Zero-Shot Learning via Semantic Similarity Embedding

Ziming Zhang and Venkatesh Saligrama
Department of Electrical & Computer Engineering, Boston University
{zzhang14, srv}@bu.edu

Abstract

*In this paper we consider a version of the zero-shot learning problem where seen class source and target domain data are provided. The goal during test-time is to accurately predict the class label of an unseen target domain instance based on revealed source domain side information (*e.g. *attributes) for unseen classes. Our method is based on viewing each source or target data as a mixture of seen class proportions and we postulate that the mixture patterns have to be similar if the two instances belong to the same unseen class. This perspective leads us to learning source/target embedding functions that map an arbitrary source/target domain data into a same semantic space where similarity can be readily measured. We develop a max-margin framework to learn these similarity functions and jointly optimize parameters by means of cross validation. Our test results are compelling, leading to significant improvement in terms of accuracy on most benchmark datasets for zero-shot recognition.*

1. Introduction

While there has been significant progress in large-scale classification in recent years [31], lack of sufficient training data for every class and the increasing difficulty in finding annotations for a large fraction of data might impact further improvements.

Zero-shot learning is being increasingly recognized as a way to deal with these difficulties. One version of zero shot learning is based on so-called source and target domains. Source domain is described by a *single* vector corresponding to each class based on side information such as *attributes* [8, 16, 21, 25, 29], *language words/phrases* [4, 9, 34], or even *learned classifiers* [42], which we assume can be collected easily. The target domain is described by a joint distribution of images/videos and labels [16, 41]. During training time, we are given source domain attributes and target domain data corresponding to only a subset of classes, which we call seen classes. During test time, source domain attributes for unseen (*i.e.* no training data provided) classes are revealed. The goal during test time is to predict for each target domain instance which of the seen/unseen classes it is associated with.

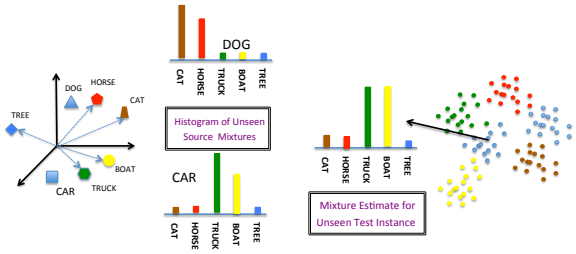


Figure 1. Proposed method with source/target domain data displayed on the leftmost/rightmost figures respectively. Light blue corresponds to unseen classes and other colors depict seen class data. Light-blue data is unavailable during training. During test-time unseen source domain data is revealed along with an arbitrary unseen instance from target domain (light-blue) is presented and we are to identify its unseen class label. Each unseen class source domain data is expressed as a histograms of seen class proportions. Seen class proportions are estimated for the target instance and compared with each of the source domain histograms.

Key Idea: Our proposed method is depicted in Fig. 1. We view target data instances as arising from seen instances and attempt to express source/target data as a mixture of seen class proportions. Our algorithm is based on the postulate that if the mixture proportion from target domain is similar to that from source domain, they must arise from the same class. This leads us to learning source and target domain embedding functions using seen class data that map arbitrary source and target domain data into mixture proportions of seen classes.

We propose parameterized-optimization problems for learning *semantic similarity embedding* (SSE) functions from training data and jointly optimize predefined parameters using cross validation on held-out seen class data. Our method necessitates fundamentally new design choices requiring us to learn class-dependent feature transforms because components of our embedding must account for contribution of each seen class. Our source domain embedding is based on subspace clustering literature [37] that are known to be resilient to noise. Our target domain embedding is based on a margin-based framework using the intersection function or the rectified linear unit (ReLU)

[22], which attempts to align seen class source domain data with their corresponding seen class target domain data instances. Finally, we employ a cross validation technique based on holding out seen class data and matching held-out seen classes to optimize parameters used in the optimization problems for source and target domain. In this way we jointly optimize parameters to best align mixture proportions for held-out seen classes and provide a basis for generalizing to unseen classes. Results on several benchmark datasets for zero-shot learning demonstrate that our method significantly improves the current state-of-the-art results.

Related Work: Most existing zero-shot learning methods rely on predicting side information for further classification. [24] proposed a semantic (*i.e*. attribute) output code classifier which utilizes a knowledge base of semantic properties. [16, 39] proposed several probabilistic attribute prediction methods. [42] proposed designing discriminative category-level attributes. [18] proposed an optimization formulation to learn source domain attribute classifiers and attribute vectors jointly. [20] proposed learning the classifiers for unseen classes by linearly combining the classifiers for seen classes. [1] proposed a label embedding method to embed each class into an attribute vector space. [2, 9, 23, 34] directly learned the mapping functions between the feature vectors in source and target domains with deep learning. Such methods may suffer from noisy (*e.g*. missing or incorrectly annotated) side information or data bias, leading to unreliable prediction.

Some recent work has been proposed to overcome some issues above. [28] proposed a propagated semantic transfer method by exploiting unlabeled instances. [10] discussed the projection domain shift problem and proposed a transductive multi-view embedding method. [14] investigated the attribute unreliability issue and proposed a random forest approach. [30] proposed a simple method by introducing a better regularizer.

An important conceptual difference that distinguishes our method from other existing works such as [1, 2], is that these methods can be interpreted as learning relationships between source attributes and target feature components (in the encoded space), while our method is based on leveraging similar class relationships (semantic affinities) in source and target domains, requiring class dependent feature transform. This leads to complex scoring functions, which cannot be simplified to linear or bilinear forms as in [1, 2].

Semantic similarity embedding (SSE) is widely used to model the relationships among classes, which is quite insensitive to instance level noise. [40] proposed learning mapping functions to embed input vectors and classes into a low dimensional common space based on class taxonomies. [3] proposed a label embedding tree method for large multi-class tasks, which also embeds class labels in a low dimensional space. [12] proposed an analogy-preserving semantic embedding method for multi-class classification. Later [13] proposed a unified semantic embedding method to incorporate different semantic information into learning. Recently [23] proposed a semantic embedding method for zero-shot learning to embed an unseen class as a convex combination of seen classes with heuristic weights. [11] proposed a semantic ranking representation based on semantic similarity to aggregate semantic information from multiple heterogeneous sources. Our embedding is to represent each class as a mixture of seen classes in both domains.

