A Probabilistic Zero Shot Learning Method via Latent Nonnegative Prototype Synthesis of Unseen Classes

Haofeng Zhang, Huaqi Mao, Yang Long, Wankou Yang, and Ling Shao, *Senior Member, IEEE*

***Abstract*—Zero Shot Learning (ZSL), a type of structured multi-output learning, has attracted much attention due to its requirement of no training data for target classes. Conventional ZSL methods usually project visual features into semantic space and assign labels by finding their nearest prototypes. However, this type of Nearest Neighbor Search (NNS) based methods often suffer from great performance degradation because of the non-uniform variances between different categories. In this paper, we propose a probabilistic framework by taking covariance into account to deal with the problem mentioned above. In this framework, we define a new latent space, which has two characteristics. The first is the features in this space should gather within classes and scatter between classes, which is implemented by triplet learning, the second is the prototypes of unseen classes are synthesized with nonnegative coefficients which are generated by Nonnegative Matrix Factorization (NMF) of relations between the seen classes and unseen classes in attribute space. During training, the learned parameters are the projection model for triplet network and the nonnegative coefficients between unseen classes and seen classes. In the testing phase, visual features are projected into latent space and assigned with the labels that have the maximum probability among unseen classes for classic ZSL or within all classes for Generalized ZSL. Extensive experiments are conducted on four popular datasets, and the results show that the proposed method can outperform the state-of-the-art methods in most circumstances.**

***Index Terms*—Nonnegative Matrix Factorization (NMF), Triplet Network, Zero Shot Learning (ZSL), Prototype Synthesis.**

I. Introduction

RECENT efforts on image classification research have been focusing on large-scale image recognition issues, such as the challenge on ImageNet [1]. Since the latest Convolutional Neural Network (CNN) based deep learning methods have achieved significant improvement and reached the accuracy of over 95% [2, 3], a question is raised that are we already able to solve large-scale classification problems. This problem can be answered based on two conditions: 1) Can the training set include all classes over the world and contain enough training samples? 2) Can the training model be extended without retraining to the classes which are not included in the training set and still retain high performance? The first condition is almost impossible to be reached because there are 8.7 million classes only in animal species [4]. Therefore, many researchers try to fulfill the second requirement with Transfer Learning [5] and Zero Shot Learning (ZSL) [6, 7].

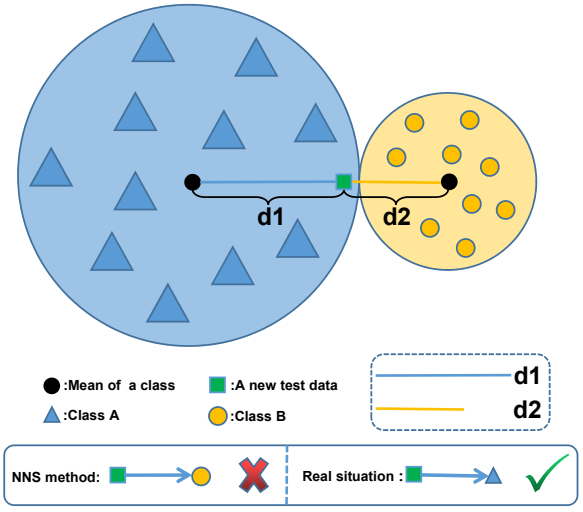


Fig. 1. Illustration of the drawback of NNS based method: the new input data which should belong to class A is misclassified to class B by NNS based methods due to the shorter distance to B.

ZSL aims to recognize the categories which have no labeled data available during training. This is usually realized by introducing auxiliary semantic information, such as attribute vectors [8] and word embeddings [9], which are often used as the prototypes of ZSL for final classification. To this end, ZSL usually learns to generate structured vectors in attribute space, which are then used to find the best prototypes as the test samples' labels. From this perspective, ZSL can be regarded as a type of Structured Multi-output Learning (SML). During training, the relationship between visual features and semantic attributes is learned with only the data from seen classes. The prediction is conducted by directly applying the learned model on the data of unseen classes. The most popular ZSL methods are based on Nearest Neighbor Search (NNS), which projects visual feature into attribute space and find the nearest attribute

Manuscript received 00-00, 2018; revised 00-00, 2018. (*Corresponding author: Haofeng Zhang*)

This work was supported in part by National Science Foundation of China under Grants 61872187 and 61929104, and in part by Medical Research Council (MRC) Innovation Fellowship under Grant MR/S003916/1.

Haofeng Zhang and Huaqi Mao are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, 210094, China. E-mail: {zhanghf, maohuaqi}@njust.edu.cn

Yang Long is with the Department of Computer Science, Durham University, Durham, UK. E-mail:yang.long@ieee.org

Wankou Yang is with the School of Automation, Southeast University, Nanjing 210018, China. E-mail:wkyang@seu.edu.cn

Ling Shao is with the Inception Institute of Artificial Intelligence (IIAI), Abu Dhabi, United Arab Emirates, and also with the University of Chinese Academy of Sciences, Beijing, China. E-mail: ling.shao@ieee.org

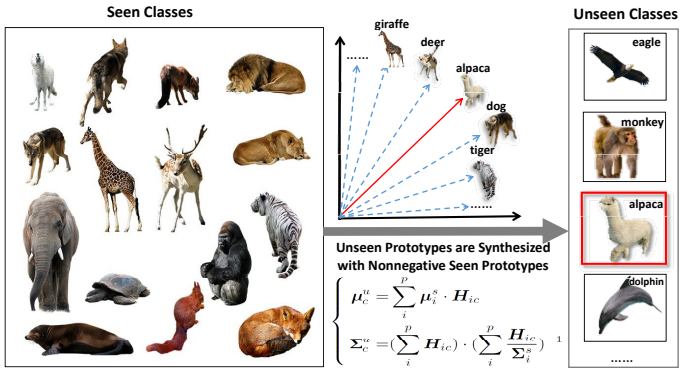


Fig. 2. Illustration of the prototype synthesis in latent space. Each prototype of unseen classes, including the mean and the variance, is synthesized by nonnegative prototypes of the seen classes. The middle bottom part is the method for synthesis.

vector as its label, or projects attribute prototypes into feature space as the feature prototypes which are then exploited for comparing with input visual features to find their class labels.

However, these NNS based methods often suffer from great performance degradation due to the neglect of variances between different classes. For example, in Fig. 1, there are two classes (denoted as blue and yellow) which have different variances, the new input feature (illustrated as a green square) is closer to the yellow class than the blue one. In the conventional NNS based methods, this new data should be classified as the yellow class, but in fact, it should belong to the blue class because it has a higher probability to be assigned with the blue label than the yellow one by considering the class variances.

Verma *et al.* first proposed a simple exponential framework for ZSL [10, 11], which treats each class distribution as an exponential family distribution, and the mean and the variance of each seen/unseen category are defined as a linear projection of the corresponding predefined class attribute. The projection is learned with only the seen classes and can be exploited to predict the parameters of the class-conditional distribution of each unseen class by utilizing the attributes of unseen classes. Then, the classification of the input unseen data can be conducted by finding the maximum probability in visual space. Wang *et al.* proposed an end-to-end Variational AutoEncoder (VAE) network based ZSL method (VZSL) [12], which represents each seen/unseen class using a class-specific latent-space distribution conditioned on class attributes. These latent-space distributions are utilized as a prior for a supervised VAE, which can facilitate to learn discriminative feature representations for the inputs. Although these two methods with the probabilistic framework can achieve significant improvement comparing to traditional NNS based methods, there still exists some problems that may hinder them to obtain better results:

1) This framework defines a linear projection from attribute space to feature space to fit the prototypes, which can be denoted as $\boldsymbol{\mu} = \boldsymbol{W}_{\mu}\boldsymbol{a}_s$ and $\boldsymbol{\sigma}^2 = \boldsymbol{W}_{\sigma^2}\boldsymbol{a}_s$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ are the mean and the variance of seen classes in visual feature space respectively, and they can also be considered as the prototypes of the seen classes. $\boldsymbol{a}_s$ is the corresponding class level attribute, and $\boldsymbol{W}_{\mu}$ and $\boldsymbol{W}_{\sigma^2}$ are its projection parameters, and directly calculated by optimizing the Least Square Error (LSE) loss, that is to say, $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ are weighted linear combination of attributes. However, there is no constraint for $\boldsymbol{W}_{\mu}$ and $\boldsymbol{W}_{\sigma^2}$, since direct optimization with LSE might lead to negative values for $\boldsymbol{W}_{\mu}$ and $\boldsymbol{W}_{\sigma^2}$, *i.e.*, the synthesized unseen prototypes sometimes may come from negative attribute, which is unreasonable for realistic scenarios. Besides, no constraint for the projection parameters also leads to over-fitting, which obtains good results on training data, but leads to bad performance on test data.

2) In the equation $\boldsymbol{\sigma}^2 = \boldsymbol{W}_{\sigma^2}\boldsymbol{a}_s$, $\boldsymbol{W}_{\sigma^2}$ is a matrix, and $\boldsymbol{a}_s$ is a vector, so the result $\boldsymbol{\sigma}^2$ should be a vector. Thus, the covariance of the synthesized prototype can only be considered as $diag(\sigma_1^2, \cdots, \sigma_{d_x}^2)$, which means that the entries of the whole matrix except the diagonal are all zeros. It is known that the diagonal matrix assumes that the elements of the features are independent of each other, which is a strong assumption and ignores the correlation between attributes. VZSL [12] uses a nonlinear deep network to generate a diagonal covariance matrix, but the problem remains unresolved.

3) As it is known that the distribution of seen data is usually different from that of unseen data, thus using the model generated with only seen data to approximate the unseen data often leads to domain shift problem [13]. The methods proposed by Verma *et al.* [10] and Wang *et al.* [12] utilize only the seen data to compute the projection matrix or network parameters and apply them on unseen prototype synthesis, which will inevitably cause that problem.

4) Verma *et al.* [10] directly used the original visual features to compute the prototypes of seen classes. However, the distribution of these features often overlapped between different classes, which will be verified in the final experiments, *i.e.*, many instances may have high probabilities to two or more classes, which will easily lead to the wrong classification. The method VZSL [12] also has such a problem. Therefore, to reduce the overlapped area of the data, it is necessary to find another space, where the processed samples are more discriminative.

To deal with these problems, in this paper, we propose a novel probabilistic method with Latent Nonnegative Prototypes Synthesis (LNPS) for unseen classes. In this method, we make three great efforts, the first one is the definition of a latent space, where the original features are projected into it by triplet learning, which can make the projected data in this space has less overlapped areas, and to be more discriminative, this effort can well solve the fourth problem mentioned above. Second, to mitigate the domain shift problem, we design a nonnegative combination model to synthesize the unseen prototypes from the seen data, and the nonnegative coefficients are generated with the relationship between the attributes of seen classes and unseen classes, which is illustrated in Fig. 2. For example, the prototype of *'alpaca'*, including the mean and the variance, is synthesized with the prototypes of *'giraffe'*, *'deer'*, *'tiger'* and so on. This effort has built connections between the seen classes and the unseen classes, which can well alleviate the domain shift problem, and improve the performance on the more realistic Generalized ZSL (GZSL) setting significantly. Third, we make the prototype of each

category in the latent space as a Gaussian distribution, which consists of a mean vector and a variation matrix, and find the maximum probability of a test sample with respect to all the synthesized prototypes. Such effort constrains the prediction within a probabilistic framework in the latent space, thus can circumvent the problem caused by the NNS when class distributions are overlapping.

