Reproducibility Challenge of FreeAnchor

Min Wen, Mahsa Mesgaran

Abstract

Modern CNN-based object detectors assign anchors for ground-truth objects under the restriction of object-anchor Intersection-over-Unit (IoU). In paper FreeAnchor [1], it proposed a learning-to-match approach and breaks the IoU restriction, which allows objects to match anchors in a flexible manner. For a better understanding of Freeanchor, we conducted different experiments based on the code published by FreeAnchor. First, we reproduced the baseline result and found this network is robust according to consistent results. Second, we did two ablation experiments by changing two components in this network, which are saturated linear function and mean-max function respectively. Basically, FreeAnchor updates hand-crafted anchor assignment to "free" anchor matching by formulating detector training as a maximum likelihood estimation (MLE) procedure. It targets learning features which best explain a class of objects in terms of both classification and localization [1]. FreeAnchor proposed a detection customized likelihood including precision and recall and improved them with a likelihood optimization process. In this paper, we conducted different experiments based on this FreeAnchor method to demonstrate its reproducibility. Overall, we reached the baseline published in the paper, and performed ablation experiments and hyperparameters tuning which showed the network's robustness.

I. Introduction

In recent years, the convolution neural network (CNN) has dominated fields such as computer vision [2]–[8]. Many works [5]–[8] use feature pyramid network and anchor boxes in multiple scales and aspect ratios to represent objects in various spatial layouts, appearance and aspect ratios. Therefore, one typical approach of object localization and classification is by assigning each object to a single or multiple anchors, bounding box regression and classification are then carried out based on those anchors.

For clarifying roles of anchor, Intersection over Unit (IoU) needs to be introduced. Anchor-based detectors leverage spatial alignment, *i.e.*, IoU between objects and anchors, as the criterion for anchor assignment [1]. Each assigned anchor independently supervises network learning for object prediction, based upon the intuition that the anchors aligned with object bounding box are most appropriate for object classification and localization. However, FreeAnchor doubts such intuition and thus states that the hand-crafted IoU is not the best choice [1]. Furthermore, for objects with acentric features, e.g., slender object, the most representative features are not close to object centers. A spatially aligned anchor might correspond to fewer representative features, which deteriorate classification and localization capabilities. On the other hand, it is infeasible to match proper anchors/features for objects using IoU when multiple objects come together [1]. FreeAnchor is an approach to solve this problem, and this report shows its robustness and reproducibility.

The rest of this report is organized as follows: FreeAnchor's working mechanism will be explained in Sections II. In Section III, we would reproduce the results in FreeAnchor and the outcome of ablation experiments is shown. Section IV discusses the results and potential further improvements of the FreeAnchor, and the contribution is drawn in Section **??**.

II. Related Work

A. CNN-based Detector

In a traditional anchor-based training process, hand-crafted criterion based on IoU is used to assign anchors for objects, and a matrix $\mathcal{C} \in \{0,1\}$ is defined to indicate whether object b_i matches a_j or not. Specially, when multiple objects' IoU are greater than this threshold, the object of the largest IoU will successfully match this anchor, which guarantees that each anchor is matched by a single object at most, *i.e.*, $\sum_i \mathcal{C}_{ij} \in \{0,1\}$, $\forall a_j \in A$. By defining $A_+ \subseteq A$ as $\{a_j \mid \sum_i \mathcal{C}_{ij} = 1\}$ and $A_- \subseteq A$ as $\{a_j \mid \sum_i \mathcal{C}_{ij} = 0\}$, the loss function is written as follows:

$$\mathcal{L}(\theta) = \sum_{a_j \in A_+} \sum_{b_i \in B} \mathcal{C}_{ij} \mathcal{L}_{ij}^{cls}(\theta) + \beta \sum_{a_j \in A_+} \sum_{b_i \in B} \mathcal{C}_{ij} \mathcal{L}_{ij}^{loc}(\theta) + \sum_{a_j \in A_-} \mathcal{L}_j^{bg}(\theta) \tag{1}$$

where θ denotes the network parameters to be learned, $\mathcal{P}_{ij}^{cls}(\theta)$ and $\mathcal{P}_{ij}^{bg}(\theta)$ denote classification confidence and $\mathcal{P}_{ij}^{loc}(\theta)$ denotes localization confidence (all parameters reference [1]). $\mathcal{L}_{ij}^{cls}(\theta) = BCE(a_j^{cls}, b_i^{cls}, \theta)$, $\mathcal{L}_{ij}^{loc}(\theta) = SmoothL1(a_j^{loc}, b_i^{loc}, \theta)$ and $\mathcal{L}_j^{bg}(\theta) = BCE(a_j^{cls}, \overrightarrow{0}, \theta)$ respectively denote the Binary Cross Entropy loss (BCE) for classification and the SmoothL1 loss defined for localization. β is a regularization factor and "bg" indicates "background".

