

Adaptive Confidence Smoothing for Generalized Zero-Shot Learning

Yuval Atzmon

Bar-Ilan University, NVIDIA Research

yuval.atzmon@biu.ac.il

Gal Chechik

Bar-Ilan University, NVIDIA Research

gal.chechik@biu.ac.il

Abstract

Generalized zero-shot learning (GZSL) is the problem of learning a classifier where some classes have samples and others are learned from side information, like semantic attributes or text description, in a zero-shot learning fashion (ZSL). Training a single model that operates in these two regimes simultaneously is challenging. Here we describe a probabilistic approach that breaks the model into three modular components, and then combines them in a consistent way. Specifically, our model consists of three classifiers: A “gating” model that makes soft decisions if a sample is from a “seen” class, and two experts: a ZSL expert, and an expert model for seen classes. We address two main difficulties in this approach: How to provide an accurate estimate of the gating probability without any training samples for unseen classes; and how to use expert predictions when it observes samples outside of its domain.

The key insight to our approach is to pass information between the three models to improve each one’s accuracy, while maintaining the modular structure. We test our approach, adaptive confidence smoothing (COSMO), on four standard GZSL benchmark datasets and find that it largely outperforms state-of-the-art GZSL models. COSMO is also the first model that closes the gap and surpasses the performance of generative models for GZSL, even-though it is a light-weight model that is much easier to train and tune.

1. Introduction

Generalized zero-shot learning (GZSL) [9] is the problem of learning to classify samples from two different domains of classes: *seen classes*, trained in a standard supervised way from labeled samples, and *unseen classes*, learned from external knowledge, such as attributes or natural language, in a zero-shot-learning fashion. GZSL poses a unique combination of hard challenges: First, the model has to learn effectively for classes without samples (zero-shot). It also needs to learn well for classes with many samples. Finally, the two very different regimes should be combined in a consistent way in a single model. GZSL can be

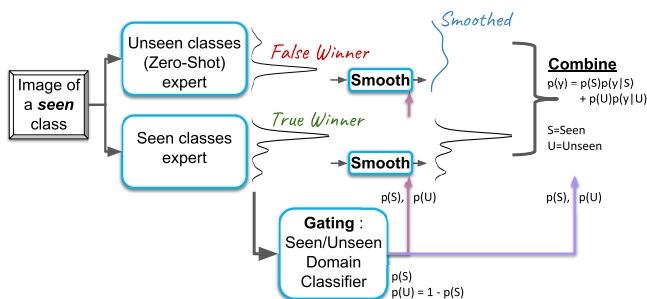


Figure 1. A qualitative illustration of COSMO : An input image is processed by two experts: A *seen*-classes expert, and an *unseen*-classes expert, which is a zero-shot model. (1) When an image is from a *seen* class, the zero-shot expert may still produce an overly-confident false-positive prediction. COSMO smooths the predictions of the unseen expert if it believes that the image is from a seen class. The amount of smoothing is determined by a novel gating classifier. (2) Final GZSL predictions are based on a soft combination of the predictions of two experts, with weights provided by the gating module.

viewed as an extreme case of classification with unbalanced classes, hence solving the last challenge can lead to better ways of addressing class imbalance, which is a key problem in learning with real-world data.

The three learning problems described above operate in different learning setups, hence combining them into a single model is challenging. Here, we propose an architecture with three modules, each focusing on one problem. At inference time, these modules share their prediction confidence in a principled probabilistic way in order to reach an accurate joint decision.

One natural instance of this modular architecture is *hard gating*: Given a test sample, the gate assigns it either to a *seen expert* - trained as a standard supervised classifier - or to an *unseen expert* - trained in a zero-shot-learning fashion [39]. Only the selected expert is used for prediction, ignoring the other expert. Here we study a more general case, where both the seen expert and the unseen expert process each test sample, and their predictions are combined in a *soft* way. Specifically, the predictions are combined by

the soft gater using the law of total probability: $p(class) = p(class|seen)p(seen) + p(class|unseen)p(unseen)$.

Unfortunately, softly combining expert decisions raises several difficulties. First, when training a gating module it is hard to provide an accurate estimate of the probability that a sample is from the “unseen” classes, because by definition no samples have been observed from those classes. Second, experts tend to behave in uncontrolled ways when presented with out-of-distribution samples, often producing confident-but-wrong predictions. As a result, when using a soft combination of the two expert models, the “irrelevant” expert may overwhelm the decision of the correct expert.

We address these issues in two ways. First, we show how to train a binary gating mechanism to classify the Seen/Unseen domain based on the distribution of softmax class predictions. The idea is to simulate the softmax response to samples of unseen classes using a held-out subset of training classes, and represent expert predictions in a class-independent way. Second, we introduce a Laplace-like prior [28] over softmax outputs in a way that uses information from the gating classifier. This additional information allows the experts to estimate class confidence more accurately.

This combined approach, named *adaptive Confidence SMOothing* (COSMO), has significant advantages. It can incorporate any state-of-the-art zero-shot learner as a module, as long as it outputs class probabilities; It is very easy to implement and apply (code provided) since it has very few hyper-parameters to tune; Finally, it outperforms competing approaches on all four GZSL benchmarks (AWA, SUN, CUB, FLOWER). Our main novel contributions are:

- A new soft approach to combine decisions of seen and unseen classes for GZSL.
- A new “out-of-distribution” (OOD) classifier to separate seen from unseen classes, and a negative result, showing that modern OOD classifiers have limited effectiveness on ZSL benchmarks.
- New state-of-the-art results for GZSL for all four main benchmarks, AWA, SUN, CUB and FLOWER. COSMO is the first model that is comparable to or better than generative models of GZSL, while being easy to train.
- A characterization of GZSL approaches on the seen-unseen accuracy plane.

2. Related work

In a broad perspective, **zero-shot learning** is a task of *compositional reasoning* [21, 22, 6, 4], where new concepts are constructed by recombining primitive elements [22]. This ability resembles human learning, where humans can easily recombine simple skills to solve new tasks [21]. ZSL has attracted significant interest in recent years [47, 13, 23, 35, 18, 3, 52, 48, 25, 40]. As our main ZSL module, we

use *LAGO* [7], a state-of-the-art approach which learns to combine attribute that describes classes using an AND-OR group structure to estimate $p(class|image)$.

