Feature Pre-processing Approach to Prediction of Post-Operative Survival in Lung Cancer Patients

Min Liu
Research School of Computer Science
Australian National University
U6339307@anu.edu.au

Abstract. Training accuracy is controlled by many parameters, such as the number of features, the number of hidden layers, etc. In this paper, I focus on one of these parameters, that is the input patterns. In order to prove the effectiveness of characteristic input pattern, I use another approach called genetic algorithm to choose the input features. By comparing characteristic input pattern with the genetic algorithm, I can fully understand which of the two methods is more effective. The dataset used in this paper is Thoracic Surgery data set. This dataset is used for predicting the post-operative life in one year period of the lung cancer patients. After analyzing all the features that impact on the prediction, I set some rules to restrict the prediction. The result of setting the rules is that the prediction accuracy increases, and it is better than genetic algorithms. However, by comparing the other paper which uses the same data set, my result is worse than the other paper.

Keywords: neural network, setting rules, post-operative survival prediction, genetic algorithm, feature selection

1 Introduction

As it is important for doctors to determine whether the patient should undergo surgery, a rational prediction based on the features of the patient could provide a reasonable suggestion for the doctor. In addition, the decision based on statistical data is more persuasive to the patient [1]. Therefore, it is necessary to find an approach that can improve the accuracy of diagnosis prediction.

Neural network is an effective method to predict disease and it also has been used in different classification problems [2, 3]. Neural network can classify outputs by learning their input features, therefore, proper input features with low noises can improve the prediction [4]. This paper intends to find an approach to obtain suitable features of a dataset and use neural network to verify this approach. The dataset chosen in this paper is Thoracic Surgery dataset, as the prediction method is used for disease prediction, using a disease-related data can be more suitable to verify the method. This data set was collected of the primary lung cancer patients who underwent major lung resections at Wroclaw Thoracic Surgery Centre. 470 instances and 17 features composed the data set. The 17 features included DGN, PRE4, PRE5 and PRE6 which had numeral or multiple values. The other features all had binary values, which means that each feature contained only true or false [5, 6]. The goal of the training is to predict the risk of death in 1 year period after lung resection. The data set fits the problem that we want to solve in the real world. Also, it was suitable for the training, as it had more than 15 features with only two outputs, which makes the training less complicated.

The main approach I use in this paper is characteristic input method [1]. The key to this method is to characterise the input patterns and set rules for them. The advantage of this method is that it can obtain a more accurate output than a normal neural network. Besides, from what Gedeon and Turner said, this method was especially useful in medical domain. In addition, I used a genetic algorithm to select important features. These features are set as input and trained by the same neural network as the characteristic one. By comparing the prediction accuracy of characteristic input method with genetic algorithm, I can choose a better one and apply it to disease prediction.

2 Method

There are many approaches to improve the accuracy of prediction. Besides the basic approaches like learning rate, the number of hidden layers and training epoch, many advanced approaches are coming out. The method used in this paper is to process the input before training. Gedeon and Bowden also provided an approach to process the input, that was heuristic pattern removal [7]. This approach aimed to reduce the size of training set through heuristic pattern removal method. However, if the input size is not big enough, the reduction of input size may lead to a lack of learning, which may decrease the accuracy. Therefore, finding an appropriate amount of the reduction is the key to the success of this approach.

2.1 Characteristic Inputs

There are two processes to process the input patterns to characteristic it. The first is to find the relation between each feature and the output. That is, determine which input feature is important to the output. The small change of an important feature can result in a significant change in output. The second process is to set rules for the output based on the first process. It is easy to be explained using real-world example. Doctors diagnose diseases based on symptoms, if a patient has the classic symptoms of the illness, doctors can give diagnosis easily [1]. The rules in neural network are the same as the classic symptoms.

In order to determine the importance of each input pattern, it is necessary to use statistical methods to analyse it. As there are 16 attributes, there must be at least one feature that is least important. It is easy to find that in the 470 instances, feature PRE19 and PRE32 only has two *True* values, their rest values in 468 instances are all *False*. In addition, when the values of PRE19 and PRE32 are *True*, the outputs are *False*. Since the output contains 400 *False* and 70 *True*, the possibility of the output being *False* is much higher than *True*. It seems that the two features, PRE19 and PRE32, have little effect on the output.

