Large-scale Parallel Deep Convolutional Neural Network on Image Classification

ZHONG RUI

Graduate School of Library, Information and Media Studies University of Tsukuba

March 2016

Table of Contents

1. Introduction

In the area of machine learning and pattern recognition, artificial neural network is considered as a competitive approach to automatically extract the features of the input data and finish recognition task. The original inspiration of artificial neural network is to simulate the biological neural network consisting of millions of neurons in animal's brain [1]. This intuition suggests one can construct a neural network with several neuron layers, each layer is equal to a nonlinear mapping function with a weight matrix. One neuron layer could be trained by calculating the derivatives of specified loss function to the weight, and update the weight value along the direction towards minimum loss. This is called the gradient descent method. By stacking up several neuron layers one can construct an artificial neural network, and adopt the dynamic programming to spread gradient information from higher layer to lower one, which is called back-propagation [2], the whole neural network could be trained efficient with this algorithm. The most important property of neural network is that it could extract the features at different levels automatically, on the other hand traditional methods require researchers to decide the data representation method, such as using SIFT [3] or HOG [4] feature extractors. Neural network would automatically extract the features of the input data and store the learned features as weight matrices. This property decreases the influence of the human factors during the learning process, and guarantee the ability of generalization of the network.

Convolutional neural network (CNN for short) is proposed by LeCun [5] and it shows great potential for visual recognition tasks. The major difference between CNN and traditional multi-perceptron neural network is the sharing weight in the convolutional layers. Most neurons in the same convolutional layer of CNN shares the same weight values. This property encourage filters to detect local patterns in the input image, which also satisfy the assumption that the most important feature for object recognition is hidden in locality. Even before the deep learning revolution in machine learning, CNN has already achieved significant results on image classification tasks, such as handwritten digit recognition [6], in addition it kept state-of-the-art results on small image data sets [7]. According to LeCun [5], they observed that the combination of convolutional layers and down sampling layers achieved the best performance. They inserted several down sampling layers into convolutional layers to create a CNN. The reason is that CNN has the similar structure and functionality compared to visual cortex neural networks found in cat's brain, which is mainly composed by simple cells, complex cells and hyper-complex cells [8]. This convolution-pooling structure has been adopted as the basis for deep neural networks specialized for visual tasks for a long period, also inherited in most visual deep learning researches.

Deep architecture has been exploited on neural network models and showed outstanding performance with tasks of image classification [9]. Adopting this novelty to traditional CNN further improves the performance of image processing. Though CNNs show great advantages in this area, it is until recently, due to the development of GPU computing, one could apply this architecture to a single machine and achieve good recognition rate while not taking too much time to finish training phase. Before deep learning revolution it has been proved very difficult to train a deep neural network without

unsupervised training [10], also deep architectures require more computing resources. Because of these limitations deep neural network were considered as unnecessary for image recognition tasks, until large image data set such as ILSVRC2010 [11] has been used as benchmark. Also recently researchers suggested the new activation function ReLU help back-propagation to spread the gradient information to deeper layers, due to its linearity for the positive input [12]. Because of the scalability and great ability for generalization, deep neural network has been widely adopted in variant recognition tasks. Recent research focused on the architecture of deep neural network, the width and depth, also local structures within one layer.

This research follows the inspiration that increasing the depth and well-designed local structure could improve the performance of deep neural network. Thus, we analyzed the local structure within one layer in the network, changed the size of filters, the number of channels and topology between convolutional layers. In addition, we exploited the fact that there is a receptive field in animal's retina cells, which take advantages of contrastive inputs, i.e. only when the incoming light hit the central part and avoid the contour part of one retina cell, this cell would fire a neural spike to others. After implemented this feature as a pooling layer that strengthen the contrastive patches for the input data, we see how this let us achieve better results.

