Supplementary Material: MOSAICOS: A Simple and Effective Use of Object-Centric Images for Long-Tailed Object Detection

Cheng Zhang[1*] Tai-Yu Pan[1*] Yandong Li[2] Hexiang Hu[3]
Dong Xuan[1] Soravit Changpinyo[2] Boqing Gong[2] Wei-Lun Chao[1]

[1]The Ohio State University [2]Google Research [3]University of Southern California

In this Supplementary Material, we provide details and results omitted in the main text.

- Appendix A: contributions. (§ 7 of the main paper)
- Appendix B: additional details and results on pseudo-label generation. (§ 5.1 of the main paper)
- Appendix C: ablation studies on image mosaicking. (§ 5.2 of the main paper)
- Appendix D: further analysis on self-training. (§ 6.1 and § 6.2 of the main paper)
- Appendix E: further analysis on data quality of object-centric images. (§ 3 and § 6.3 of the main paper)
- Appendix F: comparison to adversarial training. (§ 4 of the main paper)
- Appendix G: implementation details of MOSAICOS on object detection and instance segmentation. (§ 6.1 and § 6.4 of the main paper)
- Appendix H: detailed results on LVIS v0.5. (§ 6.2 and § 6.4 of the main paper)
- Appendix I: detailed results on LVIS v1.0. (§ 6.4 of the main paper)
- Appendix J: qualitative results of object detection on LVIS v0.5. (§ 6.2 of the main paper).

A. Contribution and Novelty

Our main contributions are in the idea of using object-centric images (OCI) to facilitate long-tailed object detection on scene-centric images (SCI) as well as a concrete implementation of this idea that is both simple and effective. This is by no means trivial; for instance, a related work [15] with a more sophisticated approach can hardly improve the accuracy (Table 5 of the main paper). While most existing works focus on *designing new algorithms* to learn from long-tailed data, our proposal is orthogonal to them, and can be combined together for further improvement.

Although leveraging auxiliary data to improve *common* object detection has been studied previously, existing works typically assume access to well prepared data from a similar visual domain, with sufficient object instances. However, collecting and annotating such auxiliary data is extremely challenging in *long-tailed* object detection. In contrast, our method does not have such a limitation as we make use of *object-centric* images readily available over the Internet (via search engines), which contains sufficient object instances though in a slight different domain. Particularly, we observe that making use of such rich *object-centric* images (from ImageNet) leads to more superior empirical performances against [15], which uses YFCC-100M [22].

To enable more general applicability, we make the design of our framework as straightforward as possible. Along this process, two challenges are identified, *i.e.*, the gap between visual domains and the lack of object labels. To address them, we investigate simple algorithms such as fixed box locations, mosaicking, and multi-stage training. We note that more sophisticated techniques can be incorporated as well. The facts that (a) *our framework performs on par with state-of-the-art long-tailed detection methods* and (b) *many existing techniques can be easily plugged into our framework* further justify the potential of this promising direction.

While several components of our framework — mosaic, pseudo-labeling, two-stage fine-tuning — have been individually explored in prior works in different contexts, *a suitable combination is essential and novel* for our idea to work. Further, our use of mosaic on OCI is different from [1, 3], as shown in Figure A. Our contributions also include extensive analysis that justifies the importance of each component. These insights led to a simple and effective

*Equal contributions

Figure A. **Different stitching methods.** MOSAICOS introduces *more diverse* examples by leveraging object-centric images, while existing methods [1, 3] only perform data augmentation using scene-centric images.

framework, which we consider a strength. For example, our LORE approach (§ B.2) could have provided methodological novelty. But its small gain over simple fixed locations does not justify the inclusion of it into our final framework.

B. Pseudo-Label Generation

B.1. Trust the calibrated detector and image labels

We provide analysis on pseudo-label generation with detector calibration and imputation using image class labels.

Detector calibration. As mentioned in § 5.1 of the main paper, we calibrate the pre-trained detector by assigning each class a different confidence threshold according to the class size — rare classes have lower thresholds. Figure B illustrates the difference with and without detector calibration, and with and without imputation using the image class labels. By assigning each class a different confidence threshold, the calibrated detector outputs more detected boxes, indicating that many rare and common objects are missed by the pre-trained detector due to low confidence scores (Figure B (a) vs. (c)). However, simply applying calibration can hardly correct the wrong labels that have already been biased toward the frequent classes (blue boxes in Figure B (c)). Next, we explore the idea of bringing the best of image class labels to correct noisy detected labels.

The importance of imputation with image class labels. For object-centric images, most of the object instances belong to the image's class label. We therefore improve "trust the pre-trained detector" (Figure B (a)) and "trust the calibrated detector" (Figure B (c)) by assigning each box the image class label (see Figure B (b) and (d)). As shown in Figure B and Table A, we see significant improvements for both the pre-trained and calibrated detectors. Specifically, assigning the image class label for each box can largely boost the performance for rare objects (AP^b_r).

B.2. Details on LORE

Figure C shows the pipeline of localization by region removal (LORE), which is introduced in § 5.1 of the main paper for pseudo-label generation. Concretely, LORE takes an object-centric image as the input and identifies the locations of the target object (*i.e.*, that of image label) in the image. The whole pipeline consists of three major components: (1) classifier training, (2) box pre-filtering, and (3)

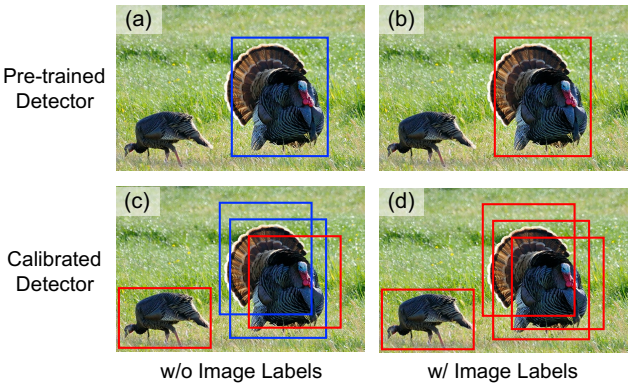

Figure B. **A comparison of pseudo-label generation with detector calibration and imputation using image class labels.** (a) trust the detector (D), (b) trust the detector + image class labels (D†), (c) trust the calibrated detector, and (d) trust the calibrated detector + class image labels (D‡). The image label is "turkey", a rare class in LVIS. Red/Blue boxes are labeled as "turkey"/other classes. See § 5.1 of the main paper for details.

