Supplementary Material for "Detecting Everything in the Open World: Towards Universal Object Detection"

1. Dataset Details

We provides more detailed descriptions in this section about the datasets we use in the main paper and in this supplementary material.

COCO [20]. The COCO dataset contains dense and high-quality annotations from human labor and covers 80 common classes from nature-domain images. For our open-world experiments, we select 35k images from its training set for training our UniDetector. The high-quality annotations also make the COCO dataset widely used in traditional object detection. Because of this, we also conduct experiments on the COCO dataset for evaluating our UniDetector in the closed world. We train our UniDetector on the 115k images from the COCO 2017 training set and evaluate it on the validation set of 5,000 images. We also report the closed-world AP on the test-dev split with 20k images.

Objects365 [29]. The Objects365 dataset contains high-quality annotations with 365 categories in the natural domain. We randomly select 60k images from the training set of its v1 version for our open-world experiments.

OpenImages [16]. We adopt the 2019 challenge version of the OpenImages dataset, which contains 500 categories and more than 1.7M images. We select 78k images for our open-world experiments. Because of its large category number, its category distribution is relatively long-tailed.

LVIS [12]. The v0.5 version contains 57,563 images for training, 5,000 images for validation and 1,230 categories. The v1 version contains 100,170 images for training, 19,809 images for validation and 1,203 categories. We report results of our UniDetector on their validation sets for evaluating the open-world performance. Besides, to compare with other open-vocabulary methods, we also train our UniDetector on the training set without rare category annotations.

ImageNet [28]. The complete set of the ImageNet dataset contains over 21k categories. However, only 3,622 categories among them have bounding box annotations. It is also used to evaluate our open-world performance. Besides, we also introduce images from the ImageNet dataset with the overlapping classes of LVIS and only category annotations for open-vocabulary experiments, for comparison with previous methods [41].

VisualGenome [15]. We adopt the most recent version (v1.4) of the VisualGenome dataset, which consists of 7,605 categories in total. It covers the largest vocabularies with bounding box annotations so far. We utilize it to evaluate the open-world performance of our UniDetector.

Mapillary Vistas [22]. The Mapillary Vistas Dataset is a street-level dataset. It contains 37 instance-level categories. In this supplementary material, we further introduce the 18k images from the training set to train our UniDetector for better generalization ability in traffic-domain images.

Pascal VOC [9]. It annotates 20 common categories from nature-domain images. The 16k images from the VOC0712 trainval set are usually adopted for training. We adopt the 4,952 images from the VOC2007 test set for evaluation.

VIPER [27]. It covers 10 categories collected from high-resolution traffic videos, with 13k images as the training set and 5k images as the validation set.

Cityscapes [4]. The Cityscapes dataset contains 8 categories from traffic images. Its training set contains 2,975 images and the validation set contains 500 images.

Scannet [5]. The Scannet dataset is an indoor dataset with 13 indoor instance-level categories. It contains 25k images in total. We select the first 80% as the training set and the rest of them as the validation set.

WildDash [38]. The WildDash dataset is mainly about robustness in driving scenarios. It provides 13 categories and 4k images for evaluation.

CrowdHuman [30]. The CrowdHuman dataset is about humans in crowded scenes. It contains 15k images. We use the visible bounding box annotations for detection.

KITTI [11]. We used the RVC challenge version that has instance segmentation labels, which contain 200 images and 8 categories.

ODinW [17]. We also adopt the 13 ODinW datasets with various domains and categories for demonstrating the uni-