Generalized ZSL extends ZSL to the more realistic scenario where the test data contains both seen and unseen classes. There are two kinds GZSL methods. First, some approaches synthesize feature vectors of unseen classes using generative models like VAE or GAN, and then use them in training [46, 10, 5, 29, 54]. Second, approaches that use the semantic class descriptions directly during training and inference [41, 52, 14, 51, 27, 39, 9]. To date, the first kind, methods that augment data, perform better.

Among previous GZSL approaches, several are closely related to COSMO. [39] uses a hard gating mechanism to assign a sample to one of two domain experts. [9] calibrates between seen and unseen class scores by subtracting a constant value from seen class scores. [27] uses temperature scaling [16] and an entropy regularizer to make seen class scores less confident and unseen scores more confident.

Detecting out-of-distribution samples: Our approach to soft gating builds on developing an out-of-distribution detector, where *unseen* images are treated as “out-of-distribution” samples. There is a large body of work on 1-class and anomaly detection which we do not survey here. In this context, the most relevant recent work includes [15, 26, 42, 37]. [15] detects an OOD sample if the largest softmax score is below a threshold [26] scales the softmax “temperature” [16] and perturbs the input with a small gradient step. [37] represents each class by multiple word-embeddings and compares output norms to a threshold. [42] trains an ensemble of models on a set of “leave-out” classes, with a margin loss that encourages high-entropy scores for left-out samples.

When testing [26, 42] on the ZSL benchmarks studied in this paper (CUB, SUN, AWA), we found the perturbation approach of [26] hurts OOD detection, and that the loss of [42] overfits on the leave-out classes. We discuss possible explanation of these effects in Supplementary (B).

Mixture of experts (MoE): In MoE [17, 49, 38], given a sample, a gating network first assigns weights to multiple experts. The sample is then classified by those experts, and their predictions are combined by the gating weights. All parts of the model are usually trained jointly, often using an EM approach. Our approach fundamentally differs from MoE in that, at training time, it is known for every sample whether or not it was already seen. As a result, experts can be trained separately without any need to infer latent variables, ensuring that each module is an expert of its own domain (seen or unseen).

3. Generalized zero-Shot learning

We start with a formal definition of zero-shot learning (ZSL) and then extend it to generalized ZSL.

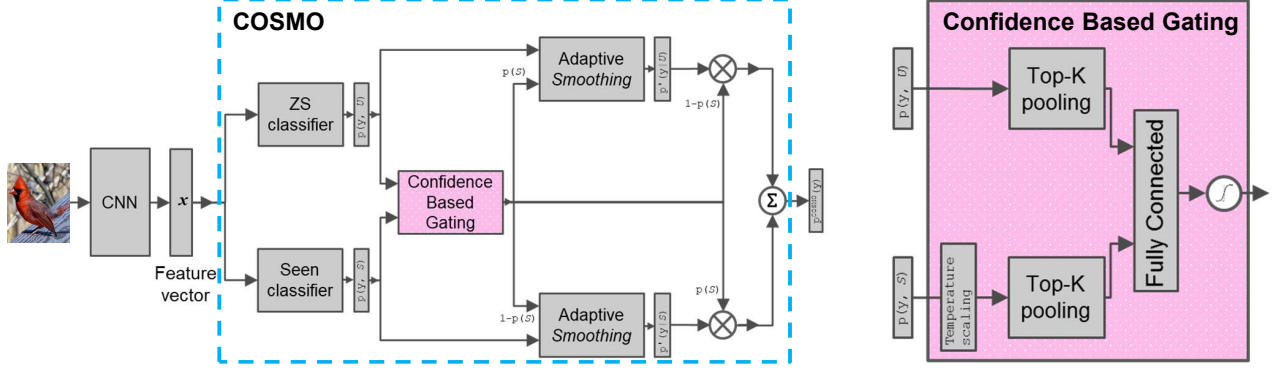


Figure 2. **Left**, COSMO Architecture: We decompose the GZSL task into three sub-tasks that can be addressed separately. (1) A model trained to classify seen \mathcal{S} classes. (2) A model classifying *unseen* \mathcal{U} classes, namely a ZSL model, conditioned on \mathcal{U} . (3) A *gating* binary classifier trained to discriminate between seen and unseen classes and to weigh the two models in a soft way; Before weighing (1) & (2) softmax distributions, we add a prior for each if the gating network provides low confidence (Figure 1 and Sec 4.2). **Right**, The gating network (Zoom-in): It takes softmax scores as inputs. We train it to be aware of the response of softmax scores to *unseen* images, with samples from held-out classes. Because test classes are different from train classes, we pool the top- K scores, achieving invariance to class identity (Section 4.1). The fully-connected layer only learns 10-50 weights (K is small) since this is a binary classifier.

In zero-shot learning, a training set \mathcal{D} has N labeled samples: $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1 \dots N\}$, where each \mathbf{x}_i is a feature vector and $y_i \in \mathcal{S}$ is a label from a *seen* class $\mathcal{S} = \{1, 2, \dots |\mathcal{S}|\}$.

At test time, a new set of samples $\mathcal{D}' = \{\mathbf{x}_i, i = N + 1 \dots N + M\}$ is given from a set of *unseen* classes $\mathcal{U} = \{|\mathcal{S}| + 1, \dots |\mathcal{S}| + |\mathcal{U}|\}$. Our goal is to predict the correct class of each sample. As a supervision signal, each class $y \in \mathcal{S} \cup \mathcal{U}$ is accompanied with a *class-description* vector \mathbf{a}_y in the form of semantic attributes [23] or natural language embedding [34, 54, 39]. The crux of ZSL is to learn a compatibility score for samples and class-descriptions $F(\mathbf{a}_y, \mathbf{x})$, and predict the class y that maximizes that score. In probabilistic approaches to ZSL [23, 24, 44, 39, 7, 27] the compatibility function assigns a probability for each class $p(Y = y|\mathbf{x}) = F(\mathbf{a}_y, \mathbf{x})$, with Y viewed as a random variable for the label y of a sample \mathbf{x} .

Generalized ZSL: While in ZSL test samples are drawn from the unseen classes $Y \in \mathcal{U}$, in GZSL samples are drawn from either the *seen or unseen* domains: $Y \in \mathcal{S} \cup \mathcal{U}$.

