

# Expanding Low-Density Latent Regions for Open-Set Object Detection

Jiaming Han<sup>1,2\*</sup>, Yuqiang Ren<sup>3</sup>, Jian Ding<sup>1,2</sup>, Xingjia Pan<sup>3</sup>, Ke Yan<sup>3†</sup>, Gui-Song Xia<sup>1,2†</sup>

<sup>1</sup>NERCMS, School of Computer Science, Wuhan University

<sup>2</sup>State Key Lab. LIESMARS, Wuhan University

<sup>3</sup>YouTu Lab, Tencent

{hanjiaming, jian.ding, guisong.xia}@whu.edu.cn

{condiren, kerwinyan}@tencent.com, xjia.pan@gmail.com

## Abstract

Modern object detectors have achieved impressive progress under the close-set setup. However, open-set object detection (OSOD) remains challenging since objects of unknown categories are often misclassified to existing known classes. In this work, we propose to identify unknown objects by separating high/low-density regions in the latent space, based on the consensus that unknown objects are usually distributed in low-density latent regions. As traditional threshold-based methods only maintain limited low-density regions, which cannot cover all unknown objects, we present a novel Open-set Detector (OpenDet) with expanded low-density regions. To this aim, we equip OpenDet with two learners, Contrastive Feature Learner (CFL) and Unknown Probability Learner (UPL). CFL performs instance-level contrastive learning to encourage compact features of known classes, leaving more low-density regions for unknown classes; UPL optimizes unknown probability based on the uncertainty of predictions, which further divides more low-density regions around the cluster of known classes. Thus, unknown objects in low-density regions can be easily identified with the learned unknown probability. Extensive experiments demonstrate that our method can significantly improve the OSOD performance, e.g., OpenDet reduces the Absolute Open-Set Errors by 25%-35% on six OSOD benchmarks. Code is available at: <https://github.com/csuhan/opendet2>.

## 1. Introduction

Although the past decade has witnessed significant progress in object detection [3, 17, 31, 40, 42, 48], modern object detectors are often developed with a close-set as-

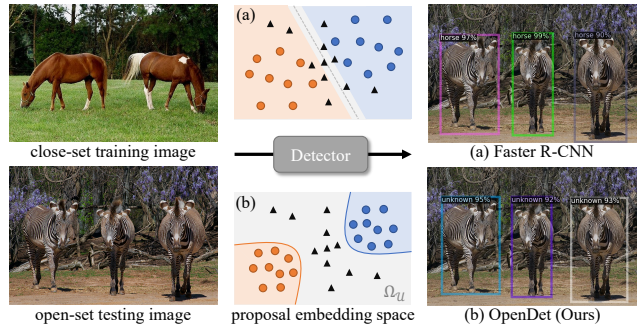


Figure 1. Trained on close-set images, (a) **threshold-based methods**, e.g., Faster R-CNN, usually misclassify unknown objects (black triangles, e.g. zebra) into known classes (colored dots, e.g. horse) due to limited low-density regions (in gray color). (b) **Our method** identifies unknown objects by expanding low-density regions. We encourage compact proposal features and learn clear separation between known and unknown classes.

sumption that the object categories appearing in the testing process are contained by the training sets, and quickly lose their efficiency when handling real-world scenarios as many objects categories have never been seen in the training. See Fig. 1 for an instance, where a representative object detector, i.e., Faster R-CNN [42] trained on PASCAL VOC [14], misclassifies zebra into horse with high confidence, as the new class zebra is not contained by PASCAL VOC. To alleviate this issue, Open-Set Object Detection (OSOD) has been recently investigated, where the detector trained on close-set datasets is asked to detect all known objects and identify unknown objects in open-set conditions.

OSOD can be seen as an extension of Open-Set Recognition (OSR) [43]. Although OSR has been extensively studied [2, 7, 16, 43, 55, 57], rare works attempted to solve the challenging OSOD. Dhamija *et al.* [12] first benchmarked the open-set performance of some representative methods [31, 40, 42], which indicates most detectors are overestimated in open-set conditions. Miller *et al.* [35, 36] adopt dropout sampling [15] to improve the robustness of

\* Work done during internship at Tencent YouTu Lab.

† Corresponding author.

detectors in open-set conditions. Joseph *et al.* [25] proposed an energy-based unknown identifier by fitting the energy distributions of known and unknown classes. In summary, prior works usually leverage hidden evidence (*e.g.*, the output logits) of pre-trained models as unknown indicators, with the cost of additional training step and complex post-processing. Can we train an open-set detector *with only* close-set data, and *directly* apply it to open-set environments *without* complex post-processing?

We draw inspiration from the consensus that known objects are usually clustered to form high-density regions in the latent space, while unknown objects (or novel patterns) are distributed in low-density regions [5, 18, 41]. From this perspective, proper separation of high/low-density latent regions is crucial for unknown identification. However, traditional methods, *e.g.*, hard-thresholding (Fig. 1 (a)), only maintain limited low-density regions, as higher thresholds will hinder the close-set accuracy. In this work, we propose to identify unknown objects by expanding low-density latent regions (Fig. 1 (b)). Firstly, we learn compact features of known classes, leaving more low-density regions for unknown classes. Then, we learn an unknown probability for each instance, which serves as a threshold to divide more low-density regions around the cluster of known classes. Finally, unknown objects distributed in these regions can be easily identified.

More specifically, we propose an Open-set Detector (OpenDet) with two learners, Contrastive Feature Learner (CFL) and Unknown Probability Learner (UPL), which expands low-density regions from two folds. Let us denote the latent space with  $\Omega = \Omega_K \cup \Omega_U$ , where  $\Omega_K$  and  $\Omega_U$  represent high/low-density sub-space, respectively. CFL performs instance-level contrastive learning to encourage intra-class compactness and inter-class separation of known classes, which expands  $\Omega_U$  by narrowing  $\Omega_K$ . UPL learns unknown probability for each instance based on the uncertainty of predictions. As we carefully optimize UPL to maintain the close-set accuracy, the learned unknown probability can serve as a threshold to divide more  $\Omega_U$  around  $\Omega_K$ . In the testing phase, we directly classify an instance into the unknown class if its unknown probability is the largest among all classes.

To demonstrate the effectiveness of our method, we take PASCAL VOC [14] for close-set training and construct several open-set settings considering both VOC and COCO [32]. Compared with previous methods, OpenDet shows significant improvements on all open-set metrics without compromising the close-set accuracy. For example, OpenDet reduces the Absolute Open-Set Errors (introduced in Sec. 4.1) by 25%-35% on six open-set settings. We also visualize the latent feature in Fig. 2, where OpenDet learns clear separation between known and unknown classes. Besides, we conduct extensive ablation experiments to analyze

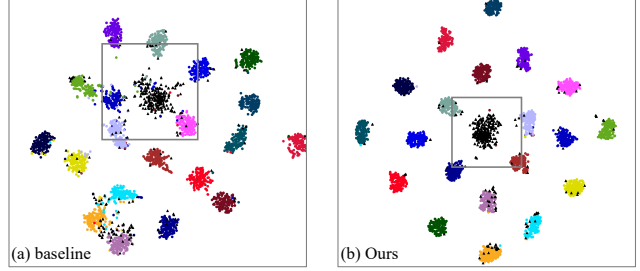


Figure 2. **t-SNE visualization of latent features.** We take VOC classes as known classes (colored dots), and non-VOC classes in COCO as unknown classes (black triangles). Our method learns a clear separation between known and unknown classes.

the effect of our main components and core design choices. Furthermore, we show that OpenDet can be easily extended to one-stage detectors and achieve satisfactory results. We summarize our contributions as:

- To our best knowledge, we are the first to solve the challenging OSOD by modeling low-density latent regions.
- We present a novel Open-set Detector (OpenDet) with two well-designed learners, CFL and UPL, which can be trained in an end-to-end manner and directly applied to open-set environments.
- We introduce a new OSOD benchmark. Compared with previous methods, OpenDet shows significant improvements on all open-set metrics, *e.g.*, OpenDet reduces the Absolute Open-Set Errors by 25%-35%.

