One-Stage Open Set Object Detection with Prototype Learning

Yongyu Xiong[1,2], Peipei Yang[1,2(✉)], and Cheng-Lin Liu[1,2,3]

[1] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, People's Republic of China
xiongyongyu2019@ia.ac.cn, {ppyang,liucl}@nlpr.ia.ac.cn
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, People's Republic of China
[3] CAS Center for Excellence of Brain Science and Intelligence Technology, Beijing, People's Republic of China

Abstract. Convolutional Neural Network (CNN) based object detection has achieved remarkable progress. However, most existing methods work on closed set assumption and can detect only objects of known classes. In real-world scenes, an image may contain unknown-class foreground objects that are unseen in training set but of potential interest, and open set object detection aims at detecting them as foreground, rather than rejecting them as background. A few methods have been proposed for this task, but they suffer from either low speed or unsatisfactory ability of unknown identification. In this paper, we propose a one-stage open set object detection method based on prototype learning. Benefiting from the compact distributions of known classes yielded by prototype learning, our method shows superior performance on identifying objects of both known and unknown classes from images in the open set scenario. It also inherits all advantages of YOLO v3 such as the high inference speed and the ability of multi-scale detection. To evaluate the performance of our method, we conduct experiments with both closed & open set settings, and especially assess the performance of unknown identification using recall and precision of the unknown class. The experimental results show that our method identifies unknown objects better while keeping the accuracy on known classes.

Keywords: Object detection · Open set recognition · Prototype learning

1 Introduction

As a fundamental problem of computer vision, object detection aims at detecting objects of a certain class of interest in images, and has been studied for many years. Recently, the development of convolutional neural network (CNN) leads to remarkable breakthroughs in object detection, and CNN based methods have become the mainstream. They can be roughly grouped into two-stage detection

T. Mantoro et al. (Eds.): ICONIP 2021, LNCS 13108, pp. 279–291, 2021.
https://doi.org/10.1007/978-3-030-92185-9_23

and one-stage detection [19], where the former [5,15] usually gives better accuracy while the latter [11,14] excels in high detection speed. Thanks to all those efforts, object detection has been successfully applied in many areas including auto-driving, robot vision, smart retail supermarkets, etc. [19].

Standard object detection approaches can detect only objects of known classes, assuming that all the foreground classes to be detected should exist in the training set. However, this assumption can be violated by a common scenario in real-world applications: a test image may contain novel foreground objects of unknown classes that never appear in training images but are also of interest. This is the problem of *open set object detection* (OSD). Unfortunately, standard detection algorithms cannot deal with this case, which either neglect novel objects as background, or even identify them as a known class by mistake. To solve this problem, we focus on OSD that is able to detect both objects of known classes and objects of interest beloning to some unknown class.

Open set object detection is closely related to *open set recognition* (OSR) [3], which aims at understanding the real world with incomplete knowledge. Although both of them aim at identifying unknown classes, the OSR methods cannot be directly exploited for OSD due to two important reasons. First, for OSR, any sample outside of known classes is regarded as unknown, while for OSD, there is an extra background class in addition to the known classes that is also available during training but of no interest. Thus, a candidate object rejected by all classes of interest may be either background or unknown-class object. Second, even if we explicitly treat the background as an additional class of interest, it is still difficult for identifying a novel object, because the background has diverse appearance and is easily confused with novel objects which are not trained.

Due to the novelty and difficulty of this topic, only very few works focus on OSD. Dhamija et al. [1] first proposes the concept of OSD and defines several evaluation metrics. A series of experiments are conducted to evaluate the performance of several popular object detection methods under the open set condition using such metrics. Joseph et al. [6] proposes to use the energy model to conduct unknown category discovery and example replay for continual incremental learning. Miller et al. [12] proposes to use dropout sampling to extract label uncertainty for increasing the performance of object detection under open set condition. These works proposed the concept of OSD and some effective ideas to solve this problem, but there are still some important issues to be solved. First, existing methods are mostly designed based on two-stage object detection framework. They first generate a large number of potential foreground boxes and then adopt common open set recognition model to identify unknown objects. This strategy suffers from the drawback of most two-stage detection methods that the speed cannot satisfy the requirement of real-time applications. Second, previous OSD methods focus on avoiding classifying an unknown object as a known class by mistake. In practical open set scenarios, it is also valuable to distinguish novel objects that may be of interest from background.