Table A. **Results with different pseudo-labels.** We use ImageNet-21K as the source of object-centric images and report the results of object detection on LVIS v0.5 val. **Detector**: object detector used for generating pseudo-label bounding boxes; **CL:** assign each box the image Class Label instead of the predicted class label.

	Detector	CL	AP^b	AP^b_r	AP^b_c	AP^b_f
Faster R-CNN⋆	–	–	23.35	12.98	22.60	28.42
MOSAICOS	Pre-trained	✗	23.04	13.93	21.51	28.14
	Pre-trained	✓	24.66	17.45	**23.62**	28.83
	Calibrated	✗	24.03	13.13	23.51	**29.04**
	Calibrated	✓	**24.93**	**19.31**	23.51	28.95

localization by removal. We describe each step as follows.

Classifier training. We train a ResNet-50 [9] image classifier with all object-centric images. For LVIS v0.5 dataset, we follow the conventional training procedure[1] to train a 1,230-way ResNet classifier. Specifically, we train the networks with 90 epoch and achieve 74% top-1 training accuracy. We use this pre-trained classifier to rank object regions in object-centric images.

Box pre-filtering. We feed an object-centric image into the pre-trained *object detector* and collect detection results. Concretely, we take the top 300 detected boxes of Faster R-CNN [17] and drop each box's predicted class label. Next, we apply non-maximum suppression (NMS) over all the 300 boxes using a threshold of 0.5 to remove highly-overlapped ones. Basically, we trust the detected box locations (*i.e.*, they do contain objects), but will recheck which of them belongs to the target object.

To further reduce the number of candidate boxes, we sort

[1] https://github.com/pytorch/examples/tree/master/imagenet

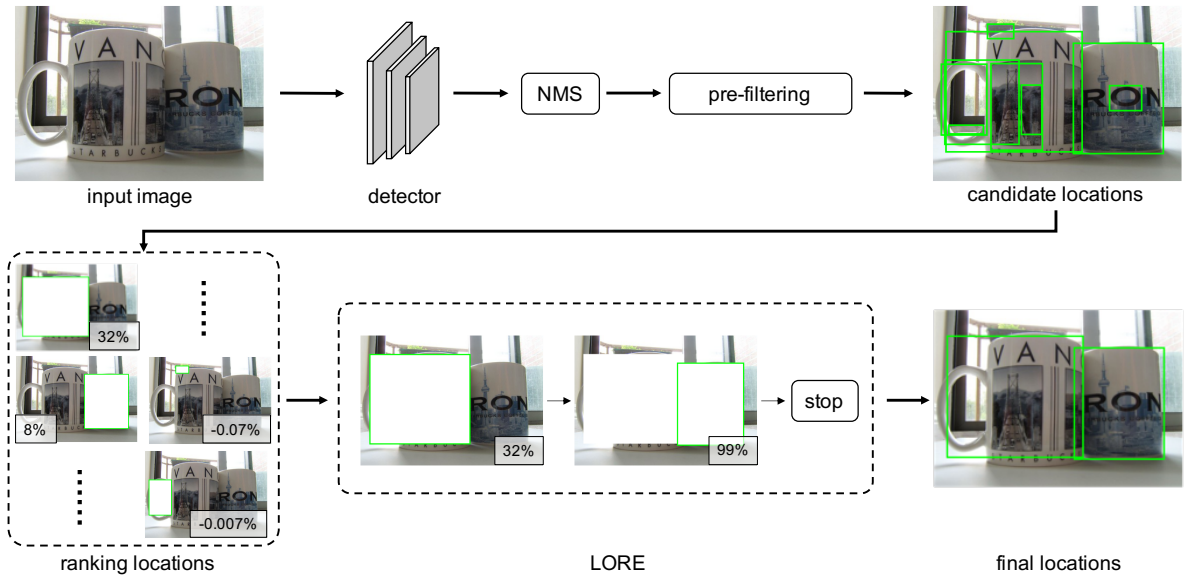

Figure C. **Illustration of LORE.** We first apply a pre-trained detector to obtain candidate boxes, followed by pre-filtering. We then sort the remaining boxes using an image classifier. Finally, we remove the boxes in turn until the classifier fail to predict the target image label. The numbers at image corners indicate the confidence reducing ratio. Negative values mean the confidence increases after removing outliers.

the boxes by their initial detection confidence (in the descending order) and then remove the corresponding regions from the image *in turn*[2], every time followed by applying the image classifier to the resulting image. We stop this process until the classification confidence of the target class goes below a certain threshold. We then collect the removed box locations, which together have likely covered the target objects (high recall, but likely low precision), to be the candidate box pool for the next step.

Localization by removal. To accurately identify which candidate truly belongs to the target class, we *re-rank* the candidates by how much removing each boxed region *alone* reduces the image classifier's confidence on the target class. We then follow the descending order to remove these boxed regions *in turn* until the classifier fail to predict the target class or the *confidence reducing ratio*[3] achieves a certain threshold. Finally, the bounding boxes of the removed regions are collected as the pseudo ground-truths for the image. More examples can be found in Figure D.

B.3. Discussion on fixed locations vs. LORE

Both fixed locations and LORE use accurate image class labels. Even though LORE gives more accurate object locations (see Figure D) in pseudo-label generation, the resulting detector with fixed location is just slightly worse than that with LORE. We attribute this small gap partially to two-stage fine-tuning, which adapts the detector back to accurately labeled scene-centric images. As shown in Table B, LORE notably surpasses fixed locations if we apply single-stage fine-tuning.

Table B. **Fixed locations vs. LORE.** We report object detection results on LVIS v0.5 val. **P-GT**: ways to generate pseudo-labels.

	P-GT	AP^b	AP^b_r	AP^b_c	AP^b_f
Single-stage	Fixed	20.09	12.96	19.08	24.20
	LORE	21.44	14.95	20.74	24.91
MOSAICOS	Fixed	24.75	19.73	23.44	28.39
	LORE	24.83	20.06	23.25	28.71

B.4. Discussion on pseudo-label generation

In this subsection, we discuss multiple ways for generating pseudo-labels in object-centric images. From the viewpoint of teacher models (*i.e.,* the pre-trained detector learned from a long-tailed distribution), we found that (1) the pre-trained detector is biased toward head classes, missing many accurate rare class predictions which have lower confidence scores; (2) detector calibration is useful to discover more bounding boxes for rare and common objects but can hardly correct wrong predicted labels. Our observations share the similar insights with a recent study [4] on large-vocabulary object detection.