2. Related work

Deep neural network was popularized since seminal work by Szegedy et al. [13]. They proposed a general model for CNN architecture to achieve state-of-the-art results on ILSVRC 2012. Their main contribution was implementation of CNN on multi-GPU machine and optimization techniques for programming. Efficient implementation and carefully design for the network would let them train more efficiently than others, additionally the combination of variant techniques, such as using ReLU activation function, inserting dropout layers, adopting multi-GPU training, preparing data augmentation and proper choice of learning rate update method let them achieve the best accuracy among the participants. These methods became widely-used, and deep learning neural network showed the potential for scalability to overcome the difficulties of traditional methods. Since then many deep learning implementations have been brought to other tasks, such as object recognition, speech recognition, image segmentation and so on.

The next milestone for deep neural network on image classification is led by the researchers from Google in 2014, named project as GoogLeNet [14]. They increased the depth of the neural network aggressively, finally achieved impressive performance with a 22-layers deep network, this novelty has also been proved very effective by other team. To draw the conclusion, for large data set such as ILSVRC 2012, increasing depth of network till 20 layers is very effective. Also such a deep network could almost achieve human-level recognition results with a dense data augmentation training and testing. Furthermore, GoogLeNet proposed the concept of inception module. Instead of using one convolutional layer for one level of the network, they combined several convolutional layers having different filter size together, also with pooling layer into one level. Their concept is that for one level

of the network, it is natural to consider detecting patterns at different size, thus they set up convolutional layers with filter size of 1*1, 3*3 and 5*5. With 1 stride for convolution and padding the input properly, the outputs of these convolutional layers will be at the same size. This novelty also seems reasonable and natural for one-layer convolutional network. The last contribution is that they widely adopted convolutional layers with 1*1 size filter before convolutional layers of larger filter size, which they explained with dimensional reduction. Because the numbers of convolutional layer output could be configured, by setting these 1*1 convolutional layers one can aggregate the feature maps together, thus prevent the dimension increase too rapidly while increase network depth. It is worth mentioning that recent researches [15] also use 1*1 convolution to replace fully-connected layers for spatial flexibility.

Another remarkable deep learning network in 2014 is VGG-net, developed by Simonyan and Zisserman [16]. Similar to GoogLeNet they also discovered deeper neural network could achieve better accuracy on large image set. Their proposed model has 19 levels for convolutional and fully-connected computation. In contrast to GoogLeNet, VGG team adopted the convolutional layers with the same size at 3*3 for all levels. They described this strategy as for increasing the network depth as much as possible. They also considered two 3*3 convolutional layers have the same function as one 5*5 layer but with less parameters to train. In addition, they decrease the use of pooling layers, preventing down-sizing the feature map too quickly, which also contribute to increasing the depth of network. Thus their model mainly consists of dense small size convolutional computation, resulting in a very deep model. It is surprising that mainly by increasing depth, they have achieved almost the same accuracy as GoogLeNet, though latter has more carefully designed network details. In our experiment, because of lacking dimensional reduction technique, the memory consumption of VGG-net is obviously much higher than GoogLeNet model, and computation also heavier. Another contribution in their research is about the training methodology of the network, including most details about image pre-process, multi-size training and dense cropping test techniques. According to our experiment, such training methodology is crucial for practical level accuracy, though these techniques could multiply the training time, they would roughly increase accuracy for about 3% to 5%. Due to fewer implementation difficulties, many researcher continue improving VGG-net model for object recognition tasks. Some achieved surpass human-level image classification [17].

3. Network modification

3.1 Modularization

Though VGG-net is widely investigated because of its simplicity, in our research we used the architecture of GoogLeNet, because the memory consumption of the latter is much lower. Carefully designed architecture of GoogLeNet contributed to reducing the training time and memory consumption, while achieving the same performance. Also assumption of our research is related to the local structure of the neural network, which we name as the modularization. This inspiration is followed with the inception module of GoogLeNet [14]. They suggested that one should consider the different sizes of patterns hide in the feature map. For example, in the lower levels close to the input image network

should capture the large size patterns, thus using 5*5 or 7*7 convolutional layers will be better. But in the middle levels one should consider using mainly 1*1 and 3*3 convolutional layers, because in this level most features have abstract meaning and feature maps also become sparse. Though it has not reached high levels that feature map could be completely represented by fully-connect layer or 1*1 convolutional layer, so one should adopt 1*1 and 3*3 convolution at the same layer to capture different size features, then concatenate the outputs.