## 2. Related Work

**Open-Set Recognition.** Early attempts on OSR [1, 24, 26, 44, 56] usually leverage traditional machine learning methods, *e.g.*, SVM [24, 44]. Bendale *et al.* [2] introduced OpenMax, the first deep learning-based OSR method, which redistributes the output probabilities of the softmax layer. Other approaches include generative adversarial network-based methods [16, 37] which generate potential open-set images to train an open-set classifier, reconstruction-based methods [38, 47, 55] which adopt auto-encoder to recover latent features and identify unknown by reconstruction errors, and prototype-based methods [6, 7] which identify open-set images by measuring the distance to learned prototypes. In addition, Zhou *et al.* [57] proposed to learn data placeholders to anticipate open-set data and classifier placeholders to distinguish known and unknown. Kong *et al.* [28] utilized an adversarially trained discriminator to detect unknown examples. Our method is more related to [57]. Differently, [57] requires close-set pre-train and calibration on validation sets, while our method is trained in an end-to-end manner, and the learned unknown probability is accurate and calibration-free.

**Open-Set Object Detection** is an extension of OSR in object detection. Dhamija *et al.* [12] first formalized OSOD

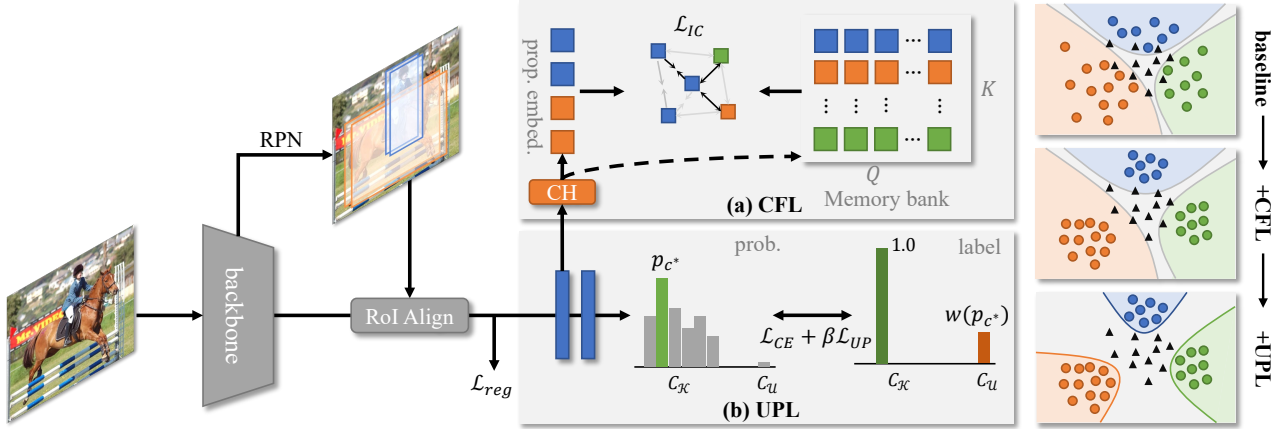


Figure 3. **Overview of our proposed method.** **Left:** OpenDet is a two-stage detector with (a) Contrastive Feature Learner (CFL) and (b) Unknown Probability Learner (UPL). **CFL** first encodes proposal features into low-dimensional embeddings with the Contrastive Head (CH). Then we optimize these embeddings between the mini-batch and memory bank with an Instance Contrastive Loss  $\mathcal{L}_{IC}$ . **UPL** learns probabilities for both known classes  $C_K$  and unknown class  $C_U$  with cross-entropy loss  $\mathcal{L}_{CE}$  and Unknown Probability Loss  $\mathcal{L}_{UP}$ . **Right:** A toy illustration of how different components work. Colored dots and triangles denote proposal features of different known and unknown classes, respectively. Our method identifies unknown by expanding low-density latent regions (in gray color).

and benchmarked some representative detectors by their classifiers. Classifiers with a background class [42] performs better than one-vs-rest [31] and objectness-based [40] classifiers in handling unknown objects. Dhamija *et al.* [12] also show that the performance of most detectors is overestimated in open-set conditions. Miller *et al.* [35, 36] utilized dropout sampling [15] to estimate uncertainty in object detection and thus reduce open-set errors. Joseph *et al.* [25] proposed an energy-based unknown identifier by fitting the energy distributions of known and unknown classes. However, the approach in [25] requires extra open-set data of unknown classes, which violates the original definition of OSOD. In summary, previous methods leverage hidden evidence (e.g., the output logits) of pre-trained models as unknown indicators. But they need additional training step and complex post-processing to estimate the unknown indicator. In contrast, OpenDet can be trained with only close-set data and directly identify unknown objects with the learned unknown probability.

**Contrastive Learning** is a methodology to learn representation by pulling together positive sample pairs while pushing apart negative sample pairs, which has been recently popularized for self-supervised representation learning [4, 8, 9, 13, 19, 20]. Khosla *et al.* [27] first extended self-supervised contrastive learning to the full-supervised setting and received a lot of attention from other fields, e.g., long-tailed recognition [10, 51], semantic segmentation [49, 52] and few-shot object detection [46]. Our approach is also inspired by supervised contrastive learning [27]. In this work, we explore *instance-level* contrastive learning to learn compact features of object proposals.

**Uncertainty Estimation.** Neural networks tend to produce over-confident predictions [29]. Estimating the uncertainty

of model predictions is important for real-world applications. Currently, uncertainty estimation can be categorized into sampling-based and sampling-free methods. Sampling-based methods ensemble predictions of multiple runs [15] or multiple models [29], which are not applicable for speed-critical object detection. Sampling-free methods learn additional confidence value [11, 45] to estimate uncertainty. Our method belongs to the latter family. The learned unknown probability can reflect the uncertainty of predictions.

### 3. Methodology

#### 3.1. Preliminary

We formalize OSOD based on prior works [12, 25]. Let us denote with  $D = \{(x, y), x \in X, y \in Y\}$  an object detection dataset, where  $x$  is an input image and  $y = \{(c_i, \mathbf{b}_i)\}_{i=1}^N$  denotes a set of objects with corresponding class label  $c$  and bounding box  $\mathbf{b}$ . We train the detector on the training set  $D_{tr}$  with  $K$  known classes  $C_K = \{1, \dots, K\}$ , and test it on the testing set  $D_{te}$  with objects from both known classes  $C_K$  and unknown classes  $C_U$ . The goal is to detect all known objects (objects  $\in C_K$ ), and identify unknown objects (objects  $\in C_U$ ) so that they will not be misclassified to  $C_K$ . As it is impossible to list infinite unknown classes, we denote them with  $C_U = K+1$ .

Different from OSR, OSOD has its unique challenges. In OSR, an image only belongs to  $C_K$  or  $C_U$ ; any example out of  $C_K$  is defined as unknown. In OSOD, an image may contain objects from both  $C_K$  and  $C_U$ , which is defined as mixed unknown [12]. That means unknown objects will also appear in  $D_{tr}$ , but have not been labeled yet. Besides, detectors usually keep a background class  $C_{bg}$  which is easily confused with  $C_U$ .

### 3.2. Baseline Setup

We setup the baseline with Faster R-CNN [42], which consists of a backbone, Region Proposal Network (RPN) and R-CNN. The standard R-CNN includes a shared fully connected (FC) layer and two separate FC layers for classification and regression. We augment R-CNN in three ways. **(a)** We replace the shared FC layer with two parallel FC layers so that the module applied to the classification branch will not affect the regression task. **(b)** Inspired by [7, 53], we use cosine similarity-based classifier to alleviate the over-confidence issue [2, 39]. Specifically, we adopt scaled cosine similarity scores as output logits:  $s_{i,j} = \frac{\alpha \mathcal{F}(x)_i^\top w_j}{\|\mathcal{F}(x)_i\| \|w_j\|}$ , where  $s_{i,j}$  denotes the similar score between  $i$ -th proposal features  $\mathcal{F}(x)_i$  and weight vector of class  $j$ .  $\alpha$  is the scaling factor ( $\alpha=20$  by default). **(c)** The box regressor is set to class-agnostic, *i.e.*, the regression branch outputs a vector of length 4 rather than  $4(K+2)$ . Note that our baseline does not improve the open-set performance, but it is effective for the whole framework (Fig. 3).

