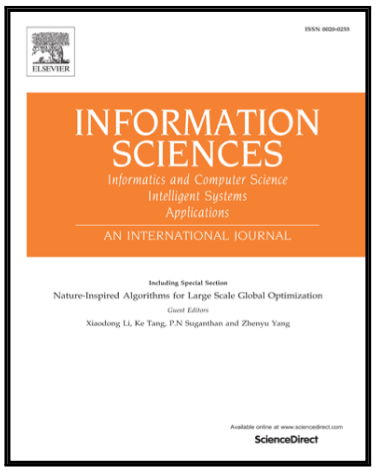


Dual-Verification Network for Zero-shot Learning

Haofeng Zhang, Yang Long, Wankou Yang, Ling Shao

PII: S0020-0255(18)30666-2
DOI: https://doi.org/10.1016/j.ins.2018.08.048
Reference: INS 13892

To appear in: *Information Sciences*

Received date: 28 April 2018
Revised date: 22 August 2018
Accepted date: 24 August 2018

Please cite this article as: Haofeng Zhang, Yang Long, Wankou Yang, Ling Shao, Dual-Verification Network for Zero-shot Learning, *Information Sciences* (2018), doi: https://doi.org/10.1016/j.ins.2018.08.048

Highlights

- Orthogonal projection is proposed to mitigate the domain shift problem;
- Semantic feature representation is included to alleviate visual category ambiguity;
- Deep model is applied to improve the performance;
- Extensive experiments show the superiority of our algorithm.

Dual-Verification Network for Zero-shot Learning

Haofeng Zhang[a], Yang Long[b], Wankou Yang[c], Ling Shao[d,e,*]

[a]*School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China*
[b]*Open Lab, School of Computing, Newcastle University, UK*
[c]*School of Automation, Southeast University, Nanjing, China*
[d]*Inception Institute of Artificial Intelligence (IIAI), Abu Dhabi, United Arab Emirates*
[e]*School of Computing Sciences, University of East Anglia, Norwich, UK*

Abstract

To mitigate the problems of visual ambiguity and domain shift in conventional zero-shot learning (ZSL), in this paper, we propose a novel method, namely, dual-verification network (DVN), which accepts features and attributes in a pairwise manner as input and verifies the result in both the attribute and feature spaces. First, the DVN projects a feature onto an orthogonal space, where the projected feature has maximum correlation with its corresponding attribute and is orthogonal to all the other attributes. Second, we adopt the concept of semantic feature representation, which computes the relationship between the semantic feature and class labels. Based on this concept, we project the attributes onto the feature space by extending the attributes and labels from the class level to instance level. In addition, we employ a deep architecture and utilize the cross entropy loss to train an end-to-end network for dual verification. Extensive experiments in ZSL and generalized ZSL are performed on four well-known datasets, and the results show that the proposed DVN exhibits a competitive performance relative to the state-of-the-art methods.

Keywords: Zero-shot Learning, Dual-verification Net, Orthogonal Projection, Semantic Feature Representation

*Corresponding author
Email addresses: zhanghf@njust.edu.cn (Haofeng Zhang), yang.long@ieee.org (Yang Long), wkyang@seu.edu.cn (Wankou Yang), ling.shao@ieee.org (Ling Shao)

1. Introduction

In recent times, numerous research studies have focused on extending image or video classification to a large-scale owing to the emergence of large-scale datasets such as ImageNet [6] and powerful techniques such as deep learning. However, image classification of large-scale datasets is still a major problem because there are many rare or fine-grained categories in addition to the common image classes, and training samples of these categories are difficult to collect. For example, there are 21,814 categories in ImageNet, among which 1,000 categories are common and easy to capture, and so, are often used for training. However, the remaining approximately 21000 categories of the sample images are uncommon and difficult to obtain. Particularly, 296 of these categories have only a single corresponding image. Therefore, it is necessary to determine methods for recognizing unseen images based on only the knowledge from the seen images. Humans can identify over 30,000 classes and are particularly good at recognizing unseen categories. For example, a child who has not seen a *'zebra'* before but knows that a *'zebra'* resembles a *'horse'* and has *'white and black stripes'*, will be able to very easily recognize a *'zebra'*. Many research studies classify the unseen classes using the method used by humans to recognize unseen classes, namely, *zero-shot learning* (ZSL) in the area of machine learning.

ZSL aims to learn a classification model that is trained on the samples belonging to the seen classes but can be transferred to be applied to the test data belonging to the unseen classes [22, 14, 46]. In zero-shot recognition, the seen and unseen classes are typically related in a high-dimensional vector space, which is called the semantic embedding space. Such a space is often an attribute space or a word vector space.

Conventional ZSL methods frequently rely on mapping visual features directly onto the semantic embedding space, *e.g.*, one of the best concepts is called attribute label embedding (ALE) [1], which learns the parameters of a function based on the max-margin loss to ensure that the projected features have the maximum distance between different class labels, whereas they have

the minimum distance between the same class labels. This type of projection for visual feature embedding is learned only from the seen classes, and hence, the projections of the unseen class images are expected to be shifted. Although the seen and unseen classes have overlapping domains in the embedding space, they are significantly different, *e.g.*, for the same embedding, the visual appearance results of the seen classes may be quite different compared with those of the unseen classes. This problem is often called the domain shift.

To mitigate the effect of domain shift, many researchers have introduced transductive learning methods [10], which assume that both labeled and unlabeled test data are available during the training process. These methods can significantly reduce the domain shift problem. However, in a realistic scenario, unlabeled test data are not strictly accessible. Thus, these methods are frequently discarded in practical applications. To solve this problem without using unseen data, E. Kodirov *et. al* proposed a method called the semantic auto-encoder (SAE)[17], which constructs an encoder–decoder paradigm, where the encoder projected a visual feature vector onto the semantic space, whereas the decoder exerted an additional constraint such that the projection was able to reconstruct the original visual feature. However, the SAE does not consider increasing the distance between the different classes, which leads to the problem of visual category ambiguity.

In this paper, we propose a novel method to exploit orthogonal projection and feature semantic representation, which can be considered as a dual verification, to solve the problems of domain shift and category ambiguity. First, to mitigate the problem of category ambiguity, we propose to project the visual features onto the semantic space and allow the projected vectors to have maximum correlation with their own attributes and be orthogonal to all the other attributes. The process of orthogonal projection can be considered as a verification conducted in the attribute space. Second, to alleviate the domain shift problem, we introduce the concept of semantic representation of features [49], which computes the relationship between the feature semantics and class labels and projects class level attributes onto the feature space. This concept

also can be considered as a verification conducted in the feature space by extending the attributes and label vectors from the class level to instance level. Therefore, our method can be treated as a dual verification in both the attribute and feature spaces. Moreover, owing to the deep neural network performs successfully in many applications [42, 34, 43, 33, 12, 37, 41, 45], thus, in this work, we adopted the cross entropy loss and replaced the linear projection matrix with an end-to-end deep network, namely, the dual-verification net (DVN), to achieve a better performance. We tested our method in both the attribute and feature spaces in four well-known datasets for examining the accuracy of both ZSL and generalized ZSL (GZSL) and obtained competitive results relative to the state-of-the-art methods.

The following is the list of our contributions: 1) Mitigation of the problem of visual ambiguity by performing an orthogonal projection to project the features onto the orthogonal attribute space, in which all the projected class level attributes were orthogonal to each other. This projection can be considered as a verification in the attribute space; 2) Alleviation of the domain shift problem by including the semantic feature representation to represent the relationship between the feature semantics and class labels, which can be treated as a verification in the feature space; 3) Construction of an end-to-end deep DVN to learn a zero-shot recognition model, which can exhibit a competitive performance compared with the state-of-the-art methods.

The remainder of this paper is organized as follows. In Section 2, we provide a brief review of the recent ZSL methods. The details of our methods for the orthogonal projection, feature semantic representation, and DVN are described in Section 3. Section 4 reports the experimental results of the ZSL and GZSL and analyses the hyper-parameter and distribution of the projected features in the attribute space. Finally, the results of this study are concluded in Section 5.

2. Related Works

Zero-shot Learning Since the proposal of visual attributes [8], extensive research studies [15, 18, 29, 35] have been conducted to identify the approach for learning the intermediate attribute classifiers for ZSL tasks. Based on the methods for using the features and attributes, we simply classify the methods into four categories, namely, compatibility learning, hybrid learning, transductive learning, and synthetic learning.

In the first category, the **compatibility learning** framework first learns a linear or non-linear projection from the feature space to the attribute space or latent space by using only the seen features and attributes and then is applied to unseen features. This category of methods includes linear models such as direct attribute prediction (DAP) [19], deep visual semantic embedding (DEVISE) [9], attribute label embedding (ALE) [1], structured joint embedding (SJE) [2], and semantic auto-encoder (SAE) [17], and non-linear models such as latent embedding (LATEM) [39], cross model transfer (CMT) [35], and semantically consistent regularization (SCoRe) [27].

