Applying a GA-based Feature Selection with Using Support Vector Machines on Bidirectional Neural Networks to Predict Final Marks

Cheng Yixin

Research School of Computing,
The Australian National University, Canberra, Australia,
u6809382@anu.edu.au

Abstract. Feature selection is vital both to speed up learning and to improve accuracy in deep learning tasks. This paper presents a genetic algorithm (GA)-based feature selection approach by using Support Vector Machines (SVM) and applies this method on simple neural networks (SNNs) and Bidirectional Neural Networks (BDNNs) respectively. In this studies, four different models are built to compare the performance on mark prediction to validate the proposed approach. The results indicate that the accuracy with using feature selection SNNs is reaching 98% which is 9% higher than the normal SNNs' and the accuracy of using feature selection BDNNs is reaching 76% which is 12% higher than the normal BDNNs' and 2% higher than A.F. Nejad and T.D. Gedeon's work[1]. This work shows that proposed method has the significant improvement on performance of models. This paper is a follow-up to Y. Cheng's work which reproduced A.F. Nejad and T.D. Gedeon's work[7].

Keywords: BDNNs, Mark Prediction, Genetic Algorithm, Feature Selection

1 Introduction

1.1 Background and Motivation

Improving student's academic performance is always the priority and challenge for university. Predicting student's performance can decrease the rate of drop-out, improve the success rate of students and make cost-saving decisions for institutions since it might allow institutions to take pre-emptive measures to help students[6]. Therefore, the prediction of performance is vital to universities and students themselves.

As for BDNNs, it is similar to electric transmission in the real world. it has been widely applied in various fields including generalization error reduction[2], confidence estimation[3], speech recognition[4] and so on. This paper will address the implementation of BDNNs on Student Final Marks (SFM) classification by using the partial marks of students related to the course and compare the results with the simple neural network.

In the real-word deep learning, the data sets sometimes have high complexity in terms of large number of features and volume. As for features, sometimes there are redundant or important. In this case, conducting features selection is vital to the deep learning project in speeding up learning and improving the accuracy[8]. This paper continues improving the performance of BDNNs which has been reproduced by Cheng [7] with introducing feature selection.

In this paper, 4 various models are built on Student Final Marks (SFM) classification by using the partial marks of students related to the course and compare the results with simple neural network. To be specific, the first model is simple neural networks (SNNs); followed by Bidirectional neural networks (BDNNs); then this paper implements genetic algorithm (GA) for features selection which applies on above two neural networks.

1.2 Problem Statement and Investigation Outline

Compared with a simple neural network, BDNNs enables output layer learning from backward and forwards. By building one-to-one relations between input vectors and output vectors, inputs and outputs neurons can connect hidden layers with two directions. However, whether BDNNs is suitable for the classification of marks is to be determined. According to the given paper, the authors implemented BDNNs on two data set, that is SFM and Gross National Product (GNP) prediction. During the process, the authors proposed two tasks, that is, context addressable memory and cluster. To achieve these, the authors created 3-layer neural networks (one input, one output, and one hidden layer), introduced the same weight connections between layers, implemented the back-propagation technique in BDNNs to get weights adjustment, built a one-to-one relation data set for implementation, etc. Finally, the authors got 74% accuracy on SFM for the first task. As for the last task, the authors trained BDNNs

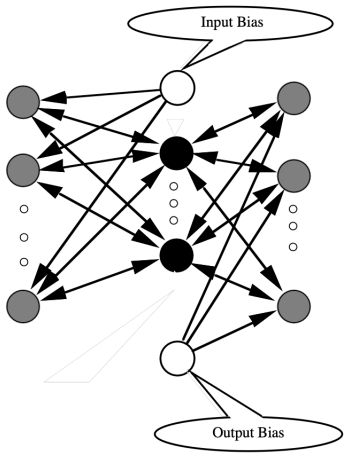

Fig. 1. BDNNs

as a cluster center finder, achieved 72% on SFM and 62% accuracy on GNP[1]. However, given paper was not compare its performance with simple neural network. Furthermore, this paper did not conduct optimization process. Thus, the objective of this paper is reproducing BDNNs and applying GA on SFM and validating the best performance models by comparing accuracy.