### 3.3. Contrastive Feature Learner

This section presents Contrastive Feature Learner (CFL) to encourage intra-class compactness and inter-class separation, which expands low-density latent regions by narrowing the cluster of known classes. As shown in Fig. 3 (a), CFL contains a contrastive head (CH), a memory bank, and an instance contrastive loss  $\mathcal{L}_{IC}$ . For a proposal feature  $\mathcal{F}(x)_i$ , we first encode it into a low-dimensional embedding with CH. Then, we optimize the embeddings from the mini-batch and memory bank with  $\mathcal{L}_{IC}$ . We give more details in the following part.

**Contrastive Head.** We build a contrastive head (CH) to map high-dimensional proposal feature  $\mathcal{F}(x)_i$  to low-dimensional proposal embedding  $\mathbf{z}_i \in \mathbf{R}^d$  ( $d = 128$  by default). In detail, CH is a multilayer perceptron with sequential FC, ReLU, FC, and L2-Norm layers, which is applied to the classification branch of R-CNN in training and abandoned during inference.

**Class-Balanced Memory Bank.** Popular contrastive representation learning usually adopts large-size mini-batch [27] or memory bank [20] to increase the diversity of exemplars. Here we build a novel class-balanced memory bank to increase the diversity of object proposals. Specifically, for each class  $c \in C_K$ , we initialize a memory bank  $M(c)$  of size  $Q$ . Then, we sample representative proposals from a mini-batch with two steps: **(a)** We sample proposals with Intersection of Union (IoU)  $> T_m$  where  $T_m$  is an IoU threshold to ensure the proposals contain relevant semantics. **(b)** For each mini-batch, we sample  $q$  ( $q \leq Q$ ) proposals that are least similar (*i.e.*, minimum cosine similarity) with existing exemplars in  $M(c)$ . This step makes our memory banks store more diverse exemplars and enable

long-term memory. Finally, we repeat (a) and (b) every iteration where the oldest proposals are out of the memory and the newest into the queue.

**Instance-Level Contrastive Learning.** Inspired by supervised contrastive loss [27], we propose an Instance Contrastive (IC) Loss to learn more compact features of object proposals. Assume we have a mini-batch of  $N$  proposals, IC Loss is formulated as:

$$\mathcal{L}_{IC} = \frac{1}{N} \sum_{i=1}^N \mathcal{L}_{IC}(\mathbf{z}_i), \quad (1)$$

$$\mathcal{L}_{IC}(\mathbf{z}_i) = \frac{1}{|M(\mathbf{c}_i)|} \sum_{\mathbf{z}_j \in M(\mathbf{c}_i)} \log \frac{\exp(\mathbf{z}_i \cdot \mathbf{z}_j / \tau)}{\sum_{\mathbf{z}_k \in A(\mathbf{c}_i)} \exp(\mathbf{z}_i \cdot \mathbf{z}_k / \tau)}, \quad (2)$$

where  $\mathbf{c}_i$  is the class label of  $i$ -the proposal,  $\tau$  is a temperature hyper-parameter,  $M(\mathbf{c}_i)$  denotes the memory bank of class  $\mathbf{c}_i$ , and  $A(\mathbf{c}_i) = M \setminus M(\mathbf{c}_i)$ . Note that we only optimize proposals with IoU  $> T_b$  where  $T_b$  is an IoU threshold similar to  $T_m$ .

Although unknown objects are unavailable in training, the separation of known classes benefits unknown identification. Optimizing  $\mathcal{L}_{IC}$  is equivalent to pushing the cluster of known classes away from low-density latent regions. As shown in Fig. 2 (b), our method learns a clear separation between known and unknown classes with only close-set training data.

### 3.4. Unknown Probability Learner

As introduced in Sec. 3.3, CFL expands low-density latent regions by narrowing the cluster of known classes (*i.e.*, high-density regions). However, we still lack explicit boundaries to separate high/low-density regions. Traditional threshold-based methods with a small score threshold (*e.g.*, 0.05) only maintain limited low-density regions, which cannot cover all unknown objects. Here we present Unknown Probability Learner (UPL) to divide more low-density latent regions around the cluster of known classes.

To this aim, we first augment the K-way classifier with the K+1-way classifier, where K+1 denotes the unknown class. Then the problem becomes: *how to optimize the unknown class with only close-set training data?* Let us consider a simple known vs. unknown classifier with available open-set data, we can directly train a good classifier by maximizing margins between classes. Now, we only have close-set data; To train such a classifier, we relax the maximum margin principle and only ensure all known objects are correctly classified, *i.e.*, maintaining the close-set accuracy. With this premise, we will introduce how to learn the unknown probability in the following section.

**Review Cross-Entropy (CE) Loss.** We first review softmax CE Loss, the default classification loss of Faster R-CNN. Let  $\mathbf{s}$  denote the classification logits of a proposal, the softmax probability  $\mathbf{p}$  of class  $c$  is defined as:



Method	VOC	VOC-COCO-20				VOC-COCO-40				VOC-COCO-60			
	mAP <sub>K</sub> ↑	WI↓	AOSE↓	mAP <sub>K</sub> ↑	AP <sub>U</sub> ↑	WI↓	AOSE↓	mAP <sub>K</sub> ↑	AP <sub>U</sub> ↑	WI↓	AOSE↓	mAP <sub>K</sub> ↑	AP <sub>U</sub> ↑
FR-CNN [42]	80.10	18.39	15118	58.45	0	22.74	23391	55.26	0	18.49	25472	55.83	0
FR-CNN* [42]	80.01	18.83	11941	57.91	0	23.24	18257	54.77	0	18.72	19566	55.34	0
PROSER [57]	79.68	19.16	13035	57.66	10.92	24.15	19831	54.66	7.62	19.64	21322	55.20	3.25
ORE [25]	79.80	18.18	12811	58.25	2.60	22.40	19752	55.30	1.70	18.35	21415	55.47	0.53
DS [36]	80.04	16.98	12868	58.35	5.13	20.86	19775	55.31	3.39	17.22	21921	55.77	1.25
OpenDet	80.02	<b>14.95</b>	<b>11286</b>	<b>58.75</b>	<b>14.93</b>	<b>18.23</b>	<b>16800</b>	<b>55.83</b>	<b>10.58</b>	<b>14.24</b>	<b>18250</b>	<b>56.37</b>	<b>4.36</b>

Table 1. **Comparisons with other methods on VOC and VOC-COCO-T<sub>1</sub>.** We report close-set performance (mAP<sub>K</sub>) on VOC, and both close-set (mAP<sub>K</sub>) and open-set (WI, AOSE, AP<sub>U</sub>) performance of different methods on VOC-COCO-{20, 40, 60}. \* means a higher score threshold (*i.e.* 0.1) for testing.

Method	VOC-COCO-0.5n				VOC-COCO-n				VOC-COCO-4n			
	WI↓	AOSE↓	mAP <sub>K</sub> ↑	AP <sub>U</sub> ↑	WI↓	AOSE↓	mAP <sub>K</sub> ↑	AP <sub>U</sub> ↑	WI↓	AOSE↓	mAP <sub>K</sub> ↑	AP <sub>U</sub> ↑
FR-CNN [42]	9.25	6015	77.97	0	16.14	12409	74.52	0	32.89	48618	63.92	0
FR-CNN* [42]	9.01	4599	77.66	0	16.00	9477	74.17	0	33.11	37012	63.80	0
PROSER [57]	9.32	5105	77.35	7.48	16.65	10601	73.55	8.88	34.60	41569	63.09	11.15
ORE [25]	8.39	4945	77.84	1.75	15.36	10568	74.34	1.81	32.40	40865	64.59	2.14
DS [36]	8.30	4862	77.78	2.89	15.43	10136	73.67	4.11	31.79	39388	63.12	5.64
OpenDet	<b>6.44</b>	<b>3944</b>	<b>78.61</b>	<b>9.05</b>	<b>11.70</b>	<b>8282</b>	<b>75.56</b>	<b>12.30</b>	<b>26.69</b>	<b>32419</b>	<b>65.55</b>	<b>16.76</b>

Table 2. **Comparisons with other methods on VOC-COCO-T<sub>2</sub>.** Note that we put VOC-COCO-2n in the appendix due to limited space.

