5th Place Solution for Open Images 2019 - Segmentation

Roman Solovyev
Institute for Design Problems in Microelectronics of Russian Academy of Sciences
3, Sovetskaya Street, Moscow 124365, Russian Federation
turbo@ippm.ru

Weimin Wang
National University of Singapore
21 Lower Kent Ridge Rd, Singapore 119077
wangweimin777@gmail.com

Abstract

In this technical report, we discuss our 5th place solution for the Segmentation track of Open Images 2019 competition on Kaggle.com. Our solution consists of two pipelines - an Object Detection pipeline that generates bounding boxes for each individual objects; and a Segmentation pipeline which generates final object mask within each predicted bounding box. We also introduce a novel Weighted Box Fusion (WBF) ensembling algorithm that boosts the performance by ensembling predictions from different models.

1. Introduction

Computer vision has advanced considerably but is still challenged in matching the precision of human perception. Google AI hopes that having a single dataset with unified annotations for image classification, object detection, visual relationship detection, and instance segmentation will stimulate progress towards genuine scene understanding.

In this 3rd track of Open Images Challenge 2019 [1], contestants are provided with a training set that represents 2.1M segmentation masks for object instances in 300 categories; with a validation set containing an additional 23k masks. These data sets are annotated using a state-of-the-art segmentation labeling process in an iterative fashion with a focus on high label quality.

Annotating segmentation masks is a much more time consuming and labor intensive process than annotating boxes, and we have more labeled classes in Object Detection (500) than in Segmentation (300), this means we have much more annotated bounding boxes than masks. Can we leverage on the richer Object Detection (OD) dataset we are given, as well as our OD models which have already achieved above 0.6 mAP, to build an even more robust segmentation model? With this idea in mind, our team has created a solution which builds a class-agonistic UNET [8] and FPN [6] models on top of our OD model.

This solution much simplified our task where we can reuse our predictions from OD models. Our OD accuracy has reached above 0.6, therefore, we have good quality of bounding boxes to begin with. On the other hand, we train a class-agonistic UNET and FPN models with all segmentation masks, which by itself has avoided the class-imbalance problem of the dataset.

To further improve our accuracy on the Leader Board, we also trained instance segmentation model separately using the segmentation masks only. This training gives slightly worse result than our first proposed UNET/FPN method, but it gives additional model diversity and score liftings when ensembled into the final submission.

We will also discuss our Weighed Box Fusion algorithm which efficiently ensembles different models. Our final submission is an ensemble of UNET/FPN and instance segmentation models using Weighted Box Fusion algorithm, which achieved 5th place on final Leader Board.

2. Methodology

The provided ground truth masks cover 300 classes with extremely skewed distribution. The smallest class has only 14 labeled masks, where the largest has 173,786. In order to overcome this class-inbalanced issue, we trained a class-agonistic UNET and FPN models that only learns the semantic features of masks, without having to memorize the object categories. To train a separate instance segmentation model, on the other hand, we first started by training the model by uniform sampling on entire images. When the model converges, we switched to sub-training on minority classes - basically, we sorted all 300 classes based on labels counts in descending order, and continued to fine tuned our converged model on the bottom 150 classes to further converge it and to boost score on the minority classes. Subtraining improved the model performance by around 0.15-0.2 of mAP on Leader Board.

Our final solution is an ensemble of both UNET/FPN models and instance segmentation models with WBF, which achieved public Leader Board score of 0.5368 and private Leader Board score of 0.5022.

Our solution is implemented based on Keras [4], Tensorpack [9] and MMDetection [3]. The UNET/FPN models are built entirely using Keras, where instance segmentation models are built using Tensorpack and MMDetection.

Figure 1 above shows our overall solution pipeline. In the next few sub-sessions, we will talk about the implementation details for each model.

2.1. Class-agonistic UNET and FPN

Our UNET/FPN is trained using segmentation masks provided by the competition as training data. Basically, we feed the cropped the image from bounding box, and resize the cropped image to a fixed dimension, which in our case is RGB 224 x 224. We also resize the ground truth mask into the same dimension, i.e. 224 x 224. This resized image with its resized binary mask forms one pair of training data for UNET and FPN.

