Research on SSD base network

Yixuan Kang1,*

1Xi’an Institute of Applied Optics, Xi’an, China
*Corresponding author e-mail: kangyixuan13@163.com

Abstract. In this paper, eight neural networks are selected instead of VGG16 in the SSD model to construct the corresponding SSD model and were tested on ImageNet and VOC datasets. The accuracy, predict time and model size of the eight neural networks in object classification are analyzed. The mAP, FPS and model size of the eight neural networks as the SSD base network in the object detection are analyzed. The result can provide reference for object detection tasks under specific requirements.

1. Introduction

In recent years, convolutional neural networks have been widely used in object classification and achieved good results. In 2012, AlexNet designed by Krizhevsky et al. achieved 16.4% top5 error rate on ImageNet [1]. In 2014, GoogLeNet (Inception-V1) designed by Szegedy et al. achieved 6.66% top5 error rate on ImageNet [2], and the parameters was only 1/12 of AlexNet. The Inception-V2 and Inception-V3 networks improved by Szegedy et al. from GoogLeNet achieved 4.5% and 3.58% top5 error rates on ImageNet [3]. The VGG network designed by the Visual Geometry Group in 2014 achieved 7.32% top5 error rate on ImageNet [4]. In 2015, the ResNet designed by He Kaiming et al. achieved 4.49% top5 error rate on ImageNet [5]. In 2016, Szegedy et al. add residual modulus to the Inception network and construct the Inception-ResNet network which has a top5 error rate of 3.7% on ImageNet [6]. The Xception network designed by Chollet in 2017 has fewer parameters than InceptionV3 but has a better result [7], DenseNet designed by Huang Gao et al. in 2017 reached the performance of the ResNet network with only 1/10 parameters of ResNet [8]. The MobileNet designed by Andrew et al in 2017, uses separable convolution replaces traditional convolution to reduce model complexity and model size, can be applied in mobile or embedded devices [9]. NASNet designed by Google Brain in 2018 is designed by two AutoML to achieve the automated design of neural network model [10].

Object detection requires not only the ability to identify the object category in the image, but also the position of the object. Object detection algorithms are generally divided into two-stage and one-stage. The two-stage algorithm includes R-CNN [11], Faster R-CNN [12], R-FCN [13] et al. The ones-tage algorithm includes Inception-based YOLO [14] and VGG-based SSD [15]. Compared to the two-stage algorithm, the one-stage only needs to make one prediction, the calculation is simpler so it’s a real-time detector.

2. Another section of your paper

The SSD is based on the VGG16 network. It removes the Dense layers of fc6 and fc7 in VGG16 by conv6 and conv7, and adds four group of convolution layers at the end. Select conv4_3 (38*38), conv7 (19*19), conv8_2 (10*10), conv9-2 (5*5), conv10_2 (3*3), conv11-2 (1*1) as the feature map. SSD structure is shown in Figure 1.
The SSD sets Default Box for each pixel on the six feature maps. The default box scale of lowest feature map is 0.2 and the default box scale of highest feature map is 0.9. The default box scale on the other layer is linearly distributed between 0.2 and 0.9. For the VOC data set, consider the size of the bounding box, the scale of default box is set as \( s_k = [0.1, 0.2, 0.37, 0.54, 0.71, 0.88, 1.05] \). The aspect ratio of default box is \( a_r = [3, 2, 1, 1/2, 1/3] \) and length and width is \( w_k^a = s_k \sqrt{a_r} \) and \( h_k^a = s_k \sqrt{1/a_r} \). For the default box with an aspect ratio of 1, add a default box with the length and width of \( s_k s_k + 1 \). So each pixel will have 6 default boxes. The default box with an aspect ratio of 3 and 1/3 is deleted on the conv4_3, conv10_2 and conv11_2 layer. The information of the size of default box is shown in Table 1.

| Predict Layer | Size | Num | Boxes | Scale | 1:1 | 2:1(1:2) | 3:1(1:3) | 1:1 Large |
|---------------|------|-----|-------|-------|-----|----------|----------|-----------|
| conv4-3       | 38*38| 4   | 5776  | 0.1   | 30*30 | 43*21    | /        | 43*43     |
| conv7         | 19*19| 6   | 2166  | 0.2   | 60*60 | 85*43    | 103*35   | 81*81     |
| conv8_2       | 10*10| 6   | 600   | 0.37  | 111*111| 157*79   | 193*65   | 135*135   |
| conv9_2       | 5*5  | 6   | 150   | 0.54  | 162*162| 229*115  | 281*93   | 185*185   |
| conv10_2      | 3*3  | 4   | 36    | 0.71  | 213*213| 301*151  | /        | 237*237   |
| conv11_2      | 1*1  | 4   | 4     | 0.88  | 264*264| 373*187  | /        | 289*289   |