Notation	Definition
$\mathcal{S}$ ($\mathcal{U}$)	Set of seen (unseen) classes
$\lvert\mathcal{S}\rvert$	Number of seen classes
s (or y) & u	Indexes for seen and unseen classes
$\Delta^{\lvert\mathcal{S}\rvert}$	Simplex in $\mathbb{R}^{\lvert\mathcal{S}\rvert}$ dimensional space
$\{\mathbf{c}_y\}$	Source domain attribute vector $\mathbf{c}_y \in \mathbb{R}^{d_s}$ for class y with ℓ_2 normalization, *i.e*. $\lVert\mathbf{c}_y\rVert = 1$.
$\{(\mathbf{x}_i, y_i)\}$	Training data: $\mathbf{x}_i \in \mathbb{R}^{d_t}$ - target feature, y_i - class
$N(N_y)$	Number of training samples (for class $y \in \mathcal{S}$)
ψ, π	Source/Target domain feature embedding functions
ϕ_y	Target domain class dependent feature transformation
$(\cdot)_{m,n}$	The nth entry in vector $(\cdot)_m$
$\mathbf{z}_y = \psi(\mathbf{c}_y)$	**Learned** source domain embedded histogram $\mathbf{z}_y \in \Delta^{\lvert\mathcal{S}\rvert}$ for class y.
$\mathcal{V} = \{\mathbf{v}_y\}$	**Learned** target domain reference vector $\mathbf{v}_y \in \mathbb{R}^{d_t}$ for class y, one vector per seen class
$\mathbf{w}$	**Learned** target domain weight vector
$f(\mathbf{x}, y)$	**Learned** structured scoring function relating the target domain sample $\mathbf{x}$ and class label y.

Table 1. Some notation used in our method.

2. Zero-Shot Learning and Prediction

Our notation is summarized in Table 1 for future reference.

2.1. Overview

Our method is based on expressing source/target data as a mixture of seen class proportions (see Fig. 1). Using seen class data we learn source and target domain embedding functions, ψ, π respectively. Our aim is to construct functions that take an arbitrary source vectors $\mathbf{c}$ and target vectors $\mathbf{x}$ as inputs and embed them into $\Delta^{\lvert\mathcal{S}\rvert}$ (histograms). Observe that components, $\pi_y(\mathbf{x})$, $\psi_y(\mathbf{c})$ of $\pi(\mathbf{x})$, $\psi(\mathbf{c})$, corresponding to seen class $y \in \mathcal{S}$, denote the proportion of class y in the instance $\mathbf{x}$, $\mathbf{c}$. During test-time source domain vectors $\mathbf{c}_u \in \mathcal{C}$ for all the unseen classes are revealed. We are then presented with an arbitrary target instance $\mathbf{x}$. We predict an unseen label for $\mathbf{x}$ by maximizing the semantic similarity between the histograms. Letting $\mathbf{z}_u = \psi(\mathbf{c}_u)$, then our zero-shot recognition rule is defined as follows:

$$u^* = \arg\max_{u \in \mathcal{U}} f(\mathbf{x}, u) = \arg\max_{u \in \mathcal{U}} \langle \pi(\mathbf{x}), \mathbf{z}_u \rangle, \qquad (1)$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product of two vectors.

We propose parameterized-optimization problems to learn embedding functions from seen class data. We then

optimize these parameters globally using held-out seen class data. We summarize our learning scheme below.

(A) *Source Domain Embedding Function (ψ):* Our embedding function is realized by means of a parameterized optimization problem, which is related to sparse coding.

(B) *Target Domain Embedding Function (π):* We model $\pi_y(\mathbf{x})$ as $\langle \mathbf{w}, \phi_y(\mathbf{x}) \rangle$. This consists of a constant weight vector $\mathbf{w}$ and a class dependent feature transformation $\phi_y(\mathbf{x})$. We propose a margin-based optimization problem to jointly learn both the weight vector and the feature transformation. Note that our parameterization may yield negative values and may not be normalized, which can be incorporated as additional constraints but we ignore this issue in our optimization objectives.

(C) *Cross Validation:* Our embedding functions are parameter dependent. We choose these parameters by employing a cross validation technique based on holding out seen class data. First, we learn embedding functions (see (A) and (B)) on the remaining (not held-out) seen class data with different values of the predefined parameters. We then jointly optimize parameters of source/target embedding functions to minimize the prediction error on held-out seen classes. In the end we re-train the embedding functions over the entire seen class data.

Salient Aspects of Proposed Method:

(a) *Decomposition:* Our method seeks to decompose source and target domain instances into mixture proportions of seen classes. In contrast much of the existing work can be interpreted as learning cross-domain similarity between source domain attributes and target feature components.

(b) *Class Dependent Feature Transformation $\pi_y(\mathbf{x})$:* The decomposition perspective necessitates fundamentally new design choices. For instance, $\pi_y(\mathbf{x})$, the component corresponding to class y must be dependent on y, which implies that we must choose a class dependent feature transform $\phi_y(\mathbf{x})$ because $\mathbf{w}$ is a constant vector and agnostic to class.

(c) *Joint Optimization and Generalization to Unseen Classes:* Our method jointly optimizes parameters of the embedding functions to best align source and target domain histograms for held-out seen classes, thus providing a basis for generalizing to unseen classes. Even for fixed parameters, embedding functions ψ, π are **nonlinear** maps and since the parameters are jointly optimized our learned scoring function $f(\mathbf{x}, y)$ couples seen source and target domain together in a rather complex way. So we cannot reduce $f(\cdot, \cdot)$ to a linear or bilinear setting as in [2].

2.2. Intuitive Justification of Proposed Method

Recall that our method is based on viewing unseen source and target instances as a histogram of seen classes proportions. Fig. 1 suggests that a target instance can be viewed as arising from a mixture of seen classes with mixture components dependent on the location of the instance. More precisely, letting P and P_y be the unseen and seen class-conditional target feature distributions respectively, we can a priori approximate P as a mixture of the P_y's, *i.e.* $P = \sum_{y \in \mathcal{S}} \bar{\pi}_y P_y + P_{\text{error}}$ (see [5] for various approaches in this context), where $\bar{\pi}_y$ denotes the mixture weight for class y. Analogously, we can also decompose source domain data as a mixture of source domain seen classes. This leads us to associate mixture proportion vector $\mathbf{z}_u$ with unseen class u, and represent attribute vector $\mathbf{c}_u$ as $\mathbf{c}_u \approx \sum_{y \in \mathcal{S}} z_{u,y} \mathbf{c}_y$, with $\mathbf{z}_u = (z_{u,y})_{y \in \mathcal{S}} \in \Delta^{|\mathcal{S}|}$.

