# Zero-Shot Learning via Joint Latent Similarity Embedding

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

## Abstract

*Zero-shot recognition (ZSR) deals with the problem of predicting class labels for target domain instances based on source domain side information* (e.g. *attributes*) *of unseen classes. We formulate ZSR as a binary prediction problem. Our resulting classifier is class-independent. It takes an arbitrary pair of source and target domain instances as input and predicts whether or not they come from the same class,* i.e. *whether there is a match. We model the posterior probability of a match since it is a sufficient statistic and propose a latent probabilistic model in this context. We develop a joint discriminative learning framework based on dictionary learning to jointly learn the parameters of our model for both domains, which ultimately leads to our class-independent classifier. Many of the existing embedding methods can be viewed as special cases of our probabilistic model. On ZSR our method shows 4.90% improvement over the state-of-the-art in accuracy averaged across four benchmark datasets. We also adapt ZSR method for zero-shot retrieval and show 22.45% improvement accordingly in mean average precision (mAP).*

## 1. Introduction

Zero-shot learning (ZSL) deals with the problem of learning to classify previously unseen class instances. It is particularly useful in large scale classification where labels for many instances or entire categories can often be missing. One popular version of ZSL is based on the so-called source and target domains. In this paper we consider the source domain as a collection of class-level vectors, where each vector describes side information of one *single* class with, for instance, attributes [10, 19, 24, 27, 31], language words/phrases [4, 11, 34], or even learned classifiers [40]. The target domain is described by a distribution of instances (*e.g.* images, videos, *etc.*) [19, 38]. During training, we are given source domain side information and target domain data corresponding to only a subset of classes, which we call *seen* classes. During test time for the source domain, side information is then provided for *unseen* classes. A target domain instance from an unknown *unseen* class is then presented. The goal during test time is to predict the class label for the unseen target domain instance.

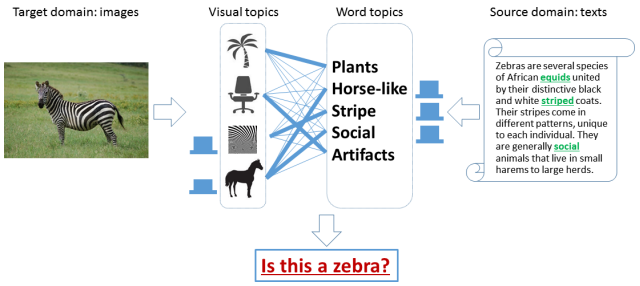



Figure 1. Illustration of our joint latent space model with images as target domain and text-documents as source domain. The bar graph next to the (latent) topics indicate the mixture weights of the topics. The links between the topics indicate the co-occurrence (thicker lines depicting larger likelihood values). Our method is based on learning a class-independent similarity function using seen class training data, which measures the likelihood of a source domain class vector and a target domain data sample being the same class, regardless of their true underlying classes.

**Intuition:** In contrast to previous methods (*e.g.* [2]) which explicitly learn the relationships between source and target domain data, we posit that for both domains there exist corresponding latent spaces, as illustrated in Fig. 1, where there is a similarity function *independent* of class labels.

Our supposition implies that, regardless of the underlying class labels, there is a statistical relationship between latent co-occurrence patterns of corresponding source and target instance pairs when the instance pairs describe the same thing. For example, with our supposition the "zebra" image in Fig. 1 on the left will share an underlying statistical relationship with the description of zebra in text on the right, and that this relationship can be inferred by means of a class-independent "universal" similarity function[1].

To mathematically formalize this intuition we formulate zero-shot recognition (ZSR) as a binary classification problem. In this framework, we train a score function that takes an arbitrary source-target instance pair as input and outputs a likelihood score that the paired source and target instances

[1]Intuitively this is a plausible mechanism. We as humans tend to draw connections from different sources to improve our understanding of objects/concepts.

come from the same class. We apply this score function on a given target instance to identify a corresponding source vector with the largest score. In this way our score function generalizes to unseen classes since it does not explicitly depend on the actual class label.

We train our binary predictor (*i.e*. score function) using seen class source and target domain data. It is well-known that for a binary classification problem the posterior probability of the binary output conditioned on data is a sufficient statistic for optimal detection. This motivates us to propose a latent parametrized probabilistic model for the posterior. We decompose the posterior into source/target domain data likelihood terms and a cross-domain latent similarity function. We develop a joint discriminative learning framework based on dictionary learning to jointly learn the parameters of the likelihood and latent similarity functions.

In test-time unseen source domain vectors are revealed. We estimate their corresponding latent source embeddings. Then, for an arbitrary target-instance, we estimate the latent target embedding. Finally we score each pair of source and target domain embeddings using our similarity function and classify based on these scores. Fig. 1 illustrates a specific scenario where visual and word embedding functions are learned using training data from seen classes and are utilized to estimate embeddings for unseen data. We test our method on four challenging benchmark datasets (*i.e*. aP&Y, AwA, CUB, SUN-attribute). Our performance on average shows 4.9% improvement in recognition accuracy. We also adapt ZSR method for zero-shot retrieval and show 22.45% improvement in mean average precision across these datasets.

Our proposed general probabilistic model is a systematic framework for ZSR. Indeed, existing methods including [1, 2, 11, 14, 23, 25] can be precisely interpreted as special cases of our method. We test our algorithm on several ZSL benchmark datasets and achieve state-of-the-art results.

### 1.1. Related Work

**(i) Attribute prediction:** A significant fraction of zero-shot methods are based on building attribute classifiers that transfer target domain data into source domain attribute space. For instance, [26] used semantic knowledge bases to learn the attribute classifiers. [19, 22, 37, 40, 41] proposed several (probabilistic or discriminative) attribute prediction methods using the information from attributes, classes, and objects. [23] proposed combining seen class classifiers linearly to build unseen class classifiers. [14] proposed first linearly projecting both source and target domain data into a common space and then training a max-margin multi-label classifiers for prediction. [32] proposed a related regularization based method for training classifiers. The main issue in such methods is that they may suffer from noisy source/target data, which often results in poor prediction. In contrast, our joint latent space model is robust to the noise issues on account of the nature of latent space learning.

**(ii) Linear embedding:** This type of methods are based on embedding both source and target domain data into a feature space characterized by the Kronecker product of source domain attributes and target domain features. Linear classifiers are trained in the product space. For instance, [1] created such spaces using label embedding, and [2, 11, 25, 34] utilized deep learning for the same purpose. Recently [20, 21] introduced semi-supervised max-margin learning to learn the label embedding.

**(iii) Nonlinear embedding:** Similar to linear embedding, here the Kronecker product feature space is constructed after a nonlinear mapping of the original features. This literature includes [3, 16, 45], where [16, 45] embed source and target domain data nonlinearly into known semantic spaces (*i.e*. seen classes) in an unsupervised or supervised way, and [3] employed deep neural networks for associating the resulting embeddings.

Different from these (linear or nonlinear) embedding based zero-shot methods, our method learns a *joint latent space* for both domains using *structured learning*. The learned joint space is used not only to fit each instance well (by dictionary learning) but also to enable recognition (by bilinear classifiers) during test time.

**(iv) Other methods:** Less related to our method includes approaches based on semantic transfer propagation [30], transductive multi-view embedding [12], random forest approach [15], and semantic manifold distance [13].

## 2. Our Method

### 2.1. Problem Setting

Let us motivate our approach from a probabilistic modelling perspective. This will in turn provide a basis for structuring our discriminative learning method. We denote by $\mathcal{X}^{(s)}$ the space of source domain vectors, by $\mathcal{X}^{(t)}$ the space of target domain vectors, and by $\mathcal{Y}$ the collection of all classes. Following convention, the random variables are denoted by capital letters, namely, $X^{(s)}$, $X^{(t)}, Y$ and instances of them by lower-case letters $\mathbf{x}^{(s)}, \mathbf{x}^{(t)}, \mathbf{y}$.

Zero-shot learning is a special case where the class corresponding to the source domain instance is revealed during test time and thus there is no uncertainty regarding the class label for any source domain vector. Thus the problem reduces to assigning target domain instances to source domain vectors (and in turn to classes) during testing. For exposition we denote by $y^{(s)}$ the label for the source domain instance $\mathbf{x}^{(s)} \in \mathcal{X}^{(s)}$ even though we know that $y^{(s)}$ is identical to the true class label $y$. With this in mind, we predict a class label $y^{(t)}$ for target domain instance $\mathbf{x}^{(t)} \in \mathcal{X}^{(t)}$.

### 2.2. General Probabilistic Modelling

Abstractly, we can view ZSR as a problem of assigning a *binary* label to a pair of source and target domain instances, namely whether or not $y^{(st)} \triangleq [y^{(s)} = y^{(t)}]$ holds.

