Pruning Technique based on Distinctiveness for Deep Neural Network

Yang Zheng

Research School of Computer Science, Australian National University
u6287751@anu.edu.au

Abstract. The article designs and implements a pruning technique beads on the distinctiveness theory of Dr. Gedeon. The pruning is used to remove the redundant units in the full connected layers in the deep neural network, such the convolutional neural network. According to the tests of the article, the results indicate that the pruning technique is effective to decrease the number of hidden units in the full connected layers. For a suitable threshold degree, the pruned model can obtain the same performance with the original model over the handwritten digits images data set. In the article, the model of convolutional neural network can acquire a 97.8% accuracy over the data set which is better than the result of one of the published papers.

Keywords: Convolutional Neural Network, Feedforward Network, Pruning, Classification, Distinctiveness

1 Introduction

In recent years, deep learning is a hot theme in artificial intelligence fields [1] and it has practical success in many applications. For example, deep learning can achieve the successful practical results on the handwritten digits recognition, and the well-renowned case is the convolutional neural network (CNN). However, these deep neural network approaches have such a possible limitation that the models usually possess massive weights and so need quite a few memory and memory resources [2]. The drawback makes the models hard to be deployed on embedded systems and mobile devices which have limited hardware resources [2]. Therefore, to overcome the limitation, pruning the network to compress the number of weights is an effective approach.

In the article, I explore a pruning technique based on distinctiveness which is proposed by Dr. Gedeon in 1990s. Through applying this technique to the convolutional neural network (CNN) and the feedforward neural network (FNN) and comparing their performance on the same data set, investigating and analyzing the effectiveness of the pruning technique. I choose the MNIST handwritten digits data set which is a classification problem to conduct the experiment.

1.2 Pruning and Distinctiveness

Pruning neural networks is not a new idea and it can go back to 1990 and before. Among the many parameters in a network, some may be redundant and do not contribute a lot to the output. The pruning technique is to remove neurons or features that are in some sense redundant to obtain a smaller and faster network. Meanwhile, a good pruning technique also needs to ensure that the performance of the new network is the same with the previous one.

Distinctiveness of the hidden units can measure whether the hidden units are redundant or not [4]. It can be determined from the hidden layer activation vector over the data set [4]. Every hidden unit has an activation value for one single sample, so for the data set, for one hidden unit we can get an activation value vector which can represent the functionality of the hidden unit over the data space[4]. By computing the angles of different vectors, we can recognize the similarity of different pairs of vectors. For example, we can consider that if the angle is less than 15 degree, the pair of units are similar and so one of them are redundant; if the angle is greater than 165 degree, the pare of units are complementary and both of them can be removed. In the article, the pruning technique is based on the concept of the distinctiveness to design and implement [4].

1.3 Data set

The task of classification applies the well-known MNIST handwritten digits database. The database is developed by LeCun, Cortes and Burges. There are 60000 training images and 10000 examples for testing in the MINST database of handwritten digits. Every image is grayscale and is normalized, and centered in a fixed-size frame where the digit lies at the center of the image which is 28 * 28 pixels with 4 bits per pixel [5]. The database is relatively simple for people to try artificial intelligence techniques and methods on real-world data while spending minimal efforts on preprocessing and formatting [5]. The figure 1 shows some training data.

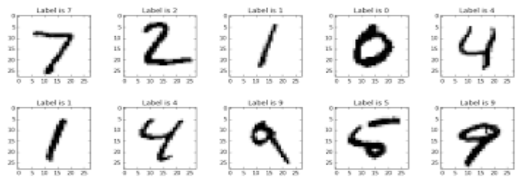
Fig. 1. Some examples from MNIST database

2 Method

In the article, firstly I implement a CNN and a simple feedforward network which includes one hidden layer on the handwritten digits data set. Secondly, I design and implement the pruning operation based on the distinctiveness and apply the pruning to two networks. Lastly, comparing the performance of pruned CNN and feedforward network on the data set with their corresponding original network respectively and then analyzing the effectiveness of the pruning. In addition, I conduct some experiments to investigate a suite threshold for angle degree to prune over the MNIST data set.

