Appendix

This appendix is organized as follows:

- Section A.1 gives the details of existing OWOD methods for Section 3.1, where we first give the details of detectors in Section A.1.1, and then give the Known-FG matching method in Section A.1.2. Finally, we give the details of $\mathcal{L}_{cls}$ and $\mathcal{L}_{reg}$ for Eq.1, in Section A.1.3.

- Section A.2 gives the details of 10,000 pre-defined bboxes and how we do postprocessing in the inference stage.

- Section A.3 gives the formal definition and proof of causalities for Section 4.

- Section A.4 provides some experiment implementation details of datasets splitting, evaluation metrics and training schedule.

- Section A.5 shows additional results of experiments. We first give some unknown-class instances detection results by using our matching score in Section A.5.1. Then we show some Open-world detection qualitative results in each task in Section A.5.2.

- Section A.6 discuss the limitation and future work. We first analyze the potential applying range of our method in Section A.6.1. Then we discuss the current large detection models and potential improvement with our random proposals A.6.2.

A.1. Preliminaries Details

In this section, we first provide the details of Faster-RCNN and DETR which are frameworks of existing OWOD methods. Then we give a more detailed discussion on the Known-FG matching. Finally, we give the details of $\mathcal{L}_{cls}$ and $\mathcal{L}_{reg}$ used in training loss.

A.1.1. Detectors of Existing OWOD Methods

Faster R-CNN is composed of two modules. The first module is a deep fully convolutional network that proposes regions, and the second module is the Fast R-CNN detector that uses the proposed regions. The entire system is a single, unified network for object detection. Using the recently popular terminology of neural networks with ‘attention’ mechanisms, the RPN module tells the Fast R-CNN module where to look. For DETR, two ingredients are essential for direct set predictions in detection: (1) a set prediction loss that forces unique matching between predicted and ground truth boxes. (2) an architecture that predicts (in a single pass) a set of objects and models their relation.

A.1.2. Known-FG Matching

The Known-FG matching is how to match the proposals with the ground truth. For Faster R-CNN, proposals are considered as Known-FGs if they have IoUs exceeding a threshold with ground truth. In that case, each ground truth can match multiple proposals. For DETR, it denote the truth set as a set of size $M$ larger than $N$, then pad ground truth as a set of size $M$ (i.e., $\{(b_i,y_i)\}_{i=1}^M$) with $\phi$ (no object). To find a bipartite matching between these two sets, search for a permutation of $M$ elements $\sigma$ with the lowest cost:

$$\arg\min_{\sigma} \sum_{i=1}^{M} L_{match}((b_i, y_i), (\hat{b}_\sigma(i), \hat{y}_{\sigma}(i)))$$  \hspace{1cm} (A1)

where $L_{match}((b_i, y_i), (\hat{b}_\sigma(i), \hat{y}_{\sigma}(i)))$ is a pair-wise matching cost between ground truth $(b_i, y_i)$ and a prediction with index $(\hat{b}_\sigma(i), \hat{y}_{\sigma}(i))$. Each element $i$ of the ground truth set can be seen as a $(b_i, y_i)$ where $y_i$ is the target class label (which may be $\phi$) and $b_i \in [0, 1]^4$ is a vector that defines ground truth box center coordinates and its height and width relative to the image size. For the prediction with index $\sigma(i)$, they define probability of class $c_i$ as $\hat{y}_{\sigma}(i)(c_i)$ and the predicted box as $\hat{b}_{\sigma}(i)$. The $L_{match}$ is defined as:

$$L_{match} = -\mathbb{I}_{c_i \neq \phi \hat{y}_{\sigma}(i)}(c_i) + \mathbb{I}_{c_i \neq \phi} L_{reg}(b_i, \hat{b}_{\sigma}(i))$$  \hspace{1cm} (A2)

where the first term is the cross-entropy and in the second term $L_{reg}$ is:

$$L_{reg} = \lambda_{iou} \mathcal{L}_{iou}(b_i, \hat{b}_{\sigma}(i)) + \lambda_{L_1} ||b_i - \hat{b}_{\sigma}(i)||$$  \hspace{1cm} (A3)

where $\lambda_{iou}, \lambda_{L_1} \in \mathbb{R}$ are hyperparameters.

A.1.3. Training Loss

For the $\mathcal{L}_{cls}$ in the Eq. (1), Faster-RCNN uses the cross-entropy loss.

$$\mathcal{L}_{cls} = -\sum y_i \log \hat{y}_i$$  \hspace{1cm} (A4)

where the $y_i$ is the $i_{th}$ sample’s one-hot encoding of ground truth category and $\hat{y}_i$ is the probability prediction on each categories. In order to solve the problem of sample imbalance, focal loss [6], a improved cross-entropy, is a common loss calculation method.

$$\mathcal{L}_{cls} = -\sum (1 - \hat{y}_i)^\gamma \log \hat{y}_i$$  \hspace{1cm} (A5)

where $\gamma$ is the focusing parameter (a non-negative number), and $(1 - \hat{y}_i)^\gamma$ is a modulating factor which can make the model focus more on difficult-to-classify samples during training by reducing the weight of easy-to-classify samples.
represents the graph where $y$, height, width). We show

d-separation

 RandBox learns the causal effect from $D$ to $Y$ in $G^{-}$. In contrast, $R$ is an instrumental variable in RandBox (Figure 5b).