Notation: Below, we denote an unseen class by $Y \in \mathcal{U}$ and a seen one by $Y \in \mathcal{S}$. Given a sample \mathbf{x} and a label y we denote the conditional distribution that a class is seen by $p(\mathcal{S}) = p(Y \in \mathcal{S}|\mathbf{x})$, or unseen $p(\mathcal{U}) = p(Y \in \mathcal{U}|\mathbf{x}) = 1 - p(Y \in \mathcal{S}|\mathbf{x})$, and the conditional probability of a label by $p(y) = p(Y = y|\mathbf{x})$, $p(y|\mathcal{S}) = p(Y = y|Y \in \mathcal{S}, \mathbf{x})$ and $p(y|\mathcal{U}) = p(Y = y|Y \in \mathcal{U}, \mathbf{x})$. For ease of reading, our notation does not explicitly state the conditioning on \mathbf{x} .

4. Our approach

We now describe COSMO, a probabilistic approach that breaks the model into three modules. The key idea is

that these modules exchange information to improve each other’s accuracy. Formally, by the law of total probability

$$p(y) = p(y|\mathcal{S})p(\mathcal{S}) + p(y|\mathcal{U})p(\mathcal{U}). \quad (1)$$

This formulation decomposes GZSL into three sub-tasks that can be addressed separately. (1) $p(y|\mathcal{S})$ can be estimated by any model trained to classify seen \mathcal{S} classes, whose prediction we denote by $p^S(y|\mathcal{S})$. (2) Similarly, $p(y|\mathcal{U})$ can be computed by a model classifying *unseen* \mathcal{U} classes, namely a ZSL model, whose prediction we denote by $p^{ZS}(y|\mathcal{U})$. (3) Finally, the two terms are weighted by $p(\mathcal{S})$ and $p(\mathcal{U}) = 1 - p(\mathcal{S})$, which can be computed by a *gating* classifier, whose prediction we denote by p^{Gate} , that is trained to distinguish seen from unseen classes. Together, we obtain a GZSL mixture model:

$$p(y) = p^S(y|\mathcal{S})p^{Gate}(\mathcal{S}) + p^{ZS}(y|\mathcal{U})p^{Gate}(\mathcal{U}) \quad (2)$$

A hard variant of Eq. (2) was introduced in [39], where the gating mechanism makes a hard decision to assign a test sample to one of two expert classifiers, p^{ZS} or p^S . Unfortunately, although conceptually simple, using a **soft** mixture model raises several problems.

First, combining models in a soft way means that each model contributes its beliefs, even for samples from the other “domain”. This tends to damage the accuracy because multiclass models tend to assign most of the softmax distribution mass to very few classes, even when their input is random noise [15]. For instance, when the unseen classifier is given an input image from a seen class, its output distribution tends to concentrate on a few spurious classes. This peaked distribution “confuses” the combined GZSL mixture model, leading to a false-positive prediction of the spurious classes. A second challenge for creating a soft gating

model is to assign accurate weights to the two experts. This is particularly complex when discriminating seen from unseen classes, because it requires access to training samples of the unseen domain.

COSMO addresses these two problems using a novel confidence-based gating network and by applying a novel prior during inference. Its inference process is summarized in Algorithm 1, and a walk-through example is provided in Supplementary A. Next, we describe COSMO in detail.

4.1. Confidence-based gating model

The gating module aims to decide if an input image comes from a seen class or an unseen class. Since no training samples are available for unseen classes, we can view this problem in the context of *Out-Of-Distribution* detection by treating seen-class (\mathcal{S}) images as “in-distribution” and unseen-class (\mathcal{U}) images as “out-of-distribution”.

Several authors proposed training an OOD detector on in-distribution data, and detect an image as out-of-distribution if the largest softmax score is below a threshold [15, 26, 42]. Here we improve this approach by training a network on top of the softmax output of the two experts, with the goal of discriminating \mathcal{U} images from \mathcal{S} images. Intuitively, this can improve the accuracy of the gating module because the output response of the two experts differs for \mathcal{S} images and \mathcal{U} images. We name this network as *confidence-based gate* (CBG). It is illustrated in Figure 2.

One important technical complication is that training the CBG cannot observe any \mathcal{U} images, because they must be used as unseen. We therefore create a hold-out set from \mathcal{S} classes that are not used for training and use them to estimate the output response of the experts over \mathcal{U} images. Below, we refer to this set of classes as held-out \mathcal{H} classes, and their images as \mathcal{H} images. Note that due to similar reasons we cannot train the gater jointly with the \mathcal{S} and \mathcal{U} experts. See Supplementary D for details.

This raises a further complexity: Training the unseen expert on \mathcal{H} classes means that at test time, when presented with *test* classes, The unseen expert should have an output layer that is different from its output layer during training. Specifically, it corresponds to new (test) classes, possibly with a different dimension. To become invariant to identity and the order of \mathcal{H} classes in the output of the expert, the CBG takes the top- K scores of the softmax and sorts them. This process, which we call *top-k pooling*, guarantees that the CBG is invariant to the specific classes presented. Top- K pooling generalizes max-pooling, and becomes equivalent to max pooling for $K=1$.

4.2. Adaptive confidence smoothing

As we described above, probabilistic classifiers tend to assign most of the softmax mass to very few classes, even when a sample does not belong to any of the classes in

the vocabulary. Intuitively, when given an image of out-of-vocabulary class as input, we would expect all classes to obtain a uniformly low probability, since they are all “equally wrong”. To include this prior belief in our model we borrow ideas from Bayesian parameter estimation. Consider the set of class-confidence values as the quantity that we wish to estimate, based on the confidence provided by the model (softmax output scores). In Bayesian estimation, one combines the data (here, the predicted confidence) with a prior distribution (here, our prior belief).

Specifically, for empirical categorical (multinomial) data, *Laplace smoothing* [28] is a common technique to achieve a robust estimate with limited samples. It amounts to adding “pseudo counts” uniformly across all classes, and functions as a prior distribution over classes. We can apply a similar technique here, and combine the predictions with an additive prior distribution $\pi^{\mathcal{U}} = p_0(y|\mathcal{U})$. This yields

$$p^\lambda(y|\mathcal{U}) = (1-\lambda)p(y|\mathcal{U}) + \lambda\pi^{\mathcal{U}}, \quad (3)$$

where λ weighs the prior, and $\pi^{\mathcal{U}}$ is not conditioned on \mathbf{x} . Similarly, for the seen distribution, we set $p^\lambda(y|\mathcal{S}) = (1-\lambda)p(y|\mathcal{S}) + \lambda\pi^{\mathcal{S}}$. When no other information is available we set the prior to the maximum entropy distribution, which is the uniform distribution $\pi^{\mathcal{U}} = 1/(\#\text{unseen classes})$ and $\pi^{\mathcal{S}} = 1/(\#\text{seen classes})$.