2.1 Network design and implementation

The CNN model includes two convolutional parts which contain one convolutional layer and one pooling layer and two full connected layers. For the first convolutional part, the convolutional layer is set to have 10 filters which have a kernel size of 5 and stride size of 1. The pooling layer is the max pooling with kernel size of 2 and the activation function is ReLU. Since the size of an original image is 1 * 28 * 28, so the size of output of first part is 10 * 12 * 12. For the second convolutional part, the convolutional layer is set to have 20 filters which have a kernel size of 5 and stride size of 1. The pooling layer and activation function are the same with the first part. So the size of output of the second part is 20 * 4 * 4. So for the first full connected layer, the size of input is 320, and the size of output is set to 64. For the second full connected layer, the size of input is 64, and the size of output is the same with the number of classes of the data space which is 10. All the activation function of the hidden layers is ReLU and the output layer uses the softmax function. The data passes the 6 layers in order to perform the classification task. The below table briefly shows the configuration of parameters for every layer in order of the CNN.

Table 1. Parameter configuration for every layer of CNN.

Layers	Parameter	Output size

Convolutional layer	In_channels = 1
	out_channels = 10
	kernel_size = 5
	stride = 1
		10 * 24 * 24

Pooling layer	kernel_size = 2	10 * 12 * 12

Convolutional layer	In_channels = 10
	out_channels = 20
	kernel_size = 5
	stride = 1
		20 * 8 * 8

Pooling layer	kernel_size = 2	20 * 4 * 4
Full connected layer	320, 64	64
Full connected layer	64, 10	10

The feedforward network model is a simple three layers structure based on the previous experiment. Its input layer is set to 784, and its hidden layer is set to 64 units, and its output layer is 10 units corresponding to ten categories. In the FNN, the activation function of hidden layer is sigmoid function, and the output layer applies the softmax function as its activation function to obtain the normalization values.

The two network models adopts the cross entropy as the cost function and the Adam algorithm as their optimization function. Their learning rates are set to 0.005.

2.2 Pruning design and implementation

According the aforementioned distinctiveness, to implement the pruning, firstly computing the output activation values of hidden units over the data space. That is, for a sample of the data space, one hidden unit can obtain an activation value, then for all samples, one hidden unit can acquire a vector. Computing the angle of every pair of vectors which is represented the hidden unites and obtaining a symmetric matrix. Every element of matrix is the angle of the every pair of vectors. We only need to use the upper triangle data of the matrix. Compare every angle to the threshold, such as 15 degree, if the angle is less than 15, then remove one of the pair of hidden units. Specially, adding the weights and bias of the one to the weights and bias of another one and setting the weights and bias of previous one to 0. If the angle is more than 165, then remove both of them. Specially, setting the weights and bias of them to 0. Summarize the process of pruning to the following steps:

i. Calculating the activation vector of all hidden units over the data space
ii. Calculating the vector angle for every two units and obtaining the matrix of angle
iii. Comparing the angles of matrix to the threshold, if it is less than the threshold, adding the weights and bias of one of them to the another and setting the weights and bias of the first one to 0; if the angle is more than (180-threshold), then setting the weights and bias of both of them to 0.

The pruning operation is applied to the full connected layers of the CNN and the FNN network, removing the useless units and optimizing the number of the hidden units. After the models are trained to be tended to convergent, applying the pruning operation to the models once and then obtaining the new pruned network without retraining.

2.3 Test and Investigation

Firstly, I need to validate the effectiveness of the pruning operation using some tests. Then I conduct an experiment to find a suitable degree for pruning the two models over the MNIST dataset. In addition, due to the limited performance of experimental computer, I select 20000 train samples from the MINST train set and 2000 test samples from the MINST test set to conduct the training and testing of the models.