From the other viewpoint of fine-tuning with pseudo scene-centric images, we found that imputation using image class labels leads to a notable performance gain regardless of inaccurate box locations (*e.g.*, fixed box locations). This is probably due to two reasons. First, dense boxes

[2]We crop out the corresponding image regions and replacing them by gray-color patches.

[3]We define the *confidence reducing ratio* as the relative confidence drop on the target class label before and after removing boxes.

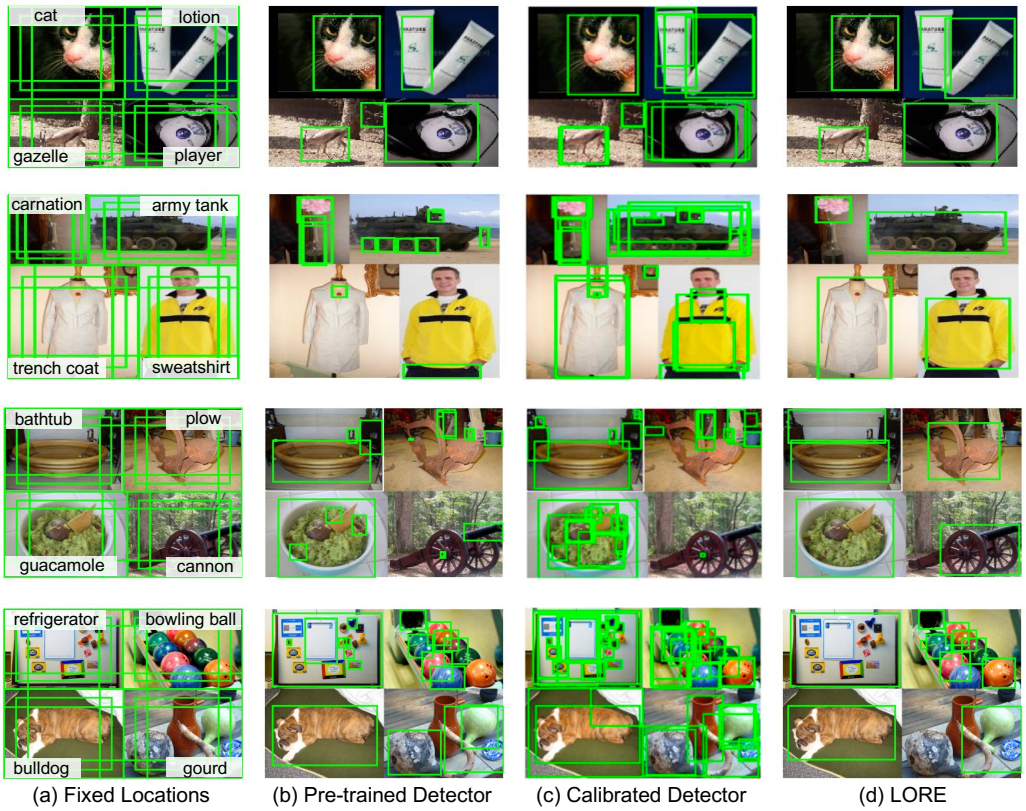

Figure D. **Box locations of different pseudo-label generation methods.** We show (a) fixed locations, (b) trust the pre-trained detector, (c) trust the calibrated detector, and (d) localization by region removals (LORE). The green boxes are the pseudo ground-truth locations found on each object-centric image alone before multiple images are stitched together. We can see that LORE accurately locates the target object in each sub-image while detection results are much noisy. Image class labels are listed on the corner of each sub-image in column (a).

(like six fixed locations) can be treated as data augmentation for training the object detector. Second, our two-stage fine-tuning is beneficial in learning with noisy data, *i.e.,* first on noisy pseudo scene-centric images and then on the clean labeled data from LVIS.

Other possibilities for pseudo-label generation include (1) iteratively improving the teacher detector by noisy student learning [29] and (2) calibrating the detector with more advanced approaches for class-imbalanced semi-supervised learning [26], etc.

C. Additional Ablation on Image Mosaicking

Does mosaicking more images help? In this section, we investigate the effect of different types of layouts for stitching object-centric images, *i.e.*, 1×1 (which is the original object-centric image), 2×2 mosaic, and 3×3 mosaic. We evaluate them under the same experimental settings: we use ImageNet-21K as the source of object-centric images ($1,016$ classes) and stitch images from the same class and use the 6 fixed locations as pseudo ground-truths. Table C shows the comparison of object detection results on LVIS v0.5 dataset. We see that 2×2 and 3×3 mosaics perform similarly and both outperform the 1×1 OCI (on AP^b and AP^b_r). An example with different layouts of 2×2 and 3×3 mosaics is shown in Figure E.

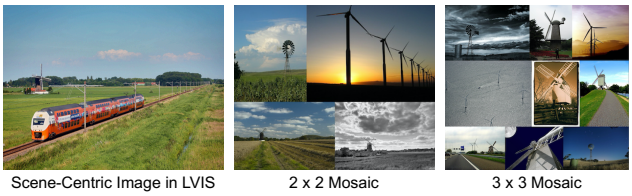

Figure E. **Different layouts of mosaics.** We show different types of mosaics from the same category ("windmill"). The 2×2 mosaic image (middle) and the real scene-centric image (left) in the LVIS dataset look alike in terms of appearance and structure while the 3×3 mosaic image (right) is much crowded.

D. Further Analysis on Self-training

We show detailed comparison results of self-training baseline in Table D to further demonstrate the effectiveness of the mosaicking and two-stage fine-tuning in our MOSAICOS framework. We follow the self-training method

Table C. **Comparison of different types of mosaic images.** Here we use ImageNet-21K as the source of object-centric images and stitch images from the *same* class and use the 6 fixed locations as pseudo ground-truths. 1×1 OCI means directly using the original object-centric images. Results are reported on LVIS v0.5 val. We can see that 2×2 mosaic gives better performance on all classes. The best result per column is in bold font.

