Comparing Multiple Neural Networks for Mark Prediction

Chan Xu,

College of Engineering and Computer Science
Australian Nation University
Canberra ACT 2601, Australia
u7076780@anu.edu.au

Abstract. Students' grades across a semester of a course, like quiz grades, lab grades, assignment grades mid-term grades, etc., could be informative indicators to their final marks, which might be helpful for instructors to estimate the final grade distribution before finalizing the final exam. It might also be helpful for students to realize their current situation before starting to review for the final. Previous research has used neural networks to achieve final mark prediction with decent accuracy. In this work, we improved the neural network from various aspects including introducing a four-layer neural network and pruning it to try to maintain accuracy and make it faster. We also introduced a deep neural network to test the effectiveness of dropout. We compared the results of these networks, and we found both our improved network and baseline network have significantly higher accuracy than the original work, and the pruned network maintained the same accuracy level. The deep network with dropout performed better than the one without dropout.

Keywords: neural network, classification, mark prediction, pruning, deep learning, dropout

1 Introduction

Neural networks, usually trained by back-propagation, have now become a useful tool to do classification and prediction. The advance in computing technology has enabled us to train such neural networks without powerful hardware. The boom in data availability and storage also provides us with opportunities to explore and extract useful information from the data.

One of the interesting applications of neural networks performed by previous researchers is to predict the final marks of a course based on students' performance across the whole semester, which takes up 40% of the final mark [1]. By knowing the projected final marks before the final exam, instructors can make adjustments to the final exam, and students can think about how they should arrange their review plan to achieve their aimed marks.

In the original research, the researchers used a three-layer neural network with the Sigmoid activation function. We based on the original work and tried to improve the neural network from several aspects. Then, we pruned our neural network based on neuron behavior [2] to make it less computationally expensive. Finally, we implemented a deep neural network to see if dropout has meaningful improvement on a more complex neural network.

2 Method

The dataset consists of 16 columns with 153 entries. Regno is the student identifier, which is irrelevant to our task, so we deleted it from the dataset. Crse/Prog, S, ES, and Tutgroup are all categorical data about the course and enrollment, which might indirectly affect students' grades. For example, one specific tutor group might be more helpful than the others that can help students achieve higher marks, or the assignment in one semester is harder than the other.

The task is to use the students' information of Crse/Prog, S, ES, Tutgroup and the marks of lab2, tutass, lab4, h1, h2, lab7, p1, f1, mid, lab10 to predict their final marks.

Table 1. The original dataset of students' mark information.

Regno	Crse/Prog	S	ES	Tutgroup	lab2	tutass	lab4	h1	h2	lab7	p1	f1	mid	lab10	final
168826	3971	2	FS	T1-yh	2	3	2.5	19.5	.	2.5	11	.	33	2.5	71
168883	3971	2	F	T9-ko	3	5	2	20	17	.	8	.	27	2.4	67
168907	3400	1	F	T6-no	3	3	3	10	.	2	.	.	9.5	2.4	30
169379	3970/1061	2	F	T3-ku	2	3	2	20	19.5	2	15.5	.	21	.	62
169717	3970/1061	2	F	T10-yh	2	3.5	1.5	19	15.5	2	17.5	.	13	2.5	58
170092	3971	2	FS	T1-yh	3	3.5	2.5	20	16	3	16	11	22	2.5	68
173607	3971	2	F	T4-ko	3	5	2.5	17	16	3	18	15	34	2.4	75
174270	3970/6806	2	F	T8-no	2.5	3	2	18	18.5	2.5	9.5	15.5	17	2.4	62
174711	3970/1000	1	F	T8-no	2.5	2	2	11.5	.	.	13	.	11	.	50
……															
8126809	3400	1	F	?	.	.	.	.	.	.	.	.	.	.	.
8130182	3420	3	F	?	.	.	.	.	.	.	.	.	.	.	.
8130678	3420	1	F	T2-no	3	2	2.5	18.5	18.5	0	13	15	8.2	2.4	32

2.1 Data Pre-processing

Before loading the data into our neural network, we pre-processed the data with the following methods.

We dealt with missing values first. The numerical missing values are represented as "." in the data, which indicates no submission or the absence from exams. Therefore, we decided to replace all numerical missing values with 0, as students would receive 0 marks under these circumstances. The categorical missing values are represented as "?", which shows the lack of record of course information. We decided to make missing values a new category.