Since it is difficult for me to understand the characteristics of each feature with only 470 examples, I use a paper that has summarized the characteristic features to understand them accurately [8]. Table 1 cited from the relative paper shows that PRE14 and DGN are the two most informative features. Thus, based on this conclusion, I try to set these two features with higher weights than the other features.

Table 1. Characteristic of selected pre-operative features [8].

ID	Description	InfoGain
PRE14	T in clinical TNM (size of the original tumour, from OC11 (smallest) to OC14 (largest))	0.029
DGN	Diagnosis (specific combination of ICD-10 codes for primary and secondary as well multiple tumours if any)	0.013
PRE4	Forced vital capacity (FVC)	0.008
PRE7	Pain (pre-surgery)	0.008
AGE	Age at surgery	0.008
PRE6	Performance status (Zubrod scale)	0.007
PRE11	Weakness (pre-surgery)	0.004
PRE9	Dyspnoea (pre-surgery)	0.004
PRE10	Cough (pre-surgery)	0.003
PRE8	Haemoptysis (pre-surgery)	0.003
PRE25	PAD (peripheral arterial diseases)	0.003
PRE19	MI up to 6 months	0.003
PRE5	Volume that has been exhaled at the end of the first second of forced expiration (FEV1)	0.002
PRE32	Asthma	0.002
PRE30	Smoking	0.002
PRE17	Type 2 DM (diabetes mellitus)	0.002
Risk1Y	1 year survival period ((T)rue value if died)	

2.2 Implement Neural Network

After theoretical analysis of this dataset, I start building a neural network. I use PyTorch to build a simple neural network without using the advanced neural network structure such as CNN or RNN. For pre-processing the dataset, all the *True* and *False* values are changed into 1 and 0. The features which values are string type are changed into integers. The dataset is split into 80% training set and 20% testing set randomly. The number of hidden layers is set to be 10 and the learning rate is 0.001. The network is trained for 1000 times.

Rules should be set based on the theory mentioned previously. From the dataset, I set two rules for the outputs.

1. if output = True:
DNG ≠ 1 or 6;
PRE4 < 5;
PRE5 <= 4.16.
2. if output = False:
PRE4 > 5;
PRE5 > 50.

These rules do not mean that the input which does not follow the rules is useless. It only means that if the input meets the above rules, the output can be accurately predicted. But if the input does not meet the rules, it is still possible to get a right output.

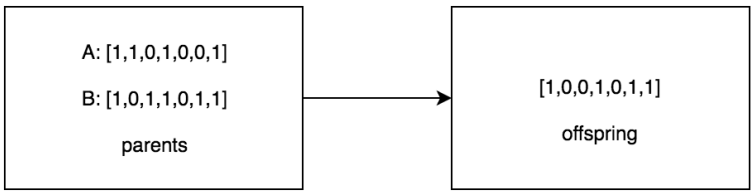
The implementation of neural network divided into three parts. The first part is loading and pre-processing data. Since PRE19 and PRE32 are least important, I delete these two features in order to reduce the training time. After analyzing the informative features, I choose DGN and PRE14 to be the most important features and set more weights to them. The second part of neural network is building neural network. A basic neural network has input neurons, hidden neurons, output neurons, learning rate and training epochs. Learning rate and training epochs have a great influence on the prediction accuracy, thus choosing a proper learning rate and training epoch are the most important in a neural network. The activation function I choose is sigmoid. As the neural network is not complicated, it is possible to use any one of the activation functions. The loss function in the neural network is Cross Entropy Loss, as it is suitable for measuring the performance of a classification model [11]. After building the neural network and training the dataset, the third part is testing whether the network performs well.

