Faster R-CNN with Anchor Proposal Networks

Jaehoon Chung
POSTECH
sk7755@postech.ac.kr

Dahye Jeong
POSTECH
dahyejeong@postech.ac.kr

Abstract

In this project, we made an advance in the performance of Faster R-CNN[3], a region-based network for object detection. In particular, we designed a novel network called the ***Anchor Proposal Network (APN)****. Given a convolutional feature map,* ***APN*** *outputs 3 aspect ratios of the anchors, which in turn will be fed into the region proposal network of the original Faster R-CNN module.*
We provide experimental results that compare the performance of the original Faster R-CNN and our network which combines ***APN + Faster R-CNN****.*

1. Introduction

Object detection has come a long way in the field of computer vision and there has been an abundance of networks that try to achieve the best performance. There are a couple of important frameworks when it comes to object detection. Region-based networks(R-CNNs) have 2 separate pipelines for region proposal and classification, which have been studied extensively due to its state-of-the-art performance at the time it was introduced. On the other hand, there are networks like YOLO[2] which use a single convolutional network to predict the bounding boxes and its classes.

In this project, we take a deeper look into region-based networks, especially Faster R-CNN and we advance its performance by dynamically choosing the ratio of the anchor boxes for each individual image.

2. Related Works

Faster R-CNN Faster R-CNN[3] is a region-based network consisted of 2 modules: (1)region proposal, (2)classification. In the region proposal pipeline, there is a *Region Proposal Network(RPN)* which takes an input image and returns a set of rectangular object proposals. Each of these object candidates have an *objectness score*. To generate the object proposals, small boxes called *anchors* are slided over the convolutional feature map. In the original paper, the authors used 9 anchors, which were generated by 3 aspect ratios and 3 scales. These proposals are then fed into a regression network and a classification layer.

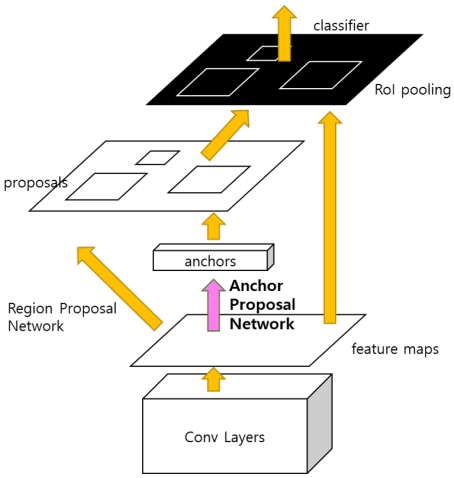

Figure 1. Faster R-CNN Network with Anchor Proposal Network

3. Anchor Proposal Networks

Given an image, the backbone network outputs a feature map. The feature map has information about the objects in the original image. Therefore, it is natural to think that the feature map also has information about the size, aspect ratio, and class of the objects in the image. We pay attention to the fact that Faster R-CNN fixes the aspect ratios of anchors, even though the feature map contains information related to aspect ratios of objects.

The Anchor Proposal Network(APN) proposes the aspect ratios of the objects in the image, and we use them as the aspect ratios of anchors which RPN needs for classification and regression. To generate aspect ratios, we feed the feature map into our APN. The APN will output the score of the specific aspect ratio which indicates the probability that an object of that aspect ratio exists. We select three aspect ratios whose scores are ranked highest and apply them to Faster R-CNN. See Figure 1.

Leaderboard	Id	Method	Title	Challenge	Status	Score	When
private	31830	Simple Faster-RCNN	Simple Faster-RCNN	VOC2012	Evaluation completed successfully	0.63470	19 Dec 2020, 10:26
private	31831	Anchor Proposal Network	APN	VOC2012	Evaluation completed successfully	0.67820	19 Dec 2020, 10:27

Figure 2. Detection results on PASCAL VOC 2012 test set. The detectors are original Faster R-CNN and Faster R-CNN with Anchor Proposal Network.

