Understanding the (Extra-)Ordinary: Validating Deep Model Decisions with Prototypical Concept-based Explanations — Supplementary Material —

Maximilian Dreyer[1], Reduan Achtibat[1],
Wojciech Samek[1,2,3,†], Sebastian Lapuschkin[1,†]
[1] Fraunhofer Heinrich Hertz Institute, [2] Technical University of Berlin,
[3] BIFOLD – Berlin Institute for the Foundations of Learning and Data
[†]corresponding authors: {wojciech.samek | sebastian.lapuschkin}@hhi.fraunhofer.de

Appendix

A. Datasets and Models

In the following, we present all models, datasets and training procedures relevant for our experiments.

A.1. Models

We use ResNet-18 [4], VGG-16 [18] and EfficientNet-B0 [20] architectures in our experiments.

ResNet-18 The ResNet-18 is a convolutional neural network architecture consisting of four `BasicBlock` layers and one fully connected layer. For all experiments, we collect activation and relevance scores after each `BasicBlock` layer.

VGG-16 The VGG-16 is a convolutional neural network architecture consisting of 13 convolutional layers and three fully connected layers. The convolutional layers are given by the identifiers `features.0`, `features.2`, `features.5`, `features.7`, `features.10`, `features.12`, `features.14`, `features.17`, `features.19`, `features.21`, `features.24`, `features.26`, `features.28`, and the dense layers by `classifier.0`, `classifier.3`, `classifier.6`.

EfficientNet-B0 The EfficientNet-B0 is a convolutional neural network architecture consisting of nine `features` layers. For all experiments, we collect activation and relevance scores after each `features` layer.

A.2. Datasets

For our experiments, we include the datasets of ImageNet [15], CUB-200 [21] and CIFAR-10 [8].

ImageNet ImageNet is a dataset for large scale visual recognition, consisting of 1,000 object classes, totaling 14,197,122 images. We use the by the authors provided splits for train and test data. Regarding data processing, we resize images to a size where the smallest edge is 256 px wide, with an additional center crop resulting in 224×224 px image size. Finally, images are normalized with mean $(0.485, 0.456, 0.406)$ and standard deviation $(0.229, 0.224, 0.225)$ over the red, green and blue color channel.

CUB-200 CUB-200 is a visual categorization task dataset consisting of 11,788 images of 200 subcategories belonging to birds, 5,994 for training and 5,794 for testing. Regarding data processing, we resize images to a size where the smallest edge is 224 px wide, with an additional center crop resulting in 224×224 px image size. Finally, images are normalized with mean $(0.47473491, 0.48834997, 0.41759949)$ and standard deviation $(0.22798773, 0.22288573, 0.25982403)$ over the red, green and blue color channel.

CIFAR-10 The CIFAR-10 dataset consists of 60,000 colour images in 10 object classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images in total. Regarding data processing, we resize images to 32×32 px without further normalization.

A.3. Training

Whereas models on ImageNet are pre-trained and taken from the PyTorch model zoo, we train models on CUB-200 and CIFAR-10. Hereby, all models are trained for 100 epochs using the stochastic gradient descent (VGG, ResNet) or ADAM [6] (EfficientNet) algorithm.

CUB-200 For training, we use a batch size of 32. The initial learning rates are 10^{-3} for the VGG and ResNet architecture, and $5 \cdot 10^{-4}$ for the EfficientNet. For data augmentation, Gaussian noise (zero mean, standard deviation of 0.05), random horizontal flips (probability of 0.5), random rotation (up to 10 degrees), random translation (up to 20 % of edge length in all directions) as well as a random scaling (between 80 and 120 %) is applied.

CIFAR-10 For training, we use a batch size of 512. The initial learning rates are 10^{-2} for the VGG and ResNet architecture, and $5 \cdot 10^{-3}$ for the EfficientNet. The learning rate is decreased to one tenth after 50 and 75 epochs. For data augmentation, random horizontal flips (probability of 0.5), as well as a random crop (to 32×32 px, probability 0.5) after padding images with four pixels of zeros is applied.

B. Alternative Metrics for Sample-to-Prototype Assignment

In the following, we present details for how to assign a new prediction with concept relevance vector $\boldsymbol{\nu}$ to a prototypical prediction strategy $\boldsymbol{\mu}_i^k$ (for prototype i of class k).

Gaussian Mixture Model (GMM) The probability density function of a Gaussian distribution (from the fitted GMM) is given as

$$p_i^k(\boldsymbol{\nu}) = \frac{1}{(2\pi)^{\frac{n}{2}} \det(\boldsymbol{\Sigma}_i^k)^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{\nu}-\boldsymbol{\mu}_i^k)^\top (\boldsymbol{\Sigma}_i^k)^{-1}(\boldsymbol{\nu}-\boldsymbol{\mu}_i^k)} \tag{B.1}$$

and serves as a direct means to assign predictions to a prototype. Concretely, we compute the log-likelihood as

$$L_i^k(\boldsymbol{\nu}) = \log p_i^k(\boldsymbol{\nu}) \tag{B.2}$$

and assign predictions to the prototype with highest log-likelihood as

$$\rho^*(\boldsymbol{\nu}) = \text{argmax}_{k,i} \log p_i^k(\boldsymbol{\nu}) \,. \tag{B.3}$$

Note, when the task is to assign any prediction to a class instead of prototype, we use the probability density p^k of the GMM as given in Equation (2).

Mahalanobis Distance Alternatively, one can also use the Mahalanobis distance given as

$$d_{\text{MD},i}^k(\boldsymbol{\nu}) = \sqrt{\left(\boldsymbol{\nu} - \boldsymbol{\mu}_i^k\right)^\top \left(\boldsymbol{\Sigma}_i^k\right)^{-1} \left(\boldsymbol{\nu} - \boldsymbol{\mu}_i^k\right)} \tag{B.4}$$

to assign predictions to prototypes. Concretely, we assign a prediction to the prototype with the smallest distance as

$$\rho^*(\boldsymbol{\nu}) = \text{argmin}_{k,i} d_{\text{MD},i}^k(\boldsymbol{\nu}) \,. \tag{B.5}$$

For assigning a prediction to a class, we assign it to the class of the closest prototype.

Euclidean Distance As a simple alternative, one can also use the Euclidean distance given as

$$d_{\text{E},i}^k(\boldsymbol{\nu}) = \sqrt{\left(\boldsymbol{\nu} - \boldsymbol{\mu}_i^k\right)^\top \left(\boldsymbol{\nu} - \boldsymbol{\mu}_i^k\right)} \,. \tag{B.6}$$

to assign predictions to prototypes. Concretely, we assign a prediction to the prototype with the smallest distance as

$$\rho^*(\boldsymbol{\nu}) = \text{argmin}_{k,i} d_{\text{E},i}^k(\boldsymbol{\nu}) \,. \tag{B.7}$$

For assigning a prediction to a class, we assign it to the class of the closest prototype. It is to note, that whereas log-likelihood and Mahalanobis distance are requiring covariance matrices, Euclidean distance does not. Thus, Euclidean distance can also be used as a lightweight alternative, when prototypes are not computed via GMMs but, *e.g.*, k-means. However, as experiments show, *e.g.*, Section 4.3.2, covariance information is beneficial for better modeling of underlying distributions. Then, as shown in Figure 2b, Mahalanobis or log-likelihood are more accurate in assigning predictions to the true class.

C. Evaluating and Inspecting Prototypes

In this section, we provide additional details and results for the experiments regarding the evaluation and inspection of prototypes, *e.g.*, Sections 4.1 and 4.2.

C.1. Choice of the Attribution Method

For the results in Table 1, where we evaluate prototypes based on concept relevance scores from different attribution methods, we provide Standard Error (SE) values in Table C.1. For faithfulness, we compute the Area Under the Curve (AUC) on eight subsets of the data (totaling 300 samples) to estimate the SE of the mean. Regarding stability and sparseness, we collect all individual values and compute the SE of the mean. For coverage, we compute the accuracy in assigning test samples to known eight sub-strategies. For estimating the sub-strategies, we use half of the training set from ImageNet, and the other half for the test set. Then, for each of the seven animal families (listed in Appendix C.4), we chose eight random classes (each corresponding to a sub-strategy) seven times. Thus, we compute the SE of the mean over 49 coverage scores. Lastly, for outlier detection, we compute the AUC for differentiating between predictions of eight known sub-stragies and five other classes of the same family. Again, for each of the seven animal families, we chose eight random classes (each corresponding to a sub-strategy) and five random outlier classes seven times. In total, we compute the SE of the mean over 49 scores.

C.2. Increasing the Number of Prototypes

In the following, we provide more results for the evaluation of the number of prototypes w.r.t. the metrics defined

Table C.1. Standard Errors for the results reported in Table 1 regarding the evaluation of different attribution methods for concept relevance scores used for prototypes. We report values for ImageNet with 20 classes using (VGG | ResNet | EfficientNet) architectures averaged over all layers.

	Faithfulness	Stability	Sparseness	Coverage	Outlier Detection
LRP (ε-rule) [2]	0.07 | 0.11 | 0.03	0.01 | 0.01 | 0.01	0.8 | 1.2 | 2.5	0.3 | 0.5 | 0.3	0.2 | 0.4 | 0.2
Input×Gradient [17]	0.07 | 0.11 | 0.03	0.01 | 0.01 | 0.03	0.8 | 1.2 | 1.9	0.3 | 0.5 | 0.3	0.2 | 0.4 | 0.2
LRP (composite) [13]	0.07 | 0.12 | 0.03	0.01 | 0.01 | 0.01	0.6 | 0.2 | 0.1	0.2 | 0.5 | 0.3	0.2 | 0.4 | 0.3
GuidedBackProp [19]	0.07 | 0.11 | 0.02	0.01 | 0.01 | 0.04	0.2 | 0.2 | 1.4	0.2 | 0.4 | 0.3	0.2 | 0.4 | 0.2
Activation (max)	0.07 | 0.10 | 0.03	0.01 | 0.01 | 0.01	0.2 | 0.1 | 0.2	0.2 | 0.4 | 0.2	0.2 | 0.5 | 0.3
Activation (mean)	0.06 | 0.11 | 0.03	0.01 | 0.01 | 0.01	0.2 | 0.2 | 0.4	0.2 | 0.5 | 0.3	0.2 | 0.5 | 0.3

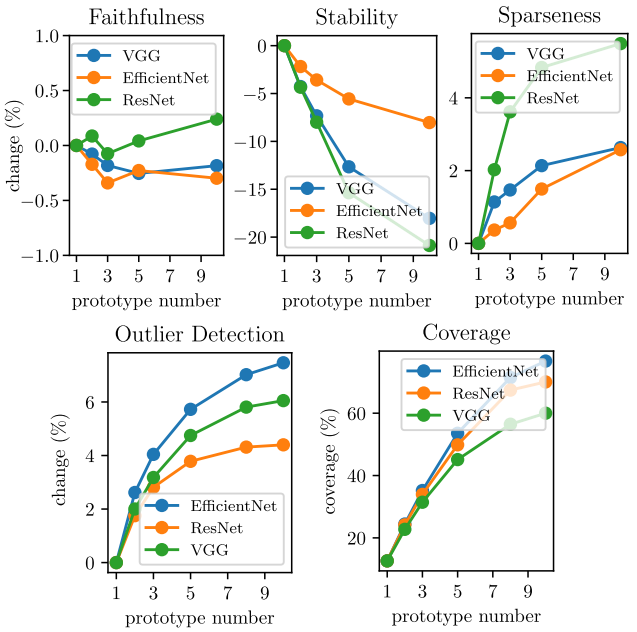

Figure C.1. Effect of increasing the number of prototypes per class on the evaluation metrics. We show the relative change compared to one prototype for all architectures on 20 ImageNet classes using LRP-ε for attributions. For better understanding, we report the actual values for the coverage metric.

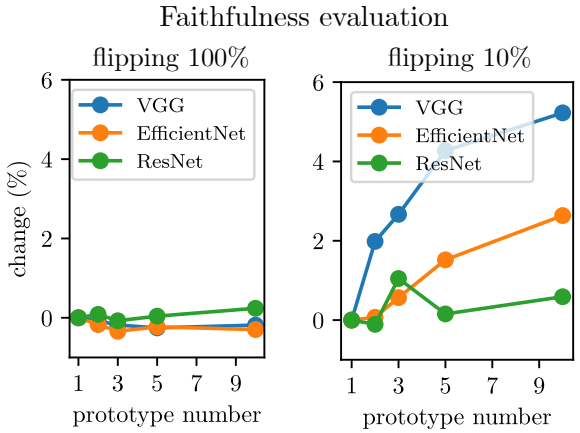

Figure C.2. The change in the faithfulness metric when increasing the number of prototypes, as measured in Section 4.2.2. (*left*): For evaluating faithfulness, all concepts are removed successively. No significant effect w.r.t. the number of prototypes is visible. (*right*): When only 10 % of all concepts are removed, we can observe a stronger increase in faithfulness.

in Section 4.2. The change in metrics when increasing the prototype number is shown in Figure C.1. Here, we can see that stability decreases for higher numbers of prototypes. Sparseness, outlier detection and coverage scores however increase. Faithfulness scores do not significantly change, as described in more detail in the following.

Faithfulness When increasing the number of prototypes, we can not measure a significant increase in faithfulness as reported in Section 4.2.2 and shown again in Figure C.2 (*left*). Notably, however, when we remove only 10 % of the most relevant concepts (instead of all), we can measure a significant effect on faithfulness when increasing the number of prototypes (faithfulness increases), as shown in Figure C.2 (*right*). The higher the number of prototypes, the more likely it is that a prototype is close to any test prediction point. Thus, it could be assumed, that removing concepts according to the closest prototype is leading to a higher faithfulness with a higher number of prototypes. This is however only true for the first removed concepts, as shown in Figure C.2 (*right*). Here, a low number of prototypes (*e.g.*, more global prototypes) seem to be favorable when a high number of concepts are removed.

C.3. Evaluating Clustering Algorithms

In Section 4.2.3, we compare the k-means algorithm against GMMs for modeling sub-strategies of a model. Concretely, using k-means, we first find prototype centroids, and thereafter, we fit a GMM using the k-means centroids as a starting point. Compared to k-means, GMMs are thus based on updated centroids and covariance information.

In order to assign test samples, we use the Euclidean

Table C.2. Effect of different clustering algorithms on the coverage and outlier detection scores using Layer-wise Relevance Propagation (LRP) (ε-rule) concept attributions. We report values for ImageNet with 20 classes using (VGG | ResNet | EfficientNet) architectures averaged over all layers. (Euc.: Euclidean distance is used instead of log-likelihood.)

	Coverage	Outlier Detection
k-means (Euc.)	55.7 | 62.8 | 66.9	69.0 | 76.7 | 77.6
GMM (Euc.)	56.1 | 63.9 | 68.6	69.0 | 77.0 | 77.7
GMM	**56.4** | **66.5** | **71.3**	**70.9** | **78.8** | **82.8**

distance for k-means, and the log-likelihood for GMMs. Specifically, for coverage scores, we assign a prediction to the closest prototype centroid, as in Equation (5) for GMM and use Equation (B.7) for Euclidean distances. For outlier detection, we compute the log-likelihood on the class-level using Equation (4) for GMMs and measure the smallest distance to any prototype for Euclidean distance using Equation B.7. As an additional baseline, we use the updated centroids from the GMM together with Euclidean distance. This way, we can decouple the effects of updated centroids and covariance information.