$$p_c = \text{softmax}(s_c) = \frac{\exp(s_c)}{\sum_{j \in C} \exp(s_j)}, \quad (3)$$

where  $C = C_{K \cup U \cup bg}$  denotes all known classes  $C_K$ , unknown class  $C_U$  and background  $C_{bg}$ . Then, we formulate softmax CE Loss  $\mathcal{L}_{CE}$  as:

$$\mathcal{L}_{CE} = - \sum_{c \in C} y_c \log(p_c), \quad y_c = \begin{cases} 1, c = c^* \\ 0, c \neq c^* \end{cases}, \quad (4)$$

where  $c^*$  means the ground truth class, and  $y$  is the one-hot class label. For simplicity, we re-write  $\mathcal{L}_{CE}$  as:

$$\mathcal{L}_{CE} = -\log(p_{c^*}). \quad (5)$$

**Learning Unknown Probability.** Since there is no supervision for the unknown probability  $p_u$ , we consider a conditional probability  $p'_u$  under the ground truth probability  $p_{c^*}$ . Formally, we define  $p'_u$  as a softmax probability without the logit of ground truth class  $c^*$ :

$$p'_u = \frac{\exp(s_u)}{\sum_{j \in C, j \neq c^*} \exp(s_j)}, \quad (6)$$

where  $u$  is short for the unknown class  $C_U$ . Then, similar to CE Loss, we formulate an Unknown Probability (UP) Loss  $\mathcal{L}_{UP}$  to optimize  $p'_u$ , which is defined as:

$$\mathcal{L}_{UP} = -\log(p'_u). \quad (7)$$

After that, we jointly optimize the CE Loss  $\mathcal{L}_{CE}$  and UP Loss  $\mathcal{L}_{UP}$  (illustrated in Fig. 3 (b)), where  $\mathcal{L}_{CE}$  aims to

maintain the close-set accuracy, and  $\mathcal{L}_{UP}$  learns the unknown probability. Take Fig. 3 (bottom-right) for an illustration, optimizing  $\mathcal{L}_{UP}$  is equivalent to dividing more low-density latent regions (in gray color) from known classes. Once we finished the training, the learned unknown probability serves as an indicator to identify unknown objects in these low-density regions.

**Uncertainty-weighted Optimization.** Although we optimize the conditional probability  $p'_u$  instead of  $p_u$ ,  $\mathcal{L}_{UP}$  will still penalize the convergence of  $\mathcal{L}_{CE}$ , leading to the accuracy drop of known classes. Inspired by uncertainty estimation [11, 45], we add a weighting factor  $w(\cdot)$  to  $\mathcal{L}_{UP}$ , which is defined as a function of  $p_{c^*}$ :

$$w(p_{c^*}) = (1 - p_{c^*})^\alpha p_{c^*}, \quad (8)$$

where  $\alpha$  is a hyper-parameter ( $\alpha=1$  by default). Despite many design choices of  $w(\cdot)$  (shown in Tab. 6), we choose a simple yet effective one in Eq. 8. We are inspired by the popular uncertainty signal: entropy  $w(\mathbf{p}) = -\mathbf{p} \log(\mathbf{p})$ . Since Eq. 8 has a similar curve shape to entropy (see our appendix), it can also reflect uncertainty. But our empirical findings suggest that Eq. 8 is easier to optimize than entropy. Finally, we formulate the uncertainty-weighted UP Loss as follow:

$$\mathcal{L}_{UP} = -w(p_{c^*}) \log(p'_u). \quad (9)$$

**Hard Example Mining.** It is unreasonable to let all known objects learn the unknown probability as they do not belong

to the unknown class. Therefore, we present uncertainty-guided hard example mining to optimize  $\mathcal{L}_{UP}$  with high-uncertainty proposals, which may overlap with real unknown objects in the latent space. Here we consider two uncertainty-guided mining methods:

- *Max entropy.* Entropy is a popular uncertainty measure [29, 34] defined as:  $H(\mathbf{p}) = -\sum_{c \in C} p_c \log(p_c)$ . For a mini-batch, We sort them in descending entropy order, and select *top-k* examples.
- *Min max-probability.* Max-probability, *i.e.*, the maximum probability of all classes:  $\max(\mathbf{p})$ , is another uncertainty signal. We select *top-k* examples with minimum max-probability.

Furthermore, since background proposals usually overwhelm the mini-batch, we sample the same number of foreground and background proposals, enabling our model to recall unknown objects from the background class.

### 3.5. Overall Optimization

Our method can be trained in an end-to-end manner with the following multi-task loss:

$$\mathcal{L} = \mathcal{L}_{rpn} + \mathcal{L}_{reg} + \mathcal{L}_{CE} + \beta \mathcal{L}_{UP} + \gamma_t \mathcal{L}_{IC}, \quad (10)$$

where  $\mathcal{L}_{rpn}$  denotes the total loss of RPN,  $\mathcal{L}_{reg}$  is smooth L1 loss for box regression,  $\beta$  and  $\gamma_t$  are weighting coefficients. Note  $\gamma_t$  is proportional to the current iteration  $t$  so that we can gradually decrease the weight of  $\mathcal{L}_{IC}$  for better convergence of  $\mathcal{L}_{CE}$  and  $\mathcal{L}_{UP}$ .

## 4. Experiment

### 4.1. Experimental Setup

**Datasets.** We construct an OSOD benchmark using popular PASCAL VOC [14] and MS COCO [32]. We take the `trainval` set of VOC for close-set training. Meanwhile, we take 20 VOC classes and 60 non-VOC classes in COCO to evaluate our method under different open-set conditions. Here we define two settings: **VOC-COCO- $\{\mathbf{T}_1, \mathbf{T}_2\}$** . For setting  $\mathbf{T}_1$ , we gradually increase open-set classes to build three joint datasets with  $n=5000$  VOC testing images and  $\{n, 2n, 3n\}$  COCO images **containing**  $\{20, 40, 60\}$  non-VOC classes, respectively. For setting  $\mathbf{T}_2$ , we gradually increase the Wilderness Ratio (WR)<sup>2</sup> [12] to construct four joint datasets with  $n$  VOC testing images and  $\{0.5n, n, 2n, 4n\}$  COCO images **disjointing** with VOC classes. See our appendix for more details.

**Evaluation Metrics.** We use the **Wilderness Impact (WI)** [12] to measure the degree of unknown objects misclassified to known classes:  $WI = (\frac{P_K}{P_{K \cup U}} - 1) \times 100$ , where  $P_K$  and  $P_{K \cup U}$  denote the precision of close-set and open-set

<sup>2</sup>Wilderness Ratio is the ratio of #images with unknown objects to #images with known objects.

CFL	UPL	WI $\downarrow$	AOSE $\downarrow$	mAP $\uparrow$	AP $\uparrow$
baseline		19.26	16433	58.33	0
✓		17.92	15162	58.54	0
	✓	16.47	12018	57.91	14.27
✓	✓	<b>14.95</b>	<b>11286</b>	<b>58.75</b>	<b>14.93</b>

Table 3. **Effect of different components** on VOC-COCO-20.

classes, respectively. Note that we scale the original WI by 100 for convenience. Following [25], we report WI under a recall level of 0.8. Besides, we also use **Absolute Open-Set Error (AOSE)** [36] to count the number of misclassified unknown objects. Furthermore, we report the **mean Average Precision (mAP)** of known classes (mAP $_K$ ). Lastly, we measure the novelty discovery ability by AP $_U$  (AP of the unknown class). Note WI, AOSE, and AP $_U$  are open-set metrics, and mAP $_K$  is a close-set metric.

**Comparison Methods.** We compare OpenDet with the following methods: Faster R-CNN (**FR-CNN**) [42], Dropout Sampling (**DS**) [36], **ORE** [25] and **PROSER** [57]. FR-CNN is the base detector of other methods. We also report FR-CNN\*, which adopts a higher score threshold for testing. We use the official code of ORE and reimplement DS and PROSER based on the FR-CNN framework.

**Implementation Details.** We use ResNet-50 [21] with Feature Pyramid Network [30] as the backbone of all methods. We adopt the same learning rate schedules with Detectron2 [54]. SGD optimizer is adopted with an initial learning rate of 0.02, momentum of 0.9, and weight decay of 0.0001. All models are trained on 8 GPUs with a batch size of 16. For **CFL**, we set memory size  $Q=256$  and sampling size  $q=16$ . We sample proposals with an IoU threshold  $T_m=0.7$  for the memory bank, and  $T_b=0.5$  for the mini-batch. For **UPL**, we sample  $k=3$  examples for foreground and background proposals respectively. Besides, we set hyper-parameters  $\alpha=1.0$  and  $\beta=0.5$ . We set the initial value of  $\gamma_t=0.1$  and linearly decrease it to zero.