DAP [19] is one of the most fundamental compatibility algorithms for ZSL; it learns probabilistic attribute classifiers and predicts a label by combining the ranks of the learnt attribute classifiers. ALE [1], DEVISE, [9] and SJE [2] use a bi-linear function to project features onto the embedding space or latent space, and thereby, maximize the similarity in the related features and attributes in that space and minimize the unrelated features and attributes. SAE utilizes an encoder-decoder paradigm that adds an additional decoder constraint to the original encoder constraint, i.e., the projected code must be able to reconstruct the original visual feature. Embarrassingly simple ZSL (ESZSL) [32] adds a regularization term to the unregularized risk minimization formulation.

The LATEM [39] model extends the linear projection to a non-linear piecewise mode, learns a set of mappings with a set of selections, and trains with a ranking-based objective function that minimizes the incorrect matching of the true class for a given image. The CMT [35] projects images onto the se-

mantic word space, in which the mapping is learnt using a neural network. Furthermore, the CMT is improved using the novelty detection method to differentiate the unseen classes from the seen classes. A study [47] proposed a deep embedding model that used the visual space as the embedding space instead of embedding in the semantic space or an intermediate space, aiming to solve the hubness problem of the subsequent nearest neighbor search. SCoRe [27] leverages the advantages of both recognition using independent semantics (RIS) [19] and recognition using semantic embedding (RULE) [31]. It enforces first-order constraints (single semantics) and second-order (linear combinations) constraints together and exploits the view of a CNN as the optimal classifier for a multi-dimensional classification code. Our proposed method also has a non-linear compatibility learning framework.

In the second category, semantic similarity embedding (SSE) [48] and combination of semantic embedding (CONSE) [28] express the features and semantic embedding attributes as a mixture of the seen class proportions and assume that the mixture patterns have to be similar if both the features belong to the same unseen class. Therefore, we call these methods as **hybrid learning**. SSE learns embedding functions that project an seen/unseen feature onto the same semantic space where the similarity can be calculated. CONSE learns the probability of a seen feature belonging to a seen class and uses a CONSE to assign an unseen feature to an unseen class. Synthesized classifiers (SYNC) [4] learn a mapping between the semantic class embedding and model spaces. In the model space, the training classes and a set of phantom classes construct a weighted bipartite graph. The semantic and model spaces are aligned by embedding real and phantom classes in the weighted graph.

Recently, a new research direction for ZSL was proposed, namely, **transductive learning** [10, 11, 16, 20], which postulates that in an ZSL problem, the seen class source including the features and their corresponding attributes is provided and unlabeled target domain data are also collected for learning a mapping function.

One of the earliest concepts of transductive learning was proposed by Y.

Fu *et al.*[10], who learned a multi-label regression model to well-generalize the unseen classes with both seen and unseen data. A semi-supervised framework [20] considers both the labeled data from the seen classes and unlabeled data from the unseen classes as input and learns a multi-class classification model on all the classes jointly. This framework can consistently learn both the label representations and model parameters across the seen and unseen classes. Y. Guo *et al.* [11] proposed a method to solve transductive ZSL with a shared model space (SMS), which is used to replace the shared attribute space in the existing works. Within an SMS, the model parameters for a target class can be generated directly via attribute representation. Unsupervised domain adaptation (UDA) [16] casts the visual-embedding projection function learning problem as a sparse coding problem, in which each dimension of the semantic embedding space is set to a dictionary basis vector and the coefficient/sparse code of each visual feature vector is its projection in the semantic embedding space. Additionally, UDA also adds constraints that the dictionary of the target domain should be similar to the that of the source domain and the embedded target data should be near to that of the unseen class prototypes. Recently, Y. Li *et al.* [21] exploited and formalized the intrinsic relationship between the semantic space manifold and transfer ability of visual-semantic mapping and cast zero-shot recognition as a joint optimization problem.

Although transductive learning can significantly reduce the domain shift problem, its setting differs from the original objective of ZSL because the target domain data is strictly inaccessible in realistic scenarios.

The last category, **synthetic learning** [47, 25, 23, 44] is a new type of method for ZSL that generates new features or new models from the original semantic embedding and then uses conventional classifiers such as SVM and LDA to train a model.

D. Wang *et al.* [36] proposed extracting the relational knowledge from a data manifold structure in the semantic knowledge space using the sparse coding theory. The extracted knowledge was then transferred backward to generate virtual data for the unseen categories in the feature space. J. Lu *et al.* [25] proposed

a new approach by generating pseudo feature representations (GPFRs) that used the dataset of the seen classes and side information of the unseen classes (e.g., attributes) to form the feature level pseudo representations for the unseen classes used to train a model of the unseen class predictor. L. Zhang *et al.* [47] suggested to use the visual space as the embedding space instead of embedding the features into the semantic space or an intermediate space, and then use a deep network model to train a generator. Y. Long *et al.* [23] proposed a framework that could generate visual features for the unseen classes using the unseen visual data synthesis (UVDS) method. The semantic attributes were utilized as intermediate clues in the generation of unseen visual features. Hereafter, ZSL recognition is converted into the conventional supervised classification problem, *i.e.*, the produced visual features can be directly fed to typical classifiers such as SVM. Y. Guo *et al.* [44] utilized the probability distribution of the seen classes and class attributes to estimate the distribution of the unseen class, which was then used to generate fake features for the subsequent training of the supervised classification.

Semantic embedding ZSL-related methods often depend on intermediate attributes, which represent the semantic embedding of both the seen and unseen classes. Conventional attributes [13] are high dimensional and typically annotated with real values by experts. This type of annotation needs expert knowledge and a high manpower cost. To solve this problem, some methods [3] use Word2Vec to generate attributes based on the dataset, *'Wikipedia'*. However, the textual description in *'Wikipedia'* might be very noisy and not directly related to the visual appearance, which often leads to a major degradation of the performance. Another semantic attribute representation is based on similarity, which can be annotated by humans [24] or textual vectors [5].

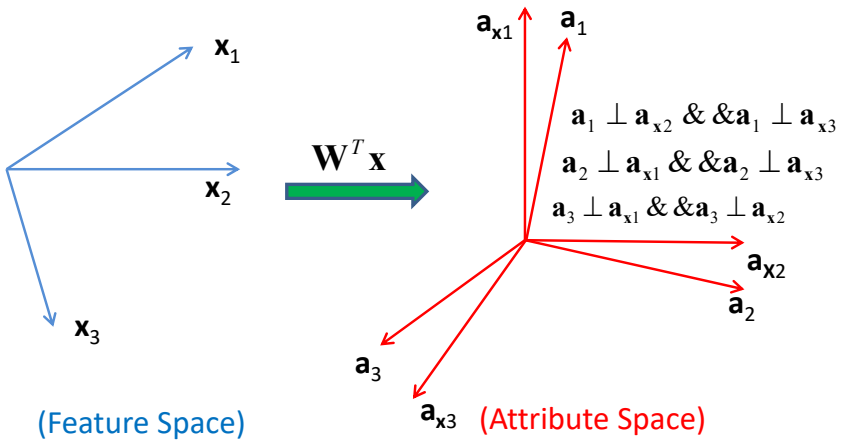


Figure 1: Illustration of the framework of the prthogonal projection. We project features onto the attribute space, where the projected vectors in the different classes are orthogonal to each other, whereas the projected features and attributes belonging to same class lie in the same/nearby directions. a_i is the attribute of the i^{th} class, and a_{xi} represents the projected vector from x_i. '$\cdot\perp\cdot$' represents $\cdot$s are perpendicular to each other.

3. Methodology

3.1. Problem Definition

Let $\boldsymbol{Y} = \{y_1, \cdots, y_s\}$ and $\boldsymbol{Z} = \{z_1, \cdots, z_u\}$ denote a set of s seen and u unseen class labels, which are disjoint $\boldsymbol{Y} \cap \boldsymbol{Z} = \emptyset$. Similarly, let $\boldsymbol{A_Y} = \{\boldsymbol{a_{y1}}, \cdots, \boldsymbol{a_{ys}}\} \in \mathbb{R}^{l\times s}$ and $\boldsymbol{A_Z} = \{\boldsymbol{a_{z1}}, \cdots, \boldsymbol{a_{zu}}\} \in \mathbb{R}^{l\times u}$ denote the corresponding s seen and u unseen attributes, respectively. Given the training data in a three-tuple of N seen samples: $(\boldsymbol{x}_1, \boldsymbol{a}_1, \boldsymbol{y}_1), \cdots, (\boldsymbol{x}_N, \boldsymbol{a}_N, \boldsymbol{y}_N) \subseteq \boldsymbol{X}_s \times \boldsymbol{A_Y} \times \boldsymbol{Y}$, where $\boldsymbol{X}_s$ denotes d-dimensional features extracted from N seen images. When testing, the preliminary knowledge is u pairs of attributes and labels:$(\hat{\boldsymbol{a}}_1, \hat{\boldsymbol{z}}_1), \cdots, (\hat{\boldsymbol{a}}_u, \hat{\boldsymbol{z}}_u) \subseteq \boldsymbol{A_Z} \times \boldsymbol{Z}$. ZSL aims to learn a classification function, f: $\boldsymbol{X}_u \rightarrow \boldsymbol{Z}$ to predict the label of the input image from unseen classes, where $\boldsymbol{x}_i \in \boldsymbol{X}_u$ is totally unavailable during training.