The resulting coverage and outlier detection scores using LRP (ε-rule) concept attributions are reported in Table C.2. Here, it becomes apparent, that both the updated centroids and covariance information are beneficial for finding correct sub-strategies and detecting outliers. We further show layer-wise scores for all models and for different number of prototypes in Figures C.3 (coverage) and C.4 (object detection). It is visible, that covariance information is especially useful for detecting outliers when the prototype number is low.

C.4. ImageNet species

For the coverage and outlier detection evaluation, we perform experiments over animal classes of the same family. The families and classes/species are as follows:

"**terrier**": ["Staffordshire bullterrier", "American Staffordshire terrier", "Bedlington terrier", "Border terrier", "Kerry blue terrier", "Irish terrier", "Norfolk terrier", "Norwich terrier", "Yorkshire terrier", "wire-haired fox terrier", "Lakeland terrier", "Sealyham terrier", "Airedale", "cairn", "Australian terrier", "Dandie Dinmont", "Boston bull", "miniature schnauzer", "giant schnauzer", "standard schnauzer", "Scotch terrier", "Tibetan terrier", "silky terrier", "soft-coated wheaten terrier", "West Highland white terrier", "Lhasa"]

"**working dog**": ["kuvasz", "schipperke", "groenendael", "malinois", "briard", "kelpie", "komondor", "Old English sheepdog", "Shetland sheepdog", "collie", "Border collie", "Bouvier des Flandres", "Rottweiler", "German shepherd", "Doberman", "miniature pinscher", "Greater Swiss Mountain dog", "Bernese mountain dog", "Appenzeller", "EntleBucher", "boxer", "bull mastiff", "Tibetan mastiff", "French bulldog", "Great Dane", "Saint Bernard", "Eskimo dog", "malamute", "Siberian husky", "affenpinscher"]

"**bird**": ["cock", "hen", "ostrich", "brambling", "goldfinch", "house finch", "junco", "indigo bunting", "robin", "bulbul", "jay", "magpie", "chickadee", "water ouzel", "kite", "bald eagle", "vulture", "great grey owl"]

"**fish**": ["tench", "goldfish", "great white shark", "tiger shark", "hammerhead", "electric ray", "stingray", "barracouta", "eel", "coho", "rock beauty", "anemone fish", "sturgeon", "gar", "lionfish", "puffer"]

"**primate**": ["orangutan", "gorilla", "chimpanzee", "gibbon", "siamang", "guenon", "patas", "baboon", "macaque", "langur", "colobus", "proboscis monkey", "marmoset", "capuchin", "howler monkey", "titi", "spider monkey", "squirrel monkey", "Madagascar cat", "indri"]

"**feline, felid**": ["tabby", "tiger cat", "Persian cat", "Siamese cat", "Egyptian cat", "cougar", "lynx", "leopard", "snow leopard", "jaguar", "lion", "tiger", "cheetah"]

"**snake, serpent, ophidian**": ["thunder snake", "ringneck snake", "hognose snake", "green snake", "king snake", "garter snake", "water snake", "vine snake", "night snake", "boa constrictor", "rock python", "Indian cobra", "green mamba", "sea snake", "horned viper", "diamondback", "sidewinder"]

C.5. Methods for Concept Attribution Computation

In the following, we provide details on how we compute concept attribution scores using activations and local eXplainable Artificial Intelligence (XAI) methods, including LRP, Input×Gradient and GuidedBackprop. For simplicity, we assume that each neuron in a layer corresponds to a concept. Therefore, no transformation between latent feature and space and concept space is required, as described by Equation (1) in the main manuscript.

Activation In each convolutional layer (with n channels of spacial dimension $w \times h$), we collect the activations $\mathbf{A} \in \mathbb{R}^{n \times w \times h}$. We then either perform max- or sum-pooling to compute concept attribution vectors $\boldsymbol{\nu} \in \mathbb{R}^n$, *i.e.*,

$$\nu_i = \max_{uv} A_{iuv} \quad \text{or} \quad \nu_i = \sum_{uv} A_{iuv}, \tag{C.1}$$

respectively.

Input×Gradient We define the function $g : \mathbb{R}^{n \times w \times h} \to \mathcal{Y}$, corresponding the part of the model architecture that maps latent activations $\mathbf{A} \in \mathbb{R}^{n \times w \times h}$ to the model output

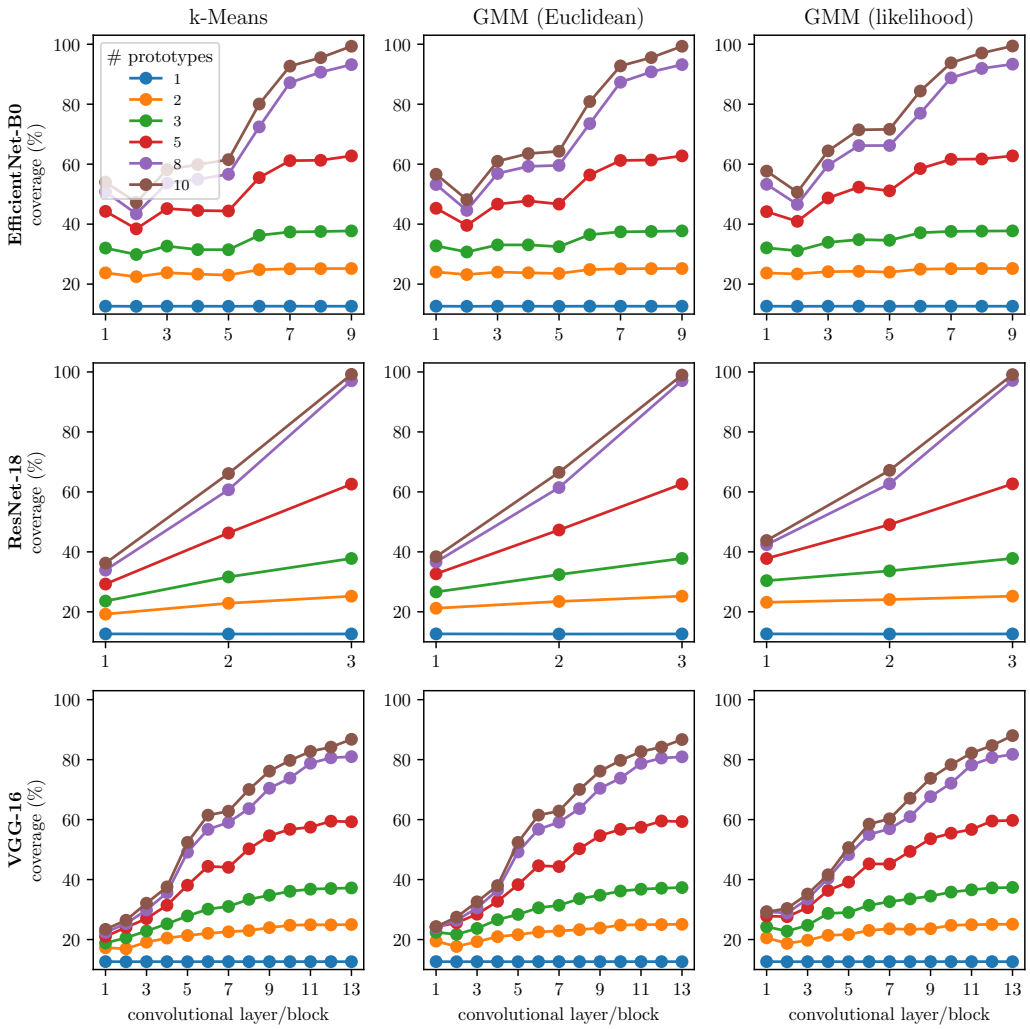

Figure C.3. Effect of different clustering algorithms on the coverage scores using LRP (ε-rule) concept attributions over all model layers for different numbers of prototypes.

y_k of class k. Then, concept relevance scores are given by

$$\nu_i = \sum_{uv} A_{iuv} \frac{\partial g_k(\mathbf{A})}{\partial A_{iuv}}, \tag{C.2}$$

multiplying latent gradients with concept activations.

GuidedBackprop GuidedBackprop is based on the idea of Input×Gradient, using Equation (C.2) to compute concept relevance scores. However, the work of [19] further proposes to prevent the backward flow of negative gradients through ReLU activation functions by setting negative entries of the top gradient to zero.

LRP Regarding our latent predictor g with L layers

$$g(\mathbf{A}) = f_L \circ \cdots \circ f_1(\mathbf{x}) \,, \tag{C.3}$$

LRP follows the flow of activations computed during the forward pass through the model in opposite direction, from the final layer f_L back to the latent layer f_1.

The LRP method distributes relevance quantities R_j^{l+1} (corresponding to neuron j in layer $l+1$) towards a lower layer l proportionally to the relative contributions z_{ij} of lower-layer neuron i to the (pre-)activation z_j as

$$R_{i\leftarrow j}^{(l,\,l+1)} = \frac{z_{ij}}{z_j} R_j^{l+1}, \tag{C.4}$$

where the pre-activation z_j of neuron j for a linear layer operation is given by $z_j = \sum_i z_{ij}$. Then, lower neuron relevance is obtained by losslessly aggregating all incoming relevance messages $R_{i\leftarrow j}$ as

$$R_i^l = \sum_j R_{i\leftarrow j}^{(l,\,l+1)}. \tag{C.5}$$

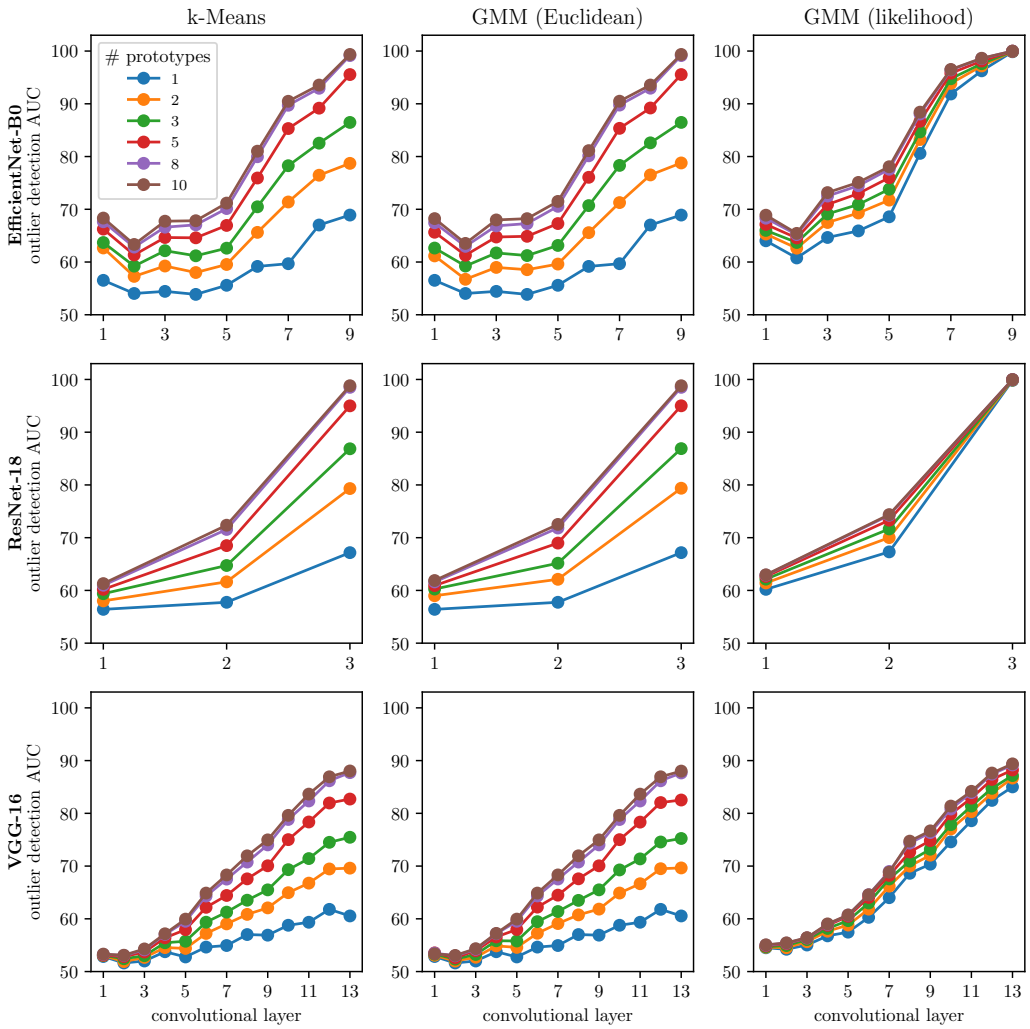

Figure C.4. Effect of different clustering algorithms on the object detection scores using LRP (ε-rule) concept attributions over all model layers for different numbers of prototypes.

For convolutional layers, we have to include the spatial dimensions of latent feature maps resulting in

$$R^{(l,\,l+1)}_{(iuv)\leftarrow(pqr)} = \frac{z_{(uvi)(pqr)}}{z_{pqr}} R^{l+1}_{pqr}. \tag{C.6}$$

Then, the latent relevance of neuron i is computed as

$$R^{l}_{i} = \sum_{uv} \sum_{pqr} R^{(l,\,l+1)}_{(iuv)\leftarrow(pqr)}. \tag{C.7}$$

Note, that final concept relevance scores are then given when we reach the input level, *i.e.*, $l = 1$.

LRP ε-rule: To ensure numerical stability, the LRP ε-rule is introduced and defined as

$$R^{(l,\,l+1)}_{i\leftarrow j} = \frac{z_{ij}}{z_j + \varepsilon \cdot \text{sign}(z_j)} R^{l+1}_{j}. \tag{C.8}$$

Note that for the purpose of numerical stability, the definition of the sign function is altered such that $\text{sign}(0) = 1$. The ε-rule is in ReLU networks highly similar to the multiplication of the input times its gradient w.r.t. the output [7]. However, due to its similarity to gradient-based attribution computation, the LRP ε-rule may result in noisy attributions in very deep models, where gradient shattering and noisy gradients appear [3].

LRP composite: An alternative approach is to combine multiple propagation rules, so-called rule composites, which result in attributions that are less influenced by a noisy gradient: the LRP εz^+-rule is an established best practice [7, 13] to keep LRP attribution maps informative, readable and representative while combating gradient shattering effects. The LRP z^+-rule operates on the convolutional layers and LRP ε-rule is utilized in standard dense

layers. The LRP z^+-rule is given as

$$R_{i\leftarrow j}^{(l,\,l+1)} = \frac{(z_{ij})^+}{z_j^+} R_j^{l+1} \tag{C.9}$$

by only taking into account positive contributions $z_j^+ = \sum_i (z_{ij})^+$ with $(\cdot)^+ = \max(0, \cdot)$.

Normalization of Concept Relevance Scores Please note, that in order to be interpretable as percentage scores, we normalize concept relevances to an absolute sum of one, *i.e.*, $\sum_i \nu_i' = 1$ with $\nu_i' = \frac{\nu_i}{\sum_j |\nu_j|}$.

