Sentiment Analysis Based On ENAS

Haijie Li

January 4, 2022

Abstract

In this project, I completed the recurrence of the general Architecture of the paper "Efficient Neural Architecture Search via Parameter Sharing" based on the given code, and improved the network structure based on the recurrence result. At the same time, it has been applied to NLP sentiment analysis task and achieved good effect.

Related work

At last decade, the research of neural network has made breakthrough and brought great progress and innovation in many fields. However, because of its wide application range, we have to design specific models for different problems, and how to design a good model is also one of the decisive factors of whether the network is efficient. For example, CNN is used in the field of CV, and RNN is used in the field of NLP, which are all examples of successful models.

The paper "Automated Machine Learning To Generate Optimal Neural Architectures" introduces the concept of NAS, which enables us to no longer need to do the complex tasks partly, including designing networks and adjusting parameters. However, it also has drawbacks, such as limited search scope, inflexible network structure and huge cost of training. To solve these problems, this project completed the tasks of realizing topology structure learning and realizing Parameter Sharing required by ENAS, Based on the idea of "Efficient Neural Architecture Search via Parameter Sharing". According to the requirements, I improved the network structure, training process and other aspects , and applied it to the NLP field ,which received good results.

The analysis of article

"Efficient Neural Architecture Search Via Parameter Sharing" (hereinafter referred to as ENAS) was improved based on NAS, and in order to understand NAS, I also read the paper "Automated Machine Learning to Generate Optimal Neural Architectures" (hereinafter referred to as NAS). Firstly, the main points of these

two papers are summarized as followed:

NAS

According to NAS, An RNN is used as the controller to obtain a network structure (called Child network in the paper) in the search space. The process of NAS can be summarized as:

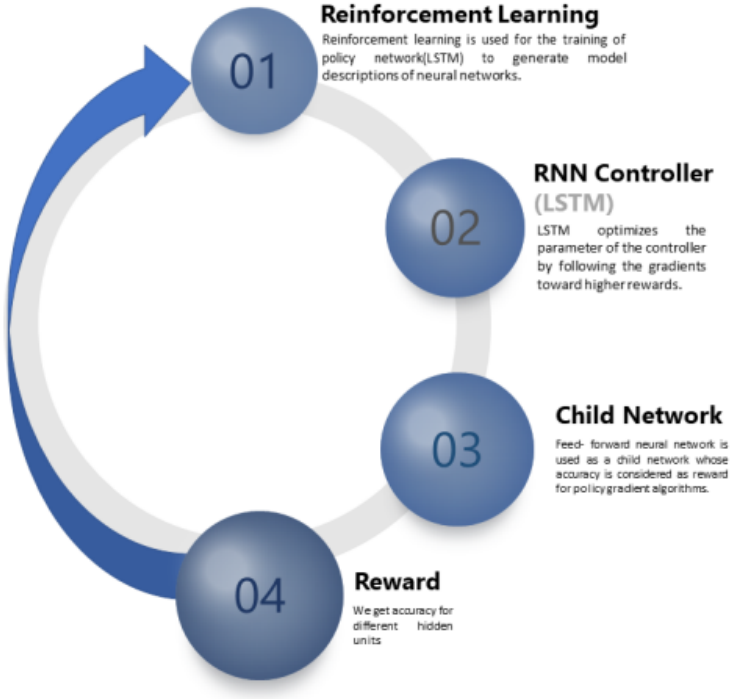
1) First, controller generates a child network.

2) Then, train the network structure on data set.

3) Gain accuracy rate the validation set.

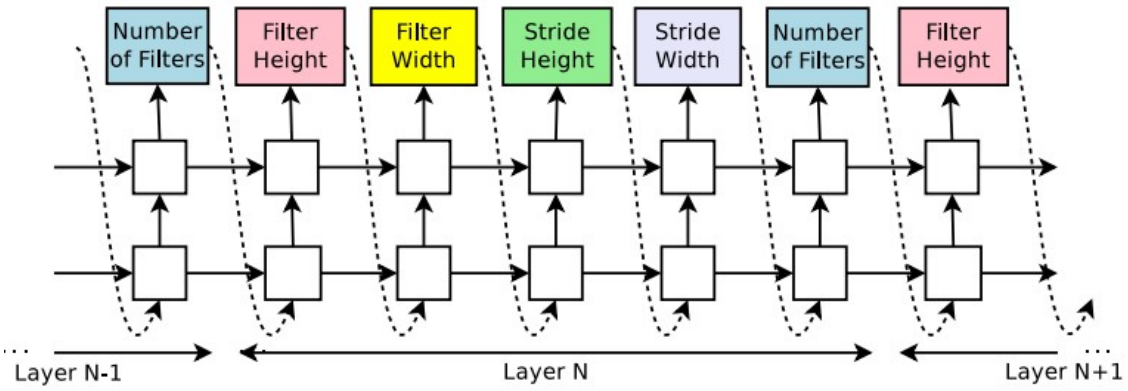
4) Return the accuracy rate to the controller as a reward. At the same time, the controller is optimized to obtain another network structure by means of reinforcement learning, and the process is repeated until the best result is obtained.