Key Postulate: The target domain instance, $\mathbf{x}$, must have on average a similar mixture pattern as the source domain pattern if they both correspond to the same unseen label, $u \in \mathcal{U}$, namely, on average $\pi(\mathbf{x})$ is equal to $\mathbf{z}_u$.

This postulate is essentially Eq. 1. This postulate also motivates our margin-based approach for learning $\mathbf{w}$. Note that since we only have a single source domain vector for each class, a natural constraint is to require that the empirical mean of the mixture corresponding to each example per class in target domain aligns well with the source domain mixture. This is empirically consistent with our postulate. Letting y, y' be seen class labels with $y \neq y'$ and $\bar{\boldsymbol{\pi}}_y$ denote the average mixture for class y in target domain, our requirement is to guarantee that

$$\langle \bar{\boldsymbol{\pi}}_y, \mathbf{z}_y \rangle \geq \langle \bar{\boldsymbol{\pi}}_y, \mathbf{z}_{y'} \rangle \tag{2}$$

$$\Leftrightarrow \sum_{s \in \mathcal{S}} \left\langle \mathbf{w}, \underbrace{\frac{1}{N_s} \sum_{i=1}^{N} \mathbb{I}_{\{y_i = s\}} \phi_s(\mathbf{x}_i)}_{\text{Emp. Mean Embedding}} \right\rangle \Big(z_{y,s} - z_{y',s} \Big) \geq 0,$$

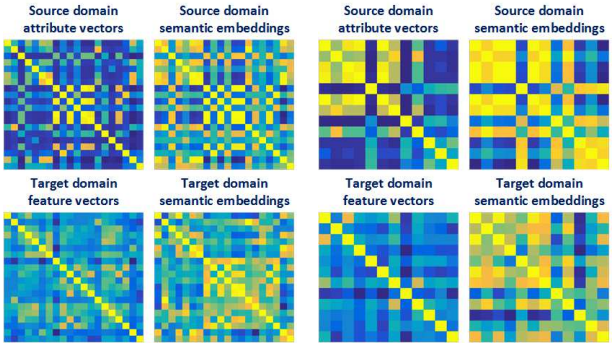
where $\mathbb{I}_{\{\cdot\}}$ denotes a binary indicator function returning 1 if the condition holds, otherwise 0. Note that the empirical mean embedding corresponds to a kernel empirical mean embedding [33] if ϕ_s is a valid (characteristic) RKHS kernel, but we do not pursue this point further in this paper. Nevertheless this alignment constraint is generally insufficient, because it does not capture the shape of the underlying sample distribution. We augment misclassification constraints for each seen sample in SVMs to account for shape.

2.3. Source Domain Embedding

Recall from Fig. 1 and **(B)** in Sec. 2.1 that our embedding aims to map source domain attribute vectors $\mathbf{c}$ to histograms of seen class proportions, *i.e.* $\psi : \mathbb{R}^{d_s} \rightarrow \Delta^{|\mathcal{S}|}$. We propose a parameterized optimization problem inspired by sparse coding as follows, given a source domain vector $\mathbf{c}$:

$$\psi(\mathbf{c}) = \underset{\boldsymbol{\alpha} \in \Delta^{|\mathcal{S}|}}{\arg\min} \left\{ \frac{\gamma}{2} \|\boldsymbol{\alpha}\|^2 + \frac{1}{2} \|\mathbf{c} - \sum_{y \in \mathcal{S}} \mathbf{c}_y \alpha_y\|^2 \right\}, \tag{3}$$

where $\gamma \geq 0$ is a predefined regularization parameter, $\|\cdot\|$ denotes the ℓ_2 norm of a vector, and $\boldsymbol{\alpha} = (\alpha_y)_{y \in \mathcal{S}}$ describes contributions of different seen classes. Note that



(a) Seen classes in training (b) Unseen classes in testing

Figure 2. Cosine similarity matrices among (a) seen and (b) unseen classes on aPascal & aYahoo [8] dataset. Brighter color depicts larger values. The type of data used to compute the matrix is shown above the corresponding matrix. Observe that in training/testing our source/target domain embedding preserves the inter-class relationships originally defined by the source domain attribute vectors. This also indicates that our target domain embeddings manage to align well the target domain distributions with the source domain attribute vectors.

even though $\mathbf{c}$ may not be on the simplex, the embeddings $\psi(\mathbf{c})$ are always. Note that the embedding ψ is in general a nonlinear function. Indeed on account of simplex constraint small values in $\boldsymbol{\alpha}$ vector are zeroed out (*i.e.* "water-filling").

To solve Eq. 3, we use quadratic programming. For large-scale cases, we adopt efficient proximal gradient descent methods. Note that there are many alternate ways of embedding such as similarity rescaling, subspace clustering [27], sparse learning [7], and low rank representation [17], as long as the embedding is on the simplex. We tried these different methods with the simplex constraint to learn the embeddings, and our current solution in Eq. 3 works best. We believe that it is probably because the goal in these other methods is subspace clustering, while our goal is to find a noise resilient embedding which has good generalization to unseen class classification.

We optimize the parameter, γ, globally by cross validation. Once the γ parameter is identified, all of the seen classes are used in our embedding function. Note that when $\gamma = 0$ or small, $\psi(\mathbf{c}_y)$ will be a coordinate vector, which essentially amounts to coding for multi-class classification but is not useful for unseen class generalization. Conceptually, because we learn tuning parameters to predict well on held-out seen classes, γ is in general not close to zero. We demonstrate class affinity matrices before and after embedding for both seen and unseen classes in Fig. 2. Here $\gamma = 10$ is obtained by cross validation. We see that in both training and testing source domain embeddings preserve the affinities among classes in the attribute space.

During test-time when unseen class attribute vectors $\mathbf{c}_u$ are revealed, we obtain $\mathbf{z}_u$ as the embeddings using Eq. 3 with the learned γ.

2.4. Target Domain Embedding

In this paper we define our target domain class dependent mapping function ϕ_y based on (1) intersection function (INT) [19], or (2) rectified linear unit (ReLU) [22]. That is,

$$\text{INT:} \quad \phi_y(\mathbf{x}) = \min(\mathbf{x}, \mathbf{v}_y), \tag{4}$$

$$\text{ReLU:} \quad \phi_y(\mathbf{x}) = \max(\mathbf{0}, \mathbf{x} - \mathbf{v}_y), \tag{5}$$

where min and max are the entry-wise operators. Note that intersection function captures the data patterns in $\mathbf{x}$ below the thresholds in each $\mathbf{v}_y$, while ReLU captures the data patterns above the thresholds. In this sense, the features generated from these two functions are complementary. This is the reason that we choose the two functions to demonstrate the robustness of our method.