Other Attribution Methods We refrain from using other popular attribution methods such as SHAP [11], LIME [14], and GradCAM [16] due to their infeasibility or inapplicability. Concretely, SHAP and LIME require perturbation of features, resulting in a multitude of separate forward passes that need to be performed. Further, compared to perturbation-based methods, the backpropagation-based feature attribution methods previously presented provide concept relevances for *all* layers in a *single* backward pass. GradCAM [16] provides input heatmaps that localize important features by up-sampling latent feature maps. As such, GradCAM can only be utilized for spatial localization of concepts, but not to compute latent feature scores.

C.6. Investigating Data Quality

In the following, we present examples for Section 4.1 on how prototypes enable an understanding of the data itself (and how the model uses the data).

C.6.1 Discovering Model (Sub-)Strategies

We begin with four examples on how prototypes reveal sub-populations in the data (corresponding to learned sub-strategies) in Figures C.5 and C.6. Here we present for the ImageNet classes "hen", "ice bear", "bald eagle" and "buckeye" eight prototypes of different models, with six example images shown corresponding to data points that are closest to a prototype. For the class "hen", the model (EfficientNet-b0, last `features` block) has learned to distinguish hen of different color, *e.g.*, brown, white and black. Regarding the "ice bear" class, the model (VGG-16, layer `features.28`) has learned to perceive ice bears in different environments, *e.g.*, on ice, in water or with gray background. For the "bald eagle" class, the model (ResNet-18, last `BasicBlock` layer) has learned to differentiate between eagles in various environments, *e.g.*, flying over water or sitting on branches. Lastly, regarding the class of "buckeye" (chestnut), the model (VGG-16, layer `features.28`) has learned to perceive buckeyes in different age stages (including the full tree form), *e.g.*, with green hull or partly visible. In the figures, we provide additional information on how many samples a prototype "covers", measured by the number of instances closest to the prototype in the training set, and how similar they are to the overall mean (concept relevance vector) in terms of cosine similarity.

C.6.2 Identifying Mislabeled Instances via Prototypes

By studying prototypes, we found several classes, where wrong objects are included unintendedly, thus receiving a wrong label. As ImageNet is a collection of images generated through keyword search (the class labels) of image databases, ambiguous class labels might result in the retrieval of unwanted or wrong images. For example, we found instances where pictures of Leopard Lacewing butterflies were mistakenly assigned to the "lacewing" class, a completely different insect species, as shown in Figure C.7 (*top*). Similarly, we found tigers in the "Tiger Cat" class, as depicted in Figure C.7 (*bottom*). More examples are shown in Figure C.8 where cars and buses are included in the "passenger (railroad) car" class, and Blue Lynx Ragdoll cats for the "lynx, catamount" class. In these two figures, we show eight prototypes for VGG, ResNet and EfficientNet models, with an additional UMAP [12] embedding illustrating the distribution of prediction strategies, *i.e.*, concept relevance vectors.

C.6.3 Spotting Correlating Features

Another noteworthy set of data artifacts are correlating features found within the ImageNet training dataset. These features include white wolfs behind fences (Figure C.9), dogs with tennis balls (Figure C.10), and cats in cartons or buckets (Figures C.11 and C.12, respectively). By being able to understand the concepts that are relevant (characteristic) for each prototype, we can further validate our observations. We thus show for all prototypes the concept relevance scores for the set of most relevant concepts (according to a VGG-16 in layer `features.28`). Concretely, the set of concepts is given by retrieving the top-2 concepts for each prototype. Further, concepts are visualized using reference samples as given by the RelMax technique [1]. With RelMax, we visualize concepts with samples, where the concept was most relevant w.r.t. the prediction outcome, illustrating how the concept is *used* by the model. To improve clarity, each reference sample is cropped to the for the concept relevant part, and all irrelevant parts are masked by a semi-transparent black mask.

C.6.4 Diagnosing for Poor Data Quality

By studying prototypes and their characteristic concepts we further were able to reveal issues of poor data quality. This

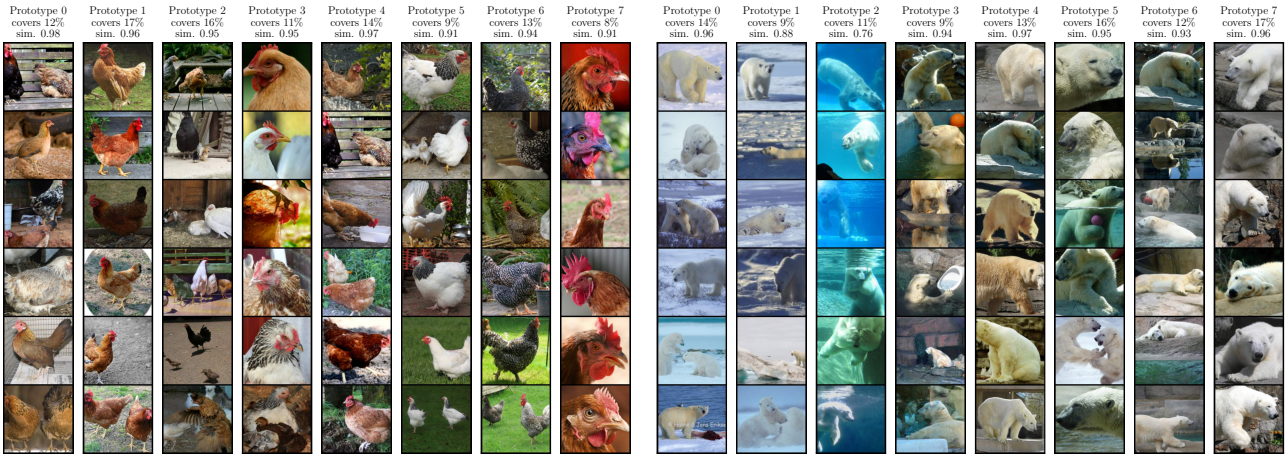

Figure C.5. Understanding model sub-strategies using (eight) prototypes. We provide information on how many samples a prototype "covers", measured by the number of instances closest to the prototype in training set, and how similar they are to the overall mean in terms of cosine similarity. (*left*): For the ImageNet class "hen" (EfficientNet-b0, last `features` block), the model has learned to perceive hen of different color, *e.g.*, brown, white and black. (*right*): For the ImageNet class "ice bear" (VGG-16, layer `features.28`), the model has learned to perceive ice bears in different environments, *e.g.*, on ice, in water or with gray background.

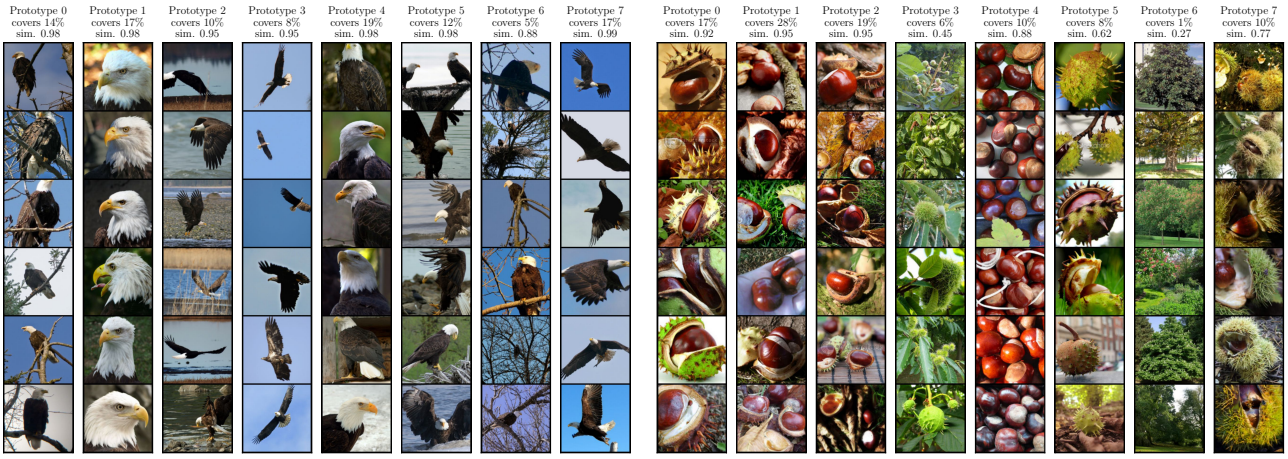

Figure C.6. Understanding model sub-strategies using (eight) prototypes. We provide information on how many samples a prototype "covers", measured by the number of instances closest to the prototype in the train set, and how similar they are to the overall mean in terms of cosine similarity. (*left*): For the ImageNet class "bald eagle" (ResNet-18, last `BasicBlock` layer), the model has learned to perceive eagles in different environments, *e.g.*, flying over water or sitting on branches. (*right*): For the ImageNet class "buckeye" (VGG-16, layer `features.28`), the model has learned to perceive buckeyes in different age stages, *e.g.*, with green hull or partly visible. Prototype 6 for "buckeye" shows not the fruit, but the whole tree, which raises the question whether corresponding samples (not showing the fruit) should be excluded from the training data set. Alternatively, predicted samples close to the prototype could be labeled with a "warning" note.

includes, *e.g.*, prototypes where a "blur" artifact is relevant, as found in the ImageNet classes of "red-breasted merganser" (birds), "milk can" and "Windsor tie", as shown in Figures C.13, C.14 and C.15 for VGG-16 and ResNet models. This dedicated "blur" concepts result from a large set of training images having poor resolution.

Further, in the class of "pickelhaube" we found that the model has learned to detect the class object not only because of the pickelhaube, but also in the absence of the object because of the uniform or face features, as shown in Figure C.16. Here, prototype 6 captures military men in uniform where the head (and thus also the pickelhaube) is

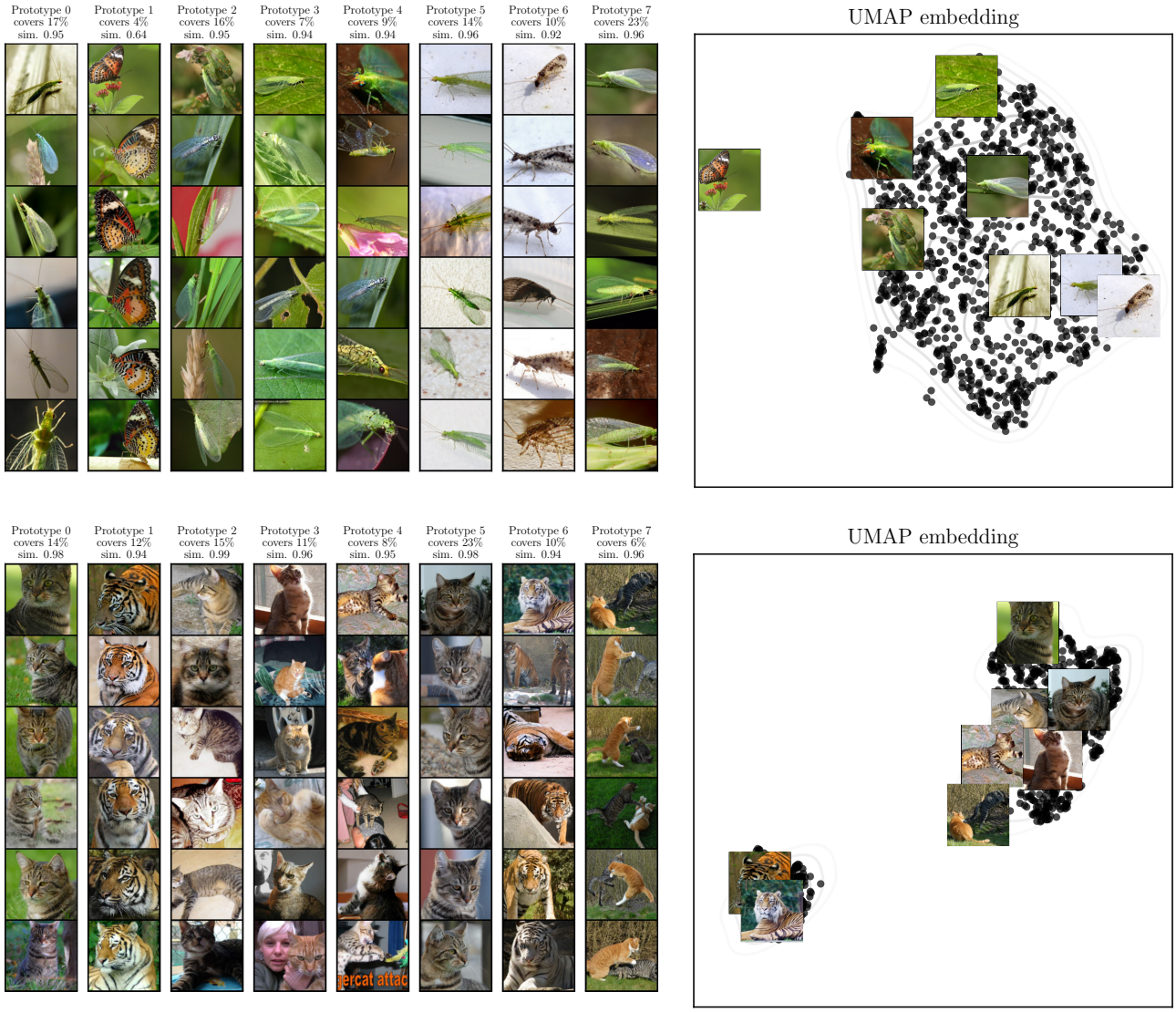

Figure C.7. Revealing wrong object samples in the ImageNet dataset using (eight) prototypes. Besides example images for each prototype, we also show a UMAP embedding. We further provide information on how many samples a prototype "covers", measured by the number of instances closest to the prototype in training set, and how similar they are to the overall mean in terms of cosine similarity. (*top*): For the ImageNet class "lacewing" (VGG-16, layer `features.28`), there are also samples of Leopard Lacewing butterflies in the training data. (*bottom*): For the ImageNet class "tiger cat" (ResNet-18, last `BasicBlock` layer), there are also samples of tigers in the training data.

cropped out, leading the model to use also alternative features. This illustrates the effect data augmentation techniques (including, *e.g.*, random cropping) can have on the model behavior.