To overcome the above problems, we propose a novel high-speed open set detection method that is able to correctly detect known objects while identifying unknown-class objects of interest. We first build a base model of open set detection using YOLO v3 [14], which is a representative one-stage detector with obvious higher speed than two-stage methods. By exploiting the objectness score to determine the foreground regions and setting class confidence thresholds for sigmoid outputs, we can identify the novel objects from these foreground regions when the confidence score of each class is below a preset threshold.

To improve the compactness of distribution of known classes in feature space and avoid confusion of unknown objects with known classes, we introduce the prototype classifier into YOLO v3 [14] to explicitly model the feature distribution of objects. The obtained model is trained to make features of each class concentrate in a separated and compact region in the feature space, which makes it simpler to distinguish unknown classes from known classes. Our method benefits from all advantages of YOLO v3 and gives a favorable performance on unknown object detection.

To evaluate our proposed method, we conduct experiments on both closed and open set settings. For the open set test, we divide the PASCAL VOC [2] into two parts according to classes. One part is used as known classes while the other as unknown. While existing works assess the performance of open set detection by counting how many unknown objects are misclassified as known, we adopt a more comprehensive evaluation criterion including recall and precision of the unknown objects. The experimental results show that our method can effectively identify the foreground objects of unknown classes while keeping satisfactory performances on known classes.

The main contributions of this work can be summarized as follows.

- We propose an open set object detection method within the one-stage detection framework. It has obviously higher speed than existing two-stage methods.
- We propose to construct open set object detection by integrating the prototype classifier into the framework. Exploiting the ideal feature distribution generated by the prototype classifier, it performs well on detecting both known and unknown objects.
- We evaluate our proposed method by both closed and open set experiments. We also assess the capability of the algorithm to discover unknown objects from background, which has not been considered before.

2 Related Work

In this section, we will give a brief review on the development of CNN-based object detection methods, followed by recent advances on open set object detection. Then, a short introduction to prototype learning and its application to open set recognition are present.

Object Detection and OSD. Object detection is a fundamental task in computer vision and has been studied for many years. Due to the great progress of deep neural networks in computer vision and pattern recognition, CNN-based object detection has been the mainstream method. They can be generally divided into two-stage and one-stage methods [19]. The former first generates a large number of potential foreground boxes, from which the objects are obtained by classification and bounding box regression. The latter outputs class labels and locations of the objects directly in one shot.

Faster R-CNN [15] is the most representative work on two-stage object detection. It firstly proposes to use the anchor as an object prior to generate class-agnostic foreground regions and then obtain the bounding boxes of the object by classification and regression. After Faster R-CNN [15], there are many works to improve the performance of two-stage object detection, including feature fusion and enhancement [7], etc. Nowadays, the state-of-the-art results are still held by two-stage methods.

With the advent of SSD [11], one-stage object detectors have attracted much attention because of their high computational efficiency. SSD [11] directly uses anchors with different aspect ratios and areas on multi-layer feature maps to generate category predictions and bounding box locations. Due to the real-time efficiency, many efforts have been devoted to future improve the performance of one-stage object detectors, including loss function to solve sample imbalance problem [8], and new architectures for object detection [13].

All the above methods work under the closed set assumption. Recently, a few methods are proposed to solve the open set object detection problem. Dhamija et al. [1] firstly propose the concept of open set object detection and design evaluation metrics to evaluate the performance of several popular object detectors under open set conditions. Joseph et al. [6] proposes open world object detection, which first identifies both known and unknown objects and then dynamically updates knowledge by continually incremental learning. Because it is based on two-stage object detection framework, the speed cannot satisfy the requirement of real-time applications. Miller et al. [12] proposes to use dropout sampling as an approximation to Bayesian inference over the parameters of deep neural network to extract label uncertainty, which increases the performance of object detection under open set conditions. Although SSD [11] is adopted in this method, it is still computationally expensive since it requires multiple inference passes per image.

Prototype Learning. Prototype learning is a classical and representative method in pattern recognition which uses prototype to represent the main characteristics of classes. The earliest prototype learning method is k-nearest-neighbor (K-NN). In order to reduce the heavy burden of storage space and computation requirement of K-NN, an online method called learning vector quantization (LVQ) is proposed. The LVQ has been studied in many works and there are a lot of variants [4,10]. After the arrival of the deep learning era, prototype learning can be incorporated into the deep neural network and trained in an end-to-end manner. It has played an important role in few-shot learning [16], robust representing learning [17], open set recognition [18], etc.

