Research on Medical Image Segmentation Based on Machine Learning

CHI Yan[1], WANG Dong-hong[2], SUN Hui-feng [3], HU Yun-qi [1], WANG Jia-ying [4]

1 College of medical information engineering
Heilongjiang University of Chinese Medicine
Harbin 150040,China
2 Information Office
Admissions Committee Office of Heilongjiang Province
Harbin 150090,China
3 College of pharmacy
Heilongjiang University of Chinese Medicine
Harbin 150040,China
4 Research Center of basic education in Heilongjiang
Harbin Normal University
Harbin 150025,China

***Abstract* — Awaiting final text.**

Keywords - Machine learning; auto encoder; convolution pooling

I. INTRODUCTION

Machine learning is a branch of artificial intelligence, and in many cases, almost become synonymous with artificial intelligence. In simple terms, machine learning is through the algorithm, the machine learning rules from a large number of historical data, so as to make predictions about the future of the new sample to do the smart identification. Since the late 1980's, the development of machine learning has experienced two waves: shallow study (Shallow Learning) and deep learning (Deep Learning) [1-3].

Shallow model has an important characteristic, is the assumption that the characteristics of experience to rely on manual sample extraction, and that model is mainly responsible for the classification or prediction. In the application of the model under the premise of no mistakes (such as Internet Co hired experts in machine learning), has become the bottleneck of the whole system performance. Therefore, [4, 5] usually a more human development team is the input features to better. To find a good feature, requires developers to solve the problem to have a deep understanding. To this extent, often require repeated exploration, even years of grinding sword. Therefore, the design of artificial sample characteristics, not a scalable way [6].

The essence of deep learning, is a lot of hidden layer by constructing a machine learning model and massive training data to learn more useful features, thus improving the accuracy of classification or prediction. [7] So "depth model" is a means, "feature learning" is the purpose. Difference to traditional shallow learning and deep learning different is: 1. Emphasis on the structure model of depth, usually have 5 layer, 6 layer, even ten layers of hidden layer nodes. Clearly highlight the importance of learning characteristics, that is, with layer by layer feature transform, the samples in the original space feature representation to a new feature space, make the classification or prediction more easily [8, 9].

Compared with the structural features of artificial rules, the use of large data to study the characteristics of the data is more able to portray the inherent information. So, in the next few years, we will see more and more examples: the depth model is applied to large data, rather than the shallow layer of the linear model.

II. DEPTH STUDY INTRODUCTION

A. sparse autoencoder

A class of sparse auto-encoder algorithm Deep learning field more famous, namely automatic encoding sparse mode. Sparse auto-encoder is a kind of automatic extraction of samples (such as image feature method). The input layer activation (such as images) with the hidden layer of the hidden layer representation, information reduction in the output layer. This information is hidden on the input layer of a compressed representation, and the information entropy decreases. And it is suitable for the characterization of these classifiers. We know that deep learning is also called

unsupervised learning, so the sparse auto-encoder should also be unsupervised. If it is a supervised learning, in the neural network, we only need to determine the structure of neural network can be calculated loss function expressions (of course, the expressions for the need of network parameters "punishment", so that each parameter is not too large), also can be used to compute the loss function partial derivative expressions, then the optimization algorithm to find out the optimal network parameters. Should be clear that the expression of the loss function, need to use a label sample. So here is sparse auto-encoder why can the unsupervised learning? Is it that the expression of the loss function does not need to be marked in the sample value (that is, usually said y value)? In fact, sparse coding in the label value is also needed, but its output theory is the value of its own input characteristic value x, in fact, the value of y=x. The advantage of this is that the hidden layer of the network can be a good substitute for the characteristics of the input, because it can be more accurate to restore the input characteristic values. A network structure diagram of auto-encoder Sparse is shown in Figure 1.