We trained FPN model with ResNet50 backbone and UNET model with ResNet152 backbone, both with input dimension of 224 x 224. We used instance normalization to replace all batch normalizations within decoders. FPN model has additional 300 inputs with class of box. This additional input transformed into (7x7x300) and (14x14x300) one-hot matrices and added to first and second block of decoder. As ablation study, we have shown that the UNET model without using additional input of object label achieved LB score of 0.4947. Using this additional input, our UNET achieved LB score of 0.5142, which has around 0.02 performance boost.

Training of both segmentation models were similar. We used uniform classes batch generation to partially beat class imbalance problem. Before applying standard augmentation we also randomly moved corners of boxes for around 5% of width and height of box. With this we emulate inaccuracy of object detection model. So we teach our segmentation model to predict on inaccurate boxes as well. After extracting image from box we also apply large set of standard augmentations from albumentation [2] library like: HorizontalFlip, Rotate, Noise, Blur, RGBShift, RandomBrightnessContrast, Elastic Transform, Grid Distortion, Jpeg Compression. It was very useful for classes with small number of entries.

During inference, we take the averaged predicted probabilities from both models for each image as the ensembled predicted probabilities. We predict two times for original image and for mirrored one. Then apply threshold of 0.5 to convert the probabilities into binary mask. We run inference for each image from our best OD predicted boxes, and the public Leader Board score for the ensembled UNET/FPN models is 0.5333.

2.2. Instance Segmentation model

We used both tensorpack and MMDetection as training tools for instance segmentation models. For tensorpack, we trained Faster RCNN models with ResNet101 as backbone, as well as with Cascade. For one model, we used group normalization while for the other one, we used freezed batch normalization. Both models are trained from COCO pretraining weights.

2.3. Weighted Box Fusion

When ensembling bounding boxes, there will be many overlapping boxes that have high IOU values (i.e. IOU >0.5) with each other. These boxes are from different models whose information may be lost if the boxes are simply removed. Therefore, instead of directly removing those boxes with lower probabilities like in algorithm of NMS, we will weighted average the boxes based on their coordinates and probabilities.

In our proposed WBF algorithm, for each class in each image, we will first find all overlapping bounding boxes with IOU larger than a pre-defined threshold. Second, within those selected boxes, we will weighted average each of the four coordinates across all boxes. The weighted averaging is done based on each box's prediction score.

2.4. Ensemble: Weighted Boxes Fusion applied to masks (WBFM)

During final model ensembles, we first convert all masks into their corresponding bounding boxes. When applying WBF to the bounding boxes, we select IOU threshold of 0.65 and merge all bounding boxes that 1) belonging to the same class 2) having IOU of 0.65. WBF will merge those bounding boxes which fullfill the above 2 criteria into one final box.

After applying WBF to bounding boxes to get the final box, we will calculate the final prediction scores for each pixel within the final bounding box, using each of the original binary mask multiplied by the mask's prediction score. For each pixel within the final bounding box, we will convert those that have prediction scores larger than the averaged score to be 1, and pixels less than the averaged to be 0. This will give us the final binary mask after applying WBF algorithm.

This wraps up the basic step of WBFM that will be applied to every mask of every class.