We view our goal in terms of evaluating how likely this proposal is true, *i.e.* $p(y^{(st)}|\mathbf{x}^{(s)}, \mathbf{x}^{(t)})$. Indeed, Bayes Optimal Risk theory tells us that the optimal classifier (see Eq. 6 in [9]), $F(\mathbf{x}^{(s)}, \mathbf{x}^{(t)})$, is obtained by suitably thresholding the posterior of $y^{(st)}$ conditioned on data, namely,

$$F(\mathbf{x}^{(s)}, \mathbf{x}^{(t)}) \triangleq \log p(y^{(st)}|\mathbf{x}^{(s)}, \mathbf{x}^{(t)}) \underset{\text{Diff}}{\overset{\text{Ident}}{\gtrless}} \theta \quad (1)$$

where $\theta \in \mathbb{R}$ is a threshold parameter. Here *Ident* is the hypothesis that source/target data describe the same class. *Diff* is the hypothesis that they are different.

Our latent embedding model supposes that the observed and latent random variables form a Markov chain [6]:

$$X^{(s)} \leftrightarrow Z^{(s)} \leftrightarrow Y \leftrightarrow Z^{(t)} \leftrightarrow X^{(t)}. \quad (2)$$

This implies that the source domain data, $X^{(s)}$, and its associated embedding, $Z^{(s)}$ is independent of the target $X^{(t)}, Z^{(t)}$ conditioned on the underlying class $Y$ (if they belong to the same class) and unconditionally independent if they belong to different classes.

It follows that the posterior probability can be factored as $p(y^{(st)}, \mathbf{z}^{(s)}, \mathbf{z}^{(t)}|\mathbf{x}^{(s)}, \mathbf{x}^{(t)}) = p(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)})p(\mathbf{z}^{(s)}, \mathbf{z}^{(t)}|\mathbf{x}^{(s)}, \mathbf{x}^{(t)})$. Next note that, in the absence of class information, it is reasonable to assume that an arbitrary pair of source and target domain latent embeddings are essentially independent, namely, $p(\mathbf{z}^{(s)}, \mathbf{z}^{(t)}) \approx p(\mathbf{z}^{(s)})p(\mathbf{z}^{(t)})$. Consequently, the posterior probability can be expressed as follows:

$$p(y^{(st)}|\mathbf{x}^{(s)}, \mathbf{x}^{(t)}) = \sum_{\mathbf{z}^{(s)}, \mathbf{z}^{(t)}} p(\mathbf{z}^{(s)}|\mathbf{x}^{(s)})p(\mathbf{z}^{(t)}|\mathbf{x}^{(t)})p(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)}), \quad (3)$$

where, $\mathbf{z}^{(s)} \in \mathbb{R}^{h_s}$ and $\mathbf{z}^{(t)} \in \mathbb{R}^{h_t}$ denote the latent coefficient vectors in the corresponding $h_s$-dim and $h_t$-dim latent spaces, respectively. Here $(\mathbf{z}^{(s)}, \mathbf{z}^{(t)})$ defines the *joint latent embedding* for data pair $(\mathbf{x}^{(s)}, \mathbf{x}^{(t)})$. This factorization provides us two important insights:

*(i) Class-independent Embeddings:* Note that the expression in Eq. 3 informs us that the probability kernels $p(\mathbf{z}^{(s)}|\mathbf{x}^{(s)})$, $p(\mathbf{z}^{(t)}|\mathbf{x}^{(t)})$ characterizing the latent embeddings depend only on the corresponding data instances, $\mathbf{x}^{(s)}$, $\mathbf{x}^{(t)}$ and independent of the underlying class labels.

*(ii) Class-independent Similarity Kernel:* The expression in Eq. 3 reveals that the term $p(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)})$ is a class-invariant function that takes arbitrary source and target domain embeddings as input and outputs a likelihood of similarity regardless of underlying class labels (recall that predicting $y^{(st)} \triangleq [y^{(s)} = y^{(t)}]$ is binary). Consequently, at a conceptual level, our framework provides a way to assign similarities of class membership between arbitrary target domain vectors and source domain vectors while circumventing the intermediate step of assigning class labels.

In our context the joint probability distributions and latent conditionals are unknown and must be estimated from data. Nevertheless, this perspective provides us with a structured way to estimate them from data. An important issue is that Eq. 3 requires integration over the latent spaces, which is computationally cumbersome during both training and testing. To overcome this issue we lower bound Eq. 3 by a straightforward application of Jensen's inequality:

$$\log p(y^{(st)}|\mathbf{x}^{(s)}, \mathbf{x}^{(t)}) \geq \max_{\mathbf{z}^{(s)}, \mathbf{z}^{(t)}} \log p(\mathbf{z}^{(s)}|\mathbf{x}^{(s)})p(\mathbf{z}^{(t)}|\mathbf{x}^{(t)})p(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)}). \quad (4)$$

In training and testing below, we employ this lower bound (*i.e.* the right hand-side (RHS) in Eq. 4) as a surrogate for the exact but cumbersome similarity function between source and target domains. That is,

$$F(\mathbf{x}^{(s)}, \mathbf{x}^{(t)}, y^{(st)}) \triangleq \max_{\mathbf{z}^{(s)}, \mathbf{z}^{(t)}} \Big\{ \log p(\mathbf{z}^{(s)}|\mathbf{x}^{(s)}) + \log p(\mathbf{z}^{(t)}|\mathbf{x}^{(t)}) + \log p(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)}) \Big\}. \quad (5)$$

Note that here $\log p(\mathbf{z}^{(s)}|\mathbf{x}^{(s)}), \log p(\mathbf{z}^{(t)}|\mathbf{x}^{(t)})$ are actually *data fitting* terms to restrict the feasible parameter spaces for $\mathbf{z}^{(s)}, \mathbf{z}^{(t)}$, respectively, performing the same functionality of regularization from the perspective of optimization. $\log p(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)})$ is essentially the *latent similarity measure* term in the joint latent space with embeddings. In the following section we show how many of the existing works in the literature can be viewed as special cases of our probabilistic framework.

#### 2.2.1 Relationship to Existing Works

Our probabilistic model can be considered as generalization of many embedding methods for ZSL. In particular, we will show that label embedding [1], output embedding [2], semantic similarity embedding [45], deep neural network based embedding [3], and latent embedding [39] can all be viewed as special cases. For concreteness, we follow the notation in the original papers of each work and show how to view them as special cases of our model.

*(i) Label embedding [1].* This approach defines a bilinear prediction function as follows:

$$f(x; \mathbf{W}) = \arg\max_{y \in \mathcal{Y}} F(x, y; \mathbf{W}) = \arg\max_{y \in \mathcal{Y}} \theta(x)^T \mathbf{W} \varphi(y), \quad (6)$$

where $F$ denotes the bilinear similarity function, $\theta(x), \varphi(y)$ denote the original image embedding and label embedding for image $x$ and label $y$, respectively. The matrix $\mathbf{W}$ is the parameter describing the bilinear classifier. In this work label embeddings are viewed as side information, for instance as attribute vectors.

We can view [1] as a special case of our general probabilistic model as follows. Define $\mathbf{x}^{(s)} \triangleq y, \mathbf{x}^{(t)} \triangleq x$. The three log-likelihoods in Eq. 5 are described as follows:

$$\log p_B(\mathbf{z}^{(s)}|\mathbf{x}^{(s)}) = \begin{cases} 0, & \text{if } \mathbf{z}^{(s)} = \varphi(y) \\ -\infty, & \text{otherwise} \end{cases} \quad (7)$$

$$\log p_D(\mathbf{z}^{(t)}|\mathbf{x}^{(t)}) = \begin{cases} 0, & \text{if } \mathbf{z}^{(t)} = \theta(x) \\ -\infty, & \text{otherwise} \end{cases} \quad (8)$$

$$\log p_W(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)}) \triangleq F(x, y; \mathbf{W}). \quad (9)$$

It can directly be verified by direct substitution that this is identical to the model described in [1].

*(ii) Output embedding [2].* The similarity function proposed here is:

$$F(x, y; \{\mathbf{W}\}_{1,\cdots,K}) = \sum_k \alpha_k \theta(x)^T \mathbf{W}_k \varphi_k(y), \quad (10)$$
$$\text{s.t.} \sum_k \alpha_k = 1,$$

where $\{\mathbf{W}\}_{1,\cdots,K}$ denotes the parameters for $K$ different bilinear functions, $\varphi_k(y)$ denotes the $k$-th type of label embedding, and $\alpha_k$ denotes the combination weight for the $k$-th bilinear function. Then Eq. 6 with the above similarity function is utilized as the prediction function.

To view [2] as a special case of our general probabilistic model, we can parametrize our model in the same way as we did for [1], except that

$$\log p_B(\mathbf{z}^{(s)}|\mathbf{x}^{(s)}) = \sum_k \log p_B(\mathbf{z}_k^{(s)}|\varphi_k(y))$$
$$= \begin{cases} -\log K, & \text{if } \mathbf{z}_k^{(s)} = \varphi_k(y), \forall k, \\ -\infty, & \text{otherwise} \end{cases} \quad (11)$$

$$\log p_W(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)}) \triangleq F(x, y; \{\mathbf{W}\}_{1,\cdots,K}). \quad (12)$$

It can directly be verified by direct substitution that this is identical to the model described in [2].

