[Group 9] Object Detection based on Faster R-CNN

Wei Sun
4711114
Embedded Systems

Kun Jiang
4701674
Embedded Systems

Xianhao Ni
4714431
Embedded Systems

Yiting Lu
4714881
Signals & Systems

Abstract

With the rapid development of the convolutional neural network (CNNs), many applications of CNN have been applied successfully in the field of the image recognition. One of the successful extension of the CNN is region-based convolutional neural network (R-CNNs), it leads to good performance beyond expectation compared with other state-of-art neural networks in object detection. This article aims to make a deep research on the configuration of the enhancement of the R-CNN called Faster R-CNN. Moreover, this project compares the performance based on the different base network of the Faster R-CNN to validate our hypothesis that better feature detector in image recognition can also perform better in object detection.

1. Introduction

Deep learning has been showing the outstanding performance in the field of image recognition than other algorithms from machine learning. Convolution neural network (CNNs) is the representative for image recognition as well as object detection. Further development of CNN has led to unprecedented performance in object detection through using a family of techniques referred to as region-based convolutional neural networks (R-CNNs). The ability to classify and locate objects in an image can be useful particularly for self-driving vehicles, which need to track the positions of surrounding objects they could potentially collide with.

In this project, our hypothesis is that better feature detector in image recognition can also perform better in object detection which is applied to the base network of the Faster R-CNN [1]. To validate our hypothesis is correct, we have used four different base network in this project, they are VGG16, ResNet50, ResNet101 and MobileNet.

The key characteristic of VGG style networks is that using small 3x3 filters in convolutional layers allows for networks with greater depth without an explosion in the number of trainable parameters. VGG16 significantly outperformed older network architectures in image recognition, showing that thinner (smaller filters in each layer), deeper networks learn better features than wider, shallower ones. Residual networks address deep networks trouble learning identity mappings by adding shortcut connections between the output of one layer and the input of another layer not directly connected to it. MobileNet[2] is a brand new family of convolutional neural network based on depthwise separable convolutions, which can reduce the computation time and number of parameters quite a lot but maintain almost same performance. Thus, the base network part will be replaced by a MobileNet and a comparison will be made.

2. Problem Statement

Although Faster R-CNN has already achieved a real-time performance by directly extracting region of interest from images via convolution network and region proposal network (RPN). Three issues draw our attention. The first hypothesis we considered is the depth of the base network. By convention, a deeper CNN should led to a better feature generation performance, however the gradient vanishing problem become serious when the depth of CNN exceeds a certain number. That's why VGG16 seems to be an optimal choice for the base network with limited convolution layers(13) and acceptable performance. To solve this problem, resnet comes into our sight. By adding short cut to discontinuous convolution layers, resnet is able to use a deeper structure without the gradient vanishing problem, resulting a better final performance. This project will compare the performance of Faster R-CNNs that based on different base networks, namely VGG16, ResNet50, ResNet101, to validate our assumption.

Secondly, whether we could lighter the Faster R-CNN and shorter the image processing time further more by repalcing the base network with simpler structure and less parameters. Thus a light MobileNet structure is taken into consideration. We hope the MoblieNet could significantly shorter the detection time without too much compromising of accuracy.

Thirdly, whether the number of ROI (region of interest) be a critical factor that influences the performance of the structure. More ROIs may led to a higher mAP while longer processing time each image. Thus for each base network structure, we set the number of ROIs to different value (64, 128, 300) to validate our hypothesis.

3. Approach Analysis

This section describes the dataset, the algorithms and the metric we used in this project.

3.1. Dataset

All the models in this experiment are trained on the image set PASCAL VOC 2007. This dataset includes 2501 images in the training set, 2510 in the validation set, and 4952 in the test set. Images have varying sizes, but are generally around 500 pixels in width and 375 pixels in height. Objects are to be classified into 20 classes: aeroplane, bicycle,bird,boat, bottle, bus, car, cat, chair, cow, diningtable, dog, horse, motorbike, person, pottedplant, sheep, sofa, train, and tvmonitor. Each image comes with annotations with metadata about the image and the objects in it. For object detection purposes, the relevant metadata include the following fields:

- The height of the image in pixels.
- The width of the image in pixels.
- A list of the objects in the image and useful metadata about each.

