ScaleDet: A Scalable Multi-Dataset Object Detector

Yanbei Chen, Manchen Wang, Abhay Mittal, Zhenlin Xu,
Paolo Favaro, Joseph Tighe, Davide Modolo
AWS AI Labs

A. Additional results

Additional results on O365 and OID. In Sec 4.3 Table 4, we follow the evaluation of Detic [6] and provide results on two training datasets: LVIS and COCO. As Detic is trained with text embeddings, it can also be tested on other datasets. In Table A, we compare ScaleDet and Detic on two additional datasets: Objects365 (O365), OpenImages (OID), along with LVIS and COCO, where both models use Swin-Base Transformer as backbone. For Detic, we use the best model "Detic_C2_SwinB_896_4x_IN-21K+COCO" from the Detic model zoo[1]. For ScaleDet, we use our best model in Table 2 in Sec 4.2. In Table A, it can be seen that our ScaleDet surpasses Detic with substantial margins, improving the mAP over all datasets from 42.7 to 56.6.

Model	LVIS	COCO	OID	O365	mAP
Detic-B [6]	44.3	50.1	21.6	54.6	42.7
ScaleDet-B	**50.7**	55.4	43.8	**76.2**	56.5
ScaleDet-B*	50.6	**58.8**	**46.8**	75.9	**58.0**

Table A. Comparison to Detic [6] on multi-dataset training using the best models from both papers. B: using Swin-Base Transformer as backbone. *: model is further fine-tuned on each dataset after multi-dataset training. mAP is the mean AP over all datasets.

Results of individual datasets on ODinW. In Table D, we provide the results on "Object Detection in the Wild" benchmark, which reports the mAP on individual datasets: AerialDrone, Aquarium, Rabbits, EgoHands, Mushrooms, Packages, PascalVOC, pistols, pothole, Raccoon, Shellfish, Thermal, Vehicles, as well as the mAP over 13 datasets.

Qualitative results. Due to space limit in our manuscript, we provide additional qualitative results on upstream and downstream datasets in Figure A and Figure B respectively.

B. Implementation details

Dataset details. We provide the licenses and websites of upstream and downstream datasets in Table B and Table E.

Unseen class labels in Figure 3 in Sec 4.2 are 12 class labels in downstream datasets: dock, jetski, lift, jellyfish, stingray, CoW, chanterelle, aeroplane, diningtable, motorbike, pottedplant, tvmonitor – which are not in upstream datasets.

Dataset	Website
LVIS	https://www.lvisdataset.org
COCO	http://cocodataset.org
Objects365	https://www.objects365.org
OpenImages	https://storage.googleapis.com/openimages/web/index.html

Table B. Dataset details on 4 upstream datasets. All datasets are under the "CC BY 4.0" license.

Model	Dataset(s)	Iterations	batch size	learning rate
baseline	L	90k	64	0.0002
	C	90k	64	0.0002
	O365	540k	64	0.0002
	OID	540k	64	0.0002
ScaleDet	L,C	90k	64	0.0002
	L,C,O365	400k	128	0.0002
	L,C,O365,OID	750k	128	0.0002

Table C. Learning schedules on different setups.

Training details. We provide our pseudo code in Algorithm A. We will release our pre-trained model weights if our paper is accepted. Our codebase is proprietary, so we could not provide it in supplementary materials. Similar to most state-of-the-art detectors, our implementation is built upon Detectron2 [4], so it is easy to reproduce our results based upon the codebase. In Eq. (8), we set the hyperparameter λ to 10 based on validation set. For the text encoder, we use "ViT-B/32" in CLIP [3][2] or "ViT-H/14" in OpenCLIP [1][3].

We detail the learning schedules in Sec 4.2 in the following for reproducibility. For Table 1, the learning schedules are in Table C. For Table 2, 4, 5, 6, 7, 8, the learning schedules are the same as the ones trained on the same datasets in Table C. For Table 3, we follow the same learning schedule as UniDet [7] to train on three datasets for fair comparison.