*(iii) Semantic similarity embedding [45].* Given a label embedding $\mathbf{c}$, [45] solves the following sparse coding problem to compute the semantic similarity embedding (SSE) for source domain:

$$\psi(\mathbf{c}) = \arg\min_{\boldsymbol{\alpha} \in \Delta^{|\mathcal{S}|}} \left\{ \frac{\gamma}{2}\|\boldsymbol{\alpha}\|^2 + \frac{1}{2}\|\mathbf{c} - \sum_{y \in \mathcal{S}} \mathbf{c}_y \alpha_y\|^2 \right\}, \quad (13)$$

where $\gamma \geq 0$ is a predefined regularization parameter, $\|\cdot\|$ denotes the $\ell_2$ norm of a vector, and $\boldsymbol{\alpha} = (\alpha_y)_{y \in \mathcal{S}}$ describes contributions of different seen classes. Given a target-domain image embedding $\mathbf{x}$, the corresponding SSE is defined as

$$\phi_y(\mathbf{x}) = \min(\mathbf{x}, \mathbf{v}_y), \text{ or } \phi_y(\mathbf{x}) = \max(\mathbf{0}, \mathbf{x} - \mathbf{v}_y), \quad (14)$$

where $\mathbf{v}_y$ denotes a parameter for class $y$ that needs to be learned. Then the similarity function in [45] is defined as

$$F(\mathbf{x}, y; \mathbf{w}) = \sum_{s \in \mathcal{S}} \langle \mathbf{w}, \phi_s(\mathbf{x}) \rangle z_{y,s}, \quad (15)$$

where $\mathcal{S}$ denotes the set of seen classes, $z_{y,s}$ denotes the $s$-th entry in the SSE for class $y$, and $\mathbf{w}$ denotes the classifier parameter. Then Eq. 6 with the above similarity function is utilized as the prediction function.

To view [45] as a special case of our general probabilistic model, we can use the same methodology to model the three log-likelihoods in Eq. 5 as follows:

$$\log p_B(\mathbf{z}^{(s)}|\mathbf{x}^{(s)}) = \begin{cases} 0, & \text{if } \mathbf{z}^{(s)} = \psi(\mathbf{x}^{(s)}) \\ -\infty, & \text{otherwise} \end{cases} \quad (16)$$

$$\log p_D(\mathbf{z}^{(t)}|\mathbf{x}^{(t)}) = \begin{cases} 0, & \text{if } \mathbf{z}^{(t)} = \phi(\mathbf{x}^{(t)}) \\ -\infty, & \text{otherwise} \end{cases} \quad (17)$$

$$\log p_W(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)}) \triangleq F(\mathbf{x}, y; \mathbf{w}). \quad (18)$$

*(iv) Deep neural network based embedding [3].* The prediction function in [3] is the same as Eq. 6, except that now functions $\varphi, \theta$ are learned using neural networks, and the learned $\mathbf{W}$ represents the weight for a fully-connected layer between the two embeddings from source and target domains, respectively.. Therefore, in test time we can use the same parametrization for our model so that [3] can be taken as our special case mathematically.

*(v) Latent embedding [39].* This approach learns the latent embedding spaces explicitly based on clustering. For each cluster a bilinear classifier is learned for measuring similarities. Correspondingly the similarity decision function in [39] is defined as follows:

$$F(\mathbf{x}, \mathbf{y}; \{\mathbf{W}\}_{1,\cdots,K}) = \max_{1 \leq i \leq K} \mathbf{x}^T \mathbf{W}_i \mathbf{y}, \quad (19)$$

where $\mathbf{x}, \mathbf{y}$ denote image and label embeddings, respectively, and $i$ denotes the $i$-th bilinear classifier with parameter $\mathbf{W}_i$ among the $K$ classifiers. Because of the max operator, the indicator variable $i$ becomes the latent variable for selecting which bilinear classifier should be utilized per data pair.

To view [39] as a special case of our general probabilistic model, we first construct a new $\mathbf{W}$ in Eq. 6 by putting $\mathbf{W}_i, \forall i$ as *blocks* along the diagonal, *i.e.* $\mathbf{W} \triangleq diag(\mathbf{W}_1, \cdots, \mathbf{W}_K) \in \mathbb{R}^{Kd_t \times Kd_s}$, where $d_t, d_s$ denote

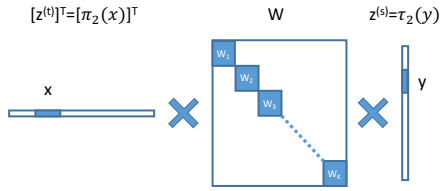



Figure 2. Illustration of our generalization for latent embedding [39]. This figure illustrates the similarity measure of $\mathbf{x}^T\mathbf{W}_2\mathbf{y}$. By searching for the maximum over different combinations of $\mathbf{z}^{(t)}, \mathbf{z}^{(s)}$, our model can exactly represent the prediction function in Eq. 19.

the dimensions of $\mathbf{x}, \mathbf{y}$ in Eq. 19, respectively, and filling in the rest entries with zeros. Here, along either columns or rows in $\mathbf{W}$ there exist $K$ blocks with dimensionality of either $d_t$ or $d_s$ per block. Then we design two functions $\pi : \mathbb{R}^{d_t} \to \mathbb{R}^{Kd_t}, \tau : \mathbb{R}^{d_s} \to \mathbb{R}^{Kd_s}$ to map the original data $\mathbf{x}, \mathbf{y}$ to higher dimensional spaces with $K$ blocks, respectively. The functionality of $\pi, \tau$ is to assign $\mathbf{x}, \mathbf{y}$ to one block $i, j \in [K]$, denoted by $\pi_i(\mathbf{x}), \tau_j(\mathbf{y})$, and fill in the rest entries using zeros. The whole construction procedure is illustrated in Fig. 2. Now we can use the same methodology to model the three log-likelihoods in Eq. 5 as follows:

$$\log p_B(\mathbf{z}^{(s)}|\mathbf{x}^{(s)}) = \begin{cases} -\log K, & \text{if } \mathbf{z}^{(s)} = \tau_j(\mathbf{y}), \forall j \\ -\infty, & \text{otherwise} \end{cases} \tag{20}$$

$$\log p_D(\mathbf{z}^{(t)}|\mathbf{x}^{(t)}) = \begin{cases} -\log K, & \text{if } \mathbf{z}^{(t)} = \pi_i(\mathbf{x}), \forall i \\ -\infty, & \text{otherwise} \end{cases} \tag{21}$$

$$\log p_W(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)}) \triangleq \left[\mathbf{z}^{(t)}\right]^T \mathbf{W}\mathbf{z}^{(s)} + \Delta(i,j), \tag{22}$$

where $\Delta(i,j) = 0$ if $i = j$, otherwise $-\infty$, which enforces $\pi, \tau$ to select the same block for similarity measure.

In the light of these observations we can view our framework as a way to describe different modes of data in a unified semantic space. Central to this observation is the key insight that zero-shot learning is fundamentally about detecting whether or not an arbitrary tuple $(\mathbf{x}^{(s)}, \mathbf{x}^{(t)})$ is associated with the same underlying label or not. This question is then fundamentally about binary classification. A second aspect of our framework is the latent embedding. This latent embedding describes source and target domain realizations as being conditionally independent of each other given their latent embeddings. For instance, this enforces the natural assumption that an image is conditionally independent of its textual description if it is conditioned on visual attributes, which serve as the latent embedding. In this way latent embeddings serve as sufficient statistics for identifying similarity of the tuple. This perspective in turn serves to unify many of the existing works in the literature. Nevertheless, for the concreteness we must choose specific

**Algorithm 1** Jointly latent embedding learning algorithm for solving Eq. 23

**Input** : training data $\{(\mathbf{x}_i^{(s)}, y_i^{(s)})\}$ and $\{(\mathbf{x}_j^{(t)}, y_j^{(t)})\}$
**Output**: $B, D, W$