Our contributions are briefly listed as follows:

- We design a latent space for zero shot classification with triplet network. In this latent space, features belonging to the same category are gathered together, otherwise they are scattered among the classes, which can greatly reduce the overlapped areas, and make the projected data more discriminative.
- We build relationships between the seen classes and the unseen classes to solve the domain shift problem. In this manipulation, Nonnegative Matrix Factorization (NMF) is utilized to learn the nonnegative coefficients to generate the unseen attributes from the seen attributes, and then the coefficients are used to synthesize unseen prototypes from the seen prototypes in latent space.
- Extensive experiments are conducted on four popular datasets for both ZSL and GZSL, the results show that the proposed method can outperform the state-of-the-art methods in most circumstances, especially on the more realistic GZSL setting.

The main content of this paper is organized as follows: In section II we briefly introduce the existing methods for ZSL. Section III describes the proposed method in detail. Section IV gives the experimental results of comparison with existing methods for both conventional ZSL and GZSL. Finally in section V, we conclude this paper.

II. Related Work

According to the usage of unseen data during training, ZSL methods can be coarsely divided into two categories: **Inductive ZSL**, which assumes that the unseen data should be strictly inaccessible during training, and **Transductive ZSL**, which utilizes the unlabeled data of unseen classes as part of the training set.

Inductive ZSL Since visual attribute learning [14] has been proposed, many researchers conduct their work to discover the intermediate attribute classifiers for zero-shot learning. One of the most popular frameworks is compatibility learning, which learns linear or non-linear mapping functions with only using seen data and attributes, and then applying on unseen data. DAP [15] is one of the earliest compatibility frameworks, which learns probabilistic attribute classifiers and estimates the label by integrating the ranks of the learned classifiers. ALE [16], SJE [17], and DEVISE [18] employ bilinear compatibility function to project features into semantic embedding space, where the features and attributes belong to the same class with depending on the correlation is maximal or minimal. Embarrassingly Simple Zero Shot Learning (ESZSL) [19] adds an additional regularization term to the unregularized risk minimization equation.

To improve the performance and reduce the usage of manual attributes, some hybrid methods are proposed, *e.g.* Combination of Semantic Embeddings (CONSE) [20] and Semantic Similarity Embedding (SSE) [21] exploits seen classes to construct the attributes of unseen classes.

Synthetic learning is a novel type of method, which typically synthesizes pseudo features from semantic attributes. The classifiers are trained by using conventional algorithms such as Decision Tree (DT) [22] or Support Vector Machine (SVM) [23]. Some well-known methods have a similar structure as the standard one. For example, Synthesised Classifiers (SYNC) [24], Unseen Visual Data Synthesis (UVDS) [8] and Generating Pseudo Feature Representation (GPFR) [25]. The most relevant method to ours is the exponential framework GFZSL [10, 11], which treats each class distribution as an exponential family distribution, and defines the mean and variance of visual prototype for each category as the projection of the class attribute. Besides, Wang *et al.* proposed an end-to-end VAE based VZSL method [12], which represents each seen/unseen class using a class-specific latent-space distribution conditioned on class attributes. These latent-space distributions are utilized as a prior for a supervised VAE, which can facilitate to learn discriminative feature representations for the inputs. However, both GFZSL and VZSL assume that all dimensions of the prototype variance are independent of each other. In addition, they may generate prototypes with negative coefficients for unseen classes, which will not happen in real feature representations.

Since the distribution of seen data often differs from that of the unseen data, thus just using the model generated with only seen data inevitably leads to the domain shift problem, *i.e.*, if the projection model from visual feature to semantic embedding is learned only from the seen classes, the projection of unseen class image is likely to be shifted due to the bias distribution of the training seen classes. Sometimes this bias might be far away from the correct unseen class prototype, leading to an error of the subsequent nearest neighbor search. Therefore, the best way to solve this problem is to include unlabeled unseen data into training, which is called **transductive ZSL**.

The earliest concept of transductive ZSL was proposed by Y. Fu *et al.* [26], who learned a multi-label regression to generalize the model to unseen classes by utilizing both seen and unseen data. Semi-supervised framework [27] takes both labeled and unlabelled data as input, and jointly learns a multi-class classification model on all classes. The framework can consistently learn both the label representations and the model parameters across the seen classes and unseen classes. Unsupervised Domain Adaptation (UDA) [28] casts the visual-embedding projection learning problem as a sparse coding problem, which sets each dimension of the semantic embedding space corresponds to a dictionary basis vector. The coefficients/sparse code of each visual feature vector is its projection in the semantic embedding space. Y. Guo *et al.* [29] proposed a method to solve transductive ZSL with a shared model space (SMS) with replacing the shared attribute space in existing works. Recently, Y. Li *et al.* [30] exploits the intrinsic relationship between the semantic space manifold and the

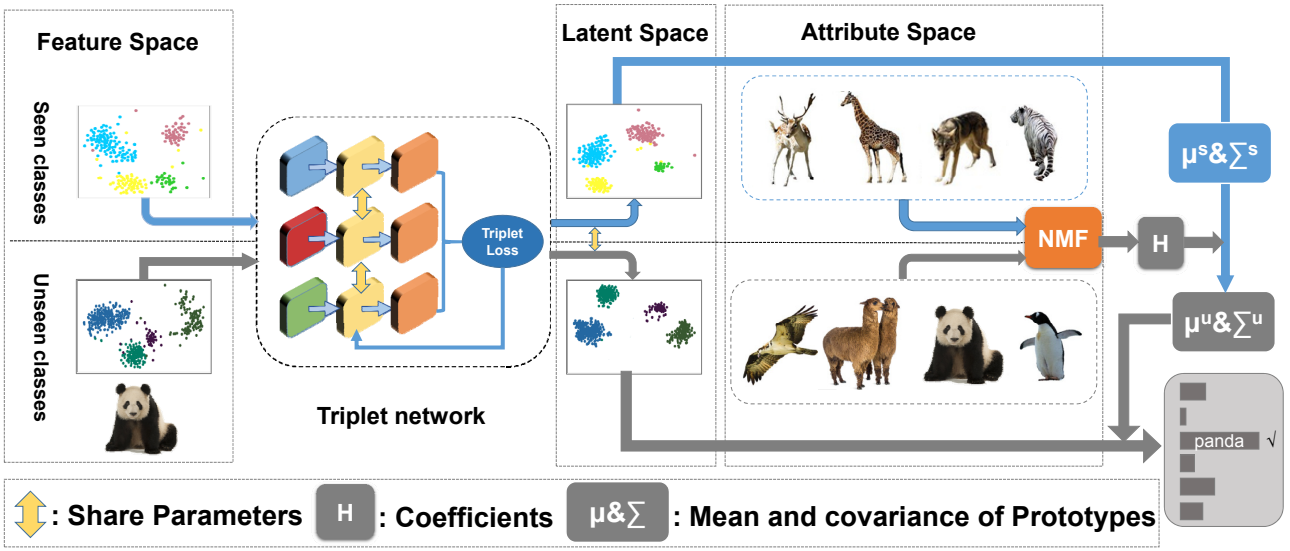


Fig. 3. The overall framework of our proposed method. The upper part with light blue arrows is the training process for generating the triplet network and the prototypes of the seen classes with the samples only from seen classes, while the bottom part with dark gray arrows is the testing process, which generates latent features with the trained triplet network, produces the coefficient matrix $\boldsymbol{H}$, synthesizes the prototypes for the unseen classes, and classifies the new test samples. Best viewed in color.

transferability of visual-semantic mapping, then formalizing their connection and cast zero-shot recognition as a joint optimization problem. J. Song et al. proposed a deep Quasi-Fully Supervised Learning network (QFSL) [31] by designing two independent objective functions for seen data and unseen data and integrates them into a whole during the training phase. Verma *et al.* extended GFZSL to transductive mode by setting the predicted unseen prototypes as the initial values, and utilizing a clustering approach with the test data to find better unseen visual prototypes. VZSL also adopts the similar strategy for transductive setting. Furthermore, VZSL extends the learned model to the few-shot setting by giving labels to several test samples of each unseen class.

GZSL According to the assumption of whether the ascription of test data is known, the ZSL task can be classified into two categories: Classical ZSL and Generalized ZSL. Classical ZSL assumes the ascription of test data is known in advance, thus the nearest neighbor searching can be conducted on only unseen classes. Chao *et al.* [32] suggests that the classical ZSL is incompatible under the actual situation because we cannot obtain the knowledge of whether the test data belongs to unseen classes beforehand in most scenarios. Therefore, they propose a new task—Generalized ZSL, which carries out the nearest neighbor searching on both seen and unseen classes. Subsequently, in CVPR2017, Y. Xian *et al.* [33] put forward a new standard split of several popular datasets for GZSL testing and released a benchmark of some recent ZSL methods, which make the later researchers more convenient to fairly compare their research results.

III. Methodology

A. Problem Definition

Let $\mathcal{S} = \{s_1, \cdots, s_p\}$ denotes a set of seen classes and $\mathcal{U} = \{u_1, \cdots, u_q\}$ denotes a set of unseen classes, where p and q are the number of seen and unseen classes respectively. The two sets are disjoint, that is to say, $\mathcal{S} \cap \mathcal{U} = \varnothing$. Besides, $\boldsymbol{A}^s = \{\boldsymbol{a}_1^s, \cdots, \boldsymbol{a}_p^s\} \in \mathbb{R}^{d_a \times p}$ and $\boldsymbol{A}^u = \{\boldsymbol{a}_1^u, \cdots, \boldsymbol{a}_q^u\} \in \mathbb{R}^{d_a \times q}$ represent for the corresponding seen and unseen class level semantic representations, such as attributes or word embeddings, where d_a is the dimension of attribute vector.

Given a set of labeled training data of seen classes $\boldsymbol{X}^s = \{\boldsymbol{x}_1^s, \cdots, \boldsymbol{x}_i^s, \cdots, \boldsymbol{x}_{N_s}^s\} \in \mathbb{R}^{d_x \times N_s}$, where d_x is the dimensionality of a single feature vector $\boldsymbol{x}_i^s$, and N_s is the number of training data. Each feature $\boldsymbol{x}_i^s$ is simultaneously associated with a label $y_i \in \mathcal{S}$ and its corresponding attribute $\boldsymbol{a}_{y_i} \in \boldsymbol{A}^s$. Let $\boldsymbol{X}^u = \{\boldsymbol{x}_1^u, \cdots, \boldsymbol{x}_i^u, \cdots, \boldsymbol{x}_{N_u}^u\} \in \mathbb{R}^{d_x \times N_u}$ represents for a set of test data, which is not assigned with its corresponding labels and semantic representations, where N_u is the number of test data. The objective of ZSL is to predict the labels of the test data $\boldsymbol{X}^u$ by learning a classifier $\mathcal{F} : \boldsymbol{X}^u \rightarrow \mathcal{U}$ with the training data $\boldsymbol{X}^s$ and the whole attribute set $\boldsymbol{A}^s \cup \boldsymbol{A}^u$.