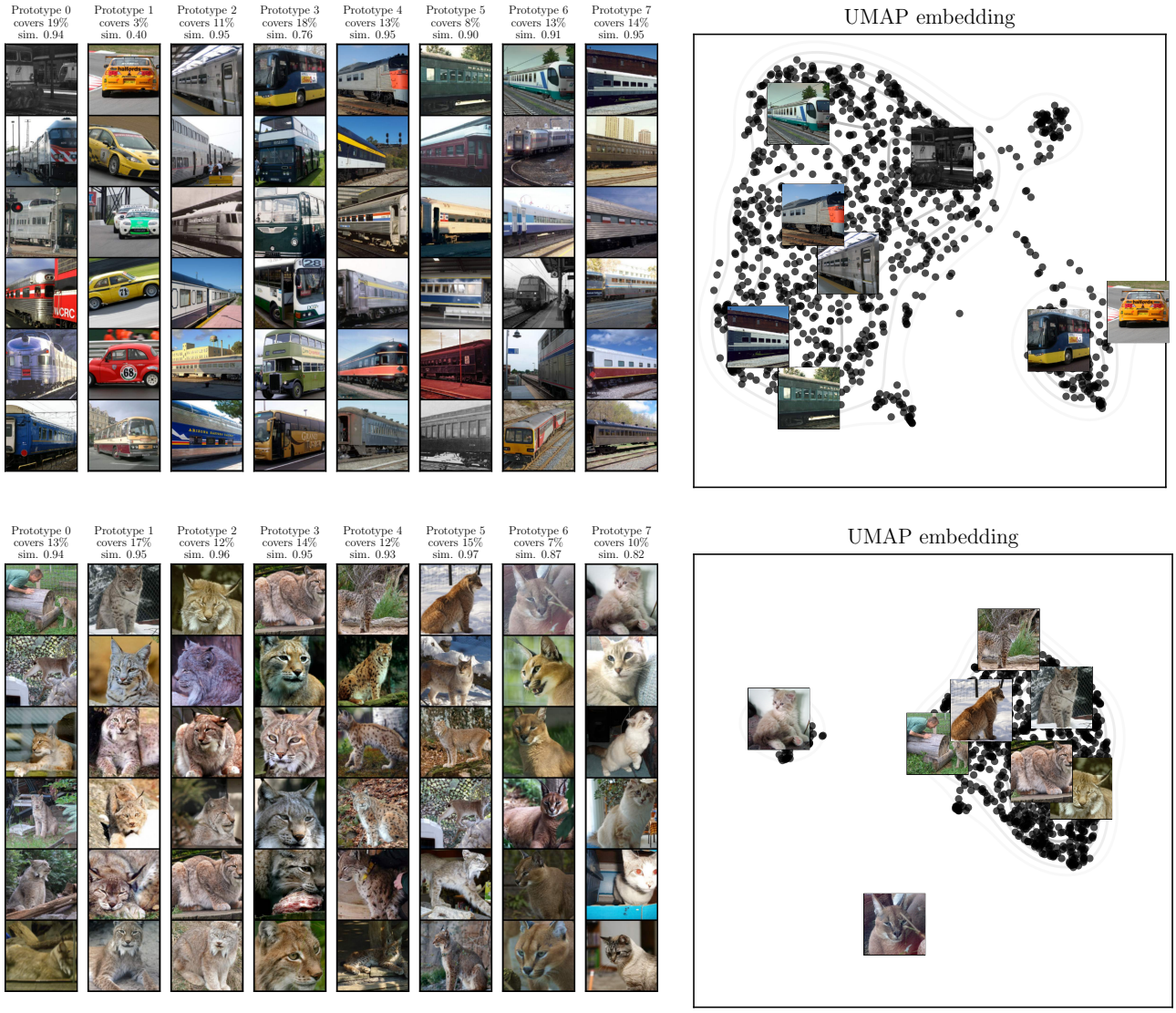

Figure C.8. Revealing wrong object samples in the ImageNet dataset using (eight) prototypes. We further provide information on how many samples a prototype "covers", measured by the number of instances closest to the prototype in training set, and how similar they are to the overall mean in terms of cosine similarity. Besides example images for each prototype, we also show a UMAP embedding. (*top*): For the ImageNet class "passenger (railroad) car" (VGG-16, layer `features.28`), there are also samples of buses and cars in the training data. (*bottom*): For the ImageNet class "lynx, catamount" (EfficientNet-b0, last `features` block), there are also samples of Blue Lynx Ragdoll cats (prototype 7) in the training data. Note the distinct cluster for catamounts (prototype 6) compared to lynx.

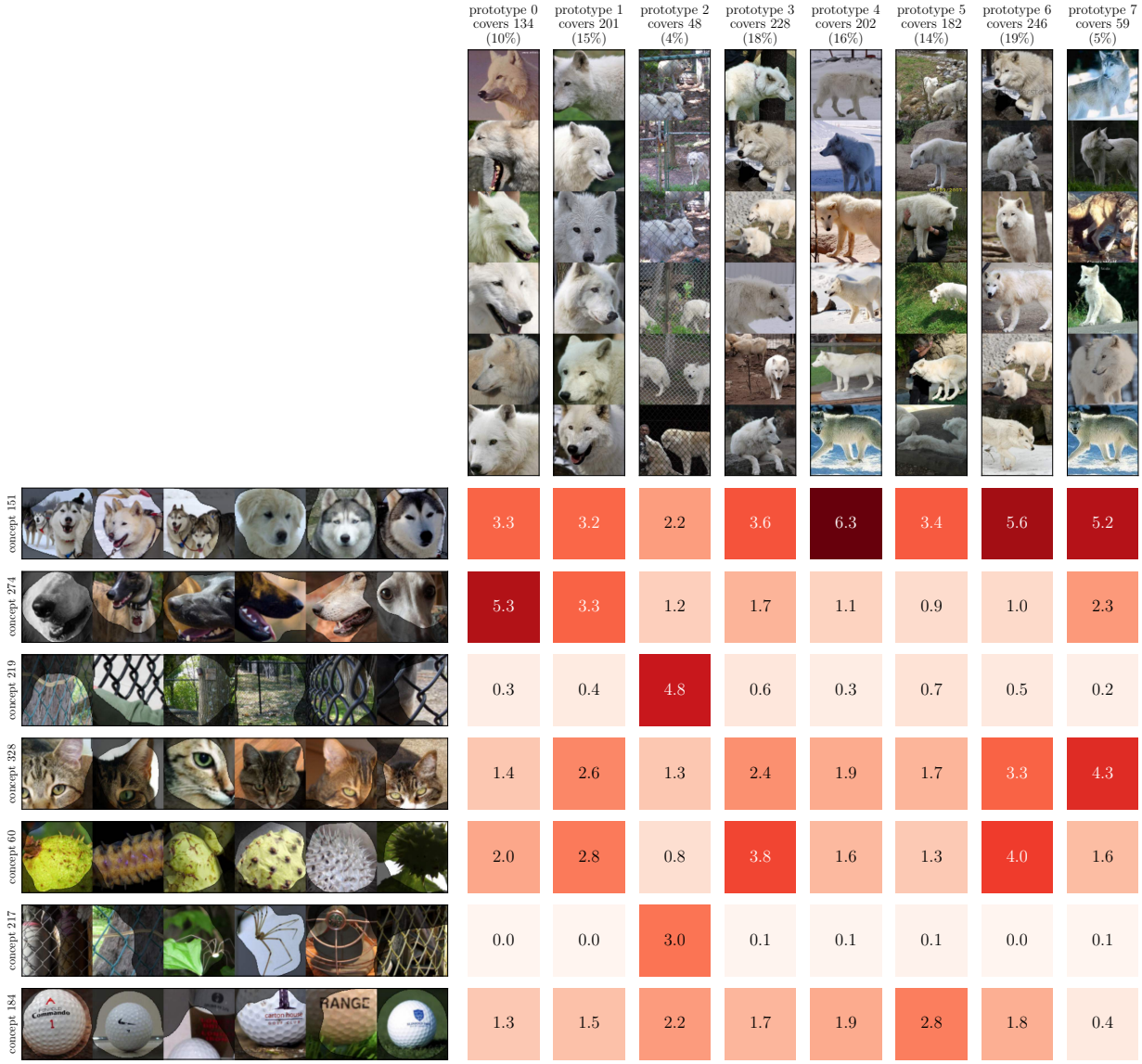

Figure C.9. Revealing correlating features in the ImageNet dataset using (eight) prototypes of a VGG-16 in layer `features.28`. For each prototype, we show relevant concepts and their corresponding relevance scores (%). We further provide information on how many samples a prototype "covers", measured by the number of instances closest to the prototype in training set. For the class "white wolf", prototype 2 deviates from the other prototypes by a high relevance on "fence" concepts.

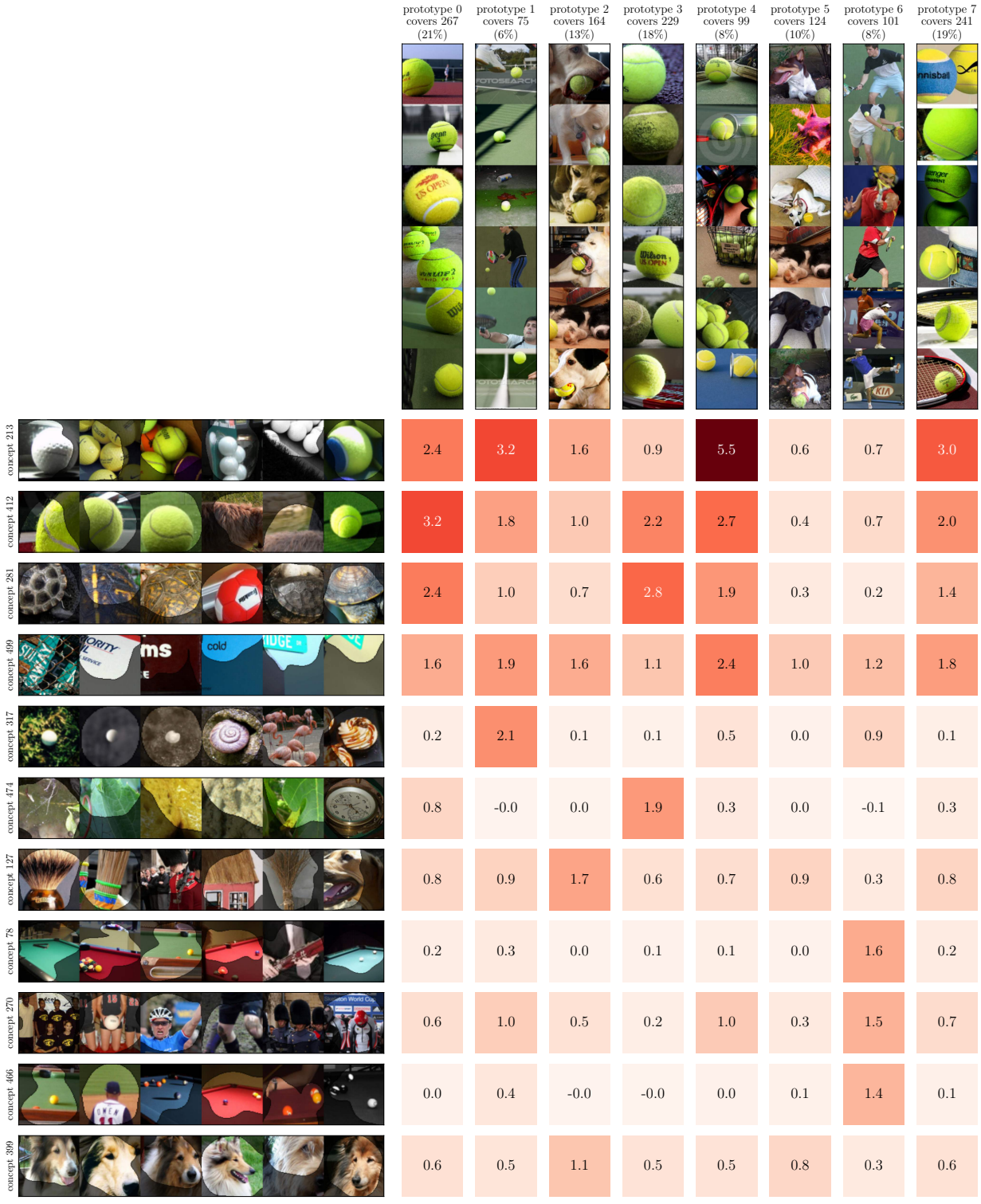

Figure C.10. Revealing correlating features in the ImageNet dataset using (eight) prototypes of a VGG-16 in layer `features.26`. For each prototype, we show relevant concepts and their corresponding relevance scores (%). We further provide information on how many samples a prototype "covers", measured by the number of instances closest to the prototype in training set. For the class "tennis ball", prototype 2 (and 5) deviate from the other prototypes by relevances on "dog" concepts. Notably prototype 6 also deviates, with a focus on concepts related to persons.

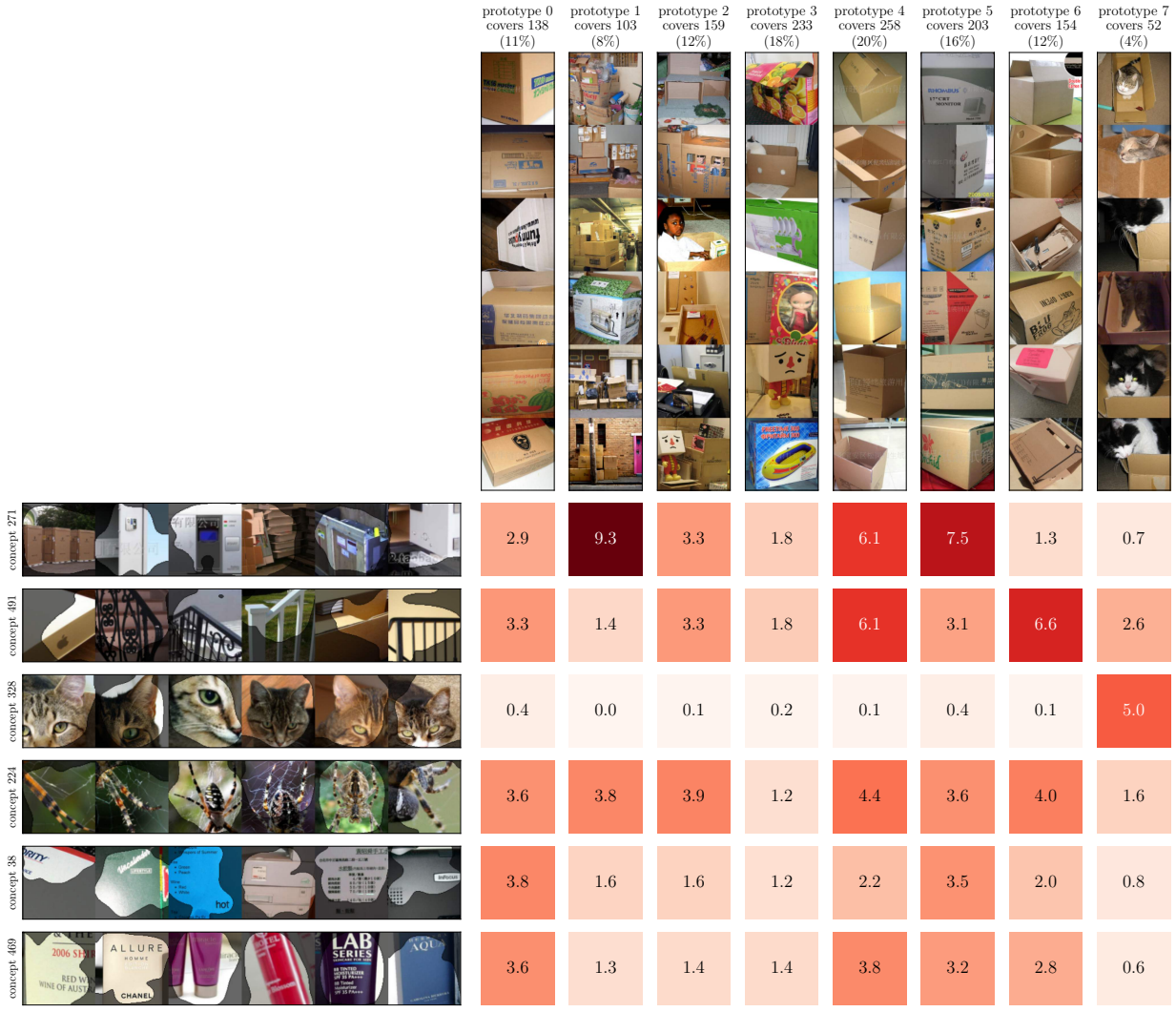

Figure C.11. Revealing correlating features in the ImageNet dataset using (eight) prototypes of a VGG-16 in layer `features.28`. For each prototype, we show relevant concepts and their corresponding relevance scores (%). We further provide information on how many samples a prototype "covers", measured by the number of instances closest to the prototype in training set. For the class "carton", prototype 7 deviates from the other prototypes by a high relevance on "cat" concepts. Note, that concept 328 refers to spider webs (thin lines), which is likely to be triggered for the writings on the outside of cartons, or the overlaid watermark seen in, *e.g.*, prototype 4.