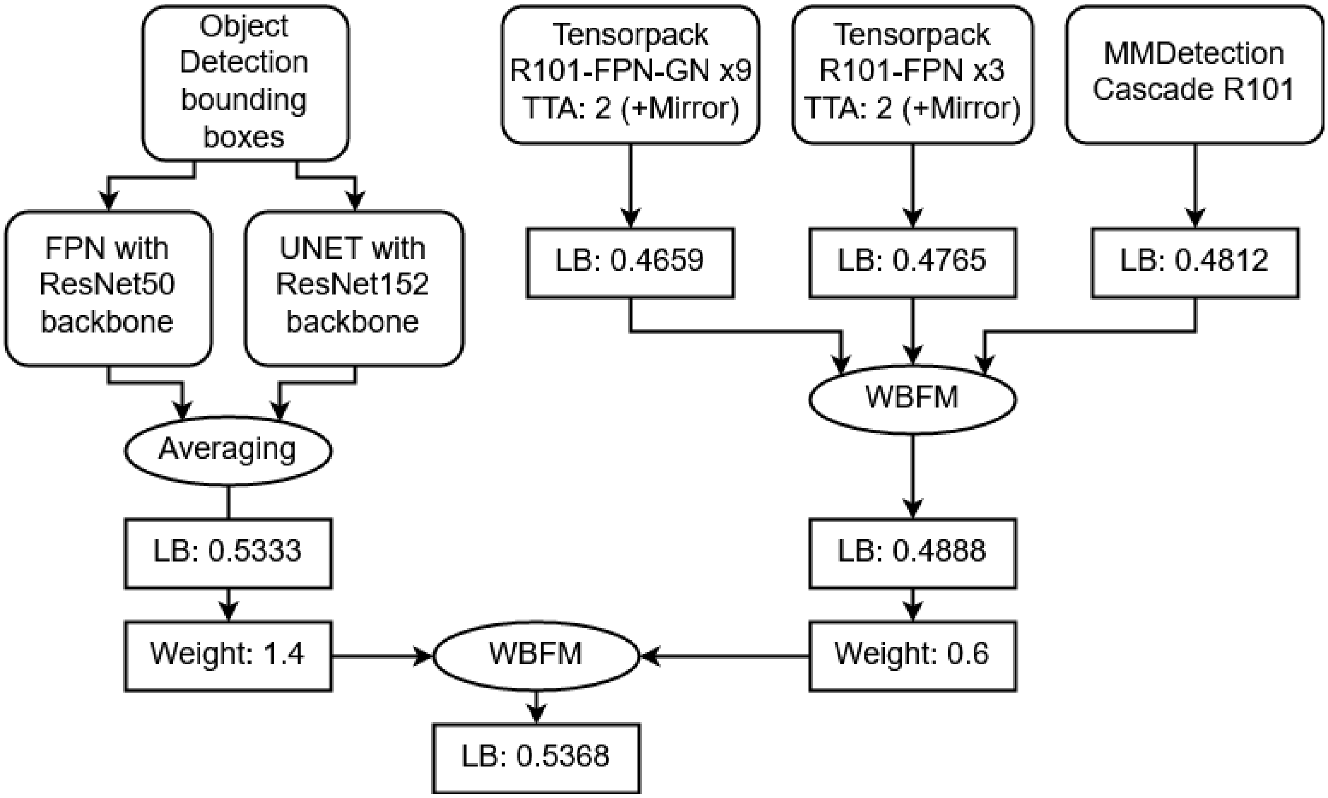
Object Detection bounding boxes
FPN with ResNet50 backbone
UNET with ResNet152 backbone
Averaging
LB: 0.5333
Weight: 1.4
Tensorpack R101-FPN-GN x9 TTA: 2 (+Mirror)
Tensorpack R101-FPN x3 TTA: 2 (+Mirror)
MMDetection Cascade R101
LB: 0.4659
LB: 0.4765
LB: 0.4812
WBFM
LB: 0.4888
Weight: 0.6
WBFM
LB: 0.5368

Figure 1. Final model.

3. Data

All our models are trained based on Open Image 2019 dataset provided by the competition. Some of our models are initialized with weights pre-trained on ImageNet [5] or COCO datasets [7].

4. Conclusion

During competition we find out that combination of Object Detection model with segmentation models like Unet and FPN in our case works better than independent instance segmentation models. Also we think that our masks ensemble algorithm can be improved to give better performance, because WBF in object detection works much better.

References

[1] Kaggle competition: Open images 2019 - instance segmentation. https://www.kaggle.com/c/open-images-2019-instance-segmentation, 2019.

[2] E. Khvedchenya V. I. Iglovikov A. Buslaev, A. Parinov and A. A. Kalinin. Albumentations: fast and flexible image augmentations. *ArXiv e-prints*, 2018.

[3] 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.

[4] François Chollet et al. Keras. https://keras.io, 2015.

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[6] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. *CoRR*, abs/1612.03144, 2016.

[7] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[9] Yuxin Wu et al. Tensorpack. https://github.com/tensorpack/, 2016.