The Pruning Methods for Convolutional Neural Networks based on SFEW database

Hengjia Zhang

Research School of Computer Science, Australian National University
u6658734@anu.edu.au

Abstract. The Neural Network can be used to solve many problems. However, as the problems to be handled become more and more complex, the level of network construction becomes deeper and deeper. Thus, the efficiency of training will be reduced. Pruning is a method that simplifies the network without decreasing network performance by removing redundant neurons. In this article, we will train two Convolutional Neural Networks to deal with the expression classification problem, and then use pruning techniques against the two networks and study their performance.

Keywords: Neural Network Pruning, Network Reduction, Convolutional Neural Networks.

1 Introduction

Neural networks are used in many different aspects, such as face recognition, recommendation systems, automatic navigation, etc. In recent years, with the increase of the available data set, the network we trained is getting deeper and deeper because complex neural networks may achieve better results than shallow neural networks. In a complex neural network, there may be many parameters that are redundant. Therefore, it is necessary to find a way to reduce redundant parameters.

Pruning neural networks is a method to reduce the size of neural networks. LeCun [1] states that some neurons which play a small role in neural networks can be reduced. The typical process of network pruning includes three steps: first train the model, then trim the trained model according to a certain standard, and finally fine-tuning the trimmed model to recover the lost performance.

In this article, we will use the Static Facial Expressions in the Wild (SFEW) database [2] to train a neural network for classification. Then, using pruning techniques to improve the performance of the network.

2 Method

2.1 The dataset

The Static Facial Expressions in the Wild (SFEW) database contains 700 images and has been divided into seven categories: angry, disgust, fear, happy, neutral, sad and surprise. There are 25 disgust which are missing so we have 675 images only not 700 like in the paper. We will use these 675 images to train a convolutional neural network of seven classifications. The only pre-processing we do is subtracting 0.5 and dividing by 0.5 to make the range of input data from [0, 1] to [-1, 1].

2.2 Network Design

2.2.1 AlexNet

The AlexNet was designed by 2012 ImageNet contest winner Hinton and his student Alex Krizhevsky [3]. The original AlexNet contained eight layers; the first five were convolutional layers, some of them followed by max-pooling layers, and the last three were fully connected layers. It used the non-saturating ReLU activation function, which showed improved training performance over tanh and sigmoid. Because the input and output dimensions of this classification task are not the same as the original AlexNet, we have made some changes to it.

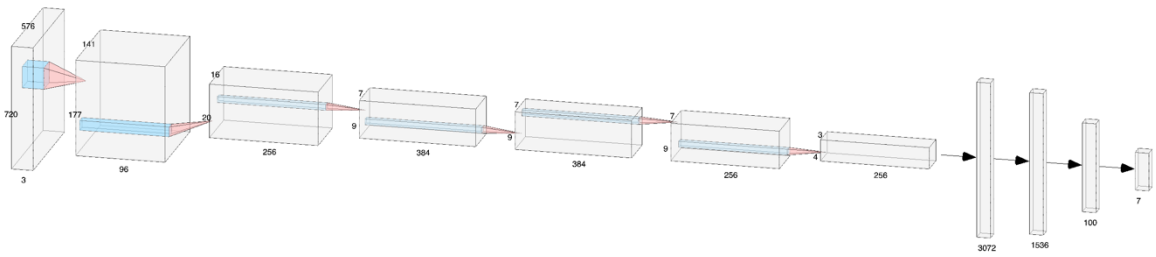

Fig. 1. An illustration of the architecture of our Modified AlexNet. The network's input is 576×720×3 = 1,244,160 dimensional.

As depicted in Figure 1, the net contains ten layers with weights; the first six are convolutional and the remaining four are fully connected. The output of the last fully connected layer is fed to a 7-way SoftMax which produces a distribution over the 7 class labels.

The first convolutional layer filters the 576×720×3 input image with 96 kernels of size 11×11×3 with a stride of 4 pixels. After that is ReLU function and a max-pooling layer.