	AP^b	AP^b_r	AP^b_c	AP^b_f
Faster R-CNN⋆	23.35	12.98	22.60	28.42
1×1 OCI	24.27	16.97	**23.29**	**28.42**
3×3 Mosaic	24.29	18.14	23.13	28.21
2×2 Mosaic	**24.48**	**18.76**	23.26	28.29

with the normalization loss in [31].

Mosaicking is also beneficial for self-training. We first study the vanilla self-training that directly learns object-centric (without mosaicking) and scene-centric images jointly. Specifically, we apply the pre-trained detector to generate pseudo-labels on the object-centric images (D). Next, the pre-trained detector is trained to jointly optimize the losses on human labels from LVIS and pseudo labels on object-centric images. We compare with and without image mosaic in Table D: image mosaicking improves AP^b/AP^b_r from $22.00/14.04$ to $22.71/14.52$, demonstrating the effectiveness of mosaicking object-centric images to mitigate the domain discrepancy between two types of images.

Self-training vs. our two-stage fine-tuning. To further improve the performance of self-training, we apply "trusted the calibrated detector + image class labels" (D‡) as the pseudo-labeling method, which leads to a much higher detection accuracy than "trusted the pre-trained detector" (D) for our MOSAICOS (cf. Table 1 of the main paper and the last row vs. the first row in Table D). With this pseudo-labeling method, we see a notable gain against "trust the pre-trained detector" (D) for self-training.

We further compare the self-training procedure that fine-tunes the detector simultaneously with object-centric and scene-centric images to our MOSAICOS with two-stage fine-tuning (again in Table D). MOSAICOS outperforms self-training (with either D or D‡) in most metrics, demonstrating the strength of two-stage fine-tuning which first learns with object-centric images and then scene-centric images. This two-stage pipeline is not only robust to noisy pseudo scene-centric data but also able to tie the detector to its final application domain with real scene-centric images.

E. Data Quality of Object-Centric Images

Our main results are based on the ImageNet dataset [5]. We included Google/Flickr images (Table 5 in the main text) mainly to analyze the effect of data quality and compare to [15]. As shown in Figure F, most Google images searched by object names are object-centric, even for those not ranked on the top. Following the experimental setup

Table D. **Comparison to self-training. Mosaic:** ✓means 2×2 image mosaicking from different classes. **P-GT:** ways to generate pseudo-labels (**D**: trust the pre-trained detector, **D**‡: trust the calibrated detector and image class label).

	Mosaic	P-GT	AP^b	AP^b_r	AP^b_c	AP^b_f
Faster R-CNN⋆	–	–	23.35	12.98	22.60	28.42
Self-training	✗	D	22.00	14.04	20.41	27.18
	✓	D	22.71	14.52	21.41	27.61
	✓	D‡	23.65	16.30	22.55	27.96
MOSAICOS	✓	D	23.04	13.93	21.51	28.14
	✓	D‡	24.93	19.31	23.51	28.95

in Table 5 of the main paper, we further experiment with 500 Google images per class: AP^b is improved from 24.45 to 24.63. For images that are less object-centric, LORE can give better pseudo-labels than the fixed heuristic; our two-stage fine-tuning is robust to noise. Moreover, there are extensive works on de-noising web data that we can leverage to further improve our scalability and applicability. That being said, we neither focus on web images/crowd-sourcing nor suggest that human efforts (*e.g.*, ImageNet) are not needed. Our claim is that rare objects that are hard to collect from SCI are easier to collect from OCI, which opens up a new way to tackle long-tailed object detection.

F. Comparison to Adversarial Training

We apply adversarial training (*e.g.*, [6]) to jointly train the detector with LVIS and pseudo scene-centric images. Concretely, we train an additional domain classifier to differentiate the LVIS images and pseudo scene-centric images, and incorporate a gradient reversal layer (GRL) [6] to minimize the discrepancy between their features to overcome the domain gap. We show comparisons in Table E. Adversarial training outperforms naive joint (*i.e.*, single-stage) training, and MOSAICOS (with two-stage fine-tuning using each source) surpasses adversarial training.

Table E. **Comparison to adversarial training.** Results are reported on LVIS v0.5 validation set.

	Mosaic	AP^b	AP^b_r	AP^b_c	AP^b_f
Single-stage	✓	20.09	12.96	19.08	24.20
Adv. training	✗	20.92	11.42	19.23	26.85
	✓	22.87	14.48	22.02	27.28
MOSAICOS	✗	24.27	16.97	23.29	**28.42**
	✓	**24.75**	**19.73**	**23.44**	28.39

G. Implementation Details of MOSAICOS

G.1. Details on object detection

As mentioned in § 6.1 of the main paper, we use Faster R-CNN [17] as our base detector and further extend the training process with another 90K iterations and select the checkpoint with the best AP^b as Faster R-CNN⋆. We use

Figure F. **Google Images for the most rare classes in LVIS.** We show the top 5 retrieved images and images ranked around 500.

Faster R-CNN⋆ as our main baseline to ensure that the improvement of MOSAICOS does not simply come from training (*i.e.*, fine-tuning) with more epochs.

For MOSAICOS, we first fine-tune Faster R-CNN⋆ with pseudo scene-centric images, and then fine-tune it with the LVIS training set again. Both stages are trained end-to-end with stochastic gradient descent with all training losses in Equation 1 of the main paper, using a mini-batch size of 16, momentum of 0.9, weight decay of 10^{-4}, and learning rate of 2×10^{-4}. *Unlike other long-tailed methods [19, 20, 23][4], there is no additional hyper-parameter in our framework.*

G.2. Details on instance segmentation

Background on instance segmentation. We apply Mask R-CNN [8], which adopts the two-stage network architecture similar to Faster R-CNN [17], with an identical first stage RPN. In the second stage, in addition to predicting the class label and box offset, Mask R-CNN further outputs a binary segmentation mask for each proposal. Formally, during training, the entire Mask R-CNN is learned with four loss terms

$$\mathcal{L} = \mathcal{L}_{\text{rpn}} + \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{reg}} + \mathcal{L}_{\text{mask}}, \tag{A}$$

where the RPN loss $\mathcal{L}_{\text{rpn}}$, classification loss $\mathcal{L}_{\text{cls}}$, and box regression loss $\mathcal{L}_{\text{reg}}$ are identical to those defined in [17]. The mask loss $\mathcal{L}_{\text{mask}}$ is learned via an average binary cross-entropy objective.