An Adaptive Prior: How should the prior weight λ be set? In Laplace smoothing, adding a constant pseudocount has the property that its relative weight decreases as more samples are available. Intuitively, this means that **when the data provides strong evidence, the prior is weighted more weakly**. We adopt this intuition for making the trade-off parameter λ adaptive. Intuitively, if we believe that a sample *does not* belongs to a seen class, we smooth the seen classifier outputs (Figure 1). More specifically, we apply an adaptive prior by replacing the constant λ with our belief about each domain (for $p'(y|\mathcal{U})$ set $\lambda = p(\mathcal{U})$):

$$\begin{aligned} p'(y|\mathcal{U}) &= p(\mathcal{U})p(y|\mathcal{U}) + (1-p(\mathcal{U}))\pi^{\mathcal{U}} \\ &= p(y, \mathcal{U}) + (1-p(\mathcal{U}))\pi^{\mathcal{U}}. \end{aligned} \quad (4)$$

Similarly $p'(y|\mathcal{S}) = p(\mathcal{S})p(y|\mathcal{S}) + (1-p(\mathcal{S}))\pi^{\mathcal{S}}$. In practice, we use the ZS model estimation for $p(y, \mathcal{U})$, and the gating model estimation for $p(\mathcal{U})$, yielding $p'(y|\mathcal{U}) = p^{\text{ZS}}(y, \mathcal{U}) + (1-p^{\text{Gate}}(\mathcal{U}))\pi^{\mathcal{U}}$ and similarly for $p'(y|\mathcal{S})$.

The resulting model has two interesting properties. First, it reduces hyper-parameter tuning, because prior weights are determined automatically. Second, smoothing adds a constant value to each score, hence it maintains the class that achieves the maximum of each individual expert, but at the same time affects their combined prediction in Eq. (2).

Algorithm 1. COSMO Inference

- 1: **Input:** Image
 - 2: Estimate $p^S(y, \mathcal{S})$ and $p^{ZS}(y, \mathcal{U})$ of two experts
 - 3: Estimate $p^{Gate}(\mathcal{S}) = f(p^S(y, \mathcal{S}), p^{ZS}(y, \mathcal{U}))$; Fig. 2
 - 4: Estimate $p'(y|\mathcal{S})$ and $p'(y|\mathcal{U})$ by smoothing; Eq. (4)
 - 5: Estimate $p(y)$ by soft-combining; Eq. (2)
-

5. Details of our approach

Our approach has three learning modules: A model for seen classes, for unseen classes, and for telling them apart. The three components are trained separately. Supp. section D explains why they cannot be trained jointly in this setup.

A model for unseen classes. For unseen classes, we use either LAGO [7] or fCLSWGAN [46] with the code provided by the authors. Each of these models achieves state-of-the-art results for part of ZSL benchmarks. LAGO predicts $p^{ZS}(y|\mathbf{x})$ by learning an AND-OR group structure over attributes. fCLSWGAN [46] uses GAN to augment the training data with synthetic feature vectors of unseen classes, and then trains a classifier to recognize the classes. We retrained the models on the GZSL split (Figure 4).

A model for seen classes. For seen classes, we trained a logistic regression classifier to predict $p^S(y|\mathbf{x})$. We used a LBFGS solver [11] with default aggressiveness hyperparameter ($C=1$) of sci-kit learn [33], as it exhibits good performance over the Seen-Val set (Figure 4).

A confidence-based gating model. To discriminate between Seen and Unseen classes, we use a logistic regression classifier to predict $p(\mathcal{S}|\mathbf{x})$, trained on the *Gating-Train set* (Figure 4). For input features, we use softmax scores of both the *unseen* expert (p^{ZS}) and seen expert (p^S). We also apply temperature scaling [26] to inputs from p^S , Figure 2.

We used the sci-kit learn LBFGS solver with default aggressiveness hyper parameter ($C=1$) because the number of weights ($\sim 10-50$) is much smaller than the number of training samples (\sim thousands). We tune the decision threshold and softness of the gating model by adding constant bias β and applying a sigmoid with γ gain on top of its scores:

$$p(\mathcal{S}|\mathbf{x}) = \sigma\left\{\gamma[\text{score} - \beta]\right\} . \quad (5)$$

γ and β were tuned using cross validation.

6. Experiments

We tested COSMO on four GZSL benchmarks and compared to 17 state-of-the art approaches.

The source code to reproduce our experiments is under <http://chechiklab.biu.ac.il/~yuvval/COSMO/>.

6.1. Evaluation protocol

To evaluate COSMO we follow the protocol of Xian [47, 45], which became the common experimental framework for comparing GZSL methods. Our evaluation uses its features (ResNet [20]), cross-validation splits, and evaluation metrics for comparing to state-of-the-art baselines.

Evaluation Metrics: By definition, GZSL aims at two different sub-tasks: classify seen classes and classify unseen classes. The standard GZSL evaluation metrics therefore combine accuracy from these two sub-tasks. Following [45], we report the harmonic mean of Acc_{tr} - the accuracy over seen classes, and Acc_{ts} - the accuracy over unseen classes, see Eq. 21 in [45], $Acc_H = 2(Acc_{ts}Acc_{tr})/(Acc_{ts} + Acc_{tr})$.

As a second metric, we compute the full seen-unseen accuracy curve using a parameter to sweep over the decision threshold. Like the precision-recall curve or ROC curve, the seen-unseen curve provides a tunable trade-off between the performance over the seen and unseen domains.

Finally, we report the *Area Under Seen-Unseen Curve* (AUSUC) [9].

6.2. Datasets

We tested COSMO on four generalized zero-shot learning benchmark datasets: CUB, AWA, SUN and FLOWER. **CUB** [43]: is a task of fine-grained classification of bird species. CUB has 11,788 images of 200 bird species. Each species described by 312 attributes (like *wing-color-olive*, *beak-shape-curved*). It has 100 seen training classes, 50 unseen validation and 50 unseen test classes.