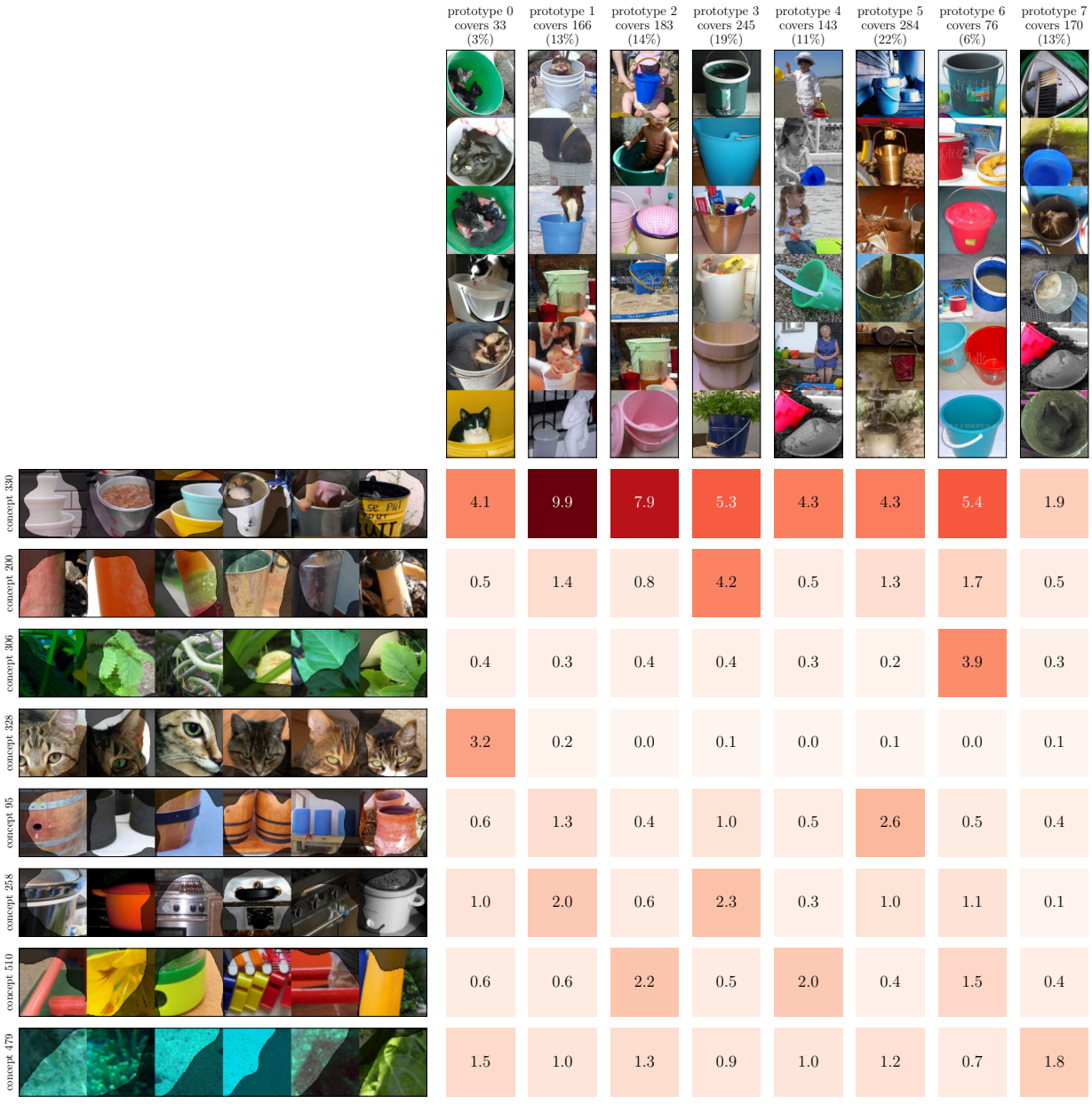

Figure C.12. Revealing correlating features in the ImageNet dataset using (eight) prototypes of a VGG-16 in layer `features.28`. For each prototype, we show relevant concepts and their corresponding relevance scores (%). We further provide information on how many samples a prototype "covers", measured by the number of instances closest to the prototype in training set. For the class "bucket", prototype 0 deviates from the other prototypes by a high relevance on "cat" concepts. Further note prototype 6 and the Chinese watermark artifact on the example images.

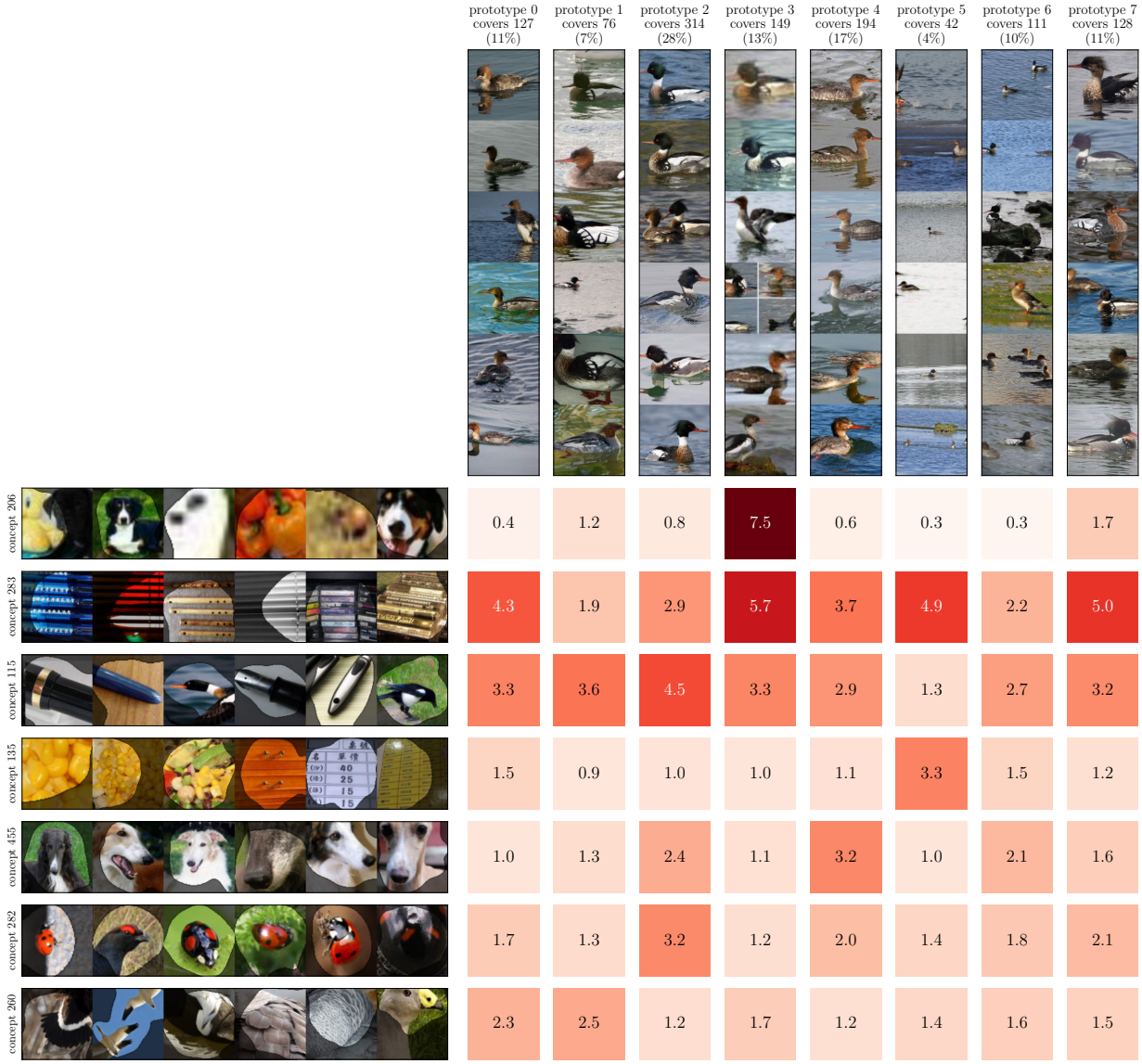

Figure C.13. Revealing correlating features in the ImageNet dataset using (eight) prototypes of a VGG-16 in layer `features.28`. For each prototype, we show relevant concepts and their corresponding relevance scores (%). We further provide information on how many samples a prototype "covers", *i.e.*, are closest to the prototype in training set. For the class "red-breasted merganser", prototype 3 deviates from the other prototypes by a high relevance on "blur" concepts.

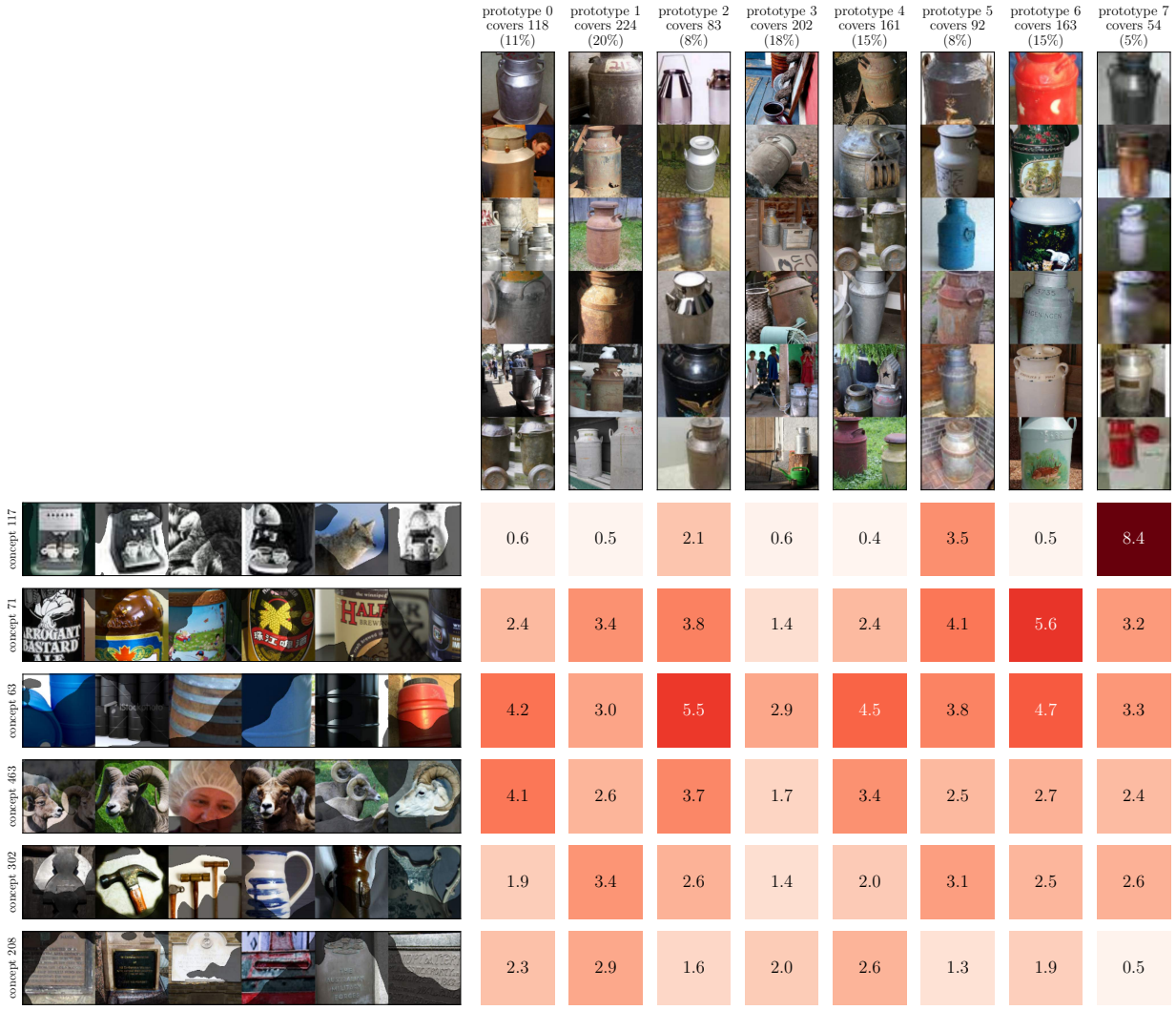

Figure C.14. Revealing data quality issues in the ImageNet dataset using (eight) prototypes of a ResNet-18 in the last `BasicBlock` layer. For each prototype, we show relevant concepts and their corresponding relevance scores (%). We further provide information on how many samples a prototype "covers", *i.e.*, are closest to the prototype in training set. For the class "milk can", prototypes 5 and 7 deviate from the other prototypes by a high relevance on a "blur" concept (concept 117).

Figure C.15. Revealing data quality issues in the ImageNet dataset using (eight) prototypes of a VGG-16 in layer `features.28`. For each prototype, we show relevant concepts and their corresponding relevance scores (%). We further provide information on how many samples a prototype "covers", *i.e.*, are closest to the prototype in training set. For the class "Windsor tie", prototype 2 and 6 deviate from the other prototypes by a high relevance on "blur" concepts.

Figure C.16. Revealing data quality issues in the ImageNet dataset using (eight) prototypes of a ResNet-18 in the last `BasicBlock` layer. For each prototype, we show relevant concepts and their corresponding relevance scores (%). We further provide information on how many samples a prototype "covers", *i.e.*, are closest to the prototype in training set. For the class "pickelhaube", prototype 6 deviates from the norm by a high relevance on "uniform" concepts, instead of the pickelhaube. Notably, concept 73 has also (almost) no relevance, which corresponds to a dome-like form (corresponding to the shape of the pickelhaube). This is also sensible as in the example images, the heads are cropped out — due to data augmentation. Also prototype 2, which corresponds to group pictures, is associated with high relevances on uniform concepts. It is further to note that prototypes 0, 2 and 6 show a high relevances on a "gray color" concept, indicating that the model has learned to detect gray-scale pictures.

D. Model (Prediction) Validation

In this section, we present additional examples and clarifications regarding the use of prototypes for model (prediction) validation.

