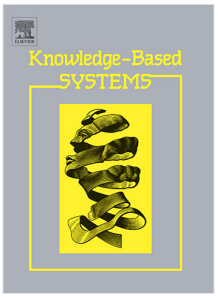
Journal Pre-proof



Learning discriminative domain-invariant prototypes for generalized zero shot learning

Yinduo Wang, Haofeng Zhang, Zheng Zhang, Yang Long, Ling Shao

PII: S0950-7051(20)30186-6
DOI: https://doi.org/10.1016/j.knosys.2020.105796
Reference: KNOSYS 105796

To appear in: *Knowledge-Based Systems*

Received date : 3 December 2019
Revised date : 17 March 2020
Accepted date : 19 March 2020

Please cite this article as: Y. Wang, H. Zhang, Z. Zhang et al., Learning discriminative domain-invariant prototypes for generalized zero shot learning, *Knowledge-Based Systems* (2020), doi: https://doi.org/10.1016/j.knosys.2020.105796.

Learning Discriminative Domain-Invariant Prototypes for Generalized Zero Shot Learning

Yinduo Wang, Haofeng Zhang, Zheng Zhang, Yang Long, Ling Shao

Highlights

1. A method called Discriminative Domain-Invariant Prototypes (DDIP) is proposed to solve the projection domain shift problem by recognizing samples in combined domains;

2. Orthogonal constraint is employed to make all the prototypes including both seen and unseen classes to be orthogonal to each other to scatter them;

3. The discriminative prototypes are restricted to distribute on the surface of an unit hyper-spherical;

4. Extensive experiments on four popular shows the effectiveness of the proposed method.

Learning Discriminative Domain-Invariant Prototypes for Generalized Zero Shot Learning

Yinduo Wang[a], Haofeng Zhang[a,*], Zheng Zhang[b], Yang Long[c], Ling Shao[d]

[a]*School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China*
[b]*Bio-Computing Research Center, Harbin Institute of Technology, Shenzhen, China.*
[c]*School of Computer Science, Durham University, Durham, UK*
[d]*Inception Institute of Artificial Intelligence (IIAI), Abu Dhabi, United Arab Emirates*

Abstract

Zero-shot learning (ZSL) aims to recognize objects of target classes by transferring knowledge from source classes through the semantic embeddings bridging. However, ZSL focuses the recognition only on unseen classes, which is unreasonable in realistic scenarios. A more reasonable way is to recognize new samples on combined domains, namely Generalized Zero Shot Learning (GZSL). Due to the fact that the source domain and target domain are disjoint and have unrelated classes potentially, ZSL and GZSL often suffer from the problem of projection domain shift. Besides, some semantic embeddings of prototypes are very similar, which makes the recognition less discriminative. To circumvent these issues, in this paper, we propose a novel method, called Learning Discriminative Domain-Invariant Prototypes (DDIP). In DDIP, both target and source domains are combined and projected into a hyper-spherical space, which is automatically learned by a regularized dictionary learning. In addition, an orthogonal constraint is employed to the latent hyper-spherical space to ensure all the class prototypes, including seen classes and unseen classes, to be orthogonal to each other to make them more discriminative. Extensive experiments on four popular benchmark and a large-scale datasets are conducted on both GZSL

*Corresponding author.
Email addresses: wangyd@njust.edu.cn (Yinduo Wang), zhanghf@njust.edu.cn (Haofeng Zhang), darrenzz219@gmail.com (Zheng Zhang), yang.long@ieee.org (Yang Long), ling.shao@ieee.org (Ling Shao)

and standard ZSL settings, and the results show that our DDIP can outperform the state-of-the-art methods.

Keywords: Generalized Zero Shot Learning (GZSL), Domain-Invariant Learning, Orthogonal Constraint, Dictionary Learning

1. Introduction

With the deep and mature application of machine learning and neural networks [6, 30, 37, 47, 18, 36, 45], the accuracy of object recognition has reached the level beyond humans. Unfortunately, models trained on one single dataset can only be applied on the same domain of the training set and hardly to be generalized to other datasets, that is to say, when there comes a novel category, which is from a different domain, the model may become ineffective. Therefore, it is necessary to find a method to solve such problem. Fortunately, Zero Shot Learning (ZSL) [52, 51, 1, 32, 23, 38, 55, 22, 25] is such a method proposed to recognize novel categories. ZSL is inspired by the behaviour of our human beings that when we meet new categories, we often utilize some auxiliary intermediate information, *e.g.*, predefined descriptions, to construct a connection between seen and unseen categories. Therefore, in the field of ZSL, we similarly employ semantic vectors, *e.g.*, attribute by experts [12], as intermediate information to achieve our purpose of recognizing novel categories. In the past decade, ZSL has made great success. However, standard ZSL only focuses on classifying new objects just within the scope of unseen categories, which is unreasonable in realistic scenarios. The more reasonable way is to find its label on both seen classes (source domain) and unseen classes (target domain), which is often called Generalized Zero Shot Learning (GZSL) [8, 53].

In ZSL setting, samples are divided into seen and unseen categories, which are disjoint from each other. Although they can be regarded as two related domains with some shared semantics, there are still lots of differences between source (train) and target (test) domains, *e.g.*, both 'tiger' and 'elephant' have same semantic 'teeth', but the teeth of tiger are short and sharp while those of

elephant are huge and coarse, and they have obvious visual differences. Thus, if the model is only trained by the samples from source domain, it will cause a huge bias on target domain during testing, and result in that the model will significantly prefer seen classes to unseen ones, and finally lead to inaccurate recognition. This phenomenon is called projection domain shift problem, which is first proposed in [14]. Due to the great influence of domain shift problem on recognition performance, how to facilitate the methods to alleviate it becomes a key issue in this field. Fu *et al.* proposed a transductive strategy to add the unlabelled unseen data into training set to solve such problem, and make a significant improvement [14]. Thereafter, many transductive methods, such as Quasi-Fully Supervised Learning (QFSL) [39], Joint Embedding Dictionary Model (JEDM) [46] have been emerging. However, in realistic scenarios, unlabelled unseen data is inaccessible during training in most circumstances. Thus, transductive method is not the best way to address this problem.

In addition, some predefined attributes are very similar to each other, which make them less discriminative, *e.g.*, the categories of 'blue wale' and 'hamperback wale' have many same attributes, which makes the embedded vectors much similar with each other, and finally lead to error classification. Zhang *et al.* [50] tried to disperse the distance between the prototypes of seen classes. However, this method only considers the prototypes of seen classes, while the unseen classes are totally ignored. The result of such method is that the prototypes of similar unseen classes still gather together, and finally lead to poor performance. Jiang *et al.* [19] built a latent space to align the visual prototypes and semantic prototypes for both seen classes and unseen classes, but they did not consider to enlarge the distance between similar prototypes, which still cannot address the semantic ambiguity of similar classes, especially on the more realistic setting GZSL.

In order to solve the aforementioned problems, in this paper, we propose a novel and effective method, namely learning Discriminative Domain-Invariant Prototypes (DDIP), to combine both domains in a latent space and disperse all the prototypes. The novelties of our model are in the following three as-

pects. Firstly, to make our model become more generalizable for both source and target domains, we employ not only seen samples, including feature and class prototypes, but also unseen class prototypes during our training phase to build relationship between source domain and target domain. Secondly, different from directly learning a projection from visual space to semantic space, in our DDIP, we first establish a latent space where we combine the both domains, then we conduct sparse coding to learn two dictionaries for visual-latent and semantic-latent projection, in this way, class prototypes of combined domains in latent space share a same public dictionary to represent themselves, which can effectively alleviate projection domain shift problem. Thirdly, a novel orthogonal constraint is applied in latent space to make our model more discriminative, especially on GZSL. Concretely, the best way is to restrain the normalization and orthogonality of each class prototype, thus, the latent space can also be recognized as a hyper-spherical space, which is illustrated in Fig. 1. To the best of our knowledge, it is the first time to establish a specific relationship between source and target domains to address the projection domain shift problem by constructing a hyper-spherical space.

In our DDIP, there are three submodules, *Visual Prototype Learning* is to learn the class prototypes in visual space; *Domain Combination* is to learn the two dictionaries for projection and *Orthogonal Constraint* adds the constraint of orthogonality in latent hyper-spherical space for both seen classes and unseen classes. After imposing the three submodules, we devise a novel objective function and develop an iterative optimization algorithm to solve it. We test our DDIP method on four benchmark and a large-scale datasets under both GZSL setting and standard ZSL setting, and extensive experimental results show that our method can outperform all the state-of-the-art methods under both two settings with a fast training speed. The contributions of our work are as follows,

- We propose a novel and effective model, called Discriminative Domain-Invariant Prototypes (DDIP), to alleviate the problems of projection domain shift and semantic ambiguity.

- To build relationship between seen classes and unseen classes, a latent hyper-spherical space is defined to adapt both source and target domains, and a dictionary learning method is also employed to learn the prototypes of all categories in this novel space.
- To make the combined domains more discriminative, especially for the more realistic setting, GZSL, we define a constraint that each prototype in the hyper-spherical space is normalized and orthogonal to each other, which can well disperse the similar prototype of both seen and unseen classes, and finally lead to performance improvement.
- Extensive experiments are conducted on five popular datasets, and the results show that our DDIP method can outperform the state-of-the-art methods on both GZSL and ZSL settings.

The rest of this paper is organized as follows: in Section 2, we briefly review some related work on ZSL& GZSL and domain shift, and our proposed model will be described in details in Section 3. In Section 4, we reports some experimental results under both GZSL, ZSL and some other pertinent settings and some discussions on these results. At last, we draw a conclusion of this paper in Section 5.