### 4.2. Main Results

We compare OpenDet with other methods on VOC-COCO- $\{\mathbf{T}_1, \mathbf{T}_2\}$ . Tab. 1 shows results on VOC-COCO- $\mathbf{T}_1$  by gradually increasing unknown classes. Compared with FR-CNN, FR-CNN\* with a higher score threshold (0.05→0.1) does not reduce WI, but results in a decrease in mAP $_K$ , where known objects with low confidence are filtered out. PROSER improves AOSE and AP $_U$  to some extent, but the WI and mAP $_K$  are even worse. Although ORE and DS achieve comparable mAP $_K$ , the improvement on open-set metrics is limited. The proposed OpenDet outperforms other methods by a large margin. Taking VOC-COCO-20 for an example, OpenDet gains about 20%, 25%, 14.93 on WI, AOSE and AP $_U$  respectively without compromising the mAP $_K$  (58.75 vs. 58.45). Besides, we also report mAP $_K$  on VOC, which indicates OpenDet is competitive in the traditional close-set setting (80.02 vs. 80.10).

Memory	size	WI $\downarrow$	mAP $\uparrow$
single-GPU mini-batch	~50	16.19	58.29
cross-GPU mini-batch	~50 $\times$ 8	15.88	58.07
class-agnostic memory bank	5120	15.99	57.47
class-agnostic memory bank*	65536	15.49	<b>58.90</b>
class-balanced memory bank	256 $\times$ 20	<b>14.95</b>	58.75

Table 4. **Class-balanced memory bank.** We compare our class-balanced memory bank with other variants. We keep class-agnostic memory bank the same size with ours (256 $\times$ 20=5120). 8 and 20 are the number of GPU and VOC classes, respectively. \* means a larger memory size.

	(a)	(b)	(c)	(d)	(e)	(f)
$T_b$	0.5	0.7	0.9	0.5	0.5	0.7
$T_m$	0.5	0.7	0.9	0.7	0.9	0.9
WI $\downarrow$	15.33	15.16	15.27	14.95	<b>14.62</b>	15.27
mAP $\uparrow$	58.29	58.55	58.32	<b>58.75</b>	58.66	58.33

(a) IoU threshold

	(a)	(b)	(c)	(d)	(e)	(f)
q	16	16	16	32	64	128
Q	128	256	512	256	256	256
WI $\downarrow$	15.36	14.95	<b>14.47</b>	15.24	15.43	14.77
mAP $\uparrow$	58.51	<b>58.75</b>	58.31	58.32	57.77	58.18

(b) Memory size and mini-batch sampling size

Table 5. **Sampling strategy in CFL.** We list different choices of (a) memory sampling threshold  $T_m$  and mini-batch sampling threshold  $T_b$ , (b) memory size  $Q$  and sampling size  $q$ .

We also compare OpenDet with other methods by increasing the WR, where the results in Tab. 2 draws similar conclusions with Tab. 1. Our method performs better as the WR increases. For example, the mAP $\uparrow$  gains on VOC-COCO- $\{0.5n, n, 4n\}$  are  $\{0.64, 1.04, 1.63\}$ , indicating that our method actually separates known and unknown classes.

### 4.3. Ablation Studies

In this section, we conduct ablation experiments on VOC-COCO-20 to analyze the effect of our main components and core design choices.

**Overall Analysis.** We first analyze the contribution of different components. As shown in Tab. 3, our two modules, CFL and UPL, show substantial improvement compared with the baseline. The combination of CFL and UPL further boosts the performance. We also visualize the latent features in Fig. 2, where our method learns clear separation between known and unknown classes.

**Contrastive Feature Learner.** We carefully study the design choices of the memory bank and example sampling strategy in CFL. As  $\mathcal{L}_{IC}$  is optimized between the current mini-batch and the memory bank, we investigate different **designs of memory** in Tab. 4. Compared with the mini-batch (*i.e.*, short-term memory), the settings with a memory bank perform better on WI. However, imbalanced training data makes the class-agnostic memory bank filled with high-frequency classes, leading to a drop in

	$w(\cdot)$	WI $\downarrow$	AOSE $\downarrow$	mAP $\uparrow$	AP $\uparrow$
	baseline	19.26	16433	58.33	0
(a)	identity	<b>10.50</b>	12185	56.42	11.33
(b)	$-p_{c^*} \log(p_{c^*})$	14.70	11384	58.13	13.71
(c)	$(1 - p_{c^*})^\alpha p_{c^*}$	14.95	<b>11286</b>	<b>58.75</b>	14.93
(d)	$(1 - p_m)^\alpha p_m$	14.86	11296	58.03	14.15
(e)	$H(\mathbf{p}) / \log(C)$	14.29	11690	57.75	14.65

Table 6. **Different designs of  $w(\cdot)$  in  $\mathcal{L}_{UP}$ .**  $p_m$  is the maximum probability of all classes:  $p_m = \max(\mathbf{p})$ . (e) denotes normalized entropy where  $H(\mathbf{p}) = -\sum_c p_c \log(p_c)$  and  $C$  is the number of known classes.

Setting	WI $\downarrow$	AOSE $\downarrow$	mAP $\uparrow$	AP $\uparrow$
<b>OpenDet</b> (w/ HEM)	14.95	<b>11286</b>	<b>58.75</b>	<b>14.93</b>
(a) w/o HEM	18.33	13733	57.41	13.91
(b) w/o bg.	<b>13.02</b>	12230	56.53	13.49
(c) top-k:				
1	<b>14.46</b>	12826	58.42	14.54
3	14.95	11286	<b>58.75</b>	<b>14.93</b>
5	14.66	10412	58.50	14.55
10	15.15	<b>10358</b>	58.25	14.86
all	18.40	11779	56.55	13.89
(d) metric:				
random	17.01	13065	56.99	<b>15.58</b>
max entropy	<b>14.29</b>	11514	58.27	15.46
min max-probability	14.95	<b>11286</b>	<b>58.75</b>	14.93

Table 7. **Hard example mining (HEM) in UPL.** (a) without HEM. (b) without background: we only sample foreground proposals. (c) varying *top-k*. the setting *all* means all foreground and equal number of background proposals. (d) mining methods.

mAP $\uparrow$  (58.76 $\rightarrow$ 57.47). Enlarging the memory bank size (5120 $\rightarrow$ 65536) can alleviate this issue, but it requires more computation. The proposed class-balanced memory bank can store more diverse examples with a small memory size, outperforming other variants.

We further study the design choices of **example sampling strategies**. For a mini-batch, we consider the IoU threshold  $T_b$ ; for the memory bank, we consider the IoU threshold  $T_m$ , memory size  $Q$  and mini-batch sampling size  $q$ . As shown in Tab. 5a, the settings (d) and (e) achieve the best result in mAP $\uparrow$  and WI, respectively, while (a)-(c) are worse than (d)-(e) in WI. This indicates that the mini-batch requires a loose constraint to gather more diverse examples, while the memory bank needs high-quality examples to represent the class centers. In Tab. 5b, (b) and (c) perform better than other settings, which demonstrates that long-term memory (*i.e.*, larger  $Q/q$ ) is a good choice for CFL.