B. The Probabilistic Framework

As the problems described in the section of the introduction, the NNS based methods might cause great performance degradation due to their neglect of data distribution, thus the classification should be determined in the probabilistic framework. Furthermore, for the sake of circumventing a large number of overlapping areas in original visual space, the features should be mapped into a latent space, where the features gather together within a class and spread out between classes. Therefore, for a feature vector $\boldsymbol{x}_i^u$ in original visual space, it should be first mapped into latent space as $\boldsymbol{z}_i^u$, and then the classification can be conducted with the multivariate Gaussian distribution,

$$\begin{aligned} y_i^u &= \arg\max_{j \in \mathcal{U}} \mathcal{N}(\boldsymbol{z}_i^u | \boldsymbol{\mu}_j^u, \boldsymbol{\Sigma}_j^u) \\ &= \arg\max_{j \in \mathcal{U}} \frac{1}{(2\pi)^{\frac{d_z}{2}}} \frac{1}{|\boldsymbol{\Sigma}_j^u|^{\frac{1}{2}}} \exp\{-\frac{1}{2}(\boldsymbol{z}_i^u - \boldsymbol{\mu}_j^u)^T (\boldsymbol{\Sigma}_j^u)^{-1} (\boldsymbol{z}_i^u - \boldsymbol{\mu}_j^u)\}, \end{aligned} \tag{1}$$

where, $\boldsymbol{\mu}_j^u$ and $\boldsymbol{\Sigma}_j^u$ are the mean vector and covariance matrix respectively of each unseen prototype, and d_z is the dimensionality of the vector in latent space.

According to Eq. 1, there are two things left to complete for this framework, one is to find a projection function $\mathcal{G}: \boldsymbol{X}^u \rightarrow \boldsymbol{Z}^u$ to map the original features into latent space, where the projected features are more discriminative; another is to synthesize the Gaussian distribution parameters $\boldsymbol{\mu}_j^u$ and $\boldsymbol{\Sigma}_j^u$ of each unseen prototype in latent space.

For the former one, the best way is adopting the triplet based metric learning method, thus, we exploit the labeled training data $\boldsymbol{X}_s$ and their corresponding label $\mathcal{S}$ to generate triplets, and learn a mapping function $\mathcal{G}': \boldsymbol{X}^s \rightarrow \boldsymbol{Z}^s$, which is then used as $\mathcal{G}$ when computing $\boldsymbol{Z}^u$. For the latter one, since we have already obtained the seen attributes $\boldsymbol{A}^s$ and the unseen attributes $\boldsymbol{A}^u$, it is feasible to compute nonnegative coefficients $\boldsymbol{H} \in \mathbb{R}^{s\times u}(\boldsymbol{H}_{ij} \geqslant 0)$ to synthesis $\boldsymbol{A}^u$ from $\boldsymbol{A}^s$, which can be addressed by applying Non-negative Matrix Factorization (NMF). Furthermore, if $\boldsymbol{Z}^s$ has been generated with the previous manipulation, it is easy to obtain the mean vector $\boldsymbol{\mu}_i^s$ and covariance matrix $\boldsymbol{\Sigma}_i^s$ of each seen prototype in latent space. With the precomputed $\boldsymbol{H}$, calculating the distribution of unseen prototypes will be an easy operation.

The whole framework is illustrated in Fig. 3 and the details are clarified in the following several subsections.

C. Triplet Network

Since the data distribution of each class in visual feature space is often overlapped with each other, which makes the features difficult to be classified, it is necessary to define a latent space to project the original features into it, where the data points are more discriminative. The latent space should have two characteristics, one is the data should be gathered within a class and scattered between classes, another is to avoid the curse of dimensionality, the data points in latent space should have less dimensionality than that in original feature space. The best way to fulfill the above two requirements is metric learning [34, 35], such as Siamese network and triplet network.

Siamese network exploits pairwise samples, including similar and dissimilar pairs, it encourages the network to pull together similar pairs and push away dissimilar pairs, while triplet network uses triplets, it conducts the same work as the Siamese network within one sample. Lots of experiments have proved that the triplet network is better than the Siamese network in most situations. Therefore, in our method, we adopt the triplet network. There are three important elements in the triplet network, including objective function, triplet selection, and network architecture, which are described detailedly as follows.

Objective

The latent embedding is denoted as $\boldsymbol{z} = \mathcal{G}'(\boldsymbol{x})$, which embeds a visual feature $\boldsymbol{x}$ into a low dimensional latent space. Here we intend to ensure that a visual feature $\boldsymbol{x}_i^a$ (anchor) of a special class is closer to the feature $\boldsymbol{x}_i^p$ (positive) of the same class than it is to any feature $\boldsymbol{x}_i^n$ (positive) of any other class.

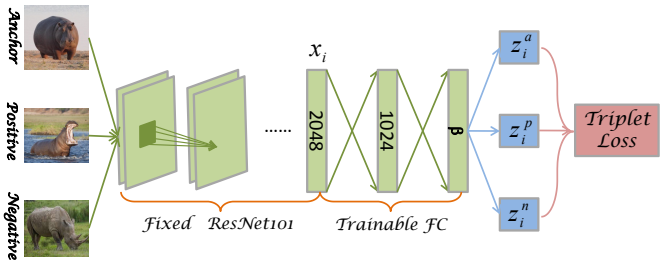


Fig. 4. Illustration of the architecture of the triplet network.

Since the triplet model in FaceNet [36] has achieved great success in face recognition, we adopt the concept and define,

$$\|\mathcal{G}'(\boldsymbol{x}_i^a) - \mathcal{G}'(\boldsymbol{x}_i^p)\|_2^2 + \alpha < \|\mathcal{G}'(\boldsymbol{x}_i^a) - \mathcal{G}'(\boldsymbol{x}_i^n)\|_2^2, \quad (2)$$

where, α is a margin, and $(\boldsymbol{x}_i^a, \boldsymbol{x}_i^p, \boldsymbol{x}_i^n)$ is a triplet sample. Therefore, the objective loss function can be defined as,

$$\begin{aligned}\mathcal{L} = \sum_{i=1}^{N_s} \max(0, &\|\mathcal{G}'(\boldsymbol{x}_i^a) - \mathcal{G}'(\boldsymbol{x}_i^p)\|_2^2 \\ &- \|\mathcal{G}'(\boldsymbol{x}_i^a) - \mathcal{G}'(\boldsymbol{x}_i^n)\|_2^2 + \alpha).\end{aligned} \quad (3)$$

Triplet Selection

Choosing suitable triplets to use is very important for achieving good performance. Inspired by FaceNet [36], we adopt the semi-hard mining strategy to select triplets, which is described as,

$$\begin{aligned}\|\mathcal{G}'(\boldsymbol{x}_i^a) - \mathcal{G}'(\boldsymbol{x}_i^p)\|_2^2 &< \|\mathcal{G}'(\boldsymbol{x}_i^a) - \mathcal{G}'(\boldsymbol{x}_i^n)\|_2^2 \\ &< \|\mathcal{G}'(\boldsymbol{x}_i^a) - \mathcal{G}'(\boldsymbol{x}_i^p)\|_2^2 + \alpha.\end{aligned} \quad (4)$$

The inequality 4 is much more stable than the hard negative strategy, which often leads to a bad local minimum early during training [36].

Architecture

As deep learning has achieved great success in metric learning methods, we also adopt the deep concept into our framework. We use the visual features extracted with ResNet101 [2] and append two full connection layers behind it. The appended two layers have the dimensionality of 1024 and β (determined by a different dataset with cross-validation, which will be explained in the experimental part) respectively, and a nonlinear activation layer (ReLU) between them, which can be found in Fig. 4.

D. Latent Nonnegative Prototype Synthesis (LNPS)

According to Eq. 1, since we have obtained the projected model from visual space to latent space, the remaining work is to obtain the mean vector $\boldsymbol{\mu}_i^u$ and the covariance matrix $\boldsymbol{\Sigma}_i^u$ of each unseen prototype. Conventional method such as that proposed by Verma *et al.* [10] optimizes direct linear projection matrix $\boldsymbol{W}_\mu$ and $\boldsymbol{W}_\Sigma$ from attribute to $\boldsymbol{\mu}_i^s$ and $\boldsymbol{\Sigma}_i^s$, and applies it to the unseen class to compute $\boldsymbol{\mu}_i^u$ and $\boldsymbol{\Sigma}_i^u$, which can achieve obvious improvement. However, $\boldsymbol{W}_\mu$ and $\boldsymbol{W}_\Sigma$ are optimized with only seen data and seen attributes, which often suffer from the domain shift problem. Besides, there is no constraint on $\boldsymbol{W}_\mu$ and $\boldsymbol{W}_\Sigma$, which has a large opportunity to have some entries of them to be smaller than

zero, which is unreasonable for feature synthesis and often leads to over-fitting.

To circumvent these problems, we build a connection between the attributes of seen classes and unseen classes, and make the nonnegative constraint on the parameters, which can be represented as,

$$\begin{aligned} \boldsymbol{A}^u &= \boldsymbol{A}^s\boldsymbol{H} \\ \text{s.t. } \boldsymbol{H}_{ij} &\geqslant 0, \end{aligned} \tag{5}$$

where, $\boldsymbol{H}$ is the synthesis coefficient matrix, and $\boldsymbol{H}_{ij}$ is the entry of $\boldsymbol{H}$ in row i and column j.

