Zero-Shot Object Detection with Textual Descriptions Using Convolutional Neural Networks

Licheng Zhang[1, 2], Xianzhi Wang[2], Lina Yao[3], Feng Zheng[1†]
[1]*Department of Computer Science and Engineering, Southern University of Science and Technology*
[2]*School of Computer Science, University of Technology Sydney*
[3]*School of Computer Science and Engineering, University of New South Wales*
lichengzhangcg@gmail.com, xianzhi.wang@uts.edu.au, lina.yao@unsw.edu.au, zhengf@sustech.edu.cn

***Abstract*—Zero-shot object detection aims to detect and recognize objects unobserved in training samples from images. Previous studies generally utilized concept names or textual descriptions to build relationships between seen and unseen classes. However, these works rarely exploited the valuable information in textual descriptions for optimizing the network. Actually, textual descriptions contain much valuable information related to categories. Exploiting this information can help training the network and improve the detection performance. Besides, textual descriptions usually contain the names of objects that need to be detected. By using this character, we can narrow the scope of candidate unseen categories, thus can improve the detection accuracy. In this regard, we propose a novel framework that incorporates both images and their text descriptions for zero-shot object detection. In particular, we employ text convolutional neural network (CNN) and Faster R-CNN to extract text features and image features respectively, and combine them to optimize the regions that contain objects in images and to classify those newly detected objects simultaneously. Besides, we try extracting potential object labels directly from textual descriptions and introducing online hard example mining (OHEM) to assist with object classification and network optimization. Our extensive experiments on two public datasets demonstrate the superior performance of our approach to state-of-the-art methods.**

***Index Terms*—zero-shot object detection, textual description, word vector representation, convolutional neural network, online hard example mining**

I. Introduction

In recent years, there has been significant progress in traditional object detection research [1], [2]. These efforts usually require lots of training samples to be collected and annotated for each object class. However, the objects to be detected may not always appear in training examples. This poses the challenge of zero-shot object detection [3], [4].

Zero-shot object detection aims to simultaneously detect and recognize objects that do not exist in training examples. A zero-shot object detection method depends on a large amount of labeled training data of seen classes and the knowledge about how an unseen class is semantically related to the seen classes [5]. Seen and unseen classes are often related in a high-dimensional space, which is called semantic space. Previous studies on zero-shot object detection usually learned embeddings from image space to semantic space [3], [4], [6], or project image space and semantic space into a joint embedding space [7]. The semantic space comes from concept names or textual descriptions. They determine the label for each object proposal by choosing the most similar category based on the learned embeddings. Although concept names or textual descriptions can contribute to predicting the category regarding each bounding box, previous work rarely has exploited textual descriptions to help training the network.

Actually, textual descriptions usually contain valuable information about the categories of objects. And since words with similar meanings usually have similar vector representations, we can transfer our network from seen classes to unseen categories. Therefore, considering textual descriptions can improve the prediction of object labels for both seen and unseen categories. Furthermore, textual descriptions often contain names of objects explicitly. By exploiting these names, we can quickly narrow down the label of an unknown object to a small scope of categories and make the prediction more accurate.

Inspired by the above, in this paper, we propose to incorporate textual descriptions into Faster R-CNN for zero-shot object detection. We first extract text features and image features separately. Then we concatenate image features and text features to assist to detect and recognize objects in images.

During testing, we use semantic embeddings only to determine the category of each object proposal. We compute the cosine similarity between two word vector representations of the predicted label and each unseen category, and choose the category with highest similarity as the label of the object proposal.

To summarize, we make the following contributions: (1)We incorporate textual descriptions into Faster R-CNN to contribute to network training; (2) We model textual descriptions via word vector representations and use deep convolutional neural network (CNN), which has shown excellent performance in natural language processing [8]–[10], to extract text features; (3) We derive object names from textual descriptions as the candidate labels and predict the most similar name as the category of each object to improve the prediction accuracy; We also introduce online hard example mining (OHEM), used initially for traditional object detection task [11], to optimize the network; (4) We have conducted extensive experiments to demonstrate the superiority of our proposed method.

