Application of Recurrent Neural Networks in Mark Prediction

Zhipeng Li
Research School of Computer Science
Australian National University
u6766505@anu.edu.au

Abstract. Artificial Neural Networks (ANNs) have attracted the attention of a large number of researchers and engineers in recent years. Because of the exceptional modeling capability and the application potential shown by ANNs, ANNs have replaced many traditional machine learning models and been applied to solve various nonlinear regression and classification problems. In this work, we propose a proof of concept combining multi-class classification recurrent neural network (RNN), which is a class of ANNs, and bimodal distribution removal (BDR) in the prediction of student marks. By applying the RNN model developed in this work, we could predict students' final grades based on the marks of their previous assessments. Our experimental results show that, in the problem of mark prediction, RNN could achieve higher prediction accuracy than conventional error back propagation neural networks.

Keywords: artificial neural networks, recurrent neural network, bimodal distribution removal, mark prediction

1. Introduction

ANNs are machine learning models that are inspired by the biological neural networks [1]. Unlike most traditional machine learning models, ANNs are non-parametric models whose implementation does not require the users to make assumptions based on the dataset, which makes ANNs simpler to use [1]. Because of their non-parametric characteristic, ANNs could learn any functional form from the training data, which means that ANNs have stronger modeling capability than traditional ML models and could detect complex nonlinear relationships between variables more implicitly [2]. Due to their excellent performance and ease of use, ANNs are now widely applied in engineering, science, finance, and many other fields. In order to cope with the increasing application demand, various ANN models have also been developed, such as Convolutional Neural Networks (CNNs) with powerful image processing capability [3], Recurrent Neural Networks (RNNs) that could fit sequential data [4], and the particularly popular generative model in recent years, Generative Adversarial Networks (GANs), which could generate high-quality data by learning from the distribution of training set [5]. RNN is a class of ANNs that have connections between nodes along time series, which allows it better exhibit temporal dynamic behavior [4]. Recurrent neural networks have proven to outperform traditional neural networks in areas such as speech recognition, machine translation, and music or text generation [6]. Unlike conventional ANN which could only process fixed-length inputs, RNN could use its internal memory to process input sequences of any length [7]. And there are both internal feedback connections and feedforward connections between the processing units of RNNs, which makes it easier for RNNs to handle time series [7].

The powerful modeling ability of ANNs and their variants also makes them very easy to overfit the training data. In the field of statistics, overfitting is defined as "the production of an analysis that corresponds too closely or exactly to a particular set of data, and may therefore fail to fit additional data or predict future observations reliably" [8]. Overfitting usually occurs when the machine learning model learns the inherent noise in the training set, in another word, the model fits the training set too well so loses generalization. In order to avoid overfitting, usually the following two approaches are applied: 1. Reducing the complexity of the model, such as regularization techniques like pruning [9], cross-validation [10], and early stopping [11]. 2. Reducing the variance of the training set, such as outlier removal techniques like BDR [12]. BDR is an outlier detection approach proposed by P. Slade and T. D. Gedeon in 1993 [12]. The mechanism of BDR to reduce overfitting is based on that the errors of the training data gradually form an almost bimodal distribution during training, with the low error peak formed by data that the model learnt well and high error peak containing the outliers

[12]. Unlike other outlier removal methods such as Least Median Squares (LMS) and Least Trimmed Squares (LTS), the pattern removal process will not start until the model has identified the outliers, and the pattern removal process is data driven [12]. The BDR method also provides a halting criterion naturally evolves preventing overfitting, which could help improve training efficiency [12].

In this work, we compared the performances of RNN and conventional ANNs in the prediction of student marks. In order to improve optimization and generalization, k-fold cross validation and learning rate decay are applied for both models. In addition, we also examined the effectiveness of BDR. Our experiment results showed that RNN could better capture the dependences between students' previous marks and their final grades and achieve higher prediction accuracies than conventional ANNs.

2. Method

2.1 Data Engineering

The dataset used to train the model is composed of 153 students' information and marks which includes results of the students' previous 10 assessments and their final grades, except the marks of the final exam. In this dataset, around 40% of entries in the dataset contain missing values. In order to mitigate the impact of these missing data on the experimental results, we assume that students with missing data did not submit their homework or participate in the related labs or exams. So, in our experiments, all the missing values in the dataset are replaced with 0. Then we standardize and convert the feature values in the dataset into Gaussian distributions by applying z-score normalization [13]. The equation used to do z-score normalization are as follows [13]: $z = \frac{x-\mu}{\sigma}$ where μ is the average value of the feature, and σ is the standard deviation of the feature. The final marks of the students are transformed into integer values between 0 to 3, which indicate the students' letter grades, following the rules as follows: • Grade D, raw marks greater or equal to 75, represented by 0. • Grade CR, raw marks between 65 and 74, represented by 1. • Grade P, raw marks between 50 and 64, represented by 2. • Grade F, raw marks less than 50, represented by 3. After data engineering, the normalized dataset contains 153 entries each having 10 normal distributed features and 1 integer target.