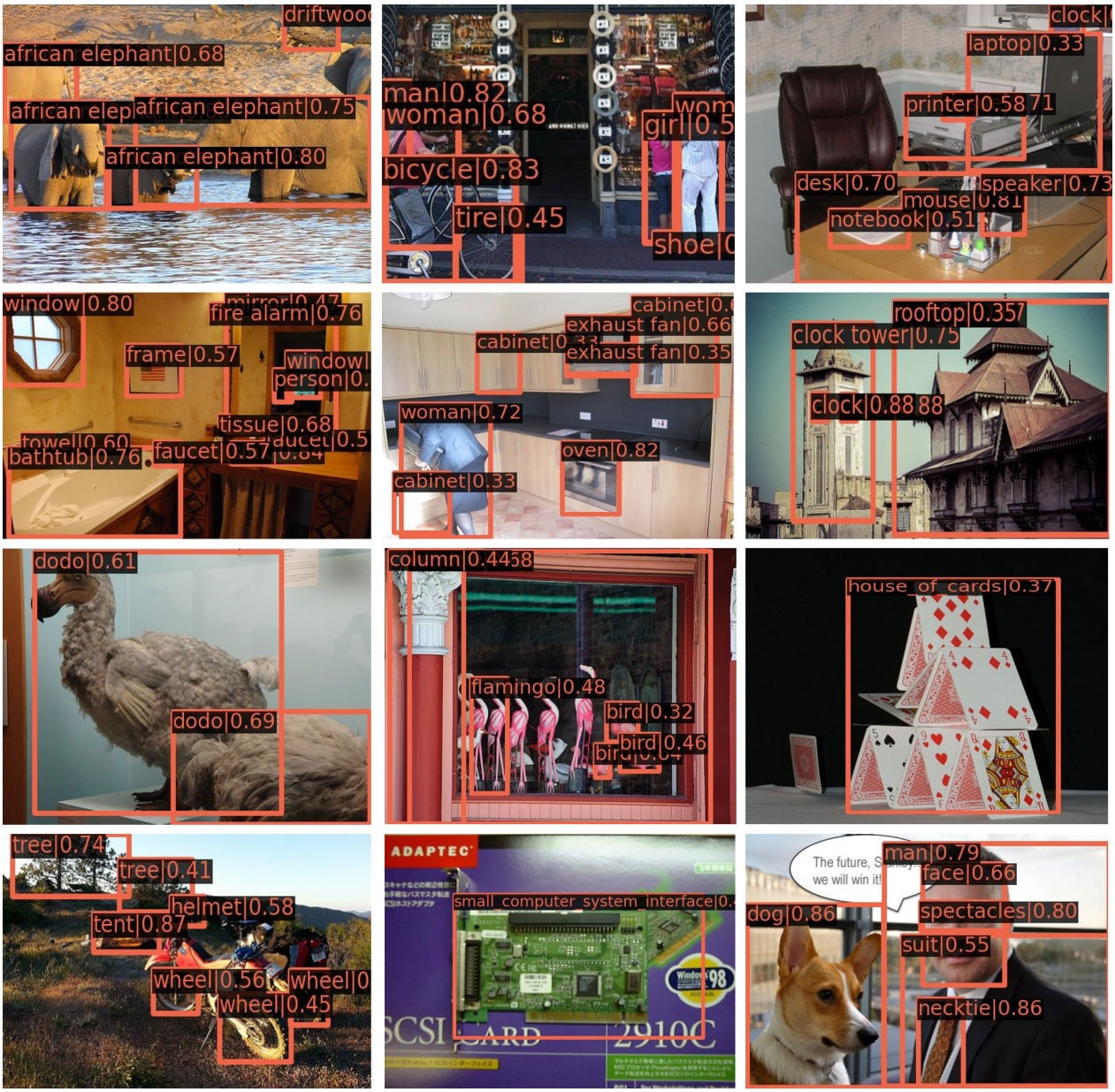

Figure 1. **Visualization of our UniDetector to detect diversified categories.**

versality of our UniDetector.

2. Visualized Results

Visualized results to detect diversified categories. We provide visualized results from our UniDetector to demonstrate its ability to detect everything. The visualization is illustrated in Fig. 1. First, our UniDetector recognizes many rare categories that are not available in training, such as `dodo`, a bird species that is extinct now, or `exhaust fan`, `rooftop` and so on. Second, our UniDetector is able to detect fine-grained categories and classes that are a part of an object, like `man`, `african elephant`, `face`. Third, our UniDetector can detect categories that are composed of many words, such as `house of cards`, `small computer system interface`. These demonstrate that our UniDetector well understands the meaning of natural languages, thus has the strong generalization ability to novel classes. The universality of our UniDetector in detecting everything can thus be demonstrated.

Visualized results to detect diversified scenes. We further provide visualized images from diversified scenes (*i.e.*,

Figure 2. **Visualization of our UniDetector to detect diversified scenes.**

domains) in Fig. 2. As we can see, our UniDetector performs well not only on common scenes that appear during training, like `traffic`, `indoor`, but also generalizes well on unseen domains like `underwater`, `UAV`, `thermal`, `watercolors`. As a result, our UniDetector makes a satisfying performance in various scenes. The universality of UniDetector in detecting every scene can be demonstrated.

