Learning a Deep Embedding Model for Zero-Shot Learning

Li Zhang Tao Xiang Shaogang Gong
Queen Mary University of London
{david.lizhang, t.xiang, s.gong}@qmul.ac.uk

Abstract

Zero-shot learning (ZSL) models rely on learning a joint embedding space where both textual/semantic description of object classes and visual representation of object images can be projected to for nearest neighbour search. Despite the success of deep neural networks that learn an end-to-end model between text and images in other vision problems such as image captioning, very few deep ZSL model exists and they show little advantage over ZSL models that utilise deep feature representations but do not learn an end-to-end embedding. In this paper we argue that the key to make deep ZSL models succeed is to choose the right embedding space. Instead of embedding into a semantic space or an intermediate space, we propose to use the visual space as the embedding space. This is because that in this space, the subsequent nearest neighbour search would suffer much less from the hubness problem and thus become more effective. This model design also provides a natural mechanism for multiple semantic modalities (e.g., attributes and sentence descriptions) to be fused and optimised jointly in an end-to-end manner. Extensive experiments on four benchmarks show that our model significantly outperforms the existing models. Code is available at: https://github.com/lzrobots/DeepEmbeddingModel_ZSL

1. Introduction

A recent trend in developing visual recognition models is to scale up the number of object categories. However, most existing recognition models are based on supervised learning and require a large amount (at least 100s) of training samples to be collected and annotated for each object class to capture its intra-class appearance variations [7]. This severely limits their scalability – collecting daily objects such as chair is easier, but many other categories are rare (*e.g.*, a newly identified specie of beetle on a remote pacific island). None of these models can work with few or even no training samples for a given class. In contrast, humans are very good at recognising objects without seeing any visual samples, i.e., *zero-shot learning* (ZSL). For example, a child would have no problem recognising a zebra if she has seen horses before and *also* read elsewhere that a zebra is a horse but with black-and-white stripes on it. Inspired by humans' ZSL ability, recently there is a surge of interest in machine ZSL [3, 54, 25, 1, 40, 46, 11, 34, 12, 15, 27, 52, 37, 5, 14, 4, 6, 55, 56].

A zero-shot learning method relies on the existence of a labelled training set of *seen classes* and the knowledge about how an *unseen class* is semantically related to the seen classes. Seen and unseen classes are usually related in a high dimensional vector space, called semantic space, where the knowledge from seen classes can be transferred to unseen classes. The semantic spaces used by most early works are based on semantic attributes [9, 10, 35]. Given a defined attribute ontology, each class name can be represented by an attribute vector and termed as a class *prototype*. More recently, semantic word vector space [46, 11] and sentence descriptions/captions [37] have started to gain popularity. With the former, the class names are projected into a word vector space so that different classes can be compared, whilst with the latter, a neural language model is required to provide a vector representation of the description.

With the semantic space and a visual feature representation of image content, ZSL is typically solved in two steps: (1) A joint embedding space is learned where both the semantic vectors (prototypes) and the visual feature vectors can be projected to; and (2) nearest neighbour (NN) search is performed in this embedding space to match the projection of an image feature vector against that of an unseen class prototype. Most state-of-the-arts ZSL models [12, 14, 3, 4, 40, 54, 25] use deep CNN features for visual feature representation; the features are extracted with pretrained CNN models. They differ mainly in how to learn the embedding space given the features. They are thus not end-to-end deep learning models.

In this paper, we focus on end-to-end learning of a deep embedding based ZSL model which offers a number of advantages. First, end-to-end optimisation can potentially lead to learning a better embedding space. For example, if sentence descriptions are used as the input to a neural language model such as recurrent neural networks (RNNs)

for computing a semantic space, both the neural language model and the CNN visual feature representation learning model can be jointly optimised in an end-to-end fashion. Second, a neural network based joint embedding model offers the flexibility for addressing various transfer learning problems such as multi-task learning and multi-domain learning [52]. Third, when multiple semantic spaces are available, this model can provide a natural mechanism for fusing the multiple modalities. However, despite all these intrinsic advantages, in practice, the few existing end-to-end deep models for ZSL in the literature [27, 11, 46, 52, 37] fail to demonstrate these advantages and yield only weaker or merely comparable performances on benchmarks when compared to non-deep learning alternatives.

We argue that the key to the success of a deep embedding model for ZSL is the choice of the embedding space. Existing models, regardless whether they are deep or non-deep, choose either the semantic space [25, 14, 46, 11] or an intermediate embedding space [27, 3, 40, 12] as the embedding space. However, since the embedding space is of high dimension and NN search is to be performed there, the hubness problem is inevitable [36], that is, a few unseen class prototypes will become the NNs of many data points, i.e., hubs. Using the semantic space as the embedding space means that the visual feature vectors need to be projected into the semantic space which will shrink the variance of the projected data points and thus aggravate the hubness problem [36, 8].

In this work, we propose a novel Deep neural network based Mmbedding Model (DEM) for ZSL which differs from existing models in that: (1) To alleviate the hubness problem, we use the output visual feature space of a CNN subnet as the embedding space. The resulting projection direction is from a semantic space, e.g., attribute or word vector, to a visual feature space. Such a direction is opposite to the one adopted by most existing models. We provide a theoretical analysis and some intuitive visualisations to explain why this would help us counter the hubness problem. (2) A simple yet effective multi-modality fusion method is developed in our neural network model which is flexible and importantly enables end-to-end learning of the semantic space representation.

The contributions of this work are as follows: (i) A novel deep embedding model for ZSL has been formulated which differs from existing models in the selection of embedding space. (ii) A multi-modality fusion method is further developed to combine different semantic representations and to enable end-to-end learning of the representations. Extensive experiments carried out on four benchmarks including AwA [25], CUB [49] and large scale ILSVRC 2010 and ILSVRC 2012 [7] show that our model beats all the state-of-the-art models presented to date, often by a large margin.

2. Related Work

Semantic space Existing ZSL methods differ in what semantic spaces are used: typically either attribute [9, 10, 35, 47], word vector [46, 11], or text description [37, 53]. It has been shown that an attribute space is often more effective than a word vector space [3, 54, 25, 40]. This is hardly surprising as additional attribute annotations are required for each class. Similarly, state-of-the-art results on fine-grained recognition tasks have been achieved in [37] using image sentence descriptions to construct the semantic space. Again, the good performance is obtained at the price of more manual annotation: 10 sentence descriptions need to be collected for each image, which is even more expensive than attribute annotation. This is why the word vector semantic space is still attractive: it is 'free' and is the only choice for large scale recognition with many unseen classes [14]. In this work, all three semantic spaces are considered.