Optimization

Due to the nonnegative constraint, Eq. 5 is not convex, thus, it can not be solved with a closed-form solution. Here, we adopt the concept of NMF to minimize the LSE loss of each entry, and define the following loss function,

$$\mathcal{J}(\boldsymbol{H}) = \frac{1}{2}\sum_{i,j}(\boldsymbol{A}^u_{ij} - (\boldsymbol{A}^s\boldsymbol{H})_{ij}). \tag{6}$$

Since $(\boldsymbol{A}^s\boldsymbol{H})_{ij}$ can be represented as,

$$(\boldsymbol{A}^s\boldsymbol{H})_{ij} = \sum_k \boldsymbol{A}^s_{ik}\boldsymbol{H}_{kj}, \tag{7}$$

thence,

$$\frac{\partial(\boldsymbol{A}^s\boldsymbol{H})_{ij}}{\partial\boldsymbol{H}_{kj}} = \boldsymbol{A}^s_{ik}. \tag{8}$$

We make derivative of Eq. 6 with respected to $\boldsymbol{H}_{kj}$, and obtain,

$$\begin{aligned} \frac{\partial(\mathcal{J}(\boldsymbol{H}))}{\partial\boldsymbol{H}_{kj}} &= \sum_i \boldsymbol{A}^s_{ik}((\boldsymbol{A}^s\boldsymbol{H})_{ij} - \boldsymbol{A}^u_{ij}) \\ &= \sum_i \boldsymbol{A}^s_{ik}(\boldsymbol{A}^s\boldsymbol{H})_{ij} - \sum_i \boldsymbol{A}^s_{ik}\boldsymbol{A}^u_{ij} \\ &= ((\boldsymbol{A}^s)^T\boldsymbol{A}^s\boldsymbol{H})_{kj} - ((\boldsymbol{A}^s)^T\boldsymbol{A}^u)_{kj}. \end{aligned} \tag{9}$$

Then we can use the Gradient Descent (GD) to compute $\boldsymbol{H}_{kj}$, which can be denoted as,

$$\boldsymbol{H}_{kj} = \boldsymbol{H}_{kj} - \lambda\cdot[((\boldsymbol{A}^s)^T\boldsymbol{A}^s\boldsymbol{H})_{kj} - ((\boldsymbol{A}^s)^T\boldsymbol{A}^u)_{kj}], \tag{10}$$

where, λ is the learning rate.

For Eq. 10, we set,

$$\lambda = \frac{\boldsymbol{H}_{kj}}{((\boldsymbol{A}^s)^T\boldsymbol{A}^s\boldsymbol{H})_{kj}}, \tag{11}$$

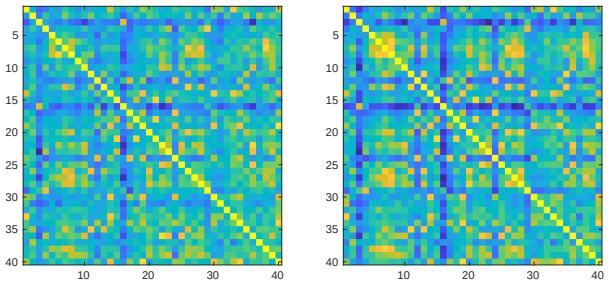
then, it can be simplified as,

$$\boldsymbol{H}_{kj} = \boldsymbol{H}_{kj}\cdot\frac{((\boldsymbol{A}^s)^T\boldsymbol{A}^u)_{kj}}{((\boldsymbol{A}^s)^T\boldsymbol{A}^s\boldsymbol{H})_{kj}}. \tag{12}$$

Eq. 12 is a typical iterative process, which can guarantee $\boldsymbol{H}_{kj}$ is nonnegative during optimization, and the convergence proof can be found in [37].

Synthesis

Due to the fact that the class attributes annotated by human experts are mostly according to their visual appearance, thus we assume that the distribution of interclass similarity in visual and attribute spaces are consistent. Furthermore, the triplet network is applied to reduce the intraclass variance so as to further alleviate interclass similarity shift. To verify our assumption, the normalized similarity matrices of the seen classes of AWA in both latent space and attribute space are shown in Fig. 5. Yellow indicates high similarity and blue is the opposite. It is clear that the distributions of interclass similarity in latent and attribute spaces are consistent. Since Coefficient Matrix $\boldsymbol{H}$ aims to capture the interclass similarities between unseen and seen classes, but the latent feature of unseen classes is not available before the ZSL test, we can estimate it from attribute space and apply it to the latent feature space as an approximate alternative.



(a) Latent space (b) Attribute space

Fig. 5. Illustration of the similarity of seen classes of AWA in both latent space and attribute space.

In the last subsection, we have obtained the projection function $\mathcal{G}$ from visual space to latent space, thus, it is easy to get the projected features $\boldsymbol{Z}^u$ and $\boldsymbol{Z}^s$, which are then exploited to compute the seen prototypes,

$$\begin{cases} \boldsymbol{\mu}^s_c = \sum_i^{N_s} \boldsymbol{z}^s_i \cdot 1(\ell(\boldsymbol{z}^s_i) == c), \quad where \quad c \in \mathcal{S} \\ \boldsymbol{\Sigma}^s_c = \frac{1}{N_c}\sum_i^{N_s}((\boldsymbol{z}^s_i - \boldsymbol{\mu}^s_c)(\boldsymbol{z}^s_i - \boldsymbol{\mu}^s_c)^T)\cdot 1(\ell(\boldsymbol{z}^s_i) == c), \end{cases} \tag{13}$$

where, $\ell(\boldsymbol{z}^s_i)$ is the label function of $\boldsymbol{z}^s_i$, and $1(\cdot)$ is the indicator function when the condition is satisfied the output is 1, otherwise 0. N_s is the number of all seen data, and $N_c = \sum_i^{N_s} 1(\ell(\boldsymbol{z}^s_i) == c)$ denotes the number of features falling into the c^{th} category.

Similar to the attribute synthesis, we exploit the same coefficient matrix $\boldsymbol{H}$ to synthesize the unseen prototypes in latent space,

$$\begin{cases} \boldsymbol{\mu}^u_c = \sum_i^p \boldsymbol{\mu}^s_i \cdot \boldsymbol{H}_{ic} \\ \boldsymbol{\Sigma}^u_c = (\sum_i^p \boldsymbol{H}_{ic})\cdot(\sum_i^p \frac{\boldsymbol{H}_{ic}}{\boldsymbol{\Sigma}^s_i})^{-1}, \end{cases} \tag{14}$$

where, $c \in \mathcal{U}$. Given an input test data $\boldsymbol{x}^u_i$, we can map it into latent space with the function $\mathcal{G}(\cdot)$, which is learned with triplet network described in last subsection, and get $\boldsymbol{z}^u_i = \mathcal{G}(\boldsymbol{x}^u_i)$. According to Eq. 1, it is easy to obtain the label of the new unseen data $\boldsymbol{x}^u_i$.

E. Transductive Setting

In standard ZSL setting, the parameters to estimate the unseen classes are learned only from the data of seen classes, and this setting is often called *Inductive Setting*. But in realistic scenarios, the distribution of unseen data often differs from

that of seen classes, which will lead to great performance degradation due to the domain shift problem. In our proposed method, we exploit the unseen attributes in the training phase to creatively construct the relationship between seen classes and unseen classes, which can alleviate the domain shift problem to a certain degree. However, only one attribute vector of each unseen class cannot thoroughly capture the distribution, thus it is unable to solve the domain shift problem perfectly.

Sometimes, we may have the opportunity to access the unlabeled data from unseen classes. Therefore, we can include them in the training phase to obtain its real distribution, which will definitely improve the performance. This setting is named *Transductive Setting*. In our work, the objective is to leverage the unlabeled unseen data to further improve the estimation $\{\boldsymbol{\mu}_c^u, \boldsymbol{\Sigma}_c^u\}, c \in \mathcal{U}$ upon the inductive results.

Suppose the given unlabeled data of each category follows the Gaussian distribution independently, the entire data set can be modeled with a Gaussian Mixture Model (GMM) [38], which can be represented as,

$$p(\boldsymbol{z}_i^u) = \sum_{k=1}^{q} \pi_k \mathcal{N}(\boldsymbol{z}_i^u | \boldsymbol{\mu}_k^u, \boldsymbol{\Sigma}_k^u), \tag{15}$$

where, π_k is the mixing coefficient, and follows two constraints, which are $0 \leqslant \pi_k \leqslant 1$ and $\sum_{k=1}^{q} = 1$. The log of the likelihood function of whole dataset of the unseen classes is given by,

$$\ln p(\boldsymbol{z}_i^u | \boldsymbol{\pi}, \boldsymbol{\mu}^u, \boldsymbol{\Sigma}^u) = \sum_{n=1}^{N_u} \ln \sum_{k=1}^{q} \pi_k \mathcal{N}(\boldsymbol{z}_i^u | \boldsymbol{\mu}_k^u, \boldsymbol{\Sigma}_k^u). \tag{16}$$

Maximizing the formulation 16 can be addressed by an elegant and powerful method, namely Expectation-Maximization (EM) algorithm [38], which can be expressed as the following procedure,

- Initialize the mean vectors $\boldsymbol{\mu}_k^u, k \in \mathcal{U}$ and covariance matrices $\boldsymbol{\Sigma}_k^u, k \in \mathcal{U}$ with the synthesized results from inductive setting in last subsection. Initialize the mixing coefficients $\pi_k = \frac{1}{q}$, and evaluate the initial value of the log likelihood with Eq. 16.
- **E**-step. Estimate the expected values using the current parameters,

$$\gamma(\ell_{nk}) = \frac{\pi_k \mathcal{N}(\boldsymbol{z}_n^u | \boldsymbol{\mu}_k^u, \boldsymbol{\Sigma}_k^u)}{\sum_{j=1}^{q} \pi_k \mathcal{N}(\boldsymbol{z}_n^u | \boldsymbol{\mu}_j^u, \boldsymbol{\Sigma}_j^u)}, \tag{17}$$

  where, ℓ_{nk} means assigning the n^{th} data point with the k^{th} label.
- **M**-step. Re-evaluate the parameters using the current expected values,

$$\boldsymbol{\mu}_k^u = \frac{1}{N_k} \sum_{n=1}^{N_u} \gamma(\ell_{nk}) \boldsymbol{z}_n^u, \tag{18}$$

$$\boldsymbol{\Sigma}_k^u = \frac{1}{N_k} \sum_{n=1}^{N_u} \gamma(\ell_{nk}) (\boldsymbol{z}_n^u - \boldsymbol{\mu}_k^u)(\boldsymbol{z}_n^u - \boldsymbol{\mu}_k^u)^T, \tag{19}$$

$$\pi_k = \frac{N_k}{N_u}, \tag{20}$$

  where,

$$N_k = \sum_{n=1}^{N_u} \gamma(\ell_{nk}). \tag{21}$$

- Evaluate the log-likelihood with Eq. 16 and check for convergence of either the log-likelihood or the parameters. If the convergence criterion is not satisfied, return to **E**-step.

After the EM step, we can leverage the converged parameters and Eq. 1 to predict the label of new input unseen features.

F. Computational Complexity

In this subsection, we discuss the computational complexity of our method. Since our method mainly consists of a triplet network, prototype synthesis, and GMM, we analyze them separately. For the triplet network, we assume the neurons in each layer is n for simplicity. Since the forward process mainly contains the matrix multiplication, the computational complexity for the forward network is $\mathcal{O}(\ell n^3)$, where ℓ is the number of layers. The backpropagation part mainly contains error propagation and gradient computation, complexity of the error propagation in all layers is $\mathcal{O}(\ell n^3)$ too, and if we assume that there are n gradient iterations, the total complexity of backpropagation is $n \times \mathcal{O}(\ell n^3) = \mathcal{O}(\ell n^4)$. Combing both forward and backward processes, we can conclude the computational complexity of the triplet network is $\mathcal{O}(\ell n^4)$ in a single iteration. For prototype synthesis, the computation of NMF mainly contains matrix multiplication and matrix element-wise division. Since the dimension of the attribute is d_a, and if we assume the iteration number is m, the computational complexity of prototype synthesis is $\mathcal{O}(md_a^3)$. For the process of EM, the main computation of E-step is the probability computation, which has the complexity of $\mathcal{O}(iq\beta N_u)$, where, i is the iteration number for EM. The main computation of M-step is the calculation of the covariance, which has the complexity of $\mathcal{O}(i\beta^2 N_u)$. Therefore, the computational complexity of GMM is $\mathcal{O}(i\beta(\beta + q)N_u)$.