Initialize $B, D$;
$\forall i, \mathbf{z}_i^{(s)} \leftarrow \arg\max_{\mathbf{z}^{(s)}} \log p_B(\mathbf{z}^{(s)}|\mathbf{x}_i^{(s)})$;
$\forall j, \mathbf{z}_j^{(t)} \leftarrow \arg\max_{\mathbf{z}^{(t)}} \log p_D(\mathbf{z}^{(t)}|\mathbf{x}_j^{(t)})$;
$W \leftarrow \arg\max_W \sum_{i=1}^C \sum_{j=1}^N \log p_W(y_{ij}^{(st)}|\mathbf{z}_i^{(s)}, \mathbf{z}_j^{(t)})$;
**repeat**
  **foreach** $i$ **do**
    **foreach** $j$ **do**
      $\mathbf{z}_{ij}^{(s)} \leftarrow \mathbf{z}_i^{(s)}$; $\mathbf{z}_{ij}^{(t)} \leftarrow \mathbf{z}_j^{(t)}$;
      **repeat**
        $\mathbf{z}_{ij}^{(s)} \leftarrow \arg\max_{\mathbf{z}^{(s)}} \log p_B(\mathbf{z}^{(s)}|\mathbf{x}_i^{(s)}) + \log p_W(y_{ij}^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}_{ij}^{(t)})$;
        $\mathbf{z}_{ij}^{(t)} \leftarrow \arg\max_{\mathbf{z}^{(t)}} \log p_D(\mathbf{z}^{(t)}|\mathbf{x}_j^{(t)}) + \log p_W(y_{ij}^{(st)}|\mathbf{z}_{ij}^{(s)}, \mathbf{z}^{(t)})$;
      **until** *Converge to a local maximum*;
    **end**
  **end**
  $B \leftarrow \arg\max_B \sum_{i=1}^C \sum_{j=1}^N \log p_B(\mathbf{z}_{ij}^{(s)}|\mathbf{x}_i^{(s)})$;
  $D \leftarrow \arg\max_D \sum_{i=1}^C \sum_{j=1}^N \log p_D(\mathbf{z}_{ij}^{(t)}|\mathbf{x}_j^{(t)})$;
  $W \leftarrow \arg\max_W \sum_{i=1}^C \sum_{j=1}^N \log p_W(y_{ij}^{(st)}|\mathbf{z}_{ij}^{(s)}, \mathbf{z}_{ij}^{(t)})$;
**until** *Converge to a local maximum*;
**return** $B, D, W$

likelihood functions. We propose a joint supervised dictionary learning approach in Sec. 2.3.

### 2.2.2 Training

During training time, we are given independent source and target domain instances, $\mathbf{x}_i^{(s)}, \mathbf{x}_j^{(t)}$, and a binary label $y_{ij}^{(st)}$ indicating whether or not they belong to the same class. We parametrize the probability kernels in Eq. 4 using $p_B(\mathbf{z}^{(s)}|\mathbf{x}^{(s)})$, $p_D(\mathbf{z}^{(t)}|\mathbf{x}^{(t)})$, $p_W(y^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)})$ in terms of *data-independent* parameters $B$, $D$, $W$ respectively, and estimate them discriminatively using training data.

Note that maximizing the RHS in Eq. 4 over latent embeddings $\mathbf{z}^{(s)}, \mathbf{z}^{(t)}$ is actually a joint optimization which needs to be conducted for every pair of source and target data instances $(\mathbf{x}^{(s)}, \mathbf{x}^{(t)})$. Therefore, in order to maximize the lower bound of the log-likelihood over the entire training data, we propose the following joint optimization problem for learning the parameters $B, D, W$:

$$\max_{B,D,W} \sum_{i=1}^{C} \sum_{j=1}^{N} \max_{\mathbf{z}^{(s)}, \mathbf{z}^{(t)}} \Big\{ \log p_B(\mathbf{z}^{(s)}|\mathbf{x}_i^{(s)}) + \log p_D(\mathbf{z}^{(t)}|\mathbf{x}_j^{(t)}) + \log p_W(y_{ij}^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}^{(t)}) \Big\}, \tag{23}$$

where $C$ is the size of the source domain training data (*i.e.* number of observed class labels) and $N$ is the size of the

target domain training data.

Here we emphasize the fact that any pair of latent embeddings $\left(\mathbf{z}^{(s)}, \mathbf{z}^{(t)}\right)$ in Eq. 23 are essentially *fully coupled*, *i.e*. one is a function of the other. In other words, the source (*resp.* target) domain latent embeddings should change with different target (*resp.* source) domain latent embeddings. This naturally suggests an alternating optimization mechanism for solving Eq. 23 as shown in Alg. 1. However, as we see clearly, this algorithm would lead to significantly high computational complexity because of the optimization for every pair of latent embeddings in source and target domains, especially for large-scale data.

Instead as a compromise for running speed, we propose the following training objective as the lower bound of Eq. 23 over the source and target domain data by pulling the operator $\max_{\mathbf{z}^{(s)},\mathbf{z}^{(t)}}$ out of double-summations:

$$\max_{B,D,W} \max_{\{\mathbf{z}_i^{(s)}\},\{\mathbf{z}_j^{(t)}\}} N\sum_{i=1}^{C} \log p_B(\mathbf{z}_i^{(s)}|\mathbf{x}_i^{(s)}) \tag{24}$$
$$+ C\sum_{j=1}^{N} \log p_D(\mathbf{z}_j^{(t)}|\mathbf{x}_j^{(t)}) + \sum_{i=1}^{C}\sum_{j=1}^{N} \log p_W(y_{ij}^{(st)}|\mathbf{z}_i^{(s)}, \mathbf{z}_j^{(t)}).$$

Although in this relaxation $\mathbf{z}^{(s)}, \mathbf{z}^{(t)}$ are still coupled, the latent embeddings for both source and target domain data are fixed. That is, for $\mathbf{x}_i^{(s)}, \forall i$ (*resp.* $\mathbf{x}_j^{(t)}, \forall j$), there exists only one corresponding latent embedding $\mathbf{z}_i^{(s)}$ (*resp.* $\mathbf{z}_j^{(t)}$). Therefore, fundamentally different from Eq. 23, the relaxation in Eq. 24 significantly reduces the computational complexity of our model in training time. In the rest of paper, we consider Eq. 24 as our training objective by default without explicit mention.

**Salient Aspects of our Training Algorithm:** Based on Eq. 24 our objective is two-fold. We need to learn a low-dimensional latent embedding that not only accurately represents the observed data in each domain but also is capable of inferring cross-domain statistical relationships when one exists. Note that the first two log-likelihoods in Eq. 24 are data fitting terms, and the last one measures the *joint latent similarity* between the two latent vectors.

With this insight we propose a general alternating optimization algorithm to jointly learn $\{\mathbf{z}_i^{(s)}\}, \{\mathbf{z}_j^{(t)}\}, B, D, W$ in Eq. 24 in Alg. 2. This follows from the exchangeability of two max operators. In this way our learning algorithm guarantees *convergence* to a local optimum within finite number of iterations. Also since the update rules for $\forall i, \mathbf{z}_i^{(s)}$ (or $\forall j, \mathbf{z}_j^{(t)}$) are independent given $\forall j, \mathbf{z}_j^{(t)}$ (or $\forall i, \mathbf{z}_i^{(s)}$) and parameters $B, D, W$, we can potentially utilize parallel or distributed computing to train our models. This has obvious computational benefits.

Our approach diverts from some of the previous works

**Algorithm 2** Simplified jointly latent embedding learning algorithm for solving Eq. 24

**Input** : training data $\{(\mathbf{x}_i^{(s)}, y_i^{(s)})\}$ and $\{(\mathbf{x}_j^{(t)}, y_j^{(t)})\}$

**Output**: $\{\mathbf{z}_i^{(s)}\}, \{\mathbf{z}_j^{(t)}\}, B, D, W$

Initialize $B, D$;

$\forall i, \mathbf{z}_i^{(s)} \leftarrow \arg\max_{\mathbf{z}^{(s)}} \log p_B(\mathbf{z}^{(s)}|\mathbf{x}_i^{(s)})$;

$\forall j, \mathbf{z}_j^{(t)} \leftarrow \arg\max_{\mathbf{z}^{(t)}} \log p_D(\mathbf{z}^{(t)}|\mathbf{x}_j^{(t)})$;

$W \leftarrow \arg\max_W \sum_{i=1}^{C}\sum_{j=1}^{N} \log p_W(y_{ij}^{(st)}|\mathbf{z}_i^{(s)}, \mathbf{z}_j^{(t)})$;

**repeat**

$\forall i, \mathbf{z}_i^{(s)} \leftarrow \arg\max_{\mathbf{z}^{(s)}} \log p_B(\mathbf{z}^{(s)}|\mathbf{x}_i^{(s)}) + \sum_{j=1}^{N} \log p_W(y_{ij}^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}_j^{(t)})$;

$\forall j, \mathbf{z}_j^{(t)} \leftarrow \arg\max_{\mathbf{z}^{(t)}} \log p_D(\mathbf{z}^{(t)}|\mathbf{x}_j^{(t)}) + \sum_{i=1}^{C} \log p_W(y_{ij}^{(st)}|\mathbf{z}_i^{(s)}, \mathbf{z}^{(t)})$;

$B \leftarrow \arg\max \sum_{i=1}^{C} \log p_B(\mathbf{z}_i^{(s)}|\mathbf{x}_i^{(s)})$;

$D \leftarrow \arg\max \sum_{j=1}^{N} \log p_D(\mathbf{z}_j^{(t)}|\mathbf{x}_j^{(t)})$;

$W \leftarrow \arg\max_W \sum_{i=1}^{C}\sum_{j=1}^{N} \log p_W(y_{ij}^{(st)}|\mathbf{z}_i^{(s)}, \mathbf{z}_j^{(t)})$;

**until** *Converge to a local maximum*;

**return** $\{\mathbf{z}_i^{(s)}\}, \{\mathbf{z}_j^{(t)}\}, B, D, W$

such as [14] where source domain vectors for unseen classes are also known during training. This perspective lets one exploit knowledge of unseen source domain classes during training. In contrast we are not provided unseen data for either the source or target domains. Thus, our data-independent variables $B, D, W$ do not contain any information about unseen data.