Based on such inspiration, GoogLeNet use inception modules for each level of the network instead of single convolutional layer. In one module they perform 1*1, 3*3 and 5*5 convolutions also with max-pooling. By setting stride of convolution to 1 and padding the input, one can guarantee the outputs of these filters have the same size. Using such module results in extracting general local structure, while by configuring the numbers of filters one can change the size of patterns needed to be learned. The figure of one module is shown in Figure 3.1.

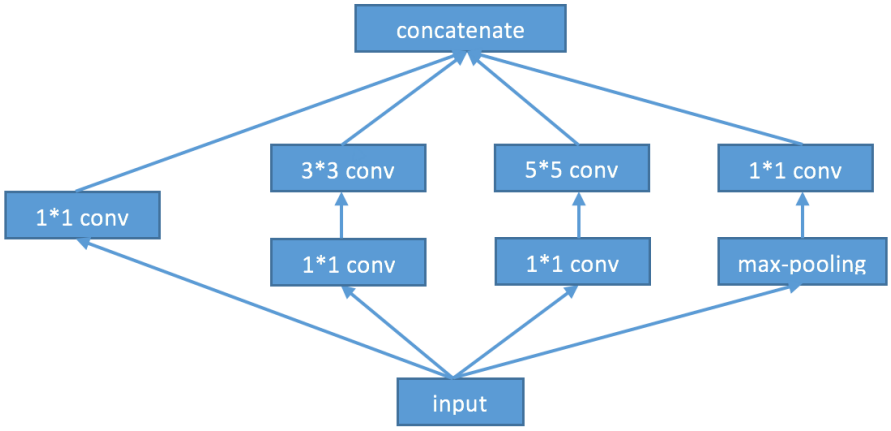

Figure 3.1 Inception module

It is noticeable that they also followed the trend of using 1*1 convolutions to do the dimensional reduction. Considering that on higher levels 3*3 and 5*5 convolution could be expensive for computing, apply 1*1 convolution before them would help to aggregate the overabundant channels together. Also it will make unimportant feature maps sparse and further reduce the cost. In our research this feature is considered as an important method to control the computation cost, as well as a pruning strategy that provide us more possibility to exploit the topology of deep neural network. That is to say, by inserting 1*1 convolutions between the branches, we encourage the network to shape itself automatically. Since it is known that deep learning network has great potential for extracting the feature from input automatically, we expected that with proper degree of freedom the network would also search for the best topology automatically.

Additionally, we noticed in other researches that the convolution at 5*5 size is considered as unnecessary. This is because two 3*3 convolutional layers have the same pattern recognition effect and less parameters to learn [16]. Thus we propose a local module structure without 5*5 convolutions. Instead we stack up 3*3 convolutions and concatenate their output feature map together. The intuition

is straight forward that by increasing the depth of the network could improve the performance, along with reducing the convolutional computing cost. Especially Szegedy et al. [14] suggested that 5*5 convolution in high levels are very expensive. We therefore propose insert another 3*3 convolution layer already existing one. We proposed our canal module as indicated in Figure 3.2.