1.3 Activities

To achieve the objectives, this paper carries out the following tasks:
Construct the genetic algorithm-based feature selection. Build a simple neural network that classifies the students' final marks. Build a data set contains students' academic information (marks from different assessment in COMP 1111) from the origin data set. Build a simple neural network. Test the data set on the built neural network. Build BDNNs that classifies the students' final marks. Build a suitable dataset contains students aggregated academic information which can be used on BDNNs. Build BDNNs. Test the data set on the built neural network. Compare the results between BDNNs and simple neural network.

1.4 Outline

Section 2 will introduce the methods this paper used. Section 3 will show and discuss the results. Section 4 will give the conclusion and future work.

2 Methodology

2.1 Data Description and Pre-processing

The given data is COMP 1111 students' grades from University of New South Wales (UNSW)includes 153 students' academic records with 16 columns, that is, Regno (registration number), Crse/Prog (course/program), S (semester), ES (enrolment statues), tutorial group, lab2, tutorial assessment, lab4, homework1, homework2, lab7, p1, f1, mid, lab10 and final. Note that all of data is shown in one column in the given data set. Moreover, there are 327 missing values throughout the data set. Furthermore, the data type also varies from categorial (Crse/Prog, ES, Tutgroup, Regno, and S) to numeric (rest of the columns) which is suitable for different processing.

Based on the data type and columns characters, the first step is to split and reorganise the organise data set. After this step, a new data set generated, and values are across different columns and rows. In each of columns, the format of values varies. For example, in the Cre/Prog, ES, Tutgroup columns, the values are in string. For string type columns, In Regno and final, the values are integer. So next step is to transform data and detect outliers. For the outliers, I replaced them with NaN which means Null in the data set. For categorical type, I use different integer to represent each category. Furthermore, I normalize the data to [0-1] in float or integer column based on their full mark, for example, a student got 2 mark in lab2 (full mark is 3) so this value is 0.67. based on this principle, I

COMP1111 More Computing Sorted on student ID

Regno	Crse/Prog	S	ES	Tutgroup	L	lab4		h2		p1		mid		final	
						tutass		h1		lab7		f1		lab10	
						3	5	3	20	20	3	20	20	45	3	100
0168826	3971	2	FS	T1-yh		2	3	2.5	19.5	.	2.5	11	.	33	2.5	71
0168883	3971	2	F	T9-ko		3	5	2	20	17	.	8	.	27	2.4	67
0168907	3400	1	F	T6-no		3	3	3	10	.	2	.	.	9.5	2.4	30
0169379	3970/1061	2	F	T3-ku		2	3	2	20	19.5	2	15.5	.	21	.	62
0169717	3970/1061	2	F	T10-yh		2	3.5	1.5	19	15.5	2	17.5	.	13	2.5	58
0170092	3971	2	FS	T1-yh		3	3.5	2.5	20	16	3	16	11	22	2.5	68
0173607	3971	2	F	T4-ko		3	5	2.5	17	16	3	18	15	34	2.4	75
0174270	3970/6806	2	F	T8-no		2.5	3	2	18	18.5	2.5	9.5	15.5	17	2.4	62

Fig. 2. Part of marks

transformed all numeric data. As for string type value, I created categories to represent them. To be specific, for Turtgroup, scale from 1 to 10 represents 1-10 tutorials then normalize them into 0-1. For Cre/Prog, get the course number then transform them into 0-1. For ES, transform it to number from 1-3 respectively representing FS and FL then transform them into 0-1. For grades, scale from 0-1 represents from 0%-100% respectively. Then I removed the first column since it is meaningless as input. In the final step, transform the final column to 1 to 4 to represent F to D.