2.3 Genetic Algorithm

Genetic algorithms (GAs) are inspired by Darwin's evolutionary ideas of natural selection and genetics [9]. They are often used in optimization problems, like finding the shortest path or the shortest cost. However, by transforming the prediction problems into finding the optimal prediction accuracy, GAs are also used in feature and instance selection. GAs have already been used in prints' classification, heart disease classification and classification of emotions on the human face [10].

In order to select important features using GAs, the first thing I do is to define a fitness function. Like the chromosomes in our body, GAs also have chromosomes, which are composed of 0 and 1. As the fitness function is to obtain an optimal prediction accuracy, each 0 or 1 in a chromosome represents the input feature of the dataset. 1 means the feature is used and 0 means the feature is not used. The whole chromosome indicates the condition of input features. In the fitness function, the dataset is trained by logistic regression to calculate an accuracy. The chromosome with high prediction accuracy has more probability to be selected as the parent. Crossover is used to combine part of the two parents together. Figure 1 is a crossover example. The odd position of the offspring is selected from parent A, the even position is selected from parent B. After the child chromosome is generated, mutation operator will randomly change the number in some positions of the child. The above steps are repeated until the number of generation is reached.

Figure 1. Crossover operation



The population of GA is set to be 200, and generation number is 10. I do not pre-process the data before using GA. Two results are obtained from the experiment.

3 Results and Discussion

Due to the lack of approach to set rules for a neural network, I cannot verify whether setting rules can improve prediction accuracy. But characteristic input features by duplicating important features many times can improve the prediction accuracy to some extent.

Table 2. The accuracy of 10 tests for each dataset.

	Original dataset	characteristic dataset	GAs dataset
1	86.02	90.12	93.88
2	80.25	88.51	88.00
3	82.29	91.53	87.50
4	82.72	82.76	81.65
5	87.95	86.79	88.35
6	83.33	86.52	87.23
7	83.87	84.54	81.25
8	81.32	87.06	87.95
9	84.62	88.89	86.90
10	87.64	84.95	86.67
Average	84.00	87.17	86.94

The neural network first trains the original dataset with no dropping and duplicating columns. The mean testing accuracy for ten tests is 84.00%. Then I use processed new data with characteristic features to train the network. After ten tests, the mean testing accuracy raises up to 87.17%. Although the extent of improvement is not large, this approach is still useful. I also use a pre-processed data that removes the unimportant features from GA's results. As there are two results in GA, I try both of them and choose a better one to list in this paper. The features which I keep in the dataset are PRE5, PRE7, PRE25 and AGE. The dataset is trained by a same neural network with same parameters. The mean testing accuracy is 86.94%. Although the mean accuracy is slightly less than the characteristic one, it is still better than the dataset with no pre-processing.

There is a problem that can be found in this result. When the neural network is trained by the original dataset, the testing accuracy is not very high, even cannot reach 90%. Though I change the parameters such as the number of hidden layers and the learning rate many times, the testing accuracy is still between 80 and 90. This issue may be due to many reasons. The first one might be the improper training set. The size of the dataset is small. 470 samples may not enough to get a high prediction accuracy. Also, the data is imbalanced. Since the values of these features are mostly binary, these binary values are not evenly distributed in the dataset. For example, PRE7, PRE8 and PRE9, the *False* value appears more than 400 times, almost six times the *True* value. What is more, there is almost no *True* value in some features. This may be the key to low accuracy. The second reason may be that the data used for training and testing is randomly divided. For example, ten tests were conducted using the data set up by the rules, and two of them had an accuracy of more than 90%, but the accuracy once was even lower than 80%. These differences may be due to random training and testing data.

By comparing the result of the characteristic dataset with the GAs dataset, there are also some interesting findings. The first one is that the highest prediction accuracy of the GAs dataset is 93.88%, which is higher than the highest accuracy of the characteristic dataset. This result shows that sometimes the features obtained by the genetic algorithm are more accurate than the characteristic features. This result may be due to the lack of accurate characterization of the input features, since the characteristic method I use cannot implement the rules I set. The second finding is about the genetic algorithm. As I mentioned above, there are two kinds of features obtained from GA. Although I choose the better one, the two kinds of features can also indicate that the experiment is not very accurate in the use of GA. As the dataset only have 470 instances, it is possible to get the same prediction accuracy from different parents in the regression of GA. In fact, when I set the population to 100, there are more than ten situations where the validation accuracies are the highest. In this situation, I find it difficult to choose the most suitable input features. Therefore, if I improve the feature selection approach using genetic algorithm, the prediction accuracy might be improved too.

