimoto, Ivan Kojadinovic, and Jean-Luc Marichal. Axiomatic characterizations of probabilistic and cardinal-probabilistic interaction indices. *Games and Economic Behavior*, 55(1):72–99, 2006.

Amirata Ghorbani, James Wexler, James Zou, and Been Kim. Towards automatic concept-based explanations. *NeurIPS*, 2019.

Leilani H Gilpin, David Bau, Ben Z Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on data science and advanced analytics*. IEEE, 2018.

Rajiv Khanna, Been Kim, Joydeep Ghosh, and Sanmi Koyejo. Interpreting black box predictions using fisher kernels. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 3382–3390, 2019.

Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *NIPS*, 2016.

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 *ICML*, 2018.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *ICML*, 2017.

Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*. IEEE, 2009.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. *arXiv preprint arXiv:1811.12359*, 2018.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *NIPS*, 2017.

Jon D Mcauliffe and David M Blei. Supervised topic models. In *NIPS*, 2008.

N. Narodytska and S. Kasiviswanathan. Simple black-box adversarial attacks on deep neural networks. In *CVPR Workshops*, 2017.

Alec Radford, Rafal Józefowicz, and Ilya Sutskever. Learning to generate reviews and discovering sentiment. *arXiv:1704.01444*, 2017.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *KDD*. ACM, 2016.

Wojciech Samek, Alexander Binder, Grégoire Montavon, Sebastian Lapuschkin, and Klaus-Robert Müller. Evaluating the visualization of what a deep neural network has learned. *IEEE transactions on neural networks and learning systems*, 28(11):2660–2673, 2016.

Lloyd S. Shapley. *A value for n-person games*, page 31–40. 1988.

Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas, and Martin Wattenberg. Smoothgrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825*, 2017.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

Joshua Brett Tenenbaum. *A Bayesian framework for concept learning*. PhD thesis, Massachusetts Institute of Technology, 1999.

Fan Yang, Mengnan Du, and Xia Hu. Evaluating explanation without ground truth in interpretable machine learning. *arXiv preprint arXiv:1907.06831*, 2019.

Chih-Kuan Yeh, Joon Kim, Ian En-Hsu Yen, and Pradeep K Ravikumar. Representer point selection for explaining deep neural networks. In *NIPS*, 2018.

Chih-Kuan Yeh, Cheng-Yu Hsieh, Arun Sai Suggala, David I. Inouye, and Pradeep Ravikumar. On the (in)fidelity and sensitivity of explanations. In *NeurIPS*, 2019.

Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In *ECCV*, 2018.

Appendix A Proof

Proof of Proposition 4.1

Proposition A.1. *When h is an isometry function that maps from $(\Phi(\cdot), \|\cdot\|_F) \rightarrow (f(\cdot), \|\cdot\|_F)$, and additionally $f(\mathbf{x}_i) = y_i,\ \forall(\mathbf{x}_i, y_i) \in V$ (i.e. the loss is minimized), and also assume $T = 1$, $\mathbb{E}[z] = \langle\phi(\mathbf{x}), \mathbf{c}\rangle$, and l is a linear function, the first m PCA vectors maximizes the L2 surrogate of η.*

Proof. By the basic properties of PCA, the first m PCA vectors (principal components) minimize the reconstruction ℓ_2 error. Define the concatenation of the m PCA vectors as a matrix $\mathbf{p}$ and $\|\cdot\|$ as the ℓ_2 norm, and define $\text{proj}(\phi(\mathbf{x}, \mathbf{p}))$ as the projection of $\mathbf{x}$ onto the span of $\mathbf{p}$, the basic properties of PCA is equivalent to that for all $\mathbf{c} = [\mathbf{c}_1\ \mathbf{c}_2 \dots\ \mathbf{c}_m]$,

$$\sum_{\mathbf{x}\subseteq V_X} \|\text{proj}(\phi(\mathbf{x}), \mathbf{p}) - \phi(\mathbf{x})\|_F^2 \leqslant \sum_{\mathbf{x}\subseteq V_X} \|\text{proj}(\phi(\mathbf{x}), \mathbf{c}) - \phi(\mathbf{x})\|_F^2.$$

By the isometry of h, we have

$$\sum_{\mathbf{x}\subseteq V_X} \|h(\text{proj}(\phi(\mathbf{x}), \mathbf{p})) - h(\phi(\mathbf{x}))\|_F^2 \leqslant \sum_{\mathbf{x}\subseteq V_X} \|h(\text{proj}(\phi(\mathbf{x}), \mathbf{c})) - h(\phi(\mathbf{x}))\|_F^2,$$

and since $f(\mathbf{x})$ is equal to Y, we can rewrite to

$$\sum_{\mathbf{x},y\subseteq V} \|h(\text{proj}(\phi(\mathbf{x}), \mathbf{p})) - \mathbb{1}[y]\|_F^2 \leqslant \sum_{\mathbf{x},y\subseteq V} \|h(\text{proj}(\phi(\mathbf{x}), \mathbf{c})) - \mathbb{1}[y]\|_F^2. \tag{7}$$

We note that under the assumptions, $\mathbb{E}[\mathbf{z}|\mathbf{x}] = \phi(\mathbf{x})\mathbf{c}$, and thus the reconstruction layer l can be written as

$$\begin{aligned} l &= \arg\max_l \sum_{\mathbf{x},y\subseteq V} \|\mathbb{1}[y] - h(l(\mathbb{E}[\mathbf{z}|\mathbf{x}]))\|_F^2 \\ &= \arg\max_l \sum_{\mathbf{x},y\subseteq V} \|\mathbb{1}[y] - h(l(\phi(\mathbf{x})\mathbf{c}))\|_F^2 \\ &= \arg\max_l \sum_{\mathbf{x}\subseteq V_x} \|\phi(\mathbf{x}) - l(\phi(\mathbf{x})\mathbf{c})\|_F^2, \end{aligned} \tag{8}$$

By definition, $\sum_{\mathbf{x}\subseteq V_x} \|\phi(\mathbf{x}) - l(\phi(\mathbf{x})\mathbf{c})\|_F^2$ is minimized by the projection, and thus $l(\phi(\mathbf{x})\mathbf{c}) = \text{proj}(\phi(\mathbf{x}), \mathbf{c})$.