D.1. Covariance Matrix Understanding

A high likelihood of a prediction with concept relevance vector $\boldsymbol{\nu}$ belonging to prototype i of class k is given for a high probability density value $p_i^k(\boldsymbol{\nu})$, *i.e.*,

$$p_i^k(\boldsymbol{\nu}) = \frac{1}{(2\pi)^{\frac{n}{2}} \det(\boldsymbol{\Sigma}_i^k)^{\frac{1}{2}}} e^{-\frac{1}{2}\left(\boldsymbol{\nu}-\boldsymbol{\mu}_i^k\right)^\top \left(\boldsymbol{\Sigma}_i^k\right)^{-1}\left(\boldsymbol{\nu}-\boldsymbol{\mu}_i^k\right)} . \tag{D.1}$$

Here, the probability density $p_i^k(\boldsymbol{\nu})$ depends on the term

$$\delta_i^k(\boldsymbol{\nu}) = \left(\boldsymbol{\nu}-\boldsymbol{\mu}_i^k\right)^\top \left(\boldsymbol{\Sigma}_i^k\right)^{-1} \left(\boldsymbol{\nu}-\boldsymbol{\mu}_i^k\right) . \tag{D.2}$$

For simplicity, we assume that we choose one prototype and one class, simplifying notation to

$$\delta = \left(\boldsymbol{\nu}-\boldsymbol{\mu}\right)^\top \boldsymbol{\Sigma}^{-1} \left(\boldsymbol{\nu}-\boldsymbol{\mu}\right) = \boldsymbol{\Delta}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{\Delta} \tag{D.3}$$

with $\boldsymbol{\Delta} = \boldsymbol{\nu} - \boldsymbol{\mu}$. Then, we have

$$\delta = \Delta_1 \Sigma_{11}^{-1} \Delta_1 + \Delta_1 \Sigma_{12}^{-1} \Delta_2 + \cdots + \Delta_m \Sigma_{mm}^{-1} \Delta_m . \tag{D.4}$$

Here, contributions result from intra-concept deviations, *i.e.*, $\Delta_i \Sigma_{ii}^{-1} \Delta_i \geq 0$ (as $\Sigma_{ii}^{-1} \geq 0$, because Σ and Σ^{-1} are positive semi-definite matrices), and inter-concept deviations $\Delta_i \Sigma_{ij}^{-1} \Delta_j$ (with $i \neq j$). Contrary to only examining the difference vector $\boldsymbol{\Delta} = \boldsymbol{\nu} - \boldsymbol{\mu}$, δ also includes information from the covariance matrix and allows to investigate contributions from inter-concept deviations.

D.2. Increasing the Number of Prototypes

We in the following provide qualitative examples of resulting prototypes when increasing their number (used to fit the GMM). The first example is shown in Figure D.1, where we visualize the emerging prototypes for ImageNet class "fireboat" setting the prototype number to one, two and four. Whereas one prototype only visualizes fireboats spraying water, we reveal that the model has learned to differentiate between fireboats spraying and not spraying water by increasing prototype numbers.

A second example is shown in Figure D.2, where we visualize the emerging prototypes for ImageNet class "American eagle" when varying the prototype number. Whereas one prototype visualizes eagles sitting on a branch or flying, we reveal that the model has learned to differentiate between flying eagles, sitting eagles and eagle heads by increasing prototype numbers. A third example is shown in Figure D.3, where we visualize the emerging prototypes for ImageNet class "space shuttle" when varying the prototype number. By increasing prototype numbers, we reveal that the model has learned to differentiate between starting space ships (with dust clouds and fire), flying space ships and space ships in a halls.

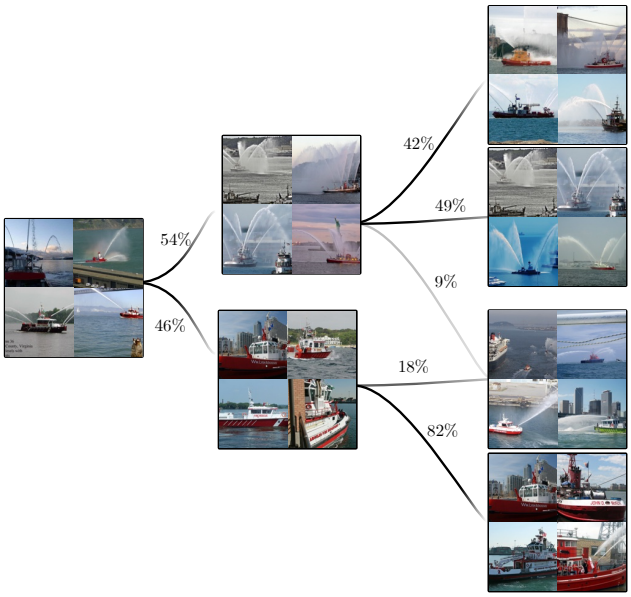

Figure D.1. Qualitative example for changing the number of prototypes to one (*left*), two (*middle*) and four (*right*). We show the prototypes for the ImageNet class "fireboat" resulting for a VGG-16 model and layer `features.28`. Whereas the single prototype depicts fireboats spraying (little) water, increasing the number leads to more distinct prototypes, *e.g.*, fireboats with *and* without water fountains of varying strength. Note that we also depict the amount of samples (closest to a prototype) that transfer from one prototype to the other.

D.3. Revealing Spurious Behavior

In Section 4.3.1, we found a cluster for the ImageNet "carton" class, in which cats were depicted sitting in cartons. As shown in Figure D.4, predictions in the cat cluster also have a high softmax probability score for the "carton" class. Thus, not only are cat features highly relevant (w.r.t. "carton") for these samples (as shown in Figure 5), the "carton" output probability score is high as well.

D.4. Out-of-Distribution Detection

We present additional results for the Out Of Distribution (OOD) detection experiment in Section 4.3.2. Concretely, the results for all models trained on CIFAR-10 are shown in Table D.1 and for CUB-200 in Table D.2. Again we compare the methods of MSP [5], Energy [10] and Mahalanobis [9] with variants of Prototypical Concept-based Explanations (PCX) based on the log-likelihood measure as defined in Equation (4), and Euclidean distance. For

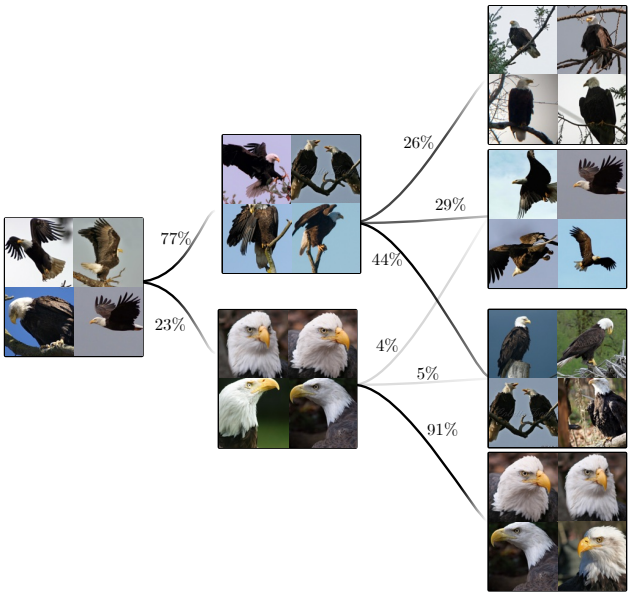

Figure D.2. Qualitative example for changing the number of prototypes to one (*left*), two (*middle*) and four (*right*). We show the prototypes for the ImageNet class "American eagle" resulting for a VGG-16 model and layer `features.28`. Whereas the single prototype depicts eagles flying and sitting on branches, increasing the number leads to more distinct prototypes, *e.g.*, close-ups, flying and sitting. Note that we also depict the amount of samples (closest to a prototype) that transfer from one prototype to the other.

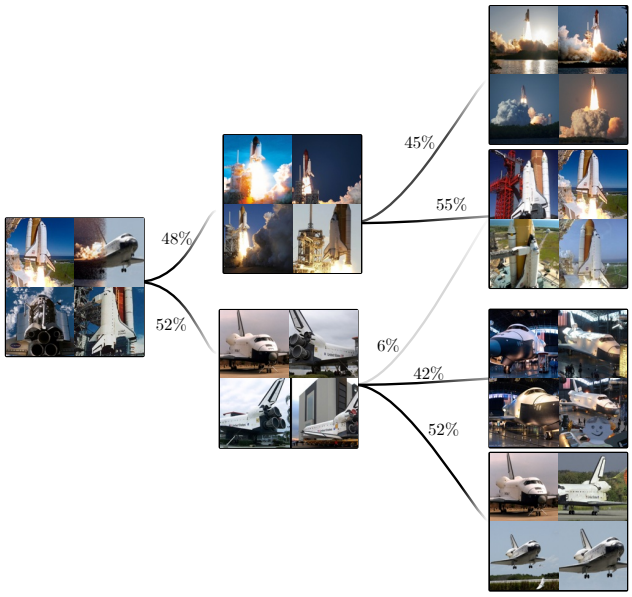

Figure D.3. Qualitative example for changing the number of prototypes to one (*left*), two (*middle*) and four (*right*). We show the prototypes for the ImageNet class "space shuttle" resulting for a VGG-16 model and layer `features.28`. Whereas the single prototype depicts various positions of a space shuttle, increasing the number leads to more distinct prototypes, *e.g.*, launching with a tail of fire, standing in exhibitions, landing on the landing site. Note that we also depict the amount of samples (closest to a prototype) that transfer from one prototype to the other.

ImageNet, we perform OOD detection on the first 50 of the 1000 ImageNet classes. Note, that for PCX, we only compute GMMs with one prototype, and leave optimization w.r.t. prototype number and feature layer for future work.

D.5. Studying (Dis-)Similarities Across Classes

In our experiments, prototypes emerge as tools for exploring both similarities and distinctions in concept utilization across various classes.

For a global understanding of the model, we present matrices depicting inter-class similarities (using cosine similarity between prototypes) for a VGG-16 trained on CUB-200 (Figure D.8), CIFAR-10 (Figure D.5), ResNet on ImageNet (Figure D.7) and EfficientNet on ImageNet (Figure D.6).

Moreover, we delve into detailed analyses of layer `features.28` for a VGG-16 model. Here, we illustrate the similarities and differences between classes by choosing three characteristic concepts. For instance, when comparing the ImageNet classes "hen" and "rooster" as in Figure D.9 (*top*), we observe an equal reliance on the "eye" concept. However, the rooster places greater importance on the "red color" compared to the hen, where the brown-gray color is more significant.

Similarly, when comparing between the ImageNet classes "goldfish" and "House Finch" as shown in Figure D.9 (*middle*), both exhibit a reliance on the "scales" texture. However, the goldfish emphasizes the orange color, while the House Finch uniquely employs the "animal on branch" concept.

In another instance, when comparing CUB-200 classes "Lazuli Bunting" and "Gray-crowned Rosy-Finch", both prioritize "brown color". However, the Finch places greater importance on "black-white wings with brown color", whereas a "blue color" concept plays a significant role (only) for the Lazuli Bunting.

D.6. Comparing Single Predictions with Prototypes

As detailed in Section 4.3 and Figure 1, we can compare a single prediction and its explanation with a prototype to understand how ordinary a prediction is in a more objective manner. Specifically, we can compare differences in how and which concepts are used. These differences can be quantitatively measured via the log-likelihood as in Equation (4), enabling for an automatic detection of outliers, or to assign predictions to the closest prototypical prediction strategy.

We present examples for validating single predictions using prototypes with samples from the ImageNet dataset in

Table D.1. OOD detection results for (VGG|ResNet|EfficientNet) models trained on **CIFAR-10**. Higher AUC scores are better.

	LSUN	iSUN	Textures	SVHN	Average
MSP [5]	90.9 | 83.4 | 86.0	87.3 | 79.2 | 87.5	90.9 | 81.9 | 83.1	90.4 | 82.0 | 83.0	85.5
Energy [10]	94.3 | 91.1 | 91.5	91.2 | 84.2 | 92.3	94.0 | 86.2 | 86.2	91.4 | 85.7 | 85.9	89.5
Mahalanobis [9]	49.0 | 43.0 | 76.3	57.1 | 61.2 | 84.5	81.5 | 55.7 | 71.5	68.9 | 53.0 | 74.9	64.7
PCX-E (ours)	88.2 | 76.0 | 81.7	81.7 | 75.9 | 88.4	91.0 | 73.9 | 78.7	89.4 | 65.7 | 80.4	80.9
PCX-GMM (ours)	94.3 | 83.7 | 86.1	88.7 | 80.2 | 91.1	95.8 | 81.6 | 83.0	94.8 | 83.0 | 84.2	87.2

Table D.2. OOD detection results for (VGG|ResNet|EfficientNet) models trained on **ImageNet**. Higher AUC scores are better.

	iSUN	Places365	Textures	CIFAR-10	Average
MSP [5]	96.3 | 93.2 | 99.1	96.7 | 95.9 | 99.5	96.0 | 95.3 | 99.0	93.5 | 88.9 | 97.4	95.9
Energy [10]	99.9 | 99.9 | 99.8	99.8 | 99.8 | 99.8	99.7 | 99.5 | 99.6	99.6 | 99.3 | 98.6	99.6
Mahalanobis [9]	38.8 | 98.7 | 99.6	88.8 | 94.5 | 98.3	86.3 | 90.5 | 99.2	19.6 | 99.0 | 98.8	84.3
PCX-E (ours)	98.7 | 99.2 | 99.2	98.3 | 98.1 | 98.3	99.3 | 99.0 | 99.7	99.3 | 98.8 | 98.0	98.8
PCX-GMM (ours)	98.3 | 98.9 | 99.5	99.3 | 99.0 | 98.9	99.7 | 99.3 | 99.8	99.1 | 98.7 | 98.6	99.1

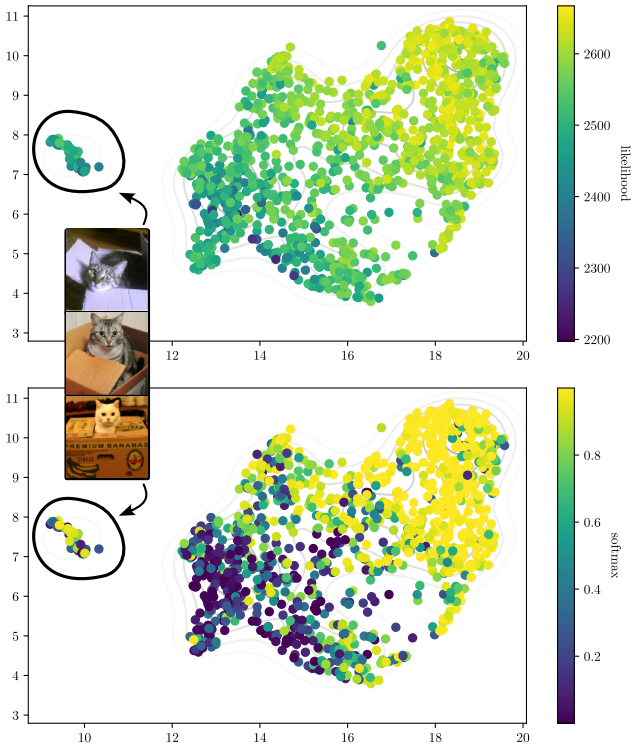

Figure D.4. UMAP embeddings of the ImageNet class "carton" using concept relevances of a VGG-16 model and layer `features.28`. (*top*): For each prediction, we plot the class log-likelihood when modeling the distribution with one Gaussian. (*bottom*): For each prediction, we plot the softmax value for the "carton" class.

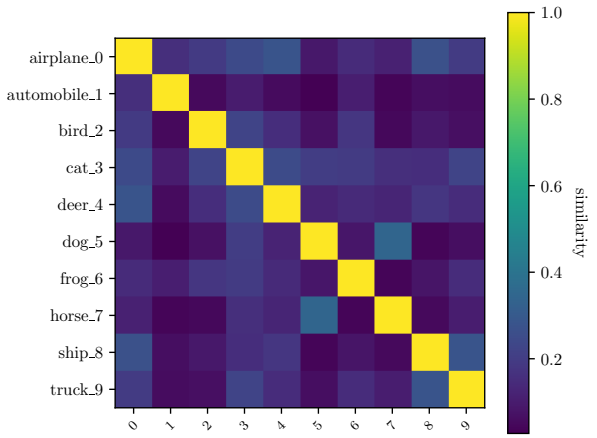

Figure D.5. Class similarity matrix between class prototypes (one per class) for the CIFAR-10 dataset and VGG-16 on layer `features.28`.