The output layer of the SSD consists of three parts: category probability (mbox_conf), priorbox (mbox_priorbox) and position deviation (mbox_loc). The prediction layer’s size is 8732*23. The SSD loss function consists of two parts: category loss (conf_loss) and position loss (loc_loss). The category loss is calculated by category probability and he position loss is calculated by priorbox and position deviation. The calculation process is shown in Figure 2.

**Table 1. SSD300 default box parameter**

Figure 1. SSD300 structure

Figure 2. SSD loss function structure
The SSD paper selected VGG16 as the base network. The structure of SSD is divided into six parts, the original network structure of VGG16, additional network structure in SSD300, mbox_conf, mbox_priorbox, mbox_loc and prediction layer. For the six parts, the sum of the parameters, the sum of the output tensor and the maximum size of output tensor are calculated and shown in Table 2. The sum of the parameters and output tensor in Table 2 are calculated based on the actual memory usage. The program calculate by float32 so the 1M data need approximately 3.81M of memory.

| Components in the structure | sum of the parameters | sum of the output tensor | maximum tensor size |
|-----------------------------|-----------------------|--------------------------|---------------------|
| original network structure of VGG16 | 56.13M                | 103.72M                  | 300*300*64          |
| additional network structure in SSD300 | 35.39M                | 9.03M                    | 38*38*512           |
| mbox_conf                   | 10.20M                | 2.56M                    | 8396*20             |
| mbox_priorbox               | 0M                    | 0.77M                    | 8396*8              |
| mbox_loc                    | 2.04M                 | 0.38M                    | 8396*4              |
| predict                     | 0M                    | 1.02M                    | 8396*32             |
| Total                       | 103.76M               | 117.48M                  | 300*300*64          |

3. Base Network

The original SSD uses VGG16 as the base network and the author pointed out that other network can be also used as base network. In recent years, many neural networks are proposed and achieved better result in some aspects of object classification than VGG16. Usually, the higher accuracy of the network, the slower speed in prediction. Sometimes we concern more about the speed and calculate consumption and sometimes we are more interested in the accuracy of the prediction. In this paper, several typical networks different from accuracy, prediction time and model size are selected to replace the VGG16 as the base network of SSD to analysis the performance of them.

ResNet-SSD: ResNet use the residual module which effectively solved the problem of precision reduction of deep network [5]. In the paper, there are three structures with layer of 50, 101 and 152. The ResNet152’s computational complexity is 3.1 times of ResNet50 but the Top5 correct rate on ImageNet only increased by 0.7%, so in this paper, ResNet50 is used as the base network. The biggest difference between ResNet and VGG is that ResNet uses global average pooling replace the fully connective layer at the end of the network. Taking the last few layers of VGG16 (300*300 input) as an example, the block5_conv3 has an output size of 18*18*512, and the block5_pool has an output size of 9*9*512. Since the block5_pool only performs pooling processing on the block5_conv3 and does not acquire new features, it is necessary to add two new convolution layers after the block5_conv3 in the SSD structure to generate the third feature map. For ResNet50 (300*300 input), the stage4_block_f’s output size is 19*19*1024 and the stage5_block_a’s output size is 10*10*2048. There is a residual block between the stage4_block_f and stage5_block_a. So the output of stage4_block_f and stage5_block_a can be used as the second and third feature map in SSD. Most of the networks mentioned later are similar to ResNet that the first three feature maps can be directly choose from the network structure. The last three feature maps are added by three sets of convolutional layers.

InceptionV3-SSD: InceptionV3 is based on the Inception network, which splits the two-dimensional convolution in the Inception network into two one-dimensional convolutions and one nonlinear layer to reduce a large number of parameters while reducing over-fitting. The InceptionV3 network selects mixed2, mixed7, and mixed8 as the first three feature maps.

Xception-SSD: The Xception network is improved from InceptionV3, which uses a separable deep convolution instead of the traditional convolution method and adds a residual connection channel to the network to accelerate the convergence rate of the network. Choose block4’s output size 37*37*728 as the first feature map, block13’s output size 19*19*1024 as the second feature map, block14’s output size 10*10*2048 as the third feature map.
InceptionResNetV2-SSD: The InceptionResNetV2 network, which adds the Inception module in the ResNet, performs well in the object classification. It improves the training speed and accuracy of the network but the complexity is higher than ResNet. The outputs of the block 35_10, block 17_20, and reduction-b block are used as the first three feature maps.