**Unknown Probability Learner.** We first explore **different variants of  $w(\cdot)$** . Compared with the baseline, Tab. 6 (a) significantly reduces WI and AOSE, but leads to mAP $\uparrow$  drop, which indicates the learned unknown probability is overestimated. The formula of (b) is similar to entropy, and (c) is our default setting. As discussed in Sec. 3.4, both (b) and (c) achieve satisfactory results in WI and AOSE, but (c) is outperforms (b) in mAP $\uparrow$  and



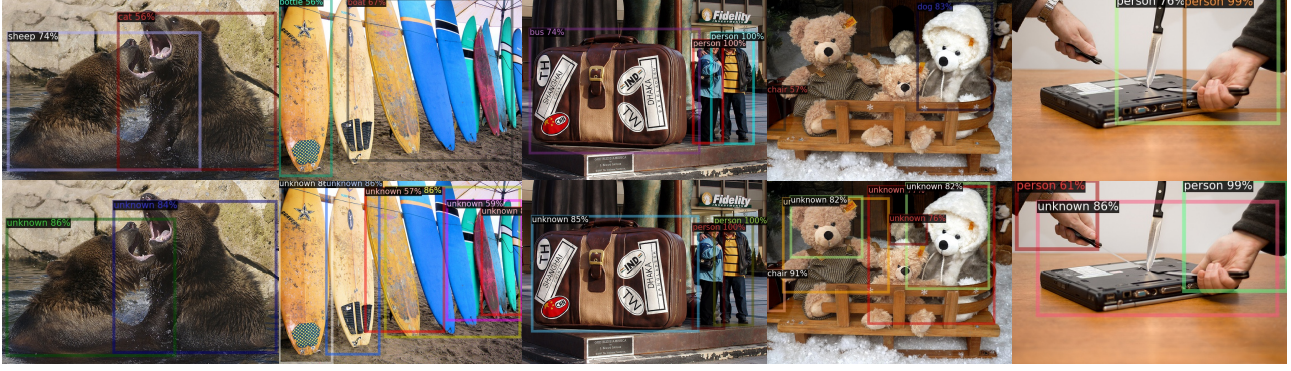


Figure 4. **Qualitative comparisons** between the baseline (top) and OpenDet (bottom). We train both models on VOC and visualize the detection results on COCO. Note that we apply NMS between known classes and the unknown class for better visualization.

$AP_U$ . (d) and (e) are two variants of  $w(\cdot)$  based on max-probability and entropy, respectively. They obtain comparable performance on open-set metrics, but the  $mAP_K$  is lower than (c).

We also analyze the effect of **Hard example mining (HEM)** in Tab. 7. Comparing Tab. 7 (a) (without HEM) with our default setting (with HEM), we show HEM is crucial for UPL. Tab. 7 (b) indicates background proposals are also necessary for unknown probability learning, *e.g.*, OpenDet without background leads to 2.22 and 1.44 drop in  $mAP_K$  and  $AP_U$ . Besides, we varying the hyper-parameter *top-k* in Tab. 7 (c) where HEM works in a wide range of  $k$  ( $\leq 10$ ), while optimizing all examples is not applicable. Tab. 7 (d) demonstrates the effectiveness of two mining methods, *i.e.*, max entropy and min max-probability. **Qualitative Comparisons.** Fig. 4 compares the qualitative results of baseline and OpenDet. OpenDet gives an unknown label to unknown objects (bottom row), while the baseline method classifies them to known classes or the background (top row). See our appendix for more qualitative results.

#### 4.4. Extend to One-Stage Detector

Although OpenDet is based on a two-stage detector, it can be easily extended to other architectures, *e.g.*, a representative one-stage detector RetinaNet [31]. RetinaNet has a backbone and two parallel sub-networks for classification and regression, respectively. Different from FR-CNN, RetinaNet adopts Focal Loss [31] for dense classification. Here we show how to extend OpenDet to RetinaNet (denote with Open-RetinaNet). For CFL, we append the contrastive head to the second-last layer of the classification sub-network. We adopt the same sampling strategies in CFL and optimize  $\mathcal{L}_{IC}$  with pixel-wise features. For UPL, we only sample hard foreground examples as RetinaNet does not preserve a background class. Then  $\mathcal{L}_{UP}$  is jointly optimized with Focal Loss. Tab. 8 reports the results on VOC-COCO-20, where Open-RetinaNet shows significant improvements

Method	WI $\downarrow$	AOSE $\downarrow$	$mAP_K\uparrow$	$AP_U\uparrow$
RetinaNet	14.58	38071	57.44	0
Open-RetinaNet	<b>10.84</b>	<b>16815</b>	57.25	<b>11.02</b>

Table 8. **Performance of Open-RetinaNet** on VOC-COCO-20.

on all open-set metrics and achieves comparable close-set  $mAP_K$ . For example, Open-RetinaNet gains 23.7%, 55.8%, and 11.02 in WI, AOSE, and  $AP_U$ , respectively.

## 5. Conclusions

This paper proposes a novel Open-set Detector (OpenDet) to solve the challenging OSOD task by expanding low-density latent regions. OpenDet consists of two well-designed learners, CFL and UPL, where CFL performs *instance-level* contrastive learning to learn more compact features and UPL learns the unknown probability that serves as a threshold to further separate known and unknown classes. We also build an OSOD benchmark and conduct extensive experiments to demonstrate the effectiveness of our method. Compared with other methods, OpenDet shows significant improvements on all metrics.

**Limitations.** We notice that some low-quality proposals belonging to known classes are given the unknown label during inference, and cannot be filtered out by per-class non-maximum suppression. Although these proposals do not hurt the close-set  $mAP_K$ , it raises a new question about reducing false unknown predictions, which is also a direction for our future work.

## Acknowledgement

This work was supported by National Nature Science Foundation of China under grant 61922065, 41820104006 and 61871299. The numerical calculations in this paper have been done on the supercomputing system in the Supercomputing Center of Wuhan University. Jian Ding is also supported by China Scholarship Council.



## A. More Experimental Details

### A.1. Datasets

In this section, we introduce more details about the dataset construction.

**PASCAL VOC [14].** We use VOC07 `train` and VOC12 `trainval` splits for the training, and VOC07 `test` split to evaluate the close-set performance. We take VOC07 `val` as the validation set.

**VOC-COCO-T<sub>1</sub>.** We divide 80 COCO classes into four groups (20 classes per group) by their semantics [25]: (1) VOC classes. (2) Outdoor, Accessories, Appliance, Truck. (3) Sports, Food. (4) Electronic, Indoor, Kitchen, Furniture. We construct VOC-COCO- $\{20, 40, 60\}$  with  $n=5000$  VOC testing images and  $\{n, 2n, 3n\}$  COCO images **containing**  $\{20, 40, 60\}$  non-VOC classes with semantic shifts, respectively. Note that we only ensure each COCO image contains objects of corresponding open-set classes, which means objects of VOC classes will also appear in these images. This setting is more similar to real-world scenarios where detectors need to carefully identify unknown objects and do not classify known objects into the unknown class.

**VOC-COCO-T<sub>2</sub>.** We gradually increase the Wilderness Ratio to build four dataset with  $n=5000$  VOC testing images and  $\{0.5n, n, 2n, 4n\}$  COCO images **disjointing** with VOC classes. Compared with the setting T<sub>1</sub>, T<sub>2</sub> aims to evaluate the model under a higher wilderness, where large amounts of testing instances are not seen in the training.

**Comparisons with existing benchmarks.** [12] proposed the first OSOD benchmark. They also use the data in VOC for close-set training, and both VOC and COCO for open-set testing. In the testing phase, they just vary the number of open-set images sampled from COCO, while ignoring the number of open-set categories. [25] proposed an open world object detection benchmark. They divide the open-set testing set into several groups by category. However, the wilderness ratio of each group is limited, and such data partitioning cannot reflect the real performance of detectors under extreme open-set conditions. In contrast, our proposed benchmark considers both the number of open-set classes (VOC-COCO-T<sub>1</sub>) and images (VOC-COCO-T<sub>2</sub>).

On the other hand, some works on open-set panoptic segmentation [23] divide a single dataset into close-set and open-set. If a image contains both close-set and open-set instances, they just remove the annotations of open-set instances. Differently, we strictly follows the definition in OSR [43] that unknown instances should not appear in training. To acquire enough open-set examples, we take both VOC and COCO from cross-dataset evaluation, which is a common practice in OSR [28, 47, 57].

### A.2. Implementation Details

**Training schedule.** Inspired by [50] that a good close-set classifier benefits OSR, we train all models with the  $3\times$  schedule (*i.e.*, 36 epochs). Besides, we enable UPL after several warmup iterations (*e.g.*, 100 iterations) to make sure the model produce valid probabilities.

**Open-RetinaNet.** We change some hyper-parameters for Open-RetinaNet. In OpenDet, we take object proposals as examples and apply CFL to proposal-wise embeddings, which are equivalent to the anchor boxes in RetinaNet. Therefore, we optimize Instance Contrastive Loss  $\mathcal{L}_{IC}$  with pixel-wise features of each anchor box. Since the number of anchor box is much larger than the proposals in OpenDet, we enlarge the memory size  $Q=1024$ , sampling size  $q=64$ , and loss weight to 0.2 in CFL. Similar, we sample 10 hard examples rather than 3 in UPL.

