Supplementary Material: Synthesized Classifiers for Zero-Shot Learning

Soravit Changpinyo*, Wei-Lun Chao*
U. of Southern California
Los Angeles, CA
schangpi, weilunc@usc.edu

Boqing Gong
U. of Central Florida
Orlando, FL
bgong@crcv.ucf.edu

Fei Sha
U. of California
Los Angeles, CA
feisha@cs.ucla.edu

In this Supplementary Material, we provide details omitted in the main text.

- Section 1: cross-validation strategies (Section 3.2 of the main paper).
- Section 2: learning metrics for semantic similarity (Section 3.1 of the main paper).
- Section 3: details on experimental setup (Section 4.1 of the main paper).
- Section 4: implementation details (Section 4.1 and 4.2.3 of the main paper).
- Section 5: additional experimental results and analyses (Section 4.2 of the main paper).

1. Cross-validation (CV) strategies

There are a few free hyper-parameters in our approach (cf. Section 3.2 of the main text). To choose the hyper-parameters in the conventional cross-validation (CV) for multi-way classification, one splits the training data into several folds such that they share the same set of class labels with one another. Clearly, this strategy is not sensible for zero-shot learning as it does not imitate what actually happens at the test stage. We thus introduce a new strategy for performing CV, inspired by the hyper-parameter tuning in [25]. The key difference of the new scheme to the conventional CV is that we split the data into several folds such that the class labels of these folds are disjoint. For clarity, we denote the conventional CV as *sample*-wise CV and our scheme as *class*-wise CV. Figure 1(b) and 1(c) illustrate the two scenarios, respectively. We empirically compare them in Section 5.1. Note that several existing models [2, 7, 25, 34] also follow similar hyper-prameter tuning procedures.

1.1. Learning semantic embeddings

We propose an optimization problem for learning semantic embeddings in Section 3.2 of the main text. There are four hyper-parameters λ, σ, η, and γ to be tuned. To reduce the search space during cross-validation, we first fix $\boldsymbol{b}_r = \boldsymbol{a}_r$ for $r = 1, \ldots, \mathsf{R}$ and tune λ, σ. Then we fix λ and σ and tune η and γ.

2. Learning metrics for computing similarities between semantic embeddings

Recall that, in Section 3.1 of the main text, the weights in the bipartite graph are defined based on the distance $d(\boldsymbol{a}_c, \boldsymbol{b}_r) = (\boldsymbol{a}_c - \boldsymbol{b}_r)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{a}_c - \boldsymbol{b}_r)$. In this section, we describe an objective for learning a more general Mahalnobis metric than $\Sigma^{-1} = \sigma^2 \boldsymbol{I}$. We focus on the case when $\mathsf{R} = \mathsf{S}$ and on learning a diagonal metric $\Sigma^{-1} = \boldsymbol{M}^T \boldsymbol{M}$, where $\boldsymbol{M}$ is also diagonal. We solve the following optimization problem.

$$\min_{\boldsymbol{M}, \boldsymbol{v}_1, \cdots, \boldsymbol{v}_\mathsf{R}} \sum_{c=1}^{\mathsf{S}} \sum_{n=1}^{\mathsf{N}} \ell(\boldsymbol{x}_n, \mathbb{I}_{y_n, c}; \boldsymbol{w}_c) \quad (1)$$

$$+ \frac{\lambda}{2} \sum_{r=1}^{\mathsf{R}} \|\boldsymbol{v}_r\|_2^2 + \frac{\gamma}{2} \|\boldsymbol{M} - \sigma \boldsymbol{I}\|_F^2, \quad (2)$$

$$\text{s.t.} \quad \boldsymbol{w}_c = \sum_{r=1}^{\mathsf{R}} s_{cr} \boldsymbol{v}_r, \quad \forall\, c \in \mathcal{T} = \{1, \cdots, \mathsf{S}\}$$

where $\ell(\boldsymbol{x}, y; \boldsymbol{w}) = \max(0, 1 - y \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x})^2$ is the squared hinge loss. The indicator $\mathbb{I}_{y_n, c} \in \{-1, 1\}$ denotes whether or not $y_n = c$.

Again, we perform alternating optimization for minimizing the above objective function. At first, we fix $\boldsymbol{M} = \sigma \boldsymbol{I}$ and optimize $\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_\mathsf{R}\}$ to obtain a reasonable initialization. Then we perform alternating optimization. To further prevent over-fitting, we alternately

[*] Equal contributions

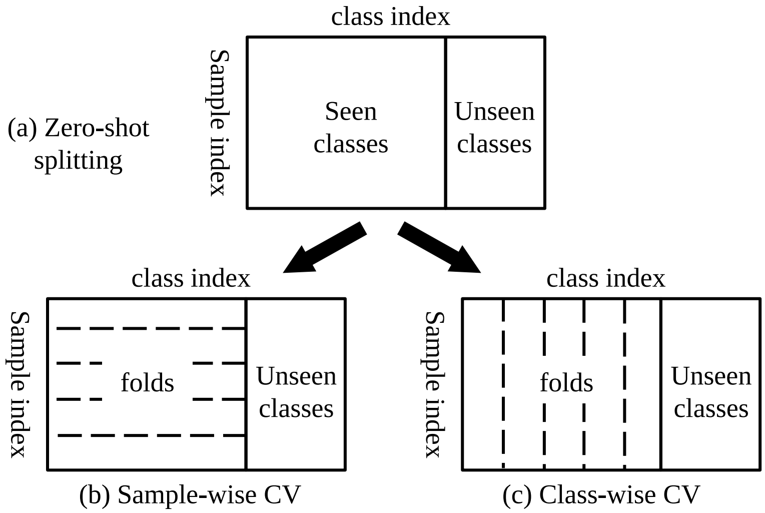

Figure 1: Data splitting for different cross-validation (CV) strategies: (a) the seen-unseen class splitting for zero-shot learning, (b) the sample-wise CV, (c) the class-wise CV (cf. Section 3.2 of the main paper).

optimize $\boldsymbol{M}$ and $\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_\mathsf{R}\}$ on different but overlapping subsets of training data. In particular, we split data into 5 folds and optimize $\{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_\mathsf{R}\}$ on the first 4 folds and $\boldsymbol{M}$ on the last 4 folds. We report results in Section 5.5.

3. Details on the experimental setup

We present details on the experimental setup in this section and additional results in Section 5.

