Assisted Diagnosis Research Based on Improved Deep Autoencoder

Zhang-Han KE[a], Yu-Kai DUAN[b], Yu TIAN[c], and Hao LIU[d]

Software College, Northeastern University, Shenyang, China
[a]*petrus@stumail.neu.edu.cn,* [b]*duanduan@stumail.neu.edu.cn,* [c]*20134838@stu.neu.edu.cn,* [d]*20134832@stu.neu.edu.cn*

Abstract: Deep Autoencoder has the powerful ability to learn features from large number of unlabeled samples and a small number of labeled samples. In this work, we have improved the network structure of the general deep autoencoder and applied it to the disease auxiliary diagnosis. We have achieved a network by entering the specific indicators and predicting whether suffering from liver disease, the network using real physical examination data for training and verification. Compared with the traditional semi-supervised machine learning algorithm, deep autoencoder will get higher accuracy.

1 Introduction

How to use medical data and extract the useful knowledge from the complex data effectively in the context of the intermittent observation of the condition, the complexity of various medical examinations, and the large amount of medical literature has become a more and more interesting problem for people. Disease auxiliary diagnosis is one of the directions, which is hoping to predict the possible illness of the subject by analyzing the information such as physical examination. Disease auxiliary diagnosis will use the most common customer's physical examination report as the original materials. From these materials we can collect the relevant data and mine the hidden information to assist in the diagnosis. It will not only help out-patient physicians in small and medium-sized hospitals to make more accurate judgments without other complex operations and testing, but also provide follow-up recommendations to help find the cause quickly in the scenario of failing to detect the disease.

With the development of machine learning, there are many studies and algorithms in the field of disease auxiliary diagnosis, but many algorithms cannot replace manual diagnosis because of the request of high accuracy of the prediction in the medical field. However, with the rapid develop in computer performance in recent years, making the deep learning rises again. At present, the deep learning in the image, voice and other aspects has been significantly beyond the previous methods, which makes the use of artificial intelligence in the field of high precision requirements possible. Therefore, the use of deep learning to achieve the diagnosis of the disease is of practical significance.

A mass of medical data does not have final conclusions, and the general recommendation cannot be identified as the disease of the subject in most cases, it means that most of the data are unlabeled. Such as the hospital experts mark their analysis on the label one by one, which not only does not play the role of assistant physician, but also increase the workload of the physician. Thanks to the implementation of unsupervised and semi-supervised learning on neural networks, we can also use deep learning to train unlabeled data. The supervised learning is a process that the learning machine guides the learning by label, which could establish a model for predicting unlabeled samples. The unsupervised learning focuses only on unlabeled samples, its task is only to describe the way data are organized or clustered. Combining with the above two approaches, the implementation of semi-supervised learning is a process based on labeled training samples and utilized by unlabeled samples themselves. Deep autoencoder is a network which can be applied for semi-supervised classification, in the case of sufficient unlabeled data, it only requires a small amount of labeled data to complete the classification task.

2 Related Work

Autoencoders have been relatively long history and have matured. Rumelhart proposed the concept of autoencoder and applied it to high dimensional complex data processing in 1986. That promotes the development of neural networks. The simplest autocoder includes an encoder and a decoder that implements the mapping of data to itself. Of course, the ability of the single layer of autoencoders is limited to expressing data. The stack autoencoder is subsequently presented. The structure trains an autoencoder every time, then it removes the decoding layer and the encoded results will train the new encoder as input, so on we will eventually get a chain encoder. The advantage is that it contains a number of encoders, each layer is an abstract representation of the

different characteristics of the data, so its ability to express data is greatly enhanced. But this structure also has the characteristics of training difficulties. On the one hand the structure will take a lot of time to carry out a complete training of each layer of encoder, on the other hand, In the case of a certain amount of data, each layer uses the same data for repeated training, it will easy to appear network fitting problem. Then Hinton improved the structure of the prototype autoencoder, which in turn produced the DAE. This structure can be a one-time training in the network of encoders which can greatly improve the speed and reduce the risk of fitting.

In order to solve the problem of autoencoder is difficult to train and make it have better application in a particular scene. A series of related work has emerged. Benjio proposed the concept of coefficient autoencoders that further deepening the DAE research in 2007[1]. Vincent proposed the noise reduction autoencoder, it adds corrupt vector into the input data to prevent over-fitting phenomenon and that achieved good results in 2008[2]. Benjio summarized the existing depth structure and expounded the general method of constructing deep learning neural network by stacking autoencoder in 2009[3]. Salah limited the process of raising and reducing dimension and proposed a shrink autoencoder in 2010[4]. Telmo studied the performance of DAE trained with different cost functions, pointing out the direction for the development of cost function optimization strategies in 2013[5]. Taylor explored the relationship between DAE and unsupervised feature learning and detailed how to construct different types of depth structures using autoencoders in 2012[6].