Complexity analysis. The label space of ScaleDet is computed before training and does not require additional computation cost for training. Compared to a standard detector using one loss for classification, ScaleDet uses two loss terms (Eq. (6), Eq. (7)) for classification. Eq. (6) has the same computation cost as standard classification loss, Eq. (7) introduces small computation cost with complexity $O(N)$, proportional to the batch size. At test time, the computation cost of ScaleDet is the same as a standard detector.

[1] https://github.com/facebookresearch/Detic/blob/main/docs/MODEL_ZOO.md

[2] https://github.com/openai/CLIP

[3] https://github.com/mlfoundations/open_clip

Model	Model Type	#Data	AerialDrone	Aquarium	Rabbits	EgoHands	Mushrooms	Packages	PascalVOC	pistols	pothole	Raccoon	Shellfish	Thermal	Vehicles	mAP
GLIP-T [2]	detection + understanding	5.5M	31.2	52.5	70.8	78.7	88.1	75.6	62.3	71.2	58.7	61.4	51.4	76.7	65.3	64.9
GLIPv2-T [5]		5.5M	30.2	52.5	74.8	80.0	88.1	74.3	66.4	73.0	60.1	63.7	54.4	63.0	83.5	66.5
GLIPv2-B [5]		20.5M	32.6	57.5	73.6	80.0	88.1	74.9	71.1	76.5	58.7	68.2	70.6	79.6	71.2	69.4
Detic-R [6]	detection + classification	12.6M	43.1	49.5	72.2	78.0	91.0	70.0	49.1	99.9	45.8	66.2	38.0	82.3	52.4	64.4
Detic-B [6]		12.6M	44.3	51.9	76.2	78.2	89.8	75.1	96.4	99.9	50.7	64.3	47.6	82.4	54.4	70.1
ScaleDet-R	detection	3.6M	42.6	49.2	72.3	78.1	89.8	75.7	91.8	99.9	47.4	67.5	40.0	82.1	54.5	68.5
ScaleDet-T		3.6M	46.0	52.1	73.1	79.2	91.6	76.5	95.1	99.9	50.1	65.6	47.8	83.1	54.7	70.4
ScaleDet-B		3.6M	44.8	53.5	75.1	78.4	91.0	76.5	95.7	99.9	51.7	67.6	55.0	83.4	60.7	**71.8**

Table D. Results of fine-tune transfer on 13 individual datasets on ODinW. R, T, B: ResNet50, Swin-Tiny, Swin-Base Transformer as backbone. Metric: mAP over 13 datasets. Note: "understanding", "classification" mean training with vision-and-language understanding, and classification datasets besides detection data.

Figure A. Qualitative results testing example images on upstream datasets: LVIS, COCO, Objects365 and OpenImages. It can be seen that on the same image (in column 1 and 2), the model can detect different classes. In different columns, the model is evaluated by setting the classifier to recognize a certain set of class labels from one dataset at test time.