3.1. Datasets

We use four benchmark datasets in our experiments. The **Animals with Attributes (AwA)** dataset [18] consists of 30,475 images of 50 animal classes. Along with the dataset, a standard data split is released for zero-shot learning: 40 seen classes (for training) and 10 unseen classes. The second dataset is the **CUB-200-2011 Birds (CUB)** [30]. It has 200 bird classes and 11,788 images. We randomly split the 200 classes into 4 disjoint sets (each with 50 classes) and treat each of them as the unseen classes in turn. We report the average results from the four splits. The **SUN Attribute (SUN)** dataset [23] contains 14,340 images of 717 scene categories (20 images from each category). Following [18], we randomly split the 717 classes into 10 disjoint sets (each with 71 or 72 classes) in a similar manner to the class splitting on CUB. We note that some previous published results [13, 25, 33, 34] are based on a simpler setting with 707 seen and 10 unseen classes. For comprehensive experimental comparison, we also report our results on this setting in Table 3.

For the large-scale zero-shot experiment on the **ImageNet** dataset [5], we follow the setting in [9, 22]. The ILSVRC 2012 1K dataset [26] contains 1,281,167 training and 50,000 validation images from 1,000 categories and is treated as the seen-class data. Images of unseen classes come from the rest of the ImageNet Fall 2011 release dataset [5] that do not overlap with any of the 1,000 categories. We will call this release the ImageNet 2011 21K dataset (as in [9, 22]). Overall, this dataset contains 14,197,122 images from 21,841 classes, and we conduct our experiment on **20,842 unseen classes**[1].

3.2. Semantic spaces

SUN Each image is annotated with 102 continuous-valued attributes. For each class, we average attribute vectors over all images belonging to that class to obtain a class-level attribute vector.

ImageNet We train a skip-gram language model [20, 21] on the latest Wikipedia dump corpus[2] (with more than 3 billion words) to extract a 500-dimensional word vector for each class. In particular, we train the model using the word2vec package[3], generate word vectors directly for the single-term class names, and use the

[1] There is one class in the ILSVRC 2012 1K dataset that does not appear in the ImageNet 2011 21K dataset. Thus, we have a total of 20,842 unseen classes to evaluate.

[2] http://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2 on September 1st, 2015

[3] https://code.google.com/p/word2vec/

word2phrase function to obtain word vectors for multi-term class names, including those terms in the ImageNet *synsets*[4]. We impose no restriction on the vocabulary size. Following [9], we use a window size of 20, apply the hierarchical softmax for predicting adjacent terms, and train the model with a single pass through the corpus. As one class may correspond to multiple word vectors by the nature of synsets, we simply average them to form a single word vector for each class. We obtain word vectors for all the 1,000 seen classes and for **20,345** (out of 20,842) unseen classes. We ignore classes without word vectors in the experiments.

3.3. Visual features

We denote features that are not extracted by deep learning as shallow features.

Shallow features On **AwA**, many existing approaches take traditional features such as color histograms, SIFT, and PHOG that come with the dataset [18, 24, 31], while others use the Fisher vectors [1, 2]. The **SUN** dataset also comes with several traditional shallow features, which are used in [13, 18, 25].

In our experiments, we use the shallow features provided by [18] , [14], and [23] for **AwA**, **CUB**, and **SUN**, respectively, unless stated otherwise.

Deep features Given the recent impressive success of deep Convolutional Neural Networks (CNNs) [17] on image classification, we conduct experiments with deep features on all datasets. We use the Caffe package [15] to extract AlexNet [17] and GoogLeNet [29] features for images from **AwA** and **CUB**. Observing that GoogLeNet give superior results over AlexNet on **AwA** and **CUB**, we focus on GoogLeNet features on large datasets: **SUN** and **ImageNet**. These networks are pre-trained on the ILSVRC 2012 1K dataset [5, 26] for **AwA**, **CUB**, and **ImageNet**. For **SUN**, the networks are pre-trained on the Places database [35], which was shown to outperform the networks pre-trained on ImageNet on scene classification tasks. For AlexNet, we use the 4,096-dimensional activations of the penultimate layer (fc7) as features, and for GoogLeNet we extract features by the 1,024-dimensional activations of the pooling units following the suggestion by [2].

For **CUB**, we crop all images with the provided bounding boxes. For **ImageNet**, we center-crop all images and do not perform any data augmentation or other preprocessing.

3.4. Evaluation protocols on ImageNet

When computing Hierarchical precision@K (HP@K), we use the algorithm in the Appendix of [9] to compute a set of at least K classes that are considered to be correct. This set is called $hCorrectSet$ and it is computed for each K and class c. See Algorithm 1 for more details. The main idea is to expand the radius around the true class c until the set has at least K classes.

Algorithm 1 Algorithm for computing $hCorrectSet$ for $H@K$ [9]

```
Input: $K$, class $c$, ImageNet hierarchy
$hCorrectSet \leftarrow \emptyset$
$R \leftarrow 0$
**while** NumberElements($hCorrectSet$) $< K$ **do**
    $radiusSet \leftarrow$ all nodes in the hierarchy which are $R$ hops from $c$
    $validRadiusSet \leftarrow$ ValidLabelNodes($radiusSet$)
    $hCorrectSet \leftarrow hCorrectSet \cup validRadiusSet$
    $R \leftarrow R + 1$
**end while**
**return** $hCorrectSet$
```

Note that $validRadiusSet$ depends on which classes are in the label space to be predicted (i.e., depending on whether we consider *2-hop*, *3-hop*, or *All*. We obtain the label sets for *2-hop* and *3-hop* from the authors of [9, 22]. We implement Algorithm 1 to derive $hCorrectSet$ ourselves.

4. Implementation details

4.1. How to avoid over-fitting?

Since during training we have access only to data from the seen classes, it is important to avoid over-fitting to those seen classes. We apply the *class-wise* cross-validation strategy (Section 1), and restrict the semantic embeddings of phantom classes to be close to the semantic embeddings of seen classes (Section 3.2 of the main text).

4.2. Combination of attributes and word vectors

In Table 5 of the main text and Section 5.2 of this material, we combine attributes and word vectors to improve the semantic embeddings. We do so by first computing s_{rc} in eq. (2) of the main text for different semantic sources, and then perform convex combination on s_{rc} of different sources to obtain the final s_{rc}. The combining weights are determined via cross-validation.

[4] Each class of **ImageNet** is a *synset*: a set of synonymous terms, where each term is a word or phrase.

4.3. Initialization

All variables are randomly initialized, unless stated otherwise. Other details on initialization can be found in Section 3.2 of the main text and Section 2 and 4.5 of this material.

