Using Evolutionary Algorithms to optimize BDNNs to generate mean clusters for student grades

Shashank Gummuluru

Research School of Computer Science
Australian National University
u7144557@anu.edu.au

Abstract: Neural networks were trained to predict the final grade of a student using instances of their internal assessments. The final marks of students of COMP1111 at UNSW were used to assign them one of 4 grades. A bi-directional neural network (BDNN) was trained to find mean data centers (or cluster centres) of these assessments for each grade. The BDNN was able to achieve up to 57% accuracy on previously unseen data. This is in accordance with previous results in this space (Choi, Gedeon 1995). This paper utilizes evolutionary algorithms to optimize the forward pass of the BDNN. The clusters generated by the BDNN were consistent with previous works on this topic, namely, (Nejad, Gedeon 1995).

Keywords: Mark-prediction, Bi-directional neural networks, mean clusters, representative clusters, Evolutionary Algorithms

1 Introduction

In today's world of remote education, it is getting more and more difficult to conduct traditional exams. Oftentimes, the integrity of such online exams is compromised and the results are not as reliable as they would have been if the exams were taken in person. Having a method to use internal marks of diverse assessments to be able to predict the final grade of students can prove invaluable to instructors. This paper aims to do exactly that. As a secondary goal, the paper aims to generate mean clusters which are representative of their cluster centre (the grade in this case) and compare the results with the results in (Nejad, Gedeon 1995).

The paper implements a bi-directional neural network to predict the final grade of the student using their internal assessment marks. It goes further to generate mean cluster centers for each grade. A mean cluster center, in this context, is a representative vector which best represents the vectors of that class (grade in this instance). The forward pass of the network is trained like any other fully connected network. The backward pass is then run to generate the mean clusters. These clusters were tested by assigning each data point to the 'nearest' cluster. Evolutionary algorithms were used to optimize the hyper parameters like number of hidden layers/neurons, the learning rate and the number of epochs the model is run over.

The dataset that was used to train and test the model was the internal assessment marks of students of the COMP1111 (placeholder course code) course at the University of New South Wales (Choi, Gedeon 1995). The dataset was filled with Null/NaN values at random columns and these were handled by dropping columns which contained more Null/NaN values than an assigned threshold value.

2 Methodology

2.1 Cleaning the dataset

The first thing done to the dataset was assigning a grade corresponding to the mark. The assignment was done as follows:

- Distinction, being a mark of 75 or above, was encoded as 1
- Credit, being a mark between 65 and 74 was encoded as 2
- Pass, being a mark between 50 and 64 was encoded as 3
- Fail, being a mark below 50 was encoded as 4

The dataset was filled with Null/NaN values and suitable measures were taken to ensure that the network was not affected by this. Any columns with 25 or more null values were dropped and not included in the processed dataset. This did not affect the distribution of grades significantly as is presented in figure 1.

The reason for not assigning the Null/NaN values to 0 or something similar, is due to the fact that enough data of how the data was collected is not available and assigning values without this knowledge could lead to significant noise in the data.

The summaries of the raw and processed data is available in table 1. The categorical columns, namely, 'crse/Prog', 'ES', 'tutGrp' were encoded using label encoding assigning each of them an integer value. After this, all columns were normalized to values between 0 and 1, using formula (1).

$$x' = (x - \min(x)) \ / (\max(x) - \min(x)) \tag{1}$$

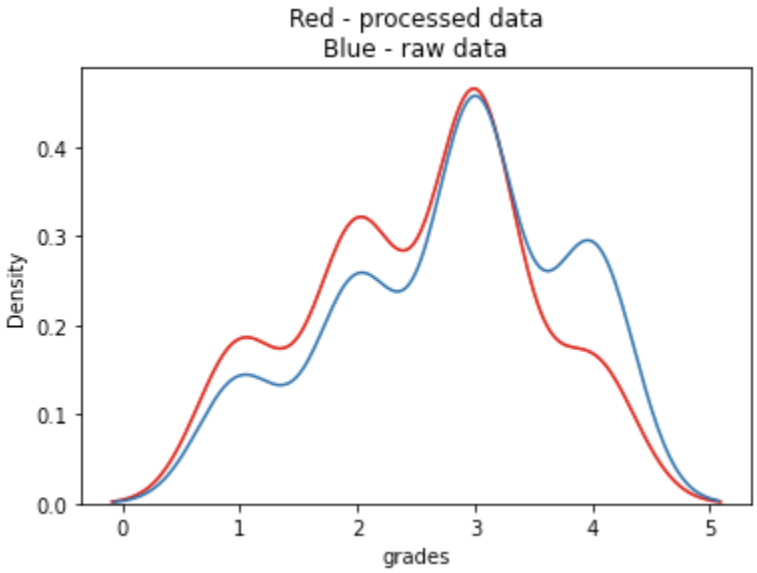

Fig. 1 The distribution of grades for the raw data and the processed data

Attribute	Non-empty columns	Data type
RegNo	153	Numeric
Crse/Pro	153	Object
S	153	Numeric
ES	153	Object
tutGrp	153	Object
lab2	129	Numeric
tutass	145	Numeric
lab4	130	Numeric
h1	132	Numeric

h2	99	Numeric
lab7	117	Numeric
p1	113	Numeric
f1	94	Numeric
mid	143	Numeric
lab10	111	Numeric
final	145	Numeric