3.2. Linear Method

3.2.1. Orthogonal Projection

Given input visual data or feature matrix $\boldsymbol{X} \in \mathbb{R}^{d \times N}$, where N is the number of input samples and d is the dimension of each feature. We also have another input matrix, namely, semantic attribute matrix $\boldsymbol{A} \in \mathbb{R}^{m \times C}$, where m is the dimension of each attribute and C is the number of categories.

We aim to discover linear projection matrix $\boldsymbol{W} \in \mathbb{R}^{d \times m}$, which is used to project feature $\boldsymbol{x_i} \in \boldsymbol{X}$ into attribute space $\mathcal{A}$. We require that if $\boldsymbol{x_i}$ and $\boldsymbol{a_i}$ belong to same category, the inner product of projected vectors $\boldsymbol{W x_i}$ and $\boldsymbol{a_i}$ should be 1, otherwise, it should be 0, implying that if these two vectors belong to same category, they should have same direction, otherwise they should be orthogonal to each other in the attribute space. Therefore, we can obtain the following equation:

$$< \boldsymbol{W}^T \boldsymbol{x}_i, \boldsymbol{a}_i >= s_i, \tag{1}$$

where $s_i \in \{0, 1\}$ is the similarity value of $\boldsymbol{x_i}$ and $\boldsymbol{a_i}$, $< \cdot >$ is the inner product. Equation (1) can also be written in matrix form,

$$\boldsymbol{X}^T \boldsymbol{W} \boldsymbol{A} = \boldsymbol{B}^T, \tag{2}$$

where in matrix $\boldsymbol{B} \in \mathbb{R}^{C \times N}$, each column $\boldsymbol{b}_i$ is the one-hot vector label of the corresponding feature, which is equivalent to the corresponding feature and attribute belonging to the same category, its value is set as 1, otherwise 0.

We aim to obtain the best $\boldsymbol{W}$ to fit equation (2) with excessive samples. To achieve a better result, we use the method of least square error (LSE) to solve the problem and define the following loss function:

$$\mathcal{L}(\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{B} | \boldsymbol{W}) = \| \boldsymbol{X}^T \boldsymbol{W} \boldsymbol{A} - \boldsymbol{B}^T \|_F^2 + \beta \| \boldsymbol{W} \|_F^2, \tag{3}$$

where $\| \cdot \|_F$ is the Frobenius norm. The first term in equation (3) corresponds to constraining the verification loss of the feature projection, the second term represents the regularization of $\boldsymbol{W}$, and β is a weighting coefficient that controls the importance of the first and second terms.

3.2.2. Learning Semantic Representation of Features

Till now, we have developed a relation between the features and attributes by an orthogonal projection. In this section, we try to impose another constraint to improve projection matrix $\boldsymbol{W}$ via learning feature semantics.

We first define new matrix $\boldsymbol{G} = \boldsymbol{X}\boldsymbol{B}^T \in \mathbb{R}^{m\times C}$ to represent the correlation between the image features and labels, which can be referred to as high-level concepts. Note that $\boldsymbol{G}_{ij} = \sum_k \boldsymbol{X}_{ik} \cdot \boldsymbol{B}_{kj}$, where $\boldsymbol{X}_{ik}$ is the value of the i^{th} visual feature in the k^{th} image, and $\boldsymbol{B}_{kj}$ is the similarity value of the j^{th} class with the k^{th} image. $\boldsymbol{G}_{ij}$ is large when some of the values of the i^{th} visual feature in the images with the j^{th} class are large, which implies that if G_{ij} is large then the i^{th} image feature and j^{th} class may have a strong correlation.

Motivated by the latent semantic analysis (LSA) [50] and to extract the latent semantic features from the image feature, we apply a matrix factorization to decompose $\boldsymbol{G}$ into latent factor matrices as,

$$\boldsymbol{G} = \boldsymbol{U}^T\boldsymbol{V} \Longleftrightarrow \boldsymbol{X}\boldsymbol{B}^T = \boldsymbol{U}^T\boldsymbol{V}, \tag{4}$$

where $\boldsymbol{U} \in \mathcal{R}^{g\times d}$, $\boldsymbol{V} \in \mathcal{R}^{g\times C}$ and g is the number of latent factors. Then $\boldsymbol{u}_i$ can be considered as a latent semantic representation of the i^{th} image feature and $\boldsymbol{v}_j$ can be treated as a latent semantic representation of thw j^{th} label. Here, we consider that $\boldsymbol{W}$ should be the semantic representation of image features and $\boldsymbol{A}$ should be the semantic representation of the labels. Therefore, here we set $g = m$ and replace $\boldsymbol{U}^T$ and $\boldsymbol{V}$ with $\boldsymbol{W}$ and $\boldsymbol{A}$, respectively, then we can obtain $\boldsymbol{X}\boldsymbol{B}^T = \boldsymbol{W}\boldsymbol{A}$.

Considering the above-mentioned orthogonal projection strategy, we combine these two constraints and obtain the following formulation:

$$\mathcal{L}(\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{B}|\boldsymbol{W}) = \|\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{A} - \boldsymbol{B}^T\|_F^2 + \alpha\|\boldsymbol{X}\boldsymbol{B}^T - \boldsymbol{W}\boldsymbol{A}\|_F^2 + \beta\|\boldsymbol{W}\|_F^2, \tag{5}$$

where α and β are the weighting parameters for controlling the balance of the three items.

Since equation (5) has a standard quadratic formulation, it is a convex function, which has a global optimal solution, and can achieve a closed-form solution.

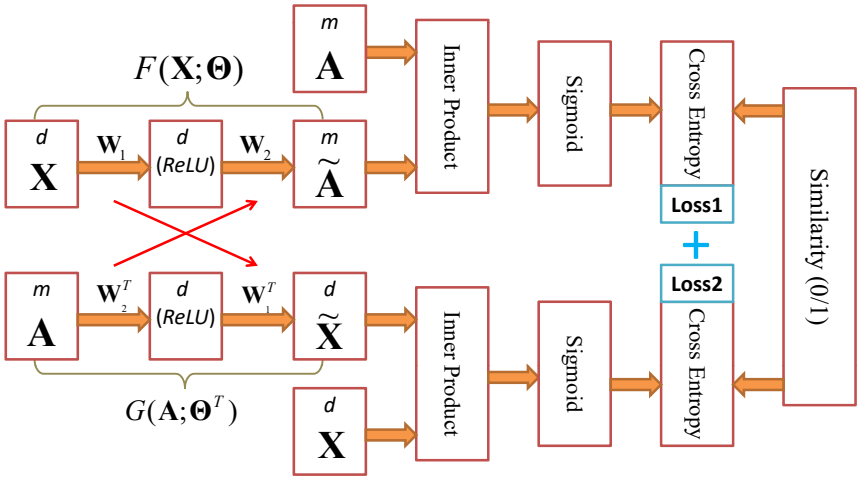


Figure 2: Illustration of the training framework of our dual-verification network.

To optimize this, we simply consider a derivative of equation (5) with respect to $\boldsymbol{W}$ and then set it as 0. We can obtain the following equation:

$$(\boldsymbol{X}\boldsymbol{X}^T + \alpha\boldsymbol{I})\boldsymbol{W} + \boldsymbol{W}(\beta(\boldsymbol{A}\boldsymbol{A}^T)^{-1}) = (1+\alpha)\boldsymbol{X}\boldsymbol{B}^T\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T)^{-1}. \quad (6)$$

If we define $\hat{\boldsymbol{A}} = \boldsymbol{X}\boldsymbol{X}^T + \alpha\boldsymbol{I}$, $\hat{\boldsymbol{B}} = \beta(\boldsymbol{A}\boldsymbol{A}^T)^{-1}$ and $\hat{\boldsymbol{C}} = (1+\alpha)\boldsymbol{X}\boldsymbol{B}^T\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{A}^T)^{-1}$, then we can obtain,

$$\hat{\boldsymbol{A}}\boldsymbol{W} + \boldsymbol{W}\hat{\boldsymbol{B}} = \hat{\boldsymbol{C}}. \quad (7)$$

Equation (7) is the well-known Sylvester equation, which can be solved efficiently by the Bartels- Stewart algorithm. It can be solved in MATLAB by using only a one line code, $W = sylvester(\hat{A}, \hat{B}, \hat{C})$[1].