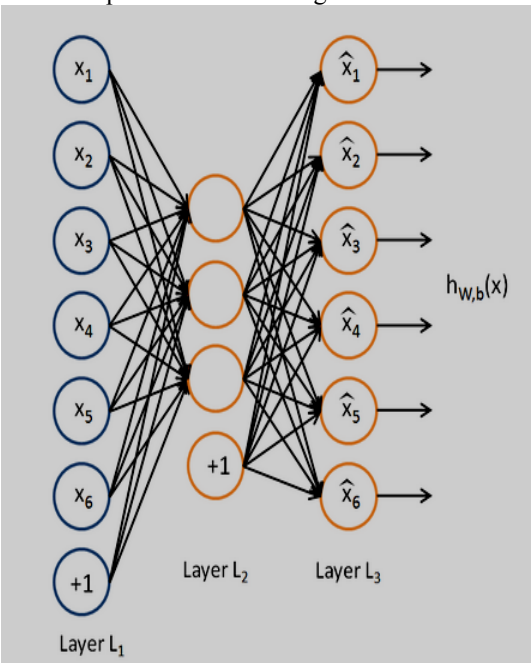

Figure 1. Network structure diagram.

B. loss function

The loss function expression for the network without sparse constraint is as follows:

$$J(W,b) = \left[\frac{1}{m}\sum_{i=1}^{m} J(W,b;x^{(i)},y^{(i)})\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2 = \left[\frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\left\|h_{W,b}(x^{(i)}) - y^{(i)}\right\|^2\right)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2 \quad (1)$$

Sparse encoding is output to the network hidden layer is the hidden layer node output constraints, i.e. the average value should be 0, so most of the hidden layer nodes are in an inactive state. Therefore, sparse auto-encoder loss function expression for the time:

$$J_{\text{sparse}}(W,b) = J(W,b) + \beta\sum_{j=1}^{s_2}\text{KL}(\rho||\hat{\rho}_j), \quad (2)$$

The following is the KL distance, whose expression is as follows:

$$\text{KL}(\rho||\hat{\rho}_j) = \rho\log\frac{\rho}{\hat{\rho}_j} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j} \quad (3)$$

The method of calculating the average value of the hidden layer nodes is as follows:

$$\hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m}\left[a_j^{(2)}(x^{(i)})\right] \quad (4)$$

The parameters are generally small, such as 0.05, that is, the probability of occurrence of small probability events. This requires that the output of each node in the hidden layer close to 0.05 (in fact, close to 0, because the network activation function is sigmoid function), so as to achieve the purpose of sparse. KL distance is expressed here is the difference between the two vectors. From the expression of the constraint function, we can see that the greater the difference, the greater the punishment, so the final output of the hidden layer nodes will be close to 0.05.

Suppose we have a fixed set of samples, which contains a sample. We can use the batch gradient descent method to solve the neural network. Specifically, for a single sample, the cost function is:

$$J(W,b;x,y) = \frac{1}{2}\left\|h_{W,b}(x) - y\right\|^2. \quad (5)$$

This is a (1/2) variance cost function. Given a set of sample data sets, we can define the overall cost function:

$$J(W,b) = \left[\frac{1}{m}\sum_{i=1}^{m} J(W,b;x^{(i)},y^{(i)})\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2 = \left[\frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{2}\left\|h_{W,b}(x^{(i)}) - y^{(i)}\right\|^2\right)\right] + \frac{\lambda}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(W_{ji}^{(l)}\right)^2 \quad (6)$$

The first item in the above formula is a mean square. The second item is a regularization term (also known as the weight decay term), which aims to reduce the magnitude of the weight and prevent over fitting.

The relative importance of weight attenuation parameters for the two items in the control formula. Here to reiterate the meaning of these two complex functions: for a single sample calculation of the variance of the cost function; is the overall sample cost function, which contains the weight of the.