Based on Eq. 1 and 2 in Section 2.1, we define the following structured scoring function $f(\mathbf{x}, y)$ as follows:

$$f(\mathbf{x}, y) = \sum_{s \in \mathcal{S}} \langle \mathbf{w}, \phi_s(\mathbf{x}) \rangle \, z_{y,s}. \tag{6}$$

In test-time for target instance $\mathbf{x}$, we can compute $f(\mathbf{x}, u)$ for an arbitrary unseen label u because the source attribute vector is revealed for u. Note that f is highly non-convex, and it cannot reduce to bilinear functions used in existing works such as [1, 2].

2.4.1 Max-Margin Formulation

Based on Eq. 6, we propose the following parameterized learning formulation for zero-shot learning as follows, which learns the embedding function π, and thus f:

$$\min_{\mathcal{V}, \mathbf{w}, \boldsymbol{\xi}, \boldsymbol{\epsilon}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{\lambda_1}{2} \sum_{\mathbf{v} \in \mathcal{V}} \|\mathbf{v}\|^2 + \lambda_2 \sum_{y,s} \epsilon_{ys} + \lambda_3 \sum_{i,y} \xi_{iy} \tag{7}$$

$$\text{s.t. } \forall i \in \{1, \cdots, N\}, \forall y \in \mathcal{S}, \forall s \in \mathcal{S},$$

$$\sum_{i=1}^{N} \frac{\mathbb{I}_{\{y_i = y\}}}{N_y} \Big[f(\mathbf{x}_i, y) - f(\mathbf{x}_i, s) \Big] \geq \Delta(y, s) - \epsilon_{ys}, \tag{8}$$

$$f(\mathbf{x}_i, y_i) - f(\mathbf{x}_i, y) \geq \Delta(y_i, y) - \xi_{iy}, \tag{9}$$

$$\epsilon_{ys} \geq 0, \xi_{iy} \geq 0, \forall \mathbf{v} \in \mathcal{V}, \mathbf{v} \geq \mathbf{0},$$

where $\Delta(\cdot, \cdot)$ denotes a structural loss between the ground-truth class and the predicted class, $\lambda_1 \geq 0$, $\lambda_2 \geq 0$, and $\lambda_3 \geq 0$ are the predefined regularization parameters, $\boldsymbol{\xi} = \{\xi_{iy}\}$ and $\boldsymbol{\epsilon} = \{\epsilon_{ys}\}$ are slack variables, and $\mathbf{0}$ is a vector of 0's. In this paper, we define $\Delta(y_i, y) = 1 - \mathbf{c}_{y_i}^T \mathbf{c}_y$ and $\Delta(y, s) = 1 - \mathbf{c}_y^T \mathbf{c}_s$, respectively. Note that in learning we only access and utilize the data from seen classes.

In fact, Eq. 8 measures the alignment loss for each seen class distribution, and Eq. 9 measures the classification loss for each target domain training instance, respectively, which

(a) Distr. alignment (b) Inst. classification (c) Our method

Figure 3. Illustration of three different constraints for learning the target domain semantic embedding function. Different shapes denote differnt classes, fill-in shapes denote the source domain embeddings, and green crosses denote the empirical means of target domain data embeddings. Our method takes into account the zero-shot learning based on both distribution alignment and instance classification.

correspond to the discussion in Sec. 2.2. On one hand, if we only care about the alignment condition, it is likely that there may be many misclassified training data samples (*i.e.* loose shape) as illustrated in Fig. 3(a). On the other hand, conventional classification methods only consider separating data instances with tight shape, but are unable to align distributions due to lack of such constraint in training (see Fig. 3(b)). By introducing these two constraints into Eq. 7, we are able to learn the target domain embedding function as well as the scoring function to produce the clusters which are well aligned and separated, as illustrated in Fig. 3(c).

Similarly, we learn the predefined parameters λ_1, λ_2, λ_3 through a cross validation step that optimizes the prediction for held-out seen classes. Then once the parameters are determined we re-learn the classifier on all of the seen data. Fig. 2 depicts class affinity matrices before and after target domain semantic embedding on real data. Our method manages to align source/target domain data distributions.

2.4.2 Alternating Optimization Scheme

To solve Eq. 7, we propose the following alternating optimization algorithm, as seen in Alg. 1.

Algorithm 1 Learning Embedding Functions

Input : $\{\mathbf{x}_i, y_i\}, \{\mathbf{c}_y\}_{y\in\mathcal{S}}, \{\mathbf{z}_y\}_{y\in\mathcal{S}}, \lambda_1, \lambda_2, \lambda_3$, learning rate $\eta \geq 0$
Initialize $\boldsymbol{\nu}^{(0)}$ with feature means of seen classes in target domain;
for $t = 0$ **to** τ **do**
  $(\mathbf{w}, \boldsymbol{\epsilon}, \boldsymbol{\xi}) \leftarrow$ linearSVM_solver$(\{\mathbf{x}_i, y_i\}, \boldsymbol{\nu}^{(t)}, \lambda_2, \lambda_3)$;
  $\boldsymbol{\nu}^{(t+1)} \leftarrow \max\{\mathbf{0}, \boldsymbol{\nu}^{(t)} - \eta\nabla h(\boldsymbol{\nu}^{(t)})\}$; `// using Eq.11`
  Check monotonic decreasing condition on the objective in Eq. 7;
end
Output: $\mathbf{w}, \boldsymbol{\nu}$

(i) Learning $\mathbf{w}$ by fixing $\mathcal{V}$: In this step, we can collect all the constraints in Eq. 8 and Eq. 9 by plugging in $\{(\mathbf{x}_i, y_i)\}, \mathcal{V}, \{\mathbf{c}_y\}_{y\in\mathcal{S}}$, and then solve a linear SVM to learn $\mathbf{w}, \boldsymbol{\epsilon}, \boldsymbol{\xi}$, respectively.