IV. Experiments

To verify the effectiveness of our method, we use four popular datasets to evaluate our model for ZSL, and compare it with a number of state-of-the-art baselines. We conduct our experiments on two settings, including inductive setting and transductive setting, and report the results on classical ZSL and GZSL respectively. In this section, we will first briefly introduce the four datasets, and then show the performance of our method comparing to some baselines, at last, the detailed analysis will be demonstrated to show the importance of some hyper-parameters.

A. Datasets and Settings

Datasets

Similar to many other ZSL methods [33, 41, 42], we also use the same four popular datasets. For the sake of fair comparison, we leverage the same split like that in [33], and the statistics of the datasets are summarized in Tab. I. The description of the datasets is as follows,

TABLE I
THE SUMMARIZATION OF THE FOUR POPULAR DATASETS USED IN OUR EXPERIMENTS.

Dataset	Attribute Dim	Samples	Seen/Unseen Class	Test samples (unseen)	Test samples (seen)	Train (seen)
SUN [39]	102	14,340	645/102	1,440	2,580	10,320
CUB [39]	312	11,788	150/50	2,967	1,764	7,057
AWA [15]	85	30,475	40/10	5,685	4,958	19,832
aPY [40]	64	15,339	20/12	7,924	1,483	5,932

- **SUN attribute (SUN)**: The SUN dataset [39] contains 14,340 images with 645 seen classes (training set) and 72 unseen classes (test set). There are 20 images in each class, which is also associated with a 102-dimensional real value class-attribute vector. Among the seen classes, we select 4 images from each class to build the seen test set, and the left is composed of the training set of seen classes.
- **Caltech-UCSD Birds-200-2011 (CUB-200)**: The CUB-200 [43] is a fine-grained dataset, which contains 11,788 images with 150 seen classes (training set) and 50 unseen class (test set). Each image has a real value 312-dimensional class-attribute vector, specifying the presence or absence of various attributes of that image. The attribute vectors for all images in a class are averaged to construct its continuous class-attribute vector. Besides, the data of seen classes are divided into two parts, one containing 7,057 images is used for seen train, and the other containing 1,764 images is employed for the seen test.
- **Animal with Attribute (AwA)**: The AwA dataset is a coarse-grained dataset [15], which contains 30,475 images with 40 seen classes (training set) and 10 unseen classes (test set). Each class has a predefined real-value 85-dimensional class-attribute vector. The set of unseen classes has 5,685 images, and the set of seen classes has 24,790 images, among which 4,958 images are used for the seen test, and the remaining is used for training.
- **a Pascal & Yahoo attribute (aPY)**: aPY [40] is also a coarse-grained dataset, which contains 15,339 images with 32 classes, and 20 of them (Pascal set) are used as seen classes and the left 12 (Yahoo set) are treated as unseen classes. Besides, each class is associated with a 64-dimensional attribute. Among the data of 20 seen classes, 5,932 images are used for seen train, and the left 1,438 are for the seen test.

Settings

We strictly evaluate our methods by using standard class-level attributes provided by [33]. For the sake of fairness, and also the convenience of comparison, the split of the datasets also follows that proposed by [33]. The images are first processed with the pre-trained ResNet101 network [2] to extract 2048 dimensional features before being sent to the triplet network. The learning rate of the deep triplet network is set as 1×10^{-4}, the iteration number is selected as 1×10^{5}, and the batch size is set as 200. Because the different dataset is often suitable with different parameters, we utilize the cross-validation to find the optimal dimension β of each dataset. We hereby compare the difference between ZSL cross-validation to conventional machine learning approaches. Compared to inner-splits of training samples within each class, the ZSL problem requires inter-splits by in turn regarding part of seen classes as unseen. In our experiments, we randomly select 20% of the seen classes as unseen classes for validation and pick the parameter of the best performance of the average of 5 executions. The optimal dimension β of latent space for each dataset is reported in Tab. II for both ZSL and GZSL.

TABLE II
THE OPTIMAL DIMENSIONS OF LATENT SPACE FOR TRIPLET NETWORK.

Dataset	SUN	CUB	AWA	aPY
ZSL	1024	512	256	32
GZSL	1024	512	256	32

B. Comparison with Baselines

Conventional ZSL metric often focuses on the average accuracy of all test data, but it has a drawback that if there is a big class occupies more than 50% of the data, then the result will be determined only by this class, such as the class *'person'* in aPY contains over 60% of the dataset, thus the class *'person'* will dominate the entire result. However, the purpose of ZSL is to achieve good performance on all unseen categories, so it is better to compute the accuracies in each class, and then average them as the final result.

Besides, the ZSL metric assumes that the test data in advance are known belonging to unseen classes, and will be tested only on unseen classes, which is unreasonable in realistic scenarios. We usually do not know the ascription of the test data in advance, thus it is necessary to find the best assignment on both seen and unseen classes. Furthermore, the model should be not only suitable for unseen classes but also should maintain the performance on seen classes. This metric is called Generalized ZSL (GZSL), which is described as follows,

- Seen test accuracy tr: Average per-class classification accuracy for seen test samples;
- Unseen test accuracy ts: Average per-class classification accuracy for unseen test samples;
- Harmonic accuracy H: traditional arithmetic mean $H = (tr+ts)/2$, which computes the average value of tr and ts, can still generate good results when one of tr and ts is high and the other is very low. However, very low accuracy on single metric often means the trained model fails, thus here we use harmonic accuracy $H = (2\times tr\times ts)/(tr+ts)$ [33] to replace the arithmetic mean.

We compare our algorithm with 20 recently proposed inductive and transductive methods. The inductive methods include DAP [15], CONSE [20], CMT [44], SSE [21], LATEM [45], ALE [16], DEVISE [18], SJE [17], ESZSL [19], SYNC [24], SAE [46] CVAEZSL [47], PRESERVE model [48],

TABLE III
COMPARISON WITH STATE-OF-THE-ART BASELINES ON GZSL SETTING. '-' MEANS NOT REPORTED OR NOT AVAILABLE.

	SUN			CUB			AWA			aPY		
Method	**ts**	**tr**	**H**	**ts**	**tr**	**H**	**ts**	**tr**	**H**	**ts**	**tr**	**H**
DAP [15]	4.2	25.1	7.5	1.7	67.9	3.3	0.0	**88.7**	0.0	4.8	78.3	9.0
CONSE [20]	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 [44]	8.1	21.8	11.8	7.2	49.8	12.6	0.9	87.6	1.8	1.4	85.2	2.8
SSE [21]	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 [45]	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 [16]	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 [18]	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 [17]	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 [19]	11.0	27.9	15.8	12.6	63.8	21.0	6.6	75.6	12.1	2.4	70.1	4.6
SAE [46]	8.8	18.0	11.8	7.8	54.0	13.6	1.8	77.1	3.5	0.4	80.9	0.9
SYNC [24]	7.0	**43.3**	13.4	11.5	70.9	19.8	8.9	87.3	16.2	7.4	66.3	13.3
CVAE-ZSL [47]	-	-	26.7	-	-	34.5	-	-	47.2	-	-	-
PRESERVE [48]	20.8	37.2	26.7	24.6	54.3	33.9	-	-	-	13.5	51.4	21.4
CDL [49]	21.5	34.7	26.5	23.5	55.2	32.9	28.1	73.5	40.6	19.8	48.6	28.1
GFZSL [10]	0.0	39.6	0.0	0.0	45.7	0.0	1.8	80.3	3.5	0.0	83.3	0.0
LAGO [50]	18.8	33.1	23.9	21.8	73.6	33.7	23.8	67.0	35.1	-	-	-
PSEUDO [51]	19.0	32.7	24.0	23.0	51.6	31.8	22.4	80.6	35.1	15.4	71.3	25.4
KERNEL [52]	21.0	31.0	25.1	24.2	63.9	35.1	18.3	79.3	29.8	11.9	76.3	20.5
TVN [6]	18.2	28.9	22.3	21.6	47.5	29.7	18.2	87.5	30.2	8.8	59.1	15.4
VZSL [12]	15.2	23.8	18.6	17.1	37.1	23.8	22.3	77.5	34.6	8.4	75.5	15.1
Our Method	**39.7**	38.9	**39.3**	**37.8**	58.2	**45.9**	**37.0**	84.7	**51.4**	**25.9**	79.5	**39.1**

TABLE IV
COMPARISON WITH STATE-OF-THE-ART ZSL BASELINES ON BOTH INDUCTIVE SETTING AND TRANSDUCTIVE SETTING.

Dataset	**SUN**	**CUB**	**AWA**	**aPY**	**Average**
DAP [15]	39.9	40.0	44.1	33.8	39.5
CONSE [20]	38.8	34.3	45.6	26.9	36.4
CMT [44]	39.9	34.6	39.5	28.0	35.5
SSE [21]	51.5	43.9	60.1	34.0	47.4
LATEM [45]	13.0	49.3	55.1	35.2	48.7
ALE [16]	58.1	54.9	59.9	39.7	53.2
DEVISE [18]	56.5	52.0	54.2	39.8	50.6
SJE [17]	53.7	53.9	65.6	32.9	51.5
ESZSL [19]	54.5	53.9	58.2	38.3	51.2
SYNC [24]	56.3	**55.6**	54.0	23.9	47.5
SAE [46]	40.3	33.3	53.0	8.3	36.2
GFZSL [10]	**60.6**	49.3	**68.3**	38.4	54.2
TVN [6]	59.3	54.9	64.7	40.9	55.0
VZSL [12]	52.0	43.8	63.7	30.3	47.5
Our Method	60.4	53.2	67.4	**42.8**	**56.0**
QFSL [31]	**63.7**	56.2	60.4	38.6	54.7
GFZSL-Trans [10]	59.4	45.2	74.7	35.9	53.8
VZSL-Trans [12]	57.6	49.3	69.1	35.7	52.9
Our Transductive	61.5	**59.3**	**82.2**	**44.4**	**61.9**

Coupled Dictionary Learning (CDL) [49], Probabilistic AND-OR Attribute Grouping Model (LAGO) [50], Pseudo Transfer (PSEUDO) [51], KERNEL model [52], Triple Verification Network (TVN) [6], and Conditional Variational ZSL (VZSL) [12]. The transductive methods include GFZSL-Trans [10], VZSL-Trans [12] and QFSL [31], and all the results are recorded in Tab. IV and Tab. III, among which VZSL and QFSL are implemented by us according to the algorithms described in their original papers, and the others are directly cited from [33] or the results reported by themselves. Besides, since we have already known the ascription of the test data in transductive setting, we do not report the performances of transductive methods such as QFSL [31] and GFZSL-Trans [10] on GZSL.