4.4. ConSE [22]

Instead of using the CNN 1K-class classifiers directly, we train (regularized) logistic regression classifiers using recently released multi-core version of LIBLINEAR [8]. Furthermore, in [22], the authors use the averaged word vectors for seen classes, but keep for each unseen class the word vectors of all synonyms. In other words, each unseen class can be represented by multiple word vectors. In our implementation, we use averaged word vectors for both seen and unseen classes for fair comparison.

4.5. Varying the number of base classifiers

In Section 4.2.3 and Figure 2 of the main text, we examine the use of different numbers of base classifiers (i.e., R). The semantic embedding $\boldsymbol{b}_r$ of the phantom classes are set equal to $\boldsymbol{a}_r, \forall r \in \{1, \cdots, \mathsf{R}\}$ at 100% (i.e., $\mathsf{R} = \mathsf{S}$). For percentages smaller than 100%, we perform K-means and set $\boldsymbol{b}_r$ to be the cluster centroids after ℓ_2 normalization (in this case, $\mathsf{R} = K$). For percentages larger than 100%, we set the first S $\boldsymbol{b}_r$ to be $\boldsymbol{a}_r$, and the remaining $\boldsymbol{b}_r$ as the random combinations of $\boldsymbol{a}_r$ (also with ℓ_2 normalization on $\boldsymbol{b}_r$).

5. Additional experimental results and analyses

We present in this section some additional experimental results on zero-shot learning. *Unless stated otherwise, we focus on learning with the one-versus-other loss (cf. eq. (5) of the main text).*

5.1. Cross-validation (CV) strategies

Table 1 shows the results on **CUB** (averaged over four splits) using the hyper-parameters tuned by class-wise CV and sample-wise CV, respectively. The results based on class-wise CV are about 2% better than those of sample-wise CV, verifying the necessity of simulating the zero-shot learning scenario while we tune the hyper-parameters at the training stage.

Table 1: Comparison between sample- and class-wise cross-validation for hyper-parameter tuning on **CUB** (learning with the one-versus-other loss).

CV Scenarios	**CUB** (AlexNet)	**CUB** (GoogLeNet)
Sample-wise	44.7	52.0
Class-wise	46.6	53.4

5.2. Additional comparison of different semantic spaces for embedding classes

Our method for synthesizing classifiers accepts different semantic embedding spaces. We expand our results on Table 5 of the main text to include AlexNet features as well. The results are in Table 2. We use word vectors provided by Fu et al. [10, 11], which are of 100 and 1000 dimensions per class, respectively. We see that the two types of features, AlexNet and GoogLeNet, demonstrate very similar trends. First, higher-dimensional word vectors often give rise to better performance. Second, human-annotated attributes outperform automatically-learned word vectors. Third, combining the word vectors and the attributes leads to better results than separately using either one of them.

5.3. Comparison to other state-of-the-art methods

In Table 3, we contrast our methods to several other state-of-the-art methods, in addition to Table 2 of the main text. We note subtle differences in the experiment setup of some of these methods from ours:

- TMV-BLP and TMV-HLP [10, 11]. These methods focus on the transductive setting, where they have access to unlabeled test data from unseen classes during the training stage. Additionally, they use OverFeat [27] features for **CUB**, OverFeat+DeCAF [6] for **AwA**, and both attributes and word vectors for class embeddings.

- [16]. This method works on the transductive setting. It uses OverFeat features for both **AwA** and **CUB**, and combines attributes and word vectors for class embeddings.

- [19] This method works on the semi-supervised setting, where a portion of unlabeled data (not used for testing) from unseen classes are available at training.

- HAT-n [3]. This method uses extra semantic information (WordNet class hiearchy). It uses CNN-M2K features [4] and extra cropped images.

- AMP (SR+SE) [12]. This method uses attributes

Table 2: Comparison between different semantic embedding spaces for our approach.

Methods	Semantic embeddings	Dimensions	Features	**AwA**
Ours$^{\text{o-vs-o}}$	word vectors	100	AlexNet	37.6
Ours$^{\text{o-vs-o}}$	word vectors	1000	AlexNet	52.4
Ours$^{\text{o-vs-o}}$	attributes	85	AlexNet	64.0
Ours$^{\text{o-vs-o}}$	attributes + word vectors	85 + 100	AlexNet	65.6
Ours$^{\text{o-vs-o}}$	attributes + word vectors	85 + 1000	AlexNet	**68.0**
SJE[2]	word vectors	400	GoogLeNet	51.2
Ours$^{\text{o-vs-o}}$	word vectors	100	GoogLeNet	42.2
Ours$^{\text{o-vs-o}}$	word vectors	1000	GoogLeNet	57.5
Ours$^{\text{o-vs-o}}$	attributes	85	GoogLeNet	69.7
Ours$^{\text{o-vs-o}}$	attributes + word vectors	85 + 100	GoogLeNet	73.2
Ours$^{\text{o-vs-o}}$	attributes + word vectors	85 + 1000	GoogLeNet	**76.3**

and (100-dimensional) word vectors and OverFeat features in their experiments.

- [32]. This method focuses on mining/discovering new (category-level) attributes. It requires extra human efforts to annotate the new attributes for unseen classes.

- [13]. The best result on **AwA** presented in this paper uses the discovered attributes in [32].

As shown in Table 3, our method outperforms all of them on the dataset **CUB** despite the fact that they employ extra images or semantic embedding information.

5.3.1 SUN-10

Some existing work [13, 25, 33, 34] considers another setting for **SUN** dataset — with 707 seen classes and 10 unseen classes. Moreover, [25, 33, 34] use the VGG-verydeep-19 [28] CNN features. In Table 3, we provide results of our approach based on this splitting. Compared to previously published results, our method again clearly shows superior performance.

5.4. Discussion on the numbers of seen and unseen classes

In this subsection, we analyze the results under different numbers of seen/unseen classes in performing zero-shot learning using the **CUB** dataset.

5.4.1 Varying the number of seen classes

We first examine the performance of zero-shot learning under different numbers of seen classes (e.g., 50, 100, and 150) while fixing the number of unseen classes to be 50. We perform 20 random selections of seen/unseen classes. Unsurprisingly, Table 4 shows that increasing the number of seen classes in training leads to improved performance on zero-shot learning.

Table 4: Performance of our method under different number S of seen classes on **CUB**. The number of unseen classes U is fixed to be 50.