2. Related work

2.1. Imbalance Learning and ZSL

In many realistic scenarios, the datasets are often unbalanced, that is to say, in these dataset, one category significantly out-number those from the other categories (imbalanced distribution of categories). It is well known that many classification and recognition algorithms are sensitive to the imbalanced distribution of categories, so many strategies have been proposed to deal with the imbalance Learning. In [5], Bi *et al.* combined the improved ECOC (Error Correcting Output Codes) method for tackling class imbalance, and the diversified ensemble learning framework for finding the best classification algorithm for

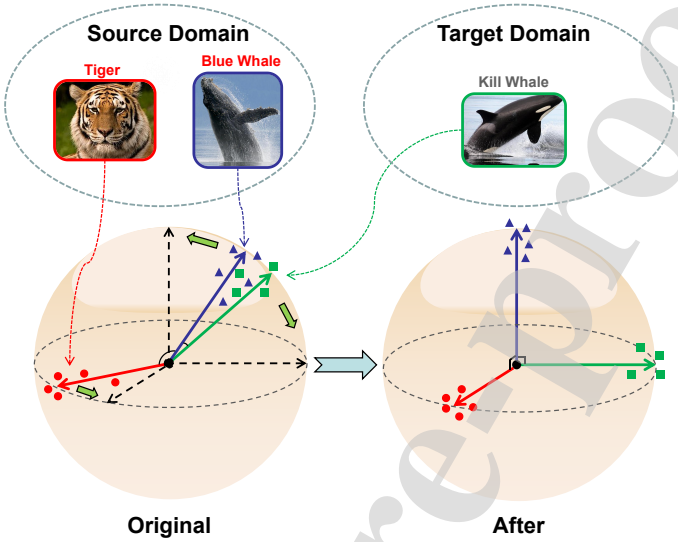


Figure 1: An illustration of the orthogonal constraint in a hyper-spherical space.

each individual sub-dataset. Moreover, Zhang *et al.* proposed an open source software for multi-class imbalanced data classification in [48]. Recently, Zhou *et al.* in [58] proposed a GAN to generate more discriminant fault samples using a scheme of global optimization, a generator was designed to generate those fault feature extracted from a few fault samples and a discriminator was designed to filter the unqualified generated samples in the sense that qualified samples are helpful for more accurate fault diagnosis. When the phenomenon of imbalance becomes more extreme, we can not get any training samples of one category, the task will become zero shot learning.

2.2. ZSL and GZSL

Both ZSL and GZSL models are trained without samples from test categories, but during testing phase, ZSL classify unseen samples within the area of unseen categories, while GZSL aims at a more realistic setting, which is a more practical and challenging task, classifying unseen instances on both seen and unseen classes. So far, many researchers have been devoting to ZSL and GZSL.

Early efforts such as DAP [22] directly train their models through proba-

bilistic attributes and try to estimate their classifiers by a maximum posterior. In ALE [2] and SJE [3], Akata *et al.* proposed a model which learns a projection from feature to semantic space by a bilinear compatibility function. Besides, other ZSL baseline methods like CONSE [31] and SSE [56] try to reduce the influence of using manual attributes by constructing unseen attribute from the instances of seen categories. Furthermore, Kodirov *et al.* for the first time used the concept of Semantic Auto-Encoder (SAE) to reconstruct visual features from semantic embeddings, which is proved to able to generalize better to the new unseen classes [21]. U. Atzmon *et al.* in LAGO[4] utilized a probabilistic model to get the natural soft and/or relations in semantic space. After that, Y. Li *et al.* in DMaP[24] devoted to exploit the intrinsic relationship among attribute manifold and the transfer ability of visual-semantic embedding. Recently, Zhao *et al.* in DIPL[57] proposed a domain-invariant feature self-reconstruction method by optimizing a simple linear formulation which casts ZSL into a min-min optimization problem. Hayashi *et al.* tried to make clustering for the training data and predict the category of the test sample by detecting whether the data falls into the learned clusters [17], which can also achieve good performance. However, these ZSL methods often pay much attention to establish a projection by employing only instances from seen classes (source domain) and do not make the full use of prototypes of unseen categories (target domain).

Verma *et al.* proposed a simple generative framework, which models each class-conditional distribution as an exponential family distribution and the parameters of the distribution of each seen/unseen class are defined as functions of the respective observed class attributes [40]. Zhang *et al.* in [50] addressed GZSL problem as a Triple Verification problem and proposed a unified optimisation of regression and compatibility functions. Long *et al.* [51, 28] proposed to construct a projection from attributes of seen classes to visual features, and then use the attributes of unseen classes to synthesize unseen visual feature, which is subsequently utilized to train a supervised model for all classes. Fu *et al.* in [15] proposed a deep weighted maximum margin framework to concentrate on the learning of class prototypes and try to make the learned class prototypes

in latent space become more discriminative. Although both seen and unseen attributes are utilized in these methods, they are processed separately, which cannot well solve the difference between them.

In addition, due to the fact that there was no agreed upon ZSL benchmark, Xian *et al.* [44] defined a new benchmark by unifying both the evaluation protocols and data splits of several publicly available datasets. They also analyzed a significant number of the state-of-the-art methods in depth, both in the classic ZSL setting and the more realistic GZSL setting, which has made a great contribution to this research field.

2.3. Domain Shift

The difference between source domain and target domain often leads to bad generalization from train to test, which is defined as the domain shift problem, and this problem is first introduced into ZSL by Fu *et al.* [14]. Fu *et al.* in this paper tried to address domain shift problem by employing Canonical Correlation Analysis (CCA) to make sure the embedding vectors still have high correlation with the original ones when mapping the low-level image features into two different latent space. Now there are two types of methods to address this problem, one is the inductive method by combining both seen and unseen attributes into training phase, and the other is transductive method, which allows models to be trained with both labeled instances of seen classes and unlabeled instances of unseen classes. For inductive setting, Long *et al.* in PSEUDO[27] put Maximum Mean Discrepancy (MMD) and Maginalized Corrupted into ZSL field to solve the domain shift problem. And Kidorov *et al.* tried to add a reconstruction item from semantic embeddings to visual features to alleviate the domain shift problem, but the effect is limited. Jiang *et al.* [19] proposed a Coupled Dictionary Learning (CDL) framework to simultaneously align the visual-semantic structures, which intents to gather the advantages of the discriminative information in the visual space and the relations in the semantic space. Although CDL can mitigate the domain shift problem to some extent, it performs bad on the more realistic GZSL setting because it ignore the

discrimination between all classes.

The concept of transductive setting is first proposed by [14], which developed a multi-view Bayesian label propagation algorithm to improve ZSL in the embedding space. Unsupervised Domain Adaptation (UDA) [20] formulates a regularized sparse coding framework, which uses the target domain class labels' projections in the semantic space to regularize the learned target domain projection. Song *et al.* [39] proposed a Quasi-Fully Supervised Learning (QFSL) method by training a transductive deep neural network, where the labelled source images are mapped to several fixed points specified by the source categories, and the unlabelled target images are forced to be mapped to other points specified by the target categories. However, we argue that this setting is not in line with the practical application that the unlabelled unseen data are usually inaccessible.

3. Methodology

3.1. Problem Definition

Given a dataset $\mathcal{D}$, which is composed of two groups, seen classes $\mathcal{S}$ and unseen classes $\mathcal{U}$, where $\mathcal{S} = \{1, \cdots, s\}$ and $\mathcal{U} = \{s+1, \cdots, s+u\}$, $\mathcal{S}$ and $\mathcal{U}$ are disjoint $\mathcal{S} \cup \mathcal{U} = \varnothing$. There are N d-dimensional visual features of labeled training samples in matrix $\boldsymbol{X}_s \in \mathbb{R}^{d \times N}$, and K d-dimensional features of unseen classes in testing set $\boldsymbol{X}_u \in \mathbb{R}^{d \times K}$. For auxiliary semantic space, given a set $\boldsymbol{A}_s \in \mathbb{R}^{l \times s}$, which represents the corresponding class-level s l-dimensional attributes of seen classes, and similarly, $\boldsymbol{A}_u \in \mathbb{R}^{l \times u}$ represents that of unseen classes. Standard inductive setting, which is employed by our proposed method, assumes that $\boldsymbol{X}_s$, $\boldsymbol{A}_s$ and $\boldsymbol{A}_u$ are known in advance, and the goal is to recognize unseen samples $\boldsymbol{X}_u$.

3.2. Framework

The general idea of our approach is to address projection domain shift problem and make the predictive model more discriminative on GZSL. Therefore,

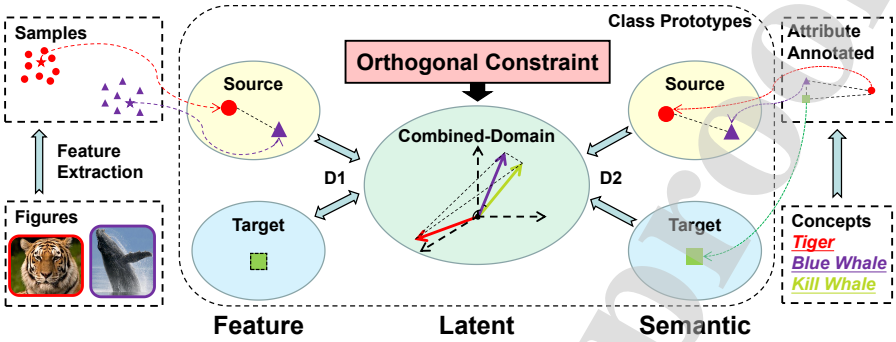


Figure 2: Illustration of the proposed DDIP framework.

semantic prototypes of target domain $\boldsymbol{A}_u$ should also be taken full usage. We first learn the visual prototypes with a minimization of Mean Square Error (MSE), and then define an intermediate latent space, where a dictionary learning method is employed to learn the prototypes with the projection constraints from both visual space and semantic space. Furthermore, we apply an orthogonal constraint in the latent space to make the learned latent prototypes more discriminative. The main framework of our proposed DDIP is illustrated in Fig. 2, and the detailed description of approach can be found in the following subsections.

