Zero-Shot Object Detection via Learning an Embedding from Semantic Space to Visual Space

Licheng Zhang[1,2], **Xianzhi Wang**[3], **Lina Yao**[4], **Lin Wu**[5] and **Feng Zheng**[1,2*]
[1]Depatment of Computer Science and Technology, Southern University of Science and Technology
[2]Research Institute of Trustworthy Autonomous Systems, Shenzhen, 518055, China
[3]University of Technology Sydney
[4]University of New South Wales
[5]Hefei University of Technology
lichengzhangcg@gmail.com, xianzhi.wang@uts.edu.au, zhengf@sustech.edu.cn

Abstract

Zero-shot object detection (ZSD) has received considerable attention from the community of computer vision in recent years. It aims to simultaneously locate and categorize previously unseen objects during inference. One crucial problem of ZSD is how to accurately predict the label of each object proposal, i.e. categorizing object proposals, when conducting ZSD for unseen categories. Previous ZSD models generally relied on learning an embedding from visual space to semantic space or learning a joint embedding between semantic description and visual representation. As the features in the learned semantic space or the joint projected space tend to suffer from the hubness problem, namely the feature vectors are likely embedded to an area of incorrect labels, and thus it will lead to lower detection precision. In this paper, instead, we propose to learn a deep embedding from the semantic space to the visual space, which enables to well alleviate the hubness problem, because, compared with semantic space or joint embedding space, the distribution in visual space has smaller variance. After learning a deep embedding model, we perform k nearest neighbor search in the visual space of unseen categories to determine the category of each semantic description. Extensive experiments on two public datasets show that our approach significantly outperforms the existing methods.

1 Introduction

Zero-shot object detection (ZSD) has gained popularity in recent years by researchers. The goal of ZSD is to detect and recognize objects unobserved in training examples [Rahman *et al.*, 2018; Bansal *et al.*, 2018].

Traditional object detection research benefits from a large amount of annotated training samples for each object class [Ren *et al.*, 2015; Liu *et al.*, 2015]. The objects to be detected during testing have appeared in training examples. However, sometimes, it is impossible to collect training data for rare concepts, for example, Okapia, and none of the existing object detection models can work with no training samples for a given class. This poses the challenge of zero-shot object detection.

Zero-shot learning has gained significant progress in recent years. A ZSL method depends on the existence of labelled training data of seen classes and the relationship between unseen classes and seen classes. Seen and unseen classes are usually semantically related in a high dimensional space, called semantic space. There are three types of embedding space for ZSL models: learning a joint embedding space between visual space and semantic space [Ba *et al.*, 2015], learning an embedding from visual space to semantic space [Frome *et al.*, 2013], and learning an embedding from semantic space to visual space [Zhang *et al.*, 2016]. Taking semantic space or a joint embedding as the embedding space means that visual feature will be projected into the semantic space or the joint space, which will shrink the variance of the projected data points and thus aggravate the hubness problem [Zhang *et al.*, 2016], which means that there is some vector that tends to be near a high proportion of items, which have incorrect labels.

Compared to the conventional ZSL task, ZSD is a relatively unexplored research problem and far more challenging. ZSL models usually recognize only one dominant object in each image, while ZSD models detect multiple objects in a single image, which commonly exists in real-world applications. Previous studies on ZSD usually learned a joint embedding space between visual space and semantic space [Li *et al.*, 2019] or learned an embedding from visual space to semantic space [Rahman *et al.*, 2018; Bansal *et al.*, 2018; Demirel *et al.*, 2018]. They determined the label of each object proposal by choosing the most similar unseen category based on the learned embeddings. Compared to learning a joint embedding space or learning an embedding from visual space to semantic space, embedding the semantic space to the visual space suffers less from the hubness problem [Zhang *et al.*, 2016], as the distribution in the visual space has a smaller variance than that in the semantic space or the joint embedding space [Shigeto *et al.*, 2015].