And thus, (7) can be written as:

$$\sum_{\mathbf{x},y\subseteq V} \|h(l(\phi(\mathbf{x})\mathbf{p})) - \mathbb{1}[y]\|_F^2 \leqslant \sum_{\mathbf{x},y\subseteq V} \|h(l(\phi(\mathbf{x})\mathbf{c})) - \mathbb{1}[y]\|_F^2.$$

and subsequently get that for any $\mathbf{c}$

$$\frac{\mathbb{E}_{\mathbf{x},y\sim V}[\|\mathbb{1}[y] - P(y'|\mathbb{E}[z_{1:T}], h, \mathbf{p})\|_F^2] - R}{\mathbb{E}_{\mathbf{x},y\sim V}[\|\mathbb{1}[y] - P(\mathbf{x}_{1:T}, f)\|_F^2] - R} \geqslant \frac{\mathbb{E}_{\mathbf{x},y\sim V}[\|\mathbb{1}[y] - P(y'|\mathbb{E}[z_{1:T}], h, \mathbf{c})\|_F^2] - R}{\mathbb{E}_{\mathbf{x},y\sim V}[\|\mathbb{1}[y] - P(\mathbf{x}_{1:T}, f)\|_F^2] - R}.$$

□

Thus, PCA vectors maximize the L2 surrogate of the completeness score. We emphasize that Proposition 4.1 has several assumptions that may not be practical. However, the proposition is only meant to show that PCA optimizes our definition of completeness under a very stringent condition, as the key idea of completeness and PCA are both to prevent information loss through dimension reduction.

Appendix B Additional Experiments Results and Settings

Creation of the Toy Example The complete list of the target y is $y_1 = \sim (\mathbf{z_1} \cdot \mathbf{z_3}) + \mathbf{z_4}, y_2 = \mathbf{z_2} + \mathbf{z_3} + \mathbf{z_4}, y_3 = \mathbf{z_2} \cdot \mathbf{z_3} + \mathbf{z_4} \cdot \mathbf{z_5}, y_4 = \mathbf{z_2}$ XOR $\mathbf{z_3}, y_5 = \mathbf{z_2} + \mathbf{z_5}, y_6 = \sim (\mathbf{z_1} + \mathbf{z_4}) + \mathbf{z_5}, y_7 = (\mathbf{z_2} \cdot \mathbf{z_3})$ XOR $\mathbf{z_5}, y_8 = \mathbf{z_1} \cdot \mathbf{z_5} + \mathbf{z_2}, y_9 = \mathbf{z_3}, y_{10} = (\mathbf{z_1} \cdot \mathbf{z_2})$ XOR $\mathbf{z_4}, y_{11} = \sim (\mathbf{z_3} + \mathbf{z_5}), y_{12} = \mathbf{z_1} + \mathbf{z_4} + \mathbf{z_5}, y_{13} = \mathbf{z_2}$ XOR $\mathbf{z_3}, y_{14} = \sim (\mathbf{z_1} \cdot \mathbf{z_5} + \mathbf{z_4}), y_{15} = \mathbf{z_4}$ XOR $\mathbf{z_5}$.

We create the dataset in matplotlib, where the color of each shape is sampled independently from green, red, blue, black, orange, purple, yellow, and the location is sampled randomly with the constraint that different shapes do not coincide with each other.

Hyper-parameter Sensitivity We set $\lambda_1 = \lambda_2 = 1.0, \beta = 0.2$ for the toy dataset. We show the completeness score for varying $\lambda_1, \lambda_2, \beta$ in Figure 6,7,8 (when varying λ_1, we fix $\lambda_2 = 1.0$, and $\beta = 0.2$.) We see that both the completeness and alignment score are above 0.9 when λ_1 and λ_2 are in the range of $[0.2, 2.0]$, and β is in the range of $[0, 0.3]$, and thus our method outperforms all baselines with a wide range of hyper-parameters. Therefore, our method is not sensitive to the hyper-parameter in the toy dataset. We set $\lambda_1 = \lambda_2 = 10.0, \beta = 0$ for AwA dataset since the optimization becomes more difficult with a deeper neural network, and thus we increase the regularizer strength to ensure interpretability. The completeness is above 0.9 when λ_1 and λ_2 are set in the range of $[2, 20]$. Overall, our method is not too sensitive to the selection of hyper-parameter.

Additional Nearest Neighbors for toy example We show 10 nearest neighbors for each concept obtained by our methods and baseline methods in the toy example in Figure 13. The 10 nearest neighbors for each concept obtained by different methods is used to perform the user study, to test if the nearest neighbors allow human to retrieve the correct ground truth concepts for each method.

User Study Setting For the user study, we set $m = 5$ (i.e. 5 discovered concepts) for all compared methods. For each discovered concept, an user is asked to find the most common and coherent shape given the top 10 nearest neighbors. An example question is shown in Figure 14. Each user is given 10 questions, which correspond to the nearest neighbors of the discovered concepts for two random methods. (each method has 5 discovered concepts, and thus two methods have 10 discovered concepts in total). There are 20 users in total, and thus each method is tested on 8 users. For each method, we report the average number of correct answers chosen by the users. For example, if an user chooses shape 1,2,5,7,5, then the number of the correct answers chosen by the user will be 3 (since 1,2,5 are the ground truth shape obtained by the user). We average the correct answers chosen by 8 users for each method to obtain the "average number of correct answers chosen by users".

Implementation Details For calculating ConceptSHAP, we use the method in kernelSHAP [Lundberg and Lee, 2017] to calculate the Shapley values efficiently by regression. For ACE in toy example, we set the number of cluster to be 15, and choose the concepts based on TCAV score. For ACE in toy example, we set the number of clusters to be 150, and choose the concepts based on TCAV score. For PCA, we return the top m principle components when the number of discovered concepts is m. For k-means, we set the cluster size to be m when the number of discovered concepts is m, and return the cluster mean as the discovered concepts.

Additional Nearest Neighbors for AwA We show addition nearest neighbors if the top concepts in AwA for all 50 classes from Figure 15 to Figure 23. For each class, the 3 concepts with the highest ConceptSHAP respect to the class with $R_1(\mathbf{c})$ above 0.8 is shown, along with the ConceptSHAP score with respect to the class. We see that many important concepts are shared between different classes, where most of them are semantically meaningful. To list some examples, concept 7 corresponds to the concept of grass, concept 33 shows a specific kind of wolf-like face (which has two different colors on the face), concept 27 corresponds to the sky/ocean view, concept 25 shows a side face that is shared among many animals, concept 46 shows a front face of cat-like animals, concept 21 shows sandy/ wilderness texture of the background, concept 38 shows gray back ground that looks like asphalt road, concept 43 shows similar ears of several animals, concept 31 shows furry/ rough texture with a plain background.