3.2.1. Visual Prototype Learning

Our DDIP aims to perform classification on the class prototypes in all spaces. Since the latent prototypes should be learned with the projection from both visual prototypes and semantic prototypes, and the class prototypes in semantic space can be directly obtained as $\boldsymbol{A}_s$ and $\boldsymbol{A}_u$, the class prototypes in visual space should be firstly computed. We have the label of each seen samples and their corresponding class prototypes $\boldsymbol{A}_s$, so the prototype of each class can be easily calculated as the average value of all samples from the same category. But there is an obvious weakness, the prototypes obtained in this way may be unrepresentative, therefore we consider to relax this kind of hard constraint and put it into automatically learning style, similar as that in [19], which can be

written as,

$$\mathcal{L}_p = \min_{\boldsymbol{P}_s} \|\boldsymbol{X}_s - \boldsymbol{P}_s \boldsymbol{T}\|_F^2, \tag{1}$$

where $\|\bullet\|_F^2$ is the Frobenius norm of a matrix, $\boldsymbol{P}_s \in \mathbb{R}^{d \times s}$ is the seen class prototypes that learn from visual space and $\boldsymbol{T}$ represents an aggregation of one-hot vectors, and each one-hot vector denotes the label of its corresponding sample.

3.2.2. Domain Combination

In the context of GZSL, most existing methods are devoted into conducting a direct projection from feature to semantic space. Since the distributions of data from source and target domains are often different, such practice often causes domain shift problem. Different from them, and inspired by [20, 19], in this paper we formulated the learning of projection as a dictionary learning problem. Each visual feature element can be considered as an instance corresponding to an attribute, for example, in visual space, the weight of a basis visual feature, like whether an animal 'has tail', can be a corresponding coefficient in dictionary, and represented by referring to it. Different from [20], which build up two dictionaries for source and target domains respectively, our DDIP learns two dictionaries too, but one is for visual to latent space projection and another is for semantic to latent space, thus in our proposed DDIP, both source and target domains share the same dictionary, thence the two domains can be linked in the latent space. Compared with those methods that do not take full usage of the target prototypes, our model is prone to become more generalizable for target domain, and can significantly alleviate the projection domain shift problem. Due to the two projection directions in framework and each direction has samples from source and target domain, thus the loss function in this part contains four terms and can be written as following,

$$\begin{aligned} \mathcal{L}_d = \min_{\boldsymbol{P}_s, \boldsymbol{P}_u, \boldsymbol{D}_1, \boldsymbol{D}_2, \boldsymbol{C}_s, \boldsymbol{C}_u} & \|\boldsymbol{P}_s - \boldsymbol{D}_1 \boldsymbol{C}_s\|_F^2 + \lambda \|\boldsymbol{A}_s - \boldsymbol{D}_2 \boldsymbol{C}_s\|_F^2 \\ & + \alpha(\|\boldsymbol{P}_u - \boldsymbol{D}_1 \boldsymbol{C}_u\|_F^2 + \lambda \|\boldsymbol{A}_u - \boldsymbol{D}_2^T \boldsymbol{C}_u\|_F^2) \\ & + \mu_1 \|\boldsymbol{D}_1\|_F^2 + \mu_2 \|\boldsymbol{D}_2\|_F^2, \end{aligned} \tag{2}$$

where, $\boldsymbol{P}_s$ is the visual prototypes of seen classes obtained in the previous part, $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ denote the two dictionaries respectively. $\boldsymbol{C}_s \in \mathbb{R}^{p\times s}$, $\boldsymbol{C}_u \in \mathbb{R}^{p\times u}$ denote the representation of prototypes in the latent space of combined domains. Then, $\boldsymbol{P}_u \in \mathbb{R}^{d\times u}$ indicates the automatic learned class prototypes of target domain in visual space, and these prototypes can help further optimizing the dictionaries. μ_1 and μ_2 control the relative importance of the two regularization terms.

3.2.3. Orthogonal Constraint

Orthogonal constraint submodule is the most important part of our architecture, where a relationship is built between source and target domains to make all prototypes in the latent space of both domains more discriminative. Since class prototypes should be normalized in combined domains firstly, the orthogonal prototypes in latent space can be considered as been projected into the same hyper-spherical space, as shown in Fig. 1. For example, in *'Original'* hyper-spherical space, it can be found that the class prototypes of *'Blue Whale'* and *'Killer Whale'* are very close to each other. Due to the fact that making all the class prototypes far away from each other is impossible on a constrained hyper-spherical space, but letting them have equal distance is the best way to encourage them more separable, *e.g.*, as shown in *'After'* of Fig. 1, restricting all the prototype vectors be orthogonal to each other can be the best and simplest way. Among prior works, Zhang *et al.* has used this kind of constraint in [50], and they conducted it only on source domain in the proposed framework. By contrast, our DDIP applies the orthogonal constraint on combined domains, which is the first time a specific relative relationship between prototypes from source and target domains has been built. This relationship can make prototypes more identifiable and easier to be differentiated. Furthermore, our model is built upon the combined domains, thus it is more suitable for GZSL.

We denote $\boldsymbol{c}^i$ as one of the prototypes in latent space of combined domains. The orthogonal constraint here is that each prototype $\boldsymbol{c}^i$ is normalized and orthogonal to the others, *i.e.*, the inner product of $\boldsymbol{c}^i$ and $\boldsymbol{c}^j (i \neq j)$ equals to 0,

and ${\boldsymbol{c}^i}^T\boldsymbol{c}^i$ equals to 1. Concretely, we first concatenate the seen prototype $\boldsymbol{C}_s$ and unseen prototype $\boldsymbol{C}_u$ as $[\boldsymbol{C}_s, \boldsymbol{C}_u]$, then the loss function can be represented as,

$$\mathcal{L}_o = \min_{\boldsymbol{C}_s, \boldsymbol{C}_u} \|[\boldsymbol{C}_s, \boldsymbol{C}_u]^T[\boldsymbol{C}_s, \boldsymbol{C}_u] - \boldsymbol{I}\|_F^2, \tag{3}$$

by expanding Eq. 3, we can obtain,

$$\mathcal{L}_o = \min_{\boldsymbol{C}_s, \boldsymbol{C}_u} \left|\left| \begin{bmatrix} \boldsymbol{C}_s^T\boldsymbol{C}_s & \boldsymbol{C}_s^T\boldsymbol{C}_u \\ \boldsymbol{C}_u^T\boldsymbol{C}_s & \boldsymbol{C}_u^T\boldsymbol{C}_u \end{bmatrix} - \begin{bmatrix} \boldsymbol{I} & 0 \\ 0 & \boldsymbol{I} \end{bmatrix} \right|\right|_F^2. \tag{4}$$

Then, the final loss function of this submodule can be written as,

$$\mathcal{L}_o = \min_{\boldsymbol{C}_s, \boldsymbol{C}_u} \|\boldsymbol{C}_s^T\boldsymbol{C}_s - \boldsymbol{I}\|_F^2 + \|\boldsymbol{C}_u^T\boldsymbol{C}_u - \boldsymbol{I}\|_F^2 + \|\boldsymbol{C}_s^T\boldsymbol{C}_u\|_F^2. \tag{5}$$

Combing all the aforementioned items, the final objective function is,

$$\mathcal{L} = \mathcal{L}_d + \beta\mathcal{L}_p + \gamma\mathcal{L}_o, \tag{6}$$

where, the relative importance of the three items are controlled by β and γ.

3.3. Optimization

The final objective function can be written as follows according to Eq. 1-Eq. 6,

$$\begin{aligned} \mathcal{L} = \min_{\boldsymbol{P}_s, \boldsymbol{P}_u, \boldsymbol{D}_1, \boldsymbol{D}_2, \boldsymbol{C}_s, \boldsymbol{C}_u} & \|\boldsymbol{P}_s - \boldsymbol{D}_1\boldsymbol{C}_s\|_F^2 + \lambda\|\boldsymbol{A}_s - \boldsymbol{D}_2\boldsymbol{C}_s\|_F^2 \\ & + \alpha(\|\boldsymbol{P}_u - \boldsymbol{D}_1\boldsymbol{C}_u\|_F^2 + \lambda\|\boldsymbol{A}_u - \boldsymbol{D}_2^T\boldsymbol{C}_u\|_F^2) + \mu_1\|\boldsymbol{D}_1\|_F^2 \\ & + \mu_2\|\boldsymbol{D}_2\|_F^2 + \beta\|\boldsymbol{X}_s - \boldsymbol{P}_s\boldsymbol{T}\|_F^2 + \gamma(\|\boldsymbol{C}_s^T\boldsymbol{C}_s - \boldsymbol{I}\|_F^2 \\ & + \|\boldsymbol{C}_u^T\boldsymbol{C}_u - \boldsymbol{I}\|_F^2 + \|\boldsymbol{C}_s^T\boldsymbol{C}_u\|_F^2). \end{aligned} \tag{7}$$

Eq. 7 is not simultaneously convex for $\boldsymbol{P}_s$, $\boldsymbol{P}_u$, $\boldsymbol{D}_1$, $\boldsymbol{D}_2$, $\boldsymbol{C}_s$, $\boldsymbol{C}_u$, but it is convex for only one variable when fixing the others, thus we can conduct the optimization with an iterative strategy.

3.3.1. Initialization

First, we calculate the similarities between unseen prototypes to seen ones as the initialization of $\boldsymbol{C}_u$ and the average vector of samples in each class as the

initialization of $\boldsymbol{P}_s$. Second, we initialize $\boldsymbol{D}_2$ by the forth and the last terms of Eq. 2. Third, $\boldsymbol{C}_s$ can be initialized by the second and the last terms of Eq. 2. Subsequently, we get $\boldsymbol{D}_1$ by the first and the second last terms of Eq. 2. The last variable $\boldsymbol{P}_u$ can be obtained by the third term of Eq. 2.