2.2 Configuration of the ANN model

In this work, a classification ANN model is developed to predict students' final letter grade. As shown in Figure 1, the ANN model is composed of three fully connected layers with the hidden layer activated by LeakyReLU activation function [14]. Unlike Rectified Linear Units (ReLU) activation function, LeakyReLU does not have zero-slope parts, so it could fix the "dying ReLU" problem [15]. The input layer contains 10 neurons, each corresponding to one of the marks from the students' previous assessments x_i. The output layer contains 4 neurons, and the output of each neuron indicates the probability that the student will get each grade. The activation function for the output layer is softmax which is an activation function commonly used in multi-class classification problem. In this setting, the outputs of the ANN model are calculated as:

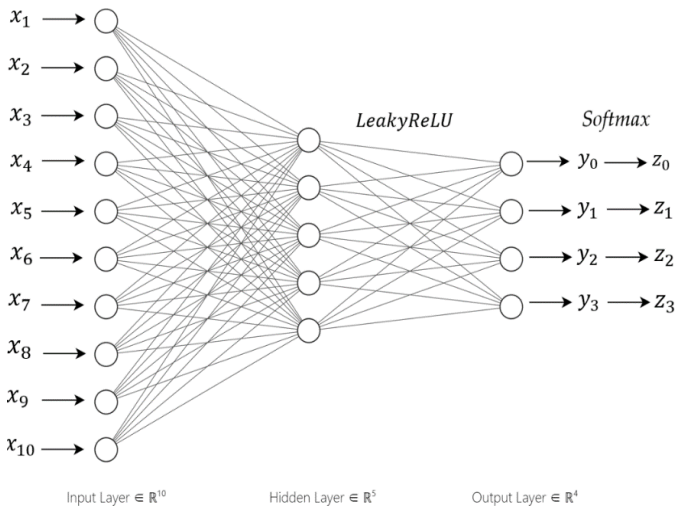

Figure 1. The configuration of the ANN model

$$z_i = \frac{e^{y_i}}{\Sigma_{j=1}^{3} e^{y_j}} \quad \text{for i} = 0, 1, 2, 3$$

where y_i is the value in the i_{th} output neuron before activation [16].

2.3 Configuration of the RNN model

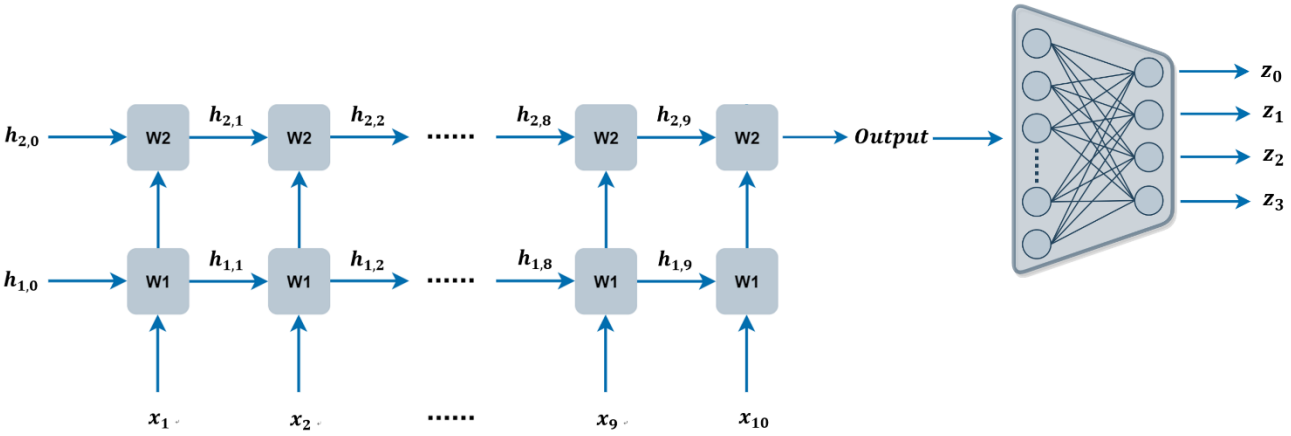

Figure 2. The configuration of the RNN model. It is noted that $\boldsymbol{x_i}$ indicates the i_{th} assessment result of the students, and $\boldsymbol{h_{j,i}}$ indicates the i_{th} hidden state in layer $\boldsymbol{j}$

The RNN model developed herein consists of a many-to-one RNN and an ANN with the same configuration as Figure 1. In the implementation of the RNN model, the students' previous assessment results are considered as a time series. The input of the RNN is a sequence of values $\boldsymbol{x_i}$ where $\boldsymbol{i} = 1 \ldots 10$, which are the 10 previous assessment results of the students. The output of the RNN is a single vector with the same dimension of the input layer of the ANN, which contains all the information memorized by the RNN in the previous time steps. This output vector will then go through the ANN and generate the probabilities of the students getting one of the four letter grades as explain in 2.2. As shown in Figure 2, the RNN has two hidden layers whose parameters are respectively represented as **W1** and **W2** in the figure. Each hidden layer is activated by a ReLU activation function. Unlike ANN using all the input values at once, the RNN model herein only takes one input value at a time step. For example, at time step $\boldsymbol{i}$, the RNN takes in the current input $\boldsymbol{x_i}$ and the two hidden states $\boldsymbol{h_{1,i-1}}$ and $\boldsymbol{h_{2,i-1}}$ from the previous time step.