In this paper, we propose to use the visual feature space as

*Corresponding Author

the embedding space for ZSD. For each textual description of the image, we extract visual feature from the image and embed the textual description into the visual space. Then we use a least square embedding loss to minimize the discrepancy between the visual feature and the semantic embedding vector in visual space. After model learning, we perform k nearest neighbor search in visual space of unseen categories and choose the category with most nearest neighbors as the corresponding label of this textual description. One image often corresponds to several textual descriptions because there are usually several objects in the image to be detected, thus we can obtain several candidate labels for each image. During testing, we choose the most similar category from candidate labels as the label of each object proposal, by computing cosine similarity between word vector representations of two label names.

To summarize, we make the following contributions: (1) We propose to use the visual space as the embedding space for ZSD and perform k nearest neighbor search in the visual space to determine the unseen category for each textual description. (2) We propose to integrate textual descriptions into Faster R-CNN when training the ZSD model and utilize online hard example mining (OHEM) [Shrivastava *et al.*, 2016] method to train the network. (3) We have conducted extensive experiments on two public datasets and demonstrate the effectiveness of our proposed approach.

The rest of this paper is organized as follows. We first review the related works on object detection, zero-shot learning and zero-shot object detection. Then we introduce our ZSD and ZSL method in detail. Lastly, we present the experimental results, followed by the conclusion.

2 Related Work

2.1 Object Detection

Object detection has experienced great development in the past decade. Researchers have demonstrated the superiority of object proposal based methods for detecting objects in the image [Girshick *et al.*, 2013; Girshick, 2015; Ren *et al.*, 2015].

R-CNN [Girshick *et al.*, 2013] first generates a large number of region proposals and utilizes convolutional neural network (CNN) to extract fixed-dimension features from each region proposal. Then support vector machine (SVM) is used to classify region proposals into different categories. Fast R-CNN [Girshick, 2015] exploits CNN rather than SVM to perform multi-class classification and predict refined bounding boxes. Faster R-CNN [Ren *et al.*, 2015] utilizes region proposal network (RPN) to generate region proposals. With RPN, Faster R-CNN can be trained in an end-to-end manner. Mask R-CNN [He *et al.*, 2017] adds an image segmentation branch and replaces RoI pooling with RoI Align to improve object detection accuracy.

Although these object detection methods work well on predefined concepts, they cannot be directly exploited to detect novel concepts.

2.2 Zero-shot Learning

ZSL models can be divided into three types. The first type is learning an embedding from the visual space to the semantic space [Socher *et al.*, 2013; Frome *et al.*, 2013]. The second type is learning a joint embedding space between semantic description and visual representation [Ba *et al.*, 2015; Reed *et al.*, 2016]. The third type is learning an embedding from the semantic space to the visual space [Zhang *et al.*, 2016].

Socher et al. [Socher *et al.*, 2013] built a ZSL model, which projected the image feature vectors into the semantic word space. They first detected images of unseen classes and then classified them to the zero-shot word vectors. They used a Gaussian discriminator as the unseen classifier. Frome et al. [Frome *et al.*, 2013] constructed a deep visual-semantic model by extracting features from images using visual object recognition network and re-training them to predict the vector representation of label text. They chose the semantic space as the embedding space and used a hinge rank loss to train the network. Ba et al. [Ba *et al.*, 2015] used deep neural networks to embed image and text features into a joint embedding space. They performed dot product between the visual feature and the semantic vector and considered three training losses: binary cross entropy loss, hinge loss and Euclidean distance loss. Reed et al. [Reed *et al.*, 2016] learned a deep structured joint embedding, which jointly embedded images and fine-grained visual descriptions. They used the inner product of image and text features as the compatibility function and maximize the compatibility between a description and its matching image, and minimize the compatibility with images from other classes. Zhang et al. [Zhang *et al.*, 2016] learned a deep embedding model from semantic space to visual space. They performed nearest neighbor search in the visual space of test prototypes, which were formed by averaging text features of test descriptions per-class. They demonstrated that using the visual space as the embedding space suffers less from the hubness problem and performed better than using semantic space as the embedding space or learning a joint embedding space between visual representation and semantic description.