As for missing data imputation, first, I tried to keep the NaN values in the data set, however, the result is quite low and unsatisfying. Therefore, I used the Random Forest algorithm to implement because for grades since this method can have a better performance than others in terms of imputation[5]. To be specific, the order of imputation is from the least column to the most ones first, fill the missing values with 0 in other columns then run the algorithm and do the iteration until each missing value will be handled. Then, I normalize all the input data except the final with the Z-score method.

2.2 GA-based feature selection construction

To implement this functionality, first parameters of GA were defined. To be specific, DNA size is 14 which stands for 14 features, population size is 200, cross rate is 0.75, mutation rate is 0.002, total generations are 100. As for fitness function, I used SVM to calculate the accuracy which represents the fitness which has a satisfying performance as a fitness function when accuracy is the fitness value [9]. The kernel of SVM is linear, 80% training and 20% test. In terms of selection, I used random selection which is 0.75. As for crossover, I used sexual uniform crossover. In the end, as for selection, I defined population select function based on fitness value and population with higher fitness value has higher chance to be selected. Note that the termination condition will trigger when the max generation reached. The process of evolution is shown in figure 3

2.3 SNNs Construction

To build the SNNs, I used the first 14 columns as input, which means input size is 14. Since output is 4 (F, P, C, D), so the output size is 4. Other hyper Parameters includes batch size (10), learning rate (0.01), hidden size (50), epochs (70). The optimizer is Adam and I conduct a back-propagation to adjust weights. 80% training data and 20% testing data. Overall, this combination is the most effective and satisfying.

2.4 BDNNs Construction

To build the BDNNs, I re-modified the training data set and testing data set to one-to-one relation (each output vector has only one representative input vector) for fitting this model. To do that, categorizing final mark as 4 classes, as for each type, representing it with an integer. Therefore, getting the mean of 4 classes. After that, creating 3 sub-classes for Fail, 6 sub-classes for Pass, 6 sub-classes for Credit and 3 classes for Distinction respectively[1]. For example, I create sub-classes for fail is F1 = 0.9 * F - 0.1 * P, F2 = 0.8 * F - 0.2 * P and F3 = 0.7 * F - 0.3 * P. Therefore, 22 patterns and 15 columns one-to-one relation data set was created. Then, in terms of hyper parameters,

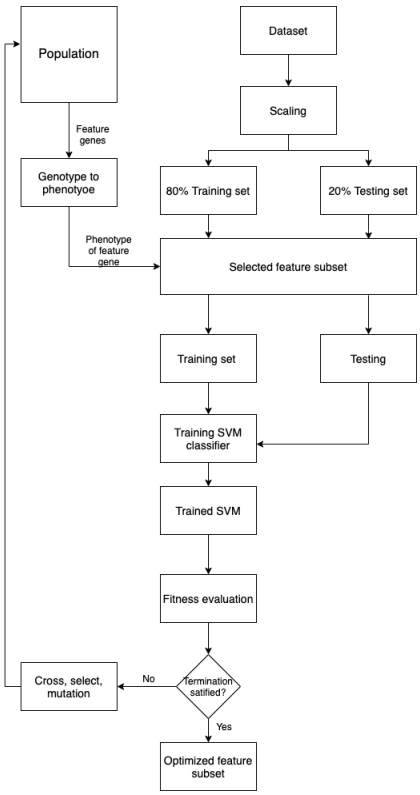

Fig. 3. GA-SVM feature selection

input size = 14, hidden size = 50, num classes = 1 epochs = 150 batch size = 14 learning rate = 0.01 and extra bias (initialized into 0) combined to the model. The optimizer is AdamW and I conduct a back-propagation to adjust weights. 80% training data and 20% testing data. As for training, the first 50 epoch, BDNNs was trained as a simple neural network. As for the next 50 epochs, the direction of training is reversed. In the final 50 epochs, the direction is reversed again to complete the whole training.

2.5 SNNs and BDNNs with GA

In the application of GA on SNNs and BDNNs, except for necessary and must settings modification such as input size, I kept the same parameters settings with models in 2.3 and 2.4 subsection which means apart from the different data set used, these two models keep as same as possible in terms of settings which can be helpful in later evaluation and discussion (GA is the control variable).