### 2.2.3 Testing

In order to avoid confusion we index unseen class data with $i'$, $j'$ corresponding to source and target domain respectively. The seen class training data is indexed as before with $i, j$. During test time the source domain data $\{(\mathbf{x}_{i'}^{(s)}, y_{i'}^{(s)})\}$ for all the unseen classes are revealed. We are then presented with an instance of unseen target domain data, $\{\mathbf{x}_{j'}^{(t)}\}$. Our objective is to identify an unseen source domain vector that best matches the unseen instance.

Considering Eq. 5 and Eq. 23, naturally we have the following test-time decision function:

$$y_{j'}^{(t)} = y_{i'_*}^{(s)}, \text{ s.t. } i'_* = \arg\max_{i' \in [C']} \Big\{ \max_{\mathbf{z}^{(s)},\mathbf{z}^{(t)}} \Big\{ \log p_B(\mathbf{z}^{(s)}|\mathbf{x}_{i'}^{(s)}) \tag{25}$$
$$+ \log p_D(\mathbf{z}^{(t)}|\mathbf{x}_{j'}^{(t)}) + \log p_W(y_{i'j'}^{(st)} = 1|\mathbf{z}^{(s)}, \mathbf{z}^{(t)}) \Big\} \Big\},$$

where $C'$ and $[C']$ denote the number of unseen classes and the index set of unseen classes starting from 1, respectively.

Similar to solving Eq. 23 in training time, Eq. 25 also suggests an alternating optimization algorithm to determine the maximum similarity between any pair of unseen source and target domain data, as shown in Alg. 3. Still the high

**Algorithm 3** Joint latent embedding testing algorithm

**Input** : test data $\{(\mathbf{x}_{i'}^{(s)}, y_{i'}^{(s)})\}$ and $\{\mathbf{x}_{j'}^{(t)}\}$; learned parameters $B, D, W$ during training

**Output**: $\{y_{j'}^{(t)}\}$

$\forall i', \mathbf{z}_{i'}^{(s)} \leftarrow \arg\max_{\mathbf{z}_{i'}^{(s)}} \log p_B(\mathbf{z}_{i'}^{(s)}|\mathbf{x}_{i'}^{(s)})$;

$\forall j', \mathbf{z}_{j'}^{(t)} \leftarrow \arg\max_{\mathbf{z}_{j'}^{(t)}} \log p_D(\mathbf{z}_{j'}^{(t)}|\mathbf{x}_{j'}^{(t)})$;

**foreach** $j'$ **do**

  $\mathcal{S} \leftarrow \emptyset$;

  **foreach** $i'$ **do**

    $\mathbf{z}_{i'j'}^{(s)}, \mathbf{z}_{i'}^{(s)}; \mathbf{z}_{i'j'}^{(t)} \leftarrow \mathbf{z}_{j'}^{(t)}$;

    **repeat**

      $\mathbf{z}_{i'j'}^{(s)} \leftarrow \arg\max_{\mathbf{z}^{(s)}} \log p_B(\mathbf{z}^{(s)}|\mathbf{x}_{i'}^{(s)}) + \log p_W(y_{i'j'}^{(st)}|\mathbf{z}^{(s)}, \mathbf{z}_{i'j'}^{(t)})$;

      $\mathbf{z}_{i'j'}^{(t)} \leftarrow \arg\max_{\mathbf{z}^{(t)}} \log p_D(\mathbf{z}^{(t)}|\mathbf{x}_{j'}^{(t)}) + \log p_W(y_{i'j'}^{(st)}|\mathbf{z}_{i'j'}^{(s)}, \mathbf{z}^{(t)})$;

    **until** *Converge to a local maximum*;

    $\mathcal{S} \leftarrow [\mathcal{S}; \log p_B(\mathbf{z}_{i'j'}^{(s)}|\mathbf{x}_{i'}^{(s)}) + \log p_D(\mathbf{z}_{i'j'}^{(t)}|\mathbf{x}_{j'}^{(t)}) + \log p_W(y_{i'j'}^{(st)}|\mathbf{z}_{i'j'}^{(s)}, \mathbf{z}_{i'j'}^{(t)})$;

  **end**

  $[s, i'_*] \leftarrow \max(\mathcal{S})$; $y_{j'}^{(t)} \leftarrow y_{i'_*}^{(s)}$;

**end**

**return** $\{y_{j'}^{(t)}\}$

computational complexity here prevents it from being used for large-scale data.

Alternatively we adopt the strategy in the relaxation of Eq. 6 to reduce the test-time computational complexity. That is, we would like to estimate the fixed latent embeddings for all the unseen source and target domain data so that prediction of the unseen classes is deterministic. In this way, there will be no $\max_{\mathbf{z}^{(s)},\mathbf{z}^{(t)}}$ involved in Eq. 25. To better estimate such embeddings we are also given seen class latent embeddings $\{\mathbf{z}_i^{(s)}\}$ and $\{\mathbf{z}_j^{(t)}\}$ and the parameters $B, D, W$ that are all learned during training. This naturally suggests the optimization algorithm in Alg. 4 by adapting the training algorithm in Alg. 2 to test time scenarios. Note that while the second term during this estimation process appears unusual we are merely exploiting the fact that the unseen class has no intersection with seen classes. Consequently, we can assume that $y_{i'j}^{(st)} = -1$, $y_{ij'}^{(st)} = -1$. Notice that the latent vector computation is again amenable to fast parallel or distributed computing.

**Decision function:** We next compute the likelihood of being the same class label, *i.e.* $p(y_{i'j'}^{(st)} = 1|\mathbf{x}_{i'}^{(s)}, \mathbf{x}_{j'}^{(t)})$, for an arbitrary target domain data $\mathbf{x}_{j'}^{(t)}$ using the source domain data $(\mathbf{x}_{i'}^{(s)}, y_{i'}^{(s)})$. Based on Eq. 25 there are two options: The first option is to directly employ latent estimates $\mathbf{z}_{i'}^{(s)}, \mathbf{z}_{j'}^{(t)}$ for $\mathbf{x}_{i'}^{(s)}, \mathbf{x}_{j'}^{(t)}$, respectively, and ignore the two data

**Algorithm 4** Test-time estimation of latent embeddings

**Input** : test data $\{(\mathbf{x}_{i'}^{(s)}, y_{i'}^{(s)})\}$ and $\{\mathbf{x}_{j'}^{(t)}\}$; learned latent embeddings for seen classes (training data) $\{\mathbf{z}_i^{(s)}\}$ and $\{\mathbf{z}_j^{(t)}\}$; learned parameters $B, D, W$ during training

**Output**: $\{\mathbf{z}_{i'}^{(s)}\}, \{\mathbf{z}_{j'}^{(t)}\}$

$\forall i', \mathbf{z}_{i'}^{(s)} \leftarrow \arg\max_{\mathbf{z}_{i'}^{(s)}} \log p_B(\mathbf{z}_{i'}^{(s)}|\mathbf{x}_{i'}^{(s)}) + \sum_{j=1}^{N} \log p_W(-1|\mathbf{z}_{i'}^{(s)}, \mathbf{z}_j^{(t)})$;

$\forall j', \mathbf{z}_{j'}^{(t)} \leftarrow \arg\max_{\mathbf{z}_{j'}^{(t)}} \log p_D(\mathbf{z}_{j'}^{(t)}|\mathbf{x}_{j'}^{(t)}) + \sum_{i=1}^{C} \log p_W(-1|\mathbf{z}_i^{(s)}, \mathbf{z}_{j'}^{(t)})$;

**return** $\{\mathbf{z}_{i'}^{(s)}\}, \{\mathbf{z}_{j'}^{(t)}\}$

fitting terms. This leads to the following expression (which is evidently related to the one employed in [1, 3, 45]):

$$y_{j'}^{(t)} = y_{i'_*}^{(s)}, \text{s.t.} i'_* = \arg\max_{i'} \left\{ \log p_W(y_{i'j'}^{(st)} = 1|\mathbf{z}_{i'}^{(s)}, \mathbf{z}_{j'}^{(t)}) \right\}. \quad (26)$$

A second option is to use Eq. 25 with fixed $\mathbf{z}_{i'}^{(s)}, \mathbf{z}_{j'}^{(t)}$ for prediction, which in turn leads us to:

$$y_{j'}^{(t)} = y_{i'_*}^{(s)}, \text{s.t.} i'_* = \arg\max_{i'} \left\{ \log p_B(\mathbf{z}_{i'}^{(s)}|\mathbf{x}_{i'}^{(s)}) + \log p_W(y_{i'j'}^{(st)} = 1|\mathbf{z}_{i'}^{(s)}, \mathbf{z}_{j'}^{(t)}) \right\}. \quad (27)$$

Note that the decision function in Eq. 27 is different from the one in Eq. 26, which is widely used in embedding methods (see Sec. 2.2.1). In Eq. 27 we also penalize source domain fit to identify the class label. Intuitively this choice optimizes the source domain embedding that best aligns with the target data. One reason for doing so is based on the fact that our information is asymmetric and the single source domain vector per class represents the strongest information about the class. Therefore, our attempt is to penalize the loss functions towards a source domain fit.