3.3.2. Optimization

We conduct optimization via an iterative strategy as follows,

(1) Update $\boldsymbol{P}_s$. By fixing the other variables, the subproblem can be written as,

$$\boldsymbol{P}_s = \arg\min_{\boldsymbol{P}_s} \|\boldsymbol{P}_s - \boldsymbol{D}_1\boldsymbol{C}_s\|_F^2 + \beta\|\boldsymbol{X}_s - \boldsymbol{P}_s\boldsymbol{T}\|_F^2. \tag{8}$$

By conducting the derivative of the right part of Eq. 8 with respect to $\boldsymbol{P}_s$, and setting it to 0, then the closed-form solution of Eq. 8 can be obtained as,

$$\boldsymbol{P}_s = (\beta\boldsymbol{X}_s\boldsymbol{T}^T + \boldsymbol{D}_1\boldsymbol{C}_s)(\boldsymbol{I} + \boldsymbol{T}\boldsymbol{T}^T)^{-1}. \tag{9}$$

(2) Update $\boldsymbol{C}_s$. By fixing the other variables, the subproblem can be represented as,

$$\begin{aligned}\boldsymbol{C}_s =& \min_{\boldsymbol{C}_s} \|\boldsymbol{P}_s - \boldsymbol{D}_1\boldsymbol{C}_s\|_F^2 + \lambda\|\boldsymbol{A}_s - \boldsymbol{D}_2\boldsymbol{C}_s\|_F^2 \\ &+ \gamma(\|\boldsymbol{C}_s^T\boldsymbol{C}_s - \boldsymbol{I}\|_F^2 + \|\boldsymbol{C}_s^T\boldsymbol{C}_u\|_F^2).\end{aligned} \tag{10}$$

By conducting the derivative of the right part of Eq. 10 with respect to $\boldsymbol{C}_s$ and setting it to 0, we can obtain,

$$\begin{aligned}&(\boldsymbol{D}_1^T\boldsymbol{D}_1 + \lambda\boldsymbol{D}_2^T\boldsymbol{D}_2 + 2\gamma\boldsymbol{C}_s\boldsymbol{C}_s^T - 2\gamma\boldsymbol{I} + \gamma\boldsymbol{C}_u\boldsymbol{C}_u^T)\boldsymbol{C}_s \\ =&(\boldsymbol{D}_1^T\boldsymbol{P}_s + \lambda\boldsymbol{D}_2^T\boldsymbol{A}_s).\end{aligned} \tag{11}$$

Since the Eq. 11 contains the cubic term of $\boldsymbol{C}_s$, a complex computation is required to obtain its solution. Thus, we use a iterative approximate solution to replace it,

$$\begin{aligned}\boldsymbol{C}_s = (\boldsymbol{D}_1^T\boldsymbol{D}_1 + \lambda\boldsymbol{D}_2^T\boldsymbol{D}_2 + 2\gamma\boldsymbol{C}_s\boldsymbol{C}_s^T - 2\gamma\boldsymbol{I} + \gamma\boldsymbol{C}_u\boldsymbol{C}_u^T)^{-1} \\ (\boldsymbol{D}_1^T\boldsymbol{P}_s + \lambda\boldsymbol{D}_2^T\boldsymbol{A}_s).\end{aligned} \tag{12}$$

It should be noted that the $\boldsymbol{C}_s$ on the right side is an abbreviation of $\boldsymbol{C}_s^{t-1}$, where t indicates the current number of iterations, and $\boldsymbol{C}_s$ on the left side stands for the result of current iteration $\boldsymbol{C}_s^t$.

(3) Update $\boldsymbol{D}_1$. We first fix the other variables, then set the derivation with respect to $\boldsymbol{D}_1$ and let it to 0, we can obtain the closed-form solution as,

$$\boldsymbol{D}_1 = (\boldsymbol{P}_s\boldsymbol{C}_s^T + \alpha\boldsymbol{P}_u\boldsymbol{C}_u^T)(\boldsymbol{C}_s\boldsymbol{C}_s^T + \boldsymbol{C}_u\boldsymbol{C}_u^T + \mu_1\boldsymbol{I})^{-1}. \tag{13}$$

(4) Update $\boldsymbol{D}_2$. By fixing the other variables, we can obtain the closed-form solution similar as computing $\boldsymbol{D}_1$,

$$\boldsymbol{D}_2 = (\lambda\boldsymbol{A}_s\boldsymbol{C}_s^T + \alpha\lambda\boldsymbol{A}_u\boldsymbol{C}_u^T)(\lambda\boldsymbol{C}_s\boldsymbol{C}_s^T + \alpha\lambda\boldsymbol{C}_u\boldsymbol{C}_u^T + \mu_2\boldsymbol{I})^{-1}. \tag{14}$$

(5) Update $\boldsymbol{C}_u$. We first fix the other variables, and then, Similar as computing $\boldsymbol{C}_s$, we can obtain the iterative approximate solution as,

$$\begin{aligned}\boldsymbol{C}_u = (\alpha\boldsymbol{D}_1^T\boldsymbol{D}_1 + \alpha\lambda\boldsymbol{D}_2^T\boldsymbol{D}_2 + 2\gamma\boldsymbol{C}_u\boldsymbol{C}_u^T - 2\gamma\boldsymbol{I} + \gamma\boldsymbol{C}_s\boldsymbol{C}_s^T)^{-1} \\ (\alpha\boldsymbol{D}_1^T\boldsymbol{P}_u + \alpha\lambda\boldsymbol{D}_2^T\boldsymbol{A}_u),\end{aligned} \tag{15}$$

where, $\boldsymbol{C}_u$ on the right side is an abbreviation of $\boldsymbol{C}_u^{t-1}$, and $\boldsymbol{C}_u$ on the left side stands for $\boldsymbol{C}_u^t$.

(6) Update $\boldsymbol{P}_u$. By fixing the other variables, we can easily obtain the closed-form solution as,

$$\boldsymbol{P}_u = \boldsymbol{D}_1\boldsymbol{C}_u. \tag{16}$$

Based on the above analysis, the learning algorithm is outlined in Algorithm 1.

3.4. Generalized Zero Shot Recognition

Since there are three distinct spaces in our DDIP, visual (v), semantic (s) and latent hyper-spherical (d) spaces, and we have obtained class prototypes in any of them respectively, thus we can make the test in any spaces. In our method, instead of simply making the final classification result in one single

Algorithm 1 Inference of the training phase

Input: The training data $\boldsymbol{X}_s$, the aggregation of its one-hot corresponding labels $\boldsymbol{T}$, the class prototypes of seen and unseen classes $\boldsymbol{A}_s$ and $\boldsymbol{A}_u$;
The hyper-parameters $\alpha, \beta, \gamma, \mu_1, \mu_2, \lambda$;
The iterative number *iter*.

Output: The learned dictionary $\boldsymbol{D}_1, \boldsymbol{D}_2$, and class prototypes in visual and hyper-spherical space $\boldsymbol{P}_s, \boldsymbol{P}_u$ and $\boldsymbol{C}_s, \boldsymbol{C}_u$.

Initialize $\boldsymbol{D}_1, \boldsymbol{D}_2, \boldsymbol{P}_s, \boldsymbol{P}_u, \boldsymbol{C}_s, \boldsymbol{C}_u$ with the strategy described in Sec. 3.3.1;
for $K = 1 \rightarrow iter$ **do**
    Update $\boldsymbol{P}_s$ based on Eq. 9;
    Update $\boldsymbol{C}_s$ based on Eq. 12;
    Update $\boldsymbol{D}_1$ based on Eq. 13;
    Update $\boldsymbol{D}_2$ based on Eq. 14;
    Update $\boldsymbol{C}_u$ based on Eq. 15;
    Update $\boldsymbol{P}_u$ based on Eq. 16;
end for
return $\boldsymbol{D}_1, \boldsymbol{D}_2, \boldsymbol{P}_s, \boldsymbol{P}_u, \boldsymbol{C}_s, \boldsymbol{C}_u$.

space, we proposed a Global Evaluation that classify a new sample based on a comprehensive evaluation of the probability distributions in all spaces.

Concretely, we firstly obtain the *Similarity* (Sim) between test sample and class prototypes in each space and then utilize these Sims to calculate the probability (p^i) of test sample belongs to each class with Softmax in individual or multiple spaces.

3.4.1. In Visual Space

Since we have obtained class prototypes of unseen classes $\boldsymbol{P}_u$ and seen classes $\boldsymbol{P}_s$ in visual space, we can directly calculate the similarities $\boldsymbol{Sim}_v$ with cosine distance between each class prototype and test sample $\boldsymbol{x}_i$ by combining them into $\boldsymbol{P}_{all} = \boldsymbol{P}_s \cup \boldsymbol{P}_u$ as our search space for GZSL. Thereafter, the probability of sample $\boldsymbol{x}_i$ belongs to M^{th} class in visual space p_v^{iM} can be obtained by applying

softmax function. Concretely, the similarities and probability can be obtained with the following formulation,

$$\begin{cases} \boldsymbol{Sim}_v^i = \boldsymbol{x}_i^T \boldsymbol{P}_{all}, \\ p_v^{iM} = \dfrac{e^{Sim_v^{iM}}}{\sum_{c \in (\mathcal{U} \cup \mathcal{S})} e^{Sim_v^{ic}}}, \end{cases} \tag{17}$$

where, Sim_v^{ic} means the similarity between the sample $\boldsymbol{x}_i$ and the c^{th} prototype in visual space. In addition, we should change $\boldsymbol{P}_{all}$ to $\boldsymbol{P}_u$, and the search space to $\mathcal{U}$ when making prediction on standard ZSL setting,.

3.4.2. In Hyper-spherical Space

Firstly, since we have built up a dictionary D_1 as a bridge between visual to hyper-spherical space, we can get the representations of test samples in this space by the following formulation,

$$\arg\min \|\boldsymbol{x}_i - D_1 \boldsymbol{c}_i\|_F^2 + \theta \|\boldsymbol{c}_i\|_F^2, \tag{18}$$

where, $\boldsymbol{c}_i$ represents the corresponding representation of test samples in hyper-spherical space. θ is the balancing coefficient parameter for the second item. Subsequently, Since $\boldsymbol{C}_s$ and $\boldsymbol{C}_u$ have been learned from the model, same as the approach employed in visual space, we first combine them as $\boldsymbol{C}_{all}$ and calculate the similarities $\boldsymbol{Sim}_d^i$ between $\boldsymbol{c}_i$ and $\boldsymbol{C}_{all}$, and then compute the probability $\boldsymbol{p}_d^i$ in hyper-spherical space by employing softmax function on $\boldsymbol{Sim}_d^i$ with the similar formulation as Eq. 17.