AWA: Animals with Attributes (AWA) [23] consists of 30,475 images of 50 animal classes. Classes and attributes are aligned with the class-attribute matrix of [31, 19], using a vocabulary of 85 attributes (like *white*, *brown*, *stripes*, *eat-fish*). It has 27 seen training classes, 13 unseen validation and 10 unseen test classes.

SUN [32]: is a dataset of complex visual scenes, having 14,340 images from 717 scene types and 102 semantic attributes. It has 580 seen training classes, 65 unseen validation and 72 unseen test classes.

FLOWER [30]: is a dataset of fine-grained classification of flowers, with 8189 images of 102 classes. Class descriptions are based on sentence embedding from [34]. We did not test COSMO+LAGO with this dataset because LAGO cannot use sentence embedding.

6.3. Cross-validation

For selecting hyper-parameters for COSMO, we make two additional splits: *GZSL-val* and *Gating Train / Val*. See Figure 4 for details.

We used cross-validation to optimize the Acc_H metric over β and γ of Eq. (5) on the *GZSL-Val* set. We tuned these

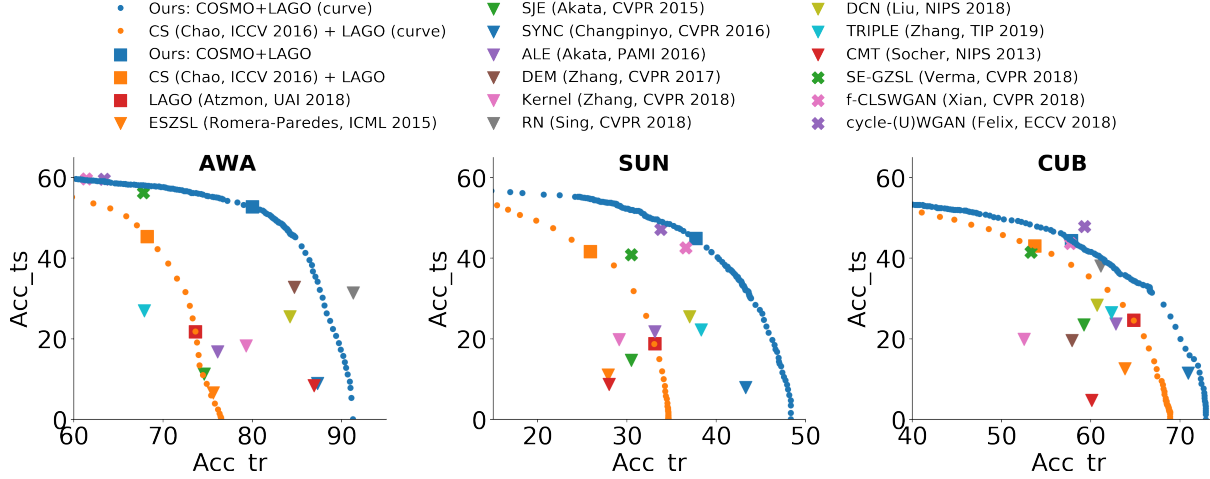


Figure 3. The Seen-Unseen curve for COSMO+LAGO, compared with: (1) The curve of CS+LAGO [9] baseline, (2) 15 baseline GZSL models. Dot markers denote samples of each curve. **Squares**: COSMO cross-validated model and its LAGO-based baselines. **Triangles**: non-generative approaches, **'X'**: approaches based on generative-models. Generative models tend to be biased toward the Unseen classes, while non-generative models tend to be biased toward the Seen classes. Importantly, the COSMO curve achieves a better or equivalent performance compared to all methods, and allows to easily choose any operation point along the curve.

DATASET	AWA			SUN			CUB			FLOWER		
	Acc_{ts}	Acc_{tr}	Acc_H	Acc_{ts}	Acc_{tr}	Acc_H	Acc_{ts}	Acc_{tr}	Acc_H	Acc_{ts}	Acc_{tr}	Acc_H
NON-GENERATIVE MODELS												
ESZSL [36]	6.6	75.6	12.1	11	27.9	15.8	12.6	63.8	21	11.4	56.8	19
SJE [2]	11.3	74.6	19.6	14.7	30.5	19.8	23.5	59.2	33.6	13.9	47.6	21.5
DEVISE [12]	13.4	68.7	22.4	16.9	27.4	20.9	23.8	53	32.8	9.9	44.2	16.2
SYNC [8]	8.9	87.3	16.2	7.9	43.3	13.4	11.5	70.9	19.8	-	-	-
ALE [1]	16.8	76.1	27.5	21.8	33.1	26.3	23.7	62.8	34.4	34.4	13.3	21.9
DEM [52]	32.8	84.7	47.3	-	-	-	19.6	57.9	29.2	-	-	-
KERNEL [50]	18.3	79.3	29.8	19.8	29.1	23.6	19.9	52.5	28.9	-	-	-
ICINESS [14]	-	-	-	-	-	30.3	-	-	41.8	-	-	-
TRIPLE [51]	27	67.9	38.6	22.2	38.3	28.1	26.5	62.3	37.2	-	-	-
RN [41]	31.4	91.3	46.7	-	-	-	38.1	61.1	47	-	-	-
GENERATIVE MODELS												
SE-GZSL [5]	56.3	67.8	61.5	40.9	30.5	34.9	41.5	53.3	46.7	-	-	-
fCLSWGAN [46]	59.7	61.4	59.6	42.6	36.6	39.4	43.7	57.7	49.7	59	73.8	65.6
fCLSWGAN* (BY PROVIDED CODE)	53.6	67	59.6	40.1	36	37.9	45.1	55.5	49.8	58.1	73.2	64.8
cycle-(U)WGAN [10]	59.6	63.4	59.8	47.2	33.8	39.4	47.9	59.3	53.0	61.6	69.2	65.2
COSMO AND BASELINES												
CMT [39]	8.4	86.9	15.3	8.7	28	13.3	4.7	60.1	8.7	-	-	-
DCN [27]	25.5	84.2	39.1	25.5	37	30.2	28.4	60.7	38.7	-	-	-
LAGO [7]	21.8	73.6	33.7	18.8	33.1	23.9	24.6	64.8	35.6	-	-	-
CS [9] + LAGO	45.4	68.2	54.5	41.7	25.9	31.9	43.1	53.7	47.9	-	-	-
OURS: COSMO+fCLSWGAN*	64.8	51.7	57.5	35.3	40.2	37.6	41.0	60.5	48.9	59.6	81.4	68.8
OURS: COSMO+LAGO	52.8	80	63.6	44.9	37.7	41.0	44.4	57.8	50.2	-	-	-