Validation

Set the epoch of CNN to 10 and set the epoch of FNN to 100, and the training of CNN and FNN is based on the batch and the batch size is 100. Train the CNN model and FNN model over the train set data of the MNIST dataset and obtain the accuracies of the test set, marking as group 1. Apply the pruning to the trained models and acquire the new pruned modes. Use the new modes to obtain the accuracies of the test set, marking as group 2. Compare the difference of the corresponding accuracy of the two groups respectively. For eliminating the contingency of training, I repeat 10 times the same above operations – training, testing, and pruning. The threshold degree of CNN model is 15 and the threshold degree of FNN model is 25 for the tests.

Investigation

To obtain a suitable threshold to pruning the models, I conduct an experiment to investigate the pruning. In the experiment, I conduct the pruning operation with a range of degree from 15 to 60 with an interval 5 and for every degree, repeating the 5 same process with the validation part. Then obtain the average accuracy of the test set for every degree for the original model and pruned model and the average removal rate of pruning for the pruned model for every degree. The removal rate of pruning is the value that the number of removed units divides the number of initial units. Lastly compare every group data and find relatively appropriate degree for the models.

3 Results and Discussion

3.1 Validation result and discussion

The below figures shows the results of validation of effectiveness of pruning operation on the CNN for 10 tests. The figure 2 shows the comparison of accuracy of the original CNN model and pruned CNN model on the test set for repeated 10 times respectively. The threshold is 15 degree. It indicates that after the pruning, the new model have the same accuracy with the previous model on the test set. The averages of accuracy of the original and pruned model for the 10 tests are all about 97.5%. The figure 3 shows the removal rate of hidden units of the CNN model for every time respectively. The highest rate is about 39% and the lowest rate is about 26.5% and the average removal rate for all 10 times is about 32.5%.

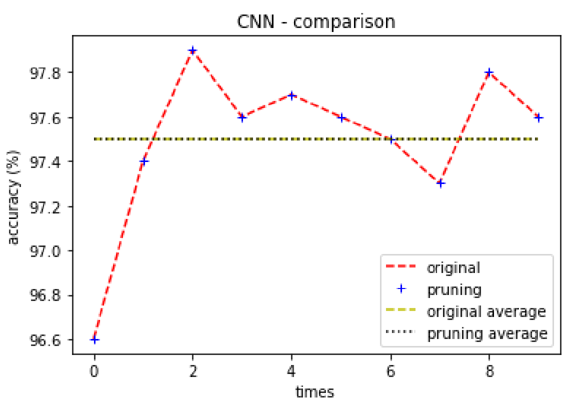

Fig. 2. Comparison of Accuracy of CNN

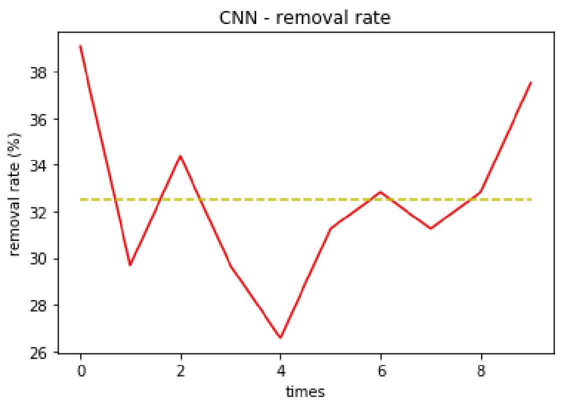

Fig. 3. Removal rate of CNN

The above validation results of CNN model indicate that the pruning operation is valid for the model over the MNIST dataset. For the CNN model, the pruning with 15 degree threshold can effectively decrease the number of hidden units in the fully connected layers, meanwhile keep the same performance. The drop of units is about 32% with original size of 64.