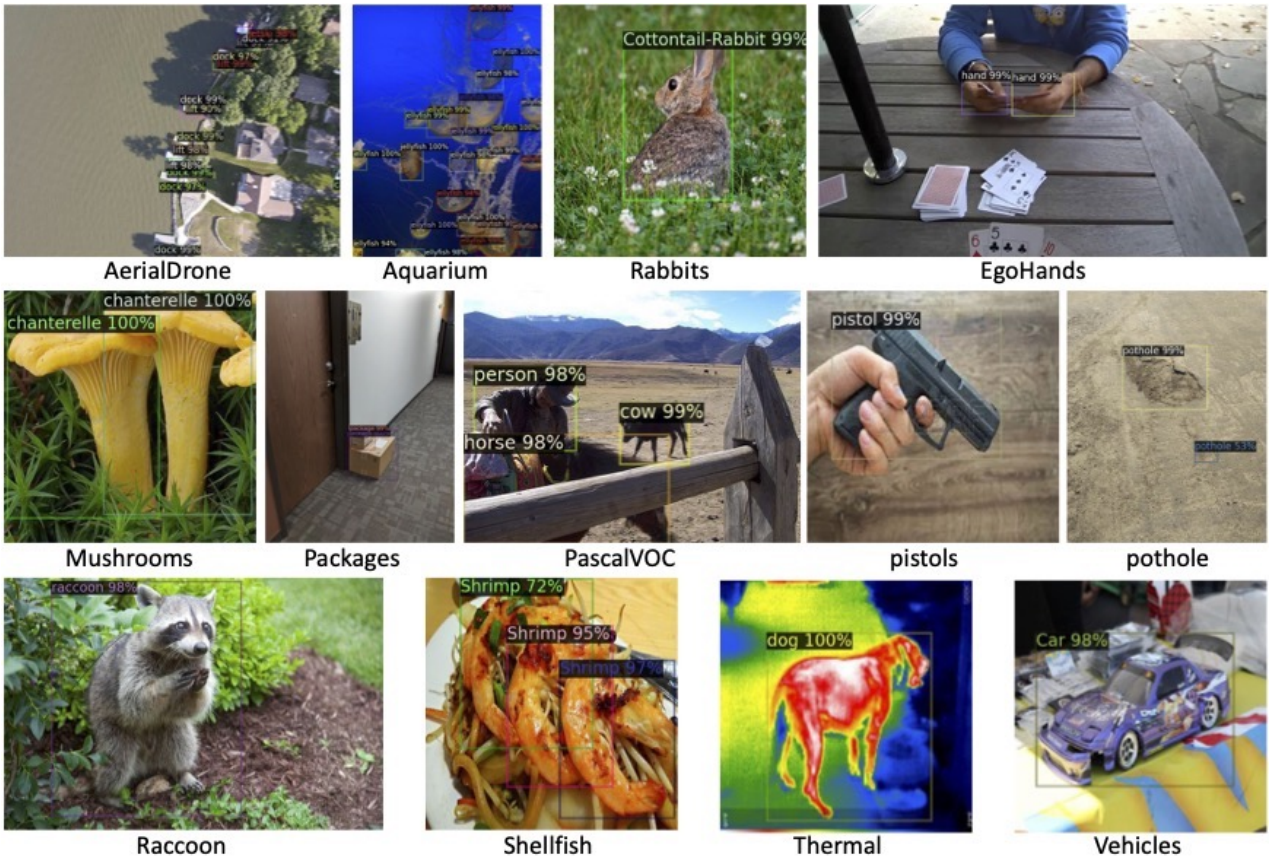

Figure B. Qualitative results testing example images on downstream datasets: AerialDrone, Aquarium, Rabbits, EgoHands, Mushrooms, Packages, PascalVOC, pistols, pothole, Raccoon, Shellfish, Thermal, Vehicles.

Dataset	License	Website
AerialMaritimeDrone	MIT	https://public.roboflow.com/object-detection/aerial-maritime
Aquarium	CC BY 4.0	https://public.roboflow.com/object-detection/aquarium
CottontailRabbits	CC BY 4.0	https://public.roboflow.com/object-detection/cottontail-rabbits-video-dataset
EgoHands	CC BY 4.0	https://public.roboflow.com/object-detection/hands
NorthAmericaMushrooms	Public Domain	https://public.roboflow.com/object-detection/na-mushrooms
Packages	Public Domain	https://public.roboflow.com/object-detection/packages-dataset
PascalVOC	CC BY 4.0	https://public.roboflow.com/object-detection/pascal-voc-2012
pistols	Public Domain	https://public.roboflow.com/object-detection/pistols
pothole	ODbL v1.0	https://public.roboflow.com/object-detection/pothole
Raccoon	MIT	https://public.roboflow.com/object-detection/raccoon
ShellfishOpenImages	CC BY 4.0	https://public.roboflow.com/object-detection/shellfish-openimages
thermalDogsAndPeople	Public Domain	https://public.roboflow.com/object-detection/thermal-dogs-and-people
VehiclesOpenImages	CC BY 4.0	https://public.roboflow.com/object-detection/vehicles-openimages

Table E. Dataset details on 13 downstream datasets from "Object Detection in the Wild" (ODinW-13) benchmark.