U and S	S = 50	S = 100	S = 150
U = 50	38.4	49.9	53.8

5.4.2 Varying the number of unseen classes

We then examine the performance of our approach to zero-shot learning under different numbers of unseen classes (e.g., within [0, 150]), with the number of seen classes fixed to be 50 during training. We again conduct experiments on **CUB**, and perform 20 random selections of seen/unseen classes. The results are presented in Figure 2. We see that the accuracy drops as the number of unseen classes increases.

5.5. Results on learning metrics for computing semantic similarities

We improve our method by also learning metrics for computing semantic similarity. Please see Section 2 for more details. Preliminary results on **AwA** in Table 5 suggest that learning metrics can further improve upon our current one-vs-other formulation.

Table 3: Comparison between our method and other state-of-the-art methods for zero-shot learning. Our results are based on the GoogLeNet features and attributes or attributes + word vectors as semantic embeddings. See Section 5.3 for the details of other methods and **SUN-10**.

Methods	Shallow features			Deep features			
	AwA	**CUB**	**SUN-10**	**AwA**	**CUB**	**SUN**	**SUN-10**
TMV-BLP [10]	***47.7***	***16.3***†‡	-	***77.8***	***45.2***†	-	-
TMV-HLP [11]	***49.0***	***19.5***†‡	-	***80.5***	***47.9***†	-	-
[16]	***49.7***	***28.1***†‡	-	***75.6***	***40.6***†	-	-
[19]	***40.0***	-	-	-	-	-	-
HAT-n [3]	-	-	-	***68.8***	***48.6***†	-	-
AMP (SR+SE) [12]	-	-	-	***66.0***	-	-	-
[32]	***48.3***	-	-	-	-	-	-
[13]	***48.7***	-	56.2	-	-	-	-
ESZSL [25]	-	-	65.8	-	-	-	***82.1***
SSE-ReLU [34]	-	-	-	-	-	-	***82.5***
[33]	-	-	-	-	-	-	***83.8***
SJE[2]	-	-	-	-	-	-	87.0
Ours$^{\text{o-vs-o}}$	42.6	35.0	-	69.7	53.4	62.8	90.0
Ours$^{\text{cs}}$	42.1	34.7	-	68.4	51.6	52.9	87.0
Ours$^{\text{struct}}$	41.5	36.4	-	72.9	54.5	62.7	85.0
§Ours$^{\text{o-vs-o}}$	-	-	-	76.3	-	-	-

†: Results reported by the authors on a particular seen-unseen split.
‡: Based on Fisher vectors as shallow features.
§: Ours with attributes + (1000-dimensional) word vectors.

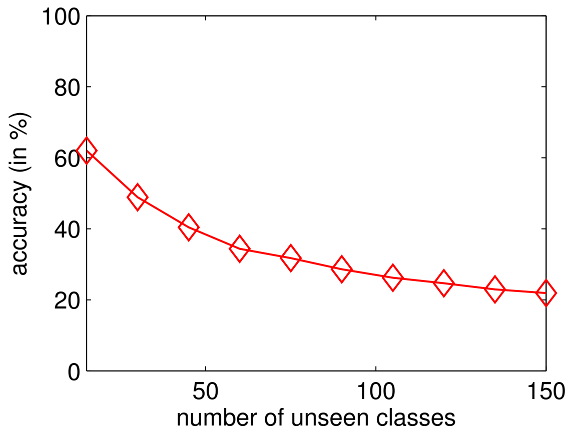

Figure 2: Performance of our method under different numbers of unseen classes on **CUB**. The number of seen classes is fixed to be 50.

Table 5: Effect of learning metrics for computing semantic similarity on **AwA**.

Dataset	Type of embeddings	w/o learning	w/ learning
AwA	attributes	69.7%	73.4%

5.6. Detailed results and analysis of experiments on ImageNet

Table 6 provides expanded zero-shot learning results on **ImageNet** (cf. Table 3 of the main text). Note that ConSE [22] has a free parameter T, corresponding to how many nearest seen classes to use for convex combination. In our implementation, we follow the paper to test on $T = 1$, 10, and 1,000. We further apply the class-wise cross-validation (cf. Section 1 of this material) to automatically set T. We also report the published best result in [22]. Our methods (Ours$^{\text{o-vs-o}}$ and Ours$^{\text{struct}}$) achieve the highest accuracy in most cases.

As mentioned in the main text, the three sets of unseen classes, *2-hop*, *3-hop*, and *All* are built according to the ImageNet label hierarchy. Note that they are not mutually exclusive. Indeed, *3-hop* contains all the classes in *2-hop*, and *All* contains all in *3-hop*. To examine if the semantic similarity/dissimilarity to the 1K seen classes (according to the label hierarchy) would affect the classification accuracy, we split *All* into three *disjoint* sets, *2-hop*, *pure 3-hop*, and *others*, which contain 1,509, 6,169, and 12,667 classes, respectively (totally 20,345). We then test on *All*, but report accuracies of images belonging to different *disjoint* sets separately. Figure 3 summarizes the results. Our method outperforms ConSE in almost all cases. The decreasing accuracies from *2-hop*, *pure 3-hop*, to *others* (by both methods) verify the high correlation of the semantic relationship to the classification accuracy in zero-shot learning. This observation suggests an obvious potential limitation: it is unrealistic to expect good performance on unseen classes that are semantically too dissimilar to seen classes.

5.7. Qualitative results

In this subsection, we present qualitative results of our method. We first illustrate what visual information the models (classifiers) for unseen classes capture, when provided with only semantic embeddings (no example images). In Figure 4, we list (on top) the 10 unseen class labels of **AwA**, and show (in the middle) the top-5 images classified into each class c, according to the decision values $\boldsymbol{w}_c^{\mathrm{T}}\boldsymbol{x}$ (cf. eq. (1) and (4) of the main text). Misclassified images are marked with red boundaries. At the bottom, we show the first (highest score) misclassified image (according to the decision value) into each class and its ground-truth class label. According to the top images, our method reasonably captures discriminative visual properties of each unseen class based solely on its semantic embedding. We can also see that the misclassified images are with appearance so similar to that of predicted class that even humans cannot easily distinguish between the two. For example, the pig image at the bottom of the second column looks very similar to the image of hippos. Figure 5 and Figure 6 present the results in the same format on **CUB** and **SUN**, respectively (both on a subset of unseen class labels).