Fusing multiple semantic spaces Multiple semantic spaces are often complementary to each other; fusing them thus can potentially lead to improvements in recognition performance. Score-level fusion is perhaps the simplest strategy [15]. More sophisticated multi-view embedding models have been proposed. Akata et al. [3] learn a joint embedding semantic space between attribute, text and hierarchical relationship which relies heavily on hyperparameter search. Multi-view canonical correlation analysis (CCA) has also been employed [12] to explore different modalities of testing data in a transductive way. Differing from these models, our neural network based model has an embedding layer to fuse different semantic spaces and connect the fused representation with the rest of the visual-semantic embedding network for end-to-end learning. Unlike [12], it is inductive and does not require to access the whole test set at once.

Embedding model Existing methods also differ in the visual-semantic embedding model used. They can be categorised into two groups: (1) The first group learns a mapping function by regression from the visual feature space to the semantic space with pre-computed features [25, 14] or deep neural network regression [46, 11]. For these embedding models, the semantic space is the embedding space. (2) The second group of models implicitly learn the relationship between the visual and semantic space through a common intermediate space, again either with a neural network formulation [27, 52] or without [27, 3, 40, 12]. The embedding space is thus neither the visual feature space, nor the semantic space. We show in this work that using the visual feature space as the embedding space is intrinsically advantageous due to its ability to alleviate the hubness problem.

Deep ZSL model All recent ZSL models use deep CNN features as inputs to their embedding model. However, few are deep end-to-end models. Existing deep neural network

based ZSL works [11, 46, 27, 52, 37] differ in whether they use the semantic space or an intermediate space as the embedding space, as mentioned above. They also use different losses. Some of them use margin-based losses [11, 52, 37]. Socher *et al* [46] choose a euclidean distance loss. Ba *et al* [27] takes a dot product between the embedded visual feature and semantic vectors and consider three training losses, including a binary cross entropy loss, hinge loss and Euclidean distance loss. In our model, we find that the least square loss between the two embedded vectors is very effective and offers an easy theoretical justification as for why it copes with the hubness problem better. The work in [37] differs from the other models in that it integrates a neural language model into its neural network for end-to-end learning of the embedding space as well as the language model. In additional to the ability of jointly learning the neural language model and embedding model, our model is capable of fusing text description with other semantic spaces and achieves better performance than [37].

The hubness problem The phenomenon of the presence of 'universal' neighbours, or hubs, in a high-dimensional space for nearest neighbour search was first studied by Radovanovic et al. [29]. They show that hubness is an inherent property of data distributions in a high-dimensional vector space, and a specific aspect of the curse of dimensionality. A couple of recent studies [8, 44] noted that regression based zero-shot learning methods suffer from the hubness problem and proposed solutions to mitigate the hubness problem. Among them, the method in [8] relies on the modelling of the global distribution of test unseen data ranks w.r.t. each class prototypes to ease the hubness problem. It is thus transductive. In contrast, the method in [44] is inductive: It argued that least square regularised projection functions make the hubness problem worse and proposed to perform reverse regression, i.e., embedding class prototypes into the visual feature space. Our model also uses the visual feature space as the embedding space but achieve so by using an end-to-end deep neural network which yields far superior performance on ZSL.

3. Methodology

3.1. Problem definition

Assume a labelled training set of N training samples is given as $\mathcal{D}_{tr} = \{(\mathbf{I}_i,\ \mathbf{y}_i^u,\ t_i^u), i = 1, \ldots, N\}$, with associated class label set $\mathcal{T}_{tr}$, where $\mathbf{I}_i$ is the i-th training image, $\mathbf{y}_i^u \in \mathbb{R}^{L\times 1}$ is its corresponding L-dimensional semantic representation vector, $t_i^u \in \mathcal{T}_{tr}$ is the u-th training class label for the i-th training image. Given a new test image $\mathbf{I}_j$, the goal of ZSL is to predict a class label $t_j^v \in \mathcal{T}_{te}$, where t_j^v is the v-th test class label for the j-th test instance. We have $\mathcal{T}_{tr} \cap \mathcal{T}_{te} = \varnothing$, i.e., the training (seen) classes and test (unseen) classes are disjoint. Note that each class label t^u or t^v is associated with a pre-defined semantic space representation $\mathbf{y}^u$ or $\mathbf{y}^v$ (e.g. attribute vector), referred to as semantic class prototypes. For the training set, $\mathbf{y}_i^u$ is given because each training image $\mathbf{I}_i$ is labelled by a semantic representation vector representing its corresponding class label t_j^u.

3.2. Model architecture

The architecture of our model is shown in Fig. 1. It has two branches. One branch is the visual encoding branch, which consists of a CNN subnet that takes an image $\mathbf{I}_i$ as input and outputs a D-dimensional feature vector $\phi(\mathbf{I}_i) \in \mathbb{R}^{D\times 1}$. This D-dimensional visual feature space will be used as the embedding space where both the image content and the semantic representation of the class that the image belongs to will be embedded. The semantic embedding is achieved by the other branch which is a semantic encoding subnet. Specifically, it takes a L-dimensional semantic representation vector of the corresponding class $\mathbf{y}_i^u$ as input, and after going through two fully connected (FC) linear + Rectified Linear Unit (ReLU) layers outputs a D-dimensional semantic embedding vector. Each of the FC layer has a l_2 parameter regularisation loss. The two branches are linked together by a least square embedding loss which aims to minimise the discrepancy between the visual feature $\phi(\mathbf{I}_i)$ and its class representation embedding vector in the visual feature space. With the three losses, our objective function is as follows:

$$\mathcal{L}(\mathbf{W}_1, \mathbf{W}_2) = \frac{1}{N}\sum_{i=1}^{N} ||\phi(\mathbf{I}_i) - f_1(\mathbf{W}_2 f_1(\mathbf{W}_1 \mathbf{y}_i^u))||^2 + \lambda(||\mathbf{W}_1||^2 + ||\mathbf{W}_2||^2) \quad (1)$$

where $\mathbf{W}_1 \in \mathbb{R}^{L\times M}$ are the weights to be learned in the first FC layer and $\mathbf{W}_2 \in \mathbb{R}^{M\times D}$ for the second FC layer. λ is the hyperparameter weighting the strengths of the two parameter regularisation losses against the embedding loss. We set $f_1(\cdot)$ to be the Rectified Linear Unit (ReLU) which introduces nonlinearity in the encoding subnet [24].