3.3. Deep Dual-verification Network

In this section, for the purpose of improving the performance of the verification function, we will extend the linear projection introduced in the last subsection to a non-linear deep projection.

[1] https://uk.mathworks.com/help/matlab/ref/sylvester.html

3.3.1. Network Model

To use a deep net structure, we extend matrix $\boldsymbol{A}$ from class level $\mathcal{R}^{C\times m}$ to instance level $\mathcal{R}^{N\times m}$. Correspondingly, matrix $\boldsymbol{B}$ is also extended from $\{0,1\}^{C\times N}$ to $\{0,1\}^{N\times N}$, which can be treated as the similarity matrix between N image features $\boldsymbol{X}$ and N instance level attributes $\boldsymbol{A}$. Furthermore, we rewrite equation (5) as follows:

$$\mathcal{L}(\boldsymbol{X},\boldsymbol{A},\boldsymbol{B}|\boldsymbol{W}) = \frac{1}{N^2}\sum_{i,j=1}^{N}(\boldsymbol{x}_i^T\boldsymbol{W}\boldsymbol{a}_j - b_{ij})^2 + \frac{\alpha}{N}\sum_{i=1}^{N}\|\boldsymbol{a}_i^T\boldsymbol{W}^T - \boldsymbol{x}_i\|_F^2 + \beta\|\boldsymbol{W}\|_F^2. \tag{8}$$

We extend and rewrite the second item as the first item using similarities, and utilize the verification form to represent the total formulation. Then we can obtain,

$$\begin{aligned}\mathcal{L}(\boldsymbol{X},\boldsymbol{A},\boldsymbol{B}|\boldsymbol{W}) = &\frac{1}{N^2}\sum_{i,j=1}^{N}(<\boldsymbol{W}^T\boldsymbol{x}_i,\boldsymbol{a}_j> - b_{ij})^2 \\ &+ \frac{\alpha}{N}\sum_{i,j=1}^{N}(<\boldsymbol{W}\boldsymbol{a}_i,\boldsymbol{x}_j> - b_{ij})^2 + \beta\|\boldsymbol{W}\|_F^2.\end{aligned} \tag{9}$$

From equation (9), we can note that if we replace linear projection matrix $\boldsymbol{W}$ with nonlinear functions $F(\boldsymbol{x}_i;\boldsymbol{\Theta})$ and $G(\boldsymbol{a}_i;\boldsymbol{\Theta}^T)$ and use cross entropy (CE) to substitute the LSE, then equation (9) can be represented as,

$$\begin{aligned}\mathcal{L}(\boldsymbol{X},\boldsymbol{A},\boldsymbol{B}|\boldsymbol{W}) = &-\frac{1}{K}\sum_{i=1}^{K}(s_i\ln S_F + (1-s_i)\ln(1-S_F)) \\ &-\frac{\alpha}{K}\sum_{i=1}^{K}(s_i\ln S_G + (1-s_i)\ln(1-S_G)) + \beta\|\boldsymbol{\Theta}\|_F^2.\end{aligned} \tag{10}$$

where $S_F = sigmoid(<F(\boldsymbol{x}_i;\boldsymbol{\Theta}),\boldsymbol{a}_i>)$ and $S_G = sigmoid(<G(\boldsymbol{a}_i;\boldsymbol{\Theta}^T),\boldsymbol{x}_i>)$, $sigmoid(\cdot)$ is the sigmoid function, K is the number of sample pairs, s_i is the similarity between feature $\boldsymbol{x}_i$ and attribute $\boldsymbol{a}_i$, if $\boldsymbol{x}_i$ and $\boldsymbol{a}_i$ belong to the same category, $s_i = 1$, otherwise $s_i = 0$.

We build an end-to-end neural network as illustrated in Figure (2) to train the deep projection model. For feature projection function $F(\boldsymbol{x}_i;\boldsymbol{\Theta}))$, we utilize a simple network with two fully connected layers and add a $ReLU$ layer between

them. Because we know that input feature x has a dimension of d, we define the dimension of the following two layers as d and m. Thus, feature projection function $F(\cdot)$ is $d \rightarrow d(ReLU) \rightarrow m$. For attribute projection function $G(\boldsymbol{a}_i; \boldsymbol{\Theta}^T)$, we also use a similar architecture as $F(\cdot)$, but with a different layer dimension, which is $m \rightarrow d(ReLU) \rightarrow d$. In our model, the cross entropy is used to calculate the verification loss and similarity is binarized as $\{0, 1\}$; hence, we should constrain the result of the inner product to approximate $\{0, 1\}$. Consequently, we adhere a sigmoid layer after the inner product. This model is a typical fully connected neural network, and hence, the loss function 10 can be minimized by mini-batch back-propagation. The network parameter is updated by subtracting the gradient of $\mathcal{L}(\boldsymbol{X}, \boldsymbol{A}, \boldsymbol{B}|\boldsymbol{W})$ with respect to $\boldsymbol{W}$, which is often called the stochastic gradient descent (SGD) method. The SGD can be easily implemented in Tensorflow with several lines of codes, so that we do not make significant effort to describe it here.

3.3.2. Feature Verification

When there is a new test feature, $\hat{\boldsymbol{x}}$, we have two approaches for verifying whether the feature belongs to a certain category. One approach is that the verification is in attribute space $\mathcal{A}$. First, the feature is fed into the network and a corresponding embedding $F(\hat{\boldsymbol{x}}_i; \boldsymbol{\Theta})$ is generated, which is tested by using the inner product with all the attributes (for GZSL) or all the unseen attributes (for ZSL). The corresponding index of the largest value of inner products is its category. This computation can be represented as,

$$z(\hat{\boldsymbol{x}}_i) = \arg \max_{1 \leqslant c \leqslant C} < F(\hat{\boldsymbol{x}}_i; \boldsymbol{\Theta}), \boldsymbol{a}_c >, \tag{11}$$

where C is the total number of unseen classes (for ZSL) or all the classes (for GZSL). Alternatively, the other approach for verification in feature space $\mathcal{X}$ is the description of computation using the following equation:

$$z(\hat{\boldsymbol{x}}_i) = \arg \max_{1 \leqslant c \leqslant C} < G(\boldsymbol{a}_c; \boldsymbol{\Theta}^T), \hat{\boldsymbol{x}}_i > . \tag{12}$$

4. Experiments

In this section, we first provide a brief review of the selected datasets for the evaluation and then present the results for both ZSL and GZSL. Finally, we discuss some details of hyper-parameter α and the distribution of the projected features.

4.1. Datasets and Settings

4.1.1. Datasets

ZSL assumes that the training and testing sets are disjointed and the samples belong to the unseen classes will not appear in the training process. However, the training sets of the conventional split [19] contain many classes that appear in the ImageNet [6], which is used for training the deep feature extraction model. ImageNet includes 7 aPY, 6 AWA, 1 CUB, and 6 SUN test classes, which break the rules of the disjoint of the training and testing sets. Therefore, in our experiments, we choose to utilize the split strategy proposed by [40], which rearranges the train and test datasets and guarantees that no test class appears in the training set and ImageNet. The statistics of the split datasets are presented in Table (1). In our experiments, we also evaluate our ZSL method using these four well-known datasets. The datasets are described as follows:

(1) SUN (SUN attributes) [30] SUN is a fine-grained and medium-sized dataset that contains 14,340 images from 717 types of scenes. Among the total number of 717 classes, 1,440 samples of 72 classes are used as the unseen testing data, and the remaining 645 classes are divided into two parts: 10,320 seen training samples and 2,580 seen testing samples.

(2) CUB (Caltech-UCSD-Birds 200-2011) [38] CUB is also a fine-grained and medium-sized dataset that is composed of 11,788 images from 200 different categories of birds. In our experiments, 50 of the total 200 classes, including 2,967 images, are set as the unseen training set, and the remaining are set as the seen training set, which contains 7,057 seen training images and 1,764 seen testing images.

Table 1: Statistics of the four benchmark datasets used in our experiments

Dataset	# classes of seen/unseen	# images	# train seen	# test seen	# test unseen
SUN [30]	645/72	14,340	10,320	2,580	1,440
CUB [38]	150/50	11,788	7,057	1,764	2,967
AWA [7]	40/10	30,475	19,832	4,958	5,685
aPY [19]	20/12	15,339	5,932	1,483	7,924

(3) Animals with attributes (AWA) [7] AWA is a coarse-grained and medium-scale dataset that contains 30,475 images in 50 categories. A study [40] proposed a split strategy in which 40 classes were used for training, of which 19,832 images were set as seen the training set, 4,958 images were set as the seen test set, and remaining 10 classes of the 5,685 images were used for testing. We also followed this setting.