3 Method

3.1 Open Set Object Detection

Before introducing our method, we first formalize the problem of Open Set Object Detection (OSD). In a common object detection setting, all classes in the label space $\mathbf{Y}$ can be broadly categorized into three types [12]: **1) Known** classes in K are labeled in the training dataset and the detector is trained to detect them. **2) Known Unknown** classes in U_K exist in the training dataset but are not labeled. The detector is trained to ignore these objects which typically appear in the background. **3) Unknown Unknown** classes in U_U are not present in the training dataset. The detector has never seen objects of those classes during training and therefore has not learned to identify them. It is a challenge to identify the unknown unknowns under open set conditions.

Traditional detection only considers the problem of detecting known objects in K and neglecting objects in U_K, without paying attention to U_U. Existing works on OSD [1,12] consider the existence of U_U and make efforts on preventing misclassifying the objects in U_U as known classes. Different from the aforementioned methods, we consider OSD as a more challenging task, which is able to not only prevent misclassifying U_U as K but also distinguish them from U_K. Thus, we adopt both precision and recall of unknown classes as well as known classes to evaluate the OSD algorithm.

3.2 Open Set Object Detection Using YOLO V3

YOLOv3 [14] has proven an efficient object detection algorithm in a lot of applications. Thus, we first adapt it to the OSD setting by a simple modification. The architecture of YOLOv3 consists of three components including feature extraction network (Darknet53), across scale multi-layer feature fusion, and detection head. Darknet53 first produces powerful feature representations and then combines intermediate layer feature maps to produce multi-scale feature maps. The detection head predicts class condition probability, objectness score, and location offset for each predefined anchor on every pixel of the feature map.

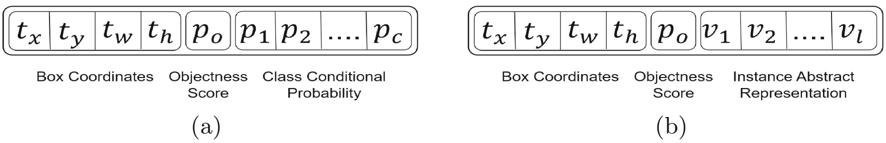

Fig. 1. Comparison of the outputs of detection heads. (a) Original YOLO v3, where c is the number of classes. (b) Our method based on prototype classifier, where l is the dimension of abstract representations.

The detailed output of the prediction head is illustrated in Fig. 1(a). YOLO v3 [14] regards the objectness score p_0 as the confidence of foreground and uses

C sigmoid functions to act as one-vs-all classifiers giving the class conditional probabilities $\{p_i\}_{i=1}^{C}$. During detection, only the candidate boxes with $p_0 > \tilde{p}_0$ are considered and the probability that the object belongs to class i is calculated by $p_0 \times p_i$. In this process, since all classifiers give a low response, the unknown objects are usually simply neglected as background and cannot be detected.

We can easily adapt YOLO v3 [14] to an open set object detector by exploiting the output of the detection head in a different way. Considering that in the training phase, the objectness score regards objects of all foreground classes as positive and background as negative, it is expected to give a higher response to similar foreground objects in the test phase. Therefore, if a candidate box is given low responses by all classifiers but a high objectness score, there is a large probability that the box contains an object of unknown class. Based on the analysis, we can identify an object with a high objectness score but lower responses from all classifiers as an unknown class. Although such a strategy provides a straightforward way to identify objects of unknown classes with YOLO v3, an inherent defect of the method affects its accuracy on unknown classes. The C sigmoid classifiers define C linear classifiers in an implicit feature space determined by the last 1×1 convolution, which cannot produce a compact distribution for each class. Therefore, features of unknown objects are liable to overlap with known classes, which makes it difficult to obtain satisfactory performance of unknown identification. In order to solve this problem, we resort to generative model and exploit prototype classifier to obtain a compact feature representation for each class.