After that, the classification of the test image $\mathbf{I}_j$ in the visual feature space can be achieved by simply calculating its distance to the embed prototypes:

$$v = \arg\min_v \mathcal{D}(\phi(\mathbf{I}_j), f_1(\mathbf{W}_2 f_1(\mathbf{W}_1 \mathbf{y}^v))) \quad (2)$$

where $\mathcal{D}$ is a distance function, and $\mathbf{y}^v$ is the semantic space vector of the v-th test class prototype.

3.3. Multiple semantic space fusion

As shown in Fig. 1, we can consider the semantic representation and the first FC and ReLU layer together as a

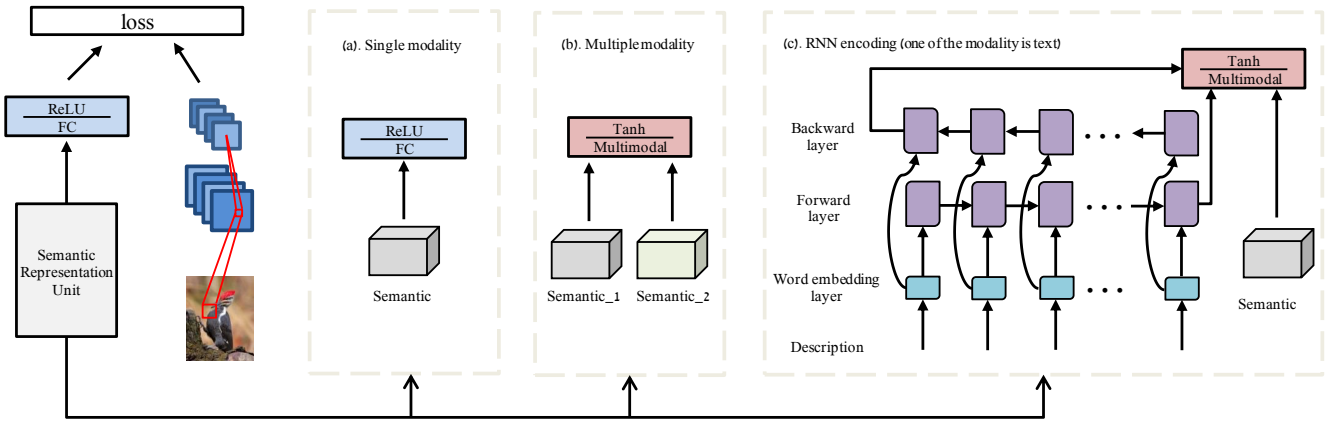


Figure 1: Illustration of the network architecture of our deep embedding model. The detailed architecture of the semantic representation unit in the left branch (semantic encoding subnet) is given in (a), (b) and (c) which correspond to the single modality (semantic space) case, the multiple (two) modality case, and the case where one of the modalities is text description. For the case in (c), the semantic representation itself is a neural network (RNN) which is learned end-to-end with the rest of the network.

semantic representation unit. When there is only one semantic space considered, it is illustrated in Fig. 1(a). However, when more than one semantic spaces are used, e.g., we want to fuse attribute vector with word vector for semantic representation of classes, the structure of the semantic representation unit is changed slightly, as shown in Fig. 1(b).

More specifically, we map different semantic representation vectors to a multi-modal fusion layer/space where they are added. The output of the semantic representation unit thus becomes:

$$f_2(\mathbf{W}_1^{(1)} \cdot \mathbf{y}_i^{u_1} + \mathbf{W}_1^{(2)} \cdot \mathbf{y}_i^{u_2}), \tag{3}$$

where $\mathbf{y}_i^{u_1} \in \mathbb{R}^{L_1 \times 1}$ and $\mathbf{y}_i^{u_2} \in \mathbb{R}^{L_2 \times 1}$ denote two different semantic space representations (e.g., attribute and word vector), "+" denotes element-wise sum, $\mathbf{W}_1^{(1)} \in \mathbb{R}^{L_1 \times M}$ and $\mathbf{W}_1^{(2)} \in \mathbb{R}^{L_2 \times M}$ are the weights which will be learned. $f_2(.)$ is the element-wise scaled hyperbolic tangent function [26]:

$$f_2(x) = 1.7159 \cdot \tanh(\frac{2}{3}x). \tag{4}$$

This activation function forces the gradient into the most non-linear value range and leads to a faster training process than the basic hyperbolic tangent function.

3.4. Bidirectional LSTM encoder for description

The structure of the semantic representation unit needs to be changed again, when text description is avalialbe for each training image (see Fig. 1(c)). In this work, we use a recurrent neural network (RNN) to encode the content of a text description (a variable length sentence) into a fixed-length semantic vector. Specifically, given a text description of T words, $x = (x_1, \ldots, x_T)$ we use a Bidirectional RNN model [42] to encode them. For the RNN cell, the Long-Shot Term Memory (LSTM) [19] units are used as the recurrent units. The LSTM is a special kind of RNN, which introduces the concept of gating to control the message passing between different times steps. In this way, it could potentially model long term dependencies. Following [17], the model has two types of states to keep track of the historical records: a cell state $\mathbf{c}$ and a hidden state $\mathbf{h}$. For a particular time step t, they are computed by integrating the current inputs x_t and previous state $(\mathbf{c}_{t-1}, \mathbf{h}_{t-1})$. During the integrating, three types of gates are used to control the messaging passing: an input gate $\mathbf{i}_t$, a forget gate $\mathbf{f}_t$ and an output gate $\mathbf{o}_t$.

We omit the formulation of the bidirectional LSTM here and refer the readers to [17, 16] for details. With the bidirectional LSTM model, we use the final output as our encoded semantic feature vector to represent the text description:

$$f(\mathbf{W}_{\overrightarrow{\mathbf{h}}} \cdot \overrightarrow{\mathbf{h}} + \mathbf{W}_{\overleftarrow{\mathbf{h}}} \cdot \overleftarrow{\mathbf{h}}), \tag{5}$$

where $\overrightarrow{\mathbf{h}}$ denote the forward final hidden state, $\overleftarrow{\mathbf{h}}$ denote the backward final hidden state. $f(.) = f_1(.)$ if text description is used only for semantic space unit, and $f(.) = f_2(.)$ if other semantic space need to be fused (Sec. 3.3). $\mathbf{W}_{\overrightarrow{\mathbf{h}}}$ and $\mathbf{W}_{\overleftarrow{\mathbf{h}}}$ are the weights which will be learned.