2.3 Zero-shot Object Detection

ZSD is a recently introduced problem that aims to simultaneously locate and recognize objects unobserved in training samples [Rahman *et al.*, 2018; Bansal *et al.*, 2018; Demirel *et al.*, 2018; Li *et al.*, 2019].

Rahman et al. [Rahman *et al.*, 2018] utilized Faster R-CNN and a zero-shot recognition framework named ConSE to detect unseen classes. They designed a semantic alignment network to project visual space to semantic space. They approximated the bounding box for an unseen object through the box proposal of a closely related seen object and predict each object as a seen or unseen class. They also reported a simplified variant of their approach when there were no pre-defined unseen classes, in which they used semantic embeddings only and computed cosine similarities with all unseen word vectors. Bansal et al. [Bansal *et al.*, 2018] first learned a zero-shot object detection model on seen classes, which is a background-aware model. Then they used an iterative Expectation Maximization like algorithm to spread back-

ground boxes over a wider range of visual concepts. They also chose to project visual space to semantic space. They predicted the label of each unseen object by choosing the category with highest similarity. Demirel et al. [Demirel *et al.*, 2018] proposed a hybrid model, which consists of two components. The first component embedded image regions into a class embedding space. The second component learns a direct mapping from region pixels to the space of class embeddings. Both region embeddings are then compared with true class embeddings to get region detection results according to similarities. Li et al. [Li *et al.*, 2019] utilized Faster R-CNN and natural language descriptions for zero-shot object detection. They made use of LSTM to model textual descriptions and performed element-wise multiply between visual space and semantic space to predict whether textual descriptions fitted the object proposal. These works chose to embed the visual space into the semantic space or learned a joint embedding space between the visual representation and the semantic description. However, these embedding spaces suffer much from the hubness problem, which is an inherent property of data distributions in a high-dimensional vector space. Due to this shortcoming of previous works, we choose the visual space as the embedding space, which has been demonstrated effective by previous ZSL works [Reed *et al.*, 2016; Zhang *et al.*, 2016].

3 Methodology

Our proposed ZSD approach contains two modules. One is the ZSD module, which is utilized to perform object detection. The other is the ZSL module, which is exploited to conduct zero-shot learning and predict possible unseen objects in the image.

3.1 Zero-shot Object Detection with Textual Description

The architecture of our ZSD model is shown in Figure 1. It consists of two branches. The first branch is Faster R-CNN, which is utilized to extract features from images and perform classification and regression for object proposals. The backbone of Faster R-CNN is the Inception-ResNet v2 model [Szegedy *et al.*, 2016]. The network first extracts a global feature map from the image. Then it utilizes RPN to generate lots of region proposals and distinguish them into foreground and background. Actually, RPN is trained on seen classes. As mentioned in [Li *et al.*, 2019], a pre-trained RPN on seen classes can be directly applied to generate region proposals for unseen classes. After RPN, the network randomly chooses region proposals and makes use of RoI pooling to extract fixed-size features for each region proposal. Then the network utilizes region CNN, which is part of Inception-ResNet v2 model, to further extract features from RoI feature map.

The second branch is deep CNN, which is utilized to extract features from textual descriptions. Different from [Li *et al.*, 2019], which utilized the textual description to determine which object proposal fits the textual description, in our work, we exploit textual descriptions to help training the ZSD network. Because textual descriptions contain valuable information related to categories of objects, adding them into the ZSD network can improve object detection accuracy. Specifically, we concatenate text features with image features of each object proposal after region CNN in Faster R-CNN, followed by two fully connected layers. Then the network performs classification and box regression.

We use the deep CNN proposed in [Conneau *et al.*, 2016]. It consists of 4 convolutional blocks for filter number 64, 128, 256, 512 respectively, on top of the first convolutional layer, whose filter number is 64. We add a new convolutional layer after the last convolutional block to adjust the output dimension, in order to match the dimension of image features. There is a max-pooling layer between two convolutional blocks, whose filter numbers are different. Each convolutional block consists of two convolutional layers, each one followed by a batch normalization layer and a Rectified Linear Unit (ReLU). The kernel size of each convolutional layer is 3. We use shortcut connections between neighboring convolutional blocks to reduce degradation, which was recommended in [He *et al.*, 2015]. The input of deep CNN are word vector representations whose height is 1, width is N, and dimension of the input channel is 300. N represents the number of words in the textual description. We concatenate several textual descriptions into one description and feed it into deep CNN.