3.3 Prototype Based Open Set Object Detection

In this section, we will present our prototype based open set object detection, which is illustrated in Fig. 2, after a brief introduction on prototype learning. By

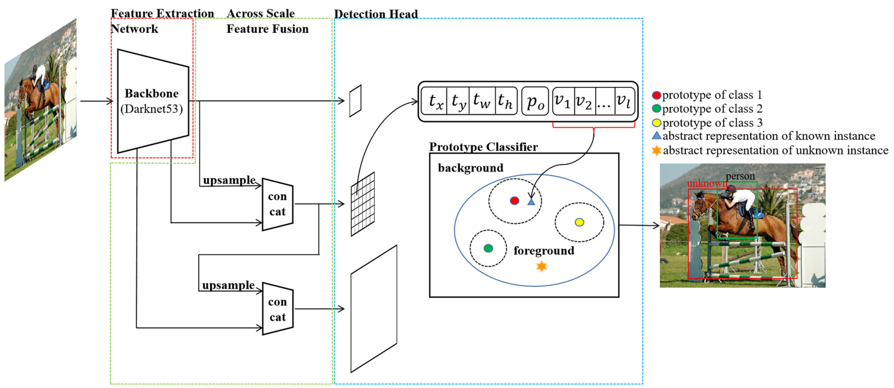

Fig. 2. The overall framework of our proposed open set object detection. Person class is known and horse class is unknown. For clarity, only the prototype classifier on mid level of feature maps is shown.

introducing prototype classifier, which has proven effective on open set recognition [18], into the framework of YOLO v3, the detector obtains the ideal capability of discovering unknown objects. Meanwhile, all advantages inherently built-in one-stage detector are inherited.

Different from sigmoid classifiers which give the class conditional probabilities directly, prototype learning maintains several prototypes in the feature space for each class and uses prototype matching for classification [17]. The prototypes are feature vectors representing the main characteristics of each class, which are denoted as m_{ij} where $i \in \{1, 2, ..., C\}$ represents the class index and $j \in \{1, 2, ..., n\}$ represents the prototype index in each class. Here we assume each class having the equal number of n prototypes and this assumption can be easily relaxed in the real application.

To obtain the class of an input X, we first calculate its abstract representation $f(X)$ by the feature extractor. Then, $f(X)$ is compared with all prototypes in each class and the nearest prototype according to the distance metric defined in the space is found. This sample is assigned by the class of the nearest prototype using Eq. 2.

$$d(f(X), m_{ij}) = \|f(X) - m_{ij}\|^2 \tag{1}$$

$$y^* = \arg\min_i \{\forall_j \ d(f(X), m_{ij})\} \tag{2}$$

Recently, some works [17,18] attempt to integrate the prototype learning with convolutional neural network, and obtain remarkable progress in robust classification and open set recognition. Following these works, we adopt a prototype classifier to replace the sigmoid classifiers in the detection head of YOLO v3. The output of detection head with respect to each predicted box x is composed of three components: location offset, objectness score, and abstract representation of this box $f(x) \in \mathbb{R}^l$, as illustrated in Fig. 1(b). During training and inference, we use this feature vector to compare with prototypes of each class. Considering that YOLO v3 uses multi-scale feature maps to detect objects of different scales and the feature distributions are different in three levels, we set a prototype classifier on each level of feature maps.

In the inference stage, the model accepts an image as input and generates feature maps. The detector generates predicted boxes based on the predefined anchors per pixel in every level of feature maps. We use the objectness score as measurement of foreground and select the candidate boxes with $p_0 > \tilde{p}_0$ as potential foreground regions. Then, the distances between $f(x)$ and all prototypes are calculated according to Eq. 1. Finally, we assign the class label of each candidate box according to Eq. 4, where T_{dis} is the predefined distance threshold.

$$(i^*, j^*) = \arg\min_{i,j} d(f(x), m_{ij}) \tag{3}$$

$$y^* = \begin{cases} i, \text{ if } d(f(x), m_{i^*j^*}) \leq T_{dis}; \\ \text{unknown, if } d(f(x), m_{i^*j^*}) > T_{dis}. \end{cases} \tag{4}$$

3.4 Model Training

The proposed model contains two parts of parameters to learn. One is the parameters of the CNN extractor θ, and the other is the prototypes in each class $M = \{m_{ij}|i = 1, ..., C; j = 1, ..., n\}$. During the training stage, the two parts of parameters are trained jointly in an end-to-end manner.