Another interesting thing is that the other paper which used Thoracic Surgery dataset also extracted rules from the dataset. They used a method called boosted SVM to extract rules [8]. They extracted rules like (DGN = DGN5) => Risk1Yr = T and (PRE14 = OC13) => Risk1Yr = T. By setting rules, the accuracy can be 97%. Although their paper is to compare different methods for extracting rules, the different accuracies between each method prove that setting rule can improve the prediction accuracy.

4 Conclusion and Future Work

Due to the limitation of PyTorch, setting rules for the output cannot be implemented. But by comparing the other paper which uses the same dataset, it is easy to find that setting rules can help improve accuracy. In order to verify the theory presented in this paper, I need to use other training methods to set rules for neural networks in the future work. In addition, there are many ways to set rules and I need to compare them to choose the most effective method. Zieba et al. indicated that extracting rules is useful for dealing with imbalanced data [8]. For balanced data, whether the extracting rules can improve its predictive accuracy is what needs further study. Furthermore, what I conclude in this paper is not comprehensive enough. The result of the experiments is that feature characteristic approach is better than feature selection using the genetic algorithm, but the result may be changed in some situations. For example, the feature selection approach using GA need to be improved, which may affect the prediction accuracy. Therefore, we still have a lot of work to find the most appropriate disease prediction method.

References

[1] T. D. Gedeon and H. S. Turner, "Explaining student grades predicted by a neural network," in *International Joint Conference on Neural Networks*, Nagoya, 1993, pp.609-612, vol.1.

[2] F. Åström, "A parallel neural network approach to prediction of Parkinson's Disease," Expert systems with applications, vol. 38, no. 10, pp. 12470-12474, 15 September 2011.

[3] S. Agrawal and J. Agrawal, "Neural Network Techniques for Cancer Prediction: A Survey," in International Conference on Knowledge Based and Intelligent Information and Engineering Systems, Singapore, 2015.

[4] C.-F. Tsai, W. Eberle and C.-Y. Chu, "Genetic algorithms in feature and instance selection," Knowledge-Based Systems, vol. 39, pp. 240-247, February 2013.

[5] Lubicz, M, Thoracic Surgery Dataset, [http://archive.ics.uci.edu/ml/datasets/Thoracic+Surgery+Data#]. Wroclaw University of Technology, wybrzeze Wyspianskiego 27, 50-370, Wroclaw, Poland

[6] Pawelczyk, K, Rzechonek, A & Kolodziej, J, Thoracic Surgery Dataset,
[http://archive.ics.uci.edu/ml/datasets/Thoracic+Surgery+Data#]. Wroclaw Medical University, wybrzeze L. Pasteura 1, 50-367 Wroclaw, Poland

[7] T. Gedeon and T. Bowden, "Heuristic pattem reduction," in International Joint Conference on Neural Networks, Beijing, 1992, pp. 449-453.

[8] M. Zięba, J. Tomczak, M. Lubicz and J. Świątek, "Boosted SVM for extracting rules from imbalanced data in application to prediction of the post-operative life expectancy in the lung cancer patients," Applied Soft Computing, vol. 14, pp. 99-108, 2014.

[9] Imperial College, "Genetic Algorithms," Imperial College, [Online].
Available: https://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol1/hmw/article1.html#top. [Accessed 28 May 2018].

[10] R. Robu and H. Stefan, "A genetic algorithm for classification," in International Conference on Computers and Computing, Harbin, 2011.

[11] R. DiPietro, "A Friendly Introduction to Cross-Entropy Loss," GitHub, 2 May 2016. [Online]. Available: https://rdipietro.github.io/friendly-intro-to-cross-entropy-loss/. [Accessed 28 May 2018].