Before fed into deep CNN, textual descriptions are firstly transformed into word vector representations word by word. We use GloVe [Pennington *et al.*, 2014] to extract vector representations for each word. The dimension of the vector representation in GloVe is 300.

We optimize the ZSD network based on the RPN loss function and the detection loss function with equal weights. RPN loss function consists of RPN two-category cross-entropy loss and RPN bounding box regression loss. The detection loss function consists of multi-class classification cross-entropy loss and bounding box regression loss. The loss functions are the same as those used in [Ren *et al.*, 2015].

3.2 Learning an Embedding from Semantic Space to Visual Space

The architecture of our ZSL model is shown in Figure 2. It consists of two branches. The first branch is the visual encoding branch, which utilizes CNN to extract visual features from images. This branch utilizes pretrained CNN model and the parameters are not updated during network training. In our work, we use the Inception-ResNet v2 model on ImageNet [Szegedy *et al.*, 2016] to extract features from images. The second branch is the semantic embedding branch. This branch uses a deep CNN, which also comes from [Conneau *et al.*, 2016] but have different number of layers, to extract features from textual descriptions and embeds semantic features to visual features. The deep CNN consists of 2 convolutional blocks for filter number 64, 128, 256, 512 respectively, on top of the first convolutional layer, which has 64 filters. We add a fully connected linear layer and a ReLU activation on top of deep CNN, in order to embed the semantic features into the visual space.

The two branches are linked by a least square loss, which aims to minimize the discrepancy between the visual feature and semantic feature. It is demonstrated by previous work

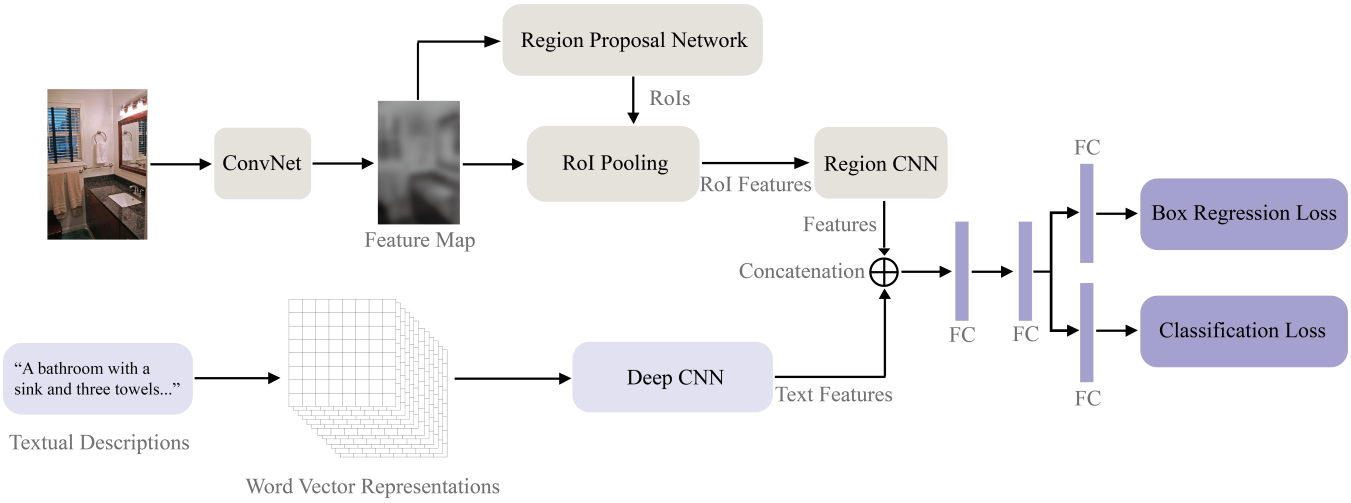

Figure 1: The architecture of our ZSD model. The upper half in the figure is Faster R-CNN. The lower half in the figure is deep CNN, which is utilized to extract features from textual descriptions. Region CNN is used to further extract features from RoI feature map. Text features are concatenated with image features as the input of the following part. "FC" represents a fully connected layer.