In general one could also view source domain embeddings $\mathbf{z}^{(s)}$ as a parameter in Eq. 27 and optimize it as well. This is computationally somewhat more expensive. While more experiments maybe necessary to see whether or not this leads to improved performance, we have not found this additional degree of freedom to significantly improve performance.

## 2.3. Parametrization

In this section we develop a *supervised dictionary learning (SDL)* formulation to parametrize Eq. 24. Specifically, we map data instances into the latent space as the coefficients based on a learned dictionary, and formulate an empirical risk function as the similarity measure which attempts to minimize the regularized hinge loss with the joint latent embeddings.

For purpose of exposition we overload notation in Sec. 2.2.2 and let $\mathbf{B} \in \mathbb{R}^{d_s \times h_s}, \mathbf{D} \in \mathbb{R}^{d_t \times h_t}, \mathbf{W} \in \mathbb{R}^{h_s \times h_t}$ as the source domain dictionary, target domain dictionary, and the cross-domain similarity matrix in the joint latent space, respectively. Here $d_s$ and $d_t$ are original feature dimensions, and $h_s$ and $h_t$ are the sizes of dictionaries. Then given the seen class source domain data $\{(\mathbf{x}_i^{(s)}, y_i^{(s)})\}$ and target domain data $\{(\mathbf{x}_j^{(t)}, y_j^{(t)})\}$, we choose to parametrize the three log-likelihoods in Eq. 24, denoted by $\log p_B, \log p_D, \log p_W$, respectively using dictionary learning and regularized hinge loss as follows. For source domain embedding, following [45], we enforce source domain latent coefficients to lie on a simplex (see Eq. 28 below). For target domain embedding, we follow the convention. We allow the latent vectors to be arbitrary while constraining the elements in the dictionary to be within the unit ball. Specifically, $\forall i, \forall j$, we have,

$$-\log p_B \triangleq \frac{\lambda_1^{(s)}}{2}\|\mathbf{z}_i^{(s)}\|_2^2 + \frac{\lambda_2^{(s)}}{2}\|\mathbf{x}_i^{(s)} - \mathbf{B}\mathbf{z}_i^{(s)}\|_2^2, \quad (28)$$
$$\text{s.t.} \quad \mathbf{z}_i^{(s)} \geq \mathbf{0}, \ \mathbf{e}^T\mathbf{z}_i^{(s)} = 1,$$
$$-\log p_D \triangleq \frac{\lambda_1^{(t)}}{2}\|\mathbf{z}_j^{(t)}\|_2^2 + \frac{\lambda_2^{(t)}}{2}\|\mathbf{x}_j^{(t)} - \mathbf{D}\mathbf{z}_j^{(t)}\|_2^2, \quad (29)$$
$$\text{s.t.} \quad \forall k, \ \|\mathbf{D}_k\|_2^2 \leq 1,$$
$$-\log p_W \triangleq \frac{\lambda_W}{2}\|\mathbf{W}\|_F^2 + \left[1 - \mathbf{1}_{y_{ij}^{(st)}}\left[\mathbf{z}_i^{(s)}\right]^T \mathbf{W}\mathbf{z}_j^{(t)}\right]_+, \quad (30)$$

where $\|\cdot\|_F$ and $\|\cdot\|_2$ are the Frobenius norm and $\ell_2$ norm operators, $\lfloor\cdot\rfloor_+ = \max\{0, \cdot\}$, $\geq$ is an entry-wise operator, $[\cdot]^T$ is the matrix transpose operator, $\mathbf{e}$ is a vector of 1's, and $\forall k, \mathbf{D}_k$ denotes the $k$-th row in the matrix $\mathbf{D}$. $\mathbf{1}_{y_{ij}^{(st)}} = 1$ if $y_i^{(s)} = y_j^{(t)}$ and $-1$ otherwise. The regularization parameters $\lambda_1^{(s)} \geq 0, \lambda_2^{(s)} \geq 0, \lambda_1^{(t)} \geq 0, \lambda_2^{(t)} \geq 0, \lambda_W \geq 0$ are fixed during training. Cross validation is used to estimate these parameters by holding out a portion of seen classes (see Sec. 3.1). With sufficient data (*i.e.* no need of regularization to avoid overfitting), our SDL approach indeed is equivalent to the relaxation of the following joint optimization problem:

$$\min_{\substack{\{\mathbf{z}_i^{(s)}\},\{\mathbf{z}_j^{(t)}\},\\ \mathbf{W},\mathbf{B},\mathbf{D}}} \sum_{i,j} \max\left\{0, 1 - \mathbf{1}_{y_{ij}^{(st)}}\left[\mathbf{z}_i^{(s)}\right]^T \mathbf{W}\mathbf{z}_j^{(t)}\right\} \quad (31)$$
$$\text{s.t.} \quad \mathbf{x}_i^{(s)} = \mathbf{B}\mathbf{z}_i^{(s)}, \ \mathbf{z}_i^{(s)} \geq 0, \ \mathbf{e}^T\mathbf{z}_i^{(s)} = 1, \ \forall i,$$
$$\mathbf{x}_j^{(t)} = \mathbf{D}\mathbf{z}_j^{(t)}, \ \forall j, \ \|\mathbf{D}_k\|_2^2 \leq 1, \ \forall k.$$

Observe that our method leverages association between the source domain and target domain vectors across all seen classes and learns a single matrix for all classes. Our objective function utilizes a hinge loss to penalize mis-associations between source and target pairs in the joint latent space.

**Training & Cross-Validation:** We hold-out data corresponding to two randomly sampled seen classes and train our method using Alg. 2 on the rest of the seen classes for different combinations of regularization parameters. Training is performed by substituting Eq. 28, 29, and 30 into Alg. 2. For efficient computation, we utilize proximal gradient algorithms [28] with simplex projection [8] for updating $\mathbf{z}_i^{(s)}, \forall i$ and $\mathbf{z}_j^{(t)}, \forall j$, respectively. We use linear SVMs to learn $\mathbf{W}$.

**Testing:** We substitute Eq. 28, 29, and 30 into Alg. 4 and run it by fixing all the parameters learned during training. This leads to estimation of the latent embeddings for unseen class source and target domain data. Then we apply Eq. 26 or 27 to predict the class label for target domain data.

Table 1. Statistics of different datasets, where "bin." and "cont." stand for binary value and continuous value, respectively.

| Dataset | # instances | # attributes | # seen/unseen classes |
|---|---|---|---|
| aP&Y | 15,339 | 64 (cont.) | 20 / 12 |
| AwA | 30,475 | 85 (cont.) | 40 / 10 |
| CUB-200-2011 | 11,788 | 312 (bin.) | 150 / 50 |
| SUN Attribute | 14,340 | 102 (bin.) | 707 / 10 |

## 3. Experiments

We test our method on four benchmark image datasets for zero-shot recognition and retrieval, *i.e.* aPascal & aYahoo (aP&Y) [10], Animals with Attributes (AwA) [17], Caltech-UCSD Birds-200-2011 (CUB-200-2011) [36], and SUN Attribute [29]. Table 1 summarizes the statistics in each dataset. In our experiments we utilized the same experimental settings as [45]. For comparison purpose we report our results averaged over 3 trials[2].

### 3.1. Implementation

**(i) Cross validation:** Similar to [45], we utilize cross validation to tune the parameters. Precisely, we randomly select two seen classes from training data for validation purpose, train our method on the rest of the seen classes, and record the performance using different parameter combinations. We choose the parameters with the best average performance on the held-out seen class data.

Wouldn't this best fit to those two classes?

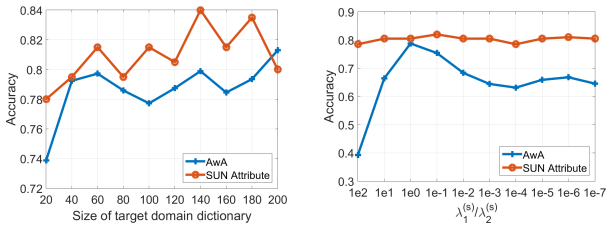
**(ii) Dictionary initialization:** For source domain, we initialize the dictionary $\mathbf{B}$ to be the collection of all the seen class attribute vectors on aP&Y, AwA, and CUB-200-2011, because of the paucity of the number of vectors. On SUN, however, for computational reasons, we initialize $\mathbf{B}$ using KMeans with 200 clusters on the attribute vectors.