(ii) Learning $\mathcal{V}$ by fixing $\mathbf{w}$ using Concave-Convex procedure (CCCP) [43]: Note that the constraints in Eq. 8 and Eq. 9 consist of difference-of-convex (DoC) functions. To see this, we can rewrite $f(\mathbf{x}_i, y) - f(\mathbf{x}_i, y_i)$ as a summation of convex and concave functions as follows:

$$f(\mathbf{x}_i, y) - f(\mathbf{x}_i, y_i) = \sum_{m,s} w_m(z_{y,n} - z_{y_i,n})\phi_{s,m}(\mathbf{x}_i), \tag{10}$$

where w_m and $\phi_{s,m}(\cdot)$ denote the mth entries in vectors $\mathbf{w}$ and $\phi_s(\cdot)$, respectively. Let $\boldsymbol{\nu} \in \mathbb{R}^{d_t|\mathcal{S}|}$ be a vector concatenation of all $\mathbf{v}$'s, $g_1(\boldsymbol{\nu}) \triangleq g_1(\mathbf{x}_i, y, \boldsymbol{\nu})$ and $g_2(\boldsymbol{\nu}) \triangleq g_2(\mathbf{x}_i, y, \boldsymbol{\nu})$ denote the summations of all the convex and all the concave terms in Eq. 10, respectively. Then we have $f(\mathbf{x}_i, y) - f(\mathbf{x}_i, y_i) = g_1(\boldsymbol{\nu}) - (-g_2(\boldsymbol{\nu}))$, *i.e.* DoC functions. Using CCCP we can relax the constraint in Eq. 9 as $\xi_{iy} \geq \Delta(y_i, y) + g_1(\boldsymbol{\nu}) + g_2(\boldsymbol{\nu}^{(t)}) + \nabla g_2(\boldsymbol{\nu}^{(t)})^T(\boldsymbol{\nu} - \boldsymbol{\nu}^{(t)})$, where $\boldsymbol{\nu}^{(t)}$ denotes the solution for $\boldsymbol{\nu}$ in iteration t, and ∇ denotes the subgradient operator. Similarly we can perform CCCP to relax the constraint in Eq. 8. Letting $h(\boldsymbol{\nu})$ denote the minimization problem in Eq. 7, 8, and 9, using CCCP we can further write down the subgradient $\nabla h(\boldsymbol{\nu}^{(t)})$ in iteration $t+1$ as follows:

$$\begin{aligned}\nabla h(\boldsymbol{\nu}^{(t)}) &= \lambda_1\boldsymbol{\nu}^{(t)} \\ &+ \lambda_2 \sum_{y,s,i} \mathbb{I}_{\{\epsilon_{ys}>0, y_i=y\}} \left[\nabla g_1(\boldsymbol{\nu}^{(t)}) + \nabla g_2(\boldsymbol{\nu}^{(t)})\right] \\ &+ \lambda_3 \sum_{y_i,y} \mathbb{I}_{\{\xi_{iy}>0\}} \left[\nabla g_1(\boldsymbol{\nu}^{(t)}) + \nabla g_2(\boldsymbol{\nu}^{(t)})\right].\end{aligned} \tag{11}$$

Then we use subgradient descent to update $\boldsymbol{\nu}$, equivalently learning $\mathcal{V}$. With simple algebra, we can show that the mth entry for class n in $\nabla g_1(\boldsymbol{\nu}^{(t)}) + \nabla g_2(\boldsymbol{\nu}^{(t)})$ is equivalent to the mth entry in $\frac{\partial f(\mathbf{x}_i, y)}{\partial \mathbf{v}_s}\Big|_{\boldsymbol{\nu}^{(t)}} - \frac{\partial f(\mathbf{x}_i, y_i)}{\partial \mathbf{v}_s}\Big|_{\boldsymbol{\nu}^{(t)}}$. In order to guarantee the monotonic decrease of the objective in Eq. 7, we add an extra checking step in each iteration.

2.5. Cross Validation on Seen Class Data

The scoring function in Eq. 6 is obtained by solving Eq. 3 and 7, which in turn depend on parameters $\boldsymbol{\theta} = (\gamma, \lambda_1, \lambda_2, \lambda_3)$. We propose learning these parameters by means of cross validation using held-out seen class data. Specifically, define $\mathcal{S}_\ell \subset \mathcal{S}$ and the held-out set $\mathcal{S}_h = \mathcal{S}\backslash\mathcal{S}_\ell$. We learn a collection of embedding functions for source and target domains using Eq. 3 and 7 over a range of parameters $\boldsymbol{\theta}$ suitably discretized in 4D space. For each parameter choice $\boldsymbol{\theta}$ we obtain a scoring function, which depends on training subset as well as the parameter choice. We then compute the prediction error, namely, the number of times that a held-out target domain sample is misclassified for this parameter choice. We repeat this procedure for different randomly selected subsets $\mathcal{S}_\ell$ and choose parameters with the minimum average prediction error. Once these parameters are obtained we then plug it back into Eq. 3 and 7, and re-learn the scoring function using all the seen classes.

3. Experiments

We test our method on five benchmark image datasets for zero-shot recognition, *i.e.* CIFAR-10 [15], aPascal & aYahoo (aP&Y) [8], Animals with Attributes (AwA) [15], Caltech-UCSD Birds-200-2011 (CUB-200-2011) [38], and SUN Attribute [26]. For all the datasets, we utilize MatConvNet [36] with the "imagenet-vgg-verydeep-19" pretrained model [32] to extract a 4096-dim CNN feature vector (*i.e.* the top layer hidden unit activations of the network) for each image (or bounding box). Verydeep features work well since they lead to good class separation, which is required for our class dependent transform (see Fig. 5). Similar CNN features were used in previous work [2] for zero-shot learning. We denote the two variants of our general method as *SSE-INT* and *SSE-ReLU*, respectively. Note that in terms of experimental settings, the main difference between our method and the competitors is the features. We report the top-1 recognition accuracy averaged over 3 trials.

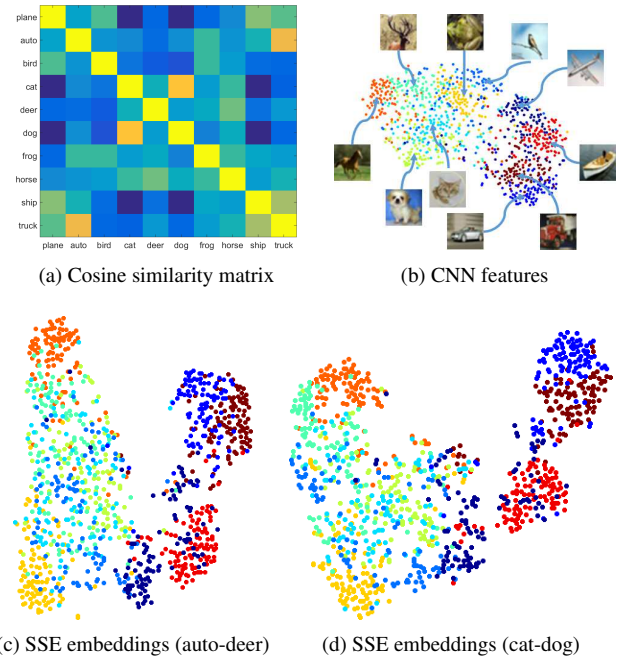
We set $\gamma, \lambda_2, \lambda_3 \in \{0, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2\}$ in Eq. 3 and 7 for cross validation. In each iteration, we randomly choose two seen classes for validation, and fix $\boldsymbol{\nu}$ in Alg. 1 to its initialization for speeding up computation. For λ_1, we simply set it to a small number 10^{-4} because it is much less important than the others for recognition.