Second, we transformed all categorical data into numerical data from 0 to N (the number of distinct values within that category).

The final mark has four categories. A mark of 75 or greater, categorized as Distinction, is replaced by 0; a mark between 65 and 74, categorized as Credit, is replaced by 1; a mark between 50 and 64, categorized as Pass, is replaced by 2; a mark below 50, categorized as Fail, is replaced by 3.

We normalized every input variable after loading the data. Finally, we randomly take out 20% of the dataset as the validation set to evaluate our neural networks.

2.2 Neural Network Topology

The original paper's topology is a three-layer neural network with the logistic function as the activation function. There are 14 features, a single hidden layer with 5 neurons, and 4 output units that represent the four categories of the final mark. We reconstructed this neural network as our baseline.

$$y = \frac{1}{1+e^{-x}} \qquad \text{Sigmoid (1)}$$

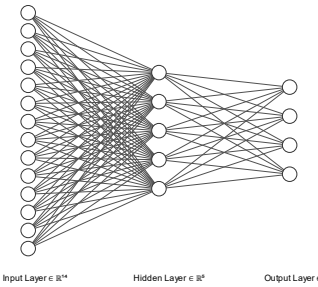

Fig. 1. A Three-layered Neural Network with a Sigmoid Activation Function

Then, we improved the neuron network by introducing another hidden layer and changing the activation function to the most recent ReLU to prevent vanishing gradients, allowing our network to learn faster and perform better. As a rule of thumb [5], we decided the number of hidden neurons to be between input size (14) and output size (4). Therefore, the first hidden layer has 10 neurons, and the second hidden layer has 8 neurons. We used the 5-Fold cross-validation during the training and tuning process. The loss function and the optimization algorithm are not mentioned in the original paper, and we decided to use the cross-entropy loss as our criteria and the Adam optimization algorithm to train both neural networks because they are the standards for classification tasks like this.

$$y = \max(0, x) \qquad \text{ReLU (2)}$$

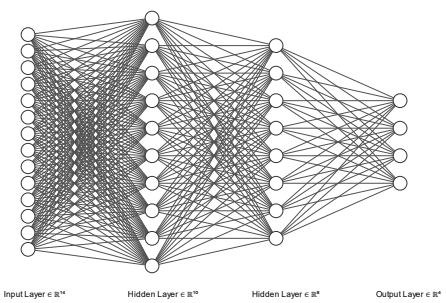

Fig. 2. A Four-layered Neural Network with a ReLU Activation Function

2.3 Cosine-similarity-based Pruning

Finally, to have a better and faster model, we implemented cosine-similarity-based pruning to reduce the number of hidden neurons while trying to maintain accuracy. Cosine-similarity-based pruning (distinctiveness) prunes neurons based on the cosine similarity of the activation vectors of neurons within the same layer. If their activation vectors have an angle smaller than a threshold (usually 15 degrees, also in our case) [2], which means these two neurons are almost doing the same thing, we added the weights from one neuron to the other and deleted the first one. When the angle is greater than a threshold (165 degrees in this case), we deleted both neurons as they do the opposite thing in a layer. In this way, we would reduce the size of the neural network without sacrificing too much accuracy, theoretically.

We pruned the neural network after each sample run and retrain the pruned network with the training/testing set. We then compared the accuracy of the pre-pruned neural network and the pruned neural network on the validation dataset.

2.4 Dropout

Dropout is a proven effective technique to avoid the problem of overfitting by randomly deactivating a certain proportion of neurons in training [3]. It can simulate the process of assembling neural networks with different architectures, and in this way, each neuron has a chance to learn a significant amount of information.

To test the effect of dropout in reducing overfitting and generalization error on a more complex neural network, we also implemented a deep neural network with an extra layer of 6 hidden neurons. We compared the results from this deep neural network with and without dropout to the three hidden layers of a probability of 0.5.

3 Results and Discussion

3.1 Comparing Results of Different Topology and Methods

After we trained our neural networks to get good accuracy and stability in both training and testing, we recorded 10 sample runs. The result is illustrated in Figures 3 and 4.