3.4.3. In Semantic Space

After the computation in latent space, we can get the representation of test samples as $\boldsymbol{c}_i$. In addition, we also have the semantic dictionary $\boldsymbol{D}_2$, thus, the sample representation in semantic space can be obtained as,

$$\boldsymbol{a}_i = \boldsymbol{D}_2 \boldsymbol{c}_i. \tag{19}$$

Then, the similarities in semantic space can be calculated by using $\boldsymbol{a}_i$ and $\boldsymbol{A}_{all}$ as same as the method in above two spaces, where $\boldsymbol{A}_{all} = \boldsymbol{A}_s \cup \boldsymbol{A}_u$.

Subsequently, the probability $\boldsymbol{p}_s^i$ in Semantic space can be easily evaluated with softmax.

3.4.4. In Multiple-Space

Vectors in visual space can take along with more visual information, while vectors in semantic space contain more semantic information, and latent spherical is a compromised space that has both the pros of the two spaces, thus making evaluation in multiple-space can improve the performance, including $v+s, v+d, d+s, v+d+s$. To be specific, we normalize the similarities in three spaces and directly add them, *e.g.*, $\boldsymbol{Sim}_{vs} = \boldsymbol{Sim}_v + \boldsymbol{Sim}_s$. Then, these new similarities can be used to obtain the probability values in multiple-space, *e.g.*, p_{vs}^i, same as the above strategy with Eq. 17.

3.4.5. Global Evaluation

At last, a Global Evaluation is designed to comprehensively predict the test samples by using the maximum probabilities of each sample $\boldsymbol{x}_i$ belongs to each class in all spaces. For example, if we have the probability of test sample $\boldsymbol{x}_i$ belongs to M^{th} class in respective space, *i.e.*, p_v^{iM}, p_d^{iM}, p_s^{iM}, p_{vs}^{iM}, p_{ds}^{iM}, p_{vd}^{iM}, p_{vds}^{iM}, we can get the global probability p_{max}^{iM} belonging to M^{th} class from the maximum of them, and then classify the test sample $\boldsymbol{x}_i$ by the following formulation,

$$\ell_i = \arg\max_{c \in (\mathcal{U} \cup \mathcal{S})} p_{max}^{ic}. \tag{20}$$

3.5. Computational Complexity

In training stage, our DDIP is optimized in an iterative strategy, so we compute the algorithm complexity of single iteration first. Optimizing every variable in our model only needs the executions of matrix multiplication and inversion. Specifically, Updating $\boldsymbol{P}_s$ requires the computational complexity of $\mathcal{O}(Nds)$, next, updating $\boldsymbol{C}_s$ and $\boldsymbol{C}_u$ need the same complexity of $\mathcal{O}(dp^2)$. Moreover, for updating $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$, it will cost the same complexity of $\mathcal{O}(sp^2)$, at last, updating $\boldsymbol{P}_u$ costs $\mathcal{O}(dps)$. Thus, the total computational complexity is

Algorithm 2 Inference of the class prediction

Input: The testing data $\boldsymbol{x}_i$, the manual annotation of seen and unseen classes $\boldsymbol{A}_s$ and $\boldsymbol{A}_u$, the learned dictionary $\boldsymbol{D}_1, \boldsymbol{D}_2$, and class prototypes in visual and hyper-spherical space $\boldsymbol{P}_s, \boldsymbol{P}_u$ and $\boldsymbol{C}_s, \boldsymbol{C}_u$.

Output: Classification result ℓ_i.

1: Combine $\boldsymbol{A}_s$ and $\boldsymbol{A}_u$, $\boldsymbol{P}_s$ and $\boldsymbol{P}_u$, $\boldsymbol{C}_s$ and $\boldsymbol{C}_u$ into $\boldsymbol{A}_{all}$, $\boldsymbol{P}_{all}$ and $\boldsymbol{C}_{all}$;
2: Project $\boldsymbol{x}_i$ into hyper-spherical and semantic spaces to get its representation in each space by $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ based on Eq. 18 and Eq. 19;
3: Calculate the similarities between test sample and class prototypes in each space, $\boldsymbol{Sim}_{v,s,d}$;
4: Normalize the $\boldsymbol{Sim}_{v,s,d}$ and add them in pairs to obtain $\boldsymbol{Sim}_{vs,vd,ds,vds}$;
5: Utilize these $\boldsymbol{Sim}$s to obtain the probability of test sample $\boldsymbol{x}_i$ belongs to M^{th} class in respective space p_v^{iM}, p_d^{iM}, p_s^{iM}, p_{vs}^{iM}, p_{ds}^{iM}, p_{vd}^{iM}, p_{vds}^{iM};
6: Get the maximum of above probabilities as p_{max}^{iM};
7: Calculate the maximum probabilities of test sample $\boldsymbol{x}_i$ belongs to each class $p_{max}^{ic}(c \in (\mathcal{U} \cup \mathcal{S})$;
8: Classify test sample $\boldsymbol{x}_i$ based on Eq. 20.

$\mathcal{O}(k(Nds+dp^2+sp^2+dps))$, where k represents the number of iterations. Due to the fact that the feature dimension d is usually bigger than the number of seen classes s, and the latent dimension p is bigger than s, so the final computational complexity is $max(\mathcal{O}(kdp^2), \mathcal{O}(kdNs))$. In the test stage, the computational complexity in three spaces are $\mathcal{O}(d(u+s))$, $\mathcal{O}(p(u+s))$,$\mathcal{O}(l(u+s))$ respectively, thus the final computational complexity is $max(\mathcal{O}(d(u+s)), \mathcal{O}(p(u+s)))$ for prediction.

4. Experiments

4.1. Datasets

In our experiments, we employ four popular benchmark and a large-scale datasets, *i.e.*, SUN (SUN attribute) [33], CUB (Caltech-UCSD-Birds 200-2011)

[41], AWA(Animals with Attributes) [22], aPY(Attribute Pascal and Yahoo) [11] and a large-scale dataset ImageNet [35]. SUN is a dataset of fine-grained complex visual scenes and CUB is of bird-species images. In addition, we also employ two coarse grained datasets, AWA and aPY, AWA is consisted of different animal pictures and the training 20 classes in aPY are known from Pascal VOC [10] and 12 classes collected from Yahoo! [11] are used for testing. The other details of these datasets can be found in Tab. 1. 'SS' refers to number of Seen Samples in training, 'TS' is the number of samples from unseen classes for test, while 'TR' is for seen ones. ImageNet is a large-scale dataset, which has totally 25400 images and 1000-dim class-level attributes. 1000 classed of ILSVRC 2012 are used as seen classes and 360 classes of ILSVRC 2010 that are not used in ILSVRC 2012 are used as unseen classes. In addition, we adopt the split strategy which is proposed by [44].

4.2. Experimental Setting

We exploit the extracted features with ResNet-101 [18] as our input, and the same attributes employed in the evaluation in [44]. Additionally, there are six hyper-parameters α, β, λ, γ, μ_1 and μ_2 in our method. Due to the fact that different dataset is often suitable with different parameters, thus we fine-tune our hyper-parameters in the range [0.01, 0.1, 1, 10, 100] by employing a cross validation strategy. To be specific, we hereby compare the difference of ZSL cross-validation to conventional machine learning approaches. Compared to inner-splits of training samples within each class, ZSL problem requires inter-splits by in turn regarding part of seen classes as unseen, for example, 20% of the seen classes are selected as the validational unseen classes in our experiments, and the parameters of best average performance of 5 executions are picked as the optimal parameters for each dataset. It should be noted that the parameters may not be the most suitable for the test set, because the labels of test data are strictly inaccessible during training.

Table 1: Summary of the five datasets.

Datasets	Dimension		Class Number		Samples Number		
	Feat.	*Att.*	*Seen*	*Unseen*	*SS*	*TS*	*TR*
SUN[33]	2048	102	645	72	10320	1440	2580
CUB[41]	2048	312	150	50	7057	2967	10320
AWA[22]	2048	85	40	10	19832	4958	5685
aPY[11]	2048	64	20	12	5932	7924	1483
ImageNet[35]	1024	1000	1000	360	20000	-	5400

4.3. Results on GZSL

The evaluation criteria employed to evaluate our model under GZSL setting is the harmonic mean H, which can be calculated by

$$H = \frac{2 \times acc_{tr} \times acc_{ts}}{acc_{tr} + acc_{ts}}, \tag{21}$$

where, acc_{tr} and acc_{ts} are the accuracies of test samples from seen classes and unseen classes respectively, and we adopt the following average per-class top-1 accuracy as the final result, which can be written as,

$$acc_{\mathcal{C}} = \frac{1}{|\mathcal{C}|} \sum_{c=1}^{|\mathcal{C}|} \frac{\#\text{ correct predictions in } c}{\#\text{ samples in } c}, \tag{22}$$

where, $|\mathcal{C}|$ is the number of test classes $\mathcal{C}$. The experimental results on all four datasets are recorded in Tab. 2. Due to the fact that our DDIP is linear, we make the comparison with some other linear state-of-the-art methods.

From Tab. 2, it is obviously that our DDIP can outperform all the other baselines method on the the most important metric H, especially on AWA and aPY. Specifically, we improve H by 4.7% on SUN, 2.0% on CUB, 18.5% on AWA and 19.4% on APY respectively. Obviously, the fundamental reason for the improvement of H is the raising of accuracies of test samples from unseen classes ts. Compared with those existing methods, which only train models with the samples from source domain and are over-fitting on tr, our proposed DDIP can

Table 2: Comparison of our DDIP and state-of-the-art methods on GZSL. Bold font stands for the best result of the corresponding column. '-' means not reported.