3. More Closed-world Detection Results

We provide more closed-world detection results in this section. We train our UniDetector on the COCO 2017 training set and evaluate its performance on the 2017 test-dev set. We adopt the ResNet101 and ResNet50x4 (*i.e.*, ResNet200) [13] as our backbone and train our UniDetector with the 1× schedule. Our UniDetector has a pure CNN structure so we compare it with existing CNN-based detectors. They mainly utilize ResNeXt [37], DCN [6], SENet [14], EfficientNet [33] or SpineNet [8] as their backbones, with similar or larger computation budgets. The comparison is listed in Tab. 1.

As we can see, with the ResNet101 backbone, we achieve the 51.8% box AP on the test-dev set, which not only surpasses existing detectors with the same ResNet101

Table 1. **The performance of UniDetector in the closed world with larger backbones.** The models are trained on the COCO train2017 set and evaluated on the COCO test-dev. TTA is test-time augmentation, ME means more epochs of training. We compare our UniDetector with existing state-of-the-art detectors with the pure CNN sturcture.

Methods	Backbone	ME	TTA	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Faster RCNN [26]	ResNet101	12e		36.7	54.8	39.8	19.2	40.9	51.6
RetinaNet [19]	ResNeXt101	18e		40.8	61.1	44.1	24.1	44.2	51.2
Cascade RCNN [1]	ResNet101	18e		42.8	62.1	46.3	23.7	45.5	55.2
Libra R-CNN [23]	ResNeXt-101	12e		43.0	64.0	47.0	25.3	45.6	54.6
FCOS [35]	ResNeXt101	24e		43.2	62.8	46.6	26.5	46.2	53.3
ATSS [40]	ResNeXt101-DCN	24e		47.7	66.5	51.9	29.7	50.8	59.4
OTA [10]	ResNeXt101-DCN	24e		49.2	67.6	53.5	30.0	52.5	62.3
IQDet [21]	ResNeXt101-DCN	24e		49.0	67.5	53.1	30.0	52.3	62.0
Sparse R-CNN [7]	ResNeXt101-DCN	36e		48.9	68.3	53.4	29.9	50.9	62.4
EfficientDet [34]	EfficientNet-B7	$\sim$ 600e		52.2	71.4	56.3	-	-	-
SpineNet [8]	SpineNet190	$\sim$ 500e		52.1	71.8	56.5	35.4	55.0	63.6
Dyhead [7]	ResNeXt101-DCN	24e		52.3	70.7	57.2	35.1	56.2	63.4
$R(Det)^2$ [18]	ResNeXt101-DCN	12e		50.0	69.2	54.3	30.9	53.0	63.9
UniDetector (ours)	ResNet101	12e		51.8	70.2	56.8	35.6	55.5	62.2
UniDetector (ours)	ResNet200	12e		**55.8**	**74.1**	**61.2**	**37.8**	**58.7**	**68.1**
ATSS [40]	ResNeXt101-DCN	24e	✓	50.7	68.9	56.3	33.2	52.9	62.4
IQDet [21]	ResNeXt101-DCN	24e	✓	51.6	68.7	57.0	34.5	53.6	64.5
OTA [10]	ResNeXt101-DCN	24e	✓	51.5	68.6	57.1	34.1	53.7	64.1
Dynamic R-CNN [39]	ResNet-101-DCN	36e	✓	50.1	68.3	55.6	32.8	53.0	61.2
TSD [31]	SENet154-DCN	36e	✓	51.2	71.9	56.0	33.8	54.8	64.2
Sparse R-CNN [7]	ResNeXt101-DCN	36e	✓	51.5	71.1	57.1	34.2	53.4	64.1
BorderDet [25]	ResNeXt101-DCN	24e	✓	50.3	68.9	55.2	32.8	52.8	62.3
RepPoints v2 [2]	ResNeXt101-DCN	24e	✓	52.1	70.1	57.5	34.5	54.6	63.6
RelationNet++ [3]	ResNeXt101-DCN	24e	✓	52.7	70.4	58.3	35.8	55.3	64.7
DyHead [7]	ResNeXt101-DCN	24e	✓	54.0	72.1	59.3	37.1	57.2	66.3
$R(Det)^2$ [18]	ResNeXt101-DCN	24e	✓	54.1	72.4	59.4	35.5	57.0	67.3
UniDetector (ours)	ResNet200	12e	✓	**56.9**	**34.8**	**62.5**	**40.0**	**59.4**	**68.4**