The cost functions are often used for classification and regression problems. In classification problems, we use or to represent two types of tags, this is because the sigmoid activation function domain for, if we use the hyperbolic tangent activation function, you should use - 1 and + 1 as a label. For regression problems, we first need to transform the output domain, in order to ensure its scope (similarly, if we use the hyperbolic tangent activation function, to make the output domain).

Our goal is to target the parameters and to find the minimum value of the function. In order to solve the neural network, we need every parameters and initialization for a very small, close to zero the random value (for example, using normal distributions to generate the random value. The setting, after the objective function using such as batch gradient descent optimization algorithm. Because it is a non convex function, the gradient descent method is very likely to converge to the local optimal solution; however, in the practical application, the gradient descent method usually can get satisfactory results. In the end, it should be emphasized again that the parameters are initialized randomly, not all of which are. If all the parameters with the same value as the initial value, then all the hidden layer units will eventually obtained with the input value related, the same function (that is to say, for all, will take the same value, then for any input will have:). The purpose of random initialization is to make symmetric failure.

C. Back propagation algorithm

Each iteration of the gradient descent method is updated according to the following formula:

$$W_{ij}^{(l)} = W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W,b)$$
$$b_i^{(l)} = b_i^{(l)} - \alpha \frac{\partial}{\partial b_i^{(l)}} J(W,b) \tag{7}$$

Where is the learning rate. The key step is the calculation of partial derivative. We are now talking about the reverse propagation algorithm, which is an effective method to calculate the partial derivative.

We first look at how to use the back propagation algorithm to calculate and, these two are the partial derivative of the cost function of a single sample. When we calculate the partial derivative, the partial derivative can derive the overall cost function:

$$\frac{\partial}{\partial W_{ij}^{(l)}} J(W,b) = \left[\frac{1}{m} \sum_{i=1}^{m} \frac{\partial}{\partial W_{ij}^{(l)}} J(W,b;x^{(i)},y^{(i)}) \right] + \lambda W_{ij}^{(l)}$$
$$\frac{\partial}{\partial b_i^{(l)}} J(W,b) = \frac{1}{m} \sum_{i=1}^{m} \frac{\partial}{\partial b_i^{(l)}} J(W,b;x^{(i)},y^{(i)}) \tag{8}$$

The above two line formula is slightly different, the first line of more than second lines out of one, because the weight is the role of the attenuation is not. Back propagation algorithm is as follows: given a sample, we first conduct a "forward" operation, to calculate all the activation values in the network, including the output value. After that, for each node of the layer, we calculate the "residual", which indicates that the node has a number of effects on the residual error of the final output value. For the final output node, we can directly calculate the difference between the activation value and the actual value generated by the network. We define the gap as the output layer. How do we deal with the hidden units? We compute the weighted average of the residuals from the node (the first layer node). The details of the reverse transmission algorithm will be given below:

The forward conduction formula is used to calculate the activation value of the output layer.

For each of the output units of the first layer (the output layer), we calculate the residuals according to the following formula:

$$\delta_i^{(n_l)} = \frac{\partial}{\partial z_i^{(n_l)}} \frac{1}{2} \|y - h_{W,b}(x)\|^2 = -(y_i - a_i^{(n_l)}) \cdot f'(z_i^{(n_l)}) \tag{9}$$

$$\begin{aligned} \delta_i^{(n_l)} &= \frac{\partial}{\partial z_i^{n_l}} J(W,b;x,y) = \frac{\partial}{\partial z_i^{n_l}} \frac{1}{2} \|y - h_{W,b}(x)\|^2 \\ &= \frac{\partial}{\partial z_i^{n_l}} \frac{1}{2} \sum_{j=1}^{S_{n_l}} (y_j - a_j^{(n_l)})^2 = \frac{\partial}{\partial z_i^{n_l}} \frac{1}{2} \sum_{j=1}^{S_{n_l}} (y_j - f(z_j^{(n_l)}))^2 \\ &= -(y_i - f(z_j^{(n_l)})) \cdot f'(z_i^{(n_l)}) = -(y_i - a_i^{(n_l)}) \cdot f'(z_i^{(n_l)}) \end{aligned} \tag{10}$$