3.2. Family of R-CNN

3.2.1 R-CNN

R-CNN (Regions with CNN features) generates region proposals via selective research method, then feeds those proposals into CNN, generating feature maps. Those feature map are classified by the SVM classifier. R-CNN is slow because it performs a ConvNet forward pass for each object proposal without sharing computation[3].

3.2.2 Fast R-CNN

Fast R-CNN still uses selective research method to generate object proposals, then feeds the entire image and the object proposals into a CNN to produce a conv feature map. For each object proposal, Fast R-CNN extracts a fixed-length feature from the feature map. Each feature vector is fed into a sequence of fully connected layers that finally branch into two sibling output layers: one that produces softmax probability estimates over K object classes plus a catch-all background class and another layer that outputs four real-valued numbers for each of the K object classes. Each set of 4 values encodes refined bounding-box positions for one of the K classes. Fast-RCNN still can not reach a real-time performance [4].

3.2.3 Faster R-CNN

Compared with the former two methods, Faster R-CNN first extracts feature maps from images via a series of convolutional layers, then directly generates region proposals through RPN. Faster R-CNN actually is a combination of RPN and traditional Fast R-CNN. The base network (e.g. VGG16) extracts feature maps from images, RPN generates object proposals based on the feature maps and raw images. Those proposals and feature maps then are fed into Fast-RCNN together[5]. For both RPN and Fast-RCNN, the base network is the shared part.It's a critical reason why this structure is so fast.

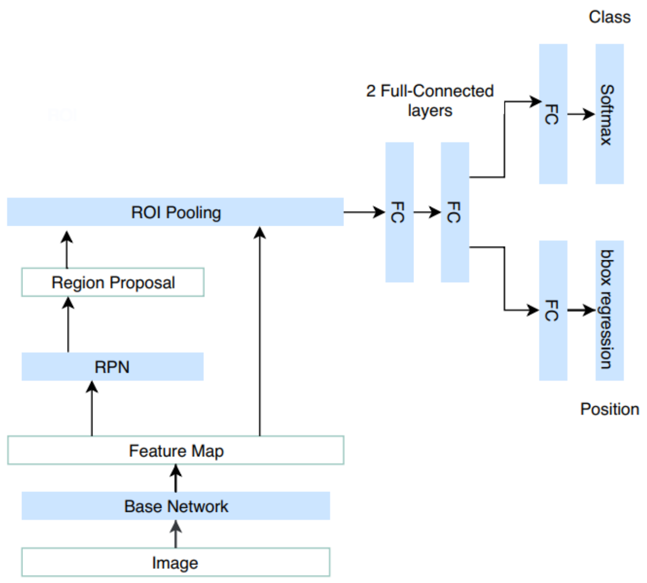

Figure 1. Faster R-CNN Structure

3.3. Base Network

3.3.1 VGG16

As we described in Section 1, VGG16 is a type of thinner and deeper neural network which can learn better features than wider and shallower ones. However, there is a drawback for deeper neural network which is that they are harder to train with todays optimization algorithms [6]. Experiments have shown that arbitrarily adding more layers to a network past a certain depth increases training error, despite the deeper network is a superset of the shallower network.

3.3.2 Residual Network

Residual Network was introduced by Microsoft Research Asia in 2015 [6]. ResNet has shown its good performance in many computer vision tasks ,for example, it led to 1st-place winning entries in all five main tracks of the ImageNet and COCO 2015 competitions, which covered image classification, object detection, and semantic segmentation. In this project, ResNet-50(50 layers) and ResNet-101(101 layers) are used as base network respectively to investigate the performance of Faster R-CNN.

3.3.3 MobileNet

The most important idea behind MobileNet [7] is that it uses depthwise separable convolutions to build light-weight deep neural networks. Figure 2 [8] shows the difference between depthwise and traditional pointwise convolution. Depthwise architecture allows the network learn features more efficient with less parameters, thus significantly reducing the computation time.