In our proposed prototype based open set object detection, the distance Eq. 1 is used as a measurement of similarity between instances and prototypes. Thus, we use distance to define the probability $p(m_{ij}|x)$ that an instance x belongs to a prototype m_{ij}.

$$p(m_{ij}|x) = \frac{e^{-d(f(x),m_{ij})}}{\sum_{p=1}^{C}\sum_{q=1}^{n} e^{-d(f(x),m_{pq})}} \tag{5}$$

According to Eq. 5 we can easily infer the posterior probability of each input instance $p(y = i|x) = \sum_{j=1}^{n} p(m_{ij}|x)$. Then, we can use cross entropy (CE) loss (called as distance-based CE (DCE) in this case [17]) to optimize the model. Although training with DCE loss can make the model classify the data correctly, its robustness is insufficient under open set conditions since it cannot make instances of the same class gather compact enough. To improve the robustness, we add a regularization named prototype loss (PL) [17]

$$pl((x, y); \theta, M) = \|f(x) - m_{yj}\|^2, \tag{6}$$

where m_{yj} is the nearest prototype to $f(x)$ belonging to ground truth class y. Thus, the classification loss of our proposed model can be defined as

$$L_{cls}((x, y); \theta, M) = -\log p(y|x) + \lambda pl((x, y); \theta, M). \tag{7}$$

Using the classification loss defined in Eq. 7 and keeping other terms from the original YOLO v3, we obtain the total loss of our proposed open set object detection as:

$$L((x, y); \theta, M) = L_{cls}((x, y); \theta, M) + \lambda_{obj} L_{obj}((x, y); \theta) + \lambda_{box} L_{box}((x, y); \theta), \tag{8}$$

where $\lambda_{obj}, \lambda_{box}$ are hyper parameters. In the training stage, we first randomly initialize the prototypes in each class of different feature levels and then optimize parameters of network and class prototypes jointly in an end-to-end manner according to Eq. 8.

4 Experiments and Results

4.1 Experimental Setting

In order to comprehensively evaluate the performance of the proposed method, we conduct both closed set object detection that does not include unknown classes and open set object detection with unknown classes. We introduce the experimental settings for both of them in the following.

Closed Set Experiments. For closed set detection experiments, we evaluate the proposed approach on PASCAL VOC [2] benchmark following standard training and test protocols. We use the training and validation sets of PASCAL VOC 2007 and 2012 for training and PASCAL VOC 2007 test data for test. The performance is measured by average precision (AP) 0.5 [2].

Open Set Experiments. Open set object detection is a relatively new task. Existing datasets [2,9] are unsuitable for evaluation of OSD algorithms since they do not explicitly label the objects of unknown classes. Previous methods [1,6] adopt twenty classes from PASCAL VOC [2] as training set and choose sixty classes from Microsoft COCO [9] that are different from training classes as the test set. Since the scenes and styles of these two datasets are obviously different, this setting cannot accurately evaluate the performance of the detector under open set conditions. Therefore, we have to propose a protocol to adapt the existing dataset to open set conditions.

In order to build a dataset containing labels of unknown unknowns, we divide PASCAL VOC [2] into two parts. The first part $\{\mathcal{D}_K^{train}, \mathcal{D}_K^{test}\}$ **only** contains training and test images belonging to the first N_1 classes that act as known classes, while the second part $\{\mathcal{D}_U^{train}, \mathcal{D}_U^{test}\}$ contains the training and test images of the remaining N_2 classes that act as unknown unknowns. We use $\mathcal{D}_K^{train}$ to train the model, which ensures that the model does not see any unknowns during training. Then, $\mathcal{D}_U^{train}$ is used as evaluation set to select suitable threshold for identifying unknowns. Finally, we conduct a closed set test using $\mathcal{D}_K^{test}$ and an open set test using $\mathcal{D}_K^{test} \cup \mathcal{D}_U^{test}$ to evaluate the OSD algorithms.

Previous works [1,6] focus on preventing misclassification of a unknown objects as known class, they use the Absolute Open Set Error (A-OSE) [12] or Wilderness Impact (WI) [1] as measurement. In this paper, we consider an OSD algorithm should not only reduce the misclassification of unknowns but also distinguish unknowns from the background. Therefore, we test the ability of algorithms to discover the candidate unknown objects from background by regarding unknown objects as a special class and calculating their recall and precision.