backbone, but even outperforms many models with the more complicated ResNeXt101-DCN backbone. For example, we are 1.8% higher in box AP than $R(Det)^2$ with the ResNeXt101-DCN backbone. We achieve better detection performance with a lighter backbone, which strongly illustrates the effectiveness of our UniDetector in the closed world. We further introduce the ResNet200 backbone, and obtain the 55.8% box AP with just 12 epochs. In comparison, Dyhead [7] obtains the 52.3% AP with 24 epochs, EfficientDet [34] requires about 600 epochs for the 52.2% AP and SpineNet [8] obtains the 52.1% AP with about 500 epochs. With significantly less training epochs, we achieve the 3.5% higher AP. The superiority of our UniDetector is consistent no matter for small or large objects - AP_S surpasses existing methods by 2.4%, AP_M is 2.5% higher and AP_L is even 4.2% higher. We also notice that our 55.8% AP is even higher than the results of existing detectors with test-time augmentation (TTA). This further demonstrates the effectiveness of our UniDetector. By further introducing test-time augmentation into our UniDetector, we achieve the 56.9% AP. The excellent performance in the close world validates its universality.

We note that we actually introduce the cascade structure for above experiments. We modify the cascade structure a little to adapt our decoupling training manner. The comparison with the original Cascade RCNN [1] is illustrated in Fig. 3. As the class-specific classification of the RoI classification stage hampers the universality ability of the proposal generation stage, we do not use the box regression layer in the RoI head to refine region proposals, which is used in Cascade RCNN (Fig. 3a). Instead, we pass the extracted region proposals directly into the RoI heads in the cascade structure, as in Fig. 3b. This structure better adapts our decoupling training manner.

4. Data Sampler Analysis

Here we analyze the effect of data samplers in our UniDetector. Large-scale training is necessary to guarantee sufficient information for universality. However, long-tailed distribution is an unavoidable problem for datasets

Table 2. **Analysis on the data sampler for detecting in the closed world and open world.** The model is trained on 78k images from OpenImages and LVIS images are adopted for open-world evaluation. For our UniDetector, the two samplers listed are used in the proposal generation stage and RoI classification stage separately.

model	sampler	open world (LVIS v0.5)				closed world (OpenImages)		
		AP	AP_r	AP_c	AP_f	$AP_{hierarchy}$	AP	AP_{50}
Faster RCNN	random	-	-	-	-	39.8	15.2	25.5
	CAS [24]	-	-	-	-	49.8	19.5	32.4
	RFS [12]	-	-	-	-	46.4	17.3	28.9
UniDetector	random + random	**16.8**	**21.8**	**17.6**	13.8	59.0	**25.7**	**36.9**
	random + CAS	15.7	20.2	16.5	13.0	58.3	23.7	34.7
	random + RFS	16.6	21.4	17.3	**14.0**	**59.5**	**25.7**	36.8
	CAS + random	14.4	20.1	15.3	10.9	49.5	15.3	24.2
	CAS + CAS	13.0	16.6	13.8	10.4	49.7	14.4	23.1
	CAS + RFS	13.8	18.0	14.5	11.1	50.3	15.9	24.9
	RFS + random	13.5	14.7	15.1	10.9	47.6	15.1	23.2
	RFS + CAS	12.5	13.8	13.8	10.4	47.5	13.4	21.4
	RFS + RFS	13.4	14.7	14.9	11.1	48.2	15.2	23.6

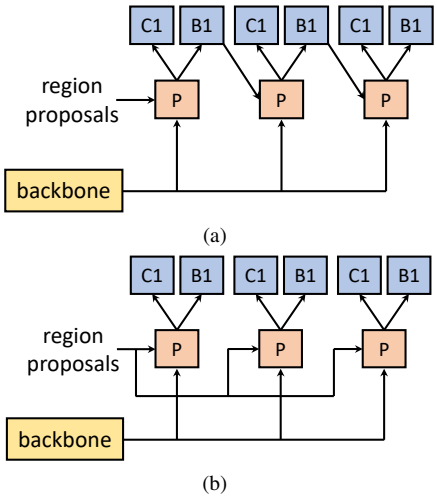

Figure 3. **Illustration for the cascade structure we use.** (a): The original cascade structure in Cascade RCNN. (b): The cascade structure we use to adapt our decoupling training manner.

with large vocabularies. Suitable data samplers are simple yet effective for alleviating the adverse effect of the long-tailed problem. The common data samplers are as follows:

Random sampler. It samples images randomly from the training set. Every image participates in the training equally. When data become long-tailed, this way samples head categories more frequently than tail classes. In traditional object detection, this makes tail classes under-represented easily.