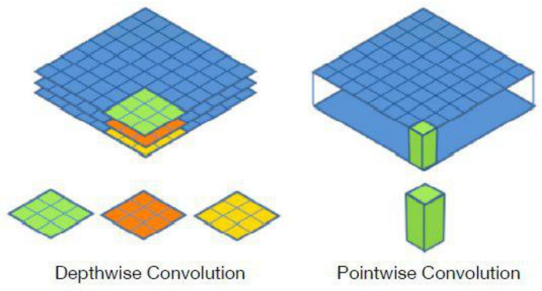

Figure 2. DepthWise v.s. PointWise

4. Experiment Setup

The experiment is performed in Google Cloud Platform and the following hardware specification is used for the whole experiment:

- 4vCPUs
- 15 GB RAM
- 1 NVIDIA Tesla P100 GPU

Further, the software specification that we used for the purpose of the experiment is as follows:

- Ubuntu 16.04 LTS Operating System
- Google Cloud Platform

5. Experiments

This section shows the steps of the experiments we did in this project.

5.1. Data Preprocessing

Preprocessing steps are performed according to the way described in paper Faster R-CNN [1]:

- For PASCAL VOC datasets, each image is resized to a minimum of 600 pixels on the shorter side and if the longer side is more than 1000 pixels, the image is shrunk into 1000 pixels.
- Training sets are augmented with an additional, horizontally flipped copy of each image in the original set. Training images are shuffled at the start of training and after each pass through the full set, a half of the training iterations use a flipped image.
- To be compatible with Imagenet-pretrained weights for base models, images RGB values are preprocessed by subtracting mean values from Imagenet [9].
- When using MobileNet (pre-trained) as base network, the input images have to be resized to 244*244 so that they can be processed by pre-trained model.

5.2. Implementation

In this project, we implement Faster R-CNN in Keras using Tensorflow as the backend engine. The source code for base network VGG16, ResNet50 and ResNet101 is referred to on one of the open source codes from GitHub [10]. Also, we modify the base network using MobileNet by ourselves, even though the implementation is not good as we expected.

The complete Faster R-CNN model is trained using a 4-step process:

1. Train the RPN only by initializing the base network with Imagenet-pretrained weights, then fine-tuning the RPN layers and a selected portion of the base network layers.
2. Train a separate Fast R-CNN model that doesnt share the RPNs base model. The base model is again initialized with Imagenet-pretrained weights. Use the RPN from step 1 to output region proposals to fine-tune the Fast R-CNN layers and a selected portion of the base network layers.
3. Create another RPN with base model weights initialized to those of the model trained in step 2.

Freeze those weights and train only the RPN-specific layers. The goal of this step is to re-train the RPN using base model weights known to work for the Fast R-CNN module.

4. Attach another Fast-RCNN module to the base model of the network trained in step 3. Use the RPN module to output region proposals to fine-tune the Fast R-CNN layers only, keeping the base network weights fixed to those learned in step 2.

The model is trained using the 4-step alternating procedure. Weights for layers shared by the base models are initialized to Imagenet-pretrained weights for image recognition. All other layers weights are initialized from a zero-mean truncated Gaussian distribution with standard deviation 0.01, which only generates values within two standard deviations of the mean. To keep training simple and enable caching of training inputs, I train the Fast R-CNN module on mini-batches of 64 ROIs from one image at a time instead of using 128 ROIs from two images. Considering that the smaller mini-batches by doubling the number of iterations in each step, each of the 4 training steps consists of 60k iterations at a learning rate of 0.001 and 20k iterations at a learning rate of 0.0001. SGD momentum is fixed to 0.9.

Because Imagenet-pretrained models learn low level features are useful for both image recognition and object detection [1], I freeze the weights in lower level layers to reduce training time. In the VGG16-based model, the conv1 and conv2 blocks are frozen. ResNet-50 and ResNet-101 are much deeper models so the conv3 block is also frozen for both.

6. Results and Discussion

This section describes the results we obtained from this project as well as the discussion for the results.