Multi-stage training for instance segmentation. We first train a Mask R-CNN using labeled scene-centric images from LVIS with instance segmentation annotations [7]. All the fours loss terms in Equation A are optimized.

We then fine-tune the model using the pseudo scene-centric images that are generated from object-centric images. We use these images (only with box pseudo-labels) to fine-tune the model using $\mathcal{L}_{\text{cls}}$, $\mathcal{L}_{\text{rpn}}$, and $\mathcal{L}_{\text{reg}}$. In other words, we do not optimize $\mathcal{L}_{\text{mask}}$. Any network parameters that affect $\mathcal{L}_{\text{cls}}$, $\mathcal{L}_{\text{rpn}}$, and $\mathcal{L}_{\text{reg}}$, especially those in the backbone feature network (except the batch-norm layers), can be updated.

[4]Both EQL(v2) [19, 20] and Seesaw loss [23] introduce (multiple) additional hyper-parameters.

Table F. **Object detection on LVIS v0.5**. We use ImageNet + Google Images. MSCOCO: for pre-training. [16]: balanced loss. Within each column, red/blue indicates the best/second best.

	MSCOCO	[16]	AP^b	AP^b_r	AP^b_c	AP^b_f
RFS [7]			23.35	12.98	22.60	28.42
EQL [20]			23.30	–	–	–
LST [10]			22.60	–	–	–
BaGS [12]	✓		25.96	17.65	25.75	29.54
TFA [25]			24.40	16.90	24.30	27.70
MOSAICOS			25.01	20.25	23.89	28.32
	✓		26.28	17.37	26.13	30.02
		✓	26.83	21.00	26.31	29.81
	✓	✓	28.06	19.11	28.23	31.41

After this stage, we fine-tune the whole network again with labeled scene-centric images from LVIS, using all the four loss terms in Equation A. The training procedure and other implementation details for instance segmentation are exactly the same as object detection in § G.1.

H. Experimental Results on LVIS v0.5

Due to space limitations, we only compared with state-of-the-art methods in Table 3 and Table 7 of the main paper. In this section, we provide detailed comparisons with more previous works on LVIS v0.5.

Object detection on LVIS v0.5. There are not many papers reporting detection results on LVIS. In Table F, we further include EQL [20] and LST [10], together with BaGS [12] and TFA [25], as the compared methods. MOSAICOS outperforms all baselines except BaGS [12]. We note that, BaGS is pre-trained on COCO [13] while MOSAICOS is initialized from ResNet-50 that is pre-trained on ImageNet-1K (ILSVRC). By using the COCO pre-trained backbone as the initialization, MOSAICOS outperforms BaGS on nearly all metrics. Moreover, when combined with [16], MOSAICOS can further boost the state-of-the-art performance.

Instance segmentation on LVIS v0.5. The comparison results on LVIS 0.5 instance segmentation are presented in Table G, including the baseline models with RFS [7] for re-sampling, EQL(v2) [19, 20] for re-weighting, LST [10] for incremental learning, SimCal [24] and BaGS [12] for de-coupled training, Forest R-CNN [27] for hierarchy clas-

Table G. **Instance segmentation on LVIS v0.5.** Our MOSAICOS uses images from ImageNet and Google Images. + [16]: include the balanced loss in the second stage fine-tuning. Within each column, red/blue indicates the best/second best.

	AP	AP_r	AP_c	AP_f
RFS [7]	24.38	15.98	23.96	28.27
EQL [20]	22.80	11.30	24.70	25.10
LST [10]	23.00	–	–	–
SimCal [24]	23.40	16.40	22.50	27.20
Forest RCNN [27]	25.60	18.30	26.40	27.60
BaGS [12]	26.25	17.97	26.91	28.74
BALMS [16]	27.00	19.60	28.90	27.50
EQL v2 [19]	27.10	18.60	27.60	29.90
MOSAICOS	26.26	19.63	26.60	28.49
MOSAICOS + [16]	27.86	20.44	28.82	29.62

sification, and BALMS [16] for a balanced softmax loss. MOSAICOS can perform on a par with or even better than the compared methods without any additional hyper-parameter tuning like in [19, 20, 23]. By combined with [16], MOSAICOS achieves the stat-of-the-art performance of $27.86/20.44$ AP/AP_r, showing the compatibility of MOSAICOS. We expect that MOSAICOS could be further improved by incorporating other long-tailed learning strategies [12, 16, 21, 23, 25].

I. Experimental Results on LVIS v1.0

I.1. Setup

Dataset statistics. We further evaluate MOSAICOS on LVIS v1.0 [7]. The total dataset size has been expanded to ∼160K images and ∼2M instance annotations. The total number of categories has decreased slightly (from 1,230 to 1,203) due to a more stringent quality control. More specifically, LVIS v1.0 adds 52 new classes while drops 79 classes from LVIS v0.5. The validation set has been expanded from 5K images to 20K images. Table I gives a summary of the statistics of the two versions of LVIS dataset. We follow the experimental setups of LVIS v0.5 to use category synset ID [14] to search for the corresponding classes in ImageNet-21K dataset [18]. In total, we collect $753,700$ object-centric images. Table H shows the detailed statistics of the number of overlapped classes in those datasets. We also search 100 images for each class via Google Images.

Our settings. For instance segmentation, we use Mask R-CNN [8] with instance segmentation annotations. The training scheme is the same as that for Faster R-CNN in object detection. Specifically, we follow the default training configurations in [28] with 1x schedule[5].

For the MOSAICOS training (cf. § G.2), we first fine-tune the baseline Mask R-CNN for 90K iterations with pseudo scene-centric images using only box annotations. Our pseudo scene-centric images are synthesized with 2×2 mosaic from random classes of ImageNet-21K and Google images. We use the boxes with 6 fixed locations as pseudo ground-truths. After that, We end-to-end fine-tune the entire model for another 90K iterations using the LVIS training set with all four losses. The network parameters of the mask head are initialized by the baseline Mask RCNN model. Both two fine-tuning steps are trained with stochastic gradient descent with a mini-batch size of 16, momentum of 0.9, weight decay of 10^{-4}, and learning rate of 2×10^{-4}.