(4) Attribute Pascal and Yahoo) (aPY) [19] aPY is a coarse-grained and small-scale dataset that has 15,339 image instances from 32 classes. Among the 32 classes, in our experiments, 20 Pascal classes of 7,415 images are utilized for training and the remaining 12 Yahoo classes are utilized for testing. For the purpose of GZSL, the 20 Pascal classes are also divided into seen training set of 5,932 images and seen test set of 1,483 images.

4.1.2. Settings

Image features As reported many times, deep features outperform shallow features by a significant margin. Therefore, we only consider the deep features in a pre-trained model of a 101-layered ResNet, which extracts 2048-dimensional features from the top layer, except the classification layer.

Training pairs sampling In our experiments, we not only need similar pairs that include their features and corresponding attributes but also the dissimilar pairs that contain features and attributes belonging to different classes, *e.g.* A feature tells the type of *'Chimpanzee'* and an attribute belongs to the type of *'Chimpanzee'* constructing a similar pair. A feature of *'Chimpanzee'* and an attribute of *'Leopard'* form a dissimilar pair. In our deep model method, we set the input as three-tuple vector $(\boldsymbol{x}_i, \boldsymbol{a}_i, s_i)$, where if feature $\boldsymbol{x}_i$ and attribute $\boldsymbol{a}_i$

belong to the same class, then s_i is set as 1, otherwise it is set as 0. At each epoch, similar pairs are selected using all the features and their corresponding attributes; thus, there are N similar pairs. Dissimilar pairs are composed of all the features and randomly selected dissimilar attributes, which also form N dissimilar pairs; thus, we have $2N$ pairs in each epoch. In addition, at the beginning of each epoch, all the dissimilar and similar pairs are regenerated and the $2N$ input pairs are shuffled.

Hyper-parameters In our method, there are three hyper-parameters, namely, deep learning rate lr, balance coefficient α, and regularisation coefficient β. β is set as $\beta = 1 \times 10^{-4}$ in all our experiments. In ZSL test, we set $lr = 1 \times 10^{-4}$ for datasets SUN and CUB, $lr = 1 \times 10^{-5}$ for dataset AWA, and $lr = 1 \times 10^{-6}$ for dataset aPY. In GZSL, we use the same learning rates as those in ZSL. For balance coefficient α, we set $\alpha = 1$ and $\alpha = 10$ for verification in attribute space $\mathcal{A}$ and feature space $\mathcal{X}$, respectively, when testing ZSL, and $\alpha = 5$ and $\alpha = 0.1$ respectively for GZSL.

4.2. Results of Zero-Shot Learning

Image classification accuracy with a single label is generally evaluated with top-1 accuracy, *i.e.*, if the predicted label is same as the real label, then we the prediction is considered to be correct. In some conventional evaluation methods [48, 17], the ZSL accuracy is averaged for all the images, which leads to a bad scenario where a high performance on densely populated classes is promoted, *e.g.*, one of the unseen aPY classes, *'person'*, accounts for 64% of the total unseen samples. However, we are interested in achieving a high performance in all the classes, even in sparsely populated classes. Hence, we choose to use the average of each class accuracy [40], which can be described as follows:

$$acc_{\mathcal{S}} = \frac{1}{\|\mathcal{S}\|} \sum_{c=1}^{\|\mathcal{S}\|} \frac{\#\text{ correct predictions in } c}{\#\text{ samples in } c}, \tag{13}$$

where $\|\mathcal{S}\|$ is the number of test classes $\mathcal{S}$. In ZSL, we set $\mathcal{S} = \boldsymbol{Z}$, and the search space is based on $\boldsymbol{Z}$.

Table 2: Results of the accuracy tests of ZSL using four well-known datasets. We set $\alpha = 1$ and $\alpha = 10$ for verification in attribute space $\mathcal{A}$ and feature Space $\mathcal{X}$, respectively.

Method	**SUN**	**CUB**	**AWA**	**aPY**
DAP[19]	39:9	40.0	44.1	33.8
IAP[19]	19.4	24.0	35.9	36.6
CONSE[28]	38.8	34.3	45.6	26.9
CMT[35]	39:9	34.6	39.5	28.0
SSE[48]	51:5	43.9	60.1	34.0
LATEM[39]	55.3	49.3	55.1	35.2
ALE[1]	58.1	54.9	59.9	39.7
DEVISE[9]	56.5	52.0	54.2	39.8
SJE[2]	53.7	53.9	65.6	32.9
ESZSL[32]	54.5	53.9	58.2	38.3
SYNC[4]	56.3	55.6	54.0	23.9
SAE[17]	53.4	42.0	58.1	32.9
Ours($\mathcal{A}$)	56.5	50.1	**68.2**	39.4
Ours($\mathcal{X}$)	**62.4**	**57.8**	67.7	**41.2**

We compare our proposed method with 12 state-of-the-art methods using the above-mentioned four datasets and their corresponding attributes. The results are recorded in Table (2), in which parts of the results come from [40] directly. SAE is implemented by us according to the description of the original paper. From the results in Table (2), we can see that before our proposed method, the best performances with the four datasets are exhibited by ALE [1], SYNC [4], SJE [2], and DEVISE [9], respectively. Our method with verification in attribute space $\mathcal{A}$ outperforms the other 12 methods on dataset AWA and ranks second among all the methods listed in Table (2) with dataset SUN, just lower than ALE [1] by 1.6%. For dataset aPY, our method is lower than the best method, DEVISE [9] by 0.4%, and ranks in the third place. The worst performance of our method is on dataset CUB, and it is lower than the best method SYNC [4] by over 5%. However, with the verification in feature space $\mathcal{X}$, our method can outperform all the other 12 methods in all the four datasets, and the differences in our results and the strongest competitors range from 1.4% to 4.3%.

Moreover, from Table (2), we can find that the result of the verification in the attribute space is slightly worse than some of the previous methods and our method in the feature space. This phenomenon is caused by the hubness problem, i.e., a few unseen class prototypes become the nearest neighbors of many data points or hubs. Using the semantic space as the embedding space implies that the visual feature vectors need to be projected onto the semantic space, which will reduce the variance in the projected data points, and thus, aggravate the hubness problem.

4.3. Results of Generalized Zero-Shot Learning

Until now, we have obtained the test accuracy of ZSL, but in real-world applications, we typically do not know whether a new image belongs to a seen class or an unseen class. Hence, in GZSL, the search space for evaluating a novel image is expanded to both test classes and train classes, which is more realistic. Furthermore, to remove the unbalanced situation of seen and unseen tests, we avoid utilizing the arithmetic mean and instead use the harmonic

Table 3: Results of the generalized zero-Shot learning with the four well-known attribute datasets. For harmonic accuracy, our method with a verification in the attribute space outperforms all the other 13 methods (CMT*: CMT with a novelty detection) with all the datasets, except SUN. The method verification in the feature space exceeds all the other methods with all the four datasets. We set $\alpha = 5$ and $\alpha = 0.1$ for verification in attribute space $\mathcal{A}$ and feature space $\mathcal{X}$, respectively.

Method	**SUN**			**CUB**			**AWA**			**aPY**		
	ts	**tr**	**H**	**ts**	**tr**	**H**	**ts**	**tr**	**H**	**ts**	**tr**	**H**
DAP [19]	4.2	25.1	7.2	1.7	67.9	3.3	0.0	**88.7**	0.0	4.8	78.3	9.0
IAP [19]	1.0	37.8	1.8	0.2	72.8	0.4	2.1	78.2	4.1	5.7	65.6	10.4
CONSE [28]	6.8	39.9	11.6	1.6	72.2	3.1	0.4	88.6	0.8	0.0	**91.2**	0.0
CMT [35]	8.1	21.8	11.8	7.2	49.8	12.6	0.9	87.6	1.8	1.4	85.2	2.8
CMT* [35]	8.7	28.0	13.3	4.7	60.1	8.7	8.4	86.9	15.3	10.9	74.2	19.0
SSE [48]	2.1	36.4	4.0	8.5	46.9	14.4	7.0	80.5	12.9	0.2	78.9	0.4
LATEM [39]	14.7	28.8	19.5	15.2	57.3	24.0	7.3	71.7	13.3	0.1	73.0	0.2
ALE [1]	21.8	33.1	26.3	23.7	62.8	34.4	16.8	76.1	27.5	4.6	73.7	8.7
DEVISE [9]	16.9	27.4	20.9	23.8	53.0	32.8	13.4	68.7	22.4	4.9	76.9	9.2
SJE [2]	14.7	30.5	19.8	23.5	59.2	33.6	11.3	74.6	19.6	3.7	55.7	6.9
ESZSL [32]	11.0	27.9	15.8	12.6	63.8	21.0	6.6	75.6	12.1	2.4	70.1	4.6
SYNC [4]	7.9	43.3	13.4	11.5	70.9	19.8	8.9	87.3	16.2	7.4	66.3	13.3
SAE [17]	17.1	28.1	21.3	17.4	50.7	25.9	11.0	83.8	19.5	6.7	59.6	12.1
Ours($\mathcal{A}$)	20.8	31.0	24.9	**29.0**	58.6	**38.8**	34.7	77.6	48.0	**24.5**	56.1	**34.1**
Ours($\mathcal{X}$)	**25.3**	34.6	**29.2**	26.2	55.1	35.5	**34.9**	73.4	**48.5**	13.7	72.2	23.1

accuracy computed from the training and testing accuracy, following the setting of [40].

$$H = \frac{2 \times acc_{tr} \times acc_{ts}}{acc_{tr} + acc_{ts}}, \tag{14}$$

where acc_{tr} and acc_{ts} are the accuracies of the test seen features and test unseen features, respectively, with all the classes. acc_{tr} and acc_{ts} are computed using equation (13), and the search space is set as $\boldsymbol{Y} \cup \boldsymbol{Z}$. $\mathcal{S} = \boldsymbol{Y}$ and $\mathcal{S} = \boldsymbol{Z}$ are executed when calculating acc_{tr} and acc_{ts}, respectively. The results are recorded and listed in Table (3), where parts of the results are directly cited from [40] and the SAE implemented by us is according to the description in its original paper [17].