We further analyze the success and failure cases; i.e., why an image from unseen classes is misclassified. The illustrations are in Figure 7, 8, and 9 for **AwA**, **CUB**, and **SUN**, respectively. In each figure, we consider **(Left)** one unseen class and show its convex combination weights $\boldsymbol{s}_c = \{s_{c1}, \cdots, s_{c\mathsf{R}}\}$ as a histogram. We then present **(Middle-Left)** the top-3 semantically similar (in terms of $\boldsymbol{s}_c$) seen classes and their most representative images. As our model exploits phantom classes to connect seen and unseen classes in both semantic and model spaces, we expect that the model (classifier) for such unseen class captures similar visual information as those for semantically similar seen classes do. **(Middle-Right)** We examine two images of such unseen class, where the top one is correctly classified; the bottom one, misclassified. We also list **(Right)** the top-3 predicted labels (within the pool of unseen classes) and their most representative images. Green corresponds to correct labels. We see that, in the misclassified cases, the test images are visually dissimilar to those of the semantically similar seen classes. The synthesized unseen classifiers, therefore, cannot correctly recognize them, leading to incorrect predictions.

5.8. Comparison between shallow and deep features of our approach

In Table 4 of the main text, our approach performs better with deep features than with shallow features compared to other methods. We propose explanations for this phenomenal. Deep features are learned hierarchically and expected to be more abstract and semantically meaningful. Arguably, similarities between them (measured in inner products between classifiers) might be more congruent with similarities computed in the semantic embedding space for combining classifiers. Additionally, shallow features have higher dimensions (around 10,000) than deep features (e.g., 1024 for GoogLeNet) so they might require more phantom classes to synthesize classifiers.

5.9. Analysis on the number of base classifiers

In Fig. 2 of the main text, we show that even by using fewer base (phantom) classifiers than the number of seen classes (e.g., around 60 %), we get comparable or even better results, especially for **CUB**. We surmise that this is because **CUB** is a fine-grained recognition benchmark and has higher correlations among classes, and provide analysis in Fig. 10 to justify this.

We train one-versus-other classifiers for each value of the regularization parameter (i.e., λ in eq. (5) of the main text) on both **AwA** and **CUB**, and then perform PCA on the resulting classifier matrices. We then plot the required number (in percentage) of PCA components to capture 95% of variance in the classifiers. Clearly, **AwA** requires more. This explains why we see the drop in accuracy for **AwA** but not **CUB** in Fig. 2 of the main text when using even fewer base classifiers. Particularly, the low percentage for **CUB** in Fig. 10 implies that fewer base classifiers are possible.

References

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for attribute-based classification. In *CVPR*, 2013. 3

[2] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele. Evaluation of output embeddings for fine-grained image classification. In *CVPR*, 2015. 1, 3, 5, 6

[3] Z. Al-Halah and R. Stiefelhagen. How to transfer? zero-shot object recognition via hierarchical transfer of semantic attributes. In *WACV*, 2015. 4, 6

Table 6: Flat Hit@K and Hierarchial precision@K performance (in %) on the task of zero-shot learning on **ImageNet**. We mainly compare it with ConSE(T) [22], where T is the number of classifiers to be combined in their paper. For ConSE(CV), T is obtained by class-wise CV. Lastly, the best published results in [22] are also reported, corresponding to ConSE(10) [22]. For both types of metrics, the higher the better.

Scenarios	Methods	Flat Hit@K					Hierarchical precision@K			
	K=	1	2	5	10	20	2	5	10	20
2-hop	ConSE(1)	9.0	12.9	20.8	28.3	38.1	21.1	22.2	24.8	28.1
	ConSE(10)	9.2	13.7	22.4	31.0	41.4	22.5	24.1	27.3	30.8
	ConSE(10) [22]	9.4	15.1	24.7	32.7	41.8	21.4	24.7	26.9	28.4
	ConSE(1000)	8.9	13.3	21.8	30.1	40.3	22.0	23.7	27.0	30.4
	ConSE(CV)	8.3	12.9	21.8	30.9	41.7	21.5	23.8	27.5	31.3
	$\text{Ours}^{\text{o-vs-o}}$	**10.5**	**16.7**	**28.6**	**40.1**	**52.0**	**25.1**	**27.7**	**30.3**	**32.1**
	$\text{Ours}^{\text{struct}}$	9.8	15.3	25.8	35.8	46.5	23.8	25.8	28.2	29.6
3-hop	ConSE(1)	2.8	4.2	7.2	10.1	14.3	6.2	18.4	20.4	22.1
	ConSE(10)	**2.9**	4.5	7.7	11.3	16.1	6.9	20.9	23.1	25.2
	ConSE(10) [22]	2.7	4.4	7.8	11.5	16.1	5.3	20.2	22.4	24.7
	ConSE(1000)	2.8	4.3	7.4	10.9	15.6	6.8	20.7	22.9	25.1
	ConSE(CV)	2.6	4.1	7.3	11.1	16.4	6.7	21.4	23.8	26.3
	$\text{Ours}^{\text{o-vs-o}}$	**2.9**	**4.9**	**9.2**	**14.2**	**20.9**	7.4	**23.7**	**26.4**	**28.6**
	$\text{Ours}^{\text{struct}}$	**2.9**	4.7	8.7	13.0	18.6	**8.0**	22.8	25.0	26.7
All	ConSE(1)	1.4	2.3	3.8	5.6	7.8	3.0	7.6	8.7	9.6
	ConSE(10)	**1.5**	2.3	4.0	6.0	8.7	3.3	8.9	10.2	11.4
	ConSE(10) [22]	1.4	2.2	3.9	5.8	8.3	2.5	7.8	9.2	10.4
	ConSE(1000)	**1.5**	2.3	3.9	5.8	8.4	3.2	8.8	10.2	11.3
	ConSE(CV)	1.3	2.1	3.8	5.8	8.7	3.2	9.2	10.7	12.0
	$\text{Ours}^{\text{o-vs-o}}$	1.4	**2.4**	**4.5**	**7.1**	**10.9**	3.1	9.0	10.9	**12.5**
	$\text{Ours}^{\text{struct}}$	**1.5**	**2.4**	4.4	6.7	10.0	**3.6**	**9.6**	**11.0**	12.2