6.1. Experimental Results

All models are trained on PASCAL VOC 2007 train dataset and evaluated on the 2007 val dataset. The results for different base networks we used are shown in the Table 1.
To give an explicit performance of different base networks, we put some images with some bounding boxes to show the locations and classes of the objects in each image, which can seen in Figure 3.

6.2. Discussion

The Table 1 shows the metrics mean Average Precision (mAP) and the average time taken to produce object detections per image for four different base networks and also with the different number of Region of Interest (ROI). From the table, we can derive that:

Table 1. Results for Different Base Networks

Base Network	Number of ROIs	Time Consuming Per Image (s)	mAP
VGG16	64	0.1628	0.463
VGG16	128	0.1995	0.470
VGG16	300	0.3277	0.474
ResNet-50	64	0.1520	0.486
ResNet-50	128	0.1993	0.503
ResNet-50	300	0.3867	0.506
ResNet-101	64	0.1697	0.538
ResNet-101	128	0.2387	0.548
ResNet-101	300	0.3867	0.550
MobileNet	64	0.1023	0.276
MobileNet	128	0.1427	0.288

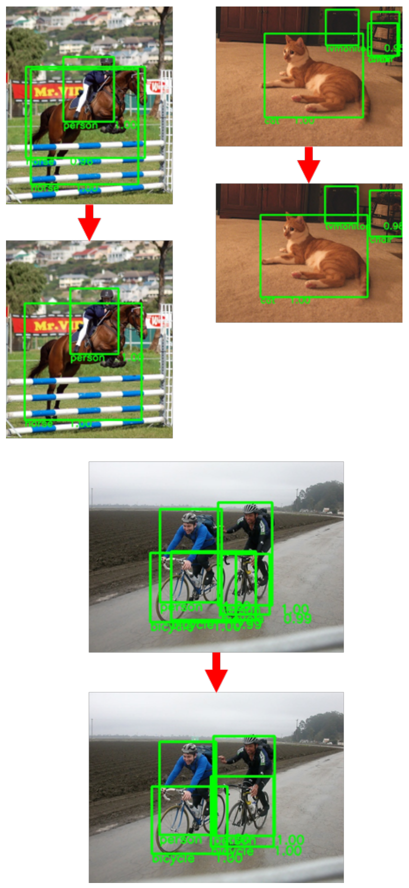
Figure 3. Object Detection between ResNet50 and ResNet101

- The base network in Faster R-CNN can be replaced by other more successful networks in image recognition which may also lead to better performance in object detection and this result validates our hypothesis.

- Less ROIs can perform better in real-time contexts but along with the mAP decreasing.

- MobileNet shows better real-time property at the considerable cost of accuracy.This could be caused by the some engineering issues because we do not have enough time to build it from scratch.

- Based on the table 1, ResNet-101 with 64 ROIs is the best choice, it has highest performance,and the processing time per image is not hight.

For the Figure 3, we only compare two different base network ResNet50 and ResNet101 which are easy for us to create the output using images. Based on the same parameters such as confident threshold, number of ROIs, learning rate and overlap threshold, we can see that ResNet101 outperforms ResNet50 in terms of the size of the bounding box as well as the probability. The reason for this is that ResNet101 has a deeper neural network which can obtain higher features of the objects.

7. Conclusion

We mainly investigate the influence of different base network and some hyper parameters on the performance of Faster R-CNN in our project. Although the results show that our assumption fails because MobileNet is not able to achieve good performance as our expectation , we learn a lot about the architecture of these widely-used networks.

References

[1] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[2] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[3] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013.

[4] Ross Girshick. Fast r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[5] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[7] Matthijs Hollemans. Googles MobileNets on the iPhone. `http://machinethink.net/blog/googles-mobile-net-architecture-on-iphone`, 2018.

[8] Pan Hu Yundong Zhang, Haomin Peng. Cs231n report. `http://cs231n.stanford.edu/reports/2017/pdfs/808.pdf`, 2018.

[9] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, pages 580–587, Washington, DC, USA, 2014. IEEE Computer Society.

[10] Ke Li. Github faster r-cnn. `https://github.com/Kelicious/faster_rcnn`, 2018.