The second convolutional layer filters the output of previous layer with 96 kernels of size 11 × 11 × 96, followed by a ReLU function and a max-pooling layer. The third convolutional layer has 256 kernels of size 5 × 5 × 96. The fourth convolutional layer has 384 kernels of size 3 × 3 × 256. The fifth convolutional layer has 384 kernels of size 3 × 3 × 384. The sixth fifth convolutional layer has 256 kernels of size 3 × 3 × 384, followed by a max-pooling layer. In the six previously mentioned convolutional layers, all the maximum pooling layers are the overlapping pooling with kernel size 3 and stride 2. The full connectivity layer has 3072, 1526, 100 and 7 neurons, respectively. Unlike the ReLU function used in the original model to connect the fully connection layer, we use the Sigmoid function. Since the output range of the Sigmoid function is [0-1], it is convenient for us to calculate the angle between the output vectors in the next pruning.

Our modified AlexNet is roughly the same as the original AlexNet in terms of convolutional layers. A convolutional layer was added because the images that needed to be processed were larger than the original AlexNet processed images. There are large differences on the fully connection layer, we use the Sigmoid activation function instead of the ReLU function, and the number of neurons on each layer is different from the original network.

2.2.2 VGG

The VGG neural network refers to a deep convolutional network for object recognition developed and trained by Oxford's renowned Visual Geometry Group (VGG), which achieved very good performance on the ImageNet dataset [4].

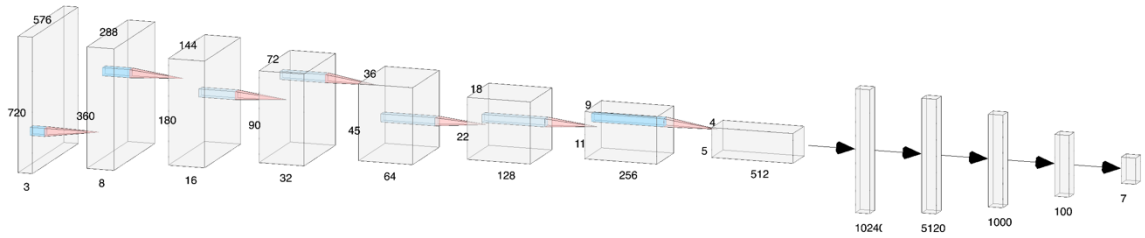

Fig. 2. An illustration of the architecture of our Modified VGG.

Our modified VGG network contains a stack of convolutional layers. The first convolutional layer filters the 576×720×3 input image with 8 kernels of size 3×3×3 with a stride of 1 pixels and padding 1 pixels. After that is ReLU activation function and max-pooling layer with kernel size 2 and stride 2. After that each convolutional layer is roughly similar, with the only difference being that the number of kernels doubles each time. A stack of convolutional layers is followed by five fully connected layers with 10240, 5120, 1000, 100 and 7 neurons, respectively. We use the Sigmoid function to connect the fully connection layer

We created the convolutional layer according to VGG's rules, but because the input picture is different, the data dimensions are different, so the number of neurons in the fully connected layer is much different from the original model.

2.2.3 Hyperparameter Selection

A number of experiments were performed using different learning rate and batch size. We divided the model randomly into a training set and a test set. The training set accounts for 80 per cent and the test set for 20 per cent. For each hyperparameter, we will run the model three times to get an average accuracy rate, instead of using k-fold cross-validation. The epochs in training is 10. different train and test accuracies were compared as shown in Table 1.

Table 1. Comparison of different hyperparameters on train and test accuracy

Model	Learning Rate	Batch Size	Average Train Acc.	Average Test Acc.
AlexNet	0.0005	64	26.85%	17.78%
AlexNet	0.001	64	28.33%	18.52%
AlexNet	0.0005	128	37.04%	22.96%
AlexNet	0.001	128	33.70%	22.22%
VGG	0.0005	64	27.78%	16.29%
VGG	0.001	64	35.00%	14.81%
VGG	0.0005	128	29.63%	18.52%
VGG	0.001	128	36.11%	21.48%

We can see that AlexNet with Learning Rate 0.0005 and Batch Size 128, VGG with Learning Rate 0.001 and Batch Size 128 have good performance. We will use those two networks for further experiments.

