

Pooling Pyramid Network for Object Detection

Pengchong Jin

Vivek Rathod

Xiangxin Zhu

Google AI Perception

{pengchong, rathodv, xiangxin}@google.com

Abstract

We'd like share a simple tweak of the SSD family detectors, which is effective in reducing the model size while maintaining the same quality. We share the predictors across all the scales, and replace the convolution between scales with max pooling. This has two advantages over the SSD: (1) avoids score miscalibration across scales; (2) the shared predictor sees the training data over all scale. Since we reduce the number of predictor to one, and trim all the convolutions between them, the model size is significantly reduced. We empirically found that it didn't hurt the model quality compared with its SSD counterpart.

1. Introduction

The SSD family detectors [2], [1] have been popular as they run fast, are simple to implementation and easily portable to different types of hardware.

Most of the SSD detectors have several feature maps representing different scales, each of which uses its own predictor to produce the boxes and class scores. In practice, especially when the data distribution is skewed over scales, this design could run into problem. Imagine a dataset with tons of large objects and very few small ones. Those predictors from the small scale feature maps will be wasted as they rarely see any positives. This data imbalance could also result in score miscalibration across scale even for the same class. Another issue under this design is, each predictor only sees the objects at its own scale. This partitions the oftentimes already small dataset into even smaller pieces. If we believe the object appearance is scale invariant, it will be a more efficient way of using the data if all the predictors see all the data.

We propose simple changes to the SSD: use the same predictor in all scales. In order for the predictor to work in the same feature space, we replace the convolutions between feature maps with max pooling.

2. Pooling Pyramid Network (PPN)

The proposed model, *Pooling Pyramid Network (PPN)*, is a single-stage convolutional object detector, very similar to SSD with simple changes. The prediction head is designed to be light-weighted, fast to run, while maintains the comparable detection accuracy with its SSD counterpart. The network architecture is illustrated in Figure 1. There are two major changes to the original SSD [2]: (1) the box predictor is shared across feature maps with different scales; (2) the convolutions between feature maps are replaced with the max pooling operations. In the following sections, we will discuss the rationales behind them and effects of these changes.

2.1. Shared Box Predictor

The SSD uses independent box predictors for feature maps at different scales. The one potential problem is miscalibration of the prediction scores across different scales.

Since each box predictor is trained independently using only a portion of the groundtruth boxes that it is assigned to, different box predictors could see very different amount of positive and negative examples during the training. This implicit data imbalance could cause the problem that scores from different predictors fall in vastly different ranges, which makes them incomparable and difficult to use in the subsequent score-based postprocessing such as non maximum suppression. We design PPN with a shared box predictor across feature maps of different scales. As a result, the box predictor sees all the training data during the training when there are imbalance groundtruth boxes with different scales. This reduces the effect of miscalibration and unstable prediction scores.

One could argue that having separate box predictor for each scale increases the total capacity, and allows each predictor to focus at its specific scale. However, we think that this may not be necessary as objects are mostly scale invariant. In practice, Faster-RCNN [3] works well with a single shared predictor.