### A.3. Evaluation Metrics

Firstly, we give a detailed formulation of the Wilderness Impact [12], which is defined as:

$$\begin{aligned} WI &= \frac{P_K}{P_{K \cup U}} - 1 \\ &= \frac{TP_K}{TP_K + FP_K} / \frac{TP_K}{TP_K + FP_K + FP_U} - 1 \quad (11) \\ &= \frac{FP_U}{TP_K + FP_K}, \end{aligned}$$

where  $FP_U$  means that any detections belonging to the unknown classes  $C_U$  are classified to one of known classes  $C_K$ . For  $AP_U$  (AP of unknown classes), we merge the annotations of all unknown classes into one class, and calculate the *class-agnostic* AP between unknown’s predictions and the ground truth.

## B. Additional Main Results

Due to limited space in our main paper, we report the results on VOC-COCO- $2n$  in Tab. A1, where OpenDet shows significant improves than other methods.

Method	WI <sub>↓</sub>	AOSE <sub>↓</sub>	mAP <sub>K↑</sub>	AP <sub>U↑</sub>
FR-CNN [42]	24.18	24636	70.07	0
FR-CNN* [42]	24.05	18740	69.81	0
PROSER [57]	25.74	21107	69.32	10.31
ORE [25]	23.67	20839	70.01	2.13
DS [36]	23.21	20018	69.33	4.84
OpenDet	<b>18.69</b>	<b>16329</b>	<b>71.44</b>	<b>14.96</b>

Table A1. **Comparisons with other methods on VOC-COCO- $2n$ .** This table is an extension of Tab.2 in our main paper.

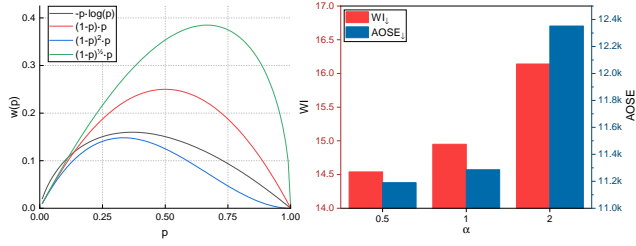


Figure A1. Visualization of different  $w(\cdot)$ .

metric	baseline	+CFL	+UPL	Ours
intra-variance	3.79	2.83	3.05	<b>2.47</b>
inter-distance	62.74	65.17	64.69	<b>66.31</b>

Table A2. Quantitative analyses of the latent space. We calculate the intra-class variance and inter-class distance of latent features.

$\gamma_t$	0.01	<b>0.1</b>	0.5	1.0	w/o decay
WI $_{\downarrow}$	16.13	14.95	12.26	9.71	15.65
mAP $_{\mathcal{K}\uparrow}$	58.90	58.75	57.47	53.36	58.43

Table A3. Loss weight of  $\mathcal{L}_{IC}$ . w/o decay:  $\gamma_t$  is a constant (*i.e.*, 0.1) instead of variable.

$\tau$	0.07 [20]	<b>0.1 [27]</b>	0.2
WI $_{\downarrow}$	15.48	14.95	15.50
mAP $_{\mathcal{K}\uparrow}$	57.80	58.75	58.87

Table A4. Temperature  $\tau$  in  $\mathcal{L}_{IC}$ .

## C. Additional Ablation Studies

**Visual analyses of  $w(\cdot)$ .** In Fig. A1, we plot the graph of different  $w(\cdot)$ . Compared with entropy:  $-p \log(p)$ , the proposed function  $(1-p)^\alpha \cdot p$  can adjust the curve shape by changing  $\alpha$ . In other words, the model adjusts the weights of examples as  $\alpha$  changes. The right of Fig. A1 reports the model’s open-set performance by varying  $\alpha$ , where smaller  $\alpha$  reduces WI and AOSE.

**Quantitative analyses of latent space.** In Fig. 2 of the main paper, we give a visual analyses of latent space. Here we give a quantitative analyses of latent space in Tab. A2. Specifically, we calculate the intra-class variance and inter-class distance of latent features. Tab. A2 shows that CFL and UPL, as well as their combination reduce intra-class variance and enlarge inter-class distance. The results further confirm our conclusion in the main paper that our method can expand low-density latent regions.

**More hyper-parameters in CFL. Loss weight:** Tab. A3 shows that loss weight is important for  $\mathcal{L}_{IC}$ , where a small weight (*e.g.*, 0.01) cannot learn compact features and a large weight (*e.g.*, 1.0) hinder the generalization ability. Besides, Tab. A3 (last column) also demonstrates the effectiveness of loss decay. **Temperature:** We try different  $\tau$  that used in pervious works [20, 27]. Tab. A4 indicates that  $\tau=0.1$  [27] works better than other settings.

setting	backbone	epoch	WI $_{\downarrow}$	mAP $_{\mathcal{K}\uparrow}$
end-to-end	-	-	<b>14.95</b>	<b>58.75</b>
fine-tune	fixed	1	17.98	56.88
	fixed	12	17.43	56.86
	trainable	12	17.01	57.19

Table A5. End-to-end vs. fine-tune in UPL. **End-to-end:** we jointly optimize UPL and other modules in OpenDet. **Fine-tune:** we pretrain a model without UPL, and optimize UPL in the fine-tuning stage.

Method	WI $_{\downarrow}$	AOSE $_{\downarrow}$	mAP $_{\mathcal{K}\uparrow}$	AP $_{\mathcal{U}\uparrow}$
<b>VOC:</b>				
RetinaNet	-	-	<b>79.84</b>	-
Open-RetinaNet	-	-	79.72	-
<b>VOC-COCO-40:</b>				
RetinaNet	17.60	58383	<b>53.81</b>	0
Open-RetinaNet	<b>13.65</b>	<b>25964</b>	53.22	<b>8.23</b>
<b>VOC-COCO-60:</b>				
RetinaNet	14.20	64327	<b>54.68</b>	0
Open-RetinaNet	<b>11.28</b>	<b>30631</b>	54.25	<b>3.20</b>

Table A6. Open-RetinaNet on more datasets.

Method	backbone	WI $_{\downarrow}$	AOSE $_{\downarrow}$	mAP $_{\mathcal{K}\uparrow}$	AP $_{\mathcal{U}\uparrow}$
FR-CNN	ResNet-50	18.39	15118	58.45	0
	<b>Swin-T</b>	<b>15.99</b>	<b>13204</b>	<b>63.09</b>	0
OpenDet	ResNet-50	14.95	11286	58.75	14.93
	<b>Swin-T</b>	<b>12.51</b>	<b>9875</b>	<b>63.17</b>	<b>15.77</b>

Table A7. Comparisons of different backbones, *i.e.*, ResNet-50 [21] and Swin-T [33].

**Training strategy.** Some works in OSR [57] adopted a pretrain-then-finetune paradigm to train the unknown identifier. We carefully design the UPL so that OpenDet can be trained in an end-to-end manner. Tab. A5 shows that jointly optimizing UPL performs better than that of fine-tuning.

**Open-RetinaNet.** To further demonstrates the effectiveness of Open-RetinaNet, we report more results in Tab. A6, where Open-RetinaNet shows substantial improvements on WI, AOSE and AP $_{\mathcal{U}}$ , and achieves comparable performance on mAP $_{\mathcal{K}}$ .

**Vision transformer as backbone.** We find the detector with vision transformer, *e.g.*, Swin Transformer [33] is a stronger baseline for OSOD. As shown in Tab. A7, models with a Swin-T backbone significantly suppress their ResNet counterparts.

**Speed and computation.** In the training stage, OpenDet only increases 14% (1.4h vs. 1.2h) training time and 1.2% (2424Mb vs. 2395Mb) memory usage. In the testing phase, as we only add the unknown class to the classifier, OpenDet keeps similar running speed and computation with FR-CNN.







Figure A3. More qualitative comparisons between the baseline and OpenDet.



