EECS 458 Introduction to Bioinformatics

Zhoumengdi Wang

zxw534@cae.edu

Using Fully Connected Neural Network to find disease gene of PTSD

**Introduction of Fully Connected Neural Network**

Fully Connected Neural Network is a basic network of machine learning. In my project, I use back-propagation algorithm based on Fully Connected Neural Network. In Fully Connected Neural Network, there are three important parts: input layer, hidden layer and output layer. Input layer and output layer only are both one layer, the number of hidden layer can be arbitrary. Also, the nodes of input layer and output layer are related to the inputs and outputs, and nodes of hidden layer can be arbitrary. In my project, I will change the number of hidden layer and nodes, in order to show different neural networks.

Here are the details of Fully Connected Neural Network:

Firstly, put a group of inputs (X1,X2,......Xn) into the input layer, and these inputs become into a group of data(S1,S2,......Sn) as the inputs of hidden layer through the weights between input layer and hidden layer. And these data become  through the active function of the nodes of hidden layer, means the output of No.j node of hidden layer. These outputs will become the inputs of the output layer through the weights between hidden layer and output layer. The process of output layer is similar to hidden layer, and there are outputs from the output layer. j means the output of No.j node of output layer.

As the active function, sigmoid function is the most commonly used function.

As mentioned before, there are outputs come from the output layer, how can we find out the accuracy of the neura network? There must be something to influence the outputs. Obviously, the inputs ,and functions are know, only the weights are variable. So how can these weights influence outputs? Now, we assume that the weights between No.i node of input layer and No,j node of hidden layer has changed a little , will influence Sj, and Sj will also cause a change , then will cause . will influence all nodes of output layer, and finally output layer will generate a err called . So adjusting weights will cause the change of outputs. Here is a problem, how to adjust weights to let outputs change into a right direction? For training data, the right results are already known, there is an error between results of neural network and right results, making the error less means the outputs close to the right results. In this situation, we can say that the neural network classify the data correctly.

In order to make the error less and less, we define an error function:



How to make L the least? Using random gradient down method. For each training sample, the weights change toward the negative direction of gradient. Now we have to get gradient of L to weights.

Here are some parameter:

: The weight from No.i node of input layer to No.j node of hidden layer.

: The weight from No.i node of hidden layer to No.j node of output layer.

: The input of No.j node of hidden layer.

: The input of No.j node of output layer.

We get:  Because:  So:  then we get: 

Now we only have to get . Because influence all nodes of output layer, so 

Because , so 

Integrate all formulas, we get that : , now we set 

So the δ of hidden layer is , the δ of output layer is ,

After calculating, .

This is back-propagation process, every node of output layer will get an err e, the neural network makes e as back-input of output layer. In this situation, output will be same as the input layer to propagate e back to the network. We will get δ of output firstly. According the weights between output layer and hidden layer, δ will be sent to hidden layer. So we will get the value of adjusting weights:

In my program, these two parameter are dW.

**Significant of the project**

According to my presentation and summary, the first problem I found is that the way that researchers used to find key words from studies is so inefficient that this way may cause too much time. So I consider that maybe we can use a smarter way to find the key words from the studies. Because we already know the symbol of PTSD studies, we can use the symbol as targets, and use Fully Connected Neural Network to find the studies we want. For example, we can input all the studies which may be related to PTSD studies, and make “PTSD” as the key word, if the studies do have “PTSD”, the output can be 1. And if the studies do not have “PTSD”, the output can be 0. After using the Fully Connected Neural Network, we will know which studies have “PTSD”, and these studies may be related to PTSD research. After finding the PTSD related studies, we also need to find the gene marker in studies. As we already know some gene marker related to PTSD, so we can use these markers as the symbol. Similarly, if the PTSD studies do have the gene markers, the output will be 1, otherwise, the output will be 0. Of course, the Fully Connected Neural Network we use need a lot of training data, we can use the studies that we have found to find as training data.