In Table (3), the results of acc_{ts} are significantly lower than the results listed in Table (2) because the seen classes are included in the search space. This extends the search space and makes it difficult for a feature to find its corresponding class. From Table (3), we can note that for harmonic accuracy, our method utilizing the verification in the attribute space yields the best performance in comparison with the 13 state-of-the-art methods with datasets CUB, AWA, and aPY. The best result is with AWA, when our method can exceed the strongest competitor, ALE by over 20%. The smallest achievement is with dataset CUB, which can also surpass the best method, ALE by 4.4%. The only failure result is with dataset SUN, but its performance is only below ALE [1]. The best method is obtained with dataset SUN, by only 1.4%, which may be because the total class number of SUN is 717, which is much larger than the attribute dimension of 102 , and so, leads to a bad extension for the orthogonal projection. For the results of ts, our method yields similar results to H: it outperforms other methods with datasets AWA, CUB, and aPY, and ranks in the second place with dataset SUN, where our result is only 1% less than the best method, ALE. For the results of tr, our method does not emerge as the method with any of the four datasets, but it exhibits a good performance for ts and H, which implies that some of these methods that have a high tr but low ts are over-fitting in the seen classes and resulting in the problem of domain shift.

Regarding the strategy of verification in the feature space in our method, the results show that our method outperforms all the other 13 methods for harmonic accuracy H and test accuracy ts with all the four datasets. Because the largest category number is 717, which is much smaller than the feature dimension of 2048 in our experiments, there is sufficient dimensionality to construct an orthogonal space. With all the four datasets, the difference in the harmonic accuracy between our results and the strongest competitor ranges from 1.1% to 21%, with the smallest value being with CUB and the largest being with AWA.

4.4. Detailed Analysis

4.4.1. Network depth

In this section, we discuss the effect of the network depth on the accuracy of ZSL with the four datasets. In our experiment, we use the same settings as those listed in Table (2) and select four different depths, which are $layers = \{1, 2, 3, 4\}$. The results are shown in Figure (3). From the figure, we can see that the model with two layers performs best with all the four datasets. The one-layer model ranks the second place, but when the depth is more than 2, its performance decreases rapidly, particularly when verifying in $\mathcal{X}$. This phenomenon reveals that the one-layer model is slightly under-fitting, whereas the multi-layer (more than 2) models lead to over-fitting. Briefly, the best model has two layers, which is the model we have selected to study in this work.

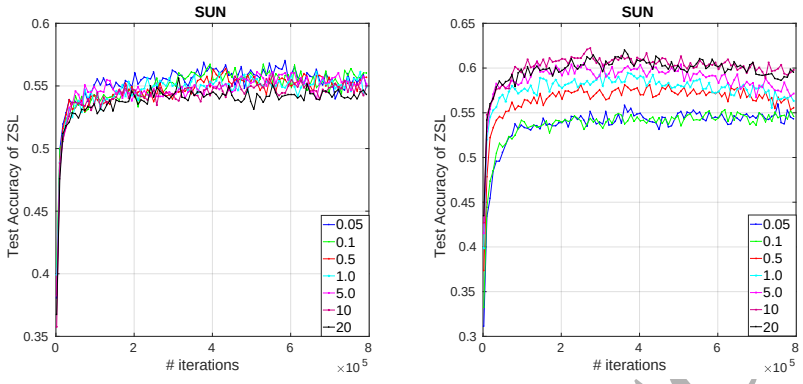
4.4.2. Hyper-parameters

Our method has two hyper-parameters, α and β. β controls the regularization item of $\mathbf{\Theta}$ and is usually set a small value, *e.g.*, $\beta = 1 \times 10^{-4}$. α is a balance coefficient, which adjusts the importance of the verifications in the attribute and feature spaces. To determine the extent of the effect of this parameter on the performance, we set the iteration time as 8×10^5 and $\alpha = \{0.05, 0.1, 0.5, 1, 5, 10, 20\}$, respectively. We utilize these settings with dataset SUN and present the corresponding curves of the ZSL test accuracy in Figure (4), unseen test accuracy in Figure (5), seen test accuracy in Figure (6), and harmonic accuracy in Figure

(a) SUN (b) CUB

(c) AWA (d) aPY

Figure 3: ZSL results of our model with different network depths.



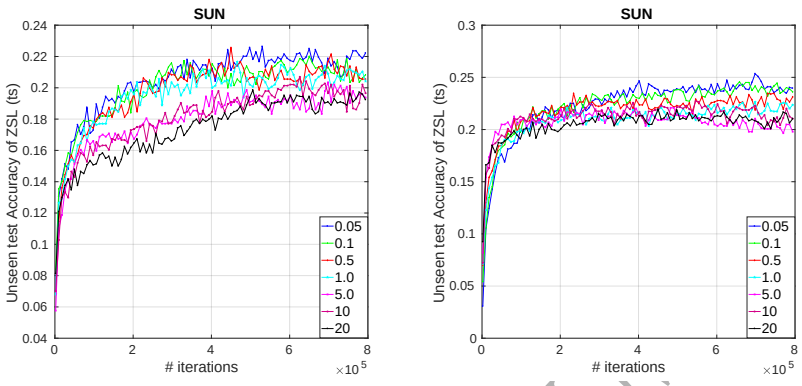
(a) Verifying in attribute space $\mathcal{A}$ (b) Verifying in feature space $\mathcal{X}$

Figure 4: Test accuracy of ZSL with different α

(7). In addition to these curves, we also list the maximum values of the test accuracy of both the ZSL and GZSL and their corresponding ts and tr in Table (4) for all the four datasets.

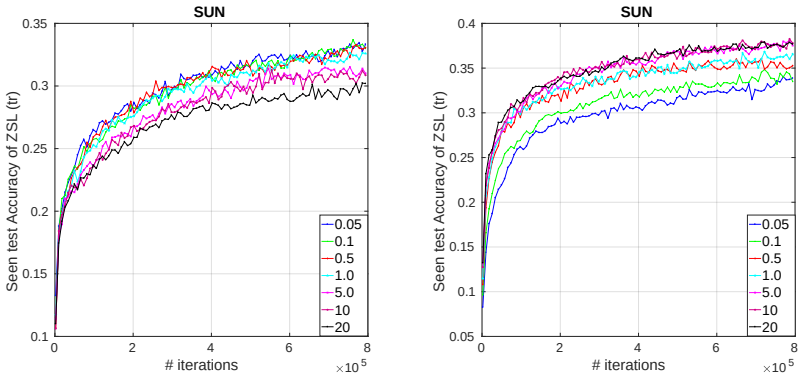
When verifying in the attribute spaces using dataset SUN, we note from figure (4) that a smaller α implies a higher accuracy of the ZSL, whereas the trend is opposite when verifying in the feature space. This phenomenon indicates that when we verify the results in the attribute space, the first term in equation (11) is more important, whereas the second term in equation (11) when verifying in the feature space. For the test accuracy of GZSL, we obtain trends different from the ZSL results. In figure (5), we see that a smaller α leads to a higher accuracy of ts of GZSL when verifying in both the attribute and feature spaces, which implies that the first term in equation (11) is more important than the second term for ts with the SUN dataset.