In the testing stage, we first extract text encoding from test descriptions and then average them per-class to form the test prototypes as in [37]. Note that since our ZSL model is a neural network, it is possible now to learn the RNN encoding subnet using the training data together with the rest of the network in an end-to-end fashion.

3.5. The hubness problem

How does our model deal with the hubness problem? First we show that our objective function is closely related to that of the ridge regression formulation. In particular, if we use the matrix form and write the outputs of the semantic representation unit as $\mathbf{A}$ and the outputs of the CNN visual feature encoder as $\mathbf{B}$, and ignore the ReLU unit for now, our training objective becomes

$$\mathcal{L}(\mathbf{W}) = ||\mathbf{B} - \mathbf{W}\mathbf{A}||_F^2 + \lambda||\mathbf{W}||_F^2, \quad (6)$$

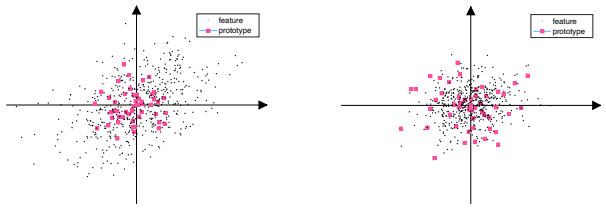
which is basically ridge regression. It is well known that ridge regression has a closed-form solution $\mathbf{W} = \mathbf{B}\mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top + \lambda\mathbf{I})^{-1}$. Thus we have:

$$\begin{aligned} ||\mathbf{W}\mathbf{A}||_2 &= ||\mathbf{B}\mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top + \lambda\mathbf{I})^{-1}\mathbf{A}||_2 \\ &\leq ||\mathbf{B}||_2||\mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top + \lambda\mathbf{I})^{-1}\mathbf{A}||_2 \end{aligned} \quad (7)$$

It can be further shown that

$$||\mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top + \lambda\mathbf{I})^{-1}\mathbf{A}||_2 = \frac{\sigma^2}{\sigma^2 + \lambda} \leq 1. \quad (8)$$

Where σ is the largest singular value of $\mathbf{A}$. So we have $||\mathbf{W}\mathbf{A}||_2 \leq ||\mathbf{B}||_2$. This means the mapped source data $||\mathbf{W}\mathbf{A}||_2$ are likely to be closer to the origin of the space than the target data $||\mathbf{B}||_2$, with a smaller variance.



(a) S → V (b) V → S

Figure 2: Illustration of the effects of different embedding directions on the hubness problem. S: semantic space, and V: visual feature space. Better viewed in colour.

Why does this matter in the context of ZSL? Figure 2 gives an intuitive explanation. Specifically, assuming the feature distribution is uniform in the visual feature space, Fig. 2(a) shows that if the projected class prototypes are slightly shrunk towards the origin, it would not change how hubness problem arises – in other words, it at least does not make the hubness issue worse. However, if the mapping direction were to be reversed, that is, we use the semantic vector space as the embedding space and project the visual feature vectors $\phi(\mathbf{I})$ into the space, the training objective is still ridge regression-like, so the projected visual feature representation vectors will be shrunk towards the origin as shown in Fig. 2(b). Then there is an adverse effect: the semantic vectors which are closer to the origin are more likely to become hubs, i.e. nearest neighbours to many projected visual feature representation vectors. This is confirmed by our experiments (see Sec. 4) which show that using which space as the embedding space makes a big difference in terms of the degree/seriousness of the resultant hubness problem and therefore the ZSL performance.

Measure of hubness To measure the degree of hubness in a nearest neighbour search problem, the *skewness* of the (empirical) N_k distribution is used, following [36, 44]. The N_k distribution is the distribution of the number $N_k(i)$ of times each prototype i is found in the top k of the ranking for test samples (i.e. their k-nearest neighbour), and its skewness is defined as follows:

$$(N_k skewness) = \frac{\sum_{i=1}^{l}(N_k(i) - E[N_k])^3/l}{Var[N_k]^{\frac{3}{2}}}, \quad (9)$$

where l is the total number of test prototypes. A large *skewness* value indicates the emergence of more hubs.

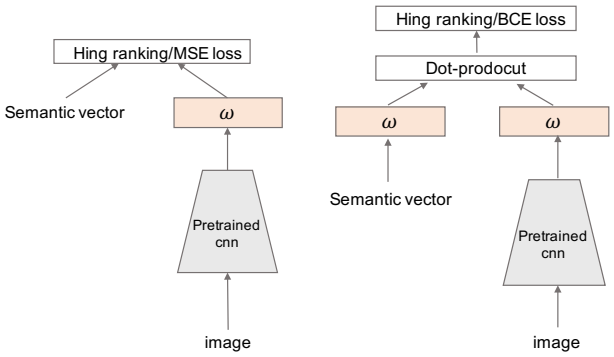
3.6. Relationship to other deep ZSL models

Let's now compare the proposed model with the related end-to-end neural network based models: DeViSE [11], Socher *et al.* [46], MTMDL [52], and Ba *et al.* [27]. Their model structures fall into two groups. In the first group (see Fig. 3(a)), DeViSE [11] and Socher *et al.* [46] map the CNN visual feature vector to a semantic space by a hinge ranking loss or least square loss. In contrast, MTMDL [52] and Ba *et al.* [27] fuse visual space and semantic space to a common intermediate space and then use a hinge ranking loss or a binary cross entropy loss (see Fig. 3(b)). For both groups, the learned embedding model will make the variance of $\mathbf{W}\mathbf{A}$ to be smaller than that of $\mathbf{B}$, which would thus make the hubness problem worse. In summary, the hubness will persist regardless what embedding model is adopted, as long as NN search is conducted in a high dimensional space. Our model does not worsen it, whist other deep models do, which leads to the performance difference as demonstrated in our experiments.