In recent years, the study of neural networks has proposed many effective schemes including ReLU and BatchNormalization. The ReLU activation function simulates the activation model of neurons accepting signals from the biological point of view. It obtains sparse activation by unilateral suppression. The sparsity makes the key factors in the data be preserved and strengthened and the secondary factors (noise) discarded. It causes the robustness of the data can be enhanced. Many practices prove that ReLU compared to Sigmoid function is a better choice. In the case of Loss large network convergence speed is faster than Sigmoid. Batch Normalization solves the Internal Covariate shift problem by normalized processing of the output of the network layer, so that the data transmitted in the network is in a relatively stable distribution. Because the network does n't have to adapt to the distribution of large input data, it can accelerate the training of the network. Batch normalization also can make the network use a greater learning rate. This results in a fairly good convergence of the general network weights initialization situation.

3 Our Work

As mentioned above, the deep autoencoder can effectively learn the distribution and characteristics from the untagged data set, which is of great significance in semi-supervisory training. We can use the untagged data training network model to modify the network structure and add the classifier, then training classifier with a small amount of tagged data alone, in order to achieve the effect of classification. Because the number of diseases is large and there is a clear distinction between the boundaries of one disease and between different indicators, we first focus on a class of diseases for analysis and research. We can easily extend this model to apply to other disease prediction. We choose liver disease as our first network prediction category. We select six categories of common liver disease I as the predictor, and select gender, age, and 12 indexes which have a strong correlation with the disease and 6 indexes which have a weak correlation, as input: X = {x_1, x_2, ..., x_20} (Table 1). The network will use these indicators to predict the final possible outcome diseases.

3.1 Data Processing

We have fetched the data from one hospital and the origin data is in a mess, so the collected data needs to be preprocessed. Firstly, we should extract indexes related to liver diseases, and discard observations with too many NAs. The remaining NA data is done by randomly generated in the normal range. Secondly, the distribution of the examination indicators are all positive and the difference between the indicators is large, which will not be conducive to network training, leading to slow convergence or even no convergence. Here, we use the simplified Batch Normalization, in order to make the mean of all data to 0 and the variance to 1. For each input data X = (X1, X2, X3 ... Xk), we normalize each dimension in the following way:

$$x_k^{'} = \frac{x_k - E\left[x_k\right]}{\sqrt{v\left[x_k\right]}} \tag{1}$$

For the untagged pre-training data and the tagged fine-tuning data, the above processing is required to obtain the data that can be used.

3.2 Improving the Network

Similar to the traditional Deep Auto encoders used for classification, the network will be divided into the pre-training part and the fine-tuning part. Table 2 shows our pre-training network structure. First, the data is passed through a series of encoders to compress the data into a low-dimensional representation, which will then be restored to data with the same dimension as the input through a series of decoders. Finally, the output with the original input are passed into squared error layer to calculate the deviation and propagates backwards, optimize the parameters of the network:

$$x_k^{'} = \varphi_\lambda\left(\phi_\lambda\left(x_k\right)\right) \tag{2}$$

$$\Gamma_{loss} = \frac{1}{2n}\sum_1^n \left\| x_k^{'} \quad x_k \right\|^2 \tag{3}$$

The training of the traditional deep autoencoder is relatively difficult, and the initialization of the network parameters also has a high demand. With the emergence of some new techniques, deep neural network training has become relatively easier. So we consider the following ways to improve the traditional depth of the automatic encoder:

1) Some of the indicators used for prediction are independent with a particular disease, but the sigmoid function assumes that all active outputs are valid, and the ReLU function selectively discards the value of the active output less than 0. So we can think that ReLU activated in the optimization process will discard independent indicators with the disease, and enhance the sparseness of the network. So we use the ReLU activation function instead of the Sigmoid activation in the traditional deep encoders.

2) With the increase of network depth, the stability of the network will decline, and the initialization of the parameters is particularly important. However, the emergence of BN makes the network initialization easier, so we add the BN layers in the network except the last encoding-decoding step, This specification adjusts the distribution of the encoded data to ensure that the next encoder can learn the characteristics of the raw data without problems with the input data. The addition of BN makes the Deep Autoencoder converge faster and easier to train.