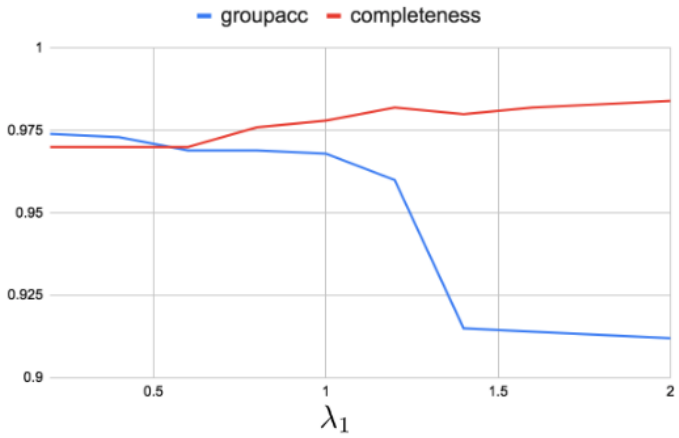

Figure 6: Completeness score and Alignment score for different hyper-parameter λ_1.

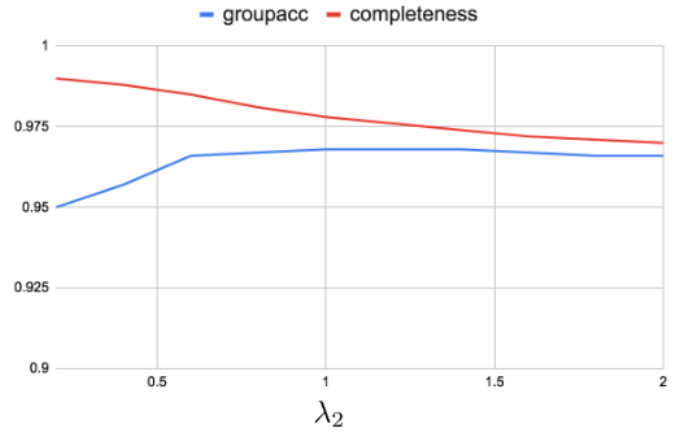

Figure 7: Completeness score and Alignment score for different hyper-parameter λ_2.

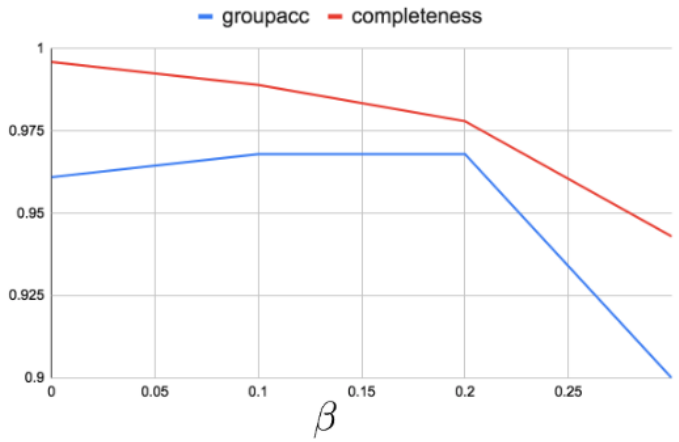

Figure 8: Completeness score and Alignment score for different hyper-parameter β.

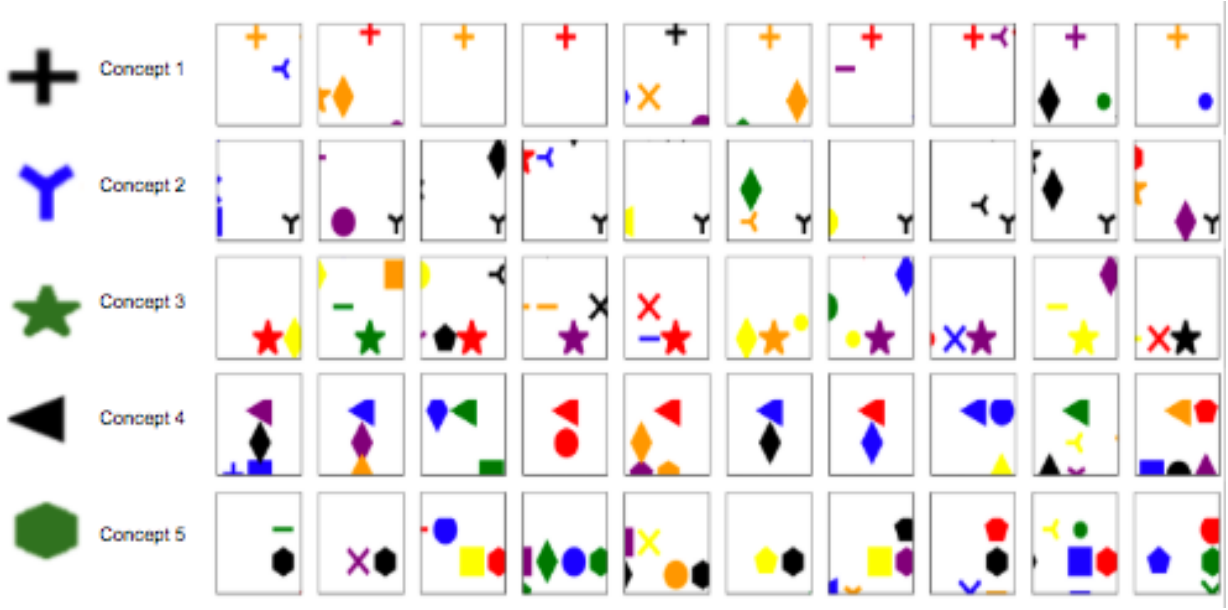

Figure 9: (Larger Version) Nearest Neighbors for each concept obtained in the toy example.