Algorithm A Scalable multi-dataset training recipe for object detection

Required: Datasets $D_1, D_2, ..., D_K$ and their concatenated label spaces $L = L_1 \oplus L_2 \oplus ... \oplus L_K$, which gives n class labels: $\{l_1, l_2, ..., l_n\}$.

Required: Text encoder from pre-trained CLIP [3] or OpenCLIP [1]. Compute the text embedding for each label with prompt engineering[4] which gives n text embeddings: $\{t_1, t_2, ..., t_n\}$ for the corresponding labels $\{l_1, l_2, ..., l_n\}$.

Required: An object detector θ_{detector} for training.

for $i \leftarrow 1$ to n **do**
  for $j \leftarrow 1$ to n **do**
    $\cos(t_i, t_j) = \frac{t_i \cdot t_j}{||t_i|| ||t_j||}$ ▷ Compute cosine similarity for text embeddings t_i, t_j
  end for
  $\alpha_i = \min\{\cos(t_i, t_j)\}_{j=1}^{n}, \beta_i = \max\{\cos(t_i, t_j)\}_{j=1}^{n} = \cos(t_i, t_i) = 1$ ▷ Compute normalization factors
  $\mathbf{s}_i = \text{sim}(l_i, l_j) = \frac{\cos(t_i, t_j) - \alpha_i}{\beta_i - \alpha_i}$ ▷ Normalize the similarity score within 0 and 1
end for

Input: A mini-batch of images $[I_1, I_2, ..., I_B]$, text embeddings $[t_1, t_2, ..., t_B]$ that represent their class labels $[y_1, y_2, ..., y_B]$.

for $b \leftarrow 1$ to B **do**
  $\{v_1, v_2, ..., v_j\} = \theta_{\text{detector}}(I_b)$ ▷ Obtain visual features from the detector for image I_b
  for $i \leftarrow 1$ to j **do**
    $\mathbf{c}_i = [\cos(v_i, t_1), \cos(v_i, t_2), ..., \cos(v_i, t_n)]$ ▷ Compute the visual-langague similarities: Eq. (5)
    $\mathcal{L}_{hl} = \text{BCE}(\sigma_{sg}(\mathbf{c}_i/\tau), l_i)$ ▷ Compute the hard label assignment loss: Eq. (6)
    $\mathcal{L}_{sl} = \text{MSE}(\mathbf{c}_i, \mathbf{s}_i)$ ▷ Compute the soft label assignment loss: Eq. (6)
    $\mathcal{L}_{lang} = \mathcal{L}_{hl} + \lambda \mathcal{L}_{sl}$ ▷ Compute the learning objective on visual feature v_i: Eq. (7)
  end for
  Backward propagation on the detector $\theta_{\text{detector}}(\cdot)$ with the overall loss on all visual features $\{v_1, v_2, ..., v_j\}$
end for

References

[1] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip. *https://doi.org/10.5281/zenodo.5143773*, 2021. 1, 3

[2] 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. 2

[3] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 1, 3

[4] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 1

[5] Haotian* Zhang, Pengchuan* Zhang, Xiaowei Hu, Yen-Chun Chen, Liunian Harold Li, Xiyang Dai, Lijuan Wang, Lu Yuan, Jenq-Neng Hwang, and Jianfeng Gao. Glipv2: Unifying localization and vision-language understanding. In *NeurIPS*, 2022. 2

[6] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Phillip Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, 2022. 1, 2

[7] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Simple multi-dataset detection. In *CVPR*, 2022. 1