The rest of this paper is organized as follows. We first

†Corresponding Author.

review the related works. Then we describe our proposed approach in detail. After that, we give the experimental results, followed by the conclusion.

II. Related Work

A. Object Detection

Object proposal methods have demonstrated the superiority for detecting objects in the image [1], [2], [12], [13].

R-CNN [12] first generates a large number of region proposals and utilizes CNN to extract fixed-dimension features from each region proposal. Then they use a support vector machine (SVM) to classify region proposals into different categories. Fast R-CNN [13] also generates lots of region proposals. Then it exploits CNN rather than SVM to perform multi-class classification and predict refined bounding boxes. Faster R-CNN [1] utilizes region proposal network (RPN) to generate region proposals. With RPN, it can be trained in an end-to-end manner. Mask R-CNN [2] adds an image segmentation branch and replaces RoI pooling by RoI Align to improve object detection performance.

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

B. Zero-shot Learning

Zero-shot learning can be classified into two categories: classifier-based methods and instance-based methods [14]. The former directly learns a classifier for unseen classes. The latter obtains labelled instances for unseen classes and uses them to learn a classifier. For classifier-based methods, researchers usually learn a binary one-vs-rest classifier for each seen class and a corresponding function between seen and unseen classes. Then they produce unseen classifiers according to the corresponding function [15], [16]. Some other works build unseen classifiers utilizing the relationship between seen and unseen classes [17], [18], for example, cosine similarities in the semantic space [17]. For instance-based methods, some works project feature space and semantic space into a common space to obtain labeled instances for unseen classes [19]. Some other works obtain unseen labeled instances by borrowing instances from seen classes [20]. And some works synthesize pseudo instances to produce labeled instances for unseen classes [21], [22].

C. Zero-shot Object Detection

ZSD is a recently introduced problem that aims to simultaneously locate and recognize objects unobserved in training samples [3], [4], [6], [7].

Rahman et al. [3] utilize Faster R-CNN and a zero-shot recognition framework named ConSE to detect unseen classes. They design a semantic alignment network to project visual space to semantic space. 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, which is similar to our method to determine the category of each object proposal. Bansal et al. [4] first learn a zero-shot object detection model on seen classes, which is a background-aware model. Then they use an iterative Expectation Maximization like algorithm to spread background boxes over a wider range of visual concepts. They also choose to project visual space to semantic space. Demirel et al. [6] build a hybrid model, which consists of two learned region embeddings. Both region embeddings are compared with true class embeddings to get region detection results according to similarities. Li et al. [7] utilize Faster R-CNN and natural language descriptions for zero-shot object detection. They make use of LSTM to model textual descriptions and perform element-wise multiply between image space and semantic space to predict whether textual descriptions fit the object proposal.

III. Methodology

A. Framework Overview

The overall framework of our proposed method is shown in Fig. 1. It consists of two parts. The first part is Faster R-CNN, which is utilized to extract features from the image and perform object classification and bounding box regression. The second part is text CNN, which is employed to extract features from textual descriptions. Image features and text features are concatenated as the input of the fully connected layers. Then the network performs classification and box regression.

The loss function of the network consists of two parts: the RPN loss function and the detection loss function. The 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. Both loss functions are the same as those used in [1].

B. Image Feature Extraction

We use Faster R-CNN to extract features from the image. The backbone of Faster R-CNN is the Inception-ResNet v2 model [23]. The network first extracts a global feature map from the image. Then it uses RPN to generate a large number of region proposals, which is utilized to distinguish between the foreground and background. After that, the network randomly chooses region proposals and makes use of RoI pooling to extract fixed-size features from each region proposal. Then region CNN is utilized to further extract features from RoI features. Region CNN is part of the Inception-ResNet v2 model. After that, we obtain image features for each region proposal.