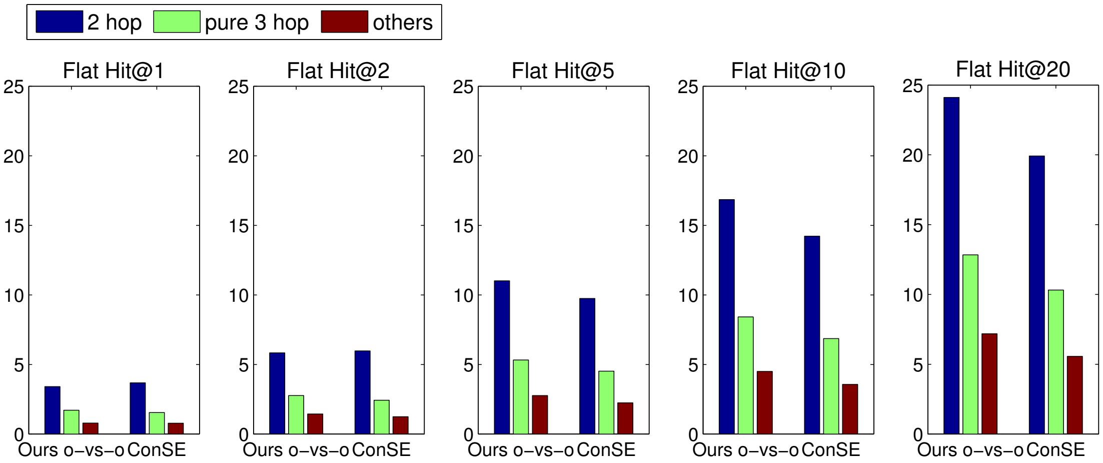

Figure 3: On the **ImageNet** dataset, we outperform ConSE (i.e., ConSE(10) of our implementation) on different *disjoint* sets of categories in the scenario *All* in almost all cases. See Section 5.6 of this material for details.

[4] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In *BMVC*, 2014. 4

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 2, 3

[6] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014. 4

[7] M. Elhoseiny, B. Saleh, and A. Elgammal. Write a classifier: Zero-shot learning using purely textual descriptions. In *ICCV*, 2013. 1

[8] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and

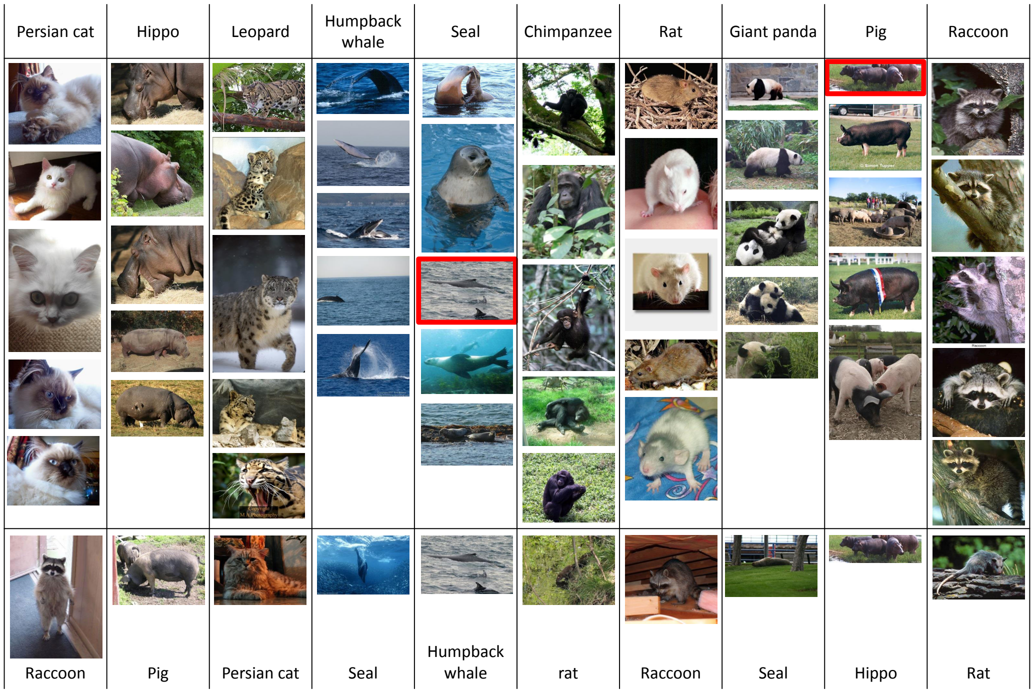

Figure 4: Qualitative results of our method (Ours$^{\text{struct}}$) on **AwA**. **(Top)** We list the 10 unseen class labels. **(Middle)** We show the top-5 images classified into each class, according to the decision values. *Misclassified images are marked with red boundaries*. **(Bottom)** We show the first misclassified image (according to the decision value) into each class and its ground-truth class label.

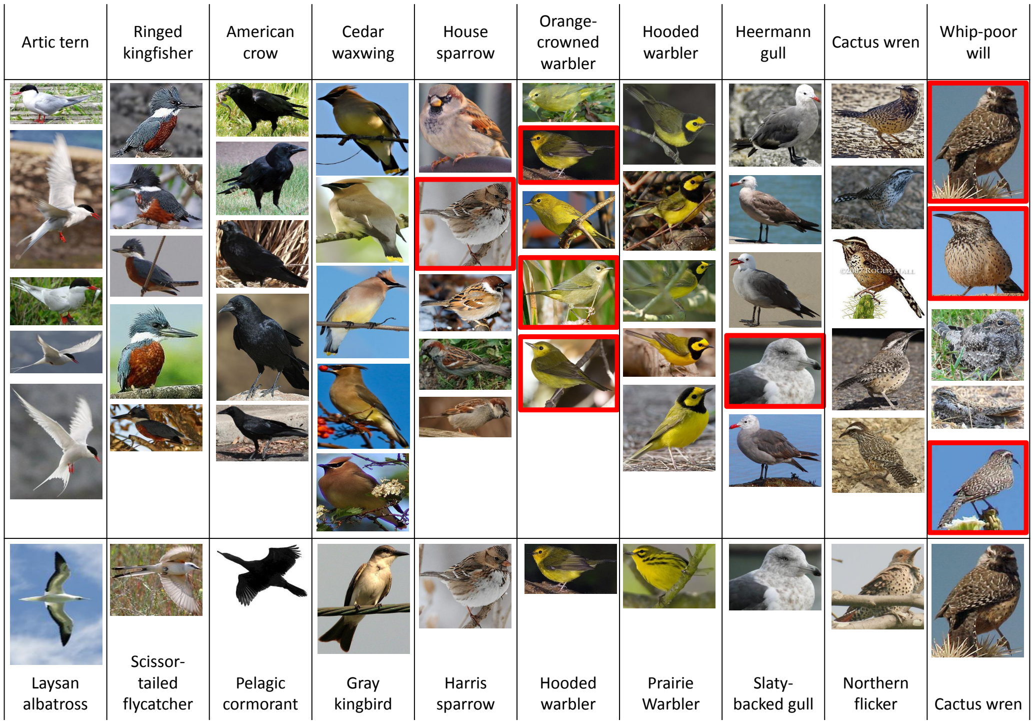

Figure 5: Qualitative results of our method (Ours$^{\text{struct}}$) on **CUB**. **(Top)** We list a subset of unseen class labels. **(Middle)** We show the top-5 images classified into each class, according to the decision values. *Misclassified images are marked with red boundaries*. **(Bottom)** We show the first misclassified image (according to the decision value) into each class and its ground-truth class label.