In my opinion, using Fully Connected Neural Network to find the key word and gene markers can save much more time. The work will be more efficient.

The second problem I found during studying gene of PTSD is that how can we find PTSD gene in people’s DNA sequence. During the studying of gene of PTSD, I found that PTSD maybe have some special gene. Assuming that gene of PTSD really exist, if we can find a way to find them from DNA sequence, we can let people who have suffered a traumatic event to test his DNA sequence. If the gene of PTSD is found in his DNA sequence, this person should better take the psychological treat in order to avoid having PTSD, and his family can also take care of him. I think the number of people who may get PTSD will decrease. My project will show that how to do it.

**Project**

The project shows the process of finding PTSD gene in a DNA sequence by matlab. Because a DNA sequence is too long to input in a personal computer, so I only used a little part of a DNA sequence. I used 10 DNA sequences which has 200 deoxynucleotides. And I find one of the PTSD gene is ANK3, one of the sequence of ANK3 is AGTCCTCAAG. So my project is finding this sequence in a DNA sequence. According to I have learned, the Fully Connected Neural Network will be more sensitive to number, and because the deoxynucleotides only contain 4 words: ACGT. So I change the DNA sequences which consist of 4 words into numbers. The way I use is changing by ASCII code.After changing, we get sequences consist of numbers.In order to make the problem easier, I make the numbers into 1,2,3,4. ‘A’ is 1, ‘C’ is 2, ‘G’ is 3 and ‘T’ is four. So the DNA sequences becomes the sequences consists of 1,2,3,4. I used 10 DNA sequences which has 200 deoxynucleotides. I make them into a 10x200 matrix. Every row is an input of the Fully Connected Neural Network. So the nodes of input layer is 200.In order to know the accuracy of the Fully Connected Neural Network, we have to know the targets which are the references of measuring the accuracy. In getting targets, I use a if statement: if (pp((j)) == 1) && (pp((j+1)) == 3) && (pp((j+2)) == 4) && (pp(j+3) == 2) && (pp(j+4) == 2) && (pp(j+5)==4) && (pp(j+6)==2) && (pp(j+7)==1) && (pp(j+8)==1) && (pp(j+9)==3). For the 10 DNA sequences, if a part of the sequence satisfy with the if statement, the value of target is 1, otherwise the value of target is 0. So there is a matrix of targets with the value of 1 or 0. I use the matrix to train the Fully Connected Neural Network.

In order to find a better Fully Connected Neural Network, I changed the number of layers and the nodes of every hidden layer. In order to make the Fully Connected Neural Network adjust all structure situations, I make some matrix. I will introduce these matrix:

The first matrix is layer\_matrix, I use this matrix to record some parameters of layer, including numbers of layer and nodes of every layer. If I want to show different structure of the network, I only need to change the value of this matrix. The form of layer\_matrix is the number of layer multiply 1. The number of layer means the number of hidden layer add 1 output layer. For example, in the program, if layer=5, that means there are 4 hidden layers in the network. Because we need many parameters of the output layer, so I have to add the output layer in the layer\_matrix. In the layer\_matrix, the value of every row is the nodes. I can adjust the nodes through changing the value of every row. Every time I add a hidden layer into the network, I have to input the value of nodes. And there is a variable value can not be changed,it is the value of the last row. The value of the last row of layer\_matrix is the nodes of output layer. Because the output in this network is 1, so the node of output layer is 1 and will not change with different structure of the network.

The second matrix is W\_matrix. The form of this matrix is the number of layer multiply 1. W\_matrix. is used to record the weights of every layer. In Fully Connected Neural Network, weights is the important part, the network gets close to the targets through adjusting the weights. The change of weights shows the influence of nodes. If the weights increase, that means this node has positive influence to the network. Otherwise, the node has negative influence to the network. The value weights of every layer are random firstly, then weights will change by the dW. Because the nodes of every layer are different, so I use cell matrix to store the weights.