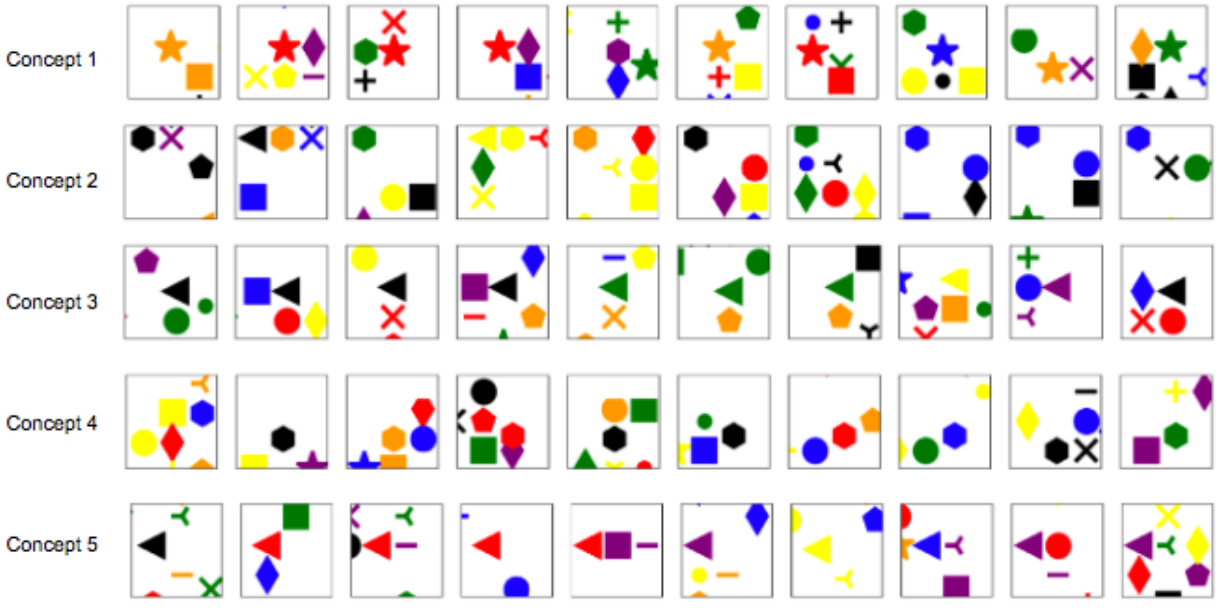

Figure 10: (Larger Version) Nearest Neighbors for each concept for ACE obtained in the toy example.

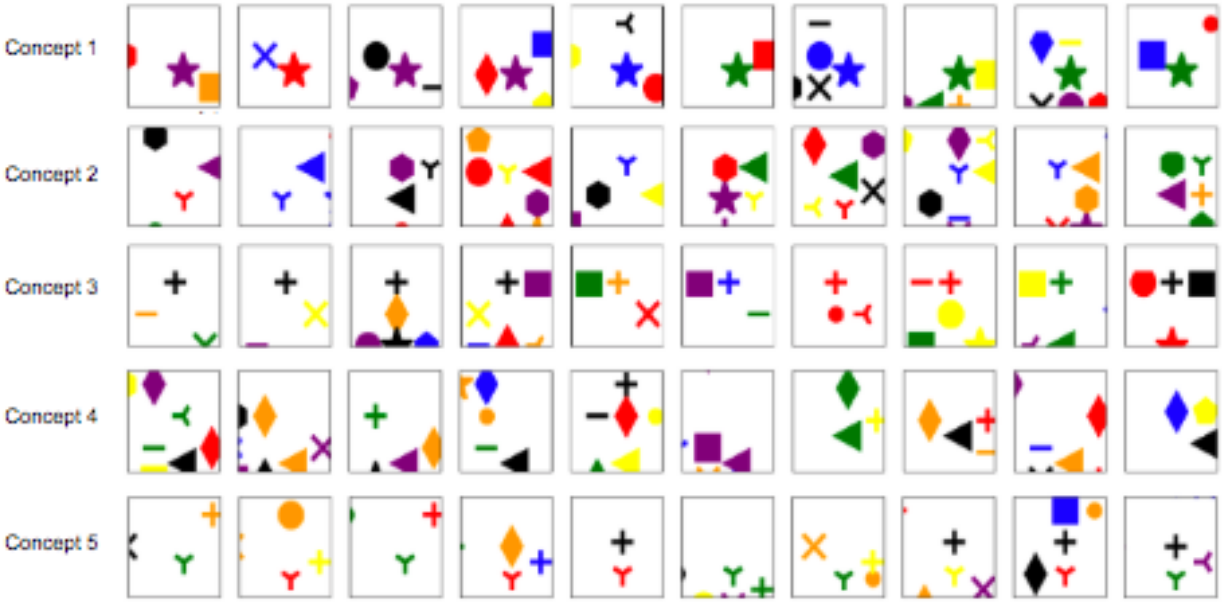

Figure 11: (Larger Version) Nearest Neighbors for each concept for PCA obtained in the toy example.

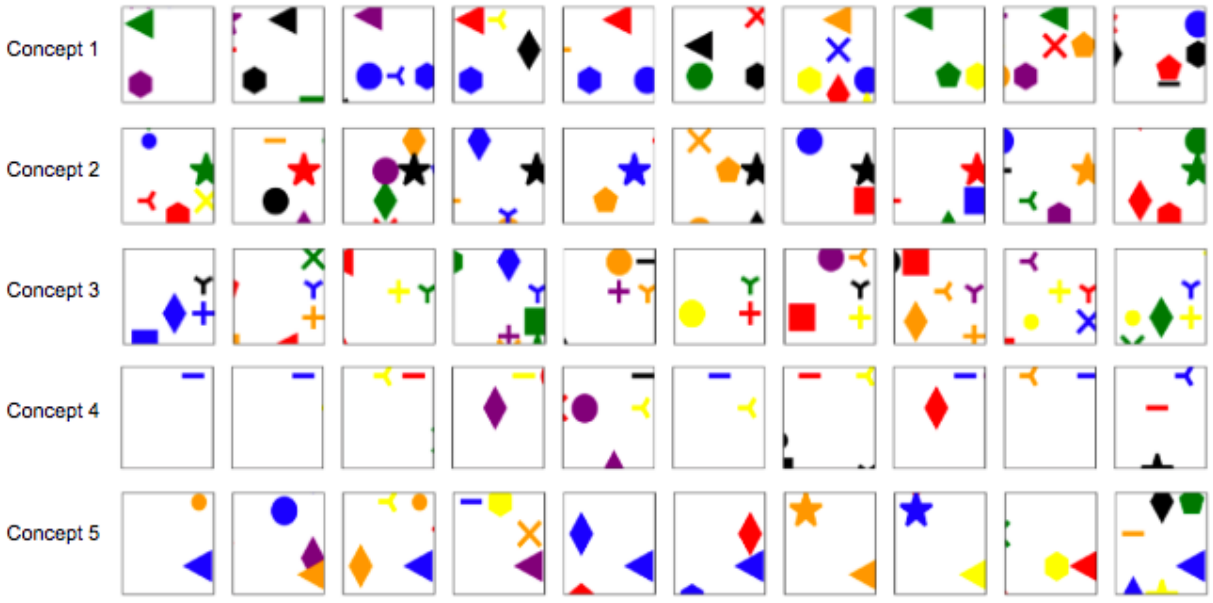

Figure 12: (Larger Version) Nearest Neighbors for each concept for Kmeans obtained in the toy example.