Attribute	Non-empty columns	Data type
RegNo	111	Numeric
Crse/Pro	111	Object
S	111	Numeric
ES	111	Object
tutGrp	111	Object
lab2	111	Numeric
tutass	111	Numeric
lab4	111	Numeric
h1	111	Numeric
mid	111	Numeric
final	111	Numeric
grades	111	Categorical

Table 1: Showing the summaries of the the (i) raw dataset (top) and (ii) processed dataset

Even though the classes in the dataset are not perfectly balanced, undersampling of the majority class was not performed. This was because the dataset was already very sparse and undersampling from certain classes will only reduce the amount of data available to train the model.

2.2 Evolutionary Algorithm

Evolutionary algorithms were used for hyper parameter optimization in the forward direction of the neural network. These were chosen for two reasons. The first reason was that there is no hard and fast way to optimize the hyper parameters and an evolutionary approach is considered to be a good one (Jaderburg, 2017). Another reason was that evolutionary algorithms are considered to be good at approximating solutions to problems that don't really have a rigid solution (Vikhar, 2016). A mean cluster, in essence, is an approximation and using these algorithms may produce more representative clusters.

A custom representation was used for each chromosome. It was decided that only one hidden layer will be used since using 2-hidden layers did not significantly improve accuracy while increasing the overall complexity by a lot. The first element in the list was the number of hidden neurons in the network, the second was the number of epochs and the final element was the learning rate of the model. The attributes used were not included in the analysis since the intent of the paper is to generate clusters and having more attributes implies a better representation.

The evaluation function maximized the accuracy of the network on unseen data. So for each iteration of the algorithm, a new instance of the network with the hyper parameters was created and tested. The selection method was the tournament method with a tournament size of 3. The rate of mutation was 0.7 and the crossover rate was 0.02. Due to the limitations of the

author's machine, the algorithm was run only for 100 generations with a population size of 50. Fig2 outlines the fitnesses by generation.

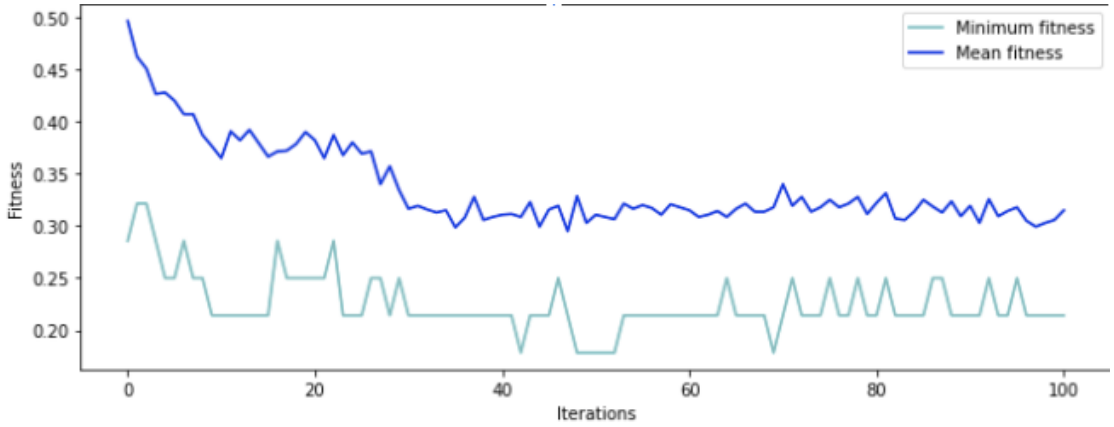

Fig 2: The mean and minimum fitness by generation

The optimal parameters after running the algorithm yielded [20, 111, 0.035]. These were then used as the hyperparameters in the final neural network.

2.3 Bi-directional Neural network

The network has 9 input neurons, 20 hidden neurons and 4 output classes. It takes 9 neurons in its input layer, 20 neurons in the hidden layer and 4 output neurons, each corresponding to a grade defined in 2.1. The sigmoid function is used as the activation function. The 9 input neurons contain the crse/Pro, S, ES, tutGrp, lab2, tutass, lab4, h1, mid values respectively. In the forward direction the network was trained using the batch training method with a batch size of 10 and a learning rate of 0.035 over 111 epochs. The data was split 75-25 for the training and testing set.