Table 1. Comparing COSMO with state-of-the-art GZSL non-generative models and with generative models that synthesize feature vectors. Acc_{tr} is the accuracy of seen classes, Acc_{ts} is the accuracy of unseen classes and Acc_H is their harmonic mean. COSMO+LAGO uses LAGO [7] as a baseline GZSL model, and respectively COSMO+fCLSWGAN uses fCLSWGAN [46]. COSMO+LAGO improves Acc_H over state-of-the-art models by 34%, 35%, 7% respectively for AWA, SUN and CUB. Comparing with generative models, COSMO+LAGO closes the non-generative:generative performance gap, and is comparable to or better than these models, while is very easy to train.

hyper params by first taking a coarse grid search, and then making a finer search around the best performing values for the threshold. Independently, we used cross-validation on *Gating-Train/Val* to optimize the out-of-distribution AUC over T (Temperature) and K (for top-K pooling).

We stress that training the gating network using *Gating-Train/Val* is not considered as training with external data, because in accordance with [45], once hyper parameters are selected, models are retrained on the union of the training and the validation sets (excluding the gating model).

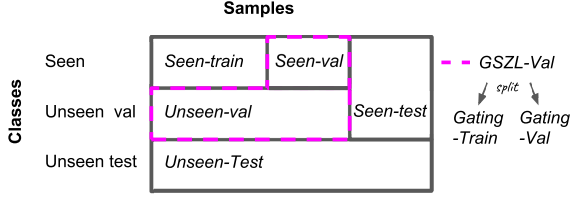


Figure 4. GZSL cross-validation splits. The data is organized across classes and samples. We define **Seen-Val** as a subset of the seen-training samples provided by [47, 45]. We define **GZSL-Val** = Seen-Val \cup Unseen-Val (in pink). We use GZSL-Val to select the model’s hyper-parameters and learn (~ 10 -50) weights of the gating network. We split GZSL-Val to **Gating-Train** and **Gating-Val** subsets, and use Gating-Train as the held-out subset to train the gating model and Gating-Val to evaluate its metrics.

6.4. Compared methods

We compare COSMO with 17 leading GZSL methods. These include widely-used baselines like **ESZSL** [36], **ALE** [1], **SYNC** [8], **SJE** [2], **DEVISE** [12], recently published approaches **RN** [41], **DEM** [52], **ICINESS** [14], **TRIPLE** [51], **Kernel** [50] and methods that provide interesting insight into the method, including **CMT** [39], **DCN** [27], **LAGO** [7] and **CS** [9], which we reproduced using LAGO as a ZSL module.

Recent work showed that generating synthetic samples of unseen classes using GANs or VAEs [46, 10, 5, 54] can substantially improve generalized zero-shot learning. The recent literature considers this generative effort to be orthogonal to modelling, since the two efforts can be combined [27, 7, 53, 14, 51]. Here we compare COSMO directly both with the approaches listed above, and with generative approaches **fCLSWGAN** [46], **cycle-(U)WGAN** [10], **SE-GZSL** [5].

	AWA	SUN	CUB	FLOWER
ESZSL	39.8	12.8	30.2	25.7
LAGO [7]	43.4	16.3	34.3	-
fCLSWGAN	46.1	22	34.5	53.1
cycle-(U)WGAN	45	22.5	40.4	56.9
COSMO & fCLSWGAN	55.9	21	35.6	58.1
COSMO & LAGO	53.2	23.9	35.7	-

Table 2. Area Under Seen-Unseen Curve (AUSUC) on the test set: On all datasets, COSMO improves AUSUC for LAGO and fCLSWGAN. COSMO introduces new state-of-the-art results on 3 of 4 datasets.

7. Results

We first describe the performance of COSMO on the test set of the four benchmarks and compare them with baseline methods. We then study in greater depth the properties of

COSMO, through a series of ablation experiments.

Table 1 describes the test accuracy of COSMO+LAGO, COSMO+fCLSWGAN and compared methods over the four benchmark datasets. Compared with non-generative models, COSMO+LAGO improves the harmonic accuracy Acc_H by a large margin for all four datasets: 63.6% vs 47.3% in AWA, 41% vs 30.3% in SUN and 50.2% vs 47% in CUB.

In addition, COSMO+LAGO closes the performance gap with generative approaches. It wins in AWA (63.6% versus 61.5%) and SUN (41% versus 39.4%), and loses in CUB (50.2% versus 53%). Interestingly, COSMO+LAGO reaches state-of-the-art performance although LAGO alone performed poorly on the *generalized* ZSL task.

COSMO+fCLSWGAN provides a new state-of-the-art result on FLOWER (68.8% vs 65.6%). On AWA, SUN and CUB it achieves lower performance than fCLSWGAN and COSMO+LAGO. This happens because the chosen operating point for (Acc_{tr} , Acc_{ts}), selected by cross validation, was not optimal for the harmonic accuracy Acc_H . More details are provided in Supplementary C.

7.1. The seen-unseen plane

By definition, the GZSL task aims to perform well in two different metrics: accuracy on seen classes and on unseen classes. It is therefore natural to compare approaches by their performance on the seen-unseen plane. This is important, because different approaches may select different operating-points to trade seen and unseen accuracy.

In Figure 3 we provide a full Seen-Unseen curve (blue dots) that shows how COSMO+LAGO trade-off the metrics. We compare it with a curve that we computed for the CS+LAGO baseline (orange dots) and also show the results (operation-points) reported for the compared methods. For plotting the curves, we sweep over the decision threshold (β) of the gating network, trading its true-positive-rate with its false-positive-rate. In the blue-square we show our operating point which was selected with cross-validation by choosing the best Acc_H on GZSL-Val set.

An interesting observation is that different types of models populate different regions of the Seen-Unseen curve. Generative models (X markers) tend to favor unseen-classes accuracy over accuracy of seen classes, while non-generative models (triangles) tend to favor seen classes. Importantly, COSMO can be tuned to select any operation point along the curve, and achieve better or equivalent performance at all regions of the seen-unseen plane.