Our baseline three-layer network recorded an average of 90.02% training accuracy and an average of 68.00% testing accuracy, which is close to the maximum we could get before overfitting starts to kick in. The number of epochs is 2700, and the learning rate is 0.001.

In our pre-pruned four-layer network, we got an average of 90.59% training accuracy and an average of 73.65% testing accuracy. The number of epochs is 800, and the learning rate is 0.001.

The five-layer deep network with dropout had an average of 80.72% training accuracy and an average of 67.35% testing accuracy. The five-layer deep network without dropout had an average of 73.27% training accuracy and an average of 61.76% testing accuracy, which are significantly lower. The number of epochs is 3000 for both, and the learning rate is 0.01.

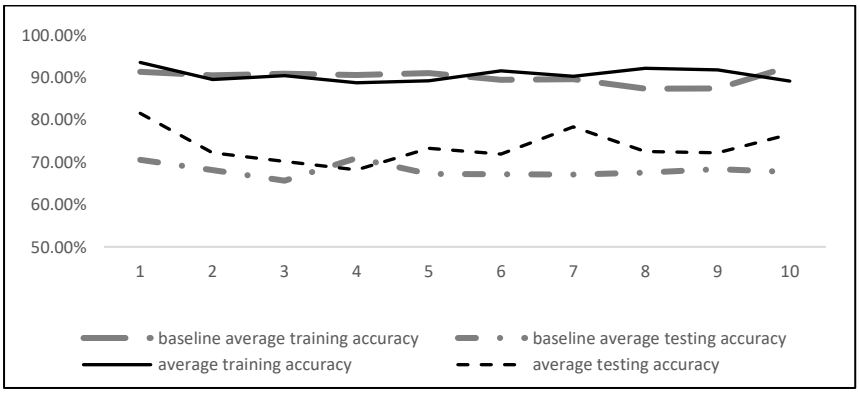

Fig. 3. The Average Training Accuracy and Average Testing Accuracy of Baseline Model and Improved Model

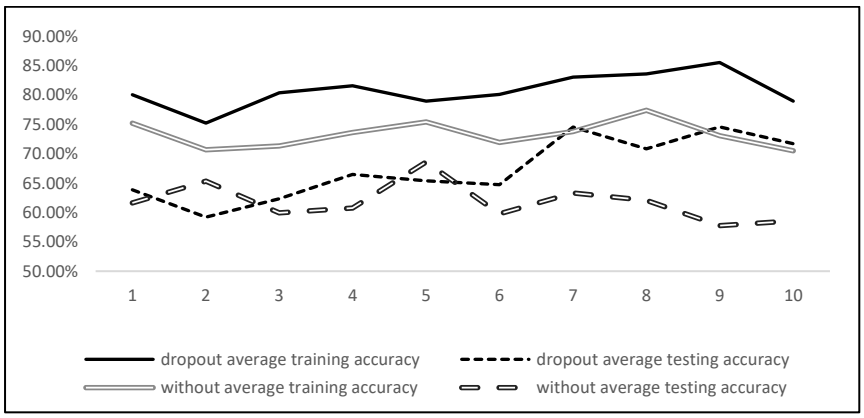

Fig. 4. The Average Training Accuracy and Average Testing Accuracy of Deep Network with/without Dropout

After each sample run, we tested these neural networks including the pruned neural network with the validation set. The result is shown in Figure 4. The average validation accuracy of the baseline network across all 10 runs is 71.03%, the average validation accuracy of the improved four-layer network is 77.45%, 74.85% for the pruned four-layer network (the number of hidden neurons reduced from 18 to around 10), and 69.61% for the five-layer deep network with dropout and 62.26% without dropout.