Class-aware sampler (CAS) [24]. The CAS sampler first randomly selects categories, then sample images containing the selected categories. Such a sampler makes different categories participate in training equally.

Repeat factor sampler (RFS) [12]. It defines the category-level repeat factor as $r_c = \max(1, \sqrt{1/f_c})$, where f_c is the category frequency. The image-level sampler is its maximum value of r_c. The RFS sampler also selects tail categories more frequently.

In this section, we train our UniDetector with 78k images from OpenImages because of the long-tailed distribution of the OpenImages dataset. We evaluate it on the LVIS v0.5 dataset for the open-world performance and on the OpenImages dataset for the closed-world performance. As the category labels of OpenImages are hierarchically defined, we also introduce the $AP_{hierarchy}$ from its official challenge to evaluate the closed-world performance. The analysis results are listed in Tab. 2.

We first train a traditional Faster RCNN model, which can only detect in the closed world. The random sampler generates a 39.8% $AP_{hierarchy}$. In comparison, the CAS and the RFS bring a 10.0% and 6.6% improvement respectively. The significant improvement demonstrates that these data sampling strategies well alleviate the long-tailed problem in the closed world. As different data samplers nearly do not bring any extra computation budget, they are widely adopted in traditional large-scale object detection.

We then conduct experiments using our UniDetector. These samplers are adopted in both the proposal generation stage and the RoI classification stage. We first train the proposal generation stage with the random sampler and introduce different samplers in the RoI classification stage. We find that compared to the random sampler, the CAS and RFS are no longer effective in the closed world. The RFS only brings a 0.5% $AP_{hierarchy}$ improvement, and the CAS even hurts the closed-world performance. Meanwhile, our UniDetector also obtains a significantly better closed-world performance, more than 10% improvement than tra-

ditional Faster RCNN. The above phenomenon is because our UniDetector utilizes language embeddings for classification. These language embeddings come from the text description, thus are not limited by the long-tailed distribution of the visual information. Therefore, the choice of data sampler is no longer effective for our UniDetector.

For the open-world performance, we find that the random sampler obtains the 16.8% box AP on LVIS v0.5. In the open world, the CAS and RFS do not contribute to the AP improvement. Instead, they decrease the box AP by 1.1% and 0.2% respectively. For open-world inference, how to generalize to novel categories is quite essential. Different data sampler cannot improve this ability, thus is on longer helpful in the open world environments.

We finally train the proposal generation stage with different samplers, and find that the CAS and RFS hurt the performance more seriously, no matter for the open world or closed world. This is easy to understand since the classification in this stage is class-agnostic, which is robust to the long-tailed distribution of categories. Because of the above reasons, we choose to use the random sampler.

5. Loss Function Analysis

Likewise, the choice of the loss function is also an important issue in traditional large-scale object detection, where the long-tailed problem happens. The commonly used loss functions are as follows:

Cross entropy loss and sigmoid based loss. The cross entropy loss and sigmoid based binary cross entropy loss are widely adopted in the classification problem. They are usually based on the softmax activation function and the sigmoid function. However, they treat different categories equally, thus suffer from the long-tailed problem in traditional object detection.

Equalization loss v2 and seesaw loss [32, 36]. These two loss functions adjust the gradients of negative samples to balance the learning process. As a result, the performance on tail classes can be improved.

Federate loss [42]. The federate loss is conducted only on a subset of classes for training to adapt the LVIS dataset. The subset contains all positive samples and a random subset of negative samples. The negative categories are sampled in proportion to their square-root frequency.

In this section, similarly, we train our UniDetector with 78k images from OpenImages and use the LVIS v0.5 dataset for open-world evaluation. The analysis results are listed in Tab. 3. We notice that both the equalization loss v2 and seesaw loss contribute to better performance in the closed world - 9.7% and 0.7% improvement separately. Their effectiveness in traditional detection is thus demonstrated.