From the MLE perspective, the training loss $\mathcal{L}(\theta)$ is converted into a likelihood probability, as follows:

$$\begin{aligned}\mathcal{P}(\theta) &= e^{-\mathcal{L}(\theta)} \\ &= \prod_{a_j \in A_+} (\sum_{b_i \in B} \mathcal{C}_{ij} e^{\mathcal{L}_{ij}^{cls}(\theta)}) \prod_{a_j \in A_+} (\sum_{b_i \in B} \mathcal{C}_{ij} e^{-\beta \mathcal{L}_{ij}^{loc}(\theta)}) \prod_{a_j \in A_-} e^{-\mathcal{L}_j^{bg}(\theta)} \\ &= \prod_{a_j \in A_+} (\sum_{b_i \in B} \mathcal{C}_{ij} \mathcal{P}_{ij}^{cls}(\theta)) \prod_{a_j \in A_+} (\sum_{b_i \in B} \mathcal{C}_{ij} \mathcal{P}_{ij}^{loc}(\theta)) \prod_{a_j \in A_-} \mathcal{P}_{ij}^{bg}(\theta)\end{aligned} \tag{2}$$

B. FreeAnchor Theory

FreeAnchor has three main objectives. First, to achieve a high recall rate, the detector is required to guarantee that for each object, at least one anchor's prediction is close to the ground-truth [1]. This is achieved by maximizing the detection customized likelihood, as defined in Eq.3:

$$\begin{aligned}\mathcal{P}'(\theta) &= \mathcal{P}_{recall}(\theta)\mathcal{P}_{precision}(\theta) \\ &= \prod_i \max_{a_j \in A_i} (\mathcal{P}_{ij}^{cls}(\theta)\mathcal{P}_{ij}^{loc}(\theta)) \times \prod_i (1 - \mathcal{P}\{a_j \in A_-\}(1 - \mathcal{P}_i^{bg}(\theta))),\end{aligned} \tag{3}$$

Second, in order to achieve high detection precision, the detector needs to classify anchors with poor localization (large bounding box regression error) into background [1]. This specifically is implemented by optimizing the flowing function, Eq.4:

$$\mathcal{P}_{precision}(\theta) = \prod_i (1 - P\{a_j \in A_-\}(1 - \mathcal{P}_j^{bg}(\theta))) \tag{4}$$

Third, the predictions of anchors should be compatible with the non-maximum suppression (NMS) procedure, *i.e.*, the higher the classification score is, the more accurate the localization is [1]. This point is realized by introducing Saturated Linear Function as following Eq. 5:

$$SaturatedLinear(x, t1, t2) = \begin{cases} 0, & x \leq t_1 \\ \frac{x - t_1}{t_2 - t_1}, & t_1 < x < t_2 \\ 1, & x \geq t_2 \end{cases} \tag{5}$$

To fulfill these objectives, FreeAnchor formulates object-anchor matching as a maximum likelihood estimation (MLE) procedure [9] which selects the most representative anchor from a "bag" of anchors for each object. This introduces learning-to-match approach, as shown in Fig. 1. For jointly optimizing object classification, object localization, the detection customized likelihood is converted to a detection customized loss function, as follows:

$$\begin{aligned}\mathcal{L}'(\theta) &= -log\mathcal{P}'(\theta) \\ &= -\sum_i log(mean - max_{a_j \in A_i}(\mathcal{P}_{ij}^{cls}(\theta)\mathcal{P}_{ij}^{loc}(\theta))) - \sum_j (1 - P\{a_j \in A_-\}(1 - \mathcal{P}_j^{bg}(\theta))),\end{aligned} \tag{6}$$

where the mean-max function is used to select the best anchor for each object. During training, a single anchor is selected from a bag of anchors A_i, which is then used to update the network parameter θ [1].

III. Experiments

In this section, we present the results of our ablation experiments and hyperparameter testing. We use the same dataset as the original paper (*i.e.* COCO 2017 [10]), which contains around 118k images for training, 5k for validation and around 20k for testing. All result are reported on validation dataset except the baseline comparison, which is based on testing dataset. For simplicity, We only choose ResNet [11] as our backbone and all setting are same with FreeAnchor [1] except specifically noted. We use 8 Tesla V100 GPUs and training with synchronized SGD of 16 images per batch.