[5]EQL v2 [19] and Seesaw loss [23] use another implementation from [2], which uses 2x schedule for training the models on LVIS v1.0.

I.2. Instance segmentation on LVIS v1.0

Table J shows detailed results on instance segmentation. We mainly compare with Mask R-CNN and two recent papers [19, 23], which reported instance segmentation results and re-implemented some other methods on LVIS v1.0. We evaluate MOSAICOS with three different backbone models: ResNet-50 [9], ResNet-101 [9], and ResNeXt-101 [30]: MOSAICOS consistently outperforms the Mask R-CNN baseline especially for rare classes.

We note that, EQL v2 [19] and Seesaw loss [23] were implemented by a different framework [2] and reported results with a stronger 2x training schedule. *Thus, the accuracy gap between different methods may be partially affected by these factors.* This can be seen by comparing the three Mask R-CNN results with ResNet-50 and the two Mask R-CNN results with ResNet-101: there is a notable difference in their accuracy. Specifically, the ones reported by [23] have a much higher accuracy.

With the same ResNet-50 backbone and 1x schedule, MOSAICOS achieves $24.49/18.30$ AP/AP_r, better than EQL v2 [19] ($23.70/14.90$), BaGS [12], and cRT [11]. With the ResNet-101 backbone, MOSAICOS with 1x schedule achieves 26.54 AP, outperforming both EQL [20] (2x schedule, 26.20 AP) and BaGS [12] (2x schedule, 25.80 AP). We also show a detailed comparison to Seesaw loss [23] in Table K. MOSAICOS demonstrates a comparable performance gain against the Mask R-CNN baseline.

J. Qualitative Results

We show qualitative results on LVIS v0.5 object detection in Figure G and Figure H. We compare the ground truth, the results of the baseline and of our method.

We observe that our method can accurately recognize more objects from rare categories that may be overlooked by the baseline detector. For example, as shown in Figure H, MOSAICOS correctly detects giant panda, scoreboard, horse carriage, and diaper. They are all rare classes and the baseline detector fails to make any correct detection (*i.e.*, localization and classification) on them. Moreover, the results demonstrate that MOSAICOS is able to correct the prediction labels that were wrongly classified to fre-

Table H. **Number of overlapped classes in LVIS and ImageNet.** In LVIS and ImageNet, each category can be identifed by a unique WordNet synset ID. We match LVIS classes to ImageNet ones and show the number of the overlapped classes. Specifically, we show # LVIS classes / # overlapped to ImageNet-21K / # overlapped to ImageNet-1K (ILSVRC).

Version	Split	Frequent	Common	Rare	Overall
v0.5	Train	315 / 253 / 85	461 / 387 / 96	454 / 385 / 71	1230 / 1025 / 252
	Val	313 / 252 / 21	392 / 329 / 84	125 / 106 / 71	830 / 678 / 176
v1.0	Train	405 / 331 / 87	461 / 390 / 96	337 / 277 / 64	1203 / 998 / 247
	Val	405 / 331 / 87	452 / 382 / 92	178 / 144 / 37	1035 / 857 / 216

Table I. **Statistics of LVIS v0.5 and v1.0 datasets.**

Version	Type	Train	Val	Test
v0.5	# Image	57,263	5,000	19,761
	# Class	1,230	830	-
	# Instance	693,958	50,763	-
v1.0	# Image	100,170	19,809	19,822
	# Class	1,203	1,035	-
	# Instance	1,270,141	244,707	-

Table J. **Instance segmentation on LVIS v1.0.** We list multiple Mask R-CNN baselines whose accuracy are notably different due to differences in implementation, which may affect the accuracy of the corresponding proposed methods. MOSAICOS outperforms Mask R-CNN and many other methods on most of the metrics.

Backbone	Method	AP	AP_r	AP_c	AP_f
R-50	Mask RCNN [7]$^{\dagger 1}$	22.59	12.31	21.30	28.55
	Mask RCNN [7]$^{\star 2}$	23.70	13.50	22.80	29.30
	Mask RCNN [7]$^{\S 1}$	22.20	11.50	21.20	28.00
	cRT [11]$^{\S 1}$	22.10	11.90	20.20	29.00
	BaGS [12]$^{\S 1}$	23.10	13.10	22.50	28.20
	EQL v2 [19]$^{\S 1}$	23.70	14.90	22.80	28.60
	EQL v2 [19]$^{\S 2}$	25.50	17.70	24.30	30.20
	Seesaw [23]$^{\star 2}$	26.40	19.60	26.10	29.80
	MOSAICOS $^{\dagger 1}$	24.49	18.30	23.00	28.87
R-101	Mask RCNN [7]$^{\dagger 1}$	24.82	15.18	23.71	30.31
	Mask RCNN [7]$^{\star 2}$	25.50	16.60	24.50	30.60
	EQL [20]$^{\star 2}$	26.20	17.00	26.20	30.20
	BaGS [12]$^{\star 2}$	25.80	16.50	25.70	30.10
	Seesaw [23]$^{\star 2}$	28.10	20.00	28.00	31.90
	MOSAICOS$^{\dagger 1}$	26.77	20.79	25.76	30.53
X-101	Mask RCNN [7]$^{\dagger 1}$	26.62	17.51	25.51	31.86
	MOSAICOS$^{\dagger 1}$	28.31	21.74	27.25	32.36

†: Our implementations with RFS [7].
*: Results reported in [23]. All models trained with RFS [7].
§: Results reported in [19].
^{1}: 1x schedule. ^{2}: 2x schedule.

quent classes without sacrificing the detection performance on common and frequent classes. As shown in the second row of Figure G, the baseline detector wrongly predicts frequent class labels like bowl and knife with high confidence score, while MOSAICOS suppresses them and successfully predicts rare classes napkin and cappuccino.

One characteristic of LVIS is that the objects may not be exhaustively annotated in each image. We find that MOSAICOS still detects those objects which are not labeled as the ground truths. In the second and third row of Figure H, the predictions on banner and horse are obviously correct while LVIS doesn't have annotations on them.