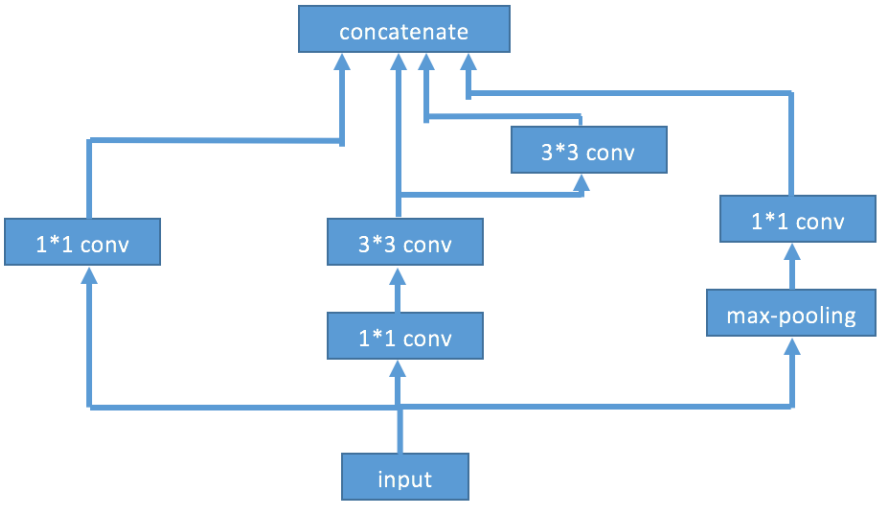

Figure 3.2 naïve canal module

Another feature of our proposal is using 1*1 convolutions as pruning and information aggregate method in the module. We could add other branches in the module with 1*1 convolution to reduce the dimension of feature maps. Also it is noticeable that in next level 1*1 convolution is performed before 3*3 convolution, which aggregates the results together from different channels. Thus the output of this new module consists of 3 parts. The first part comes from 1*1 convolutions and could be seen as the weighted input information; the second part comes from once 3*3 convolutional computing; the last part comes from second 3*3 convolutions which is aiming at capturing advanced features. During the experiment we configure the numbers of 1*1 convolutions more than the others parts, because we consider them as a prune method which encourages the network to become sparse. This could help the network to form different topology depending on input data, while we add more branches to the network than actual need. It could be difficult to train at the beginning, but our goal is to research about the topology, this strategy is worth trying.

3.2 Contrastive learning

Another important feature of our proposal is extracting the contrastive information, which react like real visual cortex neurons having receptive fields. Receptive field [18] is one of the most important features existing in the visual cells of retina and cortex. For neurons in those area most of them have two parts which can receive stimuli, generally the circle central part and the remaining part around central circle. Scientists found only when one part receives stimuli while another part doesn't, the neuron will be activated and generate spikes (action potentials) frequently. For example, some of neurons at retina activate when light stimulates the center while the surrounding part has not been

stimulated. They are called center cells. Another type behaves the opposite way, only fire when light stimulate the surrounding part. They are called off center cells. Both center cells and off center cells will not be activated in the total dark or bright environment where center and surrounding area are stimulated by light at the same time. That is to say, the existence of receptive field suggests neuron should only be activated when the information is contrastive within their sensory space. The explanation of this feature suggests during the evolution process, detecting the change in the environment was the key to survival, thus sensory system evolved to activate only when receiving contrastive information. Notice that deep convolutional neural networks could automatically extract features in the images, and after training one can find the filters of convolutional layers will contain contrastive weight values if visualize them. This also proved the importance of contrastive information during recognition. Our visualization of filters in lowest layer is indicated in Figure 3.2.

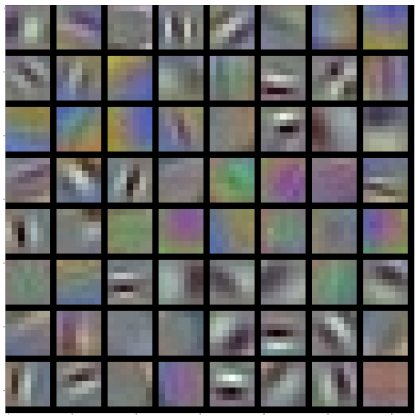

Figure 3.2 first conv layer filters

In this work we aim at combining contrastive learning method into existing CNN architecture. We consider this could improve the accuracy along with accelerating the learning process. We also hope by reducing convolutions on feature maps contain less contrastive information would contribute to improving performance. Our implementation of contrastive learning is to use a new type of pooling layer, which we called contrastive pooling. First we separately apply max-pooling and min-pooling on the same patch, then we compute the output by subtracting the result of max-pooling map with min-pooling map. For $\boldsymbol{x}$ is the input local patch of the input, the contrastive pooling is calculated as follows.