However, according to our analysis above, language embeddings in our UniDetector greatly alleviate the long-tailed distribution. This makes loss functions like equalization loss v2 or seesaw loss affect little in the open world. In this situation, the generalization ability to novel categories is quite important, and sigmoid function well enhances such ability. Unlike the softmax activation function, whose lateral inhibition effect is obvious, the classification of base and novel categories will not interfere with each other under the sigmoid function. This promotes the novel category detection in the open world. As a result, the sigmoid based loss obtains the 16.2% LVIS AP, 0.5% higher than that from the cross entropy loss. We also observe that such property ofthe sigmoid function boosts the closed world detection too, improving $\text{AP}_{\text{hierarchy}}$ from 58.2% to 59.9%.

One problem of sigmoid based loss is that its value becomes larger as the number of categories increases. For large-vocabulary datasets, this easily interferes with the collaborative learning of classification and regression. To address this issue, we only randomly select a subset of negative categories to calculate the loss function. This produces a 16.8% LVIS AP, 0.6% improvement. In comparison, although federate loss also picks negative categories, sampling according to frequency does not adapt to the situation where the long-tailed problem is not serious. Therefore, it fails to boost the performance of our UniDetector.

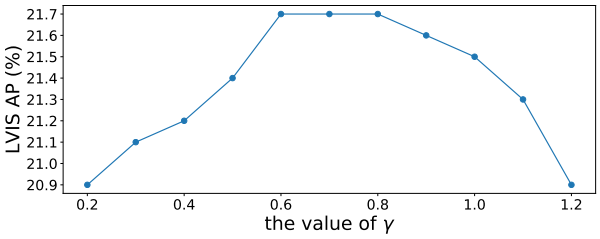

Figure 4. **The open-world performance *v.s.* the hyperparameter γ.** The LVIS AP is robust to the value of γ.

6. The hyperparameter γ

We further tune the value of γ and plot the results in Fig. 4. As we can see, the open-world AP is relatively robust to the value of γ. The AP remains the maximum value in the range from 0.6 to 0.8 of γ. Such hyper-parameter robustness makes our probability calibration easy to implement in practice.

7. Comparative Visualized Results

We finally conduct visualized analysis on the COCO open-vocabulary setting and compare our results with the Faster RCNN baseline and Detic [41]. As we can see, Faster RCNN is restricted by the pure visual information during training, thus cannot recognize novel categories at all. For

Table 3. **Analysis on the loss function for detecting in the closed world and open world.** The model is trained on 78k images from OpenImages and LVIS images are adopted for open-world evaluation.

model	loss	open world (LVIS v0.5)				closed world (OpenImages)		
		AP	AP_r	AP_c	AP_f	$AP_{hierarchy}$	AP	AP_{50}
Faster RCNN	cross entropy	-	-	-	-	39.8	15.2	25.5
	equalization loss v2 [32]	-	-	-	-	49.5	19.6	31.5
	seesaw loss [36]	-	-	-	-	40.5	14.5	24.0
UniDetector	cross entropy	15.7	20.1	16.4	13.0	58.2	24.2	35.9
	equalization loss v2 [32]	14.8	18.3	15.9	12.0	50.1	18.0	26.6
	seesaw loss [36]	13.7	16.3	14.6	11.7	58.9	27.3	39.6
	sigmoid	16.2	20.5	17.1	13.2	**59.9**	**28.3**	**41.9**
	federate loss [42]	16.3	20.4	17.0	13.7	58.7	25.3	36.2
	sigmoid (random negative)	**16.8**	**21.8**	**17.6**	**13.8**	59.0	25.7	36.9

example, it can only detect the dog as a bird for the first image, and detect the airplane as a surfboard for the second image, the bus as a truck for the third image. In addition, the couch is also missed in the fourth image. Detic adopts language embeddings from CLIP, thus has the ability to recognize categories that do not appear during training. For example, it produces the dog and the cat box for the first image. However, its classification is seriously biased to seen categories. For example, the incorrect bird prediction still exists in the first image, and the correct unseen categories still cannot be predicted for the rest three images. Because of our decoupling training manner and the probability calibration, our UniDetector well alleviates the self-bias to seen categories. As a result, our UniDetector generates correct detection boxes for unseen categories, and avoids overconfident incorrect seen category boxes. This comparative visualized result demonstrates the generalization ability of our UniDetector to unseen categories in the open world.