2.3 Network Reduction Technique

There are many methods for pruning neural networks. Sietsma and Dow [5] [6] developed the ad-hoc rules to detected neurons with little effect. Burkitt [7] uses the auxiliary layer to identify redundant units. Pelillo and Fanelli [8] focus on how to adjust the remaining network after removing redundant units. The method we use is based on calculating the angle between neurons' parameters [9]. If the aspect between the two neurons is very small, we can delete one of them because they work similarly.

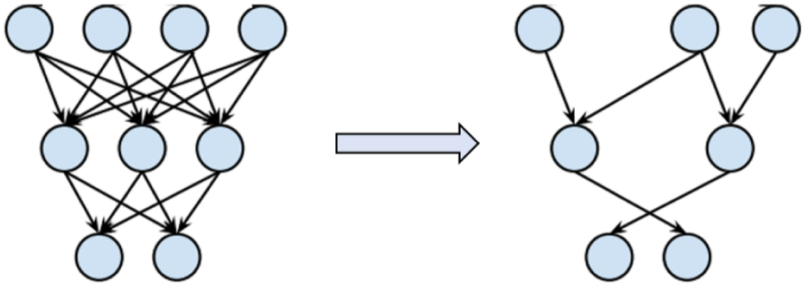

Fig. 3. The visualization of pruning feed-forward network

To use this method, we first need to get the vector of each neuron. The dimension of the neuron vector of each layer is the same, which is convenient for our calculation. These vectors represent the equations for neurons to process data. Therefore, the similarity of these vectors represents the similarity of neurons. Because all activation functions are sigmoid, the output interval of this function is [0,1], corresponding to the angle between the vectors is 0-90 degrees. To make the angles be in range 0 to 180 degrees, we need normalize it by subtracting 0.5 to make get their values from -0.5 to 0.5. Two vectors with an angle of about 90 degrees represent that the functions of the two equations do not coincide. If the angle is less than 15 degrees, the functions of the two equations are similar, and one of them can be deleted. If the angle is greater than 165 degrees, it means that they are complementary, and both should be removed.

After pruning, we recalculated the accuracy of this network using the test set without re-training and compared it with the accuracy before pruning.

3 Results and Discussion

3.1 Comparison with the Original Dataset Paper

The previous paper [2] did not use convolutional neural networks to complete the classification. Instead, it pre-processes the image to get the first two principal components (each dimension 5) of the LPQ (Local Phase Quantization) and PHOG (Pyramid of Histogram Oriented Gradients). Therefore, each image is characterized by 10 features. Then use a non-linear SVM to perform Classification. The baseline classification accuracy calculated by averaging the accuracy for the training and test sets is 19.0%.

Our AlexNet and VGG's Precision, Recall and Accuracy in test set are as follows:

Table 2. Test results of our Modified AlexNet

Emotion	Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Precision	0.21	0.19	0.25	0.31	0.28	0.25	0.19
Recall	0.20	0.21	0.23	0.29	0.27	0.26	0.15
Accuracy	0.23						

Table 3. The result of our Modified VGG

Emotion	Angry	Disgust	Fear	Happy	Neutral	Sad	Surprise
Precision	0.23	0.24	0.19	0.27	0.19	0.22	0.18
Recall	0.24	0.22	0.21	0.24	0.14	0.25	0.17
Accuracy	0.21						

We can see that these two convolutional neural networks have a higher accuracy than the Support Vector Machine. However, CNN's compute volume is much larger than SVM's, and this small boost is insignificant compared to the amount of compute spent. On the one hand, training a CNN by using the whole picture as input does get more information because of the improved accuracy. On the other hand, LPQ and PHOG have captured the main message of the picture, as the accuracy improvement is insignificant.