For target domain, we utilize the top eigenvectors of all training data samples to initialize the dictionary $\mathbf{D}$. In

[2]Our code and CNN features can be downloaded at https://zimingzhang.wordpress.com/.

Table 2. Zero-shot recognition accuracy comparison (%) on the four datasets. Except for [2] where AlexNet [18] is utilized for extracting CNN features, for all the other methods we use vgg-verydeep-19 [33] CNN features.

| Method | aP&Y | AwA | CUB-200-2011 | SUN Attribute | Ave. |
|---|---|---|---|---|---|
| Akata *et al*. [2] | - | 61.9 | ***40.3*** | - | - |
| Lampert *et al*. [19] | 38.16 | 57.23 | - | 72.00 | - |
| Romera-Paredes and Torr [32] | 24.22±2.89 | 75.32±2.28 | - | 82.10±0.32 | - |
| SSE-INT [45] | 44.15±0.34 | 71.52±0.79 | 30.19±0.59 | 82.17±0.76 | 57.01 |
| SSE-ReLU [45] | ***46.23±0.53*** | ***76.33±0.83*** | 30.41±0.20 | ***82.50±1.32*** | ***58.87*** |
| (i) init. $\forall \mathbf{z}_i^{(s)}, \forall \mathbf{z}_j^{(t)}$ + init. $\forall \mathbf{z}_{i'}^{(s)}, \forall \mathbf{z}_{j'}^{(t)}$ + Eq. 26 | 38.10±2.64 | 76.96±1.40 | 39.03±0.87 | 81.17±2.02 | 58.81 |
| (ii) init. $\forall \mathbf{z}_i^{(s)}, \forall \mathbf{z}_j^{(t)}$ + init. $\forall \mathbf{z}_{i'}^{(s)}, \forall \mathbf{z}_{j'}^{(t)}$ + Eq. 27 | 38.20±2.75 | 80.11±1.13 | 41.07±0.81 | 81.33±1.76 | 60.20 |
| (iii) init. $\forall \mathbf{z}_i^{(s)}, \forall \mathbf{z}_j^{(t)}$ + Alg. 4 + Eq. 26 | 47.29±1.45 | 74.92±2.51 | 38.94±0.81 | 80.67±2.57 | 60.46 |
| (iv) init. $\forall \mathbf{z}_i^{(s)}, \forall \mathbf{z}_j^{(t)}$ + Alg. 4 + Eq. 27 | 47.79±1.83 | 77.37±0.39 | 40.91±0.86 | 80.83±2.25 | 61.73 |
| (v) Alg. 2 + init. $\forall \mathbf{z}_{i'}^{(s)}, \forall \mathbf{z}_{j'}^{(t)}$ + Eq. 26 | 39.13±2.35 | 77.58±0.81 | 39.92±0.20 | 83.00±1.80 | 59.91 |
| (vi) Alg. 2 + init. $\forall \mathbf{z}_{i'}^{(s)}, \forall \mathbf{z}_{j'}^{(t)}$ + Eq. 27 | 38.94±2.27 | ***80.46±0.53*** | ***42.11±0.55*** | 82.83±1.61 | 61.09 |
| (vii) Alg. 2 + Alg. 4 + Eq. 26 | 50.21±2.90 | 76.43±0.75 | 39.72±0.19 | 83.67±0.29 | 62.51 |
| (viii) Alg. 2 + Alg. 4 + Eq. 27 | ***50.35±2.97*** | 79.12±0.53 | 41.78±0.52 | ***83.83±0.29*** | ***63.77*** |




(a) (b)

Figure 3. Effect of **(a)** the size of target domain dictionary, and **(b)** source domain parameter ratio $\lambda_1^{(s)}/\lambda_2^{(s)}$ on accuracy.

Fig. 3(a), we show the effect of varying the size of $\mathbf{D}$ on our accuracy on AwA and SUN Attribute datasets. As we see, within small ranges of dictionary size, our performance changes marginally. We set the initial sizes to be 40, 200, 300, and 200, for the four datasets respectively, and then tune them using cross validation.

**(iii) Regularization parameters in Eq. 28, 29, and 30:** We do a grid search to tune these parameters. In order to show how well our method adapts to different parameters, we display salient results in Fig. 3(b), for varying source domain parameter ratios ($\lambda_1^{(s)}/\lambda_2^{(s)}$) on AwA and SUN datasets.

## 3.2. Benchmark Comparison

On the four datasets, we perform two different tasks: (1) *zero-shot recognition* and (2) *zero-shot retrieval*. While both tasks are related, they measure different aspects of the system. Task 1 is fundamentally about classification of each target data instance. Task 2 measures which target domain samples are matched to a given source domain vector, and we adapt our recognition system for the purpose of retrieval. Specifically, given a source domain unseen class attribute vector we compute the similarities for all the unseen target domain data and sort the similarity scores. We can then compute precision, recall, average precision (AP) etc. to measure retrieval accuracy.

### 3.2.1 Zero-Shot Recognition

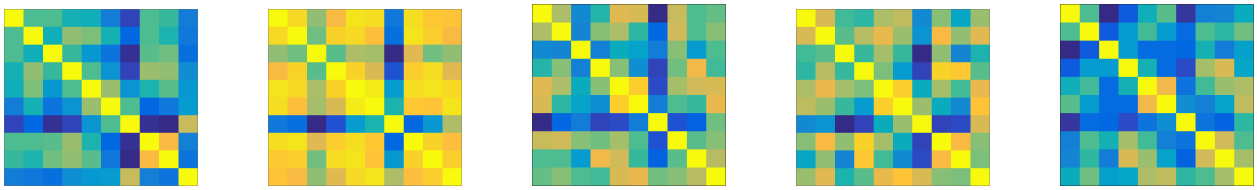
Recognition accuracy for each method is presented in Table 2. We also perform an ablative study in order to understand the contribution of different parts of our system. We experiment with the three parts of our system: (1) dictionary learning; (2) test-time latent variable estimation; (3) incorporating source domain data fit term in prediction.

Note that the source and target domain dictionaries $\mathbf{B}$ and $\mathbf{D}$ are initialized in the beginning of the dictionary learning process (see Sec 3.1 (ii)). Consequently, we can bypass dictionary learning (deleting *repeat* loop in Alg 2) and understand its impact. Next we can ignore the similarity function term for estimating the latent embeddings for unseen data during test-time. Finally, we can choose one of the two prediction rules (Eq. 26 or Eq. 27) to determine the utility of using source domain data fit term for prediction. We denote by "init. $\forall \mathbf{z}_i^{(s)}, \forall \mathbf{z}_j^{(t)}$" when dictionary learning is bypassed; We denote by "init. $\forall \mathbf{z}_{i'}^{(s)}, \forall \mathbf{z}_{j'}^{(t)}$" when similarity term is ignored during test-time. We list all the 8 choice combinations for our system in Table 2 (i) to (viii).

The overall best result is obtained for the most complex system using all parts of our system. For instance, as seen from (i) and (vii) we can see 3.70% gain in average recognition accuracy. Our algorithm "(viii) Alg. 2 + Alg. 4 + Eq. 27" achieves the best result among all the competitors, significantly outperforming the state-of-the-art by 4.90%. In the rest of the paper, we refer to (viii) as our method by default. Table 2 also demonstrates that on average, (a) the decision function in Eq. 27 performs better than that in Eq. 26, and (b) test-time learning of unseen class latent embeddings using Alg. 4 is more important than dictionary learning. For instance, by comparing (i) with (ii), using Eq. 27 the performance gains are 1.39% improvement over Eq. 26. We see modest gains (0.55%) from (iii) to (v). Still our ablative study demonstrates that on individual datasets

(a) SSE: decaf (b) Ours: decaf (c) SSE: verydeep-19 (d) Ours: verydeep-19

Figure 4. t-SNE visualization comparison between **(a, c)** SSE [45] and **(b, d)** our method using decaf and verydeep-19 features on AwA testing data from unseen classes, respectively. Clearly our method can better separate features from different classes.



(a) Attributes (b) SSE: decaf (c) Ours: decaf (d) SSE: verydeep-19 (e) Ours: verydeep-19

Figure 5. Comparison of cosine similarity matrices created using different features on AwA testing data using **(a)** source domain attribute vectors, **(b, d)** SSE [45] with decaf and verydeep-19, and **(c, e)** our method with decaf and verydeep-19, respectively. Brighter colors depict larger values.

there is no single system that dominates other system-level combinations. Indeed, for aP&Y (vi) is worse than (v).

We visually depict (see Fig. 4) the learned test-time unseen class embeddings, using t-SNE [35] on AwA to facilitate better understanding of our results with respect to the state-of-art [45]. Our method appears to learn more separable embeddings regardless of the target domain features (decaf [7] or verydeep-19). Indeed, as seen in Fig. 4 (b,d) the embeddings appear to be more cluttered than those in (a,c).