8. Limitation and Social Impact

Our method does not consider the suitable language prompt for the object detection task yet, and this can be further investigated. We also recognize that our method might lead to privacy concerns if not properly utilized.

References

[1] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018.

[2] Yihong Chen, Zheng Zhang, Yue Cao, Liwei Wang, Stephen Lin, and Han Hu. Reppoints v2: Verification meets regression for object detection. In *NeurIPS*, 2020.

[3] Cheng Chi, Fangyun Wei, and Han Hu. Relationnet++: Bridging visual representations for object detection via transformer decoder. In *NeurIPS*, 2020.

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[5] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, 2017.

[6] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *ICCV*, 2017.

[7] Xiyang Dai, Yinpeng Chen, Bin Xiao, Dongdong Chen, Mengchen Liu, Lu Yuan, and Lei Zhang. Dynamic head: Unifying object detection heads with attentions. In *CVPR*, 2021.

[8] Xianzhi Du, Tsung-Yi Lin, Pengchong Jin, Golnaz Ghiasi, Mingxing Tan, Yin Cui, Quoc V Le, and Xiaodan Song. Spinenet: Learning scale-permuted backbone for recognition and localization. In *CVPR*, 2020.

[9] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[10] Zheng Ge, Songtao Liu, Zeming Li, Osamu Yoshie, and Jian Sun. Ota: Optimal transport assignment for object detection. In *CVPR*, 2021.

[11] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012.

[12] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[14] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018.

[15] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017.

[16] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4. *IJCV*, 2020.