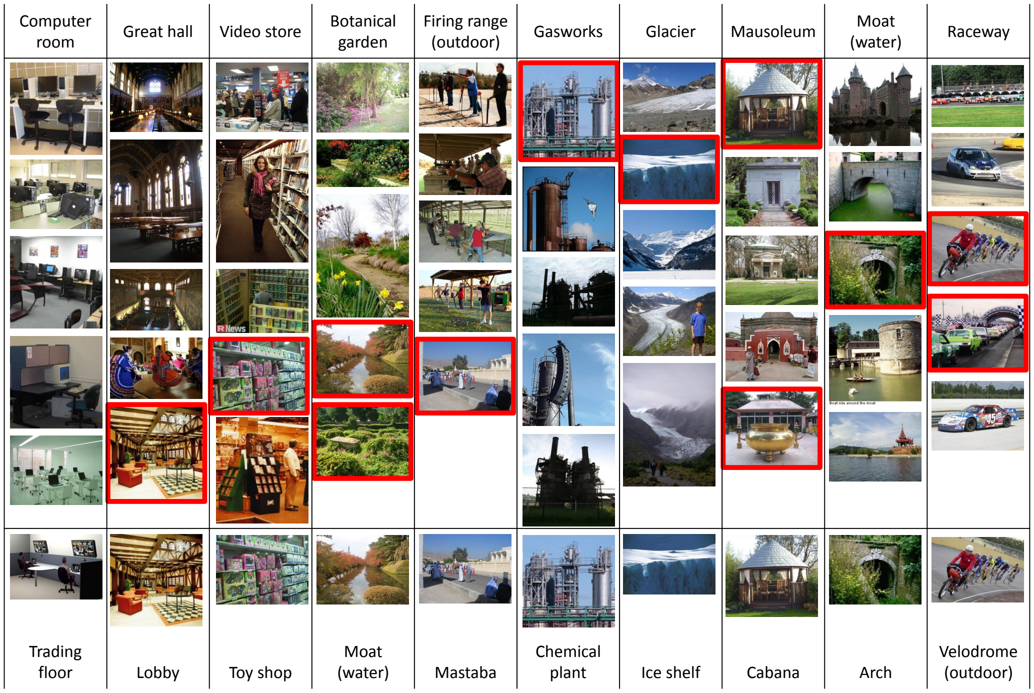

Figure 6: Qualitative results of our method (Ours$^{\text{o-vs-o}}$) on **SUN**. **(Top)** We list a subset of unseen class labels. **(Middle)** We show the top-5 images classified into each class, according to the decision values. *Misclassified images are marked with red boundaries*. **(Bottom)** We show the first misclassified image (according to the decision value) into each class and its ground-truth class label.

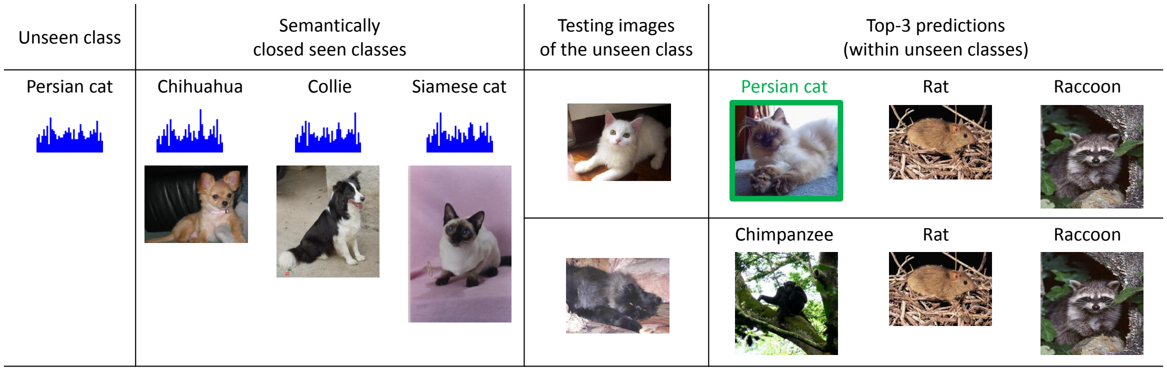
Unseen class
Semantically closed seen classes
Testing images of the unseen class
Top-3 predictions (within unseen classes)
Persian cat
Chihuahua
Collie
Siamese cat
Persian cat
Rat
Raccoon
Chimpanzee
Rat
Raccoon

Figure 7: Success/failure case analysis of our method (Ours$^{\text{struct}}$) on **AwA**: (Left) an unseen class label, (Middle-Left) the top-3 semantically similar seen classes to that unseen class, (Middle-Right) two test images of such unseen class, and (Right) the top-3 predicted unseen classes. The green text corresponds to the correct label.

C.-J. Lin. Liblinear: A library for large linear classification. *JMLR*, 9:1871–1874, 2008. 4

[9] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013. 2, 3

[10] Y. Fu, T. M. Hospedales, T. Xiang, Z. Fu, and S. Gong. Transductive multi-view embedding for zero-shot recognition and annotation. In *ECCV*, 2014. 4, 6

[11] Y. Fu, T. M. Hospedales, T. Xiang, and S. Gong. Transductive multi-view zero-shot learning. *TPAMI*, 37(11):2332–2345, 2015. 4, 6

[12] Z. Fu, T. Xiang, E. Kodirov, and S. Gong. Zero-shot object recognition by semantic manifold distance. In *CVPR*, 2015. 4, 6

[13] D. Jayaraman and K. Grauman. Zero-shot recognition with unreliable attributes. In *NIPS*, 2014. 2, 3, 5, 6

[14] D. Jayaraman, F. Sha, and K. Grauman. Decorrelating semantic visual attributes by resisting the urge to share. In *CVPR*, 2014. 3