On the various layers, the first layer of the nodes of the residual error is calculated as follows:

$$\delta_i^{(l)} = \left(\sum_{j=1}^{s_{l+1}} W_{ji}^{(l)} \delta_j^{(l+1)} \right) f'(z_i^{(l)}) \tag{11}$$

$$\begin{aligned} \delta_i^{(n_l-1)} &= \frac{\partial}{\partial z_i^{n_l-1}} J(W,b;x,y) = \frac{\partial}{\partial z_i^{n_l-1}} \frac{1}{2} \|y - h_{W,b}(x)\|^2 = \frac{\partial}{\partial z_i^{n_l-1}} \frac{1}{2} \sum_{j=1}^{S_{n_l}} (y_j - a_j^{(n_l)})^2 \\ &= \frac{1}{2} \sum_{j=1}^{S_{n_l}} \frac{\partial}{\partial z_i^{n_l-1}} (y_j - a_j^{(n_l)})^2 = \frac{1}{2} \sum_{j=1}^{S_{n_l}} \frac{\partial}{\partial z_i^{n_l-1}} (y_j - f(z_j^{(n_l)}))^2 \\ &= \sum_{j=1}^{S_{n_l}} -(y_j - f(z_j^{(n_l)})) \cdot \frac{\partial}{\partial z_i^{(n_l-1)}} f(z_j^{(n_l)}) = \sum_{j=1}^{S_{n_l}} -(y_j - f(z_j^{(n_l)})) \cdot f'(z_j^{(n_l)}) \cdot \frac{\partial z_j^{(n_l)}}{\partial z_i^{(n_l-1)}} \\ &= \sum_{j=1}^{S_{n_l}} \delta_j^{(n_l)} \cdot \frac{\partial z_j^{(n_l)}}{\partial z_i^{n_l-1}} = \sum_{j=1}^{S_{n_l}} \left(\delta_j^{(n_l)} \cdot \frac{\partial}{\partial z_i^{n_l-1}} \sum_{k=1}^{s_{n_l-1}} f(z_k^{n_l-1}) \cdot W_{jk}^{n_l-1} \right) \\ &= \sum_{j=1}^{S_{n_l}} \delta_j^{(n_l)} \cdot W_{ji}^{n_l-1} \cdot f'(z_i^{n_l-1}) = \left(\sum_{j=1}^{S_{n_l-1}} W_{ji}^{n_l-1} \delta_j^{(n_l)} \right) f'(z_i^{n_l-1}) \end{aligned} \tag{12}$$

Replace the relationship between the upper and the middle of the relationship with the relationship, you can get:

$$\delta_i^{(l)} = \left(\sum_{j=1}^{s_{l+1}} W_{ji}^{(l)} \delta_j^{(l+1)} \right) f'(z_i^{(l)}) \tag{13}$$

III. EXPERIMENT SIMULATION

A. *Extract characteristic value*

From many natural images in the given cut out the size of 8*8 small patches pictures of 10000, now need to use sparse auto-encoder method to train a hidden layer network to learn the characteristics of. The network consists of 3 layers, input layer has 64 nodes, 25 hidden layer nodes, the output layer is of course 64 nodes.

In fact, to achieve the main steps of the function still need to calculate the loss function of the network and its partial derivative. The following simple language about the steps, convenient sort algorithm.

1 to calculate each node of the network input value (i.e. program Z value) and output value (i.e., the program of the a value, a is the sigmoid function value of Z).

2 using Z and a values calculated error value of each node (i.e. program in Delta).

Each node 3 can use the above calculated a, Z, delta to express the system loss function and the loss function of the partial derivative, of course, these are some mathematical derivation.

Figure 2. Experiment random graph.