[Zhang *et al.*, 2016] that the least square loss copes with the hubness problem better. We use $V_i \in \mathbb{R}^{D\times 1}$ to represent the visual feature and use $S_i \in \mathbb{R}^{D\times 1}$ to represent the semantic feature. The least square loss is defined as follows:

$$L(\{V_i\},\{S_i\}) = \frac{1}{N}\sum_{i=1}^{N}||V_i - S_i||^2, \qquad (1)$$

in which, N is the number of samples in a minibatch and D is the dimension of visual feature vector and semantic feature vector.

ZSD model is trained on seen classes. During testing, we perform inference on unseen classes. Each unseen object is classified as a seen class, which are visually similar. ZSL model is trained on both seen and unseen classes. During inference, we first randomly choose m textual descriptions for each unseen category and embed them into the visual space, which is named as M. Then for each description, we embed it into the visual space and perform k nearest neighbor search, which is based on the distance between two feature vectors, in the visual space M. We choose the category with most nearest neighbors as the corresponding label of this textual description. One image often corresponds to several textual descriptions, thus we can obtain several candidate labels for each image. In order to determine the category of each object. We compute cosine similarity between word vector representations of the predicted seen label by our ZSD model and each candidate unseen category predicted by our ZSL model, and choose the most similar category as the label of each object proposal.

4 Experiments

4.1 Datasets

MS COCO. This dataset was designed for detecting and segmenting objects [Lin *et al.*, 2014]. It contains caption descriptions for every image, indicating the objects in the images. The ground truth includes the category, bounding box positions, and boundary points' positions for each object. We utilize bounding box positions of objects and captions of each image for experiments.

Visual Genome. This dataset was designed for visual relationship understanding [Krishna *et al.*, 2016]. It contains rich information about regions and objects in images. We focus on categories and bounding box positions of objects, as well as region descriptions that describe the objects in the image for experiments.

4.2 Data Split

For the zero-shot learning setting, unseen objects are not allowed to exist in training samples. In terms of MS COCO dataset, following [Bansal *et al.*, 2018] and [Li *et al.*, 2019], we choose the same 48 categories for training and the same 17 categories for testing. We remove the images in training samples that contain objects from unseen classes, so as not to take unseen objects as background. We use the data listed by [Bansal *et al.*, 2018] for testing. For Visual Genome dataset, following [Bansal *et al.*, 2018] and [Li *et al.*, 2019], we choose the same 478 classes for training and the same 130 classes for testing and use the data listed by [Bansal *et al.*, 2018] for testing. Because most of training images contain unseen objects, we don't remove images in training samples.

4.3 Training and Testing Settings

Detection datasets usually contain a large amount of easy examples and a small number of hard examples. Selecting hard examples to optimize the network can make training more effective and efficient. In our work, we utilize OHEM method, which was proposed in [Shrivastava *et al.*, 2016], to train our ZSD model. Specifically, we first select some examples from all region proposals generated by RPN, and perform network inference, and compute the loss for each region proposal. Then we rank the losses of these region proposals and choose the samples whose losses are among top l, to optimize the

IoU	MS COCO			Visual Genome		
	0.4	0.5	0.6	0.4	0.5	0.6
SAN [Rahman *et al.*, 2018]	35.70	26.30	14.50	6.80	5.90	3.10
SB [Bansal *et al.*, 2018]	34.46	24.39	12.55	6.06	4.09	2.43
DSES [Bansal *et al.*, 2018]	40.23	27.19	13.63	7.78	4.75	2.34
LAB [Bansal *et al.*, 2018]	31.86	20.52	9.98	8.43	5.40	2.74
ZSD-LSTM [Li *et al.*, 2019]	45.50	34.30	18.10	9.70	7.20	4.20
ZSD-CNN-ohem	**47.76**	**41.15**	**34.44**	**13.73**	**11.03**	**8.30**

Table 1: Experimental comparison of different methods on MS COCO and Visual Genome datasets. Recall@100 is used as the evaluation metric. Larger recall is better.