3.1. CIFAR-10

This dataset consists of 60000 color images with resolution of 32×32 pixels (50000 for training and 10000 for testing) from 10 classes. [34] enriched it with 25 binary attributes and 50-dim semantic word vectors with real numbers for each class. We follow the settings in [34]. Precisely, we take cat-dog, plane-auto, auto-deer, deer-ship, and cat-truck as test categories for zero-shot recognition, respectively, and use the rest 8 classes as seen class data. Our training and testing is performed on the split of training and test data provided in the dataset, respectively.

We first summarize the accuracy of [34] and our method in Table 2. Clearly our method outperforms [34] significantly, and SSE-INT and SSE-ReLU perform similarly. We observe that for cat-dog our method performs similarly as [34], while for others our method can easily achieve very high accuracy. We show the class affinity matrix in Fig. 4(a) using the binary attribute vectors, and it turns out that cat and dog have a very high similarity. Similarly the word vectors between cat and dog provide more discrimination than attribute vectors but still much less than others.

To better understand our SSE learning method, we visualize the target domain CNN features as well as the learned SSE features using t-SNE [35] in Fig. 4(b-d). Due to different seen classes, the learned functions and embeddings for Fig. 4(c) and Fig. 4(d) are different. In Fig. 4(b), CNN features seem to form clusters for different classes with some overlaps, and there is a small gap between "animals" and "artifacts". In contrast, our SSE features are guided by source domain attribute vectors, and indeed preserve the affinities between classes in the attribute space. In other words, our learning algorithm manages to align the target domain distributions with their corresponding source domain embeddings in SSE space, as well as discriminating each target domain instance from wrong classes. As we see, the gaps between animals and artifacts are much clearer in Fig. 4(c) and Fig. 4(d) than that in Fig. 4(b). For cat and dog, however, there is still a large overlap in SSE space, leading to poor recognition. The overall sample distributions in Fig. 4(c) and Fig. 4(d) are similar, because they both preserve the same class affinities.



(a) Cosine similarity matrix (b) CNN features

(c) SSE embeddings (auto-deer) (d) SSE embeddings (cat-dog)

Figure 4. **(a)** Class affinities for the 10 classes using source domain binary attribute vectors. **(b-d)** t-SNE visualization of different features with 25 attributes, where 100 samples per class in the test set are selected randomly and the same color denotes the same class. (b) shows the 4096-dim original target domain CNN features. (c) and (d) show the 8-dim learned SSE features by SSE-INT and tested on auto-deer and cat-dog, respectively. The embeddings produced by SSE-ReLU have similar patterns.

3.2. Other Benchmark Comparison

For the detail of each dataset, please refer to its original paper. For aP&Y, CUB-200-2011, and SUN Attribute datasets, we take the means of attribute vectors from the same classes to generate source domain data. For AwA dataset, we utilize the real-number attribute vectors since they are more discriminative.

We utilize the same training/testing splits for zero-shot recognition on aP&Y and AwA as others. For CUB-200-2011, we follow [1] to use the same 150 bird spices as seen classes for training and the left 50 spices as unseen classes for testing. For SUN Attribute, we follow [14] to use the

Table 2. Zero-shot recognition accuracy comparison (%, mean±standard deviation) on CIFAR-10. The compared numbers are best estimated from Fig. 3 in [34]. Notice that all the methods here utilize deep features to represent images in target domain.

Method	cat-dog	plane-auto	auto-deer	deer-ship	cat-truck	Average
Socher *et al*. [34] (50 words)	50	65	76	83	90	72.8
SSE-INT (50 words)	***59.00±0.57***	91.62±0.19	97.95±0.13	95.73±0.08	97.20±0.05	***88.30***
SSE-ReLU (50 words)	58.78±1.60	91.33±0.53	97.33±0.28	95.37±0.29	***97.32±0.12***	88.03
SSE-INT (25-dim binary vectors)	48.47±0.08	***93.93±0.59***	***99.07±0.18***	***96.03±0.03***	96.92±0.14	86.88
SSE-ReLU (25-dim binary vectors)	48.52±0.13	93.68±0.73	98.48±0.15	95.32±0.25	96.43±0.06	86.49

Table 3. Zero-shot recognition accuracy comparison (%) on aP&Y, AwA, CUB-200-2011, and SUN Attribute, respectively, in the form of mean±standard deviation. Here except our results, the rest numbers are cited from their original papers. Note that some experimental settings may differ from ours.

Feature	Method	aPascal & aYahoo	Animals with Attributes	CUB-200-2011	SUN Attribute
Non-CNN	Farhadi *et al*. [8]	32.5			
	Mahajan *et al*. [18]	37.93			
	Wang and Ji [39]	45.05	42.78		
	Rohrbach *et al*. [28]		42.7		
	Yu *et al*. [42]		48.30		
	Akata *et al*. [1]		43.5	18.0	
	Fu *et al*. [10]		47.1		
	Mensink *et al*. [20]			14.4	
	Lampert *et al*. [16]	19.1	40.5		52.50
	Jayaraman and Grauman [14]	26.02±0.05	43.01±0.07		56.18±0.27
	Romera-Paredes and Torr [30]	27.27±1.62	49.30±0.21		65.75±0.51
AlexNet	Akata *et al*. [2][a]		61.9	***40.3***	
vgg-verydeep-19	Lampert *et al*. [16]	38.16	57.23		72.00
	Romera-Paredes and Torr [30]	24.22±2.89	75.32±2.28		82.10±0.32
	SSE-INT	44.15±0.34	71.52±0.79	30.19±0.59	82.17±0.76
	SSE-ReLU	***46.23±0.53***	***76.33±0.83***	30.41±0.20	***82.50±1.32***

[a]The results listed here are the ones with 4096-dim CNN features and the continuous attribute vectors provided in the datasets for fair comparison.