4. Experiments

4.1. Dataset and settings

We follow two ZSL settings: the *old* setting and the new *GBU* setting provided by [51] for training/test splits. Under the *old* setting, adopted by most existing ZSL works before [51], some of the test classes also appear in the ImageNet 1K classes, which have been used to pretrain the image embedding network, thus violating the zero-shot assumption. In contrast, the new *GBU* setting ensures that none of the test classes of the datasets appear in the ImageNet 1K classes. Under both settings, the test set can



(a) [11, 46] (b) [52, 27]

Figure 3: The architectures of existing deep ZSL models fall into two groups: (a) learning projection function ω from visual feature space to semantic space; (b) learning an intermediate space as embedding space.

comprise only the unseen class samples (conventional test set setting) or a mixture of seen and unseen class samples. The latter, termed generalised zero-shot learning (GZSL), is more realistic in practice.

Datasets Four benchmarks are selected for the *old* setting: **AwA** (Animals with Attributes) [25] consists of 30,745 images of 50 classes. It has a fixed split for evaluation with 40 training classes and 10 test classes. **CUB** (CUB-200-2011) [49] contains 11,788 images of 200 bird species. We use the same split as in [3] with 150 classes for training and 50 disjoint classes for testing. **ImageNet (ILSVRC) 2010 1K** [41] consists of 1,000 categories and more than 1.2 million images. We use the same training/test split as [30, 11] which gives 800 classes for training and 200 classes for testing. **ImageNet (ILSVRC) 2012/2010**: for this dataset, we use the same setting as [14], that is, ILSVRC 2012 1K is used as the training seen classes, while 360 classes in ILSVRC 2010 which do not appear in ILSVRC 2012 are used as the test unseen classes. Three datasets [51] are selected for *GBU* setting: **AwA1**, **AwA2** and **CUB**. The newly released AwA2 [51] consists of 37,322 images of 50 classes which is an extension of AwA while AwA1 is same as AwA but under the *GBU* setting.

Semantic space For **AwA**, we use the continuous 85-dimension class-level attributes provided in [25], which have been used by all recent works. For the word vector space, we use the 1,000 dimension word vectors provided in [12, 13]. For **CUB**, continuous 312-dimension class-level attributes and 10 descriptions per image provided in [37] are used. For **ILSVRC 2010** and **ILSVRC 2012**, we trained a skip-gram language model [31, 32] on a corpus of 4.6M Wikipedia documents to extract $1,000$D word vectors for each class.

Model setting and training Unless otherwise specified, We use the Inception-V2 [48, 21] as the CNN subnet in the old and conventional setting, and ResNet101 [18] for the GBU and generalised setting, taking the top pooling units as image embedding with dimension $D = 1024$ and 2048 respectively. The CNN subnet is pre-trained on ILSVRC 2012 1K classification without fine-tuning, the same as the recent deep ZSL works [27, 37]. For fair comparison with DeViSE [11], ConSE [34] and AMP [15] on ILSVRC 2010, we also use the Alexnet [24] architecture and pretrain it from scratch using the 800 training classes. All input images are resized to 224×224. Fully connected layers of our model are initialised with random weights for all of our experiments. Adam [22] is used to optimise our model with a learning rate of 0.0001 and a minibatch size of 64. The model is implemented based on *Tensorflow*.

Parameter setting In the semantic encoding branch of our network, the output size of the first FC layer M is set to 300 and 700 for AwA and CUB respectively when a single semantic space is used (see Fig. 1(a)). Specifically, we use one FC layer for ImageNet in our experiments. For multiple semantic space fusion, the multi-modal fusion layer output size is set to 900 (see Fig. 1(b)). When the semantic representation was encoded from descriptions for the CUB dataset, a bidirectional LSTM encoding subnet is employed (see Fig. 1(c)). We use the `BasicLSTMCell` in *Tensorflow* as our RNN cell and employ ReLU as activation function. We set the input sequence length to 30; longer text inputs are cut off at this point and shorter ones are zero-padded. The word embedding size and the number of LSTM unit are both 512. Note that with this LSTM subnet, RMSprop is used in the place of Adam to optimise the whole network with a learning rate of 0.0001, a minibatch size of 64 and gradient clipped at 5. The loss weighting factor λ in Eq. (1) is searched by five-fold cross-validation. Specifically, 20% of the seen classes in the training set are used to form a validation set.

4.2. Experiments on small scale datasets

Competitors Numerous existing works reported results on AwA and CUB these two relatively small-scale datasets under old setting. Among them, only the most competitive ones are selected for comparison due to space constraint. The selected 13 can be categorised into the non-deep model group and the deep model group. All the non-deep models use ImageNet pretrained CNN to extract visual features. They differ in which CNN model is used: F_O indicates that overfeat [43] is used; F_G for GoogLeNet [48]; and F_V for VGG net [45]. The second group are all neural network based with a CNN subnet. For fair comparison, we implement the models in [11, 46, 52, 27] on AwA and CUB with Inception-V2 as the CNN subnet as in our model and [37]. The compared methods also differ in the semantic spaces

used. Attributes (A) are used by all methods; some also use word vector (W) either as an alternative to attributes, or in conjunction with attributes (A+W). For CUB, recently the instance-level sentence descriptions (D) are used [37]. Note that only inductive methods are considered. Some recent methods [56, 12, 13] are tranductive in that they use all test data at once for model training, which gives them a big unfair advantage.

Comparative results on AwA under old setting From Table 1 we can make the following observations: (1) Our model DEM achieves the best results either with attribute or word vector. When both semantic spaces are used, our result is further improved to 88.1%, which is 7.6% higher than the best result reported so far [55]. (2) The performance gap between our model to the existing neural network based models are particularly striking. In fact, the four models [11, 46, 52, 27] achieve weaker results than most of the compared non-deep models that use deep features only and do not perform end-to-end training. This verify our claim that selecting the appropriate visual-semantic embedding space is critical for the deep embedding models to work. (3) As expected, the word vector space is less informative than the attribute space (86.7% vs. 78.8%) even though our word vector space alone result already beats all published results except for one [55]. Nevertheless, fusing the two spaces still brings some improvement (1.4%).