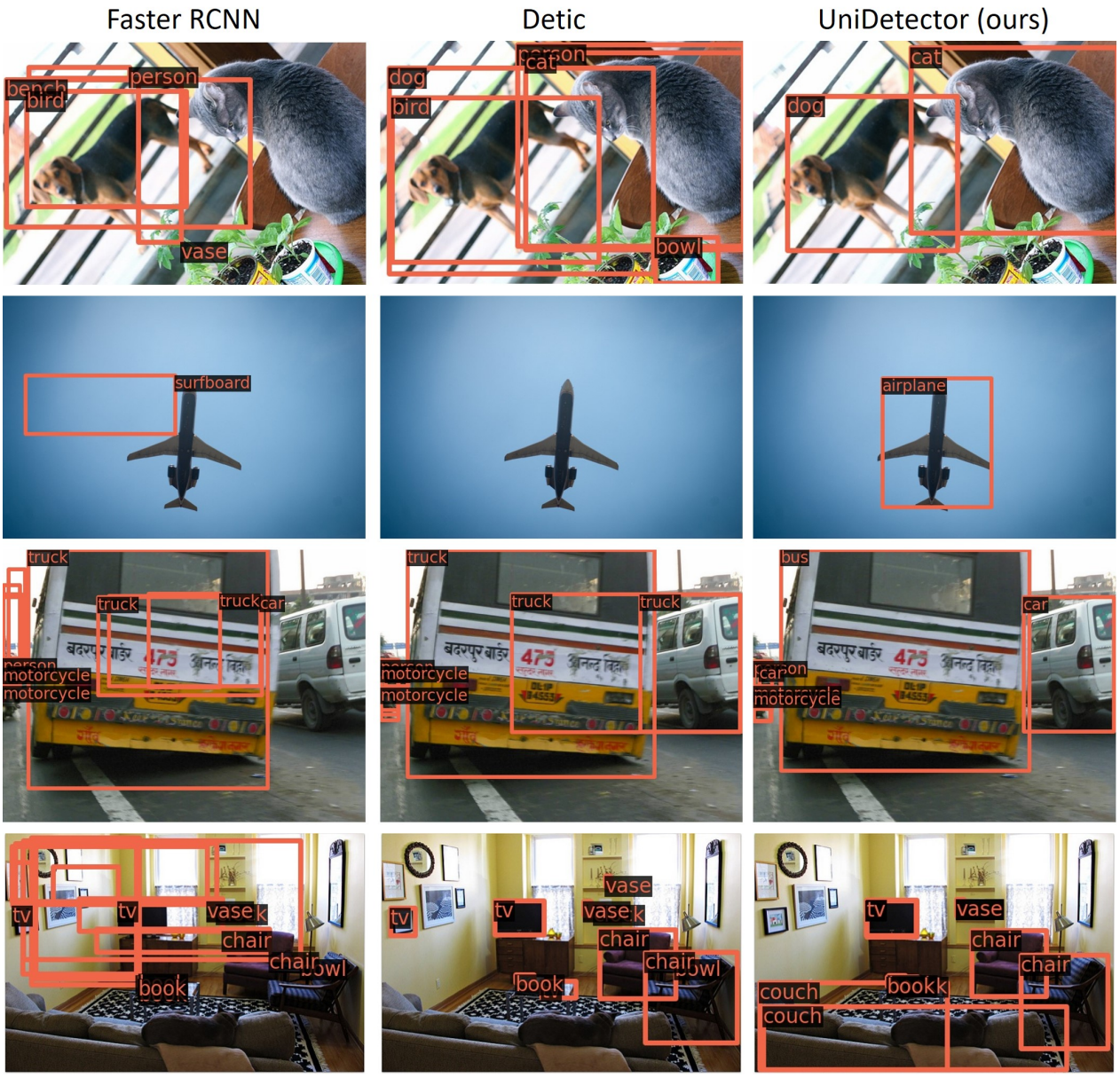

Figure 5. **Comparison with the Faster RCNN baseline and Detic on the COCO dataset.** The models are trained on the COCO open-vocabulary setting, with 48 categories seen during training. The `cat`, `dog`, `airplane`, `bus`, `couch` are all **unseen** categories in training.

[17] 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 *CVPR*, 2022.

[18] Yali Li and Shengjin Wang. R (det) 2: Randomized decision routing for object detection. In *CVPR*, 2022.

[19] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.

[20] 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*, 2014.

[21] Yuchen Ma, Songtao Liu, Zeming Li, and Jian Sun. Iqdet: Instance-wise quality distribution sampling for object detection. In *CVPR*, 2021.

[22] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, 2017.

[23] Jiangmiao Pang, Kai Chen, Jianping Shi, Huajun Feng, Wanli Ouyang, and Dahua Lin. Libra r-cnn: Towards balanced learning for object detection. In *CVPR*, 2019.

[24] Junran Peng, Xingyuan Bu, Ming Sun, Zhaoxiang Zhang, Tieniu Tan, and Junjie Yan. Large-scale object detection in the wild from imbalanced multi-labels. In *CVPR*, 2020.

[25] Han Qiu, Yuchen Ma, Zeming Li, Songtao Liu, and Jian Sun. Borderdet: Border feature for dense object detection. In *ECCV*, 2020.

[26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[27] Stephan R Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for benchmarks. In *ICCV*, 2017.

[28] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[29] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *ICCV*, 2019.

[30] Shuai Shao, Zijian Zhao, Boxun Li, Tete Xiao, Gang Yu, Xiangyu Zhang, and Jian Sun. Crowdhuman: A benchmark for detecting human in a crowd. *arXiv preprint arXiv:1805.00123*, 2018.

[31] Guanglu Song, Yu Liu, and Xiaogang Wang. Revisiting the sibling head in object detector. In *CVPR*, 2020.

[32] Jingru Tan, Xin Lu, Gang Zhang, Changqing Yin, and Quanquan Li. Equalization loss v2: A new gradient balance approach for long-tailed object detection. In *CVPR*, 2021.

[33] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.

[34] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *CVPR*, 2020.

[35] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *ICCV*, 2019.

[36] Jiaqi Wang, Wenwei Zhang, Yuhang Zang, Yuhang Cao, Jiangmiao Pang, Tao Gong, Kai Chen, Ziwei Liu, Chen Change Loy, and Dahua Lin. Seesaw loss for long-tailed instance segmentation. In *CVPR*, 2021.

[37] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

[38] Oliver Zendel, Katrin Honauer, Markus Murschitz, Daniel Steininger, and Gustavo Fernandez Dominguez. Wilddash-creating hazard-aware benchmarks. In *ECCV*, 2018.

[39] Hongkai Zhang, Hong Chang, Bingpeng Ma, Naiyan Wang, and Xilin Chen. Dynamic r-cnn: Towards high quality object detection via dynamic training. In *ECCV*, 2020.

[40] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z Li. Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection. In *CVPR*, 2020.

[41] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Phillip Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. *arXiv:2201.02605*, 2022.

[42] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Probabilistic two-stage detection. *arXiv:2103.07461*, 2021.