C. Text Feature Extraction

We use the text CNN proposed in [24] to extract text features from textual descriptions. 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 output, followed by a max-pooling layer. There is a max-pooling layer between two convolutional blocks, whose

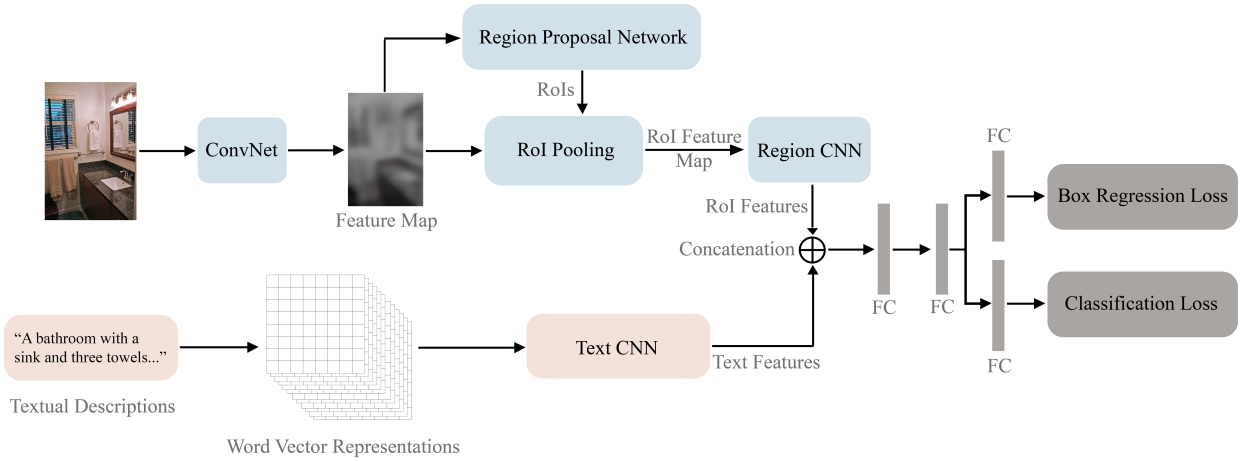

Fig. 1. Overview of our proposed approach. The upper half in the figure is Faster R-CNN. The lower half in the figure is text CNN, which is utilized to extract features from textual descriptions. Region CNN is used to extract features from RoI features further. Text features are concatenated with image features as the input of the following part. "FC" represents a fully connected layer. Text features are used to determine which region proposals should be predicted as objects and which not.

TABLE I
EXPERIMENTAL COMPARISON OF DIFFERENT METHODS ON THE 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
SAN [19]	35.70	26.30	14.50	6.80	5.90	3.10
SB [20]	34.46	24.39	12.55	6.06	4.09	2.43
DSES [20]	40.23	27.19	13.63	7.78	4.75	2.34
LAB [20]	31.86	20.52	9.98	8.43	5.40	2.74
ZSD-LSTM [22]	45.50	34.30	18.10	9.70	7.20	4.20
ZSD-CNN-ohem*	**54.14**	**47.18**	**38.12**	**23.00**	**18.04**	**13.33**

TABLE II
EXPERIMENTAL COMPARISON OF WHETHER ADDING TEXTUAL DESCRIPTION INTO FASTER R-CNN. 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-w/o	26.96	22.17	17.20	6.83	4.76	2.99
ZSD-CNN-normal	**32.56**	**27.23**	**22.01**	**10.20**	**7.58**	**5.13**

filter numbers are different, for example, 64 and 128. Each convolutional block consists of two convolutional layers, each one followed by a batch normalization layer and a ReLU activation. The kernel size of each convolutional layer is 3. We use shortcut connections between neighboring convolutional blocks to reduce degradation, which was recommended in [25]. The details of text CNN and one convolutional block are elaborated in [24].

Before fed into text CNN, textual descriptions are transformed into word vector representations word by word using public word vector representations. The input of text CNN are word vector representations whose height is 1, width is N, and the dimension of the input channel is 300. N represents the number of words in each textual description, and 300 is the dimension of one word vector representation. For public word vector representations, we use GloVe [26] to extract vector representations for each word.