From the perspective of reinforcement learning, the paper regarded the RNN controller as Agent, whose optional state space was how to arrange the topological structure of network , and then used the strategy gradient algorithm to train the controller. In a more general sense, the RNN controller takes on the function of auto-callback.



Figur 1: NAS structure

Generate network:
Taking the structure of deep convolutional neural network as an example, every five steps output of RNN controller corresponds to a layer of convolutional neural network. When the depth of the generated neural network exceeds the preset value, the output is stopped.



Figur 2: Example training process in NAS

Training:
Next, the generated neural network structure is used to build the neural network, and the training is carried out on the training set, and the error of the neural network in the verification set is recorded, and the reward is returned to the controller, which is almost the same as the ordinary neural network training.
After receiving the error, the controller regards the result of each step in the RNN controller as an action in reinforcement learning, and the corresponding state is the network structure generated by the controller up to the t step. The reinforcement learning problem gives a sparse reward/cost at the end of a trajectory, that is, the loss on the validation set corresponding to the neural network result. This allows RNN controller weights to be updated using reinforcement learning methods. In this paper, the REINFORCE with Baseline algorithm is used for training, and after the training, a neural network structure is generated for training

Application:
The convolutional neural network with skip Connection, Batchnorm, pooling and other structures can be realized by using above methods, deep RNN network can be generated too.

Advantages:
The convolutional neural network with skip Connection, Batchnorm, pooling and other structures can be realized by using above methods, deep RNN network can be generated too.

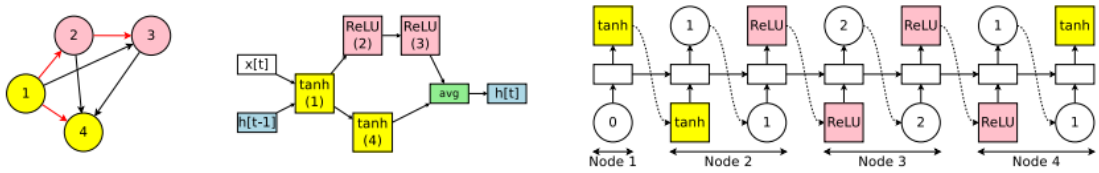
Disadvantages:
Every time a new child network structure is generated, we need to initialize the

weight of the child network and restart training. This makes training expensive and unfriendly to small data sets. Only a small number of hyper-prameters of the network can be adjusted.

ENAS

To overcome the disadvantages of NAS, ENAS made the following improvements in this paper:

1) A directed acyclic graph is proposed to represent the structure of the neural network so that the RNN controller can change the topology of the network.



Figur 3: ENAS structure

2) Child network weight learning and controller learning are synchronized. In this way, the parameters obtained by controller learning are more new rewards, which can better reflect the current training situation.

3) Nodes can share parameters, cutting the cost of training.

Provided code drawbacks and room for improvement

After comparing the paper with the given code, I consider the given code still has the following problems can be improved

1) The learning of topology learning is not implemented and the topology is fixed as a linear architecture.

2) The RNN cell structure is not implemented.

3) Create the same data set multiple times, which increasing the computational burden.

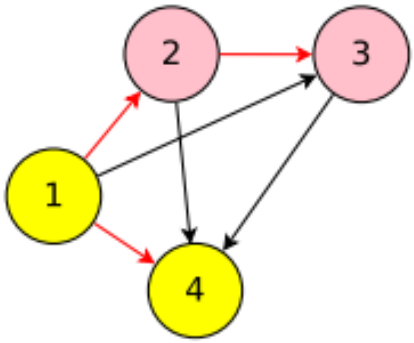
4) Parameter sharing is not implemented. Each time, the network is recreated for training and the previous parameters are discarded.

Details

After comparing the paper with the given code, I consider the given code still has the following problems can be improved

1) **Paper implementation**

   a) Building topology structure We express a directed acyclic graph as follows:



Figur 4: Example DAG in ENAS paper

node	1	2	3	4
parent	0	1	2	1

Tabel 1: Nodes and their parents

   Record the parent of each node, thus representing a topology. By changing the search space for RNN controller, it is enable to determine the parent of each node and the activation function for each node. Then in the process of defining the child network, the relationship between nodes is guided by reading the parent node of each node.

   b) Implements parameter sharing At the end of each training, we saved the parameters of different nodes and indicated the parent node of each node. If the node is activated again by the same the parent node, it can inheritance the previous parameter, avoiding repeated initialization.