$$F(\boldsymbol{x}) = a * \max(\boldsymbol{x}) - (1-a) * \min(\boldsymbol{x})$$

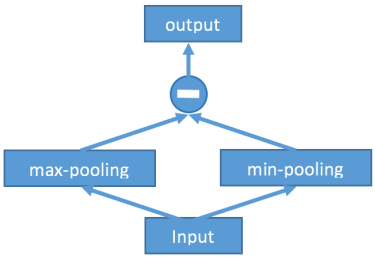

Figure 3.3 contrastive pooling

Though this is a naïve implementation of contrastive learning, it would guarantee the output feature map will response at local area with contrastive input. Additionally, this is easy to implement using existing libraries. Considering the performance of computation, implementing contrastive learning as pooling layer is more practical than adding regularization items to convolutional layers. While latter is more intuitive and effective by adopting Lagrange multipliers to update equation of convolution filters, we can also encourage convolutional filters to learn contrastive patterns. However, the implementation could be cumbersome and inefficient based on present deep learning frameworks. Moreover, complicated convolutional computing would have deteriorating impact on performance. Our implementation could not contribute to form fixed-shape receptive fields, but also amplify contrastive signals along with training the neurons that contribute to this contrastive output. Because we implement this new pooling layer with max-pooling and min-pooling layers, through back-propagation algorithm we would only train the neurons contribute to the max and min value. Computation cost is nearly the same as a single max-pooling layer.

4. Experiment

4.1 Architecture

Now we introduce the whole architecture of our model. We choose to build a very deep architecture similar to GoogLeNet [14]. Specifically, after first 3 convolutional layers, we exchange the inception modules with our canal modules. It can be seen that we reduced the filter size of these convolutions, in order to keep more information to the high level of the model, along with increasing the depth. We have 22 weighted layers in total, which is the same as GoogLeNet. Also we adopted the early output branches for fast training [14]. This follows the GoogLeNet, which will contribute to spread gradient information for back-propagation, by inducing such output branches to the lower levels. This is the basic architecture for our model but not the only one. We experimented many different combinations based on this network. Since we modularized the network with several feature extracting convolutional layers, canal module and output module, we exploit the topology of deep neural network by adding and deleting branches.

Also based on the canal module, we experiment the effect of adding new 3*3 convolutions at higher level or as a new branch. Our principle follows GoogLeNet [14] that the computing cost of network should be below 1.5 billion ops for one input to go through to output. This guarantee that we can apply this architecture to practical implementation instead of in-lab research. We will describe the details of these modules in the result section.

Before every output branches we also insert max-pooling layers to reducing the computation cost. Inside the canal module we adopt contrastive pooling instead of max-pooling. With overlapped filters we guarantee the outputs have the same size as convolutional layers.

4.2 Training methodology

All our experiments are trained and tested on ILSVRC 2014 data set, which is almost the same as ILSVRC 2012. We only tested on image classification task so far, and plan to continue our research on object detection task in future research.

It is worth mentioning that we did not adopt any dense cropping methods to apply data augmentation for classification task, because such techniques would require large amount of computing resources, and increase the training time dramatically. For example in [14] researchers increased computing cost for 10 times than the original model, which only improved accuracy by 0.94%. In our research we could also use such training method for increasing accuracy, but we consider these method is not only time consuming but also impractical for application of deep neural network. Our aim is increasing the accuracy by improving the topology of deep architectures, rather than increasing computing cost which would limit the deployment of network on embedded systems.

Thus we used original GoogLeNet one crop model as the baseline, which is implemented with open source deep learning library Caffe [19]. The only pre-process of image data is mean subtraction, i.e. subtract the mean value of all input to every pixel. Since the width and height of original data set is variant, we simply resize the longer edge of all input images to 224 pixels while keeping the image ratio unchanged, and padding the stride edge with noise pixels. Also in order to increase the accuracy we applied the bounding boxes to the images, because for a large amount of the images in ILSVRC 2012 data set, we found unrelated background interferes the object of images, e.g. in images under the class 'fish' one could found many images contain the fishing man. We consider these images obviously have different abstract feature from those only containing one fish filling the whole image. For image classification task this method could improve the accuracy a little bit, and we consider the localization of the object is not contained in this task.