In Supplementary Figure S.2, we provide the curves of COSMO+fCLSWGAN for AWA, SUN, CUB and FLOWER.

Table 2 reports the AUSUC of COSMO compared to four baseline models. To produce the full curve for the baselines, we used the code provided by the authors and applied

GATING	AWA Acc_H	AUC	FPR	SUN Acc_H	AUC	FPR	CUB Acc_H	AUC	FPR
MAX-SOFTMAX-1	52.9	86.7	67.9	38.4	60.9	92.7	43.4	74.1	82.3
MAX-SOFTMAX-3	53.1	88.6	56.6	38.4	61	92.2	43.7	73.4	79.9
CB-GATING-3 (w/o p^{ZS})	52.8	88.8	56.4	38.4	61	91.6	43.7	74.2	80.4
CB-GATING-1	53.9	85.9	58.6	39.8	72.2	78.7	45.1	79.6	67.0
CB-GATING-3	56.6	91.2	39.8	40.1	72.2	82.5	44.8	80.5	70.7

Table 3. Ablation study for various gating model variants on validation set. AUC denotes Area-Under-Curve when sweeping over detection threshold. FPR denotes False-Positive-Rate on the threshold that yields 95% True Positive Rate for detecting in-distribution samples.

calibrated-stacking [9] with a series of constants. On all four datasets, COSMO improves AUSUC for LAGO and fCLSWGAN. COSMO also introduces new state-of-the-art AUSUC on AWA, SUN and FLOWER.

7.2. Ablation experiments

To understand the contribution of the different modules of COSMO, we carried ablation experiments on COSMO+LAGO that quantify the benefits of the CBG network and adaptive smoothing.

We first compared variants of the gating model, then compared variants of the smoothing method, and finally, compared how these modules work together.

Confidence-Based Gating : Table 3 describes: (1) The OOD metrics *AUC* and *False-Positive-Rate at 95% True-Positive-Rate* on *Gating-Val*. (2) Acc_H metric on *GZSL-Val*.

We test the effect of temperature scaling and of confidence-based gating by comparing the following gating models: (1) *CB-Gating-3* is our best confidence-based gating model, from Section 4.1 with temperature $T = 3$. (2) *CB-Gating-1* is the same model with $T = 1$, revealing the effect of temperature scaling [26]. (3) *CB-Gating-3 (w/o p^{ZS})* is like *CB-Gating-3* without the inputs from the ZS expert, revealing the importance of utilizing information from both experts. (4) *Max-Softmax-1* is a baseline gating model of [15], instead of the CBG network, it classifies S/U by comparing the largest softmax score to a threshold. (5) *Max-Softmax-3* is like *Max-Softmax-1*, but with $T = 3$. In these experiments, smoothing is disabled to only quantify factors related to the gating model.

We find that both temperature scaling and confidence-based gating improves the quality metrics. Importantly, confidence-based gating has a strong contribution to performance: The AUC increases from 86.3 to 91.2 for AWA, 60.9 to 72.2 for SUN and 74.1 to 80.5 for CUB.

Adaptive Confidence Smoothing: Table 4 shows the contribution of adaptive smoothing to Acc_H on the validation set. In these experiments, gating is disabled, to only quantify factors related to the smoothing. (1) *Adaptive-Smoothing* corresponds to Eq. (4). (2) *Const-Smoothing* uses a constant smoothing weight λ (Eq. 3) for all images. λ was selected by cross validation. *Adaptive-Smoothing* shows superior performance to *Const-Smoothing* (AWA:

54.3% vs 53%, SUN: 41.4% vs 38.4%, CUB: 45.7% vs 43.6%) and even to not using smoothing at all ($\lambda = 0$).

	AWA	SUN	CUB
$\lambda = 0$	52.9	38.4	43.4
CONST-SMOOTHING	53	38.6	43.6
ADAPTIVE-SMOOTHING	54.3	41.4	45.7

Table 4. Ablation study for adaptive smoothing. Showing Acc_H on GZSL-Val set.

	AWA	SUN	CUB
INDEPENDENT-HARD	58.3	35.1	44.6
INDEPENDENT-SOFT	57.7	37.3	46.8
CB-GATING	64.3	39.7	49.1
ADAPTIVE-SMOOTHING	63.6	40.8	49.6
COSMO (GATING & SMOOTHING)	63.6	41.0	50.2

Table 5. Ablation study for combining smoothing and gating, showing Acc_H on the test set.

Combining gating and smoothing: Table 5 reports test- Acc_H when ablating the main modules of COSMO :

(1) *Independent-Hard* is the simplest approach, applying Eq. (2) where the modules don’t exchange information, and the gating is a hard decision over “Max of Softmax” [15]. It resembles reproducing CMT [39] but using LAGO as the ZSL model and Max-of-softmax gate. (2) *Independent-Soft* uses a soft gating following Eq. (5). (3) *CB-Gating* applies the CBG network (Section 4.1) for the gating module. (4) *Adaptive-Smoothing* uses Eq. (4), and max-of-softmax gating. (5) *COSMO* is our best model, applying both CB-Gating and adaptive smoothing.

We find that both adaptive confidence smoothing and confidence-based gating contribute to test accuracy. Compared with Independent-Hard, COSMO shows a relative improvement from 58.3% to 63.6% on AWA, 35.1% to 41% on SUN and 44.6% to 50.2% on CUB. Adaptive confidence smoothing and confidence-based gating are weakly synergistic, providing a 1.2% relative improvement for CUB, 0.5% for SUN and -1.1% for AWA.

Importantly, accuracy of COSMO is comparable with state-of-the-art generative models. This is important because COSMO is much easier to train and tune than GAN-based approaches.