In fact, step 1 is to carry, is calculated according to the input layer, the hidden layer, the output layer in the direction of. Step 2 is the direction of the (this is also the source of the algorithm called BP algorithm), that is, the error value of each node is in accordance with the output layer - the implicit layer - the input layer direction.

Here's a look at the experimental process:

First run the main program train.m in the step 1, that is, a random sample of 10000 small patch, and show one of the 204 patch images, image display as shown in figure 2.

Then run train.m in the steps 2 and 3, the loss function and the gradient function of the calculation and verification. Gradient checking time may be too long, here I probably used the one half an hour or more, when using the gradient checking and found the error is only 6.5101e-11, far less than 1E-9, so that in front of the loss function and its partial derivative function program is right. Followed by the optimization algorithm can be used to find the parameters, the program to the optimization algorithm is L-BFGS. After a few minutes of optimization, the results of the.

The weights of the final W1 are shown below:

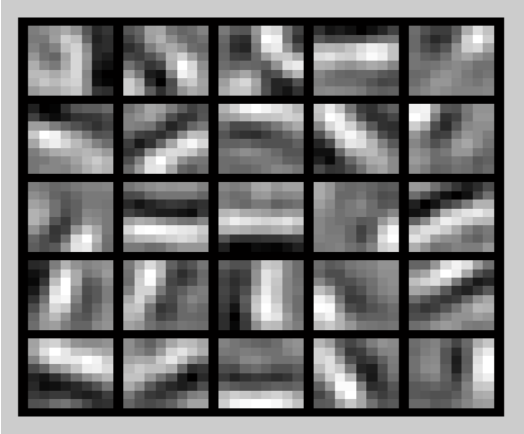

Figure 3. Last W1 weight.

B. *Self-taught learning*

Self-taught learning is a parameter for unsupervised learning to learn feature extraction, and then use supervised learning to train the classifier. Here are sparse auto-encoder and softmax regression, the experimental data is still handwritten digit database MNIST Dataset.

The digital 5~9 samples for unsupervised training, the method of sparse auto-encoder, can extract the data of weight, weight into the picture shows as follows:

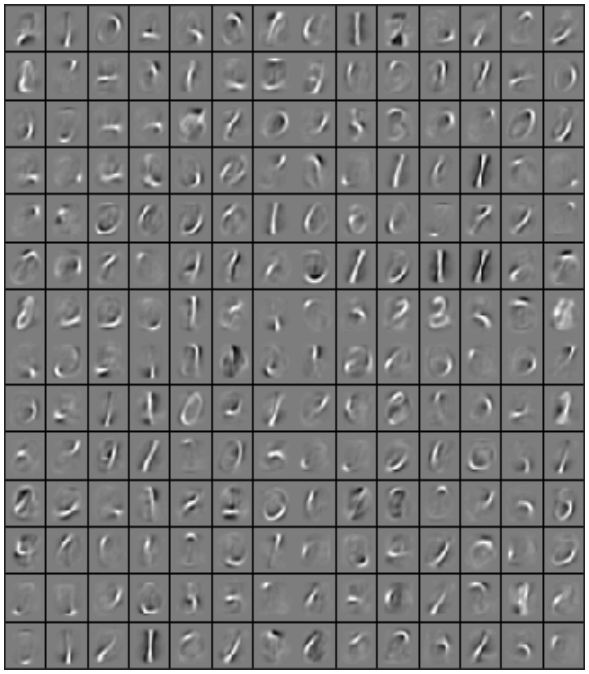

Figure 4. Weight transfer graph

But this experiment is mainly to carry out the classification of the 5 figures of 0~4, although the unsupervised training is used for digital 5~9 training samples, which still does not affect the results of the back. Just behind the classifier design is the use of regression softmax, so there is a supervised. The final result accuracy is 98%, and the direct use of the original pixel point of the design of the classifier is not only the effect of poor (96%), and the speed of training will be much slower.