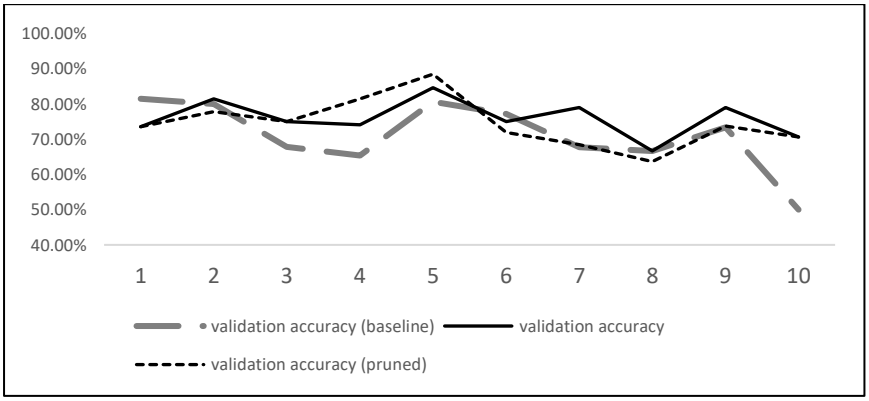

Fig. 5. The Validation Accuracy and Validation Accuracy of Baseline Model and Improved Model

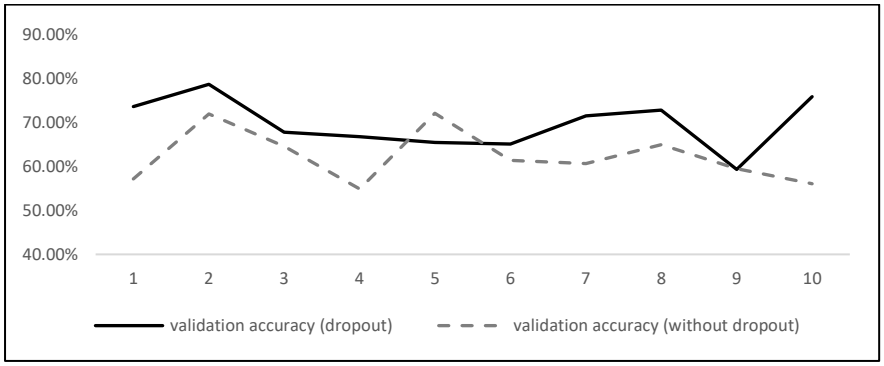

Fig. 6. The Validation Accuracy and Validation Accuracy of Deep Network with/without Dropout

We can tell from the results that the four-layer neuron network, as well as the pruned network, have the best performance overall. The baseline network follows closely, which is surprising because the same topology was used in the original work, but we managed to get significantly higher accuracy. The deep neural network with dropout performs better than the one without dropout, based on a simple t-test ($p<.01$), which indicates the effect of dropout is preventing the network from overfitting and helping its generalization.

Table 2. Comparing results from this work to the original ones.

	Original Work		This Work				
	Training Acc	Testing Acc	Val Acc (Baseline)	Val Acc	Val Acc (Pruned)	Val Acc (Dropout)	Val Acc (Without)
Run 1	83.0%	62.3%	81.48%	73.53%	73.53%	73.53%	57.14%
Run 2	87.0%	62.3%	80.00%	81.48%	77.78%	78.57%	71.88%
Run 3	66.0%	43.3%	67.86%	75.00%	75.00%	67.74%	64.52%
Run 4	90.0%	62.3%	65.38%	74.07%	81.48%	66.67%	54.84%
Run 5	80.0%	60.4%	80.65%	84.62%	88.46%	65.38%	72.00%
Run 6	74.0%	41.5%	77.14%	75.00%	71.88%	65.00%	61.29%
Run 7	66.0%	56.6%	67.74%	78.95%	68.42%	71.43%	60.61%
Run 8	67.0%	43.4%	66.67%	82.35%	63.64%	72.73%	64.86%
Run 9	67.0%	54.7%	73.33%	78.95%	73.68%	59.26%	59.46%
Run 10	73.0%	50.9%	50.00%	70.59%	70.59%	75.76%	56.00%
Average /STD	75.3%	53.8%	71.03% /9.7%	77.45% /5.3%	74.45% /6.9%	69.61% /5.8%	62.26% /6.1%

However, the accuracy may not tell the whole story. If we look at a confusion matrix (Table 3) from one of our neural networks (a confusion matrix of the validation set from the pruned network in this case), we can find something more interesting and meaningful.

Table 3. Confusion matrix of the validation set from the pruned neural network.