3 Results and discussion

Table 1. Results of four models

Model	SNNs	BDNNs	GA+BDNNs	GA+SNNs
Accuracy	89%	64%	76%	98%
Loss	1.4	0.18	0.22	0.48

In SNNs, the testing accuracy is 98% at most with loss 0.48 in testing which means the GA play a key role in the improvement of accuracy in SNNs. In BDNNs, the testing accuracy reached 76% at most and loss is 0.22. this result is 2% higher to the given paper. The reasons why GA has a satisfying performance are excluding abundant columns which have less contribution than others and getting rid of the extra noise.

The results in SNNs and BDNNs vary. In the SNNs, the testing accuracy is 89% at most with loss 1.4 in testing which means SNNs successfully classified 88% of student marks in testing. in the BDNNs, the testing accuracy is 64% which means BDNNs only correctly classified 64% of student marks in testing and corresponding loss is 0.18. Although the BDNNs model correctly majority of marks, compared with SNNs and results on given paper, the result is unsatisfying. Compared with given paper, this result is about 10% less (74% in given paper) [1] which is also unsatisfying. The latent reason why this result happened, I suspect the control of direction reverse is problematic, I did a lot of experiments. For example, I set up a loss threshold, if the over loss in one direction training exceeds the threshold, the direction will be reversed. I set up it to 1.1, then the accuracy is 42%. Then I did another experiment, the condition of direction will not reverse until the over loss is less than the over loss in previous direction of training, the testing accuracy is 33%. Overall, 3 ways have been tried and the first one has the best performance. However, it is unknown that which way the given paper used and what corresponding set-up was. The reason why simple neural network performed well is due to the imputation by Random forest which this algorithm played a key role in the training and proper settings of hyper parameters. According to the result shown, the BDNNs is not suitable for this data set.

4 Conclusion and Future Work

A mark prediction based on neural network has been carried out respectively. In the SNNs, the result is good, and this model can deal with this data set. But BDNNs preformed worse than SNNs on current data set and this paper invalidates the BDNNs. However, with using GA feature selection, the performance in both models is improved. As for the future work, there are many improvements, such as softening the sharp edges between sub classes to make them more natural interval, improving the encoding format, applying more advanced technique in training, experimenting more functions in calculating fitness, etc. Most importantly, I can make more improvements on the direction reversing because I think this leads to the low accuracy. In terms of extensions, this technique can be extended many data sets such as predicting climate with some import index or classifying problems.

References

1. A. F. Nejad and T. D. Gedeon : Bidirectional neural networks and class prototypes," Proceedings of ICNN'95 - International Conference on Neural Networks. IEEE(1995).
2. Nejad A.F., Gedeon T.D.: Bidirectional Neural Networks reduce generalisation error. IWANN (1995).
3. A. Ragni, Q. Li, M. J. F. Gales and Y. Wang. : Confidence Estimation and Deletion Prediction Using Bidirectional Recurrent Neural Networks. IEEE(1999).
4. A. Graves, N. Jaitly and A. Mohamed. : Hybrid speech recognition with Deep Bidirectional LSTM. IEEE(2013).
5. Smith B.I., Chimedza C., Bührmann J.H. : Random Forest Missing Data Imputation Methods: Implications for Predicting At-Risk Students. ISDA(2019).
6. Patron,R. : Early school dropouts in developing countries:An integer approach to guide intervention: The case of Uruguay. Journal of Economics (2008).
7. Y.Cheng. :Applying BiDirectional Neural Networks on Mark Prediction. 4th ANU Bio-inspired Computing Conference (2021).
8. Kenji Kira and Larry A. Rendell. 1992. A practical approach to feature selection. In Proceedings of the ninth international workshop on Machine learning (ML92). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 249–256.
9. Huang, C.-L., Wang, C.-J. (2006). A GA-based feature selection and parameters optimization for support vector machines. Expert Systems with Applications, 31(2), 231–240.