Figure 13: (Larger Version) Nearest Neighbors for each concept for ACE-SP obtained in the toy example.

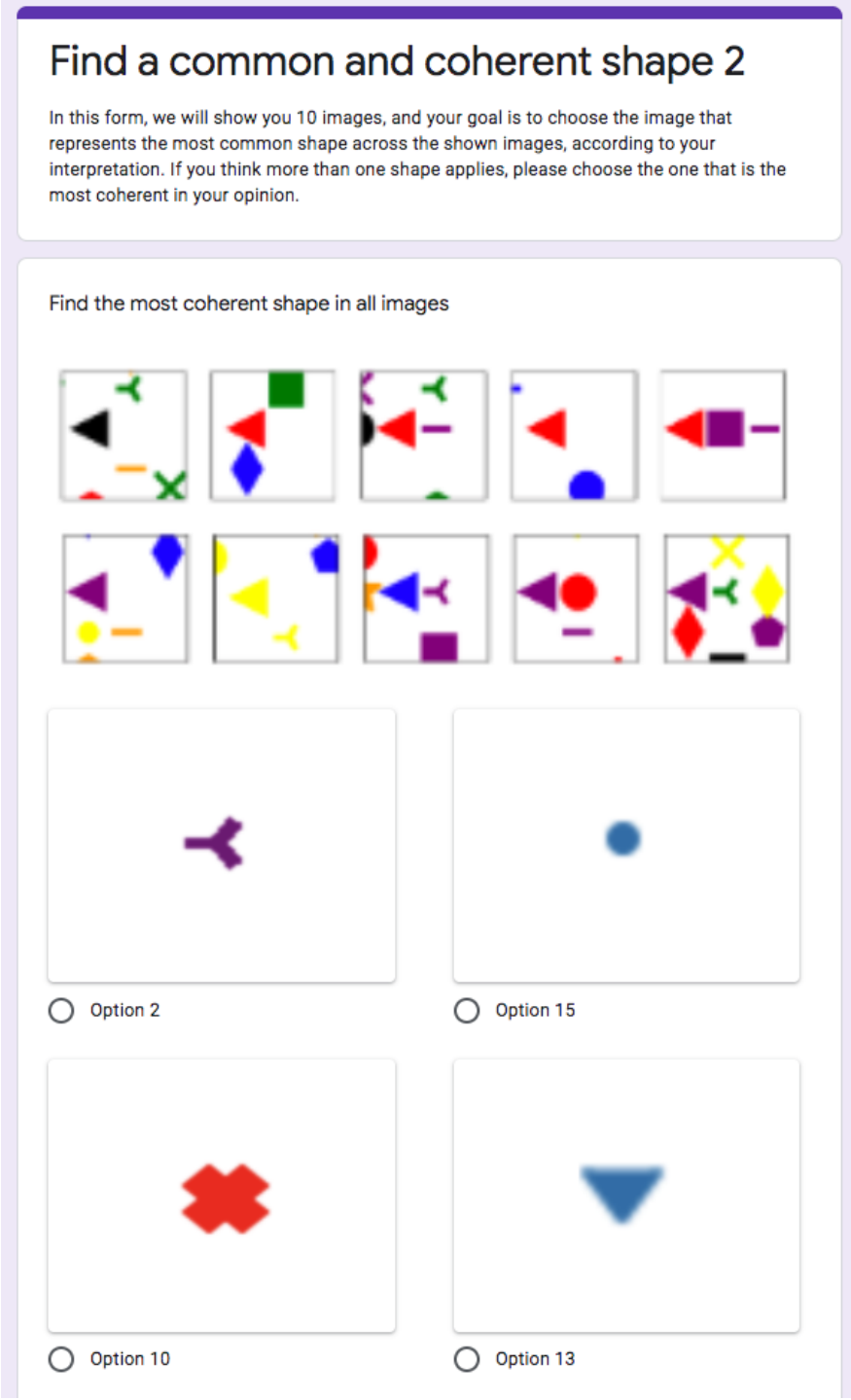

Figure 14: An example question of a screenshot of the human study.