	Predicted Distinction	Predicted Credit	Predicted Pass	Predicted Fail	Total
Distinction	3	4	0	0	7
Credit	0	7	2	0	9
Pass	0	4	7	1	8
Fail	0	0	1	4	5
Total	3	11	10	5	29

In this particular run (run 8 of the pruned neural network), the accuracy of our model on the validation set is only 63.63%, which is the lowest of all runs, but this confusion matrix shows that even if our neural network makes a wrong prediction, it is still relevant and not far off the real target. In many other scenarios, a false prediction is not informative or helpful, whether it is far from the true value. However, in terms of mark prediction, a projected final mark can greatly help students prepare for the final exam even if it does not precisely reflect the ultimate result. In addition, when the model predicts a student will fail, he/she is still somewhat likely to pass the exam if he/she studies hard but almost impossible to get Credit or even Distinction, which is very realistic. Another example would be a predicted Distinction might get Credit at last because of potential bad performance in the final exam, but it is unlikely for a student with projected Distinction to just pass or even fail the course only because he/she does not do well in the final exam. Therefore, even when the model has relatively low accuracy, it can still be relevant and helpful in the real world.

3.2 Discussion

The results of all the neural networks, including our baseline model, are better than the original paper's 53.8% average test accuracy and better than their network that is "carefully tuned to maximize the performance over the test set patterns (66.0% test accuracy)" [1], which could be partially contributed to our more advanced network topology and techniques.

Even after pruning, the retrained neural network maintained high accuracy, which proves the pruning technique is effective, and the neural network became more efficient. When we introduced a deep neural network with three hidden layers, a regularization technique, dropout, helped the complex network be trained better and prevented overfitting, the overall accuracy is higher than the one without the dropout technique.

4 Conclusion and Future Work

We have proved that pruning can be useful for reducing the neural network size, which saves computational power while maintaining accuracy. In addition, when the network gets more complex, dropout is an effective way to prevent overfitting

and improve generalization. However, our neural network is still relatively small. Even our most complex five-layer deep network has only a few neurons within each hidden layer. It remains to be tested what effect would pruning have for even larger neural networks. It should also be an interesting topic to explore whether designing a complex network and pruning it afterward or starting with a topology as simple as possible would have a better result.

Besides, we did not further prune the network during retraining to see how far we can continue to prune a network until the accuracy significantly degrades. Theoretically, we can use this technique to find the minimum hidden neurons that satisfy a user-defined threshold, which can be a direction of future work.

Also, we pruned the network after training and retrained the pruned network because retraining can alleviate the drop in accuracy after pruning [6]. However, the downside is retraining is computationally expensive, it may take a long time for a very large neural network. Alternatively, we can prune the network during the training process so that we do not have to retrain it. Future research could compare retraining vs. no retraining under different circumstances and their effectiveness and efficiency.

Finally, different techniques of pruning could be tested to see whether a specific task could benefit the most from certain pruning techniques.

In terms of the dropout technique, one can explore the relationship between the effectiveness of dropout and the complexity of the neural network to see if more complex networks can benefit more from dropout.

Overall, other advanced techniques like evolutionary algorithms could be applied to select important features and fine-tune the hyperparameters of the neural network.

References

1. Choi, E. C. Y., & Gedeon, T. D. Comparison of extracted rules from multiple networks. In Proceedings of ICNN'95-International Conference on Neural Networks (Vol. 4, pp. 1812-1815). IEEE, 1995.
2. T. D. Gedeon, "Indicators of hidden neuron functionality: the weight matrix versus neuron behaviour," Proceedings 1995 Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems, Dunedin, New Zealand, 1995, pp. 26-29, doi: 10.1109/ANNES.1995.499431.
3. Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." The journal of machine learning research 15.1 (2014): 1929-1958.
4. T. D. Gedeon and D. Harris, "Network Reduction Techniques, " Proc. Int. Conf. on Neural Networks Methodologies and Applications, San Diego, vol. 2, pp. 25-34, 1991.
5. Heaton, J. Introduction to neural networks with Java. Heaton Research, Inc, 2008.
6. M. Pietron and M. Wielgosz, "Retrain or Not Retrain? - Efficient Pruning Methods of Deep CNN Networks," In: Krzhizhanovskaya V.V. et al. (eds) Computational Science – ICCS 2020. ICCS 2020. Lecture Notes in Computer Science, vol 12139. Springer, Cham. https://doi.org/10.1007/978-3-030-50420-5_34, 2020