Comparison on ZSL

We conduct experiments on all the four datasets for the metric ZSL and report the results in Tab. IV. From the table, we can discover that our method of inductive setting can achieve the best performance on aPY, and exceed DEVISE, the best method on the corresponding dataset, by 3.0%. On the datasets SUN, AWA and CUB, our method can achieve the second, second and fifth place and lower than the best methods by 2.1%, 0.9% and 2.4% respectively. In the transductive setting, our method can reach the first place on three datasets, including CUB, AWA and aPY, and obtain 3.1%, 7.5% and 4.6% higher compared with the best methods QFSL, GFZSL-Trans and DEVISE respectively.

Although our method gets a bit worse performance than the best methods on SUN, AWA and CUB (on the inductive setting), the average performance on all dataset is much higher than the best baselines, no matter on inductive setting or transductive setting. Concretely, our method can achieve 56.0% on inductive setting and 61.9% on the transductive setting and surpass the best method TVN and QFSL by 1.0% and 7.2% respectively.

In our opinion, the standard to evaluate a method cannot only rely on the performance on a single dataset, while it should depend on as many datasets as possible and take the best performance of the average score as the winner. Therefore, we argue that our method can outperform the state-of-the-art methods of ZSL.

Comparison on GZSL

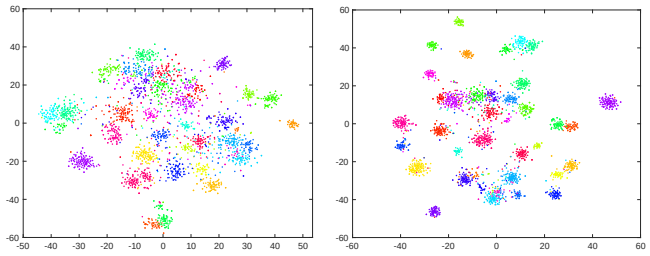
In conventional ZSL, we assume that the search range is fixed on the unseen classes, which is unrealistic because it is unable to know whether the new sample belongs to the seen classes or the unseen classes in advance. Therefore, the more realistic way is to test on all classes, *i.e.*, GZSL. In this subsection, we conduct the experiments and give the analysis on GZSL.

Since the transductive setting assumes that the ascription of the test to seen classes or unseen classes, it is pointless to report the result on GZSL. The experiments on GZSL are also carried out on the four datasets, and the results are shown in Tab. III.

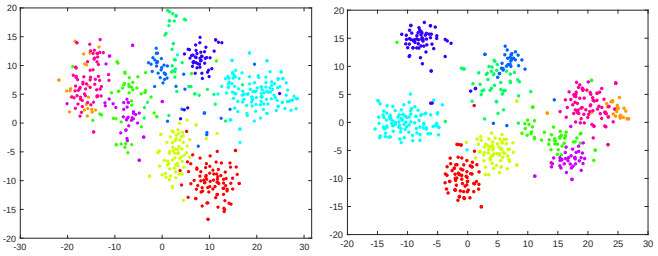
From Tab. III, it can be observed that our method can achieve the best performance on ts and H on all four datasets. Specifically, as for the metric ts, we can achieve 17.9%, 13.6%, 8.9%, and 6.1% higher than the best baselines; for the harmonic accuracy H, 12.6%, 10.8%, 4.2%, and 11.0% improvement can be obtained compared to the best methods on each dataset.

As for ts, the seen test accuracy, although our method cannot reach the first place on any dataset, it can still keep the performance on the upper-middle level. Some methods such as DAP and CONSE can get the highest score on ts on the datasets CUB, AWA and aPY, but they almost get no correct recognition for the unseen test set, which finally leads to bad performance on H. This phenomenon reveals that those methods are over-fitting on seen classes, thus they perform badly on unseen classes.

Besides, we further compare the results in Tab. III and Tab. IV. It is observed that although our method cannot get the best performances on ZSL on datasets SUN, CUB and AWA, it can surpass the best methods by a large margin on GZSL. Since we all have known that GZSL is more reasonable than ZSL in realistic scenarios due to the ascription of the input data to seen or unseen classes is usually unknown, we can conclude that our method is superior to the state-of-the-art methods.



(a) seen classes of AWA



(b) unseen classes of AWA

Fig. 6. t-SNE illustration of data points of AWA before and after the triplet network. The left column shows the data points unprocessed, and the right column demonstrates the processed data points. Points with same color belong to same category. (best view in color)

C. Ablation Study

In this section, we will give some detailed analysis of several hyperparameters, such as the effect of triplet network, the dimension of the latent space β and the influence of the LNPS.

Effect of Triplet Network

Since we have claimed that directly using visual features extracted from ResNet will cause the overlap between the distribution of each class and lead to performance degradation,

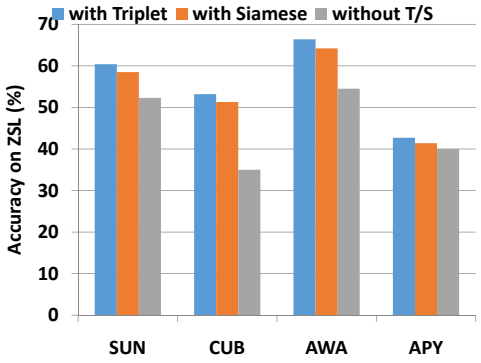


Fig. 7. Results with Triplet Network (T), Siamese Network (S) and Neither of them on ZSL.

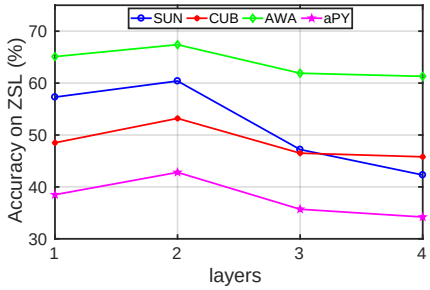


Fig. 8. Results on ZSL with different layers of Triplet Network (TN).

TABLE V
RESULTS OF OUR METHOD WITH PROBABILISTIC FRAMEWORK AND NNS ON BOTH ZSL AND GZSL. 'IND' MEANS INDUCTIVE SETTING, AND 'TRANS' MEANS TRANSDUCTIVE SETTING. 'NNS' STANDS FOR METHOD WITH NNS, AND 'PRO' REPRESENTS METHOD WITH PROBABILISTIC FRAMEWORK.

		ZSL		GZSL		
Datasets	**Methods**	**Ind**	**Trans**	**ts**	**tr**	**H**
SUN	NNS	58.9	60.2	21.8	42.5	28.8
	Pro	60.4	61.5	39.7	38.9	39.3
CUB	NNS	50.2	55.7	26.0	67.1	37.4
	Pro	53.2	59.3	37.8	58.2	45.9
AWA	NNS	63.7	79.5	27.8	83.9	41.9
	Pro	67.4	82.2	37.0	84.7	51.4
aPY	NNS	41.2	43.2	23.6	79.9	36.5
	Pro	42.8	44.4	25.9	79.5	39.1

we exploit a triplet network to process the visual features. Here, we demonstrate three issues, the first one is can the triplet network reduce the overlapped area between each class, the second one is how much does the triplet network affect the performance, and the last one is whether the proposed two-layer full connection network is the optimal choice.

In Fig. 6, we take AWA as an example and illustrate the data points before and after the process of triplet network with t-SNE [53]. The upper row shows the data points of seen classes, and the bottom row shows the data points of unseen classes. From this figure, we can clearly observe that the points belong to the same category gather together and the overlap between classes is greatly reduced for both seen classes and unseen classes.

Since we have claimed that the process of triplet network can improve the performance of ZSL, we conduct experiments on all four datasets to verify whether the triplet network

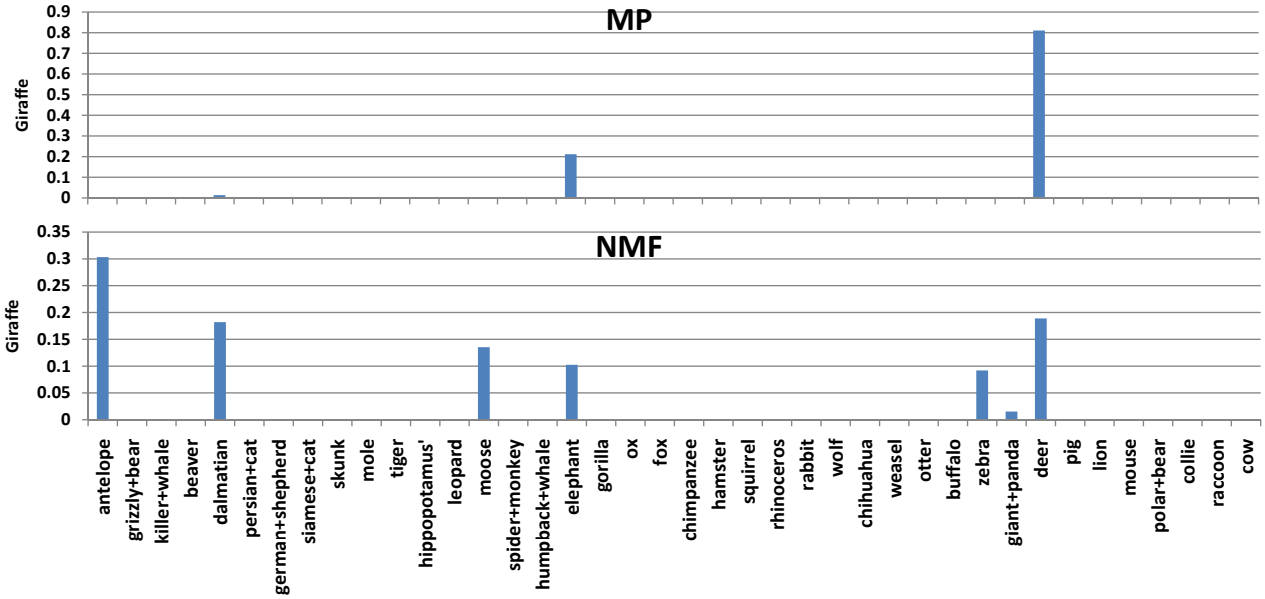
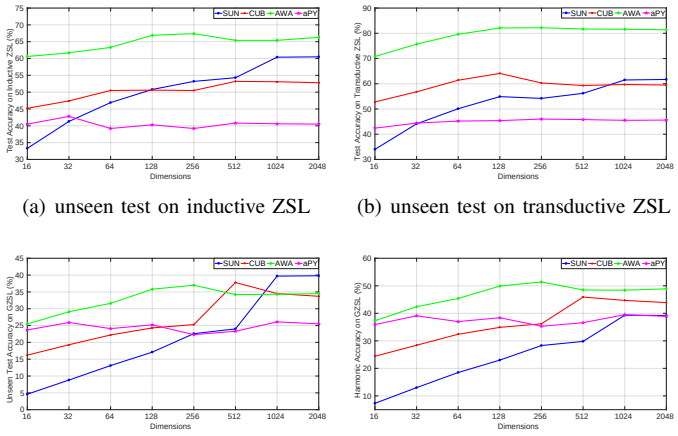


Fig. 9. Different coefficients generated with Matching Pursuit (MP) and Nonnegative Matrix Factorization (NMF) In this figure, we choose the class *'Giraffe'* in AWA as an example.