2) **Improvement**

   After careful consideration of the network of the original paper, I found that when implementing the design of RNN cell, all unused outputs were averaged to synthesize their values. Though easy to do, I think there are still some inadequacies. If our RNN Cell has multiple linear layers of output , it is theoretically better to allow them to adjust their weights adaptively, rather than simply taking averages.

   Inspired by the Attention algorithm in the NLP field, I improved the original network, enabling it to adjust the weight of each output according to the structure

of the network, and finally obtain H [t] by calculating the weighted sum of unused outputs.

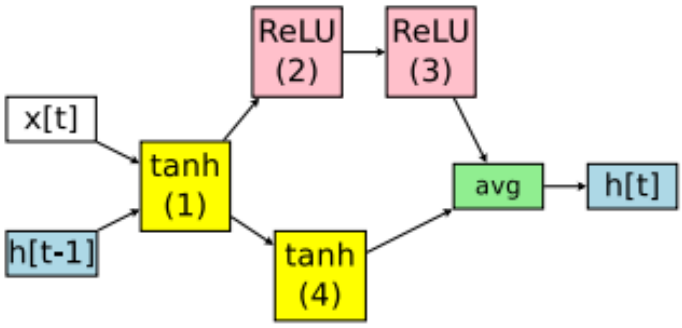
Taking the Fig1 of the original paper as an example, the improvement method is as follows:

a) Construct a new linear layer, taking the structure of the network as input and a scoring vector as output.

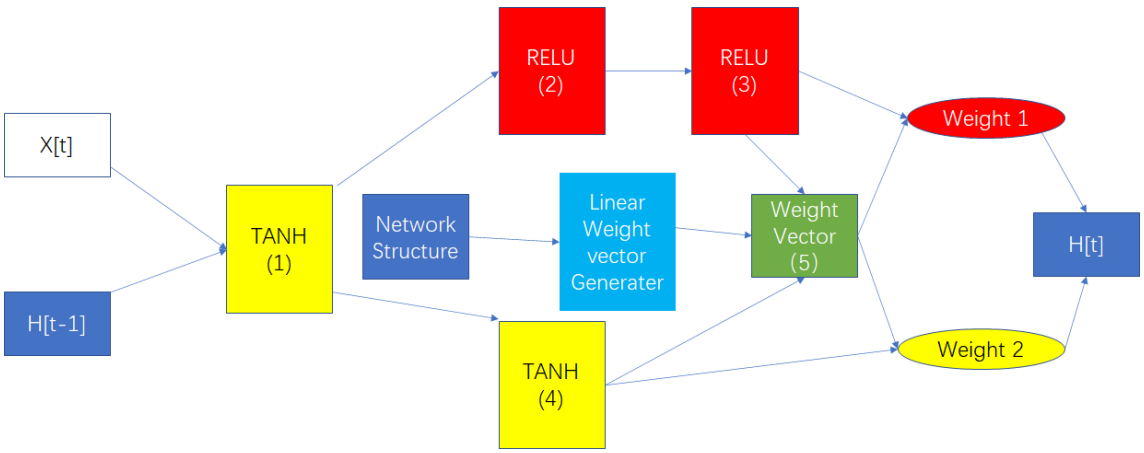
b) Multiply the scoring vector by each unused output to obtain their respective weights.

c) Calculating their weighted sum.

In this way, we can make the network adaptively adjust the weight based on the topology.



Figur 5: Example network in ENAS paper



Figur 6: Improved network

The corresponding formula is as follows:

$$node1 : k_1 = \tanh(x_t^{(h)} * W(x) + h_{t1} * W_1^{(h)})$$

$$node2 : k_2 = RELU(k_1 * W_{2,1}^{(h)})$$

$$node3 : k_3 = RELU(k_2 * W_{3,2}^{(h)})$$

$$node4 : k_4 = \tanh(k_1 * W_{4,1}^{(h)})$$

$$node5 : k_5 = LinearWeightVecotrGenerater * NetworkStrecture$$

$$weight1 = k_5 * k_3$$

$$weight2 = k_5 * k_4$$

$$weight1,weight2 = softmax(weight1,weight2)$$

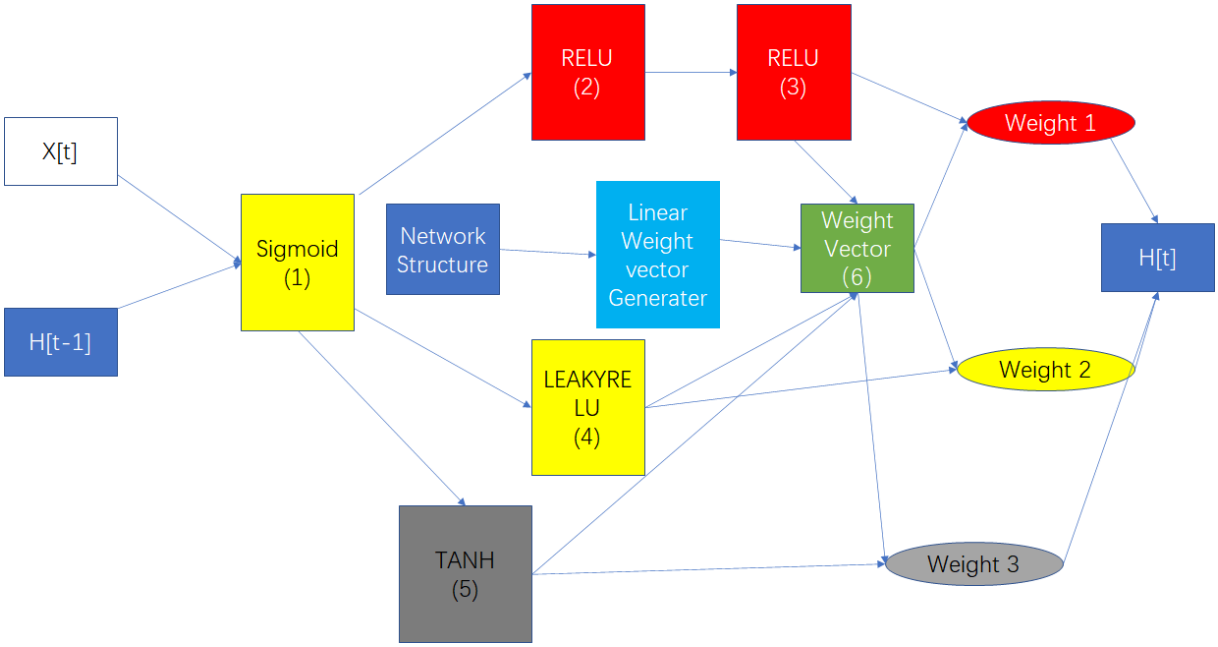
$$H[t] = weight1 * k_3 + weight2 * k_4$$

3) **Application on NLP**
We use IMDB data set to finish the sentiment analysis. IMDB data set, which contains 50,000 highly polarized reviews from the Internet Movie Database. The data set was divided into 25,000 comments for training and 25,000 comments for testing, with both the training set and test set containing 50% positive and 50% negative comments.

In order to finish the task, a word2vec model is constructed, using a vector to represent a word respectively. Each step RNN cell will take a vector of word as input, sending the hidden state value h as the input of next step. When reaching the end of sentence, we use a linear classifier to decide what sentiment that this sentence expresses.

Experimental results

The improved model can achieve a reasonable classification accuracy in NLP emotion classification task. The best model can achieve a accuracy rate of 79%, and its corresponding network structure is as follows :(when the length limit of the network is 5)



Figur 7: Final network

Analysis: Before completing this experiment, I first tried to use LSTM+Attention algorithm to classify the data, and the accuracy was close to 90%. Therefore, I expected the accuracy of this project to be above 90%, but the result was not satisfactory, and there was no obvious improvement after repeatedly adjusting hyperparameters. In my opinion, there are two main reasons for this failure:

a) LSTM model is implemented by directly using LSTM code of PyTorch which can well handle the situation of different length of sentences in the same batch. However, this situation is not taken into account in my model, resulting in a large number of zero are taken as input, which reduces the training effect.

b) When extracting information of a long sentence, Attention algorithm shows great improvement for RNN network. In contrast, my model is not good enough in this respect.

Analysis of advantages and disadvantages

Advantages: Compared with ordinary ENAS model, this model can automatically adjust the weight of network output, and has better generalization effect.

Disadvantages: The topology of the network only supports each node to have a parent node, which makes the directed graph become a tree, which is still not flexible enough. Meanwhile, under the task of NLP, the accuracy is not good enough too.

I believe that compared with the traditional model, my model has a great advantage in the information extraction of a single word, because theoretically its search space can cover LSTM cell. If we can overcome the two problems mentioned above, I think it can theoretically reach or even exceed the classical LSTM model. It can

be seen that NAS technology still has a lot of room for development in terms of generalization of specific domains.

Reference

1 Barret Zoph, Quoc V. Le “Neural Architecture Search With Reinforcemwnt Learning” 2017

2 Hieu Pham Melody Y. Guan “Efficient Neural Architecture Search via Parameter Sharing” 2018

3 Raul Perez i Gonzalo “Automated Machine Learning To Generate Optimal Neural Architectures” 2019