IoU	MS COCO			Visual Genome		
	0.4	0.5	0.6	0.4	0.5	0.6
ZSD-CNN-normal-w/o	43.47	37.02	29.22	7.99	5.47	3.42
ZSD-CNN-normal	**45.65**	**38.54**	**30.56**	**11.42**	**8.33**	**5.66**

Table 2: Experimental comparison of whether adding textual descriptions into Faster R-CNN.

network. Compared with traditional means, which randomly selects examples from all region proposals to optimize the network and update the parameters of Faster R-CNN, OHEM method reduces the losses of hard examples, which in theory is better than traditional means. For normal training, we train the network in an end-to-end manner. Firstly, we initialize the parameters of the backbone of Faster R-CNN using the pre-trained Inception-ResNet v2 model on ImageNet [Szegedy *et al.*, 2016]. The parameters of the rest part are randomly initialized, including deep CNN. Then the ZSD network is trained with a learning rate 10^{-4}, which is decayed by 0.5 after every epoch. We use a stochastic gradient descent optimizer. For OHEM training, the parameters are initialized using the parameters learned by normal training. It should be noted that, for MS COCO dataset, the number of filters of the added convolutional layer in deep CNN is 128. And textual descriptions are truncated or padded to a fixed length, 128 words. For Visual Genome dataset, the added convolutional layer of deep CNN has 32 filters. The textual descriptions for Visual Genome are truncated or padded to 640 words.

For our ZSL model, deep CNN is initialized with random weights. Adam [Kingma and Ba, 2014] is used to optimize our model with a learning rate of 0.0001. Both training and testing data are used during network training for our ZSL model. For MS COCO dataset, each textual description is truncated or padded to 16 words and we randomly choose 3000 textual descriptions for each unseen class and perform 10 nearest neighbor search in visual space for each description during testing. For Visual Genome dataset, each textual description is truncated or padded to 10 words and we randomly choose 1000 textual descriptions for each unseen class and perform 10 nearest neighbor search in the visual space.

4.4 Evaluation Metric

Following [Bansal *et al.*, 2018] and [Li *et al.*, 2019], we use recall as the evaluation metric for fair comparison. We compare our proposed method with three previous works. These methods are from [Rahman *et al.*, 2018; Bansal *et al.*, 2018; Li *et al.*, 2019]. We use the same names as the original works to present the experimental results. We utilize "ZSD-LSTM" to represent the method in [Li *et al.*, 2019]. Following [Bansal *et al.*, 2018], we keep 100 detection bounding boxes whose classification scores are among top 100 in all bounding boxes, to compute the recall.

4.5 Results

Table 1 shows the experimental comparison of different methods on MS COCO and Visual Genome datasets using Recall@100 evaluation metric. We use three IoU thresholds for experiments: 0.4, 0.5, 0.6. For previous methods, we use the results in [Bansal *et al.*, 2018] and [Li *et al.*, 2019]. Because [Rahman *et al.*, 2018] doesn't have results on MS COCO and Visual Genome, we use the results from [Li *et al.*, 2019] for their method. For our method, the name containing "ohem" means using OHEM method to train the network.

From Table 1, we can make the observation that our proposed ZSD approach (ZSD-CNN-ohem) performs better than previous methods on both two datasets. We use the threshold of 0.4 as an example. Compared with the second best baseline, our ZSD model improves the recall from 45.50% to 47.76% on the MS COCO dataset and improves the recall from 9.70% to 13.73% on the Visual Genome dataset. The two improvements are very significant in the challenging ZSD setting and demonstrate that our ZSD approach is an effective method.

4.6 Ablation Studies

We continue to present the extensive experimental results to analyse the effect of different components in our model. We use Recall@100 as the evaluation metric and use three thresholds for experiments: 0.4, 0.5, 0.6.