D. OHEM Method

Traditionally, to train the Faster R-CNN network, it usually needs to randomly select k examples from all region proposals produced by RPN to perform gradient descent and optimize the network, for instance, 64 random samples. Shrivastava et al. [11] recommended OHEM training for traditional object detection, which selected hard examples to optimize the network. In this paper, we employ OHEM method to train our zero-shot object detection network. Specifically, we first randomly select more examples, for example, 640 region proposals, from all region proposals, and perform network inference. Then we rank the losses of these region proposals and choose samples whose losses are among top k. After that, we optimize the network using these selected samples. Using OHEM method, we can reduce the losses of hard examples, which, in theory, is better than using random samples.

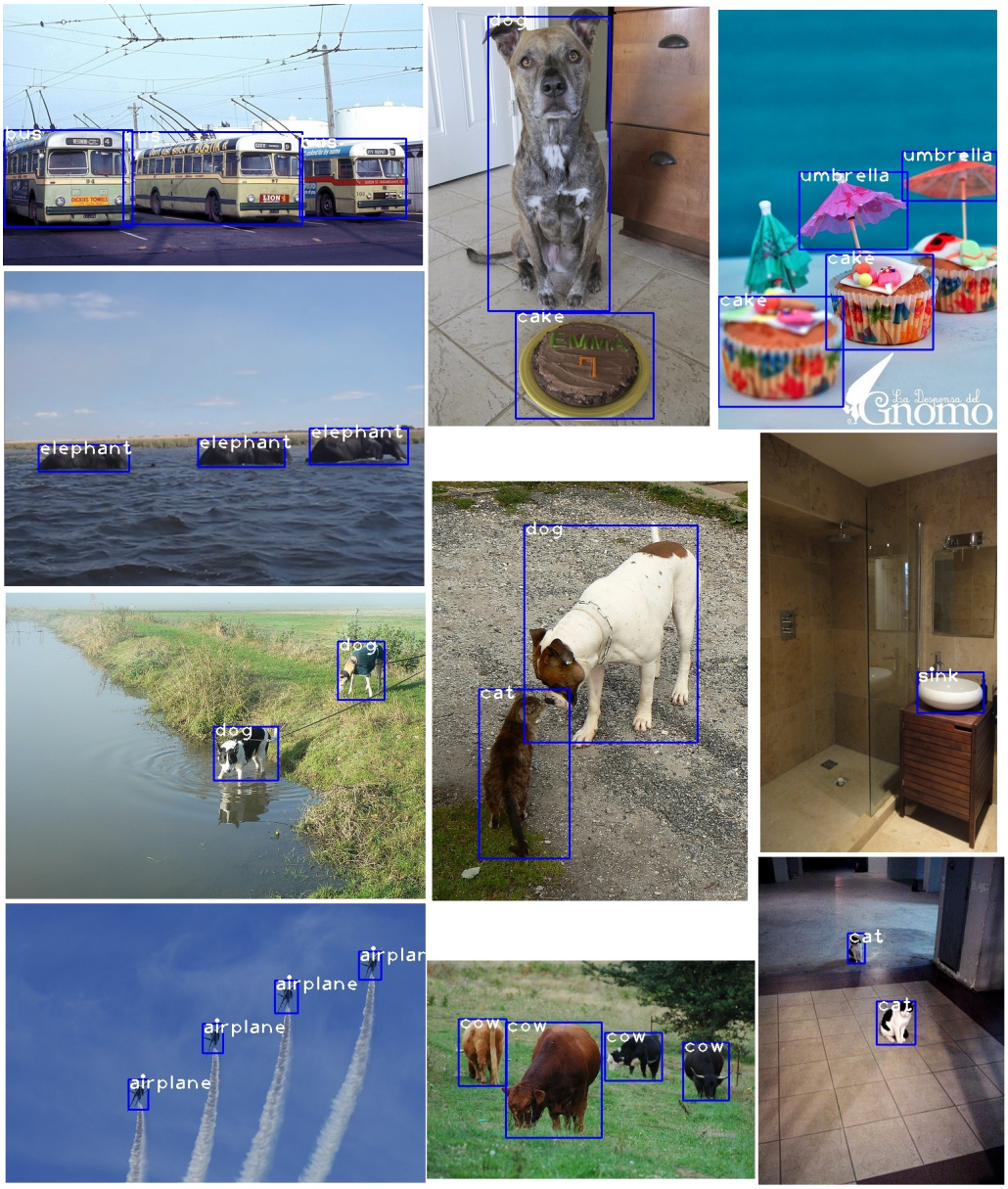

Fig. 2. Selected examples of zero-shot object detection

TABLE III
EXPERIMENTAL COMPARISON OF USING NAMES OF UNSEEN OBJECTS FROM TEXTUAL DESCRIPTIONS AS CANDIDATE LABELS OR USING ALL UNSEEN CATEGORIES AS CANDIDATE LABELS. 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	32.56	27.23	22.01	10.20	7.58	5.13
ZSD-CNN-normal*	**50.92**	**42.99**	**33.98**	**18.31**	**13.28**	**8.82**
ZSD-CNN-ohem	33.91	29.16	23.60	12.53	10.20	7.75
ZSD-CNN-ohem*	**54.14**	**47.18**	**38.12**	**23.00**	**18.04**	**13.33**

IV. EXPERIMENTS

A. Datasets

1) MS COCO: This dataset was designed for detecting and segmenting objects [27]. It contains caption descriptions for every image. We utilize bounding box positions of objects and captions of each image for experiments.

2) Visual Genome: This dataset was designed for visual relationship understanding [28]. It contains rich information about regions and objects. We focus on categories and bound-

TABLE IV
EXPERIMENTAL COMPARISON OF USING NORMAL TRAINING METHOD OR OHEM TRAINING METHOD FOR OPTIMIZING THE NETWORK. RECALL@100 IS USED AS THE EVALUATION METRIC. LARGER VALUE IS BETTER.

IoU	MS COCO			Visual Genome		
	0.4	0.5	0.6	0.4	0.5	0.6