The third matrix is bias\_matrix. The form of this matrix is the number of layer multiply 1. Bias is also an important factor in the network, it will adjust the output of each neuron. So the number of bias of each layer is equal to the neurons of each layer. I also let the value of bias randomly.

The fourth matrix is the most important matrix, the dW\_matrix. The form of this matrix is layer multiply 1.I get this matrix through using the formulas that has shown before. Firstly, I get all the outputs of every layer, and then use the outputs to calculate δ. I use δ, outputs, derivative and weights to calculate dW. For derivative, we have to talk about the active function. I use sigmoid function as the active function, so the derivative is output \* (1-output). I make the dW of all layers into dW\_matrix\_p. dW\_matrix\_p means dW of one inputs. Then I add all dW of all 10 sequences into dW\_matrix. Because I have to combine all sequences’ information, if I calculate every influence of every input, the workload will be large. So I add all dW of all sequences together, in order to change the weights of the network in one time.

After getting the dW\_matrix, I change the value by using this formula: W=W-eta\*dW. What is eta? eta is the learning factor, eta can show the ability of the network. I set the initial value of eta is 0.01, and I set a condition that can refresh eta. The condition is if the new rmserr is less than old rmserr, the eta=1.1\*eta. rmserr is a parameter that can show the accuracy of the network. It is like a err percent, if the rmserr=0.0021. that means the err percent is 0.21%. So the less rmserr, the more accurate. I record the rmserr and outputs in every iteration.

**Result analysis**

Here is some information of the results:

The iteration in all experiment is 5000 times. And I change the number of deoxynucleotides in one sequence, number of layers and number of nodes of one layer. I use rmserr as the evaluation standard of the network. The number of decimals is 10.

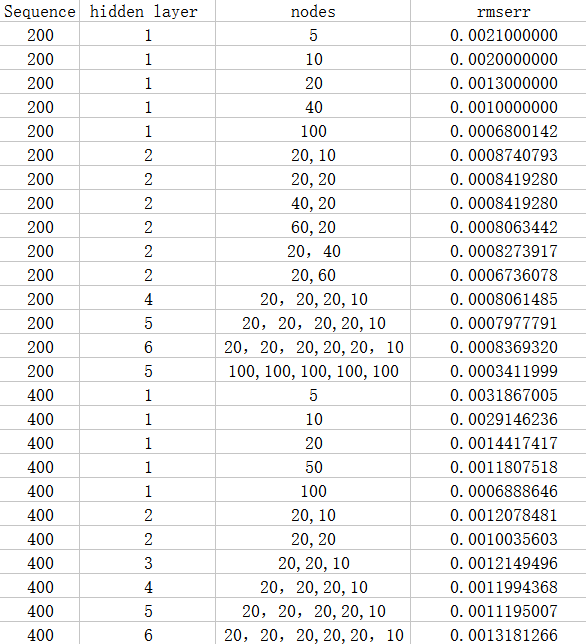
1. I use 10 sequences, every sequence has 200 deoxynucleotides. Firstly, I set 1 hidden layer, and the nodes of the hidden layer is 5. It is a simple Fully Connected Neural Network. The rmserr of this network is 0.0021000000. And I get the picture of the outputs of all iteration, the picture show that after 50000 iterations, the outputs become very close to the targets. In the beginning, the outputs is not very closed to the targets, with more iterations and bigger eta, the speed that the outputs close to targets increase. The speed shown in the pictures is the slope. I think the reason that cause the speed increase is that the eta becomes bigger and bigger. Because the new rmserr always less than old one, that means the network is good.
2. Then, I start to change the number of nodes of the hidden layer. I set the number into 10, 20,40 and 100. The rmserr are 0.0020000000, 0.0013000000, 0.0010000000 and 0.0006800142. When I change the nodes into 100, the rmserr decrease obviously. According to the rmserr, I find the more nodes of the layer, the more accurate outputs of the network. I think more nodes can treat inputs more detailed.
3. Next, I change the number of hidden layer. The first structure of network is 2 hidden layer, first hidden layer has 20 neurons and the second hidden layer has 20 neurons, too. The rmserr is 0.0008740793. This network is worse than the 1 hidden layer with 100 nodes. Then I increase the number of nodes of second layer, I change the number of nodes into 20. In this situation, first hidden layer and second hidden layer has the same number of nodes. The rmserr is 0.0008419280. Actually, the results does not change much. Then I increase the number of nodes of first hidden layer, 40 and 60. With the number of nodes increase, the less rmserr. I conclude that more nodes of first hidden layer cause more accurate results. The more complex first hidden layer has positive influence to the outputs.