(a) unseen test on inductive ZSL (b) unseen test on transductive ZSL

(c) unseen test on GZSL (d) harmonic accuracy on GZSL

Fig. 10. Results on both ZSL and GZSL with different dimensions of latent space.

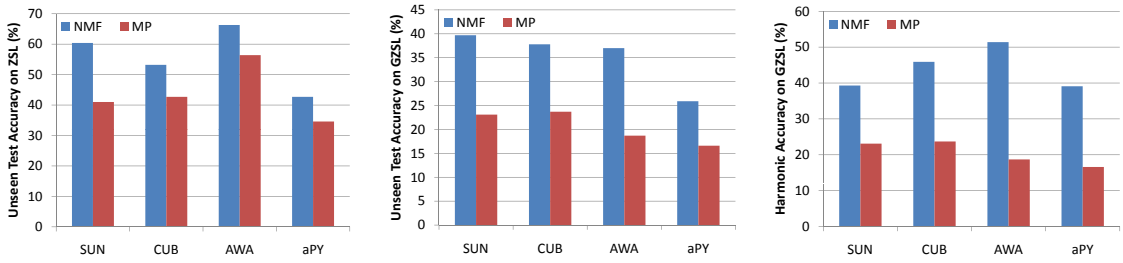
has a positive effect. Besides, we also experiment with the Siamese network to show our method is better than it too, and the results are recorded in Fig. 7. From the figure, we can discover that our method with triplet network can improve the performance significantly, especially on CUB, comparing to that with neither of triplet and Siamese networks. The reason for this phenomenon is that CUB is a fine-grained dataset, where the data points are very similar in visual space, thus there are many overlapped areas. After the process of triplet network, the data points gather together within a class and spread out between classes, *i.e.*, the data points are more discriminative. APY is a coarse-grained dataset and also has a little improvement, which further proved that the triplet network in our method plays an important role in improving the final performance. Furthermore, we can also find that our model with the triplet network slightly outperforms that with the Siamese network, which is caused by the fact that the Siamese network cannot deal with both negative and positive samples simultaneously.

To answer the third question, we design an experiment by modifying the layers of triplet network and report the results on four datasets in Fig. 8. In this experiment, besides the proposed two-layer model, we compose another three network architectures, including one-layer ($2048 \rightarrow \beta$), three-layer ($2048 \rightarrow 1024(ReLU) \rightarrow 1024(ReLU) \rightarrow \beta$) and four-layer ($2048 \rightarrow 1024(ReLU) \rightarrow 1024(ReLU) \rightarrow 1024(ReLU) \rightarrow \beta$). From the figure, we can find the best results always appear in the two-layer model on all four datasets, which reveals the two-layer model is optimal, and the one-layer model suffers from under-fitting due to the few parameters, while the three/four-layer model falls into over-fitting because of its excessive parameters.

Effect of probabilistic framework

Since we have claimed that our method with the probabilistic framework is better than that with NNS in Fig. 1, we experiment with NNS to show the superiority of our model. In this experiment, we use NNS instead of the probability framework to determine the label of test samples in the final classification stage, and the result is recorded in Tab. V. From this table, we can clearly observe that our method with the probabilistic model can exceed that with NNS by a large margin on both ZSL and GZSL.

Different Dimensions of Latent Space The dimension of latent space is determined by cross-validation, and optimal values are recorded in Tab. II. However, we also argue that it is necessary to find out how much does the dimension of latent space affect the final results. Thus, in this subsection, we experiment with different dimensions of latent space to analyze the influence on the performance on four datasets. In this experiment, we set $\beta = \{16, 32, 64, 128, 256, 512, 1024, 2048\}$ respectively and illustrate the classification accuracies in Fig. 10. From this figure, we can discover that the final results on the real unseen test are consistent with the optimal parameters of cross-validation in most circumstances. In addition, the dimension of latent space plays a more important role on SUN



(a) unseen test accuracy on ZSL (b) unseen test accuracy on GZSL (c) harmonic accuracy on GZSL

Fig. 11. Results on both ZSL and GZSL with different synthesis methods NMF and MP.

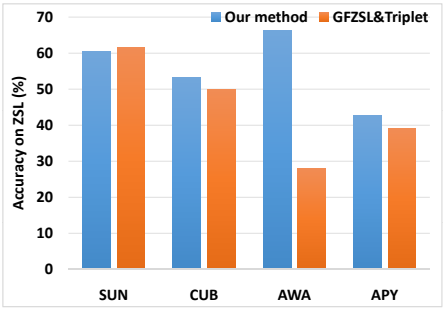


Fig. 12. Comparison with GFZSL (preprocessed with Triplet Network) on ZSL.

and CUB than on AWA and aPY. Concretely, the classification accuracy increases when the dimension of latent space grows on the dataset SUN, while the performance on aPY is affected limitedly when the dimension of latent space changes. This phenomenon is caused by that SUN has more categories than aPY, so it needs more dimensions to represent discriminative information. Besides, when the dimension reaches 1024, the performance does not change significantly, because 1024-dimensional feature can preserve enough discriminative information.

Influence of LNPS

The main creative part of our method is LNPS, thus we try to prove its superiority from multi perspectives in this subsection.

The Eq. 5 is solved by NMF, which can guarantee that the coefficients are non-negative. However, there is another method, namely Matching Pursuit (MP) [54], can also be used to generate the nonnegative coefficients. Concretely, MP first finds the most similar attributes $\boldsymbol{a}_c^s$ of $\boldsymbol{a}_i^u$ and their coefficient κ by addressing the following formulation,

$$c = \arg\min_{j \in \mathcal{S}} \|\boldsymbol{a}_i^u - \kappa \boldsymbol{a}_j^s\|_2^2 \quad \textit{suject to } \kappa \geqslant 0, \tag{22}$$

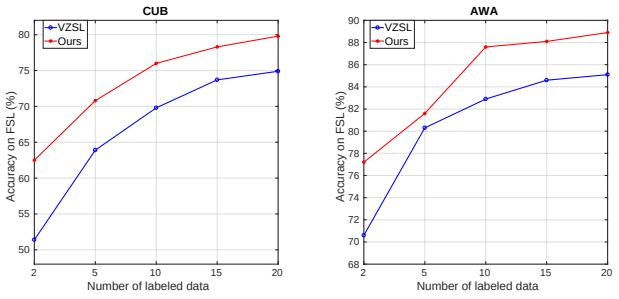
and then sets the residual $\boldsymbol{a}_i^u - \kappa \boldsymbol{a}_j^s$ as $\boldsymbol{a}_i^u$ to find the next $\boldsymbol{a}_c^s$ and κ with the formulation (22) until convergence (such as when residual reaches a very small value). Therefore, we conduct experiments on both MP and NMF to find which one is better. We first exploit MP and NMF to generate the coefficients on AWA, and choose the class *'Giraffe'* as an example, whose coefficients are drawn in Fig. 9. From the figure, we can observe that the coefficients generated by MP haves only three non-zero values, and among these values there is a big one 0.8 corresponding to the class *'deer'*, while the coefficients produced by NMF have seven non-zero values, and all the values are not so big and nearly equal to each other. The phenomenon produced by MP will lead to a very serious problem that the synthesized unseen prototype of *'Giraffe'* is very similar to the seen class *'deer'*, which will subsequently influence the classification, especially on GZSL.

We illustrate the result with NMF and MP via a histogram in Fig. 11, from which we can find that the results on ZSL with MP are a little worse than that with NMF, while the performance with MP on GZSL is lower than that with NMF by a large margin. This phenomenon is caused by the reason that MP tries to find the most similar seen classes, among which the first seen class contributes the most, while NMF attempts to synthesize the unseen class with multi reasonable seen classes. When testing on ZSL, the search range is only focused on the unseen classes, so the performances with MP and NMF are both significant. However, when testing on GZSL, the search range is extended to all the classes, thus the new test sample of unseen classes will be misclassified to be the similar seen class by the method with MP, while the method with NMF will not make such error because the synthesized unseen class is not very similar to any of the seen classes. For example, many instances of the class *'Giraffe'* will be misclassified to the category *'deer'* when testing on GZSL according to Fig. 9.

In addition, to verify the importance of our proposed LNPS, we also design another experiment comparing our method with GFZSL [10]. In our experiments, we first employ the triplet network to preprocess the input data before applying GFZSL, and the results are demonstrated in Fig. 12. In this figure, we can observe that our method can outperform the GFZSL significantly on CUB, AWA and aPY, except that the performance on SUN has a little degradation. Since the two experiments use the same preprocessing, we can argue that the probabilistic model of our method is better than that of GFZSL.

Results on Few Shot Learning

To be more generalizable, we extend our method on Few Shot Learning (FSL) and conduct an experiment to show its effect. In this experiment, we set the number of labeled samples as $\{2, 5, 10, 15, 20\}$, and report the results on the datasets of AWA and CUB. The experiment includes both our method and VZSL [12], and the results are shown in Fig. 13. From this figure, it can be clearly discovered that our method can significantly outperform VZSL under each number



(a) CUB (b) AWA

Fig. 13. Results of our method on FSL compared with VZSL [12].

of labeled data. The superiority of our method mainly comes from the nonnegative synthesis of the unseen prototypes.

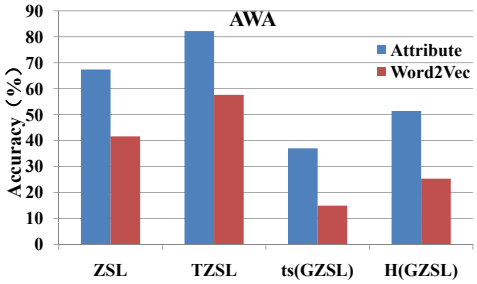


Fig. 14. Results with attribute and Word2Vec respectively on AWA.

Task/Modality Shift Problem

Task-shift is an inevitable issue in the ZSL problem because the disjoint seen/unseen distribution in the visual modality needs to be reconciled by an extra auxiliary domain. Although semantic attributes or Word2Vec [55] representation is widely adopted, they both suffer from the severe visual-semantic gap. From the experimental results in Fig. 14 it can be seen that Word2Vec suffers from more task-shift than visual attributes and the performance is remarkably degraded. This is because the semantic attribute is more associated with the visual appearance, such as color, stripe, and leg, whereas Word2Vec captures more relationships between categories with text description rather than the visual features. Therefore, we can treat semantic attribute as another type of visual representation and the task shift problem has a slight impact on final performance. However, such task-shift results in that the prototypes in the original attribute space are still not totally consistent with that in the visual feature space. Therefore, our latent visual feature is directly extracted from the visual image by deep network using the triple network as a constraint to maximumly mitigate the visual-semantic discrepancy. Evidence can be clearly observed in Tab. III, Tab. IV and Fig. 7, where the performance of using the synthesized prototype in the latent feature modality significantly outperforms that of using original visual prototypes. Therefore, the task-shift problem is alleviated by the proposed prototypes in the latent space.