ZSD-CNN-normal	32.56	27.23	22.01	10.20	7.58	5.13
ZSD-CNN-ohem	**33.91**	**29.16**	**23.60**	**12.53**	**10.20**	**7.75**

ing box positions of objects, as well as region descriptions for experiments.

B. Data Split

For the zero-shot learning setting, unseen objects are not allowed to exist in training samples. In terms of the MS COCO dataset, following [4] and [7], we chose the same 48 categories for training and the same 17 categories for testing. We used the 2014 training set as training data and used the data listed by [4] for testing, which contains 6618 samples. We removed the images in training examples that contain objects from unseen classes. For the Visual Genome dataset, we chose the same 478 classes for training and the same 130 classes for testing. We used examples from part-1 for training and the data listed by [4], which has 7819 examples, for testing.

For the MS COCO dataset, the number of filters in the last convolutional layer of the text CNN module is 128. Then the dimension of text features is 1024. They are concatenated with image features, whose dimension is 1536, as the input of the following part. Besides, the descriptions are truncated or padded to a fixed length, 128 words. And for the Visual Genome dataset, the last convolutional layer has 32 filters. Therefore, the output dimension of the text CNN module is 1280. The descriptions for Visual Genome are truncated or padded to 640 words.

C. Training and Testing Settings

We train the whole network in an end-to-end manner. Firstly, before training, we initialize the parameters of the network. The backbone of Faster R-CNN is the Inception-ResNet v2 model. Its parameters are initialized using the pertained model on ImageNet by [23]. The parameters of the rest part of the network are randomly initialized, including the text CNN module. Then the network is trained with a learning rate of 10^{-4}. The learning rate is decayed by 0.5 after every epoch. During training, we use a stochastic gradient descent optimizer. When using OHEM training, the parameters of the network are initialized using parameters of the model learned before.