(4) I increase the number of nodes of second hidden layer, although the the change of rmserr is little, the rmserr still decrease with more nodes. So the more nodes of second hidden layer also have the positive influence to the outputs. According to the observation, all the hidden layer should have many nodes.

1. I decide to change the structure of the network. I set 3,4,5 and 6 hidden layer. The rmserr are 0.0008010211, 0.0008061485,0.0007977791 and 0.0008369320. These rmserr show that more hidden layer may not cause better results. The less rmserr is generate when there is a 5 hidden layer.

(6) The network with 5 hidden layer and each layer has 100 nodes cause very good outputs. The rmserr is 0.0003411999. The picture of outputs show that, the first outputs is far from the targets, the reason may be the random weights. And the first half outputs are still little far from targets, and the second half of the outputs suddenly become very close to targets. Of curse, this structure of the network is more complex, and the process cause more time than other networks.

1. I try to make 2 sequences into one sequence. So each input is a DNA sequence with 400 deoxynucleotides. And I repeat all steps used in 200 deoxynucleotides. I find the same results. In 1 hidden layer network, more nodes, better outputs. Because the number of inputs is double now, so it cause much more time to getting the outputs. I can conclude than if the inputs is a complete DNA sequence, it must cause much more time.
2. Here is the results of the experiments of my project.



Conclusion: According to the experiments, I think the Fully Connected Neural Network with 5 hidden layers and every hidden layer has 100 nodes is an appropriate architecture.

**Shorts of the project**

1. The training data is too few: I only use 10x200 deoxynucleotides as the training data. These deoxynucleotides are far from the real DNA sequence.
2. The times of experiment is too few. I just change few parameters of the network, there can be more and more changes of parameters. Such as set the number of nodes more detailed, not just 20,40 and 60. And I find that the network with every layer has 100 nodes has very less rmserr, but maybe more nodes cause better outputs.
3. The PTSD gene may be the special one. I only use one gene of PTSD. I can not sure that this gene is not unique. Maybe other genes of PTSD can not use Fully Connected Neural Network to find.
4. The weights are random. The initial value of all weights in all experiment are random. Random value cause the results uncertain.

According to these shorts, I make a conclusion: Fully Connected Neural Network maybe a good way to find disease gene from a DNA sequence. Using this network, we can let computer find the relationship between inputs and outputs. And the workload will decrease much. But whether Fully Connected Neural Network is really appropriate to the problem still need be confirmed. There are a lot of experiments need to be done, including using the real hole DNA sequence, and trying more and more possible structures of the network. Value of the weights also need to be verified. And maybe other kinds of neural network may have better performance in finding disease gene. Finally, all genes which may relate to PTSD need to be found. Not only for testing the accuracy of the network, but also for understanding PTSD more deeply. If using neural network is really a good way to finding disease gene from DNA sequence, it can be popularized into other diseases.

During the hole study of gene of PTSD, I think we should focus more about the gene of disease. If we pay more attention in gene, maybe some diseases can be found early, and some treatment can be intervened early. And I think gene detection is very useful to every, if we find that we may have some disease gene, we can take more care of ourselves.