In figure (6) presenting seen test accuracy tr, we observe that with dataset SUN, the best results for verification in the feature space are much better than that for verification in the attribute space. We know that the attribute dimensionality for dataset SUN is 102 but the category number is 645; therefore, when conducting verification in the attribute space, the dimension is insufficient to



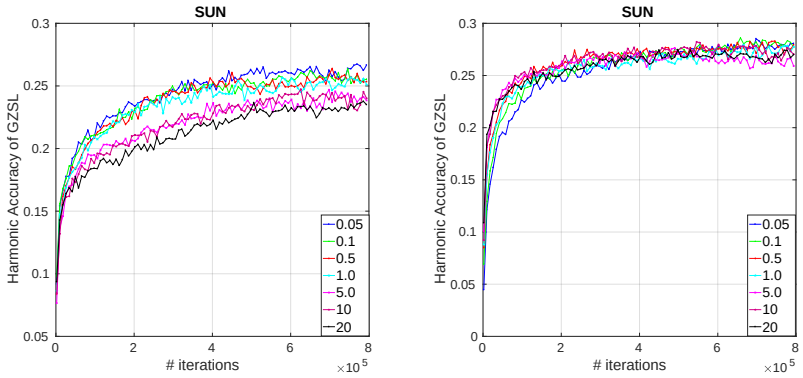
(a) Verifying in attribute space $\mathcal{A}$ (b) Verifying in feature space $\mathcal{X}$

Figure 5: Unseen test accuracy ts of GZSL with different α.



(a) Verifying in attribute space $\mathcal{A}$ (b) Verifying in feature space $\mathcal{X}$

Figure 6: Seen test accuracy tr of GZSL with different α



(a) Verifying in attribute space $\mathcal{A}$ (b) Verifying in feature space $\mathcal{X}$

Figure 7: Harmonic accuracy H of GZSL with different α

project all the classes onto the orthogonal space. However, when verifying in the feature space, the feature space has 2048 dimensions, which is much larger than the category number. Therefore, we can conclude that in our method, the dimension of the verification space plays an important role in improving the performance of tr. The accuracy curves of H are illustrated in Figure (7). We obtain similar results with different α when verifying in the feature space because these results are a combination of ts and tr.

The maximum values of ZSL and GZSL are also presented in Table (4), and they show that typically, ts of ZSL is inconsistent with ts of GZSL. The results with datasets SUN and CUB are same with a smaller α in the attribute space or larger α in the feature space leading to a better performance. In comparison, with dataset AWA, when testing ZSL, a larger α implies a higher ts, and but the effect is opposite when testing GZSL. The difference in the performances with these datasets is caused by the differences in the attribute structure, dimensionality of the attributes, and categories of the samples.

In addition, we also show the results of optimization with verification in only one space, i.e., we optimize the equation (10) by discarding the first or second term. Table (5) lists the accuracy of both ZSL and GZSL using the four datasets.

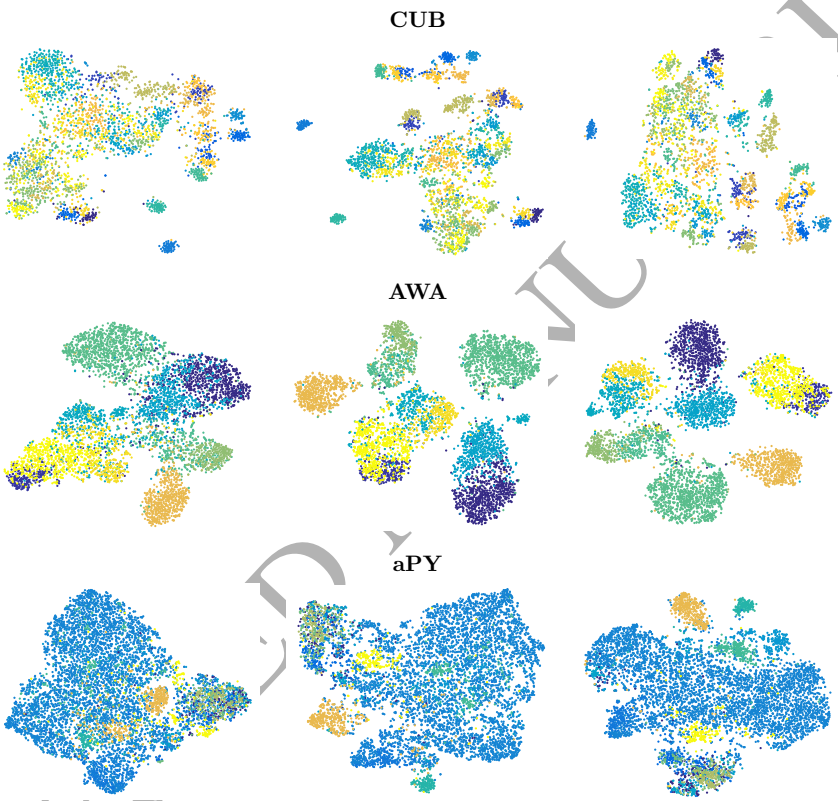
Table 4: Maximum values of test accuracy ts of ZSL and harmonic accuracy H of GZSL.

DataSet	Space	ZSL		GZSL			
		ts	α	ts	tr	H	α
SUN	$\mathcal{A}$	57.2	0.5	22.6	33.1	26.9	0.05
	$\mathcal{X}$	62.4	10	25.3	34.6	29.2	10
CUB	$\mathcal{A}$	51.3	5	29.6	57.2	39.0	0.05
	$\mathcal{X}$	58.5	20	25.8	63.1	36.7	20
AWA	$\mathcal{A}$	69.4	0.05	36.3	77.7	49.5	10
	$\mathcal{X}$	67.7	10	38.8	77.4	51.7	0.5
APY	$\mathcal{A}$	39.8	20	24.5	56.1	34.1	5
	$\mathcal{X}$	41.2	10	16.0	66.0	25.8	0.05

We observe three phenomena based on this table. First, mostly, the verification in $\mathcal{X}$ outperforms the verification in $\mathcal{A}$, which affirms the occurrence of the hubness problem [47] again. Second, we can see that both the results in the single verification space are worse than the results presented in Table (2), which demonstrates the effectiveness of the proposed DVN. Third, in this experiment, we also compute the results in $\mathcal{X}$ while training with $\mathcal{A}$ and the results in $\mathcal{A}$ but training with $\mathcal{X}$. These results reveal that the performance is better when the training and testing are in the same space than when they are in different spaces.

4.4.3. Distribution of projected features

To better demonstrate the performance of our method, it is necessary to show the distributions of the projected features or attributes. Because the attributes are class level, there is no need to show the projected attributes in the feature space. Thus, in this experiment, we only show the distribution of the projected features in the attribute space. Concurrently, we also present the results of two baseline methods DAP [19] and SAE [17]. The distribution figures drawn with t-sne [26] are displayed in Figure (8). Because dataset SUN has 72 classes of testing samples, which will make recognition difficult for humans, we discard



(a) **DAP** [19] (b) **SAE** [17] (c) **Ours**

Figure 8: Distribution of the projected unseen features in the attribute space using three selected datasets

Table 5: Results of the optimization with verification in only one space.

Dataset	Space	Optimisation in $\mathcal{A}$ only				Optimisation in $\mathcal{X}$ only			
		ZSL	GZSL			ZSL	GZSL		
		ts	ts	tr	H	ts	ts	tr	H
SUN	$\mathcal{A}$	55.1	18.3	33.8	23.7	52.8	12.8	22.6	16.4
	$\mathcal{X}$	37.1	17.2	24.6	20.3	58.8	17.8	37.2	24.1
CUB	$\mathcal{A}$	49.8	18.0	64.8	28.2	50.4	20.0	46.9	28.1
	$\mathcal{X}$	24.7	11.0	18.9	13.9	56.7	25.5	62.7	36.3
AWA	$\mathcal{A}$	67.2	17.8	67.1	28.1	60.2	17.0	80.0	28.0
	$\mathcal{X}$	59.4	12.4	81.4	21.5	67.6	25.0	87.0	38.0
aPY	$\mathcal{A}$	37.6	8.7	56.4	15.1	35.8	7.2	56.1	12.8
	$\mathcal{X}$	36.3	2.2	83.4	4.2	36.3	16.1	77.2	26.6

this set and utilize the remaining three datasets. Figure (8) reveals that the distribution of the projected features generated by our method is easier to be classified, *e.g.*, the points belonging to the same category cluster are much closer than those generated by other methods, particularly using datasets AWA and aPY. This implies that the projected features of the same class generated by our method are easier to be classified with the same label than those obtained by the other two methods.

5. Conclusion

In this paper, we proposed a new method, namely, dual-verification network for zero-shot learning. Our method constructs an orthogonal projection from the feature space to the attribute space, where all the projected vectors have maximum correlation with these attributes in the same categories and are orthogonal to those from different classes. Furthermore, in this method, the feature semantic representation is adopted to learn the relationship between the semantic features and class labels. Through this representation, the attributes can be mapped to the feature space and should be orthogonal to the correspond-

ing features. In addition, to optimize these two verifications simultaneously, we introduced a deep network, which utilizes the cross entropy loss as its objective function. Extensive experiments for ZSL and GZSL were performed with four popular datasets, and the results show that our method outperforms all the current competitive methods. Detailed analysis also shows the effect of the hyper-parameters on the performance.

6. Acknowledgement

This work was supported in part by National Natural Science Foundation of China (No.61872187, No.61871444, No.61773215) and Major Special Project of Core Electronic Devices, High-end Generic Chips and Basic Software (No.2015ZX01041101).