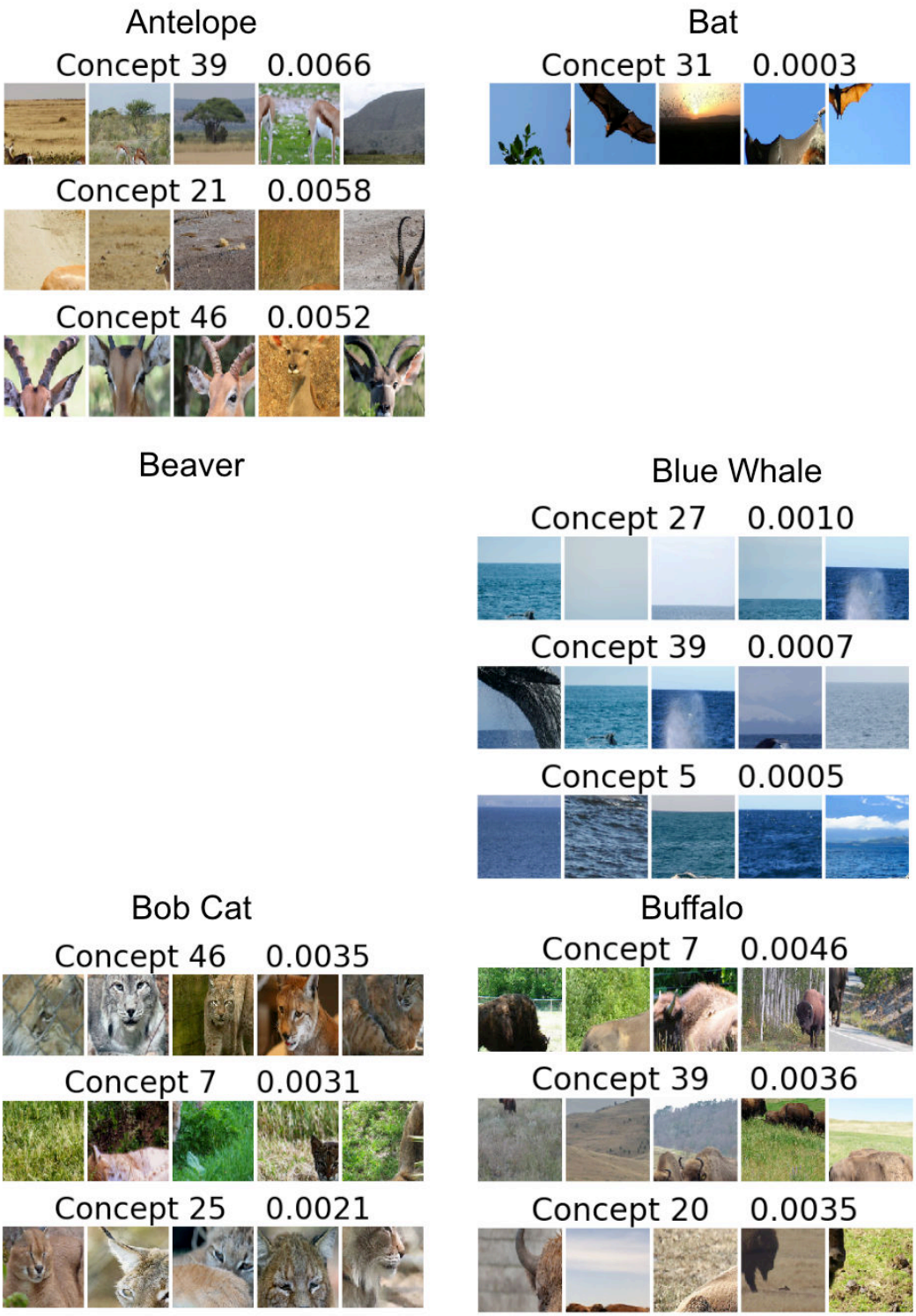

Figure 15: (Larger Version) Nearest Neighbors for each concept obtained in AwA.

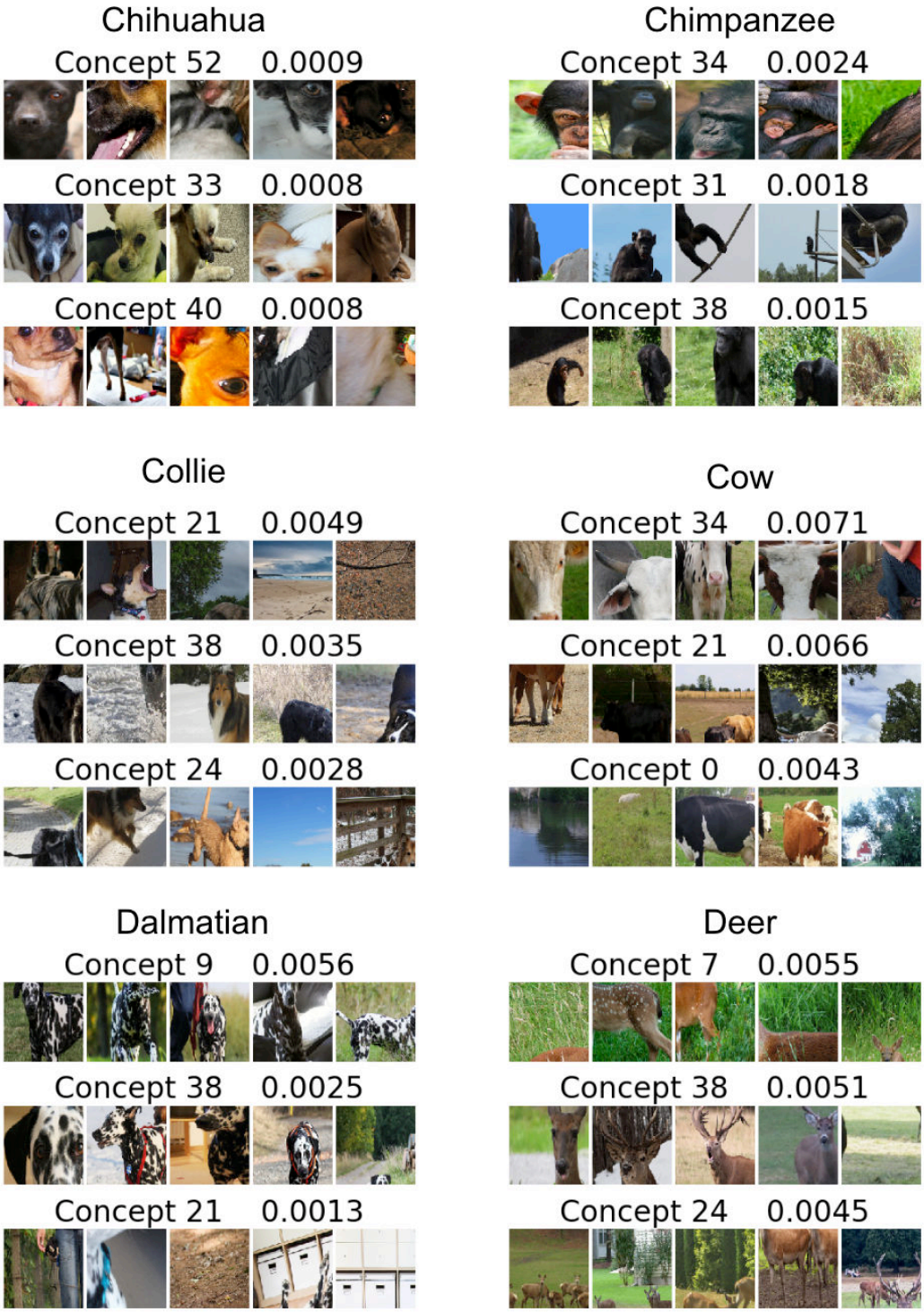

Figure 16: (Larger Version) Nearest Neighbors for each concept obtained in AwA.