Next, in Fig. 5 we plot the cosine similarity matrices for the learned embeddings as in [45] on the AwA dataset. Note that [45] employs so called semantic similarity embedding (SSE). The figures demonstrate that our method can generate a cosine similarity matrix which is much more similar to the source domain attribute cosine similarity (a). Fig. 4 and Fig. 5 together demonstrate that our method is capable of aligning the source and target domain data better than the state-of-the-art method [45]. In addition it is capable of learning qualitatively better (clustered) embedding representations for different classes, leading to improvements in recognition accuracy on the four benchmark datasets.

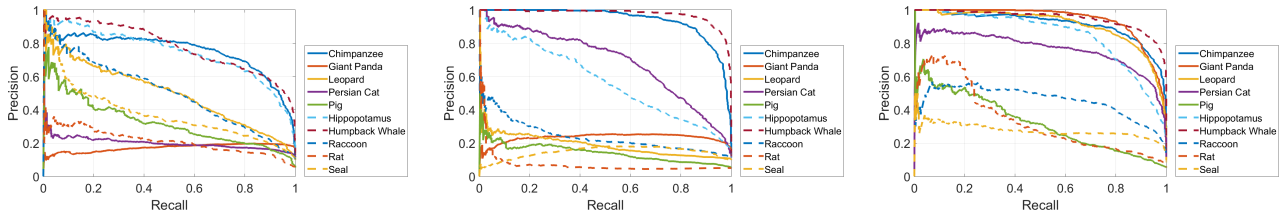
### 3.2.2 Zero-Shot Retrieval

We list comparative results for the mean average precision (mAP) for the four datasets in Table 3. Since retrieval is closely related to recognition and, SSE [45] is the state-of-art, we focus on comparisons with it. As we can see our method significantly and consistently outperforms SSE by 22.45% on average. Our superior performance in retrieval is due to the better domain alignment and more clustered

Figure 7. Top-5 zero-shot retrieval results using our method for class **(from top to down)** "Pig", "Raccoon", "Rat", and "Seal", respectively. Images with red rectangles are false-positive returns.

embedding representations. This leads to better matching of target domain data to source domain vectors. Our retrieval results are based on adapting the recognition models for the retrieval task. It is possible that incorporating pairwise ranking constraints into the training (*e.g.* into Eq. 30 for our method) may improve performance, but it is outside the scope of this paper.

We again attempt to further analyze our method on the AwA dataset. We list class-wise AP as well as mAP comparison in Table 4, and illustrate the precision-recall curves for different methods in Fig. 6. Our method achieves over 70% AP for 6 out of 10 classes, and performs the best in 6 out of 10 classes. Fig. 6 depicts illustrative examples for different categories. Nevertheless, we note that for some




(a) SSE-INT (b) SSE-ReLU (c) Ours

Figure 6. Illustration of precision-recall curve comparison on AwA.

Table 3. Retrieval performance comparison (%) using mAP.

| Method | aP&Y | AwA | CUB | SUN | Ave. |
|---|---|---|---|---|---|
| SSE-INT [45] | 15.43 | 46.25 | 4.69 | 58.94 | 31.33 |
| SSE-ReLU [45] | 14.09 | 42.60 | 3.70 | 44.55 | 26.24 |
| Ours | ***38.30*** | ***67.66*** | ***29.15*** | ***80.01*** | ***53.78*** |

Table 4. Retrieval performance comparison (%) using AP on AwA.

| Chim. | Panda | Leop. | Cat | Pig | Hipp. | Whale | Racc. | Rat | Seal | mAP |
|---|---|---|---|---|---|---|---|---|---|---|
| 76.05 | 19.67 | 50.12 | 20.33 | 32.83 | 74.88 | 78.31 | ***50.52*** | 21.85 | ***37.96*** | 46.25 |
| ***94.20*** | 24.81 | 19.24 | 69.08 | 14.73 | 57.51 | ***97.56*** | 24.11 | 7.59 | 17.20 | 42.60 |
| 91.75 | ***94.06*** | ***91.09*** | ***76.95*** | ***33.00*** | ***84.85*** | 95.13 | 47.05 | ***34.58*** | 28.18 | ***67.66*** |

classes our method is unable to achieve satisfactory performance (although other methods also suffer from performance degradation). For instance, we only get 28.18% AP for class "seal". Note that in Fig. 5(e), we can see that the last row (or column), which corresponds to "seal", shows some relatively high values in off-diagonal elements. This is because the problem of differentiating data within this class from data from other classes is difficult. Similar situations can be observed in SSE as well.

We also visualize our retrieval results in Fig. 7 with the top-5 returns for "difficult" cases (classes with AP less than 50%) in Table 4. Interestingly for the most difficult class "seal", all five images are correct. This is probably because the global patterns such as texture in the images are similar, leading to highly similar yet discriminative CNN features.

## 4. Conclusion

In this paper we propose a novel general probabilistic method for ZSL by learning joint latent similarity embeddings for both source and target domains. Based on the equivalence of ZSR and binary prediction, and the conditional independence between observed data and predicted class, we propose factorizing the likelihood of binary prediction using our probabilistic model to jointly learn the latent spaces for each domain. In this way, we generate a joint latent space for measuring the latent similarity between source and target data. Our similarity function is invariant across different classes, and hence intuitively it fits well to ZSR with good generalization to unseen classes. We further propose a new supervised dictionary learning based ZSR algorithm as parametrization of our probabilistic model. We conduct comprehensive experiments on four benchmark datasets for ZSL with two different tasks, *i.e.* zero-shot recognition and retrieval. We evaluate the importance of each key component in our algorithm, and show significant improvement over the state-of-the-art. Possible applications are person re-identification [42, 43, 44] and zero-shot activity retrieval [5].

## 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, 3, 4, 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. 1, 2, 3, 4, 9

[3] J. L. Ba, K. Swersky, S. Fidler, and R. Salakhutdinov. Predicting deep zero-shot convolutional neural networks using textual descriptions. *arXiv preprint arXiv:1506.00511*, 2015. 2, 3, 4, 7

[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. D. Castanon, Y. Chen, Z. Zhang, and V. Saligrama. Efficient activity retrieval through semantic graph queries. In *ACM Multimedia*, 2015. 11

[6] T. M. Cover and J. A. Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006. 3

[7] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of The 31st International Conference on Machine Learning*, pages 647–655, 2014. 10

[8] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the l 1-ball for learning in high dimensions. In *ICML*, pages 272–279, 2008. 8

[9] R. O. Duda, P. E. Hart, and D. G. Stork. Pattern classification and scene analysis 2nd ed, 1995. 3

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

[11] 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

[12] 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

[13] Z. Fu, T. Xiang, E. Kodirov, and S. Gong. Zero-shot object recognition by semantic manifold distance. In *CVPR*, pages 2635–2644, 2015. 2

[14] B. Hariharan, S. Vishwanathan, and M. Varma. Efficient max-margin multi-label classification with applications to zero-shot learning. *Machine learning*, 88(1-2):127–155, 2012. 2, 6

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

[16] E. Kodirov, T. Xiang, Z. Fu, and S. Gong. Unsupervised domain adaptation for zero-shot learning. In *ICCV*, 2015. 2

[17] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. Master's thesis, 2009. 8

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105. 2012. 9

[19] 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, 9

[20] X. Li and Y. Guo. Max-margin zero-shot learning for multi-class classification. In *AISTATS*, 2015. 2

[21] X. Li, Y. Guo, and D. Schuurmans. Semi-supervised zero-shot classification with label representation learning. In *ICCV*, 2015. 2

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

[23] 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

[24] 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

[25] 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

[26] 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

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

[28] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239, 2014. 8

[29] 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. 8

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

[31] 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

[32] B. Romera-Paredes and P. H. S. Torr. An embarrassingly simple approach to zero-shot learning. In *ICML*, 2015. 2, 9

[33] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 9

[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

[35] L. Van der Maaten and G. Hinton. Visualizing data using t-SNE. *JMLR*, 9(2579-2605):85, 2008. 10

[36] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011. 8

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

[38] 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

[39] Y. Xian, Z. Akata, G. Sharma, Q. Nguyen, M. Hein, and B. Schiele. Latent embeddings for zero-shot classification. In *CVPR*, 2016. 3, 4, 5

[40] 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

[41] X. Yu and Y. Aloimonos. Attribute-based transfer learning for object categorization with zero/one training example. In *ECCV*, pages 127–140. 2010. 2

[42] 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. 11

[43] Z. Zhang, Y. Chen, and V. Saligrama. Group membership prediction. In *ICCV*, 2015. 11

[44] Z. Zhang and V. Saligrama. PRISM: Person re-identification via structured matching. *arXiv preprint arXiv:1406.4444*, 2014. 11

[45] Z. Zhang and V. Saligrama. Zero-shot learning via semantic similarity embedding. In *ICCV*, 2015. 2, 3, 4, 7, 8, 9, 10, 11