Textual Description. Table 2 presents the experimental results of whether adding textual descriptions into Faster R-CNN. The name containing "w/o" means not adding textual descriptions into Faster R-CNN. The name containing "normal" means using normal training method to optimize the net-

IoU	MS COCO			Visual Genome		
	0.4	0.5	0.6	0.4	0.5	0.6
ZSD-CNN-normal-all	32.56	27.23	22.01	10.20	7.58	5.13
ZSD-CNN-normal	**45.65**	**38.54**	**30.56**	**11.42**	**8.33**	**5.66**

Table 3: Comparative results of using our ZSL model to generate candidate labels or using all unseen categories as candidate labels. Recall@100 is the evaluation metric. Larger recall is better.

IoU	MS COCO			Visual Genome		
	0.4	0.5	0.6	0.4	0.5	0.6
ZSD-CNN-normal	45.65	38.54	30.56	11.42	8.33	5.66
ZSD-CNN-ohem	**47.76**	**41.15**	**34.44**	**13.73**	**11.03**	**8.30**

Table 4: Experimental comparison of whether using OHEM method to train our ZSD model.

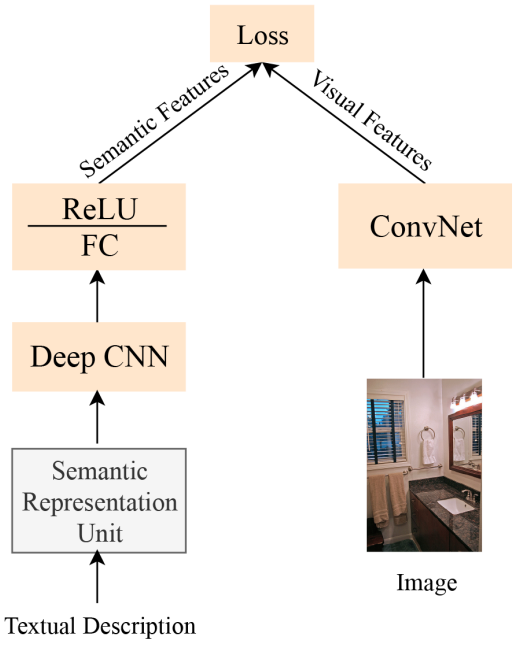

Figure 2: Illustration of the network architecture of our ZSL model. "FC" represents a fully connected layer. "ReLU" is a Rectified Linear Unit. "ConvNet" is Inception-ResNet v2 model. We use deep CNN proposed in [Conneau *et al.*, 2016]. Semantic representation unit is used to transfer textual description into word vector representations using GloVe. Least square loss is used as the loss function.

work. From the results, we can find that, textual description is a crucial component in our model because it can improve the recall by a large margin. For example, on MS COCO dataset and threshold 0.4, without textual descriptions, the recall of the ZSD model is 43.47%, and with textual descriptions, the model improves the recall to 45.65%, achieving 2.18% improvement, and on Visual Genome dataset and threshold 0.4, the recall is improved from 7.99% to 11.42%, achieving 3.43% improvement, compared with not adding textual descriptions. It demonstrates that textual descriptions can help training the network, and improve the ZSD performance, because they contain much valuable information related to categories.

ZSL Method. Table 3 gives the comparative results of using all unseen categories as candidate labels or using our ZSL model to generate unseen candidate labels. The name containing "all" means using all unseen categories as candidate labels rather than using our ZSL model to generate candidate unseen labels. From the results, we observe that our ZSL model is also a crucial component for our ZSD approach. For example, on MS COCO dataset and threshold 0.4, our model improves the recall from 32.56% to 45.65%, achieving 13.09% improvement, which is a very significant improvement. It demonstrates that using the visual space as the embedding space and performing k nearest neighbor search in the visual space is an effective approach.

OHEM Training. Table 4 presents the experimental results of whether using OHEM method to train the network. According to the results, we can make the observation that OHEM training can slightly improve the ZSD performance, averaging 2 percent on the two datasets at threshold 0.4.