Our learning rate update method follows polynomial learning policy, which is known to be the faster learner for the architecture. This method greatly reduced the iterations until the network converges while achieving almost the same accuracy as the original SGD policy. The equation is described below.

$$F(i) = l * (1 - \frac{i}{m})^p$$

In this equation, i is the present iteration, $m = 2400000$ is the max iteration, $l = 0.01$ is the initial learning rate and $p = 0.5$ is the decay power parameter. It can be seen when reaching max iteration, the learning rate becomes 0.

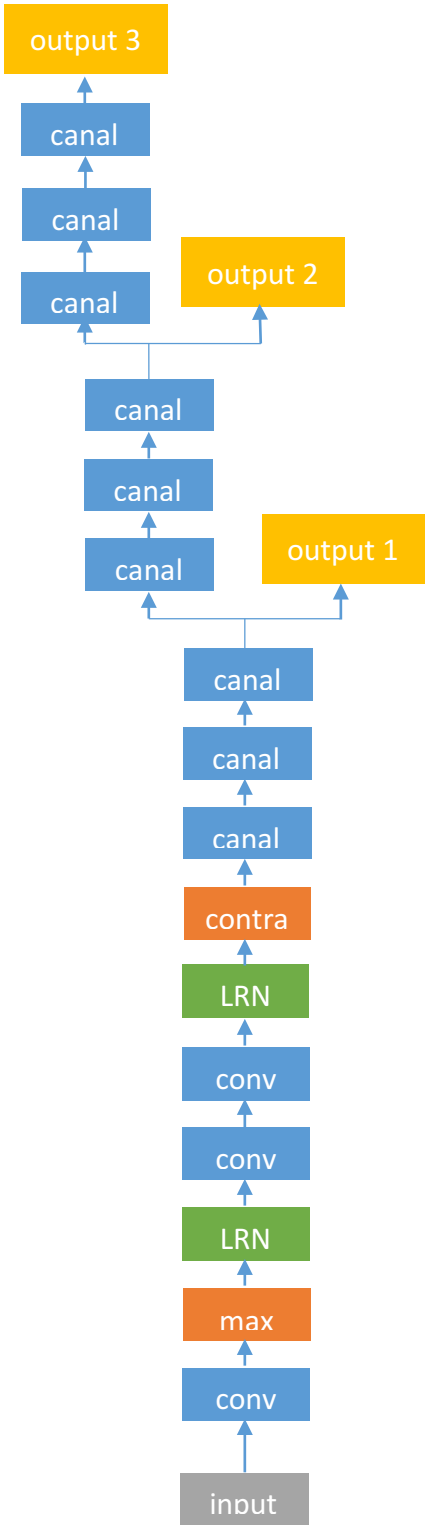

figure 4.1 network architecture

4.3 Results

In the first experiment we focused on topology of deep neural network, and inspect the influence of each part in inception module of GoogLeNet [14]. We observed that by applying cropping and resizing of training images using their bounding-boxes would help accelerate the training process, which greatly shortening the training time for the network to reach 75% accuracy. In all experiments we used these revised training data and used GoogLeNet we implemented on our environment as the baseline, and achieved a final accuracy of 89.39%. Due to experiment resource limitation we could only achieve results for early training phases. We stopped these models at nearly 420000 iterations (about 11 epochs). To our surprise, during the early phase these modifications did not bring big impact on network accuracy.