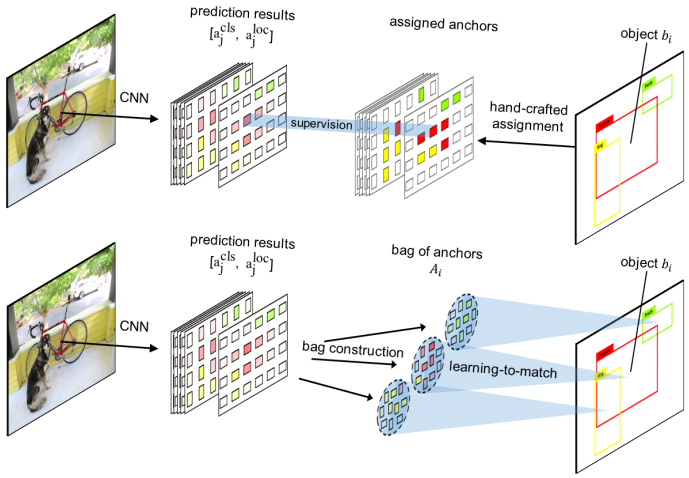

Fig. 1: Comparison of hand-crafted anchor assignment (top) and FreeAnchor (bottom).

TABLE I: Baseline Result

Backbone	FreeAnchor	Training Time	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
ResNet-50	Published	5.27h	38.7	57.3	41.6	20.2	41.3	50.1
ResNet-50	Reproduced	8.43h	38.8	57.4	41.5	21.4	42.3	51.5
ResNet-101	Published	7.26h	40.9	59.9	43.8	21.7	43.8	53.0
ResNet-101	Reproduced	10.17	40.8	59.8	43.8	21.7	44.0	53.0

A. Baseline Result

Table I and table II gives us the reproduced results of FreeAnchor. Generally, there is no difference between reproduced results and that of the paper published. Increase of training time is negligible in this content because it happened in both backbones. In this case, we would say freeanchor is stable and robust.

TABLE II: Reproduced NMS recall (%) on COCO val set.

FreeAnchor	Inference Time	NR	NR_{50}	NR_{60}	NR_{70}	NR_{80}	NR_{90}
Published	7.02m	83.8	99.2	97.5	89.5	74.3	53.1
Reproduced(sigmoid)	11.27m	83.8	99.3	97.5	89.3	74.5	52.9

B. Ablation Experiments

We tested two model components, *i.e.* Saturate linear function and Mean-max function in this section. From the original paper, both components are set for solving certain problem, which are compatibility with NMS and wrongly learning when training insufficiently. For NMS compatibility, we evaluated NMS recall; for wrongly learning, we made parameter learning curve.

1) Saturated Linear Function: To achieve the third objective in II-B, FreeAnchor introduced Saturated Linear Function [1], Eq. 5, which has the following three properties: (1) $P\{a_j \rightarrow b_i\}$ is a monotonically increasing function of the IoU between a_j^{loc} and b_i, IoU_{ij}^{loc}. (2) When IoU_{ij}^{loc} is smaller than a threshold t, $P\{a_j \rightarrow b_i\}$ is close to 0. (3) For an object b_i, there exists one and only one a_j satisfying $P\{a_j \rightarrow b_i\} = 1$. We replace linearity with Sigmoid (Eq. 7), which also meets those requirements listed above except it is a continuous function. As shown in Fig. 2, we set its upper limit to infinitely near 1 and lower limit to infinitely near 0. For simplicity, we define that when IoU reaches lower threshold, its probability reaches 0.01; and 0.99 in the opposite. The result is shown in table III and IV. As the results, training time increased while other index mostly remain constant. This is because anchor bag construction only recognizes the rank of probabilities of anchors. In other words, changing to sigmoid does not vary the rank. Though some miner increase among AP, which may due to increase of anchor probability, it costs large inference time. Inference is carried on COCO2017 validation set, and time is calculated based on 4 Tesla V100 GPU. Therefore, saturated linear function is efficient enough for being compatible with NMS.

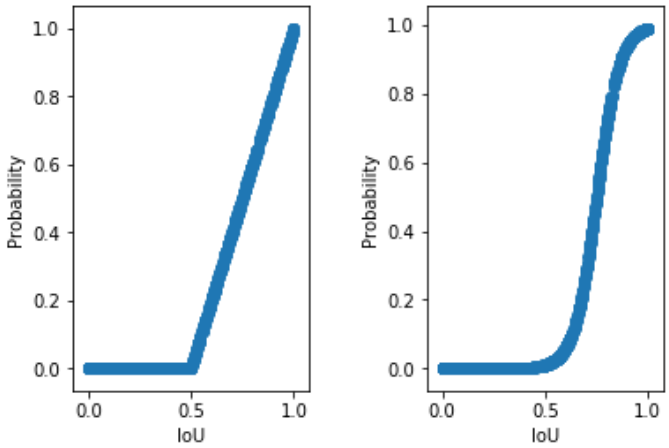

Fig. 2: Change Saturated Linear Function to Sigmoid Function

TABLE III: Comparision of Saturated Linear Function and Sigmoid Function

Function	Training Time	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Linear	8.43h	38.8	57.4	41.5	21.4	42.3	51.5
Sigmoid	9.09h	38.3	56.7	41.0	20.9	41.3	52.3

$$Probability(x, t_1, t_2) = \frac{1}{1 + \mathrm{e}^{k(a-x)}} \quad \text{where} \quad a = \frac{(t_1 + t_2)}{2}, k = \log(99)(2/t_2 - t_1) \tag{7}$$

TABLE IV: Comparison of NMS recall (%) on COCO val set.