	SUN			CUB			AWA			APY		
Method	*ts*	*tr*	*H*	*ts*	*tr*	*H*	*ts*	*tr*	*H*	*ts*	*tr*	*H*
DAP[22]	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[31]	6.8	**39.9**	11.6	1.6	**72.2**	3.1	0.4	88.6	0.8	0.0	**91.2**	0.0
LATEM[43]	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[2]	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[13]	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[3]	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[34]	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[7]	7.0	43.4	13.4	11.5	70.9	19.8	8.9	87.3	16.2	7.4	66.3	13.3
SAE[21]	8.8	18.0	11.8	7.8	54.0	13.6	1.8	77.1	3.5	1.1	82.2	2.2
CDL [19]	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[40]	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[4]	18.8	33.1	23.9	21.8	73.6	33.7	23.8	67.0	35.1	-	-	-
PSEUDO[27]	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[49]	21.0	31.0	25.1	24.2	63.9	35.1	18.3	79.3	29.8	11.9	76.3	20.5
TRIPLE[50]	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 [42]	15.2	23.8	18.6	17.1	37.1	23.8	22.3	77.5	34.6	8.4	75.5	15.1
LESAE [26]	21.9	34.7	26.9	24.3	53.0	33.3	19.1	70.2	30.0	12.7	56.1	20.1
LESD [9]	15.2	19.8	17.2	14.6	38.5	21.2	12.6	71.0	21.4	11.8	49.3	19.0
Ours	**36.8**	27.7	**31.6**	**36.6**	37.5	**37.1**	**53.6**	65.9	**59.1**	**37.3**	65.6	**47.5**

obtain a much more balance result on *ts* and *tr*. We ascribe this improvement to the advantage of our DDIP, which combines both source and target domains in latent space, and makes the learned prototypes more discriminative from each other by the constrain of orthogonal.

Table 3: Comparison of our DDIP and state-of-the-art methods on ZSL. Bold font stands for the best result of the corresponding column. '-' means not reported.

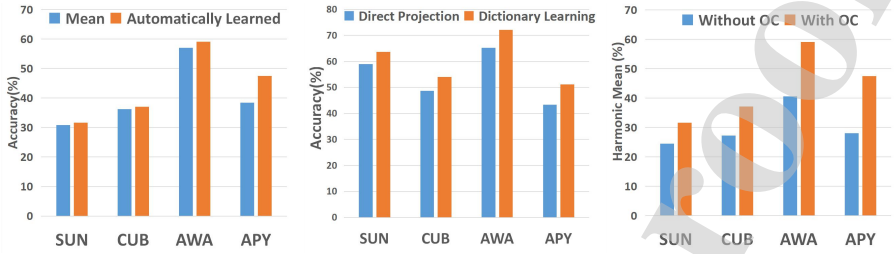
Method(%)	**SUN**	**CUB**	**AWA**	**aPY**	**Average**
DAP[22]	39.9	40.0	44.1	33.8	39.5
CONSE[31]	38.8	34.3	45.6	26.9	36.4
LATEM [43]	55.3	49.3	55.1	35.2	48.7
ALE [2]	58,1	54.9	59.9	39.7	53.2
DEVISE[13]	56.5	52.0	54.2	39.8	50.6
SJE [3]	53.7	53.9	65.6	32.9	51.5
ESZSL[34]	54.5	53.9	58.2	38.3	51.2
SYNC[7]	56.3	55.6	54.0	23.9	47.5
SAE[21]	40.3	33.3	43.0	8.3	46.6
CDL [19]	63.6	54.5	69.9	43.0	57.8
GFZSL [40]	62.5	42.0	55.6	32.8	48.2
LAGO[4]	57.5	**57.8**	-	-	-
PSEUDO[27]	60.4	57.2	66.2	40.4	56.1
KERNEL[49]	61.7	57.1	71.0	45.3	58.8
TRIPLE[50]	59.3	54.9	64.7	40.9	55.0
VZSL [42]	52.0	43.8	63.7	30.3	47.5
LESAE [26]	60.0	53.9	66.1	40.8	55.2
LESD [9]	50.4	38.9	53.4	29.8	43.1
Ours	**63.7**	54.0	**72.1**	**51.2**	**60.2**

Table 4: ZSL results on ImageNet dataset. Bold font stands for the best result of the corresponding column.

Method(%)	**Top-1**	**Top-5**
DEVISE[13]	5.2	12.8
AMP[16]	6.1	13.1
CONSE[31]	7.8	15.5
ESZSL[34]	8.3	18.2
EMBED[54]	11.0	25.7
SAE[21]	12.9	27.2
Ours	**13.1**	**27.9**

4.4. Results on ZSL

Since many methods give the results only on ZSL, we here also conduct the same experiment to show the priority of our method. For four benchmark datasets, similar with the experiment on GZSL, we adopt the average per-class top-1 accuracy as the final accuracy for ZSL, and the final results can be found in Tab. 3. It can be clearly observed that our method obtains the best performance on SUN, AWA and APY and a comparable result on CUB. To be specific, our DDIP can outperform the listed best methods by 0.1% on SUN, 1.1% on AWA and 5.9% on APY respectively. As for CUB, we get a little bit lower result. This phenomenon is caused by the fact that CUB is a fine-grained dataset and the discriminative features of CUB are local, while our method focuses on global prototypes. Despite that, our method still can obtain 54% classification accuracy, and just has 3.8% lower than the best method LAGO [4]. However, we believe that a good method should perform well on every dataset rather than just on a single one, thus we also compare the average performance on different datasets, and record the result in the last column of Tab. 3. According to the average values, we can clearly find the average performance of our DDIP can outperform all the state-of-the-art methods, which indicates the effectiveness of our approach. For large-scale dataset ImageNet, we test Top-1 and Top-5 accuracy under ZSL setting. As shown in Tab. 4, our method outperforms the



(a) Average or Automatically Learning

(b) Direct Projection or Dictionary Learning

(c) With/Without Orthogonal Constraint (OC).

Figure 3: Influence of each submodule.

listed best methods by 0.2% on Top-1 and 0.7% on Top-5, demonstrating the scalability of our model to large-scale problems.

4.5. Ablation Study

4.5.1. Effect of Each Submodule

In this section, we conduct experiments to show the effect of dictionary learning and orthogonal constraint respectively.

Firstly, we remove the Visual Prototype Learning submodule. Instead of automatically learning, we let the average value of each class as its corresponding class prototype representation, and the results under convention ZSL setting can be found in Fig. 3(a). It can be obviously found that the prototypes that are learned from Visual Prototype Learning submodule can generate better recognition results, especially on APY. By conducting the average as the initialization value and adding minor adjustments automatically based on that, the prototypes in visual space can become more representative.

Secondly, we replace the dictionary learning part into direct projection by utilizing an embedding matrix and test the model under ZSL setting, the results can be found in Fig. 3(b). It is clear that the recognition performances of dictionary learning are better than those of direct projection on all four datasets, which indicates that sharing the same dictionary for both seen and unseen classes can effectively deal with the problem that data from seen and unseen classes

Table 5: Influence of Orthogonal Constrain on Different Categories

Settings \ Datasets	SUN	CUB	AWA	APY
None	24.7	27.4	40.6	28.1
Unseen Only	24.5	27.7	44.6	43.1
Seen Only	26.6	34.2	46.9	46.9
Seen + Unseen	**31.6**	**37.1**	**59.1**	**47.5**

always has different distribution, that is to say, our model can significantly alleviate the projection domain shift problem.

Thirdly, in order to investigate the influence of Orthogonal Constraint, we train our model with and without this constraint, test them under the GZSL setting, and record the results in Fig. 3(c). From the figure, we can clearly see that training with this constraint can dramatically boost the harmonic mean on all four datasets. The orthogonal constraint on latent hyper-spherical space can make the both seen and unseen class prototypes we learned become more discriminative, thus the harmonic mean, which is calculated by accuracy of both seen and unseen categories, can make a great improvement. Especially on AWA and APY, due to the fact that these two datasets are both coarse-grained, in other words, making the prototypes of coarse-grained datasets orthogonal is much easier than that of a fine-grained one, *e.g.*, CUB only contains images of birds, thus the effect with this constraint can be less conspicuous than that on AWA and APY.

4.5.2. Influence of Orthogonal Constraint on Different Categories

To further illustrate the detail effectiveness of our proposed orthogonal constraint on combined domain in latent hyper-spherical space, in this section, we conduct experiments to investigate whether orthogonal constraint on different categories can make different performance. To be specific, we employ two variants of our model: constraint on unseen class only and constraint on seen class only. Tab. 5 records the performance of these two variants on GZSL setting,

and the results with the original full model and the model without orthogonal constraints can also be found in Tab. 5. It is obvious that employing the orthogonal constraint of unseen or seen classes only can enhance the performance but the improvement is limited, while applying constraint on both seen and unseen classes can boost the final results significantly on all four datasets. We ascribe this phenomenon to the orthogonality of combined domains, which indicates that employing orthogonal constraint on seen or unseen classes individually is fragmentary, while combining them and taking the relationship between them into account is much more suitable for GZSL.

4.5.3. Different Dimensions of Latent Space

In this section, we conduct experiment to show whether the dimension of the latent hyper-spherical space has an effect on the final recognition results. The results on four datasets are illustrated in Fig. 4, where the X-axis represents the multiple of dimension of latent space relative to the number of categories. To be specific, we simply change the initialization method of $\boldsymbol{C}_u$ by concatenating K $\boldsymbol{C}_u$, then the dimension of latent space can be changed to K times to the original one. From Tab. 4, it is obvious that there is a common characteristic on the four datasets: H can reach the peak when the dimension is around two times to the number of categories, and it can be lower when dimension is too low or too high. Since we need at least same dimension as the category number to make all classes orthogonal to each other in latent space, the dimension cannot be too low. Besides, too high dimension may bring too much redundant information.