C. Self-taught learning

In the global connection in the network, if our image greatly, for example for 96 * 96. The hidden layer is to learn 100 characters, at this time the input layer of all points and in the hidden layer node connection, you need to learn 10 ^ 6 parameters, so when using the BP algorithm speed was significantly slow a lot.

So back to the development of the local network, that is to say each hidden layer node is only connected to a part of the input row. The advantage is that the simulation of the visual cortex brain cortex in different positions only to the local area response. The local connection method using convolution network in neural networks. It in the neural network theory is for natural images, because they have the stability, other parts of the image in a part of the statistical characteristics and similar, so we learn that a part of the feature also applies to other parts.

The following specific look at an example on how to realize the convolution, if on a large image data set, the size of R, you first need to on this data set of random sampling size for the little picture of a * B, and then use these small picture patch learning (for example, sparse autoencoder). At this time of the hidden nodes into k. So the number of features in the final study is:

$$k \times (r - a + 1) \times (c - b + 1) \tag{14}$$

Although according to the convolution method can reduce many need to train the parameters of the network, for example, 96 * 96, and 100 hidden layer, the 8*8patch, 100 a hidden layer, is the need to train the parameters number reduces to 10 ^ 3, greatly reduced feature extraction process difficult. But at the same time is also a problem, namely its output vector dimension becomes great, originally fully connected network output only 100 dimensional, now on the output of the network 89*89*100=792100 and greatly change big, this on the back of the classifier design also difficult, so the pooling method appears.

Why does the pooling method work? First in front of the use convolution is makes use of the image of stationary characteristics, that is, the statistical characteristics of different parts of the image is the same, then when using the convolution calculation of pictures in a local area, a vector should on the image local feature, since the image is steady features, then to the feature vector obtained from statistical calculations, the images of all the local block should also can get similar results. The results obtained by the convolution statistical calculation process is called pooling, so pooling is also effective. Common pooling methods are pooling average and pooling max, etc.. And the characteristics of the study have rotation invariance.

From the above we can know that the simple, convolution is in order to solve the previous unsupervised feature extraction learning computational complexity of the problem, and pooling method is to behind the supervised feature classifier learning of, but also to reduce the required training system parameters (of course, this is in the common example of understanding. That is to say we use unsupervised method to extract target feature, and use a supervised method to train the classifier).

This experiment is to practice the use of convolution and pooling, a deeper understanding of how the big picture using convolution to get each feature of the output, then the pooling method to calculate these results, is shift invariant properties.

First, look at the whole process of training and testing the process: in the training phase, the whitening is a small patches. Because the input data is the big picture, so each convolution, there is a need for whitening and network weights calculation, so every learning to implicit node layer characteristics of each picture can get a smaller image features, then to the special sign images mean pooling. With these characteristics and the value of the label, you can use softmax to train a multi classifier.

During the testing phase is the convolution taken on the big picture, each convolution of image block also need were pretreated with the training of whitening parameter, respectively after convolution and pooling feature extraction,

and in front of the training process. Then the trained softmax classifier can be used to predict the.

Training feature extraction of the network parameters with more time, and the training for example, softmax classifier is used for a short time.

In MATLAB when there are n-dimensional array is generally from right to left peeling calculation, because the MATLAB output is in accordance with this method. Of course, if you want to understand the words, from left to right and from right to left are all right, as long as it is easy to understand.

Program of convolution test of reason is: first with the function of cnnConvolve calculated to sample convolution values and then randomly selected from a plurality of patch, with direct algebra method that network output value, if for all (for example here 1000) of the patch, the difference between the two are very small that convolution calculation is correct.

Procedures for the pooling test is the reason: the use of function cnnPool to calculate, and the parameters of the function for the polling dimension and the need for pooling data. So the program first casually to a set of data, and manual method to calculate the mean pooling of results, finally with the function of cnnPool also calculate a result, if the two results are the same, pooling function is correct.