Comparative results on CUB under old setting Table 1 shows that on the fine-grained dataset CUB, our model also achieves the best result. In particular, with attribute only, our result of 58.3% is 3.8% higher than the strongest competitor [5]. The best result reported so far, however, was obtained by the neural network based DS-SJE [37] at 56.8% using sentence descriptions. It is worth pointing out that this result was obtained using a word-CNN-RNN neural language model, whilst our model uses a bidirectional LSTM subnet, which is easier to train end-to-end with the rest of the network. When the same LSTM based neural language model is used, DS-SJE reports a lower accuracy of 53.0%. Further more, with attribute only, the result of DS-SJE (50.4%) is much lower than ours. This is significant because annotating attributes for fine-grained classes is probably just about manageable; but annotating 10 descriptions for each images is unlikely to scale to large number of classes. It is also evident that fusing attribute with descriptions leads to further improvement.

Comparative results under the GBU setting We follow the evaluation setting of [51]. We compare our model with 13 alternative ZSL models in Table 2. We can see that on AwA1, AwA2 and aPY, the proposed model DEM is particularly strong under the more realistic GZSL setting measured using the harmonic mean (H) metric. In particular, DEM achieves state-of-the-art performance on AwA1, AwA2 and SUN under conventional setting with 68.4%, 67.1% and 61.9%, outperforming alternatives by big margins.

Model	F	SS	AwA	CUB
AMP [15]	F_O	A+W	66.0	-
SJE [3]	F_G	A	66.7	50.1
SJE [3]	F_G	A+W	73.9	51.7
ESZSL [40]	F_G	A	76.3	47.2
SSE-ReLU [54]	F_V	A	76.3	30.4
JLSE [55]	F_V	A	80.5	42.1
SS-Voc [14]	F_O	A/W	78.3/68.9	-
SynC-struct [5]	F_G	A	72.9	54.5
SEC-ML [4]	F_V	A	77.3	43.3
DeViSE [11]	N_G	A/W	56.7/50.4	33.5
Socher *et al.* [46]	N_G	A/W	60.8/50.3	39.6
MTMDL [52]	N_G	A/W	63.7/55.3	32.3
Ba *et al.* [27]	N_G	A/W	69.3/58.7	34.0
DS-SJE [37]	N_G	A/D	-	50.4/**56.8**
DEM	N_G	A/W(D)	**86.7/78.8**	**58.3**/53.5
DEM	N_G	A+W(D)	**88.1**	**59.0**

Table 1: Zero-shot classification accuracy (%) comparison on AwA and CUB (hit@1 accuracy over all samples) under the old and conventional setting. SS: semantic space; A: attribute space; W: semantic word vector space; D: sentence description (only available for CUB). F: how the visual feature space is computed; For non-deep models: F_O if overfeat [43] is used; F_G for GoogLeNet [48]; and F_V for VGG net [45]. For neural network based methods, all use Inception-V2 (GoogLeNet with batch normalisation) [48, 21] as the CNN subnet, indicated as N_G.

4.3. Experiments on ImageNet

Comparative results on ILSVRC 2010 Compared to AwA and CUB, far fewer works report results on the large-scale ImageNet ZSL tasks. We compare our model against 8 alternatives on ILSVRC 2010 in Table 3, where we use hit@5 rather than hit@1 accuracy as in the small dataset experiments. Note that existing works follow two settings. Some of them [33, 20] use existing CNN model (e.g. VGG/GoogLeNet) pretrained from ILSVRC 2012 1K classes to initialise their model or extract deep visual feature. Comparing to these two methods under the same setting, our model gives 60.7%, which beats the nearest rival PDDM [20] by over 12%. For comparing with the other 6 methods, we follow their setting and pretrain our CNN subnet from scratch with Alexnet [24] architecture using the 800 training classes for fair comparison. The results show that again, significant improvement has been obtained with our model.

Comparative results on ILSVRC 2012/2010 Even fewer published results on this dataset are available. Table 4 shows that our model clearly outperform the state-of-the-art

Model	AwA1				AwA2				CUB				aPY				SUN			
	ZSL	GZSL			ZSL	GZSL			ZSL	GZSL			ZSL	GZSL			ZSL	GZSL		
	T1	u	s	H	T1	u	s	H	T1	u	s	H	T1	u	s	H	T1	u	s	H
DAP [25]	44.1	0.0	88.7	0.0	46.1	0.0	84.7	0.0	40.0	1.7	67.9	3.3	33.8	4.8	78.3	9.0	39.9	4.2	25.1	7.2
IAP [25]	35.9	2.1	78.2	4.1	35.9	0.9	87.6	1.8	24.0	0.2	**72.8**	0.4	36.6	5.7	65.6	10.4	19.4	1.0	37.8	1.8
ConSE [34]	45.6	0.4	88.6	0.8	44.5	0.5	90.6	1.0	34.3	1.6	72.2	3.1	26.9	0.0	**91.2**	0.0	38.8	6.8	39.9	11.6
CMT [46]	39.5	8.4	86.9	15.3	37.9	8.7	89.0	15.9	34.6	4.7	60.1	8.7	28.0	10.9	74.2	19.0	39.9	8.7	28.0	13.3
SSE [54]	60.1	7.0	80.5	12.9	61.0	8.1	82.5	14.8	43.9	8.5	46.9	14.4	34.0	0.2	78.9	0.4	51.5	2.1	36.4	4.0
DeViSE [11]	54.2	13.4	68.7	22.4	59.7	17.1	74.7	27.8	52.0	23.8	53.0	32.8	**39.8**	4.9	76.9	9.2	56.5	16.9	27.4	20.9
SJE [3]	65.6	11.3	74.6	19.6	61.9	8.0	73.9	14.4	53.9	23.5	59.2	33.6	32.9	3.7	55.7	6.9	53.7	14.7	30.5	19.8
LATEM [50]	55.1	7.3	71.7	13.3	55.8	11.5	77.3	20.0	49.3	15.2	57.3	24.0	35.2	0.1	73.0	0.2	55.3	14.7	28.8	19.5
ESZSL [40]	58.2	6.6	75.6	12.1	58.6	5.9	77.8	11.0	53.9	12.6	63.8	21.0	38.3	2.4	70.1	4.6	54.5	11.0	27.9	15.8
ALE [2]	59.9	16.8	76.1	27.5	62.5	14.0	81.8	23.9	54.9	23.7	62.8	34.4	39.7	4.6	73.7	8.7	58.1	**21.8**	33.1	**26.3**
SYNC [5]	54.0	8.9	87.3	16.2	46.6	10.0	90.5	18.0	55.6	11.5	70.9	19.8	23.9	7.4	66.3	13.3	56.3	7.9	**43.3**	13.4
SAE [23]	53.0	1.8	77.1	3.5	54.1	1.1	82.2	2.2	33.3	7.8	54.0	13.6	8.3	0.4	80.9	0.9	40.3	8.8	18.0	11.8
Relation Net [47]	68.2	31.4	**91.3**	46.7	64.2	30.0	**93.4**	**45.3**	**55.6**	**38.1**	61.1	**47.0**	-	-	-	-	-	-	-	-
DEM	**68.4**	**32.8**	84.7	**47.3**	**67.1**	**30.5**	86.4	45.1	51.7	19.6	57.9	29.2	35.0	**11.1**	75.1	**19.4**	**61.9**	20.5	34.3	25.6