During the testing, given one image and its descriptions, the network locates and recognizes unseen objects. Each unseen object is first classified as the most similar category from seen classes. Then, for each bounding box, we compute the cosine similarity between the predicted category and every label name from unseen classes. Finally, we choose the category with the highest similarity as the label of this bounding box.

D. Evaluation Metric

Following [4] and [7], we use recall as the evaluation metric for fair comparison. We compare our proposed method with three previous works. These methods are from [3], [4] and [7]. We use the same names as the original works, to present the experimental results. We utilize "ZSD-LSTM" to represent the method in [7]. Following [4], for each image, we keep 100 detection bounding boxes whose classification scores are larger than 0.07 and among the top 100 in all bounding boxes, to compute the recall.

E. Results

Table I shows the experimental comparison of different methods on the two datasets. The table shows the Recall@100 performance of different methods. A larger recall is better. We use three different IoU overlap thresholds in the experiments: 0.4, 0.5, 0.6. For previous methods, we use the experimental results from [4] and [7]. Because [3] didn't conduct experiments on MS COCO and Visual Genome datasets, we use the results from [7] for their method. For our method, the name containing "ohem" means using the OHEM method during network training. The name containing "*" represents the method that uses names of unseen objects from textual descriptions as candidate labels. And, if we can't extract labels from textual descriptions, we still use all unseen categories as candidate labels.

Table I shows that our proposed method performs better than previous methods. It demonstrates the effectiveness of our proposed method, which incorporates textual descriptions into Faster R-CNN, and uses names of unseen objects from textual descriptions as candidate labels, and applies OHEM method during network training.

F. Component Studies

In this part, we present the extensive experimental results to study the effects of different components in our approach. We still use recall as the evaluation metric. We use three different IoU overlap thresholds in the experiments: 0.4, 0.5, 0.6. For our method, the name containing "w/o" means not adding textual descriptions in the network. The names containing "normal" mean using normal training method. The names containing "ohem" mean using the OHEM method during network training. The names without "*" represents the method that uses all unseen categories as candidate labels. The names containing "*" represent the method that uses names of unseen objects from textual descriptions as candidate labels, and if we can't extract labels from textual descriptions, we still use all unseen categories as candidate labels.

1) Adding Textual Descriptions: Table II shows the results concerning adding textual descriptions into Faster R-CNN. Textual description improves the recall by a large margin. For example, adding textual descriptions improves the recall by 5.6% to 32.56% for the MS COCO dataset and threshold of 0.4. It contributes to network training and improve the detection accuracy. The results also show the effectiveness of public word vector representations and deep CNN in text representation and feature extraction.

2) Using Labels from Textual Descriptions: Table III shows the results of using object names from textual descriptions as candidate labels and using all unseen categories as candidate labels for normal training and OHEM training. They turn out to improve the recall significantly. For ohem method trained with MS COCO dataset and threshold 0.4, the recall is increased from 33.91% to 54.14%.

3) Using OHEM Training: Table IV gives the experimental results of using normal training method and using OHEM method to optimize the network, respectively. From the results, we can find that OHEM training can slightly improve the recall, nearly 2 percent, compared with normal training.

G. Qualitative Results

In this part, we present some detection results to demonstrate the performance of our proposed method. We use MS COCO as an example. The detection results are shown in Fig. 2. From the figure, we can find that although there are several missed detections, for example, in the right top image, one cake and one umbrella are missed, our proposed method is capable of detecting novel objects that are unseen in training examples. In the future, we will continue to improve the framework to achieve better performance.

V. Conclusion

In this paper, we propose a new approach for zero-shot object detection. It incorporates textual descriptions into Faster R-CNN to contribute to network training, and uses deep CNN to extract features from word vector representations of textual descriptions and exploits online hard example mining to optimize the network. It also uses object names from textual descriptions as candidate labels and chooses the category from candidate labels with the highest similarity as the label of each object proposal. We conduct extensive experiments on two real-world datasets. The experimental results demonstrate that our proposed approach performs better than previous methods.