In the above process, the depth automatic encoder learns and optimizes the non-label data. It obtains a network that can express itself finally. In order to have the network classification function, the network decoder need to be partially removed and the encoder need to be at the end of a classifier. Then we use the tagged data {x, Labal} to train the classifier individually. the classifier outputs optimize the classifier by Softmax layers and calculating deviations from the data labels. Because the fine-tuning network only needs to optimize the classification layer parameters, only a small amount of label data can be made to have a classification effect. The classifier and error calculation are as follows:

$$z_k = h\left(\phi_\lambda\left(x_k\right)\right) \tag{4}$$

$$\sigma_i\left(z\right) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} \tag{5}$$

$$\Gamma_{loss} = \log \sigma_i\left(z\right) \tag{6}$$

3.3 Network Structure

The final Pre-train network contains four encoders and decoders and the input data is a 20-dimensional vector. The first layer encoder will enhance the representation of the data. The output vector expands to 24-dimensional and compresses it. The three-tier decoder compresses the data to 18,14 and 8 dimensional. Then there are four decoders corresponding to the four encoders, and finally reduce the data to 20-dimensional vectors. At the meanwhile, calculate the square error layer together with the input data and update reversely by BP algorithm. Then there are four decoders corresponding to the four encoders, and finally reduce the data to 20-dimensional vectors. At the meanwhile, calculate the square error layer together with the input data and update reversely by BP algorithm. Table 1 is the PRE-TRAIN network structure.

In the Fine-tune stage, all decoders are removed, and after the fourth-level encoder, a full-link classifier is added. At this stage, the dimension of the output data is the same as the final class. At the last, the loss is calculated by the Softmax layer with the data tag. Table 2 is the fine-tune network structure.

Table 1. Pre-Train Network

Pre-Train Network
encoder1 + bn + relu
encoder2 + bn + relu
encoder3 + bn + relu
encoder4
decoder4 + bn + relu
decoder3 + bn + relu
decoder2 + bn + relu
decoder1
square loss

Table 2. Fine-Tune Network

Fine-Tune Network
encoder1+ bn + relu
encoder1+ bn + relu
encoder1+ bn + relu
encoder1
classifier
softmax + loss

4 Experiments

It is natural that optimize our network by using the stochastic gradient descent with minibatch (based on BP with momentum). Minibatch size is 128 in pre-train stage while 64 in fine tune stage. Since the amount of disease-free data is many times than diseased data, in order to prevent the overfitting of disease-free data, each minibatch in the fine tune phase used 16 positive samples (disease) and 48 negative samples (disease-free) as input. We also use weight decay normalization to prevent the overfitting problem (L2 factor is 10^{-4}), base learning rate is 10^{-3}. During pre-train phase, learning rate will be adjusted several times with factor 0.1.We final get a high accuracy of test task(93%), compare our improved deep autoencoder with common deep autoencoder (use sigmoid and without batch normalization), the convergence rate of

the network has been acceleration, and the accuracy of network have a little improvement. Comparisons with other semi-supervised learning methods still require more experiments, and we will continue to do experiments in the next work.

5 Conclusion

We tried to use the network-optimized deep autoencoder for liver disease prediction and have gotten good results. It could demonstrate that we can accomplish the actual classification tasks in the condition of less labeled data. In addition, due to the uniformity of the physical indicators of the physical examination, when we put all the indicators as input, the deep autoencoder can predict more class of disease effects by fine tuning a trained network.

References

1. Bengio Y, Lamblin P, Popovici D, et al. Greedy layer-wise training of deep networks[C]. Proc, of the 20th Annual Conference on Neural Information Processing System. 2006:153-160.
2. Vincent p, Larochelle H, Bengio Y, Extracting and coin-posing robust features with denoising auto-encoders[C].Proc. of the 25th International Conference on Machine Learning. 2008:1096-1103.
3. Bengio Y. Learning deep architectures for Al[J]. Foundations and Trends in Machine Leaning, 2009,2(1):1-127.
4. Salah R, Vincent P, Muller X, et al. Contractive autp-encoders: Explicit invariance during feature extraction[C]. Proc. of the 28th International Conference on Machine Learning.2011:833-840.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
5. Amaral T, Silva L M, Alexande L A, et al. Using different cost functions to train stacked auto_encoders[C]. Proc. of the 12th Mexican International Conference on Artificial Intelligence. 2013:114-120.
6. Guyon I, Dror G, Leaire V, et al. Auto-encoders unsupervised learning and deep architectures[C]. Proc. of the 28th International Conference on Machine Learning.2012:37-50