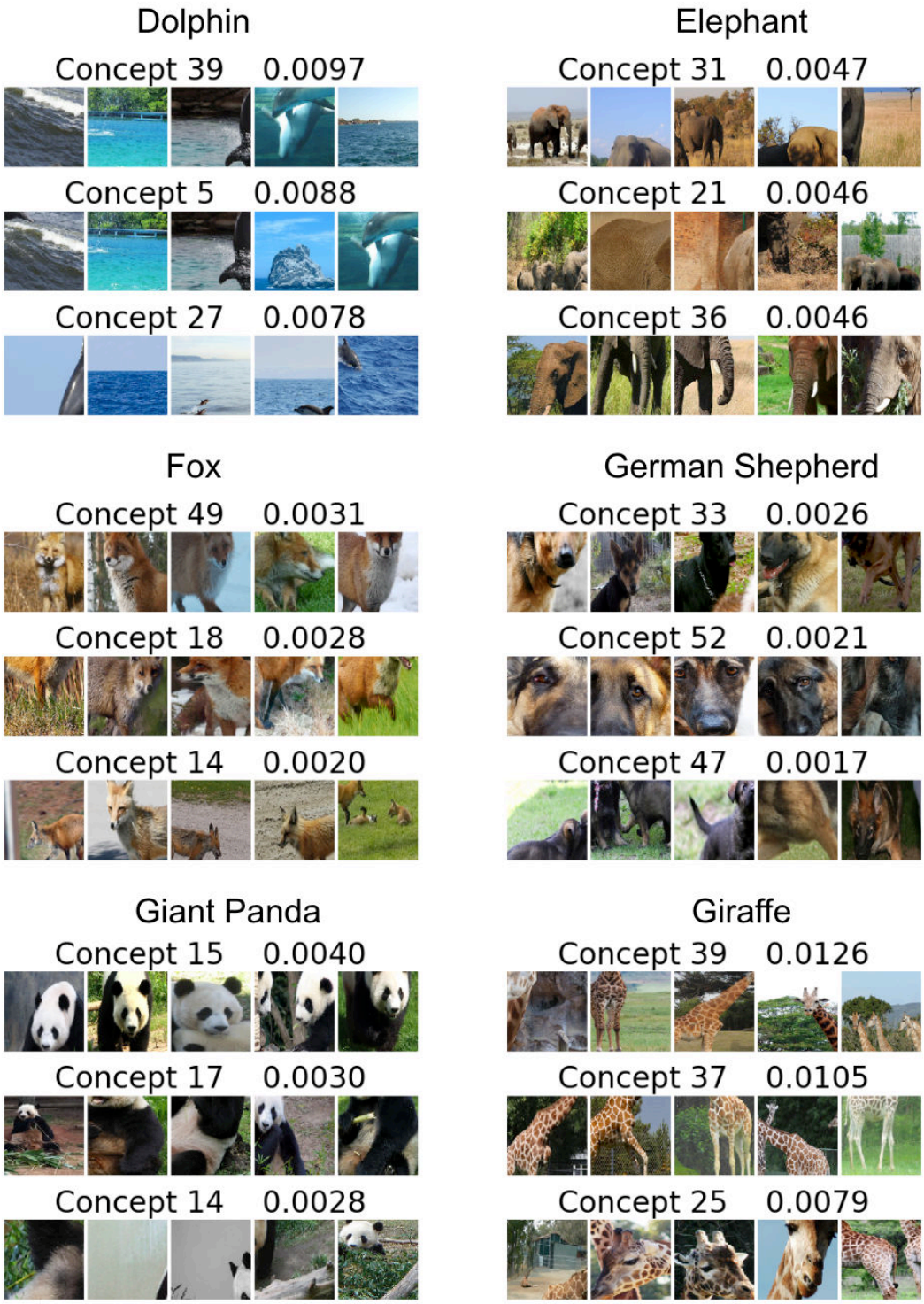

Figure 17: (Larger Version) Nearest Neighbors for each concept obtained in AwA.

Figure 18: (Larger Version) Nearest Neighbors for each concept obtained in AwA.

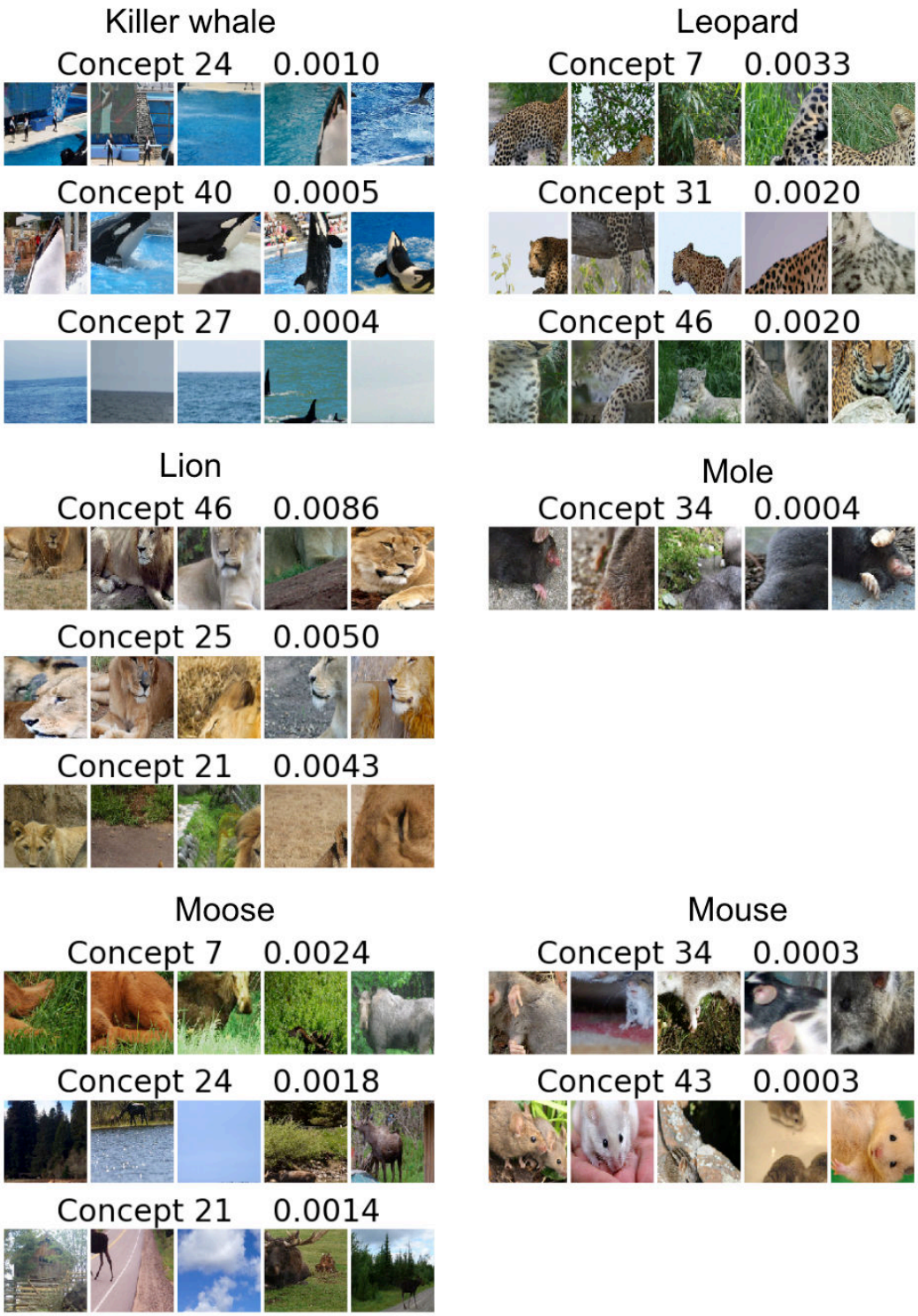

Figure 19: (Larger Version) Nearest Neighbors for each concept obtained in AwA.