[15] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long,

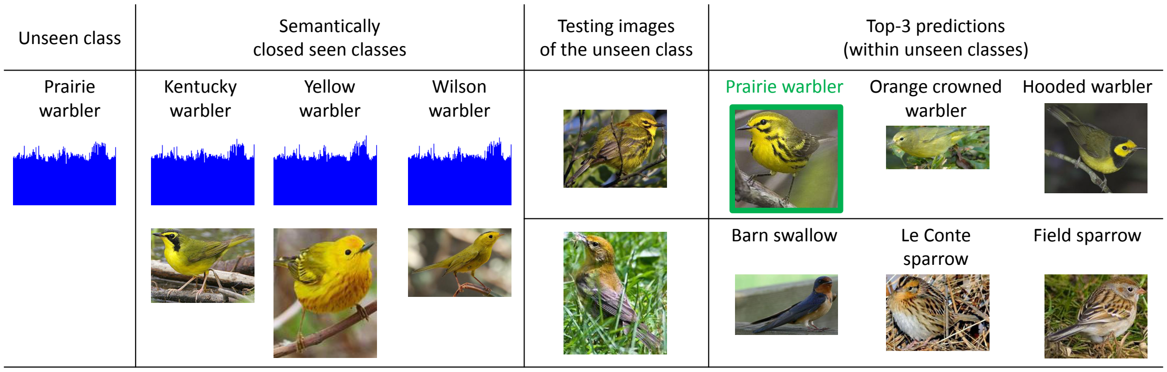

Figure 8: Success/failure case analysis of our method (Ours$^{\text{struct}}$) on **CUB**. (Left) an unseen class label, (Middle-Left) the top-3 semantically similar seen classes to that unseen class, (Middle-Right) two test images of such unseen classes, and (Right) the top-3 predicted unseen class. The green text corresponds to the correct label.

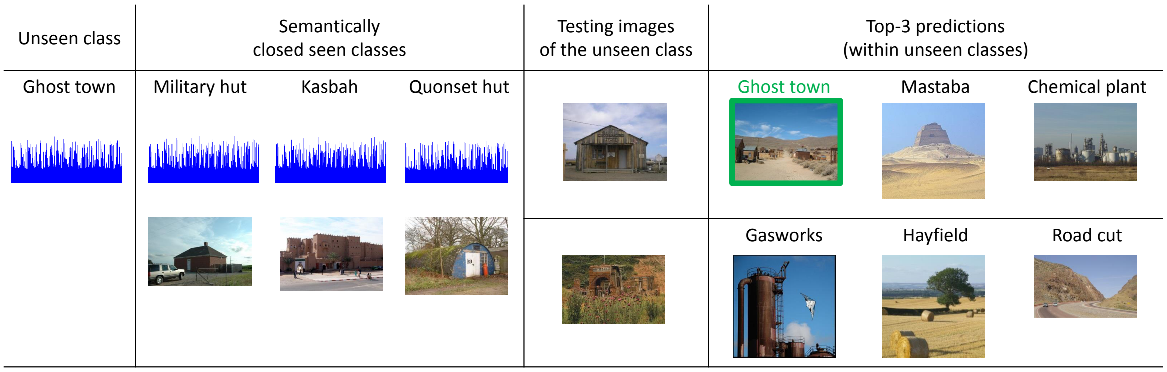
Unseen class
Semantically closed seen classes
Testing images of the unseen class
Top-3 predictions (within unseen classes)
Ghost town
Military hut
Kasbah
Quonset hut
Ghost town
Mastaba
Chemical plant
Gasworks
Hayfield
Road cut

Figure 9: Success/failure case analysis of our method (Ours$^{\text{o-vs-o}}$) on **SUN**. (Left) an unseen class label, (Middle-Left) the top-3 semantically similar seen classes to that unseen class, (Middle-Right) two test images of such unseen classes, and (Right) the top-3 predicted unseen class. The green text corresponds to the correct label.

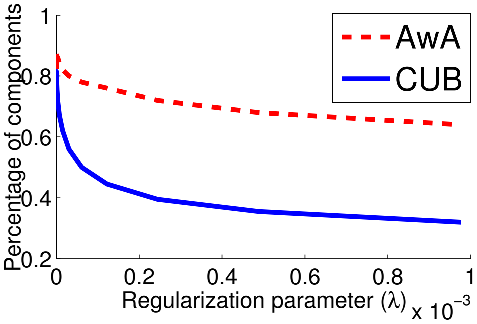

Figure 10: Percentages of basis components required to capture 95% of variance in classifier matrices for **AwA** and **CUB**.

R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM Multimedia*, 2014. 3

[16] E. Kodirov, T. Xiang, Z. Fu, and S. Gong. Unsupervised domain adaptation for zero-shot learning. In *ICCV*, 2015. 4, 6

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 3

[18] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *TPAMI*, 36(3):453–465, 2014. 2, 3

[19] X. Li, Y. Guo, and D. Schuurmans. Semi-supervised zero-shot classification with label representation learning. In *ICCV*, 2015. 4, 6

[20] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *ICLR Workshops*, 2013. 2

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013. 2

[22] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean. Zero-shot learning by convex combination of semantic embeddings. In *ICLR*, 2014. 2, 3, 4, 6, 8

[23] G. Patterson, C. Xu, H. Su, and J. Hays. The sun attribute database: Beyond categories for deeper scene understanding. *IJCV*, 108(1-2):59–81, 2014. 2, 3

[24] M. Rohrbach, M. Stark, G. Szarvas, I. Gurevych, and B. Schiele. What helps where–and why? semantic relatedness for knowledge transfer. In *CVPR*, 2010. 3

[25] B. Romera-Paredes and P. H. S. Torr. An embarrassingly simple approach to zero-shot learning. In *ICML*, 2015. 1, 2, 3, 5, 6

[26] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 2, 3

[27] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014. 4

[28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 5

[29] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 3

[30] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 2

[31] X. Wang and Q. Ji. A unified probabilistic approach modeling relationships between attributes and objects. In *ICCV*, 2013. 3

[32] F. X. Yu, L. Cao, R. S. Feris, J. R. Smith, and S.-F. Chang. Designing category-level attributes for discriminative visual recognition. In *CVPR*, 2013. 5, 6

[33] Z. Zhang and V. Saligrama. Classifying unseen instances by learning class-independent similarity functions. *arXiv preprint arXiv:1511.04512*, 2015. 2, 5, 6

[34] Z. Zhang and V. Saligrama. Zero-shot learning via semantic similarity embedding. In *ICCV*, 2015. 1, 2, 5, 6

[35] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014. 3