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 1, 2

[2] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 7

[3] Yukang Chen, Peizhen Zhang, Zeming Li, Yanwei Li, Xiangyu Zhang, Gaofeng Meng, Shiming Xiang, Jian Sun, and Jiaya Jia. Stitcher: Feedback-driven data provider for object detection. *arXiv preprint arXiv:2004.12432*, 2020. 1, 2

[4] Achal Dave, Piotr Dollár, Deva Ramanan, Alexander Kirillov, and Ross Girshick. Evaluating large-vocabulary object detectors: The devil is in the details. *arXiv preprint arXiv:2102.01066*, 2021. 3

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 5

[6] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 17(1):2096–2030, 2016. 5

[7] Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 6, 7, 8, 9

[8] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017. 6, 7

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 7

[10] Xinting Hu, Yi Jiang, Kaihua Tang, Jingyuan Chen, Chunyan Miao, and Hanwang Zhang. Learning to segment the tail. In *CVPR*, 2020. 6, 7

[11] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decou-

Table K. **Comparisons of instance segmentation on LVIS v1.0.** MOSAICOS achieves comparable improvements against the Mask R-CNN baseline. We note that, Seesaw loss [23] uses a different implementation and training schedule (*i.e.*, 2x). Thus, the results may not be directly comparable.

Backbone	Schedule	Method	AP	AP_r	AP_c	AP_f
R-50	2x	Mask RCNN [7]	23.70	13.50	22.80	29.30
		Seesaw [23]	**(+2.70)** 26.40	**(+6.10)** 19.60	**(+3.30)** 26.10	**(+0.50)** 29.80
	1x	Mask RCNN [7]	22.59	12.31	21.30	28.55
		MOSAICOS	**(+1.90)** 24.49	**(+5.99)** 18.30	**(+1.70)** 23.00	**(+0.32)** 28.87
R-101	2x	Mask RCNN [7]	25.50	16.60	24.50	30.60
		Seesaw [23]	**(+2.60)** 28.10	**(+3.40)** 20.00	**(+3.50)** 28.00	**(+1.30)** 31.90
	1x	Mask RCNN [7]	24.82	15.18	23.71	30.31
		MOSAICOS	**(+1.95)** 26.77	**(+5.61)** 20.79	**(+2.05)** 25.76	**(+0.22)** 30.53

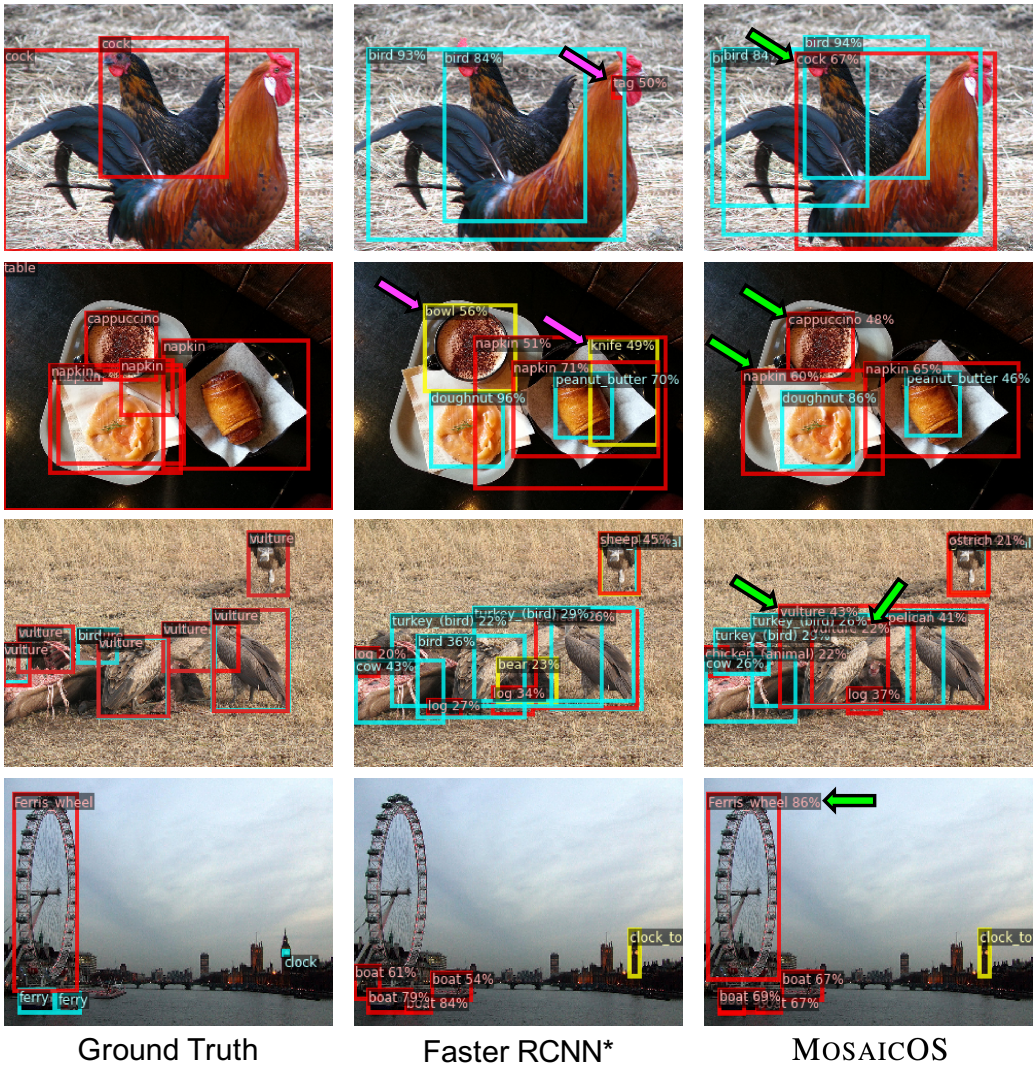

Figure G. **Qualitative results on object detection.** Our approach can detect the rare objects missed by the baseline detector (*e.g.*, *cock*, *cappuccino*, *ferris wheel*) and correct the labels that were wrongly classified to frequent categories (*e.g.*, *bear*, *knife*, *bowl*). We superimpose **green** arrows to show where we did right while the baseline did wrong (**magenta**). **Yellow**/**Cyan**/**Red** boxes indicate frequent/common/rare (predicted) class labels.

pling representation and classifier for long-tailed recognition. In *ICLR*, 2020. 7, 8

[12] Yu Li, Tao Wang, Bingyi Kang, Sheng Tang, Chunfeng Wang, Jintao Li, and Jiashi Feng. Overcoming classifier im-