FreeAnchor	Inference Time	NR	NR_{50}	NR_{60}	NR_{70}	NR_{80}	NR_{90}
linear	11.27m	83.8	99.3	97.5	89.3	74.5	52.9
Sigmoid	11.35m	83.8	99.2	97.4	89.6	74.3	52.7

2) Mean-max Function: One key element in learning-to-match is the Mean-max function, Eq. 8. Due to the confidence of all anchors being small at early training stage, it is meaningless to select the anchor with maximum confidence to train because selected anchor may not contain useful information of object. Mean-max function can alleviate this defect by calculating mean instead of max when training is insufficient. In our study, we tested it by replacing it with max function and found that accuracy dropped significantly, but training time shrank as shown in table V.

$$Mean - max(X) = \frac{\sum_{x_j \in X} \frac{x_j}{1-x_j}}{\sum_{x_j \in X} \frac{1}{1-x_j}} \tag{8}$$

C. Hyperparameter Tuning

In this section, we present our results of implementing different FreeAnchor detectors with different hyper-parameter settings. Results are listed in Table VI. In general, Focal loss parameters *i.e.* alpha and gamma have relatively large effects, which indicates using validated settings of Focal Loss Eq. 9 [8] is optimal. γ is the focusing parameter and it facilitates detector to focusing objective. α balances the importance of positive/negative examples during training. Theses two parameters may depends on dataset type. Besides, the results of changing the background threshold and anchor bag size show that 0.6 and 80 is the best to fit COCO dataset.

$$FL(\mathcal{P}_t) = -(1 - \mathcal{P}_t)^{\gamma} \log(\mathcal{P}_t) \tag{9}$$

TABLE V: Comparison between Mean-max and Max

Function	Training Time	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Mean-max	8.43h	38.8	57.4	41.5	21.4	42.3	51.5
Max	7.09h	34.2	53.4	36.1	19.4	37.1	45.3

TABLE VI: Hyperparamenter Tuning Results

Anchor Bag Size	Background IoU threshold	alpha	gamma	AP	AP50	AP75	APS	APm	APL
40	0.6	0.5	2.0	38.5	57.1	41.4	20.8	42.1	51.8
50	0.6	0.5	2.0	38.8	57.4	41.5	21.4	42.3	51.5
60	0.6	0.5	2.0	38.2	56.6	40.8	20.1	41.5	52.0
100	0.6	0.5	2.0	38.1	56.4	40.8	20.0	41.7	51.7
50	0.5	0.5	2.0	38.4	57.2	41.1	20.2	41.9	52.1
50	0.7	0.5	2.0	38.5	56.8	41.1	21.2	42.0	52.0
50	0.6	0.25	2.0	37.8	56.0	40.6	19.4	41.5	51.3
50	0.6	0.75	2.0	38.0	56.2	40.2	19.5	41.2	50.9
50	0.6	0.5	1.5	38.2	54.2	40.1	20.1	41.9	48.9
50	0.6	0.5	2.5	38.1	55.2	41.0	20.2	40.2	40.2

IV. Discussion and Conclusion

In this project, we successfully reproduced the baseline result of FreeAnchor paper. We also conducted ablation experiments to testify the practicality of Saturated Linear function and Mean-max function. Moreover, we explored the effect brought by changing different hyperparameters in the model. After setting control variables towards those hyperparameters, (i.e. anchor bag size, Background IoU threshold and Focal Loss parameters), we gained a deeper understanding of the network's learning process. Overall, results show that FreeAnchor is robust and reproducible.

Some possible directions for future investigation may lie in more complicated anchor-selection criterion, such as a combination of hand-crafted and learning-to-match.

REFERENCES

[1] X. Zhang, F. Wan, C. Liu, R. Ji, and Q. Ye, "Freeanchor: Learning to match anchors for visual object detection," in *Advances in Neural Information Processing Systems*, 2019, pp. 147–155.

[2] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," 2013.

[3] R. B. Girshick, "Fast R-CNN," *CoRR*, vol. abs/1504.08083, 2015. [Online]. Available: http://arxiv.org/abs/1504.08083

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2015.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[6] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," 2015.

[7] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," 2016.

[8] L. Tsung-Yi, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *IEEE International Conference on Computer Vision, ICCV 2017*, 2017, pp. 2999–3007.

[9] O. Maron and T. Lozano-Pérez, "A framework for multiple-instance learning," in *Advances in neural information processing systems*, 1998, pp. 570–576.

[10] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.