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 by $\{(b_i, y_i)\}_{i=1}^{N}$ the ground truth set of objects (N is the number of objects in the image), and $\{(\hat{b}_i, \hat{y}_i)\}_{i=1}^{M}$ the set of M predictions. Assuming M is larger than N, then pad ground truth as a set of size M (*i.e.*, $\{(b_i, y_i)\}_{i=1}^{M}$) with ø(no object). To find a bipartite matching between these two sets, search for a permutation of M elements σ with the lowest cost:

$$\underset{\sigma}{\operatorname{argmin}} \sum_{i=1}^{M} \mathcal{L}_{match}((b_i, y_i), (\hat{b}_{\sigma(i)}, \hat{y}_{\sigma(i)})) \tag{A1}$$

where $\mathcal{L}_{match}((b_i, y_i), (\hat{b}_{\sigma(i)}, \hat{y}_{\sigma(i)}))$ 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 ø) 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 $\mathcal{L}_{match}$ is defined as:

$$\mathcal{L}_{match} = -\mathbb{1}_{c_i \neq ø}\hat{y}_{\sigma(i)}(c_i) + \mathbb{1}_{c_i \neq ø}\mathcal{L}_{reg}(b_i, \hat{b}_{\sigma(i)}) \tag{A2}$$

where the first term is the cross-entropy and in the second term $\mathcal{L}_{reg}$ is:

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

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.

$$\mathcal{L}_{cls} = -\sum y_i \log \hat{y}_i \tag{A4}$$

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.

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

where γ 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.

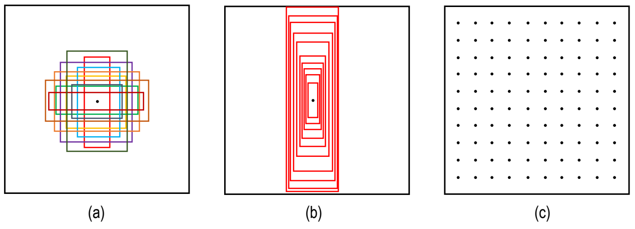

Figure A1: Our pre-defined inference bboxes.

For $\mathcal{L}_{reg}$, Faster-RCNN uses the smooth L1 loss:

$$\mathcal{L}_{reg} = \begin{cases} |x| - 0.5 & |x| > 1 \\ 0.5x^2 & |x| < 1 \end{cases} \tag{A6}$$

where x is the mean absolute error. The $\mathcal{L}_{cls}$ and $\mathcal{L}_{reg}$ of DETR are detailed in A.1.2, which the same as Eq. (A2) and Eq. (A3)

A.2. Inference

In this section, we will give the details of our inference stage including pre-defined bboxes and postprocessing.

Pre-defined bboxes. For each image, we have 10,000 pre-defined bboxes as input for inference. The 10,000 pre-defined bboxes are composed with 100 spatial locations, 10 scales and 10 aspect ratios. Specifically, for each bbox, we first evently drawn from $[0, 1]$ (stride is 0.1). These 4 numbers are the (center_x, center_y, height, width). We show them in Figure A1. The Figure A1a shows that we pre-defined 10 aspect ratios when fixing the position and scale. In the Figure A1b, we fix the position and aspect ratio, and pre-defined 10 scales. In the Figure A1c, we give the 100 pre-defined enter coordinates of the bboxes.

Postprocessing. As we can only input 500 bboxes each time (our model is trained for 500 bboxes input), we infer 20 times on each image and merge all the results to get the predictions of the 10,000 pre-defined bboxes. Finally, we use non-maximum suppression to prune the bboxes at IoU threshold of 0.6.

A.3. Theory

In this section, we will formalize the intuition of "blocked" in Section 4 of the main paper, and prove that RandBox learns the causal effect from D to Y using *do*-calculus [11]. We will first provide the definition of d-separation and instrument variable below.

d-separation. A set of nodes Z blocks a path p if and only if 1) p contains a *chain* $A \to B \to C$ or a *fork* $A \leftarrow B \to C$ and the middle node B is in Z; 2) p contains a *collider* $A \to B \leftarrow C$ such that the middle node B and its descendants are not in Z. If conditioning on Z blocks every path between X and Y, we say X and Y are *d-separated* conditional on Z, *i.e.*, X and Y are independent given Z ($X \perp\!\!\!\perp Y|Z$).