References

- [1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for image classification. *TPAMI*, 2016.
- [2] Z. Akata, S. Reed, D. Walter, Honglak Lee, and B. Schiele. Evaluation of output embeddings for fine-grained image classification. In *CVPR*, 2015.
- [3] Z. Al-Halah and R. Stiefelhagen. How to transfer? zero-shot object recognition via hierarchical transfer of semantic attributes. In *WACV*, 2015.
- [4] J. Andreas, M. Rohrbach, T. Darrell, and Klein D. Learning to compose neural networks for question answering. In *NAACL*, 2016.
- [5] G. Arora, V-K. Verma, A. Mishra, and P. Rai. Generalized zero-shot learning via synthesized examples. In *CVPR*, 2018.
- [6] Y. Atzmon, J. Berant, V. Kezami, A. Globerson, and G. Chechik. Learning to generalize to new compositions in image understanding. *arXiv preprint arXiv:1608.07639*, 2016.
- [7] Yuval Atzmon and Gal Chechik. Probabilistic and-or attribute grouping for zero-shot learning. In *UAI*, 2018.
- [8] S. Changpinyo, W. L. Chao, B. Gong, and F. Sha. Synthesized classifiers for zero-shot learning. In *CVPR*, 2016.
- [9] R. Chao, S. Changpinyo, B. Gong, and Sha F. An empirical study and analysis of generalized zero-shot learning for object recognition in the wild. In *ICCV*, 2016.
- [10] R. Felix, V. Kumar, I. Reid, and G. Carneiro. Multi-modal cycle-consistent generalized zero-shot learning. In *ECCV*, 2018.
- [11] R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, New York, NY, USA, second edition, 1987.
- [12] A. Frome, G. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013.
- [13] Y. Fu, T. Xiang, Y-G Jiang, X. Xue, L. Sigal, and S. Gong. Recent advances in zero-shot recognition. *IEEE Signal Processing Magazine*, 2018.
- [14] Y. Guo, G. Ding, J. Han, S. Zhao, and B. Wang. Implicit non-linear similarity scoring for recognizing unseen classes. In *IJCAI*, 2018.
- [15] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.
- [16] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.
- [17] R. Jacobs, S. Nowlan, M. Jordan, and G. Hinton. Adaptive mixtures of local experts. *Neural computation*, 1991.
- [18] D. Jayaraman and K. Grauman. Zero-shot recognition with unreliable attributes. In *NIPS*, 2014.
- [19] C. Kemp, JB. Tenenbaum, T.L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *AAAI*, 2006.
- [20] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
- [21] B. Lake. *Towards more human-like concept learning in machines: compositionality, causality, and learning-to-learn*. Massachusetts Institute of Technology, 2014.
- [22] B. Lake, T. Ullman, J. Tenenbaum, and Gershman S. Building machines that learn and think like people. In *Behavioral and Brain Sciences*, 2017.
- [23] C.H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, 2009.
- [24] C.H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *TPAMI*, 2014.
- [25] Y. Li, D. Wang, H. Hu, Y. Lin, and Y. Zhuang. Zero-shot recognition using dual visual-semantic mapping paths. In *CVPR*, 2017.
- [26] S. Liang, Y. Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.
- [27] S. Liu, M. Long, J. Wang, and M. Jordan. Generalized zero-shot learning with deep calibration network. In *NIPS*, 2018.
- [28] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.
- [29] A. Mishra, M. Reddy, A. Mittal, and H. A. Murthy. A generative model for zero shot learning using conditional variational autoencoders. In *WACV*, 2018.
- [30] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *ICCVGI*, 2008.
- [31] D. N. Osherson, J. Stern, O. Wilkie, M. Stob, and E. Smith. Default probability. *Cognitive Science*, 15(2), 1991.
- [32] G. Patterson and J. Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*, 2012.
- [33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *JMLR*, 2011.
- [34] S. Reed, Z. Akata, H. Lee, and B. Schiele. Learning deep representations of fine-grained visual descriptions. In *CVPR*, 2016.
- [35] M. Rohrbach, M. Stark, and B. Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *CVPR*, 2011.
- [36] B. Romera-Paredes and P. Torr. An embarrassingly simple approach to zero-shot learning. In *ICML*, 2015.
- [37] G. Shalev, Y. Adi, and J. Keshet. Out-of-distribution detection using multiple semantic label representations. In *NIPS*, 2018.
- [38] N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. Le, G. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *ICLR*, 2017.
- [39] R. Socher, M. Ganjoo, C.D. Manning, and A. Ng. Zero-shot learning through cross-modal transfer. In *NIPS*, 2013.
- [40] J. Song, C. Shen, Y. Yang, Y. Liu, and M. Song. Transductive unbiased embedding for zero-shot learning. In *CVPR*, 2018.
- [41] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. HS Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018.
- [42] A. Vyas, N. Jammalamadaka, X. Zhu, S. Das, B. Kaul, and T. L. Willke. Out-of-distribution detection using an ensemble of self supervised leave-out classifiers. In *ECCV*, 2018.

- [43] 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.
- [44] X. Wang and Q. Ji. A unified probabilistic approach modeling relationships between attributes and objects. In *ICCV*, 2013.
- [45] Y. Xian, C.H. Lampert, B. Schiele, and Z. Akata. Zero-shot learning - A comprehensive evaluation of the good, the bad and the ugly. *TPAMI*, 2018.
- [46] Y. Xian, T. Lorenz, B. Schiele, and Z. Akata. Feature generating networks for zero-shot learning. In *CVPR*, 2018.
- [47] Y. Xian, B. Schiele, and Z. Akata. Zero-shot learning - the good, the bad and the ugly. In *CVPR*, 2017.
- [48] X. Xu, F. Shen, Y. Yang, D. Zhang, H. T. Shen, and J. Song. Matrix tri-factorization with manifold regularizations for zero-shot learning. In *CVPR*, 2017.
- [49] S. E. Yuksel, J. N. Wilson, and P. D. Gader. Twenty years of mixture of experts. *IEEE Transactions on Neural Networks and Learning Systems*, 2012.
- [50] Hongguang Zhang and Piotr Koniusz. Zero-shot kernel learning. In *CVPR*, 2018.
- [51] H. Zhang, Y. Long, Y. Shao Guan, and L. Triple verification network for generalized zero-shot learning. *IEEE Transactions on Image Processing*, 2019.
- [52] L. Zhang, T. Xiang, and S. Gong. Learning a deep embedding model for zero-shot learning. In *CVPR*, 2017.
- [53] A. Zhao, M. Ding, J. Guan, Z. Lu, T. Xiang, and J-R Wen. Domain-invariant projection learning for zero-shot recognition. In *NIPS*, 2018.
- [54] Y. Zhu, M. Elhoseiny, B. Liu, X. Peng, and A. Elgammal. A generative adversarial approach for zero-shot learning from noisy texts. In *CVPR*, 2018.