The below figures shows the results of validation of effectiveness of pruning operation on the FNN for 10 tests. The figure 4 shows the comparison of accuracy of the original FNN model and pruned FNN model on the test set for repeated 10 times respectively. The threshold is 25 degree. It indicates that after the pruning, the new model have almost the same accuracy with the previous model on the test set. The average difference of accuracies is about 0.5% with an about 9.5% removal rate of units, and the highest difference is about 2% with an about 12.5% removal rate of units.

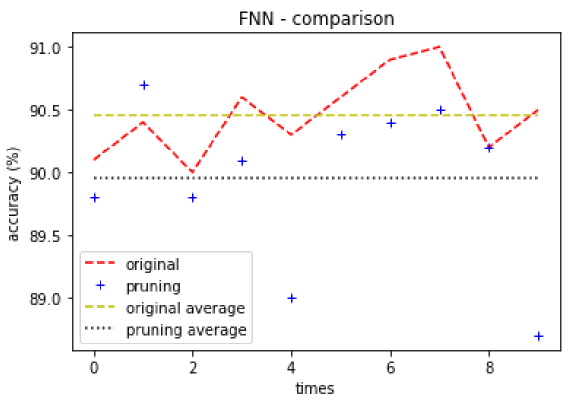

Fig. 4. Comparison of Accuracy of FNN

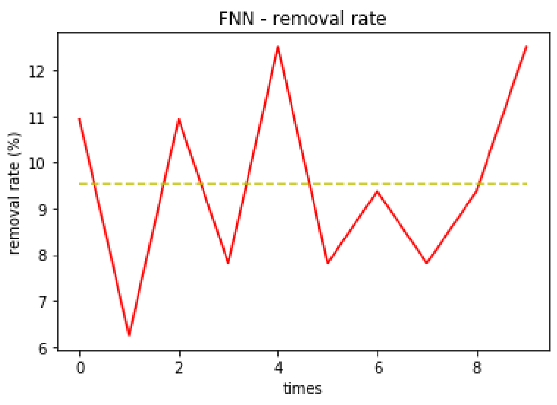

Fig. 5. Removal rate of FNN

The above validation results of FNN model indicate that the pruning operation is also valid for the model over the MNIST dataset. Through the pruned mode has an about 0.5% average accuracy difference with the original mode, the average removal rate of units is about 9.5% with the original size of 64 and in some case the pruned mode has a better performance than the original model. So the pruning operation seem to be able to improve the generalization of the model to some degree. In addition, one of the main reasons which result in the slight accuracy difference may be the threshold degree 25 which may be not the optimal one and the degree may lead to removing certain units which do not function identically with the others.

Summing up the above, the tests demonstrate that the pruning operation is valid to optimize the number of hidden units in the two kinds of deep neural network. The pruning technique based distinctiveness can indeed find out the redundant units in the hidden layers and keep almost the same performance with the original models. In addition, for certain cases, the pruning technique seems to be able to help generalization to improve the performance of the models on the test set since the pruning reduce the number of parameters.

According to the tests of FNN, the threshold degree of 25 seems not to be the optimal since it result in that the pruning removes some irredundant units to slightly decrease the performance of pruned model on the test set. So to investigate the suite degree for the models, an investigation which is introduced in the previous section is conducted.

3.2 Experiment result and discussion

In the figure 6, every red point shows the average accuracy of original CNN model for 5 times training, and the blue pint shows the corresponding value of pruned CNN model for the certain threshold degree. The figure 6 shows that when the threshold degree is no more than 30, the pruned CNN model has the same performance with the original CNN model over the MINIST test set. The figure 7 shows the removal rate of hidden units for the pruning for corresponding degree.

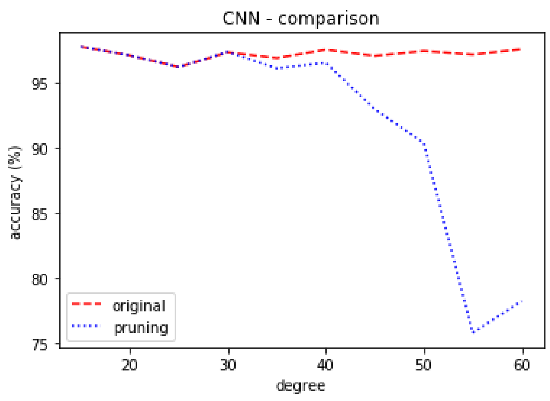

Fig. 6. Accuracy of CNN for degree

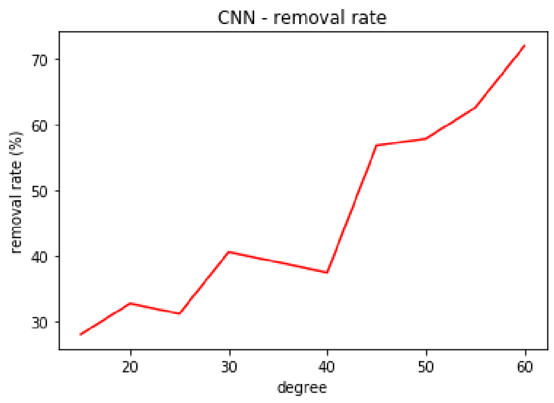

Fig. 7. Removal rate of CNN for degree

The above experiment results show that for the designed CNN model the 30 degree is a relatively suitable threshold for the pruning. It means that when the angle of two hidden units of full connected layers of the CNN model is less than 30 degree, then one of them is redundant. Meanwhile, when threshold degree is 30, the remote rate is about 40% which is relatively high. Actually, the exact value of accuracy of pruned model for 30 degree is 97.4%, and the accuracy of original model is 97.37%, and the pruned model has a slightly better accuracy than the value of original model, which means the pruning may improve the model generalization.

The below figures show the results for FNN. In the figure 8, every red point shows the average accuracy of original FNN model for 5 times training, and the blue pint shows the corresponding value of pruned FNN model for the certain threshold degree. The figure 8 shows that when the threshold degree is no more than 30, the pruned FNN model has almost the same performance with the original FNN model over the MINIST test set. The figure 9 shows the removal rate of hidden units for the pruning for corresponding degree.

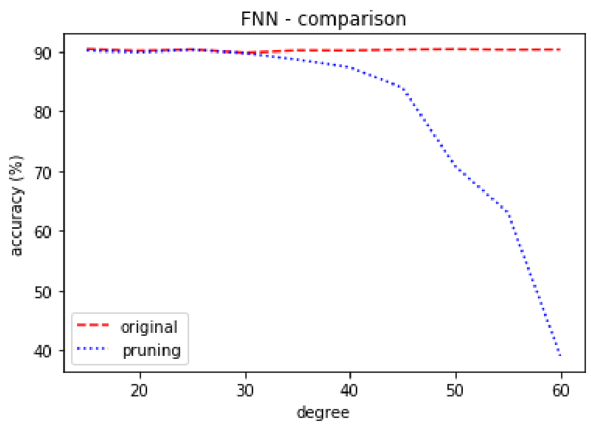

Fig. 8. Accuracy of FNN for degree

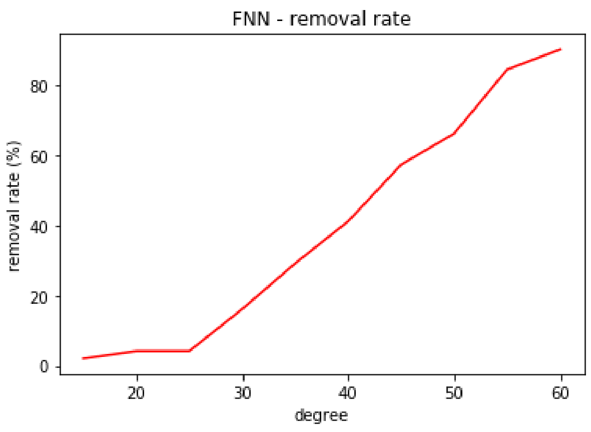

Fig. 9. Removal rate of FNN for degree

The above experiment results show a similar result with the CNN. That is, for the FNN model designed by the article the 30 degree is a relatively suitable threshold for the pruning.

3.3 Comparison with published paper

On the MNIST data set, many people implement many methods to classify the images such as using SVM, K-Means, and Neural Network and so on. In the article, I compare my method with other two studies which use other technologies. Detailed comparison is shown in the table 2.

Table 2. Comparison of results on MNIST dataset

Study	Method	Accuracy
Hinton et. al. [6]	Autoencoder+NN	99%
Singh et. al. [3]	Autoencoder+NN	97%
This article	CNN	97.8%
This article	Pruned CNN	97.8%

The Hinton's Autoencoder and classifier neural network have a total 9 layers including the input and output layers. In his paper, the Autoencoder which is used to extract the features of image to reduce the dimensions is complex. Singh's Autoencoder also has a total 7 layers including the input and output layers. Comparing with their results, the accuracy of CNN is less than the Hinton's method and is more than the Singh's method. CNN is indeed capable for classifying the image data and it is also easy to design and implement comparing with the method of Autoencoder.

4 Conclusion and Future work

This article designs and implements a pruning technique according to the Dr. Gedeon's distinctiveness theory. The pruning technique is used to remove the redundant units in the full connected layers for deep neural networks, such as the feedforward neural network and CNN. The article also detailedly tests and verifies the effectiveness of pruning technique based on the distinctiveness. It can find out the redundant units and remove them and keep the same performance with the original network model. After pruning the model, the new pruned mode also does not need to retrain. So using the pruning technique, we can actually obtain a smaller and faster network model.

Therefore, according to the tests and validations of this article, some conclusions can be summarized:

· The pruning technique based on the distinctiveness is effective.

· For the CNN model and FNN model designed by the article, the relatively suitable threshold degree for pruning is about 30 degree over the MNIST dataset.

· For certain cases, the pruned model plays a slightly better performance than the original model, which means the pruning can improve the generalization ability of the original model to some extent.

In the future, there are still some work to research. Firstly, in the article, the number of epoch, the size of hidden layer, the number of filter and other some parameters are all fixed. In further research, we can investigate which parameters have an important influence to the neural networks and the pruning techniques. Thus, we can research to find the most suitable parameters configuration for the models. Secondly, according to the tests of the article, I have observed that the pruned model can obtain a better performance than the original model, which means pruning technique can improve the generalization since it reduce the number of features and thus decrease the redundancy in the feature space. So in further research, we can investigate the relationship between generalization, pruning and feature redundancy.

References:

[1] Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, *35*(8), 1798-1828.

[2] Manessi, F., Rozza, A., Bianco, S., Napoletano, P., & Schettini, R. (2017). Automated Pruning for Deep Neural Network Compression. arXiv preprint arXiv:1712.01721.

[3] Singh, V., Kumar, K. R., & Eswaran, K. (2016). Learning Discriminative Features using Encoder-Decoder type Deep Neural Nets. *arXiv preprint arXiv:1607.01354*.

[4] Gedeon, TD, Harris, D. (1991) Network Reduction Techniques *Proc. Int. Conf. on Neural Networks Methodologies and Applications, AMSE, San Diego, vol. 2, pp. 25-34.*

[5] Du, T., & Liao, L. (2015, December). Deep Neural Networks with Parallel Autoencoders for Learning Pairwise Relations: Handwritten Digits Subtraction. In *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on* (pp. 582-587). IEEE.

[6] Salakhutdinov, R., & Hinton, G. (2007, March). Learning a nonlinear embedding by preserving class neighbourhood structure. In *Artificial Intelligence and Statistics* (pp. 412-419).