Instrumental Variable. For a structual causal model $\mathcal{G}$, a variable Z is an *instrumental variable* (IV) to $X \to Y$ by satisfying the graphical criteria [10]: 1) $(Z \perp\!\!\!\perp Y)_{\mathcal{G}_{\overline{X}}}$; 2) $(Z \not\perp\!\!\!\perp X)_{\mathcal{G}}$, where $\mathcal{G}_{\overline{X}}$ is the manipulated graph where all incoming arrows to node X are deleted.

One can verify that R is not an instrumental variable in the causal graph of existing methods (Figure 5a), as it violates the first criteria with the unlocked path $R \leftarrow D \to Y$ in $\mathcal{G}_{\overline{X}}$. In contrast, R is an instrument 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 $\mathcal{G}$, let X, Y, Z and W be arbitrary disjoint sets of nodes. We use $\mathcal{G}_{\overline{X}}$ to denote the manipulated graph where all incoming arrows to node X are deleted. Similarly $\mathcal{G}_{\underline{X}}$ represents the graph where 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 $\mathcal{G}$, we have the following three rules:

Rule 1 Insertion/deletion of observations:

$$\begin{aligned} &P(y|do(x), z, w) = P(y|do(x), w), \\ &\text{if} (Y \perp\!\!\!\perp Z|X, W)_{\mathcal{G}_{\overline{X}}} \end{aligned} \tag{A7}$$

Rule 2 Action/observation exchange:

$$\begin{aligned} &P(y|do(x), do(z), w) = P(y|do(x), z, w), \\ &\text{if} (Y \perp\!\!\!\perp Z|X, W)_{\mathcal{G}_{\overline{X}\underline{Z}}} \end{aligned} \tag{A8}$$

Rule 3 Insertion/deletion of actions:

$$\begin{aligned} &P(y|do(x), do(z), w) = P(y|do(x), w), \\ &\text{if} (Y \perp\!\!\!\perp Z|X, W)_{\mathcal{G}_{\overline{X Z(W)}}}, \end{aligned} \tag{A9}$$

where $Z(W)$ is the set of nodes in Z that are not ancestors of any W-node in $\mathcal{G}_{\overline{X}}$.

In our causal graph (Figure 5b), the desired interven-

	Task 1	Task 2	Task 3	Task 4
Semantic split	VOC [2] Classes	Outdoor, Accessories, Appliance, Truck	Sports, Food	Electronic, Indoor, Kitchen, Furniture
train images	16551	45520	39402	40260
test images	4952	1914	1642	1738
train instances	47223	113741	114452	138996
test instances	14976	4966	4826	6039

Table A1: Task composition in the OWOD evaluation protocol.

tional distribution $P(Y|do(R=\mathbf{r}))$ can be derived by:

$$P(Y|do(\mathbf{r})) = \sum_d P(Y|do(\mathbf{r}),d)P(D=d|do(\mathbf{x})) \quad \text{(A10)}$$

$$= \sum_d P(Y|do(R=\mathbf{r}),d)P(D=d) \quad \text{(A11)}$$

$$= \sum_d P(Y|R=\mathbf{r},d)P(D=d) \quad \text{(A12)}$$

$$= \sum_d P(Y|R=\mathbf{r},d)P(D=d|R=r) \quad \text{(A13)}$$

$$= P(Y|R=r) \quad \text{(A14)}$$

where Eq. (A10) and Eq. (A14) follow the law of total probability; Eq. (A11) uses Rule 3 given $D \perp\!\!\!\perp X$ in $\mathcal{G}_{\overline{X}}$; Eq. (A12) uses Rule 2 to change the intervention term to observation as $(Y \perp\!\!\!\perp X|D)$ in $\mathcal{G}_{\underline{X}}$. Eq. (A13) is because D and R are d-separated by $\emptyset$, *i.e.*, $D \perp\!\!\!\perp R$.

A.4. Implementation Details

In this section, we give the details of how the datasets splitted 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 threshold of 0.5, Unknown-Recall (U-R) is at IoU threshold of 0.5 , Wilderness Impact (WI) is at 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.

warm up	U-AP	U-AP50	U-AP75
	1.98	4.46	1.57
✓	**2.57**	**5.25**	**2.23**

Table A2: Unknown-class Average precision (U-AP) on whether to use warm up operation.

ratio	scale	location	U-AP	U-AP50	U-AP75
5	5	20	2.33	4.47	2.10
10	10	20	2.56	5.20	2.22
10	10	100	**2.57**	**5.25**	**2.23**

Table A3: Unknown-class Average precision (U-AP) on different inference bboxes.