In $do$-calculus, the average causal effect from $R$ to $Y$ is given by $P(Y|do(R))$. We will show the derivation of the backdoor adjustment for $P(Y|do(R))$ in Figure 5b using the three rules of $do$-calculus [9], and prove that $P(Y|do(R)) = P(Y|R)$, i.e., by learning to predict $Y$ from random $R$, our detector learns the causal effect.

For a causal directed acyclic graph $G$, let $X, Y, Z$ and $W$ be arbitrary disjoint sets of nodes. We use $G^{-}$ to denote the manipulated graph where all incoming arrows to node $X$ are deleted. Similarly $G_{X}$ represents the graph where all outgoing arrows from node $X$ are deleted. We use lower case $x, y, z$ and $w$ for specific values taken by each set of nodes: $X = x, Y = y, Z = z$ and $W = w$. For any interventional distribution compatible with $G$, we have the following three rules:

**Rule 1** Insertion/deletion of observations:

$$P(y|do(x), z, w) = P(y|do(x), w),$$

if $(Y \not\perp \! \! \! \perp Z|X, W)_{G^{-}}$ (A7)

**Rule 2** Action/observation exchange:

$$P(y|do(x), do(z), w) = P(y|do(x), z, w),$$

if $(Y \not\perp \! \! \! \perp Z|X, W)_{G^{-}}$ (A8)

**Rule 3** Insertion/deletion of actions:

$$P(y|do(x), do(z), w) = P(y|do(x), w),$$

if $(Y \not\perp \! \! \! \perp Z|X, W)_{G^{-}}$ (A9)

where $Z(W)$ is the set of nodes in $Z$ that are not ancestors of any $W$-node in $G_{X}$.

In our causal graph (Figure 5b), the desired interven-
Evaluation Metrics in each task are shown in Table A1. The 80 classes of MS-COCO [7] are split into 4 tasks and the number of images as well as instances are d-separated by $\emptyset$, i.e, $\mathcal{D} \perp \mathcal{R}$.

### Evaluation Metrics

In Section 5, we give 4 metrics, i.e., the standard Known-class mAP (K-mAP), recall of unknown classes (U-R), Wilderness Impact (WI) [1] and Absolute Open-Set Error (A-OSE) [8]. In this Section, we add a metric: standard Unknown-class AP (U-AP) at different IoU threshold (e.g., U-AP50 is the unknown-class average precision at 0.5 IoU threshold and U-AP is the average of U-AP[50,95]). This can more intuitively measure both the recall and precision of unknown-class instances. We calculate the $\mathcal{L}^U$ in Eq.2 until 500 iterations in each task due to the matching score is not accurate at the beginning. All model are trained with a mini-batch size 12 on 2 A100 GPUs.

### A.5. Additional Results

In this section, we first give some results on unknown-class detection in A.5.1. Then, we give the previously/currently known scores in A.5.2. In addition, we give some open-world detection qualitative results in A.5.3.

#### A.5.1. Unknown-Class Detection

In Section 5, we give 4 metrics, i.e., the standard Known-class mAP (K-mAP), recall of unknown classes (U-R), Wilderness Impact (WI) [1] and Absolute Open-Set Error (A-OSE) [8]. In this Section, we add a metric: standard Unknown-class AP (U-AP) at different IoU threshold (e.g., U-AP50 is the unknown-class average precision at 0.5 IoU threshold and U-AP is the average of U-AP[50,95]). This can more intuitively measure both the recall and precision of unknown-class instances. We calculate the $\mathcal{L}^U$ in Eq.2 until 500 iterations in each task due to the matching score is not accurate at the beginning. We call this operation 'warm up'. In Table A2, we follow the same setting in Section 5.2 and show the results on with warm up operation or not.

#### A.5.2. Previously/Currently Known Scores

In Table A1, we give the previously/currently known detection in A.5.1. Then, we give the previously/currently known scores in A.5.2. In addition, we give some open-world detection qualitative results in A.5.3.

### A.4. Implementation Details

In this section, we give the details of how the datasets split in 4 tasks, specific IoU threshold of evaluation metrics and the detailed training schedule for each task.

### Datasets Split

The 80 classes of MS-COCO [7] are split into 4 tasks and the number of images as well as instances in each task are shown in Table A1.

### Evaluation Metrics

As described in Section 5, we give the specific IoU threshold for 4 metrics here. The Known-mAP (K-mAP) is at IoU of 0.5, Unknown-Recall (U-R) is at an IoU threshold of 0.5, Wilderness Impact (WI) is at an IoU threshold of 0.8 and Absolute open set error (A-OSE) is at IoU threshold of 0.5.