Acknowledgement

This work was supported partially by the National Natural Science Foundation of China under Grant 61972188.

References

[1] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015.

[2] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," *CoRR*, vol. abs/1703.06870, 2017.

[3] S. Rahman, S. H. Khan, and F. Porikli, "Zero-shot object detection: Learning to simultaneously recognize and localize novel concepts," *CoRR*, vol. abs/1803.06049, 2018.

[4] A. Bansal, K. Sikka, G. Sharma, R. Chellappa, and A. Divakaran, "Zero-shot object detection," *CoRR*, vol. abs/1804.04340, 2018.

[5] L. Zhang, T. Xiang, and S. Gong, "Learning a deep embedding model for zero-shot learning," *CoRR*, vol. abs/1611.05088, 2016.

[6] B. Demirel, R. G. Cinbis, and N. Ikizler-Cinbis, "Zero-shot object detection by hybrid region embedding," *CoRR*, vol. abs/1805.06157, 2018.

[7] Z. Li, L. Yao, X. Zhang, X. Wang, S. Kanhere, and H. Zhang, "Zero-shot object detection with textual descriptions," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 8690–8697, Jul. 2019.

[8] Y. Kim, "Convolutional neural networks for sentence classification," *CoRR*, vol. abs/1408.5882, 2014.

[9] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in Neural Information Processing Systems 28*, 2015, pp. 649–657.

[10] A. Conneau, H. Schwenk, L. Barrault, and Y. LeCun, "Very deep convolutional networks for natural language processing," *CoRR*, vol. abs/1606.01781, 2016.

[11] A. Shrivastava, A. Gupta, and R. B. Girshick, "Training region-based object detectors with online hard example mining," *CoRR*, vol. abs/1604.03540, 2016.

[12] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *CoRR*, vol. abs/1311.2524, 2013.

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

[14] W. Wang, V. W. Zheng, H. Yu, and C. Miao, "A survey of zero-shot learning: Settings, methods, and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 13:1–13:37, Jan. 2019.

[15] M. Elhoseiny, B. Saleh, and A. Elgammal, "Write a classifier: Zero-shot learning using purely textual descriptions," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[16] X. Wang, Y. Ye, and A. Gupta, "Zero-shot recognition via semantic embeddings and knowledge graphs," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[17] C. Gan, M. Lin, Y. Yang, Y. Zhuang, and A. G. Hauptmann, "Exploring semantic inter-class relationships (sir) for zero-shot action recognition," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, ser. AAAI'15, 2015, pp. 3769–3775.

[18] S. Changpinyo, W. Chao, B. Gong, and F. Sha, "Synthesized classifiers for zero-shot learning," *CoRR*, vol. abs/1603.00550, 2016.

[19] M. Ye and Y. Guo, "Zero-shot classification with discriminative semantic representation learning," 07 2017, pp. 5103–5111.

[20] Y. Guo, G. Ding, J. Han, and Y. Gao, "Zero-shot learning with transferred samples," *IEEE Transactions on Image Processing*, vol. 26, no. 7, July 2017.

[21] Y. Xian, T. Lorenz, B. Schiele, and Z. Akata, "Feature generating networks for zero-shot learning," *CoRR*, vol. abs/1712.00981, 2017.

[22] G. Arora, V. K. Verma, A. Mishra, and P. Rai, "Generalized zero-shot learning via synthesized examples," *CoRR*, vol. abs/1712.03878, 2017.

[23] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, inception-resnet and the impact of residual connections on learning," *CoRR*, vol. abs/1602.07261, 2016.

[24] A. Conneau, H. Schwenk, L. Barrault, and Y. LeCun, "Very deep convolutional networks for natural language processing," *CoRR*, vol. abs/1606.01781, 2016.

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[26] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543.

[27] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: common objects in context," *CoRR*, vol. abs/1405.0312, 2014.

[28] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L. Li, D. A. Shamma, M. S. Bernstein, and F. Li, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," *CoRR*, vol. abs/1602.07332, 2016.