As mentioned in the previous paper [2], this low accuracy is attributed to the complex nature of conditions in the database. I'm guessing there are several reasons why my CNN accuracy isn't high. The first is that the dataset is too small, with only a few hundred images, to support the training of a huge CNN. The second is that there is a lot of other information in the picture besides the face. If we pre-processed the images, took the faces out of them, then trained the CNN, we might get better results.

3.2 Result of Pruning Technique

We performed three pruning techniques on the fully connected layer with 100 neurons in AlexNet and VGG with threshold angles of 15, 10 and 5 degrees, respectively, and the results are presented in Table 4.

Table 4. Accuracy after Pruning and the number of neurons being removed

Model	Test Accuracy before Pruning	threshold angle	The number of neurons in this layer	The number of neurons being removed	Test Accuracy after Pruning
AlexNet	22.96%	15°	100	95	14.81%
AlexNet	22.96%	10°	100	82	18.51%
AlexNet	22.96%	5°	100	65	22.22%
VGG	21.48%	15°	100	93	12.59%
VGG	21.48%	10°	100	85	17.04%
VGG	21.48%	5°	100	73	20.74%

It can be found that most of the neurons in this full connectivity layer are functionally similar, and if the threshold angles are set to 15 degrees, more than 90 neurons out of these 100 are subtracted. However, the accuracy of this network is significantly reduced. Accordingly, we consider that neurons with angles close to 15 degrees should still be

preserved although they are similar in function. When we set the threshold angle to 5 degrees, more than half of the neurons were still cut, but the accuracy dropped by no more than 1%. We consider 5 degrees is the suitable threshold angle for pruning. It can greatly optimize the fully connected layers without hurt the performance of the network.

4 Conclusion and Future Work

By building two CNN and pruning the full connectivity layer of them, this article demonstrates that pruning techniques can greatly streamline networks with little to no damage to network performance. The CNN has a slightly higher accuracy rate than the SVM used in the previous paper [2], which demonstrates that when data is complex, building neural networks may yield better results than supporting vector machines. However, the accuracy rate is still very low and does not achieve the effect of human eye recognition.

In future work, we will do better pre-processing of the dataset, such as taking out the faces and regularizing them. We'll also be looking for a better CNN structure to improve accuracy. For pruning techniques, we have used only one in this article, and will use several different pruning techniques and compare their advantages and disadvantages.

Reference

[1] Y. LeCun, J. S. Denker, S. A. Solla, "Optimal brain damage", *Advances in Neural Information Processing Systems 2*, pp. 598-605, 1990.

[2] A. Dhall, R. Goecke, S. Lucey and T. Gedeon, "Static facial expression analysis in tough conditions: Data, evaluation protocol and benchmark", *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops),* Barcelona, 2011, pp. 2106-2112.

[3] A. Krizhevsky, I. Sutskever and G. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM,* vol. 60, *(6),* pp. 84-90, 2017. DOI: 10.1145/3065386.

[4] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *Proc. Int. Conf. Learn. Representations*, 2015.

[5] J. Sietsma, R. J. F. Dow, "Neural net pruning - Why and how", *Proc. Int. Conf. Neural Networks*, pp. 1:325-333, 1988.

[6] J. Sietsma, R. J. F. Dow, "Creating artificial neural networks that generalize", *Neural Networks*, vol. 4, pp. 67-79, 1991.

[7] A. N. Burkitt, "Optimization of the architecture of feed-forward neural networks with hidden layers by unit elimination", *Complex Syst.*, vol. 5, pp. 371-380, 1991.

[8] M. Pelillo, A. M. Fanelli, "A method of pruning layered feed-forward neural networks", *Proc. IWANN'93*, 1993-June.

[9] T. D. Gedeon, D. Harris, "Network Reduction Techniques", *Proc. International Conference on Neural Networks Methodologies and Applications*, vol. 2, pp. 25-34, 1991.