(a) decaf (b) verydeep-19

Figure 5. t-SNE visualization comparison of SSE distributions using the two CNN features on AwA testing data. Our method works well if there is good separation for classes and verydeep features are particularly useful.

same 10 classes as unseen classes for testing (see their supplementary file) and take the rest as seen classes for training.

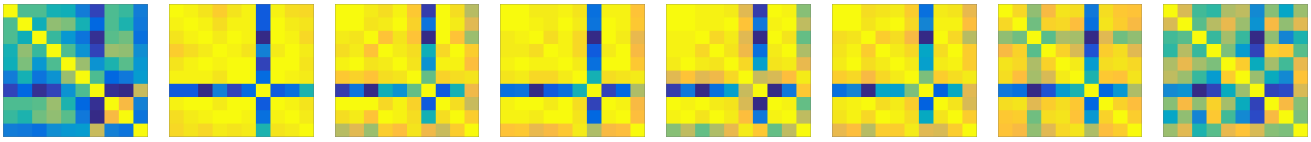
We summarize our comparison in Table 3, where the blank spaces indicate that the proposed methods were not tested on the datasets in their original papers. Still there is no big performance difference between our SSE-INT and SSE-ReLU. On 4 out of the 5 datasets, our method works best except for CUB-200-2011. On one hand, [2] specifically targets at fine-grained zero-shot recognition such as this dataset, while ours aims for general zero-shot learning. On the other hand, we suspect that the source domain projection function may not work well in fine-grained recognition, and we will investigate more on it in our future work.

To understand our method better with different features, we test 7 features on AwA dataset[1]. We show the SSE distribution comparison using decaf CNN features and vgg-verydeep-19 CNN features in Fig. 5. There is a large difference between the two distributions: (a) while with decaf features clusters are slightly separated they are still cluttered with overlaps among different classes. (b) vgg-verydeep-19 features, in contrast, form crisp clusters for different classes, which is useful for zero-shot recognition. Also we plot the cosine similarity matrices created using different features in Fig. 6. As we see, the matrix from vgg-verydeep-19 features (*i.e*. the last) is the most similar to that from the source domain attribute vectors (*i.e*. the first). This demonstrates that our learning method with vgg-verydeep-19 features can align the target domain distribution with the source domain attribute vectors. We can attribute this to the fact that we need a class dependent feature transform $\phi_y(\mathbf{x})$ that has good separation on seen classes.

Our implementation[2] is based on unoptimized MATLAB code. However, it can return the prediction results on any of these 5 datasets within 30 minutes using a multi-thread CPU (Xeon E5-2696 v2), starting from loading CNN features. For instance, on CIFAR-10 we manage to finish running the code less than 5 minutes.

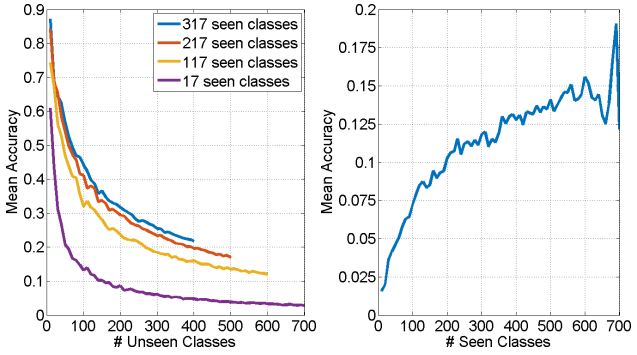
[1]We downloaded these features from http://attributes.kyb.tuebingen.mpg.de/

[2]Our code is available at https://zimingzhang.wordpress.com/source-code/.



(a) attributes (b) cq-hist (31.5) (c) lss-hist (30.3) (d) rgsift-hist (33.6) (e) sift-hist (29.8) (f) surf-hist (36.5) (g) decaf (52.0) (h) verydeep-19 (71.5)

Figure 6. Cosine similarity matrices created using different features on AwA testing data. The numbers in the brackets are the mean accuray (%) achieved using the corresponding features. Our learning method performs the best with vgg-verydeep-19 features. We can attribute this to the fact that we need a class dependent feature transform $\phi_y(\mathbf{x})$ that has good separation on seen classes.



(a) Recognition on unseen classes (b) Recognition on all classes

Figure 7. Large-scale zero-shot recognition on SUN Attribute.

3.3. Towards Large-Scale Zero-Shot Recognition

We test the generalization ability of our method on the SUN Attribute dataset for large-scale zero-shot recognition. We design two experimental settings: (1) Like in benchmark comparison, we randomly select M classes as seen classes for training, and then among the rest $717 - M$ classes, we also randomly select $10, 20, \cdots, 717 - M$ classes as unseen classes for testing; (2) We randomly select $10, 20, \cdots, 700$ classes as seen classes for training, and categorize each data sample from the rest unseen classes into one of the 717 classes. Fig. 7 shows our results, where (a) and (b) correspond to the settings (1) and (2), respectively.

In Fig. 7(a), we can see that with very few seen classes, we can achieve reasonably good performance when unseen classes are a few. However, with the increase of the number of unseen classes, the curve drops rapidly and then changes slowly when the number is large. From 200 to 700 unseen classes, our performance is reduced from 8.62% to 2.85%. With the increase of the number of seen classes, our performance is improving, especially when the number of unseen classes is small. With 10 unseen classes, our performance increases from 61.00% to 87.17% using 17 and 317 seen classes, respectively. But such improvement is marginal when there are already a sufficient number of seen classes, for instance from 217 to 317 seen classes.

In Fig. 7(b), generally speaking, with more seen classes our performance will be better, because there will be better chance to preserve the semantic affinities among classes in source domain. With only 10 seen classes, our method can achieve 1.59% mean accuracy, which is much better than the random chance 0.14%. Notice that even though we use all the 717 classes as seen classes, we cannot guarantee that the testing results are similar to those of traditional classification methods, because the source domain attribute vectors will guide our method for learning. If they are less discriminative, *e.g.* the attribute vectors for cat and dog in CIFAR-10, the recognition performance may be worse.

To summarize, our method performs well and stably on SUN Attribute with a small set of seen classes and a relatively large set of unseen classes. Therefore, we believe that our method is suitable for large-scale zero-shot recognition.