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 threshol 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. The results show that our warm up operation can improve the U-AP because after some warm up iteration, our matching score can recall the Unknown-FG with higer precision. Then, we show the U-AP on different inference Bboxes in Table A3. The results show that increasing the number of inference bboxes generally improves the U-AP due to the increased recall on unknown-class instances.

A.5.2. Previously/Currently Known Scores

In Table A4, we give the previously/currently known details as supplementary to Table 1.

β	ORE [4]		OW-DETR [3]		RandBox	
	K-mAP	U-R	K-mAP	U-R	K-mAP	U-R
0.05	55.8	4.3	59.3	6.9	61.3	9.4
0.1	56.0	4.9	59.3	7.5	**61.8**	10.6
0.2	56.4	5.3	58.5	8.2	60.2	11.5
0.5	56.2	5.8	57.6	9.1	59.5	**11.6**

Table A4: Additional results supplementary to Table 1.

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.

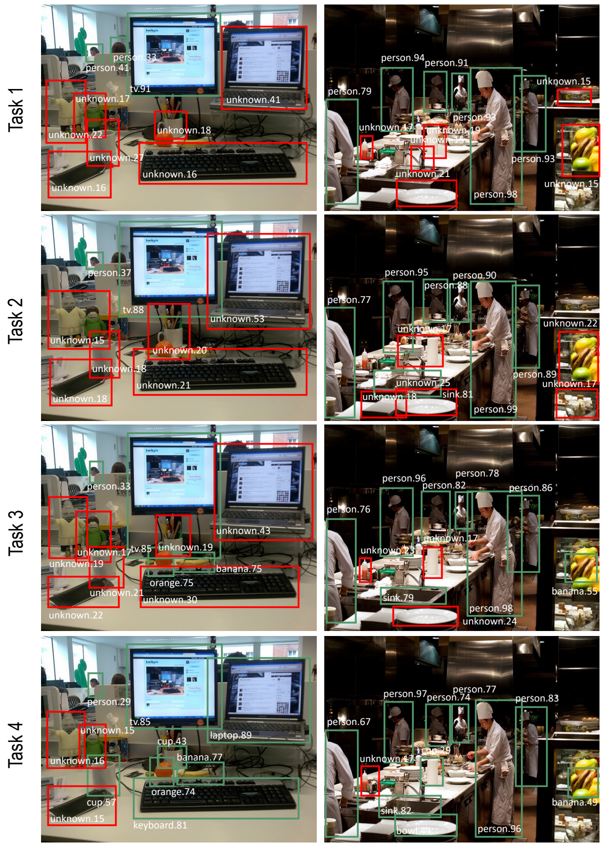

Figure A2: **Open-world Detection Qualitative Results. Green: known. Red: unknown.**

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.

References

[1] Akshay Dhamija, Manuel Gunther, Jonathan Ventura, and Terrance Boult. The overlooked elephant of object detection: Open set. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1021–1030, 2020. 3

[2] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 3

[3] Akshita Gupta, Sanath Narayan, KJ Joseph, Salman Khan, Fahad Shahbaz Khan, and Mubarak Shah. Ow-detr: Open-world detection transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9235–9244, 2022. 4

[4] KJ Joseph, Salman Khan, Fahad Shahbaz Khan, and Vineeth N Balasubramanian. Towards open world object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5830–5840, 2021. 4

[5] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022. 4

[6] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 1

[7] 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 *European conference on computer vision*, pages 740–755. Springer, 2014. 3

[8] Dimity Miller, Lachlan Nicholson, Feras Dayoub, and Niko Sünderhauf. Dropout sampling for robust object detection in open-set conditions. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3243–3249. IEEE, 2018. 3

[9] Judea Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan kaufmann, 1988. 2

[10] Judea Pearl. *Causality*. Cambridge university press, 2009. 2

[11] Judea Pearl et al. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress*, 19(2), 2000. 2

[12] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *European Conference on Computer Vision*, pages 350–368. Springer, 2022. 4