The study of color features in the program is reflected in: each time only one channel of convolution for RGB, respectively, calculated 3 times, and then the three channels to get the convolution results matrix corresponding elements can be added. In this case, the back of the Pooling operation can only be carried out on an image.

Because only 4 categories of training softmax classifier, so its speed is very fast, less than 1 minutes.

The characteristic image is trained as:

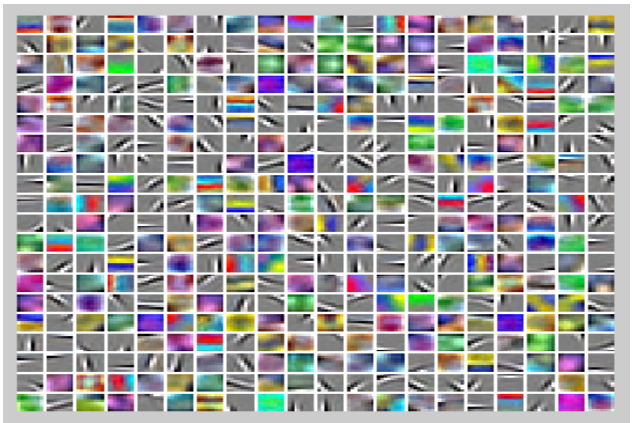

Figure 5. Training image

The final prediction accuracy is 80.406%.

IV. CONCLUSION

Promote deep learning of learning theory and the theory of computation at the same time, whether we can put forward a new hierarchical model, so that it not only has the traditional depth model has powerful representation ability and also has other benefits, such as easier to do theoretical analysis. In addition, for the specific application problems, how do we design a most suitable depth model to solve the problem? We have seen, both in the image depth model, or the language depth model, seems to be in the presence of depth and convolution and other common information processing structures. Even for voice acoustic models, researchers are also exploring the depth of the network.

ACKNOWLEDGMENT

Heilongjiang Province Education Science "twelfth five-year" planning subject (GJB1214017).

Education and teaching research foundation of Heilongjiang University Of Chinese Medicine(XJJ2013010).

REFERENCES

[1] BENGIO Y. “Learning deep architectures for A1”. Foundations and Trends in Machine Learning, vol. 2, no. 1, pp.1-124, 2009.

[2] D. Ciresan, U.Meier, J.Masci, and J. Schmidhuber. “A committee of neural networks for traffic sign classification”. In Neural Networks (IJCNN), The 2011 International Joint Conference on, pages 1918–1921. IEEE, vol. 10, no. 4, pp. 90-98, 2011.

[3] HINTON G, OSINDERO S, TEH Y. “A fast learning algorithm for deep belief nets”. Neural Computation, vol.18, no. 7, pp. 1527-1554, 2006.

[4] Y. Boykov, O. Veksler, and R. Zabih. “Fast approximate energy minimization via graph cuts. IEEE Trans”. Pattern Anal. Mach.Intell., vol. 23, no. 11, pp. 1222–1239, 2001.

[5] LECUN Y, BOTTOU L, BENGIO Y, et a1. “Gradient—based learning applied to document recognition”. Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998.

[6] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. “Scene parsing with multiscale feature learning, purity trees, and optimal covers”. In Proceedings of the International Conference on Machine Learning(ICML), vol. 2, no. 6, pp. 45-51, 2012.

[7] J. Carreira and C. Sminchisescu. “CPMC: Automatic Object Segmentation Using Constrained Parametric Min-Cuts”. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 7, no. 9, pp. 122-125, 2012.

[8] Y. Boykov and V. Kolmogorov. “An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision.IEEE Trans”. Pattern Anal. Mach. Intell., vol. 26, no. 9, pp.1124–1137, 2004.

[9] Y. Boykov and M. P. Jolly. “Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images”. In Proceedings of International Conference of Computer Vision (ICCV), vol 1, no. 11, pp. 105–112, 2001.