Table 2: Comparative results on four datasets. Under that ZSL setting, the performance is evaluated using per-class average Top-1 (**T1**) accuracy (%), and under GZSL, it is measured using **u** = **T1** on unseen classes, **s** = **T1** on seen classes, and **H** = harmonic mean.

Model	hit@5
ConSE [34]	28.5
DeViSE [11]	31.8
Mensink *et al.* [30]	35.7
Rohrbach [39]	34.8
PST [38]	34.0
AMP [15]	41.0
Ours	**46.7**
Gaussian Embedding [33]	45.7
PDDM [20]	48.2
DEM	**60.7**

Table 3: Comparative results (%) on ILSVRC 2010 (hit@1 accuracy over all samples) under the old and conventional setting.

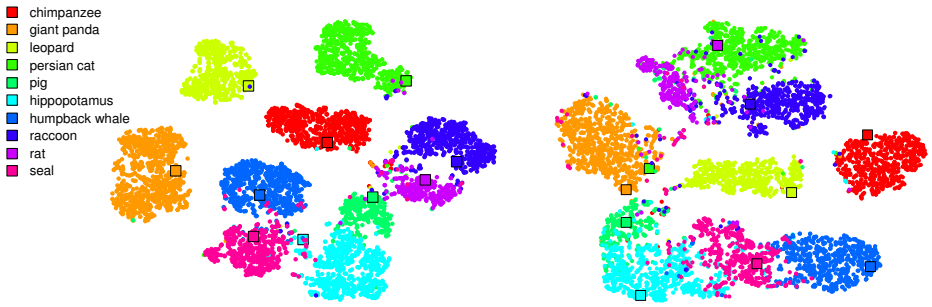
alternatives by a large margin.

Model	hit@1	hit@5
ConSE [34]	7.8	15.5
DeViSE [11]	5.2	12.8
AMP [15]	6.1	13.1
SS-Voc [14]	9.5	16.8
DEM	**11.0**	**25.7**

Table 4: Comparative results (%) on ILSVRC 2012/2010 (hit@1 accuracy over all samples) under the old and conventional setting.

4.4. Further analysis

Importance of embedding space selection We argued that the key for an effective deep embedding model is the use of the CNN output visual feature space rather than the semantic space as the embedding space. In this experiment, we modify our model in Fig. 1 by moving the two FC layers from the semantic embedding branch to the CNN feature extraction branch so that the embedding space now becomes the semantic space (attributes are used). Table 5 shows that by mapping the visual features to the semantic embedding space, the performance on AwA drops by 26.1% on AwA, highlighting the importance of selecting the right embedding space. We also hypothesised that using the CNN visual feature space as the embedding layer would lead to less hubness problem. To verify that we measure the hubness using the skewness score (see Sec. 3.5). Table 6 shows clearly that the hubness problem is much more severe when the wrong embedding space is selected. We also plot the data distribution of the 10 unseen classes of AwA together with the prototypes. Figure 4 suggests that with the visual feature space as the embedding space, the 10 classes form com-



(a) S → V (b) V → S

Figure 4: Visualisation of the distribution of the 10 unseen class images in the two embedding spaces on AwA using t-SNE [28]. Different classes as well as their corresponding class prototypes (in squares) are shown in different colours. Better viewed in colour.

pact clusters and are near to their corresponding prototypes, whilst in the semantic space, the data distributions of different classes are much less separated and a few prototypes are clearly hubs causing miss-classification.

Loss	**Visual → Semantic**	**Semantic → Visual**
Least square loss	60.6	**86.7**
Hinge loss	57.7	72.8

Table 5: Effects of selecting different embedding space and different loss functions on zero-shot classification accuracy (%) on AwA.

N_1 **skewness**	**AwA**	**CUB**
Visual → Semantic	0.4162	8.2697
Semantic → Visual	**-0.4834**	**2.2594**

Table 6: N_1 skewness score on AwA and CUB with different embedding space.

Neural network formulation Can we apply the idea of using visual feature space as embedding space to other models? To answer this, we consider a very simple model based on linear ridge regression which maps from the CNN feature space to the attribute semantic space or vice versa. In Table 7, we can see that even for such a simple model, very impressive results are obtained with the right choice of embedding space. The results also show that with our neural network based model, much better performance can be obtained due to the introduced nonlinearity and its ability to learn end-to-end.

Choices of the loss function As reviewed in Sec. 2, most existing ZSL models use either margin based losses or binary cross entropy loss to learn the embedding model. In this work, least square loss is used. Table 5 shows that when the semantic space is used as the embedding space, a slightly inferior result is obtained using a hinge ranking loss in place of least square loss in our model. However, least square loss is clearly better when the visual feature space is the embedding space.

Model	**AwA**	**CUB**
Linear regression (V → S)	54.0	40.7
Linear regression (S → V)	74.8	45.7
DEM	**86.7**	**58.3**

Table 7: Zero-shot classification accuracy (%) comparison with linear regression on AwA and CUB.

5. Conclusion

We have proposed a novel deep embedding model for zero-shot learning. The model differs from existing ZSL model in that it uses the CNN output feature space as the embedding space. We hypothesise that this embedding space would lead to less hubness problem compared to the alternative selections of embedding space. Further more, the proposed model offers the flexible of utilising multiple semantic spaces and is capable of end-to-end learning when the semantic space itself is computed using a neural network. Extensive experiments show that our model achieves state-of-the-art performance on a number of benchmark datasets and validate the hypothesis that selecting the correct embedding space is the key for achieving the excellent performance.

Acknowledgement

This work was funded in part by the European FP7 Project SUNNY (grant agreement no. 313243).