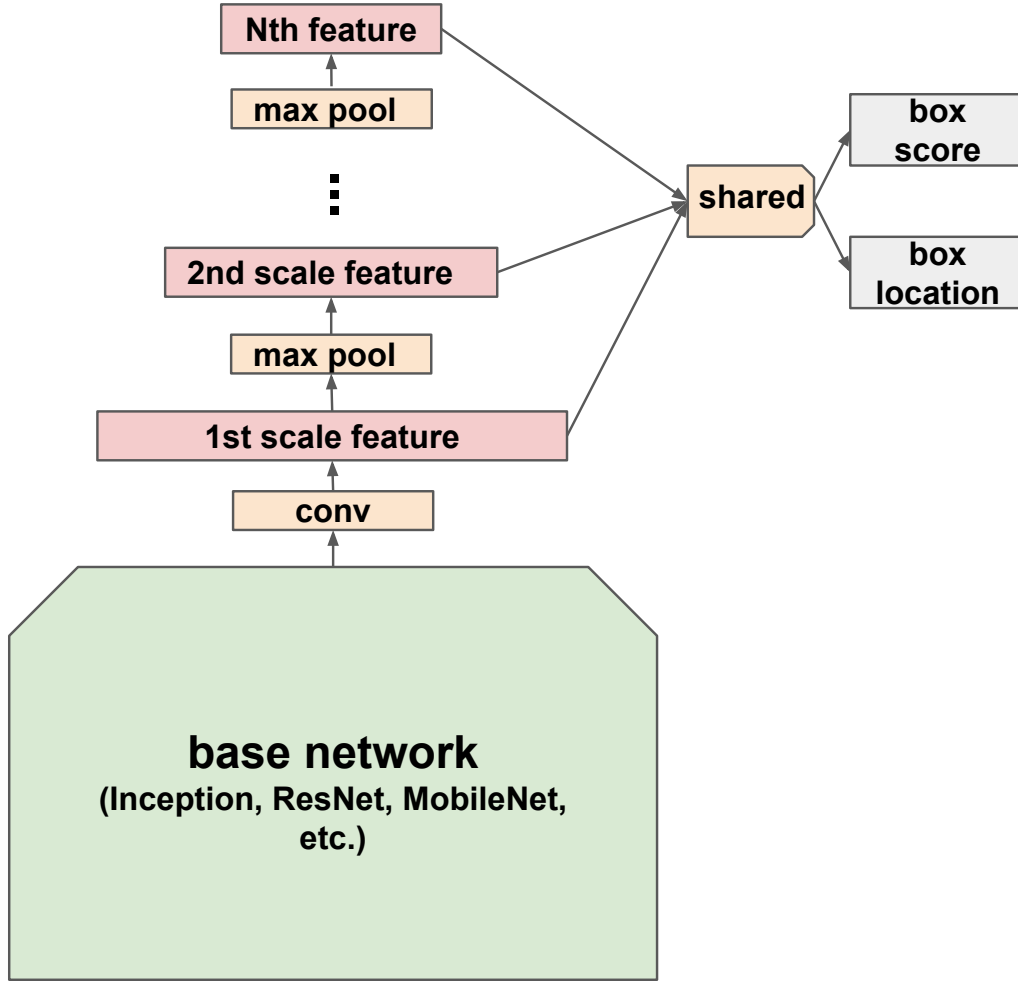


Figure 1. The Pooling Pyramid Network (PPN) architecture. [Todo: Replace this with a comparison between SSD and PPN to illustrate the two main changes.]

2.2. Max Pooling Pyramid

Our goal is to build a multi-scale feature pyramid structure, from which we can make the predictions using the shared box predictor. We achieve this by shrinking down a base feature map from the backbone network several times using a series of max pooling operations. This is different from SSD where feature maps are built by extracting layers from backbone network and shrinking them using additional convolutions, and FPN where feature maps are built by a top-down pathway with skip connections. We choose max pooling mainly for two reasons. First, using the pooling operations ensures feature maps with different scales live in the same embedding space, which makes training the shared box predictor more effective. In addition, since max pooling does not require any additions and multiplications, it is very fast to compute during the inference, therefore, making it suitable for many latency sensitive applications.

2.3. Overall Architecture

The final network architecture of our Pooling Pyramid Network (PPN) detector is illustrated in Figure 1. Followed by the backbone network, an optional 1×1 convolution is used to transform the features from the backbone network to a space with desired dimensions. We then apply a series of stride-2 max pooling operations to shrink the feature map down to 1×1 . A shared box predictor is applied to feature maps of different scales in order to produce classification scores and location offsets of box predictions. We add one additional shared convolution in the box predictor after pooling operations to prepare the feature to be used for predictions.

Model	AP	AP50	AP75	FLOPs	number of parameters
MobileNet SSD	1	1	1	1	1
MobileNet PPN	1	1	1	1	1

Table 1. COCO detection: MobileNet SSD vs MobileNet PPN

3. Experiments

3.1. Comparing SSD and PPN

References

- [1] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar. Focal loss for dense object detection. In *ICCV*, 2017.
- [2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016.
- [3] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015.