MobileNet-SSD: MobileNet is a lite CNN model proposed by Google which solves the problem of slow response and large memory usage when traditional models lay out on embedded platforms or mobile devices. MobileNet use the width multiplier and resolution multiplier as hyper-parameter to proportionally reduce the number of channels and the size of the feature map. The MobileNet has less model parameters and less computational complexity at the expense of certain precision. Use MobileNet's conv_pw_5, conv_pw_11, and conv_pw_12 as the first three layers of the feature map. Improved from MobileNet, MobileNetV2 adds a 1*1 convolution to reduce the dimension and then increase the dimension before the 3*3 convolution layer to reduce the amount of parameter in the model. The block_5 output, block_12 output, and block_13 of MobileNetV2 are choosed as the first three feature maps.

DenseNet-SSD: DenseNet's method is similar to ResNet. ResNet connects the network module to the previous module but DenseNet connects the network module to all previous modules. DenseNet reuse feature through bypassing and achieve better performance than ResNet with fewer parameters and computations. DenseNet has four type of layers range from 121, 169, 201 and 264. In the experiment, the network of 121 and 169 layers are tested. The network of 121 layers adopts conv3_block12, conv4_block24, and conv5_block1 as the first three layers of feature maps, and the 169 layer adopts conv3_block12, conv4_block32, and conv5_block1 as the first three feature maps.

NASNet-SSD: NASNet used two AutoML to design base module and then stack them into a big module. AutoML first searches for the base module on the CIFAR-10 dataset and then transplant the module into the ImageNet dataset. The NASNet has improved the top1 accuracy by 1.2% over the network architecture designed by humans in the ImageNet dataset. Use NASNet’s concatenate of normal_concat_4 and normal_concat_5, concatenate of normal_concat_11 and normal_concat_12 and normal_concat_18 as the first three feature maps. NASNetMobile is a light NASNet with 1/16 parameter, using the concatenate of normal_concat_2 and normal_concat_3, the concatenate of normal_concat_7 and normal_concat_8, and the normal_concat_12 as the first three feature maps.

4. Statistics of Network Parameters

This paper selects VGG19, ResNet50, InceptionV3, Xception, InceptionResNetV2, MobileNetV2, DenseNet121, DenseNet169, NASNetLarge and NASNetMobile for comparison. Table 3 counts the ten networks in the memory of network weights(excluding top classification layer), the memory of the feature map, the predict time per-image and the classification accuracy of the network on ImageNet.

| Base network       | memory of weights | memory of feature map | Tesla T4 predict time | ImageNet Top1 accuracy |
|--------------------|-------------------|-----------------------|-----------------------|------------------------|
| VGG19              | 76.39M            | 112.52M               | 31.0ms                | 75.6%                  |
| ResNet50           | 89.98M            | 246.83M               | 26.1ms                | 77.1%                  |
| InceptionV3        | 83.17M            | 148.66M               | 39.9ms                | 78.8%                  |
| Xception           | 79.58M            | 240.25M               | 34.3ms                | 79.0%                  |
| InceptionResNetV2  | 207.28M           | 380.21M               | 90.6ms                | 80.1%                  |
| MobileNetV2        | 8.61M             | 138.05M               | 10.6ms                | 72.0%                  |
| DenseNet121        | 26.85M            | 331.35M               | 34.0ms                | 75.0%                  |
| DenseNet169        | 48.23M            | 395.26M               | 53.4ms                | 76.2%                  |
| NASNetMobile       | 16.29M            | 176.37M               | 34.5ms                | 74.0%                  |
| NASNetLarge        | 323.93M           | 846.74M               | 144.3ms               | 82.7%                  |

Figure 3a) shows the relationship between mainstream neural network’s (add VGG16, ResNet101, ResNet152, GoogLeNet, MobileNet, DenseNet201 and DenseNet264 seven neural networks) Top1
accuracy on ImageNet and model parameters. Figure 3b) shows the relationship between the predict time per-image for the neural network on Tesla T4 and the model parameters.