References

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for attribute-based classification. In *CVPR*, 2013. 1

[2] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for image classification. *TPAMI*, 2016. 8

[3] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele. Evaluation of output embeddings for fine-grained image classification. In *CVPR*, 2015. 1, 2, 6, 7, 8

[4] M. Bucher, S. Herbin, and F. Jurie. Improving semantic embedding consistency by metric learning for zero-shot classiffication. In *ECCV*, 2016. 1, 7

[5] S. Changpinyo, W.-L. Chao, B. Gong, and F. Sha. Synthesized classifiers for zero-shot learning. In *CVPR*, 2016. 1, 7, 8

[6] 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 *ECCV*, 2016. 1

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 2

[8] G. Dinu, A. Lazaridou, and M. Baroni. Improving zero-shot learning by mitigating the hubness problem. In *ICLR workshop*, 2014. 2, 3

[9] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *CVPR*, 2009. 1, 2

[10] V. Ferrari and A. Zisserman. Learning visual attributes. In *NIPS*, 2007. 1, 2

[11] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov, et al. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013. 1, 2, 3, 5, 6, 7, 8

[12] Y. Fu, T. M. Hospedales, T. Xiang, Z. Fu, and S. Gong. Transductive multi-view embedding for zero-shot recognition and annotation. In *ECCV*, 2014. 1, 2, 6, 7

[13] Y. Fu, T. M. Hospedales, T. Xiang, and S. Gong. Transductive multi-view zero-shot learning. *PAMI*, 2015. 6, 7

[14] Y. Fu and L. Sigal. Semi-supervised vocabulary-informed learning. In *CVPR*, 2016. 1, 2, 6, 7, 8

[15] Z. Fu, T. Xiang, E. Kodirov, and S. Gong. Zero-shot object recognition by semantic manifold distance. In *CVPR*, 2015. 1, 2, 6, 7, 8

[16] A. Graves, N. Jaitly, and A.-r. Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *ASRU*, 2013. 4

[17] A. Graves, A. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*, 2013. 4

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 6

[19] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 1997. 4

[20] C. Huang, C. C. Loy, and X. Tang. Local similarity-aware deep feature embedding. In *NIPS*, 2016. 7, 8

[21] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 6, 7

[22] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 6

[23] E. Kodirov, T. Xiang, and S. Gong. Semantic autoencoder for zero-shot learning. In *CVPR*, 2017. 8

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 3, 6, 7

[25] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *PAMI*, 2014. 1, 2, 6, 8

[26] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, 2012. 4

[27] J. Lei Ba, K. Swersky, S. Fidler, and R. Salakhutdinov. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *ICCV*, 2015. 1, 2, 3, 5, 6, 7

[28] L. v. d. Maaten and G. Hinton. Visualizing data using t-SNE. *JMLR*, 2008. 9

[29] B. Marco, L. Angeliki, and D. Georgiana. Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In *ACL*, 2015. 3

[30] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Metric learning for large scale image classification: Generalizing to new classes at near-zero cost. In *ECCV*, 2012. 6, 8

[31] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. In *arXiv preprint arXiv:1301.3781*, 2013. 6

[32] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013. 6

[33] T. Mukherjee and T. Hospedales. Gaussian visual-linguistic embedding for zero-shot recognition. In *EMNLP*, 2016. 7, 8

[34] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean. Zero-shot learning by convex combination of semantic embeddings. In *ICLR*, 2014. 1, 6, 8

[35] D. Parikh and K. Grauman. Relative attributes. In *ICCV*, 2011. 1, 2

[36] M. Radovanović, A. Nanopoulos, and M. Ivanović. Hubs in space: Popular nearest neighbors in high-dimensional data. *JMLR*, 2010. 2, 5

[37] S. Reed, Z. Akata, B. Schiele, and H. Lee. Learning deep representations of fine-grained visual descriptions. In *CVPR*, 2016. 1, 2, 3, 4, 6, 7

[38] M. Rohrbach, S. Ebert, and B. Schiele. Transfer learning in a transductive setting. In *NIPS*, 2013. 8

[39] M. Rohrbach, M. Stark, and B. Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *CVPR*, 2011. 8

[40] B. Romera-Paredes and P. Torr. An embarrassingly simple approach to zero-shot learning. In *ICML*, 2015. 1, 2, 7, 8

[41] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015. 6

[42] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 1997. 4

[43] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization

and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 6, 7

[44] Y. Shigeto, I. Suzuki, K. Hara, M. Shimbo, and Y. Matsumoto. Ridge regression, hubness, and zero-shot learning. In *ECML/PKDD*, 2015. 3, 5

[45] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6, 7

[46] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng. Zero-shot learning through cross-modal transfer. In *NIPS*, 2013. 1, 2, 3, 5, 6, 7, 8

[47] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *CVPR*, 2018. 2, 8

[48] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 6, 7

[49] C. Wah, S. Branson, P. Perona, and S. Belongie. Multiclass recognition and part localization with humans in the loop. In *ICCV*, 2011. 2, 6

[50] Y. Xian, Z. Akata, G. Sharma, Q. Nguyen, M. Hein, and B. Schiele. Latent embeddings for zero-shot classification. In *CVPR*, 2016. 8

[51] Y. Xian, C. H. Lampert, B. Schiele, and Z. Akata. Zero-shot learning-a comprehensive evaluation of the good, the bad and the ugly. *arXiv preprint arXiv:1707.00600*, 2017. 5, 6, 7

[52] Y. Yang and T. M. Hospedales. A unified perspective on multi-domain and multi-task learning. In *ICLR*, 2015. 1, 2, 3, 5, 6, 7

[53] L. Zhang, F. Sung, F. Liu, T. Xiang, S. Gong, Y. Yang, and T. M. Hospedales. Actor-critic sequence training for image captioning. In *NeurIPS Workshop on Visually-Grounded Interaction and Language*, 2017. 2

[54] Z. Zhang and V. Saligrama. Zero-shot learning via semantic similarity embedding. In *ICCV*, 2015. 1, 2, 7, 8

[55] Z. Zhang and V. Saligrama. Zero-shot learning via joint latent similarity embedding. In *CVPR*, 2016. 1, 7

[56] Z. Zhang and V. Saligrama. Zero-shot recognition via structured prediction. In *ECCV*, 2016. 1, 7