5 Conclusion

Previous ZSD works suffer much more from the hubness problem. To alleviate the hubness problem, in this paper, we propose to use the visual space as the embedding space, i.e., embedding the semantic space into the visual space. We perform k nearest neighbor search in the visual space of test categories and determine the unseen category for each description. We also add textual descriptions into Faster R-CNN to help training the network and employ OHEM method to optimize the network. Extensive experiments on two public datasets show that our model achieves state-of-the-art performance compared with previous baselines, and demonstrate that using the visual space as the embedding space achieves more excellent performance.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61972188), Guangdong Provincial Key Laboratory (Grant No. 2020B121201001), the Program for University Key Laboratory of Guangdong Province (Grant No. 2017KSYS008).

References

[Ba *et al.*, 2015] Jimmy Lei Ba, Kevin Swersky, Sanja Fidler, and Ruslan Salakhutdinov. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pages 4247–4255. IEEE Computer Society, 2015.

[Bansal *et al.*, 2018] Ankan Bansal, Karan Sikka, Gaurav Sharma, Rama Chellappa, and Ajay Divakaran. Zero-shot object detection. *CoRR*, abs/1804.04340, 2018.

[Conneau *et al.*, 2016] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann LeCun. Very deep convolutional networks for natural language processing. *CoRR*, abs/1606.01781, 2016.

[Demirel *et al.*, 2018] Berkan Demirel, Ramazan Gokberk Cinbis, and Nazli Ikizler-Cinbis. Zero-shot object detection by hybrid region embedding. *CoRR*, abs/1805.06157, 2018.

[Frome *et al.*, 2013] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc' Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *Advances in Neural Information Processing Systems 26*, pages 2121–2129. Curran Associates, Inc., 2013.

[Girshick *et al.*, 2013] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013.

[Girshick, 2015] Ross B. Girshick. Fast R-CNN. *CoRR*, abs/1504.08083, 2015.

[He *et al.*, 2015] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[He *et al.*, 2017] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[Kingma and Ba, 2014] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[Krishna *et al.*, 2016] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Fei-Fei Li. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *CoRR*, abs/1602.07332, 2016.

[Li *et al.*, 2019] Zhihui Li, Lina Yao, Xiaoqin Zhang, Xianzhi Wang, Salil Kanhere, and Huaxiang Zhang. Zero-shot object detection with textual descriptions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):8690–8697, Jul. 2019.

[Lin *et al.*, 2014] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[Liu *et al.*, 2015] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. *CoRR*, abs/1512.02325, 2015.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[Rahman *et al.*, 2018] Shafin Rahman, Salman Hameed Khan, and Fatih Porikli. Zero-shot object detection: Learning to simultaneously recognize and localize novel concepts. *CoRR*, abs/1803.06049, 2018.

[Reed *et al.*, 2016] Scott E. Reed, Zeynep Akata, Bernt Schiele, and Honglak Lee. Learning deep representations of fine-grained visual descriptions. *CoRR*, abs/1605.05395, 2016.

[Ren *et al.*, 2015] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[Shigeto *et al.*, 2015] Yutaro Shigeto, Ikumi Suzuki, Kazuo Hara, Masashi Shimbo, and Yuji Matsumoto. Ridge regression, hubness, and zero-shot learning. In *ECML/PKDD*, 2015.

[Shrivastava *et al.*, 2016] Abhinav Shrivastava, Abhinav Gupta, and Ross B. Girshick. Training region-based object detectors with online hard example mining. *CoRR*, abs/1604.03540, 2016.

[Socher *et al.*, 2013] Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. Zero-shot learning through cross-modal transfer. In *Advances in Neural Information Processing Systems 26*, pages 935–943. Curran Associates, Inc., 2013.

[Szegedy *et al.*, 2016] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261, 2016.

[Zhang *et al.*, 2016] Li Zhang, Tao Xiang, and Shaogang Gong. Learning a deep embedding model for zero-shot learning. *CoRR*, abs/1611.05088, 2016.