4.2 Implementation Details

We use PyTorch for implementation[1], adopt 4 GPUs for training with a batch size of 64 (16 images per GPU) using SGD, and optimize for 300 epochs in total. First three epochs are used for warmup and the initial learning rate is set to 0.01. Then, onecycle learning rate scheduler is used and the final learning rate is 0.002. We use a weight decay of 0.0005 and a momentum of 0.937. Input images are resized to 640×640, and we also perform random horizontal image flipping, mosaic, color space transformation, and random scale for data augmentation.

[1] https://github.com/ultralytics/yolov3.

4.3 Main Results

Results on Closed Set Detection. At first, in order to evaluate the performance of our proposed method on closed set object detection, we conduct common object detection experiments on PASCAL VOC [2] datasets. For the original YOLO v3 with a sigmoid classifier, we follow the standard setting. For our proposed method, we use one prototype with the 128-dimensional feature for each class on each level of features, keeping other settings unchanged. From Table 1, we can see that our method has comparable performance as the original YOLO v3. Assessing with precision and recall, our proposed method can surpass by 1% than original YOLO v3.

Table 1. Detection performance under closed set setting

Model	Precision	Recall	mAP@0.5
Sigmoid classifier (original yolo v3)	0.777	0.822	**0.821**
Prototype classifier (ours)	**0.781**	**0.831**	0.818

Results on Open Set Detection. As presented in Sect. 3.2, YOLO v3 can be used as an open-set classifier by modifying the classification strategy. A candidate box with a high objectness score but low responses on all sigmoid classifiers is identified as unknown. In contrast, our proposed method uses the objectness score to select the foreground candidate boxes and exploit the prototype classifier to identify the unknowns.

According to the open set setting, we choose the first ten classes from PASCAL VOC as known and the remaining classes as unknown. For both methods, we train the model using the set of known classes and test on the union of known and unknown classes. During test, we adopt the same threshold of objectness score for both methods and use $\mathcal{D}_U^{train}$ as the evaluation set to determine the suitable threshold. Table 2 shows that our proposed method performs better than YOLO v3 in discovering unknown classes, which verifies the advantage of compact feature representations on unknown discovery. We also use t-SNE to visualize the feature distribution of test samples of unknown class and known classes to further verify the effectiveness of our proposed method. From Fig. 3, we can see the features of each known class gather together, while the features of unknown class are distributed far away from the center.

Table 2. Detection performance under open set setting

Model	Category	Precision	Recall
Sigmoid classifier (original yolo v3)	Known Classes	0.572	**0.785**
Prototype classifier (ours)	Known Classes	**0.595**	0.747
Sigmoid classifier (original yolo v3)	Unknown Classes	0.271	0.162
Prototype classifier (ours)	Unknown Classes	**0.322**	**0.210**

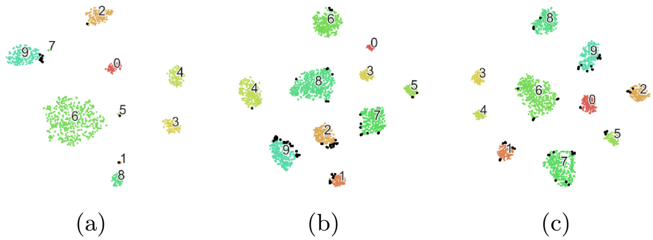

Fig. 3. Feature distribution of samples. Black points and colored points represent unknown class and known classes respectively. Subfigures (a-c) correspond to level 1–3 of the feature map. (Color figure online)

4.4 Configuration of Prototype on Different Scales

YOLO v3 [14] adopts multi-scale feature map fusion strategy similar to FPN [7] to improve the object detection performance. Each level of multi-scale feature maps is responsible for detecting objects of a scale, and we assume that feature maps of different scales should have different feature distributions. In order to verify this assumption, we design two different settings for prototypes: 1) shared setting: multi-scale features share the same class prototypes; 2) separated setting: each scale of feature maps owns a separate set of class prototypes. From Table 3, we can see that the separated setting performs much better than the shared setting, which indicates that each scale of the feature map indeed has a different distribution. In order to further investigate the distributions of abstract representations of samples, we adopt t-SNE to visualize them. From Fig. 4, the separated setting can obtain a more separable classification surface than the shared setting, and the samples within the same class gather more compact. Thus, it gives better performance.