3.1. Selecting Candidates of aspect ratios

There are twenty candidate values for the aspect ratios of the proposed anchors. The number of candidates is a hand-tuned hyper parameter, which was determined under the observation that there is not a huge difference in the aspect ratios between objects in the same class. Thus we set it to the number of classes of PASCAL VOC 2012, twenty. We use the k-means clustering algorithm to the VOC 2012 images to extract the representatives of aspect ratios, so that we can get rid of the alienation between detected objects and anchor boxes.

3.2. Designing Ground-Truth

For training APNs, we devise the ground-truth vector GT whose length is 20. If any representative is sufficiently closer to the ground-truth aspect ratio in the image, we will give high value to the position corresponding to it. Then each element of the vector indicates the likelihood of existence of the corresponding aspect ratio.

First, we define the distance metric between aspect ratios. As the name tells us, the distance of aspect ratios can be measured by their ratio. Our distance is defined as:

$$d(r_1, r_2) = \frac{max(r_1, r_2)}{min(r_1, r_2)} \tag{1}$$

where r_1, r_2 are aspect ratios. Note that the distance always ranges from 0 to 1. For example, $d(2.0, 4.0) = d(4.0, 8.0) = 2$.

According to the metric, we set each value of the ground-truth vector as maximum of the inverse of the distance between the representative and the aspect ratios of the objects. (This way, the aspect ratio that is closest to the representative is chosen.) That is,

$$GT(i) = max_{r_{gt} \in R_{GT}}(1/d(r_i, r_{gt})) \quad , 1 \leq i \leq 20 \tag{2}$$

where r_i is the representative aspect ratio of i-th index, and R_{GT} is the set of ground-truth aspect ratios in an image. If there is an aspect ratio of the ground-truth boxes which is the same as the representative, it will be set to one.

3.3. Training APNs

The APN consists of two fully connected layers where the input size is 512*7*7, the size of hidden layer is 2048, and the output size is 20. Since APN uses only the feature map extractor to propose aspect ratios, it can be trained regardless of the Faster R-CNN. Thus we use the VGG-16 as the backbone network and concatenate our APN with it. Since it has the risk of overfitting on PASCAL VOC dataset, we use dropout($p = 0.5$) and proper regularization such as weight decay. We select a sigmoid function as activation of the last layer. The predicted vector also ranges from 0 to 1. This allows us to use the mean squared error(MSE) for computing our loss. After training, we add this network in the middle between backbone network and RPN in Faster R-CNN and train the entire network, while freezing the weights of APN.

4. Experiments

We train and evaluate our method on the PASCAL VOC 2012 dataset. The test dataset consists of 11K trainval images and 10K test images. On the PASCAL VOC 2012 detection benchmark, we evaluate detection mean Average Precision(mAP). The base code we use, called **simple-faster-rcnn**[1], is a simplified implementation of Faster R-CNN and it matches the performance reported in original paper. We insert our pretrained APN to the base code and construct Faster R-CNN with APN. We compare the mAP of the original with ours to verify that the performance of Faster R-CNN can be improved due to our APN. See Figure 2.

5. Conclusion

We have achieved higher performance than the orginal Faster R-CNN by introducing a novel anchor proposal network(APN). We defined a new metric for measuring the similarity between ratios, and devised a way to solve the regression of aspect ratios by transforming it to a classification problem of 20 representatives. Using this result, we proposed tailored anchor boxes for each individual image.

Although we have achieved better performance than the original Faster R-CNN, we have some room for improvement. First of all, although the APN is capable of predicting several aspect ratios at once, we only use the top 3. It would be better if the RPN can dynamically vary the number of aspect ratios when generating the anchor boxes for each image. Also, we would like to mention that we only considered aspect ratios to predict the shape of anchor boxes. However, the shape of an anchor box is determined

not only by the aspect ratio, but also the scale. Therefore, if the APN can also predict the proper scales, we expect the object detection performance to improve.

References

[1] chenyuntc, ZuochaoLee, and GreenTeaHua. A simple and fast implementation of faster r-cnn. *https://github.com/chenyuntc/simple-faster-rcnn-pytorch*. 2

[2] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2015. 1

[3] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2015. 1