V. Conclusion

In this paper, to alleviate the misclassification problem in traditional NNS based methods, we have proposed a probabilistic framework for ZSL. In this method, the visual features of seen classes are first used to train a triplet network to gather within a class and scatter between classes in latent space, and then further be leveraged to generate seen prototypes. The most creative part of the proposed work is the latent nonnegative prototypes synthesis of unseen classes, which exploits the relations between seen attributes and unseen attributes to compute the synthesis coefficients, which is subsequently utilized to synthesize prototypes of unseen classes. In addition, we also extend our method on the transductive setting. Extensive results on both ZSL and GZSL have proved that our method can outperform most state-of-the-art methods on four popular datasets, and the detailed analysis of some hyper-parameters also shows the superiority of the proposed method.

References

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

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[3] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Internation Conference on Learning Representations*, 2015.

[4] A. Zhao, M. Ding, J. Guan, Z. Lu, T. Xiang, and J.-R. Wen, "Domain-invariant projection learning for zero-shot recognition," in *Advances in neural information processing systems*, 2018, pp. 1025–1036.

[5] Y. Yan, Q. Wu, M. Tan, M. K. Ng, H. Min, and I. W. Tsang, "Online heterogeneous transfer by hedge ensemble of offline and online decisions," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 7, pp. 3252–3263, 2018.

[6] H. Zhang, Y. Long, Y. Guan, and L. Shao, "Triple verification network for generalized zero-shot learning," *IEEE Transactions on Image Processing*, vol. 28, no. 1, pp. 506–517, 2019.

[7] H. Zhang, Y. Long, W. Yang, and L. Shao, "Dual-verification network for zero-shot learning," *Information Sciences*, vol. 470, pp. 43–57, 2019.

[8] Y. Long, L. Liu, L. Shao, F. Shen, G. Ding, and J. Han, "From zero-shot learning to conventional supervised classification: Unseen visual data synthesis," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 1627–1636.

[9] H. Zhang, Y. Long, and L. Shao, "Zero-shot hashing with orthogonal projection for image retrieval," *Pattern Recognition Letters*, vol. 117, pp. 201–209, 2019.

[10] V. K. Verma and P. Rai, "A simple exponential family framework for zero-shot learning," in *The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*. Springer, 2017, pp. 792–808.

[11] A. Gaure, A. Gupta, V. K. Verma, and P. Rai, "A probabilistic framework for zero-shot multi-label learning," in *The Conference on Uncertainty in Artificial Intelligence (UAI)*, 2017.

[12] W. Wang, Y. Pu, V. K. Verma, K. Fan, Y. Zhang, C. Chen, P. Rai, and L. Carin, "Zero-shot learning via class-conditioned deep generative models," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018, pp. 4211–4218.

[13] L. Zhang, T. Xiang, and S. Gong, "Learning a deep embedding model for zero-shot learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. IEEE, 2017, pp. 2021–2030.

[14] V. Ferrari and A. Zisserman, "Learning visual attributes," in *Advances in neural information processing systems*, 2008, pp. 433–440.

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

[16] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for attribute-based classification," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2013, pp. 819–826.

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

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

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

[20] 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 *Internation Conference on Learning Representations*, 2014.

[21] Z. Zhang and V. Saligrama, "Zero-shot learning via semantic similarity embedding," in *Internation Conference on Computer Vision*, 2015, pp. 4166–4174.

[22] W. Liu and I. W. Tsang, "Making decision trees feasible in ultrahigh feature and label dimensions," *Journal of Machine Learning Research*, vol. 18, no. 1, pp. 2814–2849, 2017.

[23] W. Liu, X. Shen, B. Du, I. W. Tsang, W. Zhang, and X. Lin, "Hyperspectral imagery classification via stochastic hhsvms," *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 577–588, 2019.

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

[25] J. Lu, J. Li, Z. Yan, and C. Zhang, "Zero-shot learning by generating pseudo feature representations," *ArXiv*, vol. abs/1703.06389, 2017.

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

[27] X. Li, Y. Guo, and D. Schuurmans, "Semi-supervised zero-shot classification with label representation learning," in *Internation Conference on Computer Vision*, 2015, pp. 4211–4219.

[28] E. Kodirov, T. Xiang, Z. Fu, and S. Gong, "Unsupervised domain adaptation for zero-shot learning," in *Internation Conference on Computer Vision*, 2015, pp. 2452–2460.

[29] Y. Guo, G. Ding, X. Jin, and J. Wang, "Transductive zero-shot recognition via shared model space learning," in *AAAI Conference on Artificial Intelligence*, 2016, pp. 3494–3500.

[30] Y. Li, D. Wang, H. Hu, Y. Lin, and Y. Zhuang, "Zero-shot recognition using dual visual-semantic mapping paths," in *Internation Conference on Computer Vision*, 2017, pp. 3279–3287.

[31] J. Song, C. Shen, Y. Yang, Y. Liu, and M. Song, "Transductive unbiased embedding for zero-shot learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 1024–1033.

[32] W.-L. Chao, S. Changpinyo, B. Gong, and F. sha, "An empirical study and analysis of generalized zero-shot learning for object recognition in the wild," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 52–68.

[33] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata, "Zero-shot learning - a comprehensive evaluation of the good, the bad and the ugly," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 41, no. 9, pp. 2251–2265, 2019.

[34] W. Liu and I. W. Tsang, "Large margin metric learning for multi-label prediction." in *AAAI Conference on Artificial Intelligence Conference on Artificial Intelligence Conference on Artificial Intelligence*, vol. 15, 2015, pp. 2800–2806.

[35] W. Liu, D. Xu, I. Tsang, and W. Zhang, "Metric learning for multi-output tasks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 2, pp. 408–422, 2018.

[36] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2015, pp. 815–823.

[37] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, 2001, pp. 556–562.

[38] C. M. Bishop, *Pattern Recognition and Machine Learning*. springer, 2006.

[39] G. Patterson, C. Xu, H. Su, and J. Hays, "The sun attribute database: Beyond categories for deeper scene understanding," *International Journal of Computer Vision*, vol. 108, no. 1-2, pp. 59–81, 2014.

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

[41] H. Zhang, Y. Long, L. Liu, and L. Shao, "Adversarial unseen visual feature synthesis for zero-shot learning," *Neurocomputing*, vol. 329, pp. 12–20, 2019.

[42] H. Zhang, Y. Long, and C. Zhao, "Attribute relaxation from class level to instance level for zero-shot learning," *Electronics Letters*, vol. 54, no. 20, pp. 1170–1172, 2018.

[43] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-ucsd birds 200," California Institute of Technology, Tech. Rep., 2010.

[44] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng, "Zero-shot learning through cross-modal transfer," in *Advances in neural information processing systems*, 2013, pp. 935–943.

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

[46] E. Kodirov, T. Xiang, and S. Gong, "Semantic autoencoder for zero-shot learning," *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 3174–3183, 2017.

[47] A. Mishra, S. K. Reddy, A. Mittal, and H. A. Murthy, "A generative model for zero shot learning using conditional variational autoencoders," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 2188–2196.

[48] Y. Annadani and S. Biswas, "Preserving semantic relations for zero-shot learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018, pp. 7603–7612.

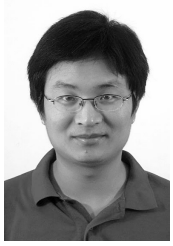
[49] H. Jiang, R. Wang, S. Shan, and X. Chen, "Learning class prototypes via structure alignment for zero-shot recognition," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 118–134.

[50] Y. Atzmon and G. Chechik, "Probabilitic and-or attribute grouping for zero-shot learning," in *The Conference on Uncertainty in Artificial Intelligence*, 2018.

[51] T. Long, X. Xu, Y. Li, F. Shen, J. Song, and H. Shen, "Pseudo transfer with marginalized corrupted attribute for zero-shot learning," in *ACM Conference on Multimedia*, 2018, pp. 1802–1810.

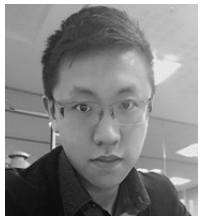
[52] H. Zhang and P. Koniusz, "Zero-shot kernel learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7670–7679.

[53] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 11, pp. 2579–2605, 2008.
[54] S. F. Cotter and B. D. Rao, "Sparse channel estimation via matching pursuit with application to equalization," *IEEE Transactions on Communications*, vol. 50, no. 3, pp. 374–377, 2002.
[55] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.



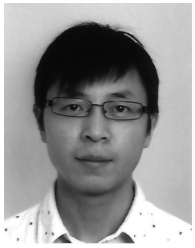
Haofeng Zhang received the B.Eng. degree and the Ph.D. degree in 2003 and 2007 respectively from School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. He is currently an associate professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology, China. His research interests include computer vision and robotics.

Huaqi Mao received the B.S. degree in Software Engineering from Chongqing University of Technology, Chongqing, China, in 2017, and is currently working towards the master degree in the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. His current research interests include image classification and deep learning.



Yang Long is currently an Assistant Professor with the Department of Computer Science, Durham University, UK. He received his Ph. D. degree in Computer Vision and Machine Learning from the Department of Electronic and Electrical Engineering, the University of Sheffield, UK, in 2017. He received the M.Sc. degree from the same institution, in 2014. His research interests include Artificial Intelligence, Machine Learning, Computer Vision, Deep Learning, Zero-shot Learning, with focus on Transparent AI for Healthcare Data Science.

Wankou Yang received the B.S., M.S., and Ph.D. degrees from the School of Computer Science and Technology, Nanjing University of Science and Technology, China, in 2002, 2004, and 2009, respectively. From 2009 to 2011, he was a Post-Doctoral Fellow with the School of Automation, Southeast University, China. From 2010 to 2011, he was a Post-Doctoral Fellow with the Face Aging Group, University of North Carolina at Wilmington, Wilmington, NC, USA. From 2011 to 2015, he was an Assistant Professor with the School of Automation, Southeast University. He is currently an Associate Professor with the School of Automation, Southeast University. His research interests include pattern recognition, computer vision, and machine learning.



Ling Shao (M'09–SM'10) is the CEO and Chief Scientist of the Inception Institute of Artificial Intelligence (IIAI), Abu Dhabi, United Arab Emirates. His research interests include computer vision, machine learning and medical imaging. He is an associate editor of IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, and several other journals. He is a fellow of the International Association of Pattern Recognition, the Institution of Engineering and Technology and the British Computer Society.