4. Conclusion

We proposed learning a semantic similarity embedding (SSE) method for zero-shot recognition. We label the semantic meanings using seen classes, and project all the source domain attribute vectors onto the simplex in SSE space, so that each class can be represented as a probabilistic mixture of seen classes. Then we learn similarity functions to embed target domain data into the same semantic space as source domain, so that not only the empirical mean embeddings of the seen class data distributions are aligned with their corresponding source domain embeddings, but also the data instance itself can be classified correctly. We propose learning two variants using intersection function and rectified linear unit (ReLU). Our method on five benchmark datasets including the large-scale SUN Attribute dataset significantly outperforms other state-of-art methods. As future work, we would like to explore other applications for our method such as person re-identification [44, 45, 46] and zero-shot activity retrieval [6].

Acknowledgement

We thank the anonymous reviewers for their very useful comments. This material is based upon work supported in part by the U.S. Department of Homeland Security, Science and Technology Directorate, Office of University Programs, under Grant Award 2013-ST-061-ED0001, by ONR Grant 50202168 and US AF contract FA8650-14-C-1728. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the social policies, either expressed or implied, of the U.S. DHS, ONR or AF.

References

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for attribute-based classification. In *CVPR*, pages 819–826, 2013. 2, 4, 6, 7

[2] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele. Evaluation of output embeddings for fine-grained image classification. In *CVPR*, June 2015. 2, 3, 4, 6, 7

[3] S. Bengio, J. Weston, and D. Grangier. Label embedding trees for large multi-class tasks. In *NIPS*, pages 163–171, 2010. 2

[4] T. L. Berg, A. C. Berg, and J. Shih. Automatic attribute discovery and characterization from noisy web data. In *ECCV*, pages 663–676, 2010. 1

[5] G. Blanchard and C. Scott. Decontamination of mutually contaminated models. In *AISTATS*, 2014. 3

[6] G. D. Castanon, Y. Chen, Z. Zhang, and V. Saligrama. Efficient activity retrieval through semantic graph queries. In *ACM Multimedia*, 2015. 8

[7] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *PAMI*, 35(11):2765–2781, 2013. 4

[8] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, pages 1778–1785, 2009. 1, 4, 6, 7

[9] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. A. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In *NIPS*, pages 2121–2129, 2013. 1, 2

[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. 2, 7

[11] J. Hamm and M. Belkin. Probabilistic zero-shot classification with semantic rankings. *CoRR*, abs/1502.08039, 2015. 2

[12] S. J. Hwang, K. Grauman, and F. Sha. Analogy-preserving semantic embedding for visual object categorization. In *ICML*, pages 639–647, 2013. 2

[13] S. J. Hwang and L. Sigal. A unified semantic embedding: Relating taxonomies and attributes. In *NIPS*, pages 271–279, 2014. 2

[14] D. Jayaraman and K. Grauman. Zero-shot recognition with unreliable attributes. In *NIPS*, pages 3464–3472, 2014. 2, 6, 7

[15] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Master's thesis, 2009. 6

[16] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *PAMI*, 36(3):453–465, 2014. 1, 2, 7

[17] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *PAMI*, 35(1):171–184, 2013. 4

[18] D. Mahajan, S. Sellamanickam, and V. Nair. A joint learning framework for attribute models and object descriptions. In *ICCV*, pages 1227–1234, 2011. 2, 7

[19] S. Maji, A. C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *CVPR*, pages 1–8, 2008. 4

[20] T. Mensink, E. Gavves, and C. G. M. Snoek. Costa: Co-occurrence statistics for zero-shot classification. In *CVPR*, pages 2441–2448, June 2014. 2, 7

[21] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In *ECCV*, pages 488–501. 2012. 1

[22] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, pages 807–814, 2010. 2, 4

[23] 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

[24] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell. Zero-shot learning with semantic output codes. In *NIPS*, pages 1410–1418, 2009. 2

[25] D. Parikh and K. Grauman. Interactively building a discriminative vocabulary of nameable attributes. In *CVPR*, pages 1681–1688, 2011. 1

[26] 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. 6

[27] X. Peng, L. Zhang, and Z. Yi. Constructing l2-graph for subspace learning and segmentation. *arXiv preprint arXiv:1209.0841*, 2012. 4

[28] M. Rohrbach, S. Ebert, and B. Schiele. Transfer learning in a transductive setting. In *NIPS*, pages 46–54, 2013. 2, 7

[29] M. Rohrbach, M. Stark, and B. Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *CVPR*, pages 1641–1648, 2011. 1

[30] B. Romera-Paredes and P. H. S. Torr. An embarrassingly simple approach to zero-shot learning. In *ICML*, 2015. 2, 7

[31] 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, 2014. 1

[32] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6

[33] A. Smola, A. Gretton, L. Song, and B. Schölkopf. A hilbert space embedding for distributions. In *Algorithmic Learning Theory*, pages 13–31, 2007. 3

[34] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng. Zero-shot learning through cross-modal transfer. In *NIPS*, pages 935–943, 2013. 1, 2, 6, 7

[35] L. Van der Maaten and G. Hinton. Visualizing data using t-SNE. *JMLR*, 9(2579-2605):85, 2008. 6

[36] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for MATLAB. *CoRR*, abs/1412.4564, 2014. 6

[37] R. Vidal. A tutorial on subspace clustering. *Signal Processing Magazine*, pages 52–68, 2010. 1

[38] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011. 6

[39] X. Wang and Q. Ji. A unified probabilistic approach modeling relationships between attributes and objects. In *ICCV*, pages 2120–2127, 2013. 2, 7

[40] K. Weinberger and O. Chapelle. Large margin taxonomy embedding for document categorization. In *NIPS*, pages 1737–1744. 2009. 2

[41] S. Wu, S. Bondugula, F. Luisier, X. Zhuang, and P. Natarajan. Zero-shot event detection using multi-modal fusion of weakly supervised concepts. In *CVPR*, pages 2665–2672, 2014. 1

[42] 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*, pages 771–778, 2013. 1, 2, 7

[43] A. L. Yuille and A. Rangarajan. The concave-convex procedure. *Neural computation*, 15(4):915–936, 2003. 5

[44] Z. Zhang, Y. Chen, and V. Saligrama. A novel visual word co-occurrence model for person re-identification. In *ECCV Workshop on Visual Surveillance and Re-Identification*, 2014. 8

[45] Z. Zhang, Y. Chen, and V. Saligrama. Group membership prediction. In *ICCV*, 2015. 8

[46] Z. Zhang and V. Saligrama. PRISM: Person re-identification via structured matching. *arXiv preprint arXiv:1406.4444*, 2014. 8