References

References

[1] Akata, Z., Perronnin, F., Harchaoui, Z., Schmid, C., 2016. Label-embedding for image classification. IEEE transactions on pattern analysis and machine intelligence 38, 1425–1438.

[2] Akata, Z., Reed, S., Walter, D., Lee, H., Schiele, B., 2015. Evaluation of output embeddings for fine-grained image classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2927–2936.

[3] Al-Halah, Z., Stiefelhagen, R., 2017. Automatic discovery, association estimation and learning of semantic attributes for a thousand categories, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[4] Changpinyo, S., Chao, W.L., Gong, B., Sha, F., 2016. Synthesized classifiers for zero-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5327–5336.

[5] Demirel, B., Cinbis, R.G., Cinbis, N.I., 2017. Attributes2classname: A discriminative model for attribute-based unsupervised zero-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[6] Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. Imagenet: A large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE. pp. 248–255.

[7] Farhadi, A., Endres, I., Hoiem, D., Forsyth, D., 2009. Describing objects by their attributes, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE. pp. 1778–1785.

[8] Ferrari, V., Zisserman, A., 2008. Learning visual attributes, in: Advances in Neural Information Processing Systems, pp. 433–440.

[9] Frome, A., Corrado, G.S., Shlens, J., Bengio, S., Dean, J., Mikolov, T., et al., 2013. Devise: A deep visual-semantic embedding model, in: Advances in neural information processing systems, pp. 2121–2129.

[10] Fu, Y., Hospedales, T.M., Xiang, T., Fu, Z., Gong, S., 2014. Transductive multi-view embedding for zero-shot recognition and annotation, in: European Conference on Computer Vision, Springer. pp. 584–599.

[11] Guo, Y., Ding, G., Jin, X., Wang, J., 2016. Transductive zero-shot recognition via shared model space learning, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, pp. 3494–5000.

[12] Hong, C., Yu, J., Wan, J., Tao, D., Wang, M., 2015. Multimodal deep autoencoder for human pose recovery. IEEE Transactions on Image Processing 24, 5659–5670.

[13] Huang, S., Elhoseiny, M., Elgammal, A., Yang, D., 2015. Learning hypergraph-regularized attribute predictors, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 409–417.

[14] Ji, Z., Yu, Y., Pang, Y., Guo, J., Zhang, Z., 2017. Manifold regularized cross-modal embedding for zero-shot learning. Information Sciences 378, 48–58.

[15] Kankuekul, P., Kawewong, A., Tangruamsub, S., Hasegawa, O., 2012. Online incremental attribute-based zero-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE. pp. 3657–3664.

[16] Kodirov, E., Xiang, T., Fu, Z., Gong, S., 2015. Unsupervised domain adaptation for zero-shot learning, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 2452–2460.

[17] Kodirov, E., Xiang, T., Gong, S., 2017. Semantic autoencoder for zero-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[18] Lampert, C.H., Nickisch, H., Harmeling, S., 2009. Learning to detect unseen object classes by between-class attribute transfer, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, IEEE. pp. 951–958.

[19] Lampert, C.H., Nickisch, H., Harmeling, S., 2014. Attribute-based classification for zero-shot visual object categorization. IEEE Transactions on Pattern Analysis and Machine Intelligence 36, 453–465.

[20] Li, X., Guo, Y., Schuurmans, D., 2015. Semi-supervised zero-shot classification with label representation learning, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 4211–4219.

[21] Li, Y., Wang, D., Hu, H., Lin, Y., Zhuang, Y., 2017. Zero-shot recognition using dual visual-semantic mapping paths, in: Proceedings of the International Conference on Computer Vision.

[22] Long, Y., Liu, L., Shao, L., 2016. Attribute embedding with visual-semantic ambiguity removal for zero-shot learning, in: Proceedings of the British Machine Vision Conference.

[23] Long, Y., Liu, L., Shao, L., Shen, F., Ding, G., Han, J., 2017. From zero-shot learning to conventional supervised classification: Unseen visual data synthesis, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[24] Long, Y., Shao, L., 2017. Describing unseen classes by exemplars: Zero-shot learning using grouped simile ensemble, in: Proceedings of the IEEE Winter Conference on Applications of Computer Vision, IEEE. pp. 907–915.

[25] Lu, J., Li, J., Yan, Z., Zhang, C., 2017. Zero-shot learning by generating pseudo feature representations, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[26] Maaten, L.v.d., Hinton, G., 2008. Visualizing data using t-sne. Journal of Machine Learning Research 9, 2579–2605.

[27] Morgado, P., Vasconcelos, N., 2017. Semantically consistent regularization for zero-shot recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[28] Norouzi, M., Mikolov, T., Bengio, S., Singer, Y., Shlens, J., Frome, A., Corrado, G.S., Dean, J., 2014. Zero-shot learning by convex combination of semantic embeddings, in: Internation conference on Learning Representation(ICLR).

[29] Palatucci, M., Pomerleau, D., Hinton, G.E., Mitchell, T.M., 2009. Zero-shot learning with semantic output codes, in: Advances in neural information processing systems, pp. 1410–1418.

[30] Patterson, G., Xu, C., Su, H., Hays, J., 2014. The sun attribute database: Beyond categories for deeper scene understanding. International Journal of Computer Vision 108, 59–81.

[31] Reed, S., Akata, Z., Lee, H., Schiele, B., 2016. Learning deep representations of fine-grained visual descriptions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 49–58.

[32] Romera-Paredes, B., Torr, P., 2015. An embarrassingly simple approach to zero-shot learning, in: International Conference on Machine Learning, pp. 2152–2161.

[33] Shao, L., Cai, Z., Liu, L., Lu, K., 2017. Performance evaluation of deep feature learning for rgb-d image/video classification. Information Sciences 385, 266–283.

[34] Sharan, R.V., Moir, T.J., 2017. Robust acoustic event classification using deep neural networks. Information Sciences 396, 24–32.

[35] Socher, R., Ganjoo, M., Manning, C.D., Ng, A., 2013. Zero-shot learning through cross-modal transfer, in: Advances in neural information processing systems(NIPS), pp. 935–943.

[36] Wang, D., Li, Y., Lin, Y., Zhuang, Y., 2016a. Relational knowledge transfer for zero-shot learning, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence.

[37] Wang, Y., Wang, X., Liu, W., 2016b. Unsupervised local deep feature for image recognition. Information Sciences 351, 67–75.

[38] Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., Perona, P., 2010. Caltech-ucsd birds 200 .

[39] Xian, Y., Akata, Z., Sharma, G., Nguyen, Q., Hein, M., Schiele, B., 2016. Latent embeddings for zero-shot classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 69–77.

[40] Xian, Y., Schiele, B., Akata, Z., 2017. Zero-shot learning-the good, the bad and the ugly, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[41] Yu, Z., Li, T., Luo, G., Fujita, H., Yu, N., Pan, Y., 2017a. Convolutional networks with cross-layer neurons for image recognition. Information Sciences 433–434, 241–254.

[42] Yu, Z., Yu, J., Fan, J., Tao, D., 2017b. Multi-modal factorized bilinear pooling with co-attention learning for visual question answering, in: Proceedings of the International Conference on Computer Vision.

[43] Yu, Z., Yu, J., Xiang, C., Fan, J., Tao, D., 2018. Beyond bilinear: Generalized multi-modal factorized high-order pooling for visual question answering. IEEE Transactions on Neural Networks and Learning Systems doi:10.1109/TNNLS.2018.2817340.

[44] Yuchen Guo, Guiguang Ding, J.H.Y.G., 2017. Synthesizing samples for zero-shot learning, in: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, pp. 1774–1780.

[45] Zhang, H., Liu, L., Long, Y., Shao, L., 2018a. Unsupervised deep hashing with pseudo labels for scalable image retrieval. IEEE Transactions on Image Processing 27, 1626–1638.

[46] Zhang, H., Long, Y., Shao, L., 2018b. Zero-shot hashing with orthogonal projection for image retrieval. Pattern Recognition Letters doi:doi.org/10.1016/j.patrec.2018.04.011.

[47] Zhang, L., Xiang, T., Gong, S., 2017. Learning a deep embedding model for zero-shot learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[48] Zhang, Z., Saligrama, V., 2015. Zero-shot learning via semantic similarity embedding, in: Proceedings of the IEEE International Conference on Computer Vision, pp. 4166–4174.

[49] Zhu, Y., Chen, Y., Lu, Z., Pan, S.J., Xue, G.R., Yu, Y., Yang, Q., 2011a. Heterogeneous transfer learning for image classification., in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence.

[50] Zhu, Y., Chen, Y., Lu, Z., Pan, S.J., Xue, G.R., Yu, Y., Yang, Q., 2011b. Heterogeneous transfer learning for image classification., in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence.