6.867 Project: Neural Network with Structure Growth During Training

Lei Zhou LEIZHOU@MIT.EDU
Massachusetts Institute of Technology, 77 Mass Ave, Cambridge, MA, 02139, USA

Abstract

This project studies the effect of structure growth for neural networks during training. The general idea is to start with a simple network and gradually crank up the network size. With the small and simple network, the big features of the data can be captured roughly, and adding some more neurons/layers after training the small network allows better capturing the details of the data. Several heuristic network structure update methods are tested, and the experimental results with several standard benchmark datasets are discussed. It is shown that compared with directly training a large (in depth and width) neural net, the growing network will allow better accuracy and convergence.

1. Introduction

Neural networks has become a very powerful method for machine learning and is receiving increasing success in recent years. However, there remains many concerns in the training for the neural network.

When training a neural network, one usually need to prespecify a network architecture and then solve the nonconvex optimization problem for the given network. Directly training of large network often suffer from difficulties in convergence, and finding the appropriate initialization is a challenging task. Meanwhile, the pre-specified network structure is often parametrized by several hyperparameters and then selected via cross-validation. The training for different networks cannot be reused, which slows down the selection process for the appropriate network structure.

The thoughts of allowing the neural network structural growth arises from the analogy to the natural learning process of human. When learning from examples, people usually learn better when the data are presented at early years, when his/her brain is growing. This is also helpful when the person is learning the same knowledge when at childhood and also after grown up, where the early impression is helpful for the person to grasp the main concept for the knowledge, instead of being distracted by details.

In this project, we studies the effect of the structure growth of neural network during training. The general idea is to start with a simple network, and gradually crank up the network size. With the small and simple network, the big features of the data can be captured roughly, and adding some more neurons/layers after training the small network enables the network to better capturing the details. By initialing a large size network the knowledge from a smaller network, the training of the large network may be faster. Also, this may help to avoid being trapped at a local minimum during training. Experiments with a simple feedforward neural network shows that this method can help the stability of the final network via providing good initialization with the small network, and can help get a better accuracy at the end.

2. Related Work

The network structure tuning is closely related to the pretraining and the transfer learning for neural networks. Pretraining (Simonyan & Zisserman, 2014) is a strategy proposed to facilitate the convergence of very deep neural networks, and transfer learning (Oquab et al., 2014) is introduced to overcome the over-fitting problem when training large neural networks on relatively small datasets. Network pruning has been studied in (Han et al., 2015), where not important neurons and connections are removed to improve the computational and storage efficiency of a network. Work `Net2Net` (Chen et al., 2015) studied how to transfer a network to a new one such that the network function is preserved. Recent work *AdaNet* (Cortes et al., 2016) formulated the network structure training as an optimization problem and provided convergence analysis, however the experiments on this algorithm has not been studied yet.

The thoughts of gradually training a network to reach better performance has been studied in (Bengio et al., 2009). In this work, data are shown to the network in certain order

6.867 Machine Learning Class Project, Fall 2016.

as a curriculum during training the network. The network structure does not update during the training.

3. Network Structure Growth during Training

We will start with a very simple neural network, for example single hidden layer and a small number of neurons. This small network usually have limited performance however it is very easy to train. There are two types of network growth mechanisms being considered: growing wider (`AddNode`) and growing deeper (`AddLayer`). Fig. 1 and Fig. 2 illustrate the two actions in a neural network.

The knowledge of the smaller network should be preserved after the network has grown larger. In this work, two heuristic `AddNode` implementations were used. In the first method, when growing the network wider, the weight matrices are appended with matrices with small value random coefficients. The matrix being append has its coefficients follows Gaussian distribution and has a small magnitude. In this way, the newly added neurons have very small coefficients and thus does not heavily distort the result of the network. In the second method, some of the existing neurons in the smaller network are separated into several neurons with equal weights. The newly added connection has the coefficients are also computed via slitting the original weight matrix with the same ratio. In this way, the new wider network is mathematically equivalent to the original network.

In the function `AddLayer`, when growing the network deeper, a diagonal weight matrix with small amplitude Gaussian noises is being added when a hidden layer is being added to the network. These operations allows the network after growth preserve the network values after the network size becomes larger.

3.1. Regularization

Choosing the appropriate regularization influence the performance of the network growing during training. $L1$ regularization penalizes non-zero parameters resulting in more parameters near zero. This limits the accuracy after growth, where a lot of parameters are just initialized near zero. After comparison, $L2$ regularization gives the best training result. In our implementation, $L2$ regularization was being selected.

3.2. Stop Criteria

There are two stop criteria need to be designed: the stop criteria for the training for the network of each size, and the stop criteria for structure growth.

During the training process, each step the network need

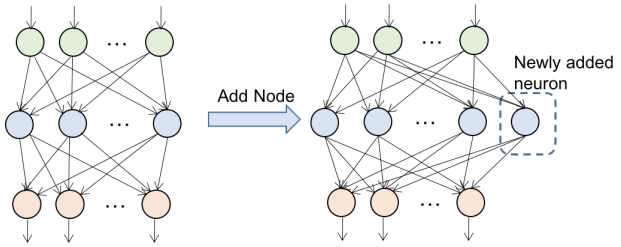

Figure 1. Network before and after adding neuron.

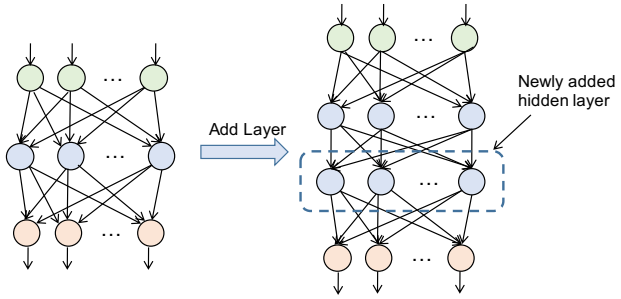

Figure 2. Network before and after adding layer.

to be trained to a certain convergence. In each iteration, the loss function on a validation data set is being evaluated. The iteration is stopped when the increment of the loss function value is smaller than a certain threshold.

The network structure growth should stop at appropriate time to avoid over-fitting. In this work, the increment of network accuracy on the validation data set is selected as the criteria of the structure update. When the accuracy increment of the network is smaller than a certain threshold, the network structure growth should be stopped, and the network with a smaller size is being reported as the final structure.

3.3. Network Growth Sequence

The network growth sequence, which specifies how many nodes and layers to grow at each step, has a strong effect to the network effect. However, in this project, we haven't yet developed a systematic method to grow the network to achieve the best performance. This is included in the suggestion of future work.

4. Experiments

The neural network with structural growth was implemented in Matlab on the basis of problem set 3 code. Softmax output function was being used, and the activation function for the hidden units was selected as ReLU. $L2$ reg-

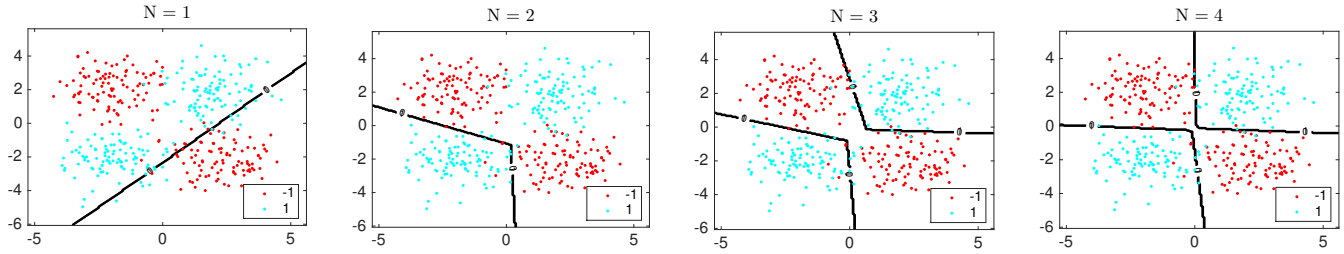

Figure 3. Neural net with structure growth test on binary data classification.

Table 1. Comparison of test accuracy between network with growth and regular neural network on two-class classification benchmark.

DATA SET	REGULAR NETWORK	GROWING NET
DATASET1	100%	100%
DATASET2	80.2%	94.7%
DATASET3	95.9%	98.8%
DATASET4	88.9%	96.7%

ularization is used, and each network was trained until the increment loss evaluated on a validation dataset is smaller than a certain threshold. The functions for network structure growth, `AddNode` and `AddLayer`, are implemented, and the neural net with structural growth is tested on several different datasets.