4.5.4. Performance in Different Spaces

As described above, there are three different spaces in our model, and each of them can be taken as the testing space. Therefore, in this section we illustrate the testing results in different spaces separately and the effect of our proposed Global Evaluation on four datasets in Fig. 5. Bars on the left show the accuracies on ZSL setting and the right ones record the evaluation criteria H on GZSL. Each dataset has four bars, which from left to right means the

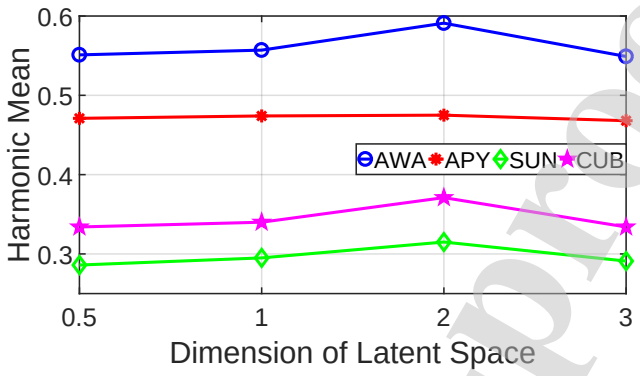
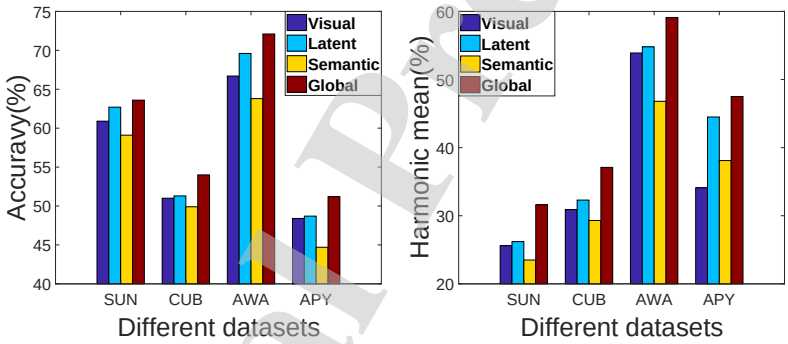


Figure 4: Comparison with different dimensions for latent space.



(a) Accuracy on ZSL (b) Harmonic mean

Figure 5: An illustration of the results of ZSL&GZSL on four datasets in different spaces.

testing results in Visual, Latent (hyper-spherical), Semantic and results of our Globel Evaluation. It can be obviously found that compared with the results in visual and hyper-spherical spaces (see Semantic vs. Visual and Latent), the test results in semantic space obtains the worst performance. This phenomenon is caused by that the original manual class prototypes in semantic space lack discriminative properties. Also the results in Latent space (see Latent vs. Visual and Semantic) are always better than those in other two spaces, which also indicates that the class prototypes we learned in latent space are more discriminative. For GZSL, performance in latent hyper-spherical space is also better

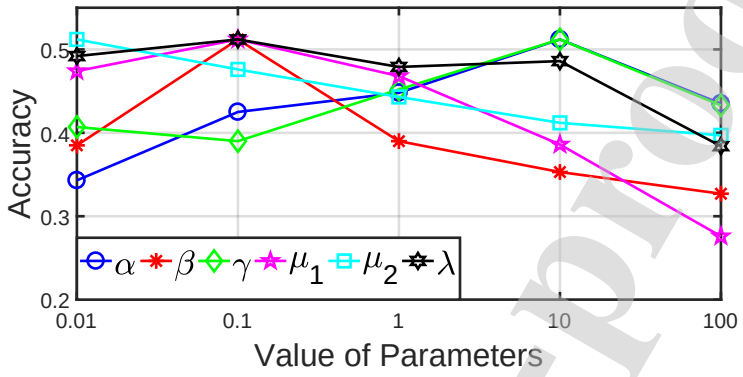


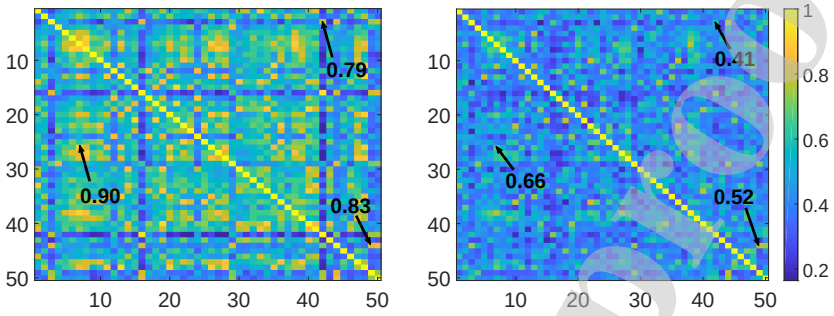
Figure 6: ZSL accuracies (Y-axis) with different value of hyper-parameters(X-axis).

than that in other two spaces, which reveals that our hyper-spherical space is generalizable to the whole domain. Moreover, by taking our proposed Global Evaluation (see Global vs. Others), the recognition performance in ZSL and GZSL can be further improved, which can be ascribed to the combination of the advantages of all three spaces.

4.5.5. Parameters Analysis

There are six hyper-parameters in our proposed method, α, β, λ, γ, μ_1 and μ_2, in order to discuss the robustness of using different values of each hyper-parameter, in this section, we take APY as an example to show the influence of each hyper-parameter. To be specific, since we set all the hyper-parameters range from 1×10^{-2} to 1×10^{2}, and the stride is ten times, in this experiment, we fix the five of six hyper-parameters on the optimal value, which is [10, 0.1, 10, 0.1, 0.01, 0.1] for APY, and change the left one. We record the result on ZSL setting, and draw the curves in Fig. 6.

Generally speaking, different hyper-parameters often mean different importance of each constraint, thus they often lead to different performance, which can be seen in Fig. 6. However, we can still clearly find that the performance is kept stable within a wide range, which proves that our method can be robust



(a) Semantic space (b) Latent space

Figure 7: The cosine similarities of class prototypes in latent hyper-spherical space on AWA. Best viewed in color.

for using different values of hyper-parameters.

4.6. Similarities of Class Prototypes

Since we all know that the more different the class prototypes are from each other, the easier the input samples can be classified, we illustrate the change of similarities of class prototypes in this section. We calculate the cosine similarity of each class prototype on AWA, and visualize the similarity matrix in Fig. 7. Specifically, vectors from 0# to 40# in matrix are seen classes prototypes and the rest belong to unseen classes. Fig. 7(a) demonstrates the original manual prototypes in semantic space, while Fig. 7(b) illustrates the class prototypes that our DDIP learned in latent hyper-spherical space. From the comparison of two figures, we can obviously found that the prototypes we learned are much more differentiable from each other, which demonstrates the effectiveness of our model. Noted that not only seen classes become more discriminative against seen, seen against unseen, and unseen against unseen also become more discriminative. For example, we pick out three pairs, the left one is the similarity between *'weasel'*(Seen) and *'hamster'*(Seen), the upper one denotes the similarity of *'Killer Whale'* (Seen) and *'Blue Whale'* (Unseen) and the last one is *'Walrus'* (Unseen) to *'Seal'* (Unseen). These three pairs are very similar under the attributes of manual annotation, over 0.78 in cosine similarity, and it

Table 6: The time cost of train and test of different methods on AWA.

Method	Train Time(s)	Test Time(s)
SSE[56]	2981	20.72
ESZSL[34]	53	**0.21**
AMP[16]	1936	0.43
SAE[21]	4.9	0.27
Ours	**1.4**	0.73

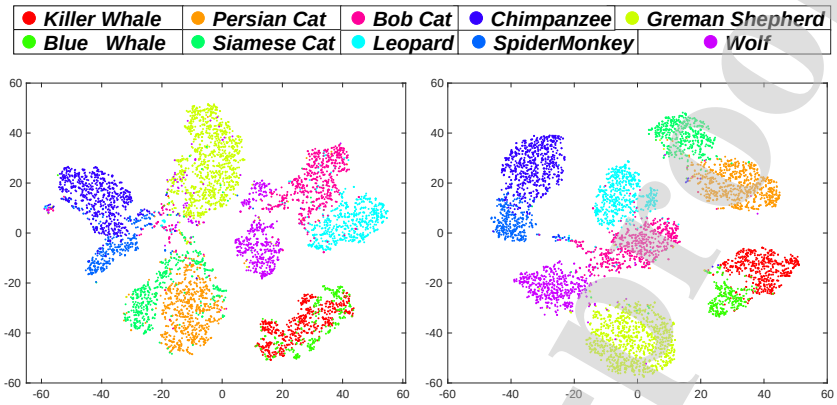
is difficult to classify them directly using original attribute, while our model can make them much more different and the similarities between them decrease to less than 0.67, which shows the superiority of our submodule of orthogonal constraint.

4.7. Computational Cost Analysis

Since our DDIP is a linear model, the train and test can be efficient. Therefore, in this section we make a comparison with some other no-deep models on the efficiency on AWA and the results are illustrated in Tab. 6. Due to the fact that our model is consist of simple matrix addition, subtraction, multiplication and division only and without Sylvester function, which costs the most time in SAE [21], thus our model is much faster than SAE. Since our model makes evaluations on three independent and four combined spaces, the test time can be a sightly longer, but the train phase of our model is much faster than the other methods, which means our model can be extended to larger datasets.

4.8. Visualization in Latent Space

The objective of orthogonal constraint in latent hyper-spherical space is to disperse all classes and make them more discriminative. Thus, in order to have a more intuitive understanding, we employ t-SNE [29] on AWA to illustrate the distributions of samples in this space. Specifically, we choose representative class pairs whose cosine similarities of prototypes in semantic space is rank high on the list, *i.e.*, they are very similar and hard to be classified. After that,



(a) without OC (b) with OC

Figure 8: Visualization of similar classes on AWA in latent space, where 'OC' means Orthogonal Constraint. Best viewed in color.

we finally get five pairs, including eight seen classes and two unseen classes, which can be found in the legend of Fig. 8. In Fig. 8, we illustrate the data distributions of the samples from the selected classes with and without the orthogonal constraint in the latent space. From this figure, it can be clearly seen that the samples of 'Killer Whale' (Seen) and 'Blue Whale' (Unseen) are overlapped without orthogonal constraint, while our DDIP can separate them effectively. This phenomenon can also be found in seen-seen pairs, *e.g.*, 'Persian Cat' and 'Siamese Cat', which indicates our DDIP can perform well not only in the source domain, but also in the whole domain.