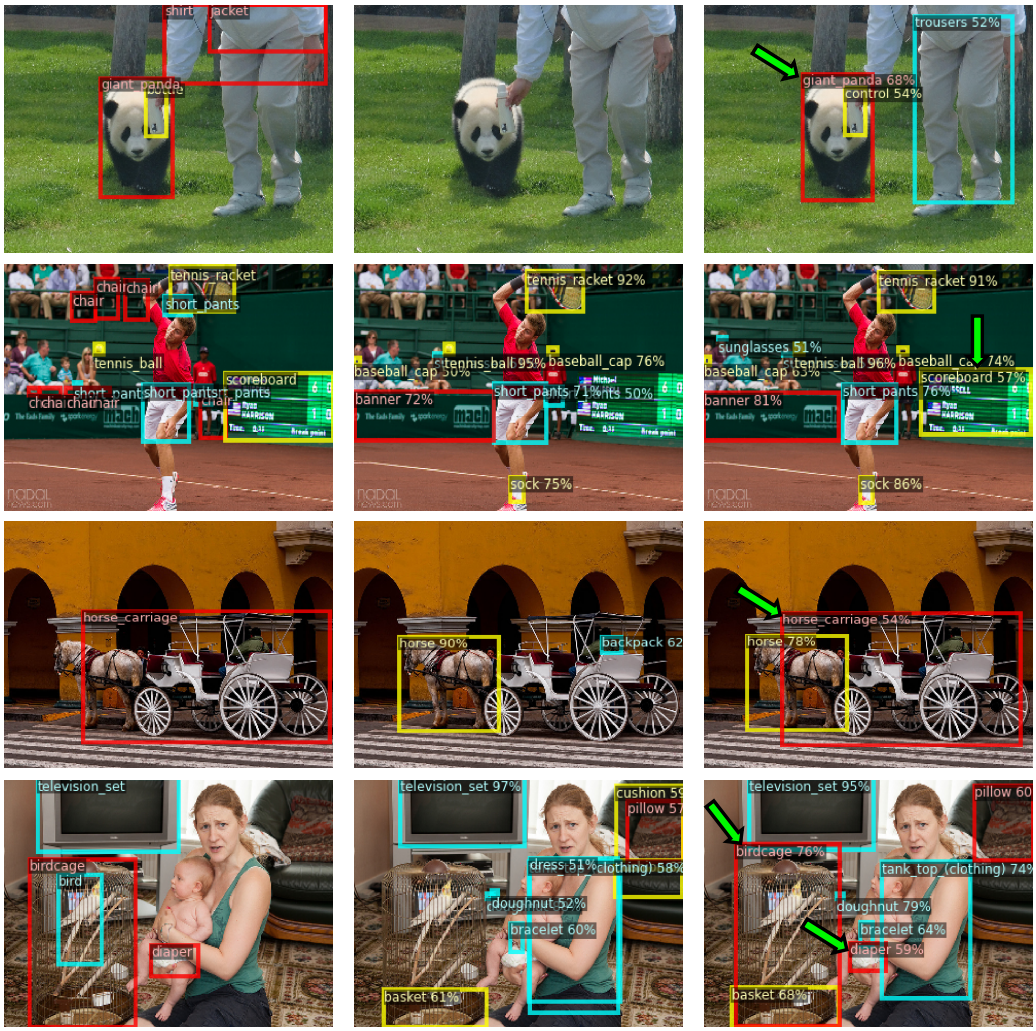

Figure H. **Additional qualitative results on object detection.** We superimpose **green** arrows to show that our approach can detect the objects missed by the baseline detector (*e.g.*, *gaint panda*, *scoreboard*, *horse carriage*, *birdcage*, *diaper*). **Yellow**/**Cyan**/**Red** boxes indicate frequent/common/rare (predicted) class labels.

balance for long-tail object detection with balanced group softmax. In *CVPR*, 2020. 6, 7, 8

[13] 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. 6

[14] George A Miller. WordNet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995. 7

[15] Vignesh Ramanathan, Rui Wang, and Dhruv Mahajan. DLWL: Improving detection for lowshot classes with weakly labelled data. In *CVPR*, 2020. 1, 5

[16] Jiawei Ren, Cunjun Yu, Shunan Sheng, Xiao Ma, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Balanced meta-softmax for long-tailed visual recognition. In *NeurIPS*, 2020. 6, 7

[17] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 2, 5, 6

[18] 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*, 115(3):211–252, 2015. 7

[19] 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. 6, 7, 8

[20] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *CVPR*, 2020. 6, 7, 8

[21] Jingru Tan, Gang Zhang, Hanming Deng, Changbao Wang, Lewei Lu, Quanquan Li, and Jifeng Dai. 1st place solution of lvis challenge 2020: A good box is not a guarantee of a

good mask. *arXiv preprint arXiv:2009.01559*, 2020. 7

[22] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *arXiv preprint arXiv:1503.01817*, 2015. 1

[23] 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. 6, 7, 8, 9

[24] Tao Wang, Yu Li, Bingyi Kang, Junnan Li, Junhao Liew, Sheng Tang, Steven Hoi, and Jiashi Feng. The devil is in classification: A simple framework for long-tail instance segmentation. In *ECCV*, 2020. 6, 7

[25] Xin Wang, Thomas E Huang, Trevor Darrell, Joseph E Gonzalez, and Fisher Yu. Frustratingly simple few-shot object detection. In *ICML*, 2020. 6, 7

[26] Chen Wei, Kihyuk Sohn, Clayton Mellina, Alan Yuille, and Fan Yang. Crest: A class-rebalancing self-training framework for imbalanced semi-supervised learning. *arXiv preprint arXiv:2102.09559*, 2021. 4

[27] Jialian Wu, Liangchen Song, Tiancai Wang, Qian Zhang, and Junsong Yuan. Forest R-CNN: Large-vocabulary long-tailed object detection and instance segmentation. In *ACM MM*, 2020. 6, 7

[28] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 7

[29] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *CVPR*, 2020. 4

[30] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 7

[31] Barret Zoph, Golnaz Ghiasi, Tsung-Yi Lin, Yin Cui, Hanxiao Liu, Ekin Dogus Cubuk, and Quoc Le. Rethinking pre-training and self-training. In *NeurIPS*, 2020. 5