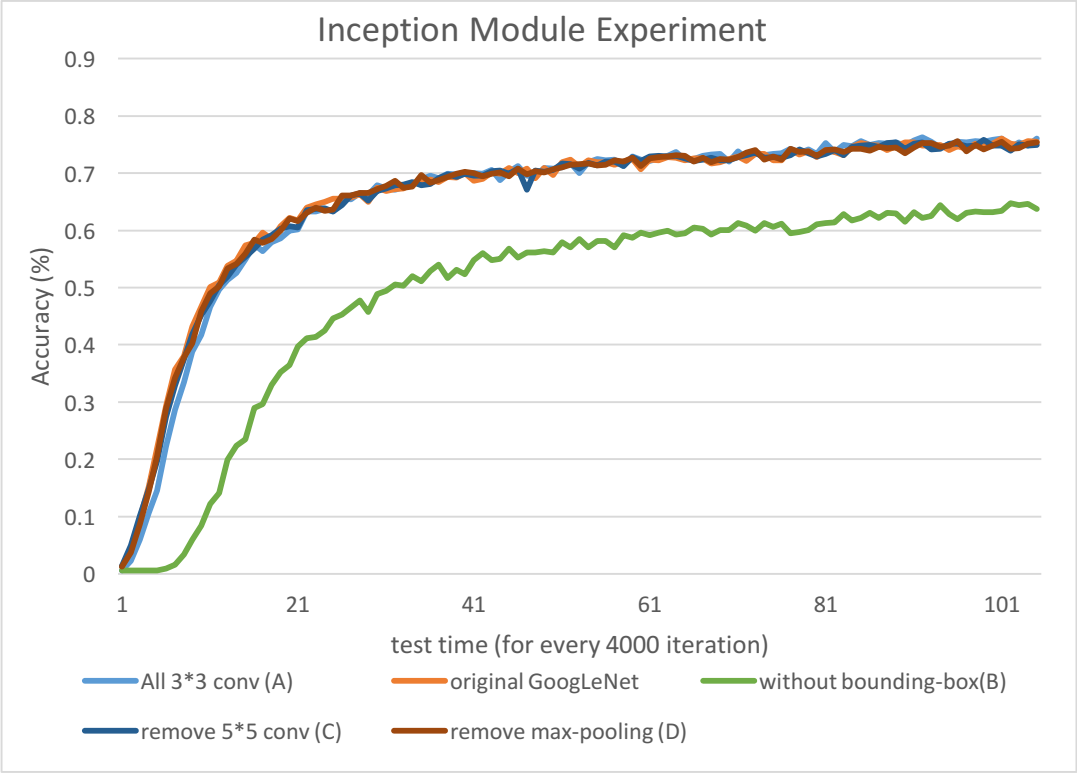

Figure 4.1 experiment on inception module

The training process is illustrated in Figure 4.1. The x-axis indicates the training process, which tests accuracy for every 4000 iterations during training, and y-axis is test accuracy. Based on original GoogLeNet, we first tested the learning process not using bounding boxes to crop and resize the image (model B), it shows at the beginning accuracy increase slower. Then we tested the network without the 5*5 convolutional layers (model C) or max-pooling layers (model D) in inception module, which is to our surprise achieved almost the same accuracy as the original network. At last we experimented exchanging 5*5 to 3*3 convolutions (model A), which increased 0.5% recognition rate after 420000 iterations, while original GoogLeNet also reached 75% accuracy.

For the second experiment we inspect the effect of 5*5 convolution in inception module. We

changed all 5*5 convolution layers in inception module to 3*3 convolutions. We observed almost no difference on final accuracy, though it decreased for 0.4% but also indicated that 3*3 convolution can replace 5*5 convolution in higher layers. We also proposed a model where two 3*3 convolutional layers is replaced by one 5*5 and we denote this as CanalNet-A. Finally, we experimented topology described in Figure 3.2, it has been shown that adopting two stacked 3*3 convolutional layers can improve performance with fewer weight parameters. We denote this model as CanalNet-B. These experiment are listed in Table 4.2. Notice that we finished training this time, which took nearly 40 epochs for our fast learning method.

model	top-5 error
GoogLeNet-reported [14]	10.07%
GoogLeNet-ours	10.61%
CanalNet-A	9.58%
CanalNet-B	[1]*

Table 4.2 module modification

As we could see by removing 5*5 and stack up two 3*3 convolutions improved performance for 1.02% after finish the training process. This indicated that in such a deep neural network, large size convolution filters become unnecessary. Also by further increasing depth of the module could contribute to improving performance on large data set.