After the forward training, the model (at that stage) along with the training inputs is passed in the reverse direction. Here, the model predicts the grades of the training inputs and partitions the data into 4 categories based on the grade. The mean of each of the 9 inputs (crse/Pro, S, ES, tutGrp, lab2, tutass, lab4, h1, mid) is determined for each grade. This mean vector is used as a cluster centre for the testing section. This meant that we were left with 4 vectors, each representing a particular grade. For testing, the euclidean distance between the input and each cluster centre was calculated. The grade assigned to the input was the cluster closest to the input, that is, the cluster with the minimum euclidean distance to the input. The formula is as follows.

$$d(a,b) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2} \quad \textbf{(2)}$$

2.4 Evaluation method

The evaluation of the neural network itself is done by using the Euclidean distance to the center cluster (as mentioned in 2.3). The evaluation of the cluster centers themselves are done by comparing the generated clusters to previous works, namely (Nejad, Gedeon 1995) and analyzing the trends.

3 Results and Discussion

3.1 Accuracy of the BDNN

The network implemented was re-trained and tested on different subsets of the data. It is important to note that under ideal conditions, there would be a validation set that was used on the evaluation function in the evolutionary algorithm part. However, due to the sparsity of the data, this would mean that there would be even less data available for training and hence the network was retrained on the new architecture.

The accuracy of the BDNN on unseen data was within a 10% margin of previous works of 62.5% (Choi, Gedeon 1995). The BDNN achieved a mean score of 57.3% over the course of 10 runs. One point of improvement that this model has over a fully connected neural network that is simply predicting the data, is performance on previously unseen data. The fully-connected basic neural network will have to call the network and pass the data through it to obtain the result, on the other hand, the BDNN will just have to compute the euclidean distance of that datapoint with the 4 mean clusters. This means that the BDNN will predict unseen data faster and is computationally more efficient.

3.2 Mean cluster centers and comparison with technique paper

Table 3: Mean clusters for each grade extracted by the BDNN

Feature	Distinction	Credit	Pass	Fail
Crse/Pro	0.65416664	0.53333324	0.32272732	0.14285715
S	0.41666666	0.375	0.18181819	0.17857143
ES	0.5833333	0.41666666	0.015151516	0.0
tutGrp	0.7037037	0.30555555	0.5488215	0.37301585
Lab2	0.9305556	0.8055556	0.74242425	0.71428573
Tutass	0.86666673	0.64166677	0.7030304	0.57857144
Lab4	0.90000004	0.85833335	0.69696975	0.5714286
H1	0.9291666	0.91979164	0.9280304	0.63035715
Mid	0.81111115	0.5388889	0.30989897	0.12609524

Table 3 shows the mean clusters for each of the 4 grades and some patterns are evident upon a closer inspection. The midterm values are significantly lower than the cutoff for each of the grades which indicates that the mid term was particularly difficult for all students across the board. This pattern is in accordance with the technique paper (Nejad, Gedeon 1995). Lab 2 and h1 values are very high for all students indicating that these values are not representative of a student's final grade.

Conclusions and extensions

We have shown that even on a limited dataset, a bi directional neural network can significantly outperform random chance. On top of this, the generated mean clusters can be used to provide feedback to instructors on how representative (of the final grade) certain assessments are and can help them polish the contents/assessments of the course.

Any extensions of this project would require the data gathering process to be more transparent and thorough. This would mean that we can confidently write scripts to predict Null/NaN values in the data and use more data to train and test our models. One way to improve on the model with the limited data we already have is to implement a k-fold cross validation which will maximize the usage of our data.

Another possible extension to the model would be to use fuzzy logic to assign soft boundaries for the grades. This would allow us to generate rules to predict the grades along with confidence levels. This has partially been implemented in (Choi, Gedeon 1995).

References

1. Choi, Edwin Che Yiu, and Tamás Domonkos Gedeon. "Comparison of extracted rules from multiple networks." Proceedings of ICNN'95-International Conference on Neural Networks. Vol. 4. IEEE, 1995.
2. Nejad, Akbar Farhoodi, and Tamás D. Gedeon. "Bidirectional neural networks and class prototypes." Proceedings of ICNN'95-International Conference on Neural Networks. Vol. 3. IEEE, 1995.
3. Jaderberg, Maxwell Elliot, et al. "Population based training of neural networks." U.S. Patent Application No. 16/766,631, 2017.
4. Vikhar, Pradnya A. "Evolutionary algorithms: A critical review and its future prospects." 2016 International conference on global trends in signal processing, information computing and communication (ICGTSPICC). IEEE, 2016.