Figure 3. a)Left: Relationship between mainstream neural network’s Top 1 accuracy on ImageNet and the model parameter (single-crop, original input, 1000 category); b)Right: Relationship between the predict time per-image for the neural network and the model parameters (Tesla T4, batch_size=1)

5. Experiment

The SSD module use data augmentation which can effectively increase the size of dataset and improve the classification ability of the model. The data augmentation can be divided into two types. The first type is geometric transformation of the image, including random horizontal flipping, cropping, enlarging the image and changing the aspect ratio of the image. The second type is optical transformation of the input image, including randomly changing the brightness, contrast, saturation and chrominance of the input image. This paper adds random noise and random erase in data augmentation. Random noise can weaken the sensitivity of the neural network to high-frequency information and random erase can better deal with the situation of the object occlusion [16].

This Paper test the performance of SSD models based on different networks in object detection on VOC2012 and IMAGENET datasets. The VOC2012 dataset contains 20 categories of objects, 11540 images and 27450 objects. Each image contains one or more objects, and the location of the object bounding box is tagged and provided as an xml file. The ImageNet dataset consists of 1000 categories of objects, each categories have 1300 training images, 50 validation images, and 100 test images. ImageNet images usually contain only one object. The training set of the ImageNet only labels the border for about half of the images. Due to the huge amount of data in the ImageNet dataset, this paper selected 10 categories from the ImageNet for testing. The ten categories include aircraft carrier, container ship, lighthouse, port, dyke, viaduct, armored vehicle, rifle, military aircraft and missile. The training set contains 5,382 pictures and the test set contains 1,500 pictures.

Load a pre-trained model on ImageNet before training. Due to the limitation of the training set, the use of the pre-training model can solve the problem of difficulty in feature extraction. Using the pre-training model can also accelerate the training process and reduce consumption of computing resource. The convolutional layer added by the SSD after the base network adopts the he_normal initialization method. The SSD uses the SGD optimizer. The update of SGD depends on the current batch, which will lead to unstable update which will cause the loss function oscillating and training convergence slow. The Adam optimizer stores the average of the square exponential decay of past gradients. The update depends on the recent rounds of batches. The update is more stable and the convergence is faster, but the memory usage is higher and the convergence point is not as good as SGD. In order to speed up the training, Adam is used instead of SGD as the optimizer. The FPS and mAP of the model are calculated on the test set after the training is completed. FPS refers to the number of images that the model can process per second, and mAP refers to the average of the AP values for each category.

Using the Tensorflow 1.10.0 as backend, use Keras to write the SSD architecture, cudnn version 7.3.1. Accelerate computing with Nvidia Tesla T4 GPU(16G memory,7.5 computing capacity), CPU(16G memory). Training batch_size set to 32 (the base network of DenseNet121, DenseNet169 and InceptionResnetV2 with batch_size of 16), the initial learning rate is 0.002, the learning rate is
multiplied by 0.95 for every 20,000 pictures, and the learning rate no longer change after the learning rate is smaller than 0.00001. Set batch_size to 1 when calculating FPS. When calculate mAP, the 200 prediction boxes with the highest prediction probability are selected as the sample set in each picture, and the IOU is set to 0.5. Table 4 summarizes the FPS and mAP under the ImageNet-10 and VOC2012 datasets for different SSD models. The test found that the accuracy of the SSD model is largely related to the accuracy of the base network.

Table 4. SSD300 performance under various module

| Module               | ImageNet-10 mAP | VOC2012 (train) mAP | FPS  | Module size |
|----------------------|-----------------|---------------------|------|-------------|
| ResNet50-SSD         | 73.3%           | 66.9%               | 24.6 | 77.3M       |
| InceptionV3-SSD      | 74.5%           | 66.3%               | 20.8 | 58.5M       |
| Xception-SSD         | 75.7%           | 67.0%               | 21.2 | 113.6M      |
| InceptionResNetV2-SSD| 79.0%           | 70.4%               | 12.6 | 138.6M      |
| MobileNetV2-SSD      | 69.9%           | 57.5%               | 39.4 | 29.9M       |
| DenseNet121-SSD      | 70.4%           | 59.8%               | 25.4 | 35.3M       |
| DenseNet169-SSD      | 72.9%           | 60.2%               | 21.5 | 43.6M       |
| NASNetMobile-SSD     | 70.0%           | 58.8%               | 20.8 | 44.2M       |