In the last experiment we test our proposal for contrastive learning and branch-increasing network. We choose the best result of our second experiment, CanalNet-B as our basic network architecture. Based on topology of CanalNet-B, we exchanged the max-pooling layers to our contrastive-pooling, and we denote this new model as CanalNet-C. In addition, we add several branches to CanalNet-C in order to observe the influence on such topology. We denote this network as CanalNet-D. The results are listed in table 4.3.

model	top-5 error
CanalNet-A	9.58%
CanalNet-C	2*
CanalNet-D	3*

Table 4.3 contrastive learning

*1,2,3 These experiments are not finish now, the results will be added in future version.

5. Conclusions

This research aims at increasing deep convolutional neural network performance by exploiting the network topology and improving the accuracy with implementation for contrastive learning. Our proposal focus on increasing depth of local network module, and replacing expensive large filter size convolution with stacked small size ones, adding branches in order to help network search for best topology automatically. In addition, we proposed a new pooling method for fast contrastive learning, by implementing the feature of receptive field to accelerate the learning process along with further reducing cost. We achieved better results compare to previous research without increasing computing cost by applying complicated training methodology, thus our model could use for practical applications. Finally our conclusion on exploiting network topology is compatible to all data augmentation methods that contributed to present state-of-the-art results. In future work we are expecting to experiment on such conditions and achieve better results.

6. Acknowledgement

I would like to appreciate my associate professor advisor, he helped me with the experiment preparation and provided countless valuable inspirations for my research. Also I want to thank to my associate advisor and all professors who gave their precious advices and opinions to this research. At last I would like to thank every committer and researcher who shared their effort for developing deep learning frameworks, without their selfless contributions this research could not be brought.

7. References

[1] K. Hornik, M. Stinchcombe and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks,* vol. 2(5), pp. 395-366, 1989.

[2] D. C. Plaut and G. E. Hinton, "Learning sets of filters using back-propagation," *Computer Speech & Language,* vol. 2(1), pp. 35-61, 1987.

[3] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision,* vol. 60(2), pp. 91-110, 2004.

[4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition*, 2005.

[5] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks,* vol. 3361(10), 1995.

[6] Y. LeCun, C. Cortes and C. J. C. Burges, "The MNIST database of handwritten digits," [Online]. Available: http://yann.lecun.com/exdb/mnist/.

[7] A. Krizhevsky, "The CIFAR-10 dataset," 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html.

[8] A. M. Sillito, "Inhibitory processes underlying the directional specificity of simple, complex and hypercomplex cells in the cat's visual cortex," *The Journal of physiology,* vol. 271(3), pp. 699-720, 1977.

[9] Y. Bengio, "Learning deep architectures for AI," *Foundations and trends® in Machine Learning,* vol. 2(1), pp. 1-127, 2009.

[10] Y. Bengio, P. Lamblin, D. Popovici and H. Larochelle, "Greedy layer-wise training of deep networks," *Advances in neural information processing systems,* vol. 19, p. 153, 2007.

[11] A. Berg, J. Deng and F.-F. Li, "Large Scale Visual Recognition Challenge 2010," Stanford Vision Lab, 2010. [Online]. Available: http://image-net.org/challenges/LSVRC/2010/index.

[12] N. V and H. G E, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning*, 2010.

[13] A. Krizhevsky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012.

[14] C. Szegedy, W. Liu, Y. Jia and et al, "Going deeper with convolutions," *arXiv preprint arXiv: 1409.4842,* 2014.

[15] P. Sermanet, D. Eigen, X. Zhang and et al, "Overfeat: Integrated recognition, localization and detection using convolutional networks[," *arXiv preprint arXiv:1312.6229,* 2013.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556,* 2014.

[17] K. He, X. Zhang, S. Ren and et al, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *arXiv preprint arXiv:1502.01852,* 2015.