5. Conclusion

In this paper, we proposed a novel and effective GZSL recognition model named DDIP, which aims to alleviate the influence of projection domain shift problem. Specifically, to construct relationship between the prototypes of seen classes and unseen classes, we define a effective latent hyper-spherical space to combine both source domain and target domain. In addition, a sparse coding is employed to learning the dictionaries to project features and semantics into latent space, and an orthogonal constraint is also applied in latent space to

make the prototypes more discriminative. We developed an iterative optimizing algorithm to solve the proposed DDIP method, and conducted extensive experiments on four popular datasets. The results on both GZSL and ZSL show that our DDIP can outperform the state-of-the-art methods, which demonstrate the superiority of our method.

6. Acknowledgement

This work was supported in part by National Natural Science Foundation of China (NSFC) under Grants No. 61872187 and No. 61929104, in part by the Medical Research Council (MRC) Innovation Fellowship (UK) under Grant No. MR/S003916/1, and in part by the "111" Program under Grant No.B13022.

References

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

[2] 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.

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

[4] Atzmon, Y., Chechik, G., 2018. Probabilistic and-or attribute grouping for zero-shot learning. arXiv preprint arXiv:1806.02664 .

[5] Bi, J., Zhang, C., 2018. An empirical comparison on state-of-the-art multi-class imbalance learning algorithms and a new diversified ensemble learning scheme. Knowledge-Based Systems 158, 81–93.

[6] Capitaine, H.L., 2018. Constraint selection in metric learning. Knowledge-Based Systems 146, 91 – 103.

[7] Changpinyo, S., Chao, W.L., Gong, B., Sha, F., 2016. Synthesized classifiers for zero-shot learning, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)., pp. 5327–5336.

[8] Chao, W.L., Soravit, C., Gong, B., Sha, F., 2016. An empirical study and analysis of generalized zero-shot learning for object recognition in the wild, in: European Conference on Computer Vision, pp. 52–68.

[9] Ding, Z., Shao, M., Fu, Y., 2017. Low-rank embedded ensemble semantic dictionary for zero-shot learning, in: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2050–2058.

[10] Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A., 2010. The pascal visual object classes (voc) challenge. International journal of computer vision 88, 303–338.

[11] Farhadi, A., Endres, I., Hoiem, D., Forsyth, D., 2009. Describing objects by their attributes, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)., pp. 1778–1785.

[12] Ferrari, V., Zisserman, A., 2008. Learning visual attributes, in: Advances in neural information processing systems, pp. 433–440.

[13] 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.

[14] 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, pp. 584–599.

[15] Fu, Y., Wang, X., Dong, H., Jiang, Y.G., Wang, M., Xue, X., Sigal, L., 2019. Vocabulary-informed zero-shot and open-set learning. IEEE transactions on pattern analysis and machine intelligence .

[16] Fu, Z., Xiang, T., Kodirov, E., Gong, S., 2015. Zero-shot object recognition by semantic manifold distance, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)., pp. 2635–2644.

[17] Hayashi, T., Fujita, H., 2020. Cluster-based zero-shot learning for multivariate data. arXiv preprint arXiv:2001.05624 .

[18] He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: Proceedings of The IEEE Conference on Computer Vision and Pattern Recognition., pp. 770–778.

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

[20] 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.

[21] Kodirov, E., Xiang, T., Gong, S., 2017. Semantic autoencoder for zero-shot learning, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)., pp. 3174–3183.

[22] 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.

[23] Li, X., Fang, M., Feng, D., Li, H., Wu, J., 2018. Learning unseen visual prototypes for zero-shot classification. Knowledge-Based Systems 160, 176 – 187.

[24] Li, Y., Wang, D., Hu, H., Lin, Y., Zhuang, Y., 2017. Zero-shot recognition using dual visual-semantic mapping paths, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3279–3287.

[25] Liu, B., Yao, L., Ding, Z., Xu, J., Wu, J., 2018a. Combining ontology and reinforcement learning for zero-shot classification. Knowledge-Based Systems 144, 42 – 50.

[26] Liu, Y., Gao, Q., Li, J., Han, J., Shao, L., 2018b. Zero shot learning via low-rank embedded semantic autoencoder., in: International Joint Conference on Artificial Intelligence, pp. 2490–2496.

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

[28] Long, Y., Liu, L., Shen, F., Shao, L., Li, X., 2018b. Transductive zero-shot learning with a self-training dictionary approach. IEEE transactions on pattern analysis and machine intelligence 40, 2498–2512.

[29] Maaten, L.v.d., Hinton, G., 2008. Visualizing data using t-sne. Journal of machine learning research 9, 2579–2605.

[30] Nguyen, B., Morell, C., Baets, B.D., 2018. Distance metric learning for ordinal classification based on triplet constraints. Knowledge-Based Systems 142, 17 – 28.

[31] 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. International Conference on Learning Representation (ICLR) .

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

[33] 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.

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

[35] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al., 2015. Imagenet large scale visual recognition challenge. International journal of computer vision 115, 211–252.

[36] Sanodiya, R.K., Mathew, J., 2019. A framework for semi-supervised metric transfer learning on manifolds. Knowledge-Based Systems 176, 1 – 14.

[37] Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition, in: International Conference on Learning Representation (ICLR).

[38] Socher, R., Ganjoo, M., Sridhar, H., Bastani, O., Manning, C.D., Ng, A.Y., 2013. Zero-shot learning through cross-modal transfer, in: Advances in neural information processing systems, pp. 935–943.

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

[40] Verma, V.K., Rai, P., 2017. A simple exponential family framework for zero-shot learning, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, pp. 792–808.

[41] Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S., 2011. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001. California Institute of Technology.

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

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

[44] Xian, Y., Schiele, B., Akata, Z., 2017. Zero-shot learning-the good, the bad and the ugly, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)., pp. 4582–4591.

[45] Xiao, Q., Dai, J., Luo, J., Fujita, H., 2019. Multi-view manifold regularized learning-based method for prioritizing candidate disease mirnas. Knowledge-Based Systems 175, 118–129.

[46] Yu, Y., Ji, Z., Li, X., Guo, J., Zhang, Z., Ling, H., Wu, F., 2018. Transductive zero-shot learning with a self-training dictionary approach. IEEE transactions on cybernetics 48, 2908–2919.

[47] Zabihzadeh, D., Monsefi, R., Yazdi, H.S., 2019. Sparse bayesian approach for metric learning in latent space. Knowledge-Based Systems 178, 11 – 24.

[48] Zhang, C., Bi, J., Xu, S., Ramentol, E., Fan, G., Qiao, B., Fujita, H., 2019a. Multi-imbalance: An open-source software for multi-class imbalance learning. Knowledge-Based Systems 174, 137–143.

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

[50] Zhang, H., Long, Y., Guan, Y., Shao, L., 2019b. Triple verification network for generalized zero-shot learning. IEEE Transactions on Image Processing 28, 506–517.

[51] Zhang, H., Long, Y., Liu, L., Shao, L., 2019c. Adversarial unseen visual feature synthesis for zero-shot learning. Neurocomputing 329, 12–20.

[52] Zhang, H., Long, Y., Yang, W., Shao, L., 2019d. Dual-verification network for zero-shot learning. Information Sciences 470, 43–57.

[53] Zhang, H., Mao, H., Long, Y., Yang, W., Shao, L., 2019e. A probabilistic zero-shot learning method via latent nonnegative prototype synthesis of unseen classes. IEEE Transactions on Neural Networks and Learning Systems doi:10.1109/TNNLS.2019.2955157.

[54] Zhang, L., Xiang, T., Gong, S., 2017. Learning a deep embedding model for zero-shot learning, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)., pp. 2021–2030.

[55] Zhang, Z., Saligrama, V., 2015a. Zero-shot learning via joint latent similarity embedding, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6034–6042.

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

[57] Zhao, A., Ding, M., Guan, J., Lu, Z., Xiang, T., Wen, J.R., 2018. Domain-invariant projection learning for zero-shot recognition, in: Advances in Neural Information Processing Systems 31, pp. 1019–1030.

[58] Zhou, F., Yang, S., Fujita, H., Chen, D., Wen, C., 2020. Deep learning fault diagnosis method based on global optimization gan for unbalanced data. Knowledge-Based Systems 187, 104837.

Credit Author Statement

Yinduo Wang: Writing- Original draft preparation, Software;

Haofeng Zhang: Conceptualization, Methodology;

Zheng Zhang: Visualization, Investigation;

Yang Long: Data Curation, Editing;

Ling Shao: Writing- Reviewing and Supervision:

Yinduo Wang received the B.S. degree in Electronic Engineering from Huazhong University of Science and Technology, Wuhan, 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 semantic segmentation, zero shot learning and deep learning.

Dr. Haofeng Zhang currently is an Associate Professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology. He was an academic visitor with the School of Computing Sciences, University of East Anglia, Norwich, U.K., in 2017. He received the B.Eng. and the Ph.D. degrees from the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China, in 2003 and 2007, respectively. His research interests include computer vision, deep learning, and mobile robotics.

Dr. Zheng Zhang received the M.S and Ph.D. degree from Harbin Institute of Technology in 2014 and 2018, respectively. Currently, he is an assistant professor with the Bio-computing Center in Harbin Institute of Technology, Shenzhen, China. Dr. Zhang was a Postdoctoral Research Fellow in The University of Queensland and a Research Associate in The Hong Kong Polytechnic University respectively in the past two years. He has authored or co-authored over 20 technical papers published at prestigious international journals and conferences. His current research interests include machine learning and computer vision.

Dr. Yang Long is currently an Assistant Professor at Durham University, Durham, UK. He was a Research Fellow with OpenLab, School of Computing, Newcastle University from July 2018 to Jun 2019. 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 Health-care Data Science.

Prof. Ling Shao 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, and several other journals. He is a fellow of the International Association of Pattern Recognition (IAPR), the Institution of Engineering and Technology (IET) and the British Computer Society (BCS).

Declaration of interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Paper Title:

Learning Discriminative Domain-Invariant Prototypes for Generalized Zero Shot Learning

Authors:

Yinduo Wang, Haofeng Zhang, Zheng Zhang, Yang Long, Ling Shao