### Training Schedule

For task 1, the training schedule is 20K iterations, with the learning rate divided by 10 at 15K and 18K iterations. For task 2, 3, 4, the training schedules are 15K iterations, with the learning rate divided by 10 at 10K iterations. After task 2, 3, 4, we fine-tuning the model for 15K iterations, with the learning rate divided by 10 at 5K, 10K, 12K iterations. We calculate the $\mathcal{L}^U$ in Eq.2 until 500 iterations in each task due to the matching score is not accurate at the beginning. All model are trained with a mini-batch size 12 on 2 A100 GPUs.

#### A.5.3. Additional Results

In Table A4, we give the previously/currently known details as supplementary to Table 1.
The current large detection models (e.g., GLIP [5] and Detic [12]) can detect all kinds of objects. They can also achieve the open-world detection to some extent. These models are strong because they use large scale training data. Both GLIP and Detic form pseudo labels based on their proposals, so the proposals are very important in the training stage. Nevertheless, it is noteworthy that both models employ the Region Proposal Network (RPN) as the methodology for proposal generation. This choice may engender a bias towards past training data in the resultant proposals. Similarly, we can introduce our randomness into their training stage to make more explorations and generate more abundant pseudo labels. This is a significant and work-intensive topic. We will explore this and evaluate the effectiveness in our future work.

### A.5.3. Open-world Detection Qualitative Results

We give some open-world detection qualitative results in Figure A2. This figure show a category incremental learning process from top to bottom (Task 1 to Task 4). Two columns are two cases.

First, we look at the first column (case 1). We can see the first row (Task 1), RandBox detects ‘tv’, ‘person’ and multiple ‘unknown’ instances. In these ‘unknown’ instances, we can known they are ‘keyboard’, ‘cup’, ‘laptop’ and so on. These ‘unknown’ instances will be detected as known-class instances as the increasing task number. And in the second row (Task 2), there are still ‘tv’ and ‘person’ are known classes. In the third row (Task 3), ‘orange’ and ‘banana’ were detected as known class after they were introduced to RandBox in Task 3. Finally, in the forth row (Task 4), ‘keyboard’, ‘laptop’ and ‘cup’ were detected as known class. Similarly, we can see another case in the second column. In the first row (Task 1), there are some ‘person’ are detected as known-class instances and multiple ‘unknown’ instances. In these ‘unknown’ instances, we can known they are ‘sink’, ‘banana’, ‘bowl’ and so on. In the second row (Task 2), the ‘sink’ was detected as known class besides ‘person’. In the third row (Task 3), the ‘banana’ was detected as known class after it was introduced in Task 3. Finally, in the forth row (Task 4), RandBox detected the ‘bowl’ and ‘cup’. These results show how our RandBox implement the Open-world detection.

### A.6. Future Work

In this section, We first analyze the potential applying range of our method in Section A.6.1. Then we discuss the current large detection models (e.g., GLIP [5] and Detic [12]) and potential improvement with our Random proposals A.6.2.

#### A.6.1. Applying Range

Our random proposals bridge the gap between incomplete training data and open-world detection by remove the proposal bias. It can be potentially utilized as a plugin to generate unbiased proposals when traditional proposal generation methods would confounded by the training data. However, for the close-set detection, which the distribution of training data and test data are same, random proposals may not surpass the region proposal network or query-based proposal.

#### A.6.2. Improvement for Large Scale Detection Models

Large-scale models have gained widespread popularity and efficacy in contemporary contexts. To be specific, large open-vocabulary detection models like GLIP [5] and Detic [12] can detect all kinds of objects. They can also achieve the open-world detection to some extent. These models are strong because they use large scale training data. Both GLIP and Detic form pseudo labels based on their proposals, so the proposals are very important in the training stage. Nevertheless, it is noteworthy that both models employ the Region Proposal Network (RPN) as the methodology for proposal generation. This choice may engender a bias towards past training data in the resultant proposals. Similarly, we can introduce our randomness into their training stage to make more explorations and generate more abundant pseudo labels. This is a significant and work-intensive topic. We will explore this and evaluate the effectiveness in our future work.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>K-mAP U-R</td>
<td>K-mAP U-R</td>
<td>K-mAP U-R</td>
</tr>
<tr>
<td>0.05</td>
<td>55.8 4.3</td>
<td>59.3 6.9</td>
<td>61.3 9.4</td>
</tr>
<tr>
<td>0.1</td>
<td>56.0 4.9</td>
<td>59.3 7.5</td>
<td><strong>61.8</strong> 10.6</td>
</tr>
<tr>
<td>0.2</td>
<td>56.4 5.3</td>
<td>58.5 8.2</td>
<td>60.2 11.5</td>
</tr>
<tr>
<td>0.5</td>
<td>56.2 5.8</td>
<td>57.6 9.1</td>
<td>59.5 <strong>11.6</strong></td>
</tr>
</tbody>
</table>

Table A4: Additional results supplementary to Table 1.

Figure A2: Open-world Detection Qualitative Results. **Green:** known. **Red:** unknown.

![Figure A2](image-url)
References