Figures D.10, D.11, D.12 and D.13. Here, we show correct predictions that have a low log-likelihood compared to samples of the same class in the training set, as illustrated in the plot in the first column and third row of each figure. Each sample is compared to the most similar prototype (of six in total). We further show three concepts that are most relevant for either prediction or prototype, followed by two concepts where the differences in terms of concept relevance scores between sample and prototype are largest.

In the first example in Figure D.10, an ostrich is correctly predicted by a VGG-16 model. The ostrich's head and part of the neck is visible, mimicking the prototype. However, the test prediction differs in a fence, behind which the ostrich is depicted. By studying the concept relevance scores, we can understand that the fence was highly relevant for the

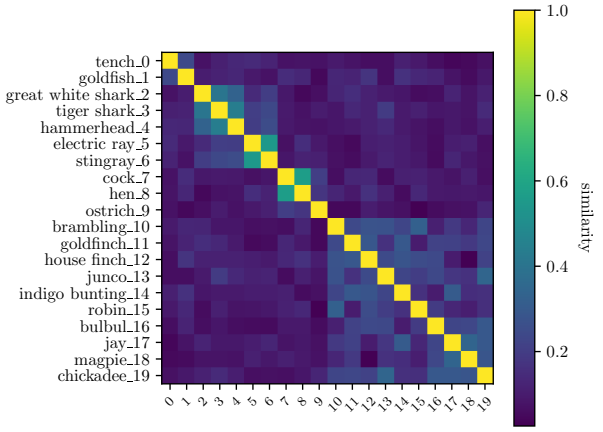

Figure D.6. Class similarity matrix between class prototypes (one per class) for the ImageNet dataset (first 20 classes) and EfficientNet-B0 on the last convolutional `features` block.

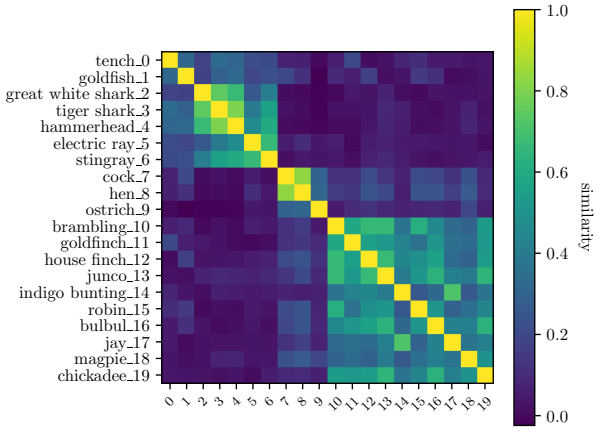

Figure D.7. Class similarity matrix between class prototypes (one per class) for the ImageNet dataset (first 20 classes) and ResNet-18 on the last convolutional `BasicBlock`.

prediction, which is not the case for the prototype. Further, concepts regarding the head's shape are missing.

A second example is shown in Figure D.11, where a Tiger Cat is correctly predicted by a ResNet-18 model. Here, the test sample shows an orange Tiger Cat lying (sleeping) on a blanket with jaguar fur pattern. Notably, the most similar prototype is not a Tiger Cat, but a tiger, as already observed in Figure C.7 (*bottom*) due to a high number of tiger samples in the training dataset with class label "Tiger Cat". For both sample and prototype, a "stripes-texture" concept is relevant, however, the test sample deviates in a stronger use of cat- and jaguar-like concepts (concepts 298 and 253). On the other side, underrepresented are concepts associated with cat eyes and heads (concept 156), which can be expected, as the eyes of the cat in the test prediction are closed.

The third example is shown in Figure D.12, where a bald eagle is correctly predicted by a ResNet-18 model. Here the eagle is depicted as an exhibit hanging in a room, whereas the prototype shows a close-up picture of an eagle's head. While concepts related to the feathers (concepts 245 and 352) are relevant for both, concepts related to the environment (concept 182) or yellow beak (concept 141) are missing.

A last example is shown in Figure D.13, where an ambulance is correctly predicted by a VGG-16 model. Here, the test input has very low resolution and shows an ambulance car from the side. The prototype shows an ambilance car from the side as well, but in higher resolution. By comparing the used concepts, we can understand that the model strongly uses features related to blur (concept 206) for the outlier prediction, which are not apparent for the prototype. Further, compared to the prototype, concepts regarding side of buses with black windows (concept 25) are missing.

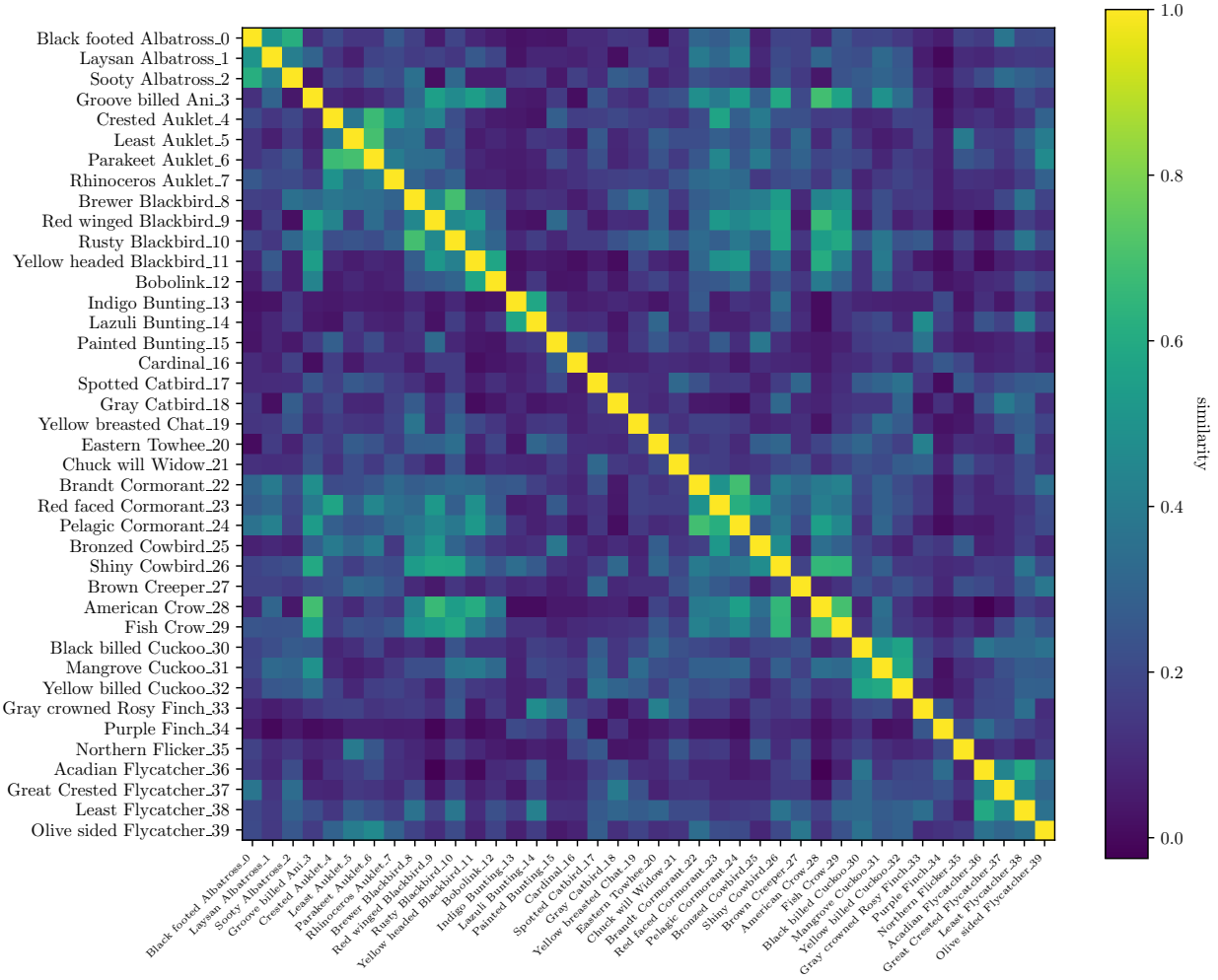

Figure D.8. Class similarity matrix between class prototypes (one per class) for the CUB-200 dataset (first 40 classes) and VGG-16 on layer `features.28`.

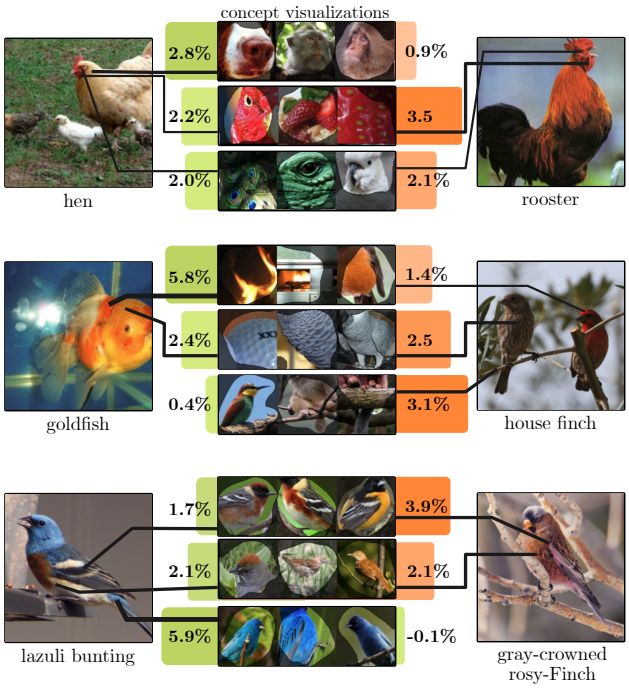

Figure D.9. Studying class (dis-)similarities across classes in terms of concepts found in layer `features.28` for a VGG-16 model. (*top*): ImageNet "hen" compared to "rooster" prototype. Whereas the "eye" concept is used for both equally, the "red color" concept is more important for the rooster class. For the hen, the brown-gray color is more important. (*middle*): ImageNet "goldfish" compared to "House Finch" prototype. Whereas for both, the "scales" texture is important, the orange color is much more important for the goldfish. On the other side, the "animal on branch" concept is only used for the House Finch. (*bottom*): CUB-200 "Lazuli Bunting" compared to "Gray-crowned Rosy-Finch" prototype. Wheras for both "brown color" is important, "black-white wings with brown color" is more relevant for the Finch. A "blue color" concept is only relevant for the Lazuli Bunting.

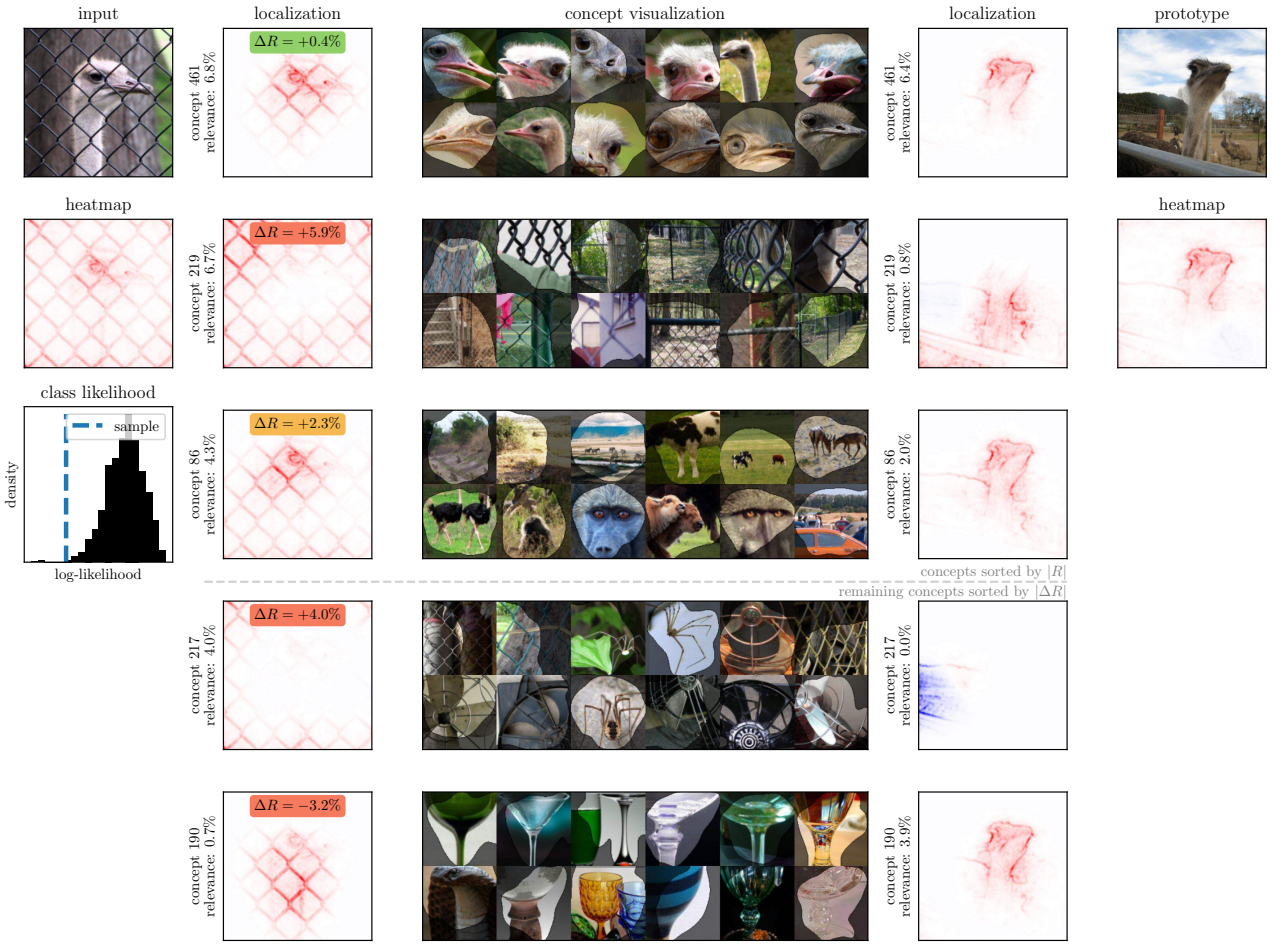

Figure D.10. Comparing a single prediction (*left*) with a prototype (*right*) allows to validate predictions in more objective manner. Here, an ostrich (ImageNet class "ostrich") is correctly predicted by a VGG-16 model, but is detected as an outlier based on the likelihood measure as defined in Equation (4) (illustrated in the plot in the first column and third row). In this example, we compare against the most similar prototype (of six) based on concept relevance scores R from layer `features.28`. By comparing the used concepts, we can understand that the model strongly uses fence features for the outlier prediction, which are not apparent for the prototype. Compared to the prototype, concepts regarding the head's shape are missing.