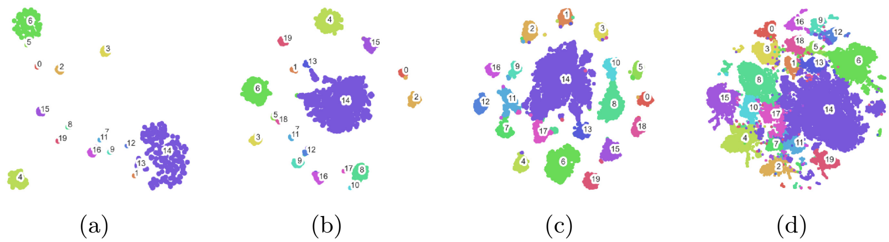

Fig. 4. Visualization of the feature distribution for different configurations of prototype in closed set test. Subfigures (a-c) correspond to the first to third level in the separated setting, while (d) corresponds to the shared setting.

Table 3. Detection performance with different prototypes setting

Model	Precision	Recall	mAP@0.5
Separated setting	**0.781**	0.831	**0.818**
Shared setting	0.764	**0.834**	0.809

5 Conclusion

In this paper, we focus on the problem of open set object detection which intends to detect unseen objects from images that may be of interest. To this aim, we propose a method by incorporating the idea of prototype learning into the framework of popular YOLO v3. It inherits the outstanding detection performance from YOLO v3 and obtains the ability of identifying foreground objects of unknown classes by exploiting the prototype classifiers. As a one-stage detection approach, it benefits from higher inference speed than existing OSD methods that are mostly two-stage. The experimental results show that our method is able to effectively identify unseen objects of unknown classes, while keeping the performance on known objects. By visualizing the distribution of feature representations, we see that samples of different classes are well separated in the feature space. This characteristic is in favor of unknown class identification, which explains the effectiveness of our method from one perspective. Furthermore, our method can be easily adapted to other one-stage detection methods and is expected to be effective too.

Acknowledgments. This work has been supported by the National Key Research and Development Program Grant No. 2018AAA0100400, the National Natural Science Foundation of China (NSFC) grant 61721004.

References

1. Dhamija, A.R., Gunther, M., Ventura, J., Boult, T.E.: The overlooked elephant of object detection: open set. In: 2020 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1021–1030 (2020)
2. Everingham, M., Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. Int. J. Comput. Vis. **88**(2), 303–338 (2010). https://doi.org/10.1007/s11263-009-0275-4
3. Geng, C., Huang, S.J., Chen, S.: Recent advances in open set recognition: a survey. IEEE Trans. Pattern Anal. Mach. Intell. **43**(10), 3614–3631 (2021)
4. Geva, S., Sitte, J.: Adaptive nearest neighbor pattern classification. IEEE Trans. Neural Netw. **2**(2), 318–322 (1991)
5. Girshick, R.: Fast r-cnn. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 1440–1448 (2015)
6. Joseph, K.J., Khan, S., Khan, F.S., Balasubramanian, V.N.: Towards open world object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2021)

7. Lin, T.Y., Dollar, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 936–944 (2017)
8. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollar, P.: Focal loss for dense object detection. IEEE Trans. Pattern Anal. Mach. Intell. **42**(2), 318–327 (2020)
9. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
10. Liu, C.L., Eim, I.J., Kim, J.: High accuracy handwritten Chinese character recognition by improved feature matching method. In: Proceedings of the Fourth International Conference on Document Analysis and Recognition, vol. 2, pp. 1033–1037 (1997)
11. Liu, W., et al.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
12. Miller, D., Nicholson, L., Dayoub, F., Sunderhauf, N.: Dropout sampling for robust object detection in open-set conditions. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 3243–3249 (2018)
13. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788 (2016)
14. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement, Computer Vision and Pattern Recognition arXiv preprint arXiv:1804.02767 (2018)
15. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. **39**(6), 1137–1149 (2017)
16. Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. In: Advances in Neural Information Processing Systems, vol. 30, pp. 4077–4087 (2017)
17. Yang, H.M., Zhang, X.Y., Yin, F., Liu, C.L.: Robust classification with convolutional prototype learning. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3474–3482 (2018)
18. Yang, H.M., Zhang, X.Y., Yin, F., Yang, Q., Liu, C.L.: Convolutional prototype network for open set recognition. IEEE Trans. Pattern Anal. Mach. Intell. 1 (2020, early access)
19. Zou, Z., Shi, Z., Guo, Y., Ye, J.: Object detection in 20 years: A survey. arXiv preprint arXiv:1905.05055 (2019)