## References

- [1] Abhijit Bendale and Terrance Boulton. Towards open world recognition. In *CVPR*, pages 1893–1902, 2015. 2
- [2] Abhijit Bendale and Terrance E Boulton. Towards open set deep networks. In *CVPR*, pages 1563–1572, 2016. 1, 2, 4
- [3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020. 1
- [4] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint arXiv:2006.09882*, 2020. 3
- [5] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning. 2006. *Cambridge, Massachusetts: The MIT Press View Article*, 2006. 2
- [6] Guangyao Chen, Peixi Peng, Xiangqian Wang, and Yonghong Tian. Adversarial reciprocal points learning for open set recognition. *arXiv preprint arXiv:2103.00953*, 2021. 2
- [7] Guangyao Chen, Limeng Qiao, Yemin Shi, Peixi Peng, Jia Li, Tiejun Huang, Shiliang Pu, and Yonghong Tian. Learning open set network with discriminative reciprocal points. In *ECCV*, pages 507–522. Springer, 2020. 1, 2, 4
- [8] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607, 2020. 3
- [9] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, pages 15750–15758, 2021. 3
- [10] Jiequan Cui, Zhisheng Zhong, Shu Liu, Bei Yu, and Jia-aya Jia. Parametric contrastive learning. *arXiv preprint arXiv:2107.12028*, 2021. 3
- [11] Terrance DeVries and Graham W Taylor. Learning confidence for out-of-distribution detection in neural networks. *arXiv preprint arXiv:1802.04865*, 2018. 3, 5
- [12] Akshay Dhamija, Manuel Gunther, Jonathan Ventura, and Terrance Boulton. The overlooked elephant of object detection: Open set. In *WACV*, pages 1021–1030, 2020. 1, 2, 3, 6, 9, 11
- [13] Jian Ding, Enze Xie, Hang Xu, Chenhan Jiang, Zhenguo Li, Ping Luo, and Gui-Song Xia. Unsupervised pretraining for object detection by patch reidentification. *arXiv preprint arXiv:2103.04814*, 2021. 3
- [14] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010. 1, 2, 6, 9
- [15] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, pages 1050–1059. PMLR, 2016. 1, 3
- [16] ZongYuan Ge, Sergey Demyanov, Zetao Chen, and Rahil Garnavi. Generative openmax for multi-class open set classification. In *BMVC*, 2017. 1, 2
- [17] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014. 1
- [18] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. *NeurIPS*, 17, 2004. 2
- [19] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Dohersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent: A new approach to self-supervised learning. *arXiv preprint arXiv:2006.07733*, 2020. 3
- [20] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020. 3, 4, 10
- [21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 6, 10
- [22] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016. 11
- [23] Jaedong Hwang, Seoung Wug Oh, Joon-Young Lee, and Bohyung Han. Exemplar-based open-set panoptic segmentation network. In *CVPR*, pages 1175–1184, 2021. 9
- [24] Lalit P Jain, Walter J Scheirer, and Terrance E Boulton. Multi-class open set recognition using probability of inclusion. In *ECCV*, pages 393–409. Springer, 2014. 2
- [25] KJ Joseph, Salman Khan, Fahad Shahbaz Khan, and Vineeth N Balasubramanian. Towards open world object detection. In *CVPR*, pages 5830–5840, 2021. 2, 3, 5, 6, 9, 11
- [26] Pedro R Mendes Júnior, Roberto M De Souza, Rafael de O Werneck, Bernardo V Stein, Daniel V Pazinato, Waldir R de Almeida, Otávio AB Penatti, Ricardo da S Torres, and Anderson Rocha. Nearest neighbors distance ratio open-set classifier. *Machine Learning*, 106(3):359–386, 2017. 2
- [27] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschiot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020. 3, 4, 10
- [28] Shu Kong and Deva Ramanan. Opengan: Open-set recognition via open data generation. *arXiv preprint arXiv:2104.02939*, 2021. 2, 9
- [29] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *arXiv preprint arXiv:1612.01474*, 2016. 3, 6
- [30] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017. 6
- [31] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2980–2988, 2017. 1, 3, 8
- [32] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 2, 6
- [33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021. 10
- [34] Andrey Malinin and Mark Gales. Predictive uncertainty esti-

- mation via prior networks. *arXiv preprint arXiv:1802.10501*, 2018. 6
- [35] Dimity Miller, Feras Dayoub, Michael Milford, and Niko Sünderhauf. Evaluating merging strategies for sampling-based uncertainty techniques in object detection. In *ICRA*, pages 2348–2354. IEEE, 2019. 1, 3
  - [36] Dimity Miller, Lachlan Nicholson, Feras Dayoub, and Niko Sünderhauf. Dropout sampling for robust object detection in open-set conditions. In *ICRA*, pages 3243–3249. IEEE, 2018. 1, 3, 5, 6, 9, 11
  - [37] Lawrence Neal, Matthew Olson, Xiaoli Fern, Weng-Keen Wong, and Fuxin Li. Open set learning with counterfactual images. In *ECCV*, pages 613–628, 2018. 2
  - [38] Poojan Oza and Vishal M Patel. C2ae: Class conditioned auto-encoder for open-set recognition. In *CVPR*, pages 2307–2316, 2019. 2
  - [39] Shreyas Padhy, Zachary Nado, Jie Ren, Jeremiah Liu, Jasper Snoek, and Balaji Lakshminarayanan. Revisiting one-vs-all classifiers for predictive uncertainty and out-of-distribution detection in neural networks. *arXiv preprint arXiv:2007.05134*, 2020. 4
  - [40] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016. 1, 3
  - [41] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. *ICLR*, 2018. 2
  - [42] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE TPAMI*, pages 1137–1149, 2017. 1, 3, 4, 5, 6, 9, 11
  - [43] Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boulton. Toward open set recognition. *IEEE TPAMI*, 35(7):1757–1772, 2012. 1, 9
  - [44] Walter J Scheirer, Lalit P Jain, and Terrance E Boulton. Probability models for open set recognition. *IEEE TPAMI*, 36(11):2317–2324, 2014. 2
  - [45] Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. *arXiv preprint arXiv:1806.01768*, 2018. 3, 5
  - [46] Bo Sun, Banghuai Li, Shengcai Cai, Ye Yuan, and Chi Zhang. Fsce: Few-shot object detection via contrastive proposal encoding. In *CVPR*, pages 7352–7362, 2021. 3
  - [47] Xin Sun, Zhenning Yang, Chi Zhang, Keck-Voon Ling, and Guohao Peng. Conditional gaussian distribution learning for open set recognition. In *CVPR*, pages 13480–13489, 2020. 2, 9
  - [48] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, pages 9627–9636, 2019. 1
  - [49] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, and Luc Van Gool. Unsupervised semantic segmentation by contrasting object mask proposals. *arXiv preprint arXiv:2102.06191*, 2021. 3
  - [50] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Open-set recognition: A good closed-set classifier is all you need. *arXiv preprint arXiv:2110.06207*, 2021. 9
  - [51] Peng Wang, Kai Han, Xiu-Shen Wei, Lei Zhang, and Lei Wang. Contrastive learning based hybrid networks for long-tailed image classification. In *CVPR*, pages 943–952, 2021. 3
  - [52] Wenguan Wang, Tianfei Zhou, Fisher Yu, Jifeng Dai, Ender Konukoglu, and Luc Van Gool. Exploring cross-image pixel contrast for semantic segmentation. *arXiv preprint arXiv:2101.11939*, 2021. 3
  - [53] Xin Wang, Thomas E Huang, Trevor Darrell, Joseph E Gonzalez, and Fisher Yu. Frustratingly simple few-shot object detection. *arXiv preprint arXiv:2003.06957*, 2020. 4
  - [54] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019. 6
  - [55] Ryota Yoshihashi, Wen Shao, Rei Kawakami, Shaodi You, Makoto Iida, and Takeshi Naemura. Classification-reconstruction learning for open-set recognition. In *CVPR*, pages 4016–4025, 2019. 1, 2
  - [56] He Zhang and Vishal M Patel. Sparse representation-based open set recognition. *IEEE TPAMI*, 39(8):1690–1696, 2016. 2
  - [57] Da-Wei Zhou, Han-Jia Ye, and De-Chuan Zhan. Learning placeholders for open-set recognition. In *CVPR*, pages 4401–4410, 2021. 1, 2, 5, 6, 9, 10