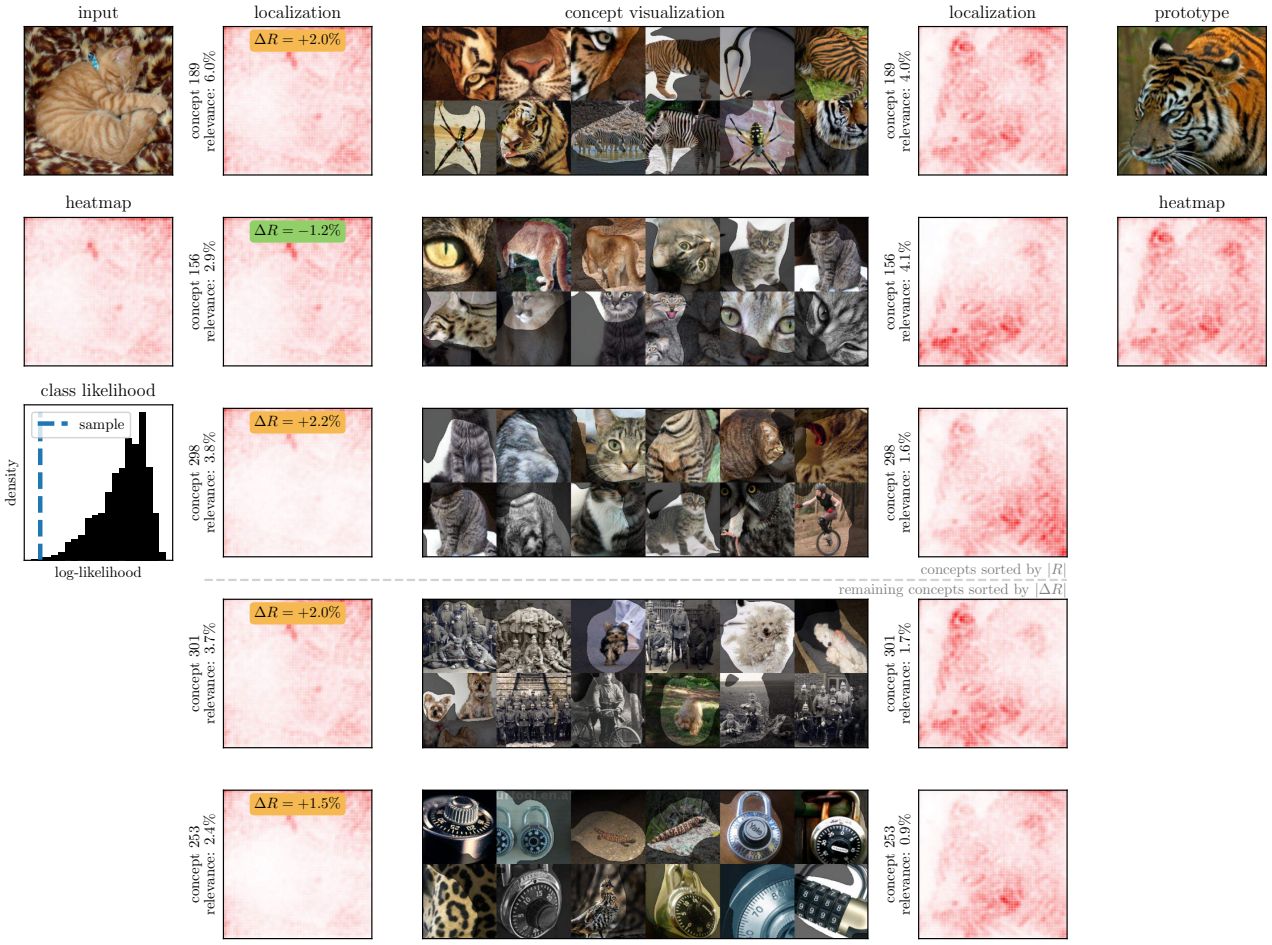

Figure D.11. Comparing a single prediction (*left*) with a prototype (*right*) allows to validate predictions in more objective manner. Here, a Tiger Cat (ImageNet class "Tiger Cat") is correctly predicted by a ResNet-18 model, but is detected as an outlier based on the likelihood measure as defined in Equation (4) (illustrated in the plot in the first column and third row). In this example, we compare against the most similar prototype (of six) based on concept relevance scores R from the last `BasicBlock`. The test sample shows an orange Tiger Cat lying on a blanket with jaguar fur pattern. Interestingly, the most similar prototype is not a Tiger Cat, but a tiger, as already observed in Figure C.7 (*bottom*) due to a high number of tiger samples in the training dataset with class label "Tiger Cat". By comparing the used concepts, we can understand that the sample and prototype are similar in a "stripes-texture" concept. However, the sample deviates in a stronger use of cat- and jaguar-like concepts (concepts 298 and 253). Underrepresented are concepts associated with cat eyes and heads (concept 156), which is sensible, as the eyes of the cat in the test prediction are closed.

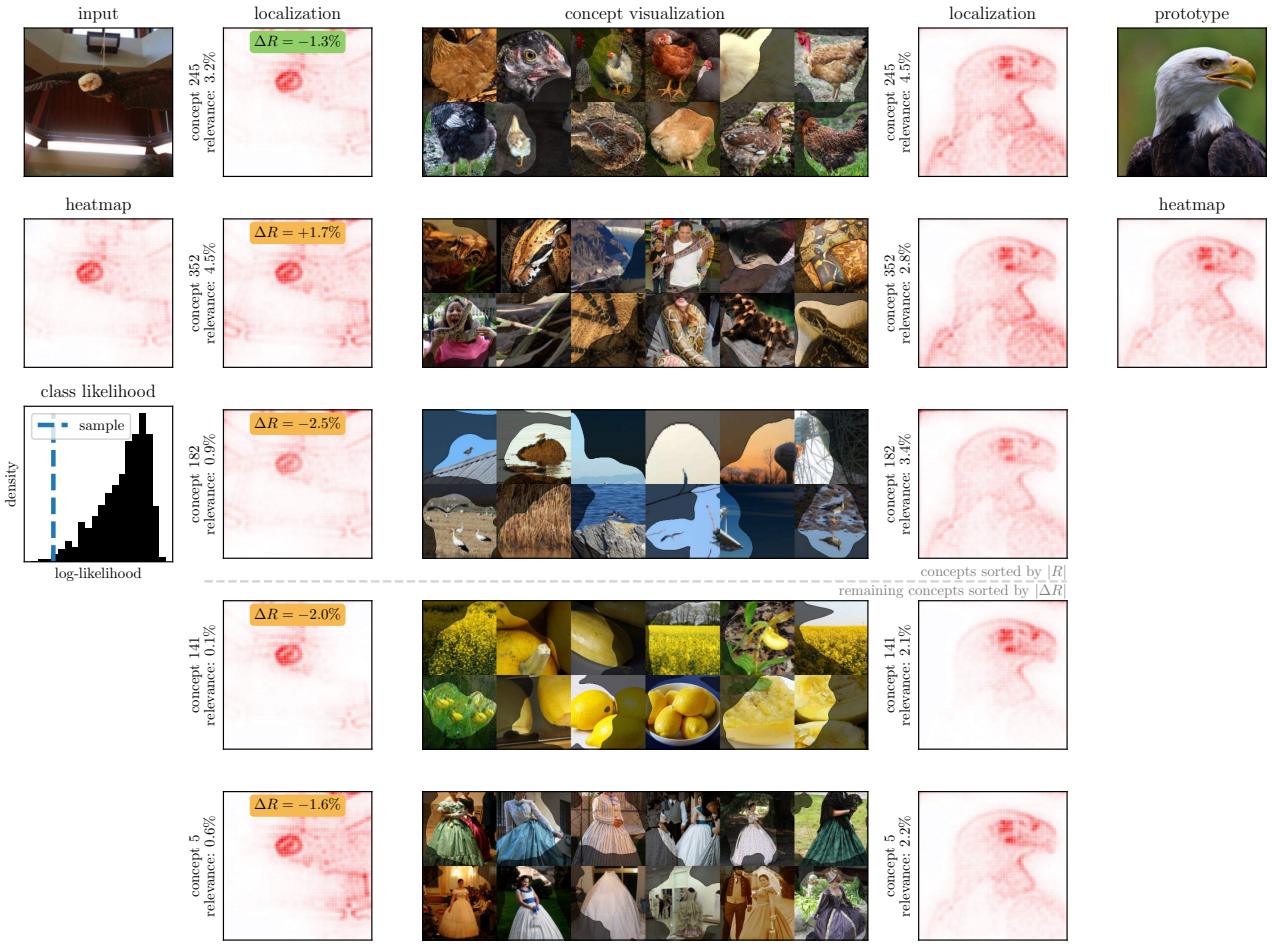

Figure D.12. Comparing a single prediction (*left*) with a prototype (*right*) allows to validate predictions in more objective manner. Here, a bald eagle (ImageNet class "bald eagle") is correctly predicted by a ResNet-18 model, but is detected as an outlier based on the likelihood measure as defined in Equation (4) (illustrated in the plot in the first column and third row). In this example, we compare against the most similar prototype (of six) based on concept relevance scores R from the last `BasicBlock`. Whereas concepts related to the feathers (concepts 245 and 352) are relevant for both, concepts related to the environment (concept 182) or yellow beak (concept 141) are missing.

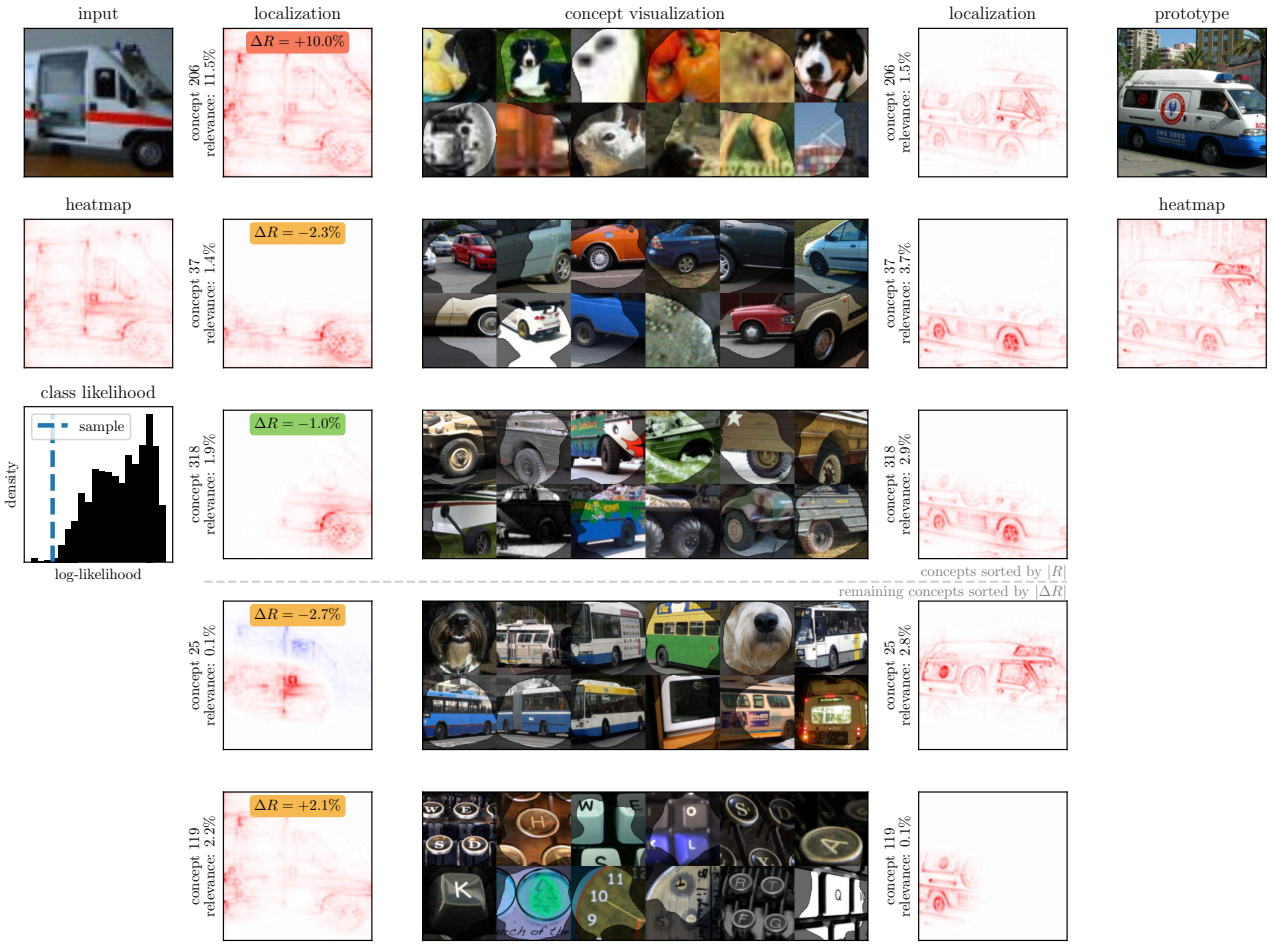

Figure D.13. Comparing a single prediction (*left*) with a prototype (*right*) allows to validate predictions in more objective manner. Here, an ambulance car (ImageNet class "ambulance") is correctly predicted by a VGG-16 model, but is detected as an outlier based on the likelihood measure as defined in Equation (4) (illustrated in the plot in the first column and third row). In this example, we compare against the most similar prototype (of six) based on concept relevance scores R from layer `features.28`. By comparing the used concepts, we can understand that the model strongly uses features related to blur (concept 206) for the outlier prediction, which are not apparent for the prototype. Compared to the prototype, concepts regarding side of buses with black windows (concept 25) are missing.

References

[1] Reduan Achtibat, Maximilian Dreyer, Ilona Eisenbraun, Sebastian Bosse, Thomas Wiegand, Wojciech Samek, and Sebastian Lapuschkin. From attribution maps to human-understandable explanations through concept relevance propagation. *Nature Machine Intelligence*, 5(9):1006–1019, 2023. 7

[2] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015. 3

[3] David Balduzzi, Marcus Frean, Lennox Leary, JP Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In *International Conference on Machine Learning*, pages 342–350. PMLR, 2017. 6

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1

[5] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2016. 19, 21

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 1

[7] Maximilian Kohlbrenner, Alexander Bauer, Shinichi Nakajima, Alexander Binder, Wojciech Samek, and Sebastian Lapuschkin. Towards best practice in explaining neural network decisions with lrp. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2020. 6

[8] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 1

[9] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. *Advances in neural information processing systems*, 31, 2018. 19, 21

[10] Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. Energy-based out-of-distribution detection. *Advances in neural information processing systems*, 33:21464–21475, 2020. 19, 21

[11] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017. 7

[12] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, 2018. 7

[13] Grégoire Montavon, Alexander Binder, Sebastian Lapuschkin, Wojciech Samek, and Klaus-Robert Müller. Layer-wise relevance propagation: an overview. *Explainable AI: interpreting, explaining and visualizing deep learning*, pages 193–209, 2019. 3, 6

[14] 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. 7

[15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 1

[16] 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. 7

[17] Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In *International conference on machine learning*, pages 3145–3153. PMLR, 2017. 3

[18] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015. 1

[19] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin A. Riedmiller. Striving for simplicity: The all convolutional net. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015. 3, 5

[20] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 1

[21] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and Pietro Perona. Caltech-UCSD birds 200. 2010. 1