Object detection based on neural network can be divided into two main methods, one is the object detection algorithm based on regional proposal, including RCNN, SPP-Net, Fast-RCNN and Faster-RCNN. SPP-Net extracts only one feature map from the neural network in one prediction compared to the RCNN network extracts a feature map for each region proposal to shorten the prediction time. Fast-RCNN improves from SPP-Net and adds bounding regression to achieve end-to-end training. The second method directly predict the bounding box and confidence of multiple categories, including YOLO, YOLOv2, YOLOv3, SSD, and so on. YOLO uses feature maps on top of the neural network to predict categories and bounding boxes. The SSD is similar to YOLO but incorporates the idea of regional proposal in RCNN. The SSD set a fixed-size regional proposal box into the network. YOLOv3[17] simplifies the basic network of YOLO and adopts multi-scale training approach. YOLOv3 mainly improves the accuracy of small object detection. Table 5 summarizes the mAP (IOU=0.5) and FPS (batch_size=1) of the above object detection models on the VOC2007, VOC2012, and COCO data sets. The COCO dataset contains 80 object categories. The image in the COCO dataset contains 7.7 objects in average. The COCO data set is evaluated by AP@[0.5:0.95]. The method is calculating the average of mAP by IOU from 0.5 to 0.95.

Table 5. Performance of other target detection models on VOC and COCO dataset

| Module       | VOC2007 mAP | VOC2012 mAP | COCO train mAP | FPS Titan X | input size |
|--------------|-------------|-------------|----------------|-------------|------------|
| Fast-RCNN    | 70.0%       | 68.4%       | 19.7%(train)   | 0.5         | 600*600    |
| Faster-RCNN  | 73.2%       | 70.4%       | 21.9%          | 7           | 600*600    |
| YOLO         | 63.4%       | 57.9%       | /              | 45          | 448*448    |
| YOLOv2-544   | 78.6%       | 73.4%       | 21.6%          | 40          | 544*544    |
| SSD300       | 74.3%       | 72.4%       | 23.2%          | 46          | 300*300    |
| SSD512       | 76.8%       | 74.9%       | 26.8%          | 19          | 512*512    |
| R-FCN ResNet101 | 79.5%   | 77.6%       | 29.2%          | 9           | /          |
6. Summary

The impact of the base network on the SSD model is significant. Different types of networks can be selected as the base network in different situations. Using ResNet and Inception as the base network is more accurate. MobileNet-SSD can be applied to some embedded platforms for faster detection. But on the whole, no matter what kind of front-end network is used, the original characteristics of the SSD model will still be retained so the detection accuracy of the small target is not as good as the R-CNN method.

7. Acknowledgments

This work was supported by Xi’an Institute of Applied Optics.

References

[1] Krizhevsky A, Sutskever I, Hinton G. ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems, 1(2012), 1097-1105.
[2] Szegedy C, Liu W, Jia Y, et al. Going Deeper with Convolutions. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6(2015), 1-9.
[3] Szegedy C, Vanhoucke V, Ioffe S, et al. Rethinking the Inception Architecture for Computer Vision. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 12(2015), 2818-2826.
[4] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556, 4(2015).
[5] He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 12(2016), 770-778.
[6] Szegedy C, Ioffe S, Vanhoucke V. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. AAAI Conference on Artificial Intelligence, 2(2016).
[7] Chollet, François. Xception: Deep Learning with Depthwise Separable Convolutions. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 7(2017), 1800-1807.
[8] Huang G, Liu Z, Laurens V D M, et al. Densely Connected Convolutional Networks. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 7(2017), 4700,4708.
[9] Howard A G, Zhu M, Chen B, et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv:1704.04861, 4(2017).
[10] Zoph Barret, Vasudevan Vijay, Shlens Jonathan, Le Quoc. Learning Transferable Architectures for Scalable Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6(2018), 8697-8710.
[11] R. Girshick, J. Donahue, T. Darrell and J. Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. IEEE Conference on Computer Vision and Pattern Recognition, 11(2014), 580-587.
[12] Ren S, He K, Girshick R, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis & Machine Intelligence, 39(2015), 1137-1149.
[13] Dai J, Yi L, He K, et al. R-FCN: Object Detection via Region-based Fully Convolutional Networks. Advances in Neural Information Processing Systems, 5(2016), 1-9.
[14] Redmon J, Divvala S, Girshick R, et al. You Only Look Once: Unified, Real-Time Object Detection. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6(2016), 779-788.
[15] Liu W, Anguelov D, Erhan D, et al. SSD: Single Shot MultiBox Detector. European Conference on Computer Vision, 10(2016), 21-37.
[16] Zhong Z, Zheng L, Kang G, et al. Random Erasing Data Augmentation. arXiv:1708.04896, 8(2017).
[17] Redmon Joseph, Farhadi Ali. YOLO9000: Better, Faster, Stronger. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 7(2017), 6517-6525.