Figure 20: (Larger Version) Nearest Neighbors for each concept obtained in AwA.

Figure 21: (Larger Version) Nearest Neighbors for each concept obtained in AwA.

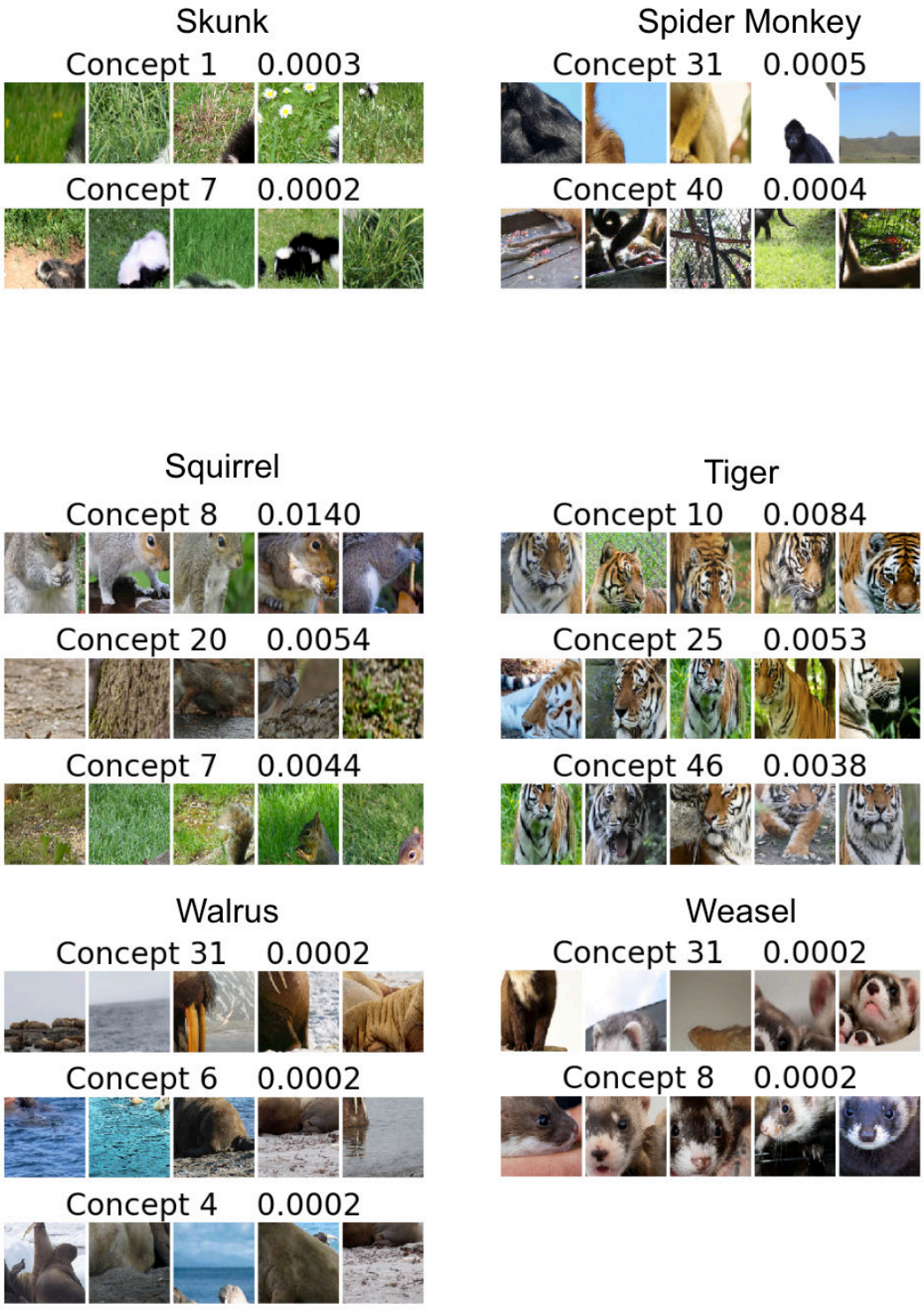

Figure 22: (Larger Version) Nearest Neighbors for each concept obtained in AwA.

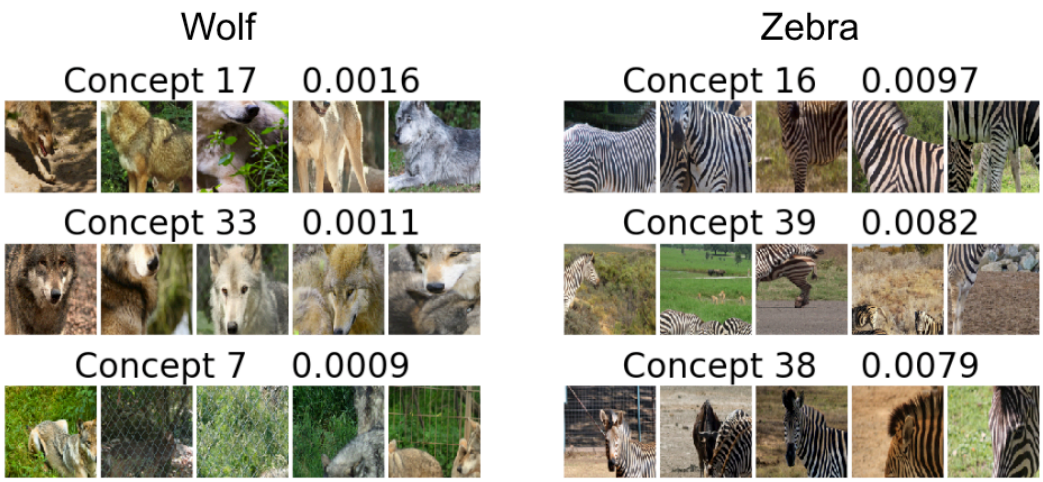

Figure 23: (Larger Version) Nearest Neighbors for each concept obtained in AwA.