4.1. Binary Data Classification

The neural network with structure growth was first tested with the toy example of binary data classification task, which is easy to visualize the effect of the structure growth. The data being used was the data in problem set 3. We started with a network of one single hidden layer and 5 neurons, and in the test gradually increased the network size follows the sequence of `AddNode(10), AddLayer(1), AddNode(50), AddLayer(100)`. Fig. 3 shows the classifier at different steps, and Table 1 shows the accuracy comparison between the network with growth and the a regular network network of the structure at final step, which is two hidden layers and 165 neurons in every layer, the network with structure growth has more repeatable and in general results in higher accuracy.

4.2. MNIST Test

The network with structure growth is tested with MNIST data set. In these experiments, 3×10^4 data were used for training, and 1×10^4 data were used for testing. Each class (digit) has equal number of data. The network is a fully connected neural network with two hidden layers. The network started with 5 neurons in the each hidden layer, and the neuron number in each hidden layer was grown during training. Table 2-4 reports the accuracy on training data and test data of several different growing sequences. It shows that with large node increase steps (results in Table 4) the network can generally achieve the best accuracy. The average error rate in 10 runs is 0.34%, with the best error rate being 0.08%. In comparison, the accuracy of a regular network of the same structure has 91.3% average test accuracy, which corresponds to an error rate of 8.7%.

Table 2. MNIST dataset test performance with growing sequence `AddNode(30), AddNode(30), AddNode(30)`.

NODE NUM	TRAINING ACCURACY	TEST ACCURACY
5	45.12%	45.01%
35	94.3%	94.07%
65	99.41%	98.85%

Table 3. MNIST dataset test performance with growing sequence `AddNode(10), AddNode(50), AddNode(100)`.

NODE NUM	TRAINING ACCURACY	TEST ACCURACY
5	36.72%	36.43%
15	94.15%	92.94%
65	97.80%	97.00%
165	99.83%	98.41%

The two implementations for `AddNode` were compared in the experiment with MNIST dataset. There is no obvious accuracy difference when using different implementation. However, when using the "splitting node" implementation, the training process usually takes less iteration numbers than using the "appending node" implementation.

4.3. Cifar-10 test

The simple feedforward neural network was also test with the Cifar-10 dataset. Four batch of data, with 1×10^4 figures in each batch, were used for training data. One data batch was used as validation data for loss func-

Table 4. MNIST dataset test performance with growing sequence `AddNode(95), AddNode(100), AddNode(100)`.

NODE NUM	TRAINING ACCURACY	TEST ACCURACY
5	34.70%	34.11%
100	99.11%	98.51%
200	100%	99.66%

tion evaluation, and one data batch was used for testing. With a neural network start with two hidden layer and 10 neurons in each layer, and gradually increase the network size by `AddNode(290), AddLayer(1), AddNode(300)`, the network reached an average accuracy of 54.09% on test data, and 82.90% on training data. In comparison, the regular neural network with the final network structure (three hidden layer, 600 neurons in each layer) is hard to get a converging result.

5. Conclusion and Future Work

In this project, the preliminary study for the effect of the structural growth of the neural network is conducted. Experiments on different data sets show that the network structure growth can significantly improve the performance of the neural network. Future work include generalizing the concept to other form of neural networks such as convolution neural network, and study a systematic method to design the network growth sequence.

References

Bengio, Yoshua, Louradour, Jérôme, Collobert, Ronan, and Weston, Jason. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pp. 41–48. ACM, 2009.

Chen, Tianqi, Goodfellow, Ian, and Shlens, Jonathon. Net2net: Accelerating learning via knowledge transfer. *arXiv preprint arXiv:1511.05641*, 2015.

Cortes, Corinna, Gonzalvo, Xavi, Kuznetsov, Vitaly, Mohri, Mehryar, and Yang, Scott. Adanet: Adaptive structural learning of artificial neural networks. *arXiv preprint arXiv:1607.01097*, 2016.

Han, Song, Pool, Jeff, Tran, John, and Dally, William. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, pp. 1135–1143, 2015.

Oquab, Maxime, Bottou, Leon, Laptev, Ivan, and Sivic, Josef. Learning and transferring mid-level image representations using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1717–1724, 2014.

Simonyan, Karen and Zisserman, Andrew. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.