2.4 Loss function and bimodal distribution removal

During the training process, the model is trained to minimize the cross-entropy loss which measures the difference between the predicted grades and the target grades. The cross-entropy loss function is defined as $-\Sigma_{i=1}^{K} t_i \, log(z_i)$, where t_i is the target label of the training data, and z_i is the model output [17]. After every 50 epochs of training, the loss for each training data then will be used to recognize the outliers in the training set through the BDR process [12]. The BDR process begins once the normalized variance of errors for the training set v_{ts} is below 0.1, which indicates the two error peaks have already formed [12]. The mean error $\overline{\delta_{ts}}$ is then calculated and used to divide the training data into two groups. One group with errors bigger than the mean error $\overline{\delta_{ts}}$, and the other with errors below $\overline{\delta_{ts}}$. Then we take the data from the first group and calculate their standard deviation σ_{ss} and mean $\overline{\delta_{ss}}$. After that, the patterns that have errors $\geq \overline{\delta_{ss}} + \alpha\sigma_{ss}$ will be permanently removed from the training set, where α is a preset parameter. As illustrated in [12], the BDR method also provides a natural stopping criterion to terminate training. In our experiments, we keep monitor the value of v_{ts} after each epoch of training and terminate the training process once $v_{ts} < 0.01$.

2.5 K-fold Cross Validation

Before training, the dataset is randomly shuffled and split into two sets: a training set containing 80% of the data and a test set containing the rest. To mitigate the impact of uneven data distribution on model performance and avoid overfitting, k-fold cross validation is applied [18]. The training set is further split into k groups, each containing about $1/\mathrm{k}$ of the training data. For each epoch of training, k models are trained by holding one group as the validation set and using the remaining groups to train the model. After training, we calculate and record the average prediction accuracies of the models on the training and test set.

2.6 Learning rate decay

In this work, instead of comparing the impact of different values of learning rate on model performance, a variable learning rate approach is applied. As reported by Bowling and Veloso, a variable learning rate could help improve model generalization and optimization [19]. Here we decay the value of the learning rate based on the model's prediction accuracy on the validation set. In our experiments, the model's learning rate is originally set to 0.01, and the learning rate decays by a factor of 0.5. The frequency of learning rate decay is inversely proportional to the growth rate of the model's prediction accuracy. At the beginning of training, the prediction accuracy of the model increases rapidly, and the change in learning rate is relatively small. With the training goes on, as the model gradually approaches optimization, the growth rate of the model's prediction accuracy decreases, which causes the frequency of learning rate decay to increase until the learning rate approaches zero.

Table 1. The average training and test accuracy of the ANN and RNN models.

Models	Learning rate decay	Average training accuracy	Average test accuracy
ANN	-	95.76%	65.24%
ANN	✓	87.63%	69.57%
RNN	-	94.49%	67.19%
RNN	✓	88.91%	73.74%
Results from [2]	-	75.30%	53.80%

3 Results and discussion

In our experiments, to verify the effectiveness of learning rate decay and compare the performances of the RNN and ANN models, each model is trained on the normalized dataset with and without learning rate decay. In total, four models are trained. After training each model 100 times, the average training and testing accuracies are as shown in Table 1. As we can see from the table, The ANN model which does not apply learning rate decay has the lowest test accuracy of 65.24%. And the RNN model with learning rate decay achieves the highest test accuracy of 73.74%. It is noted that both models achieve lower training accuracies and higher test accuracies with learning rate decay, which is a good demonstration of the ability of learning rate decay against overfitting. With the implementation of learning rate decay, the average test accuracy of the RNN model is around 4% higher than the ANN model, while the training accuracy stayed almost the same. This is probably because the students' marks have a chronological order, and RNN has an advantage over ANN in processing time series data [7]. And all the training and test accuracies of our models are higher than the reported values in [20], which could be because our experiment used LeakyReLU and ReLU as the activation functions of the hidden layer, which provides better gradients than the sigmoid functions used by [20].

To see how does learning rate decay help avoid overfitting and what exactly happened during the training process. We plot the training, validation, and test loss of the RNN models against their training epochs in Figure 3. As shown in Figure 3, in the first 20 epochs of training, all the three losses of the two RNN models gradually decreased as the training progressed. As the training goes on, the training loss of the RNN model that does not apply learning rate decay continues to decrease, and its validation and test losses begin to increase and fluctuate after about 30 epochs, showing signs of overfitting. However, for the RNN model that uses learning rate decay, the effect of overfitting is avoided. By continuously reducing the learning rate, the loss of the model on the three data sets gradually converged, and the validation loss and test loss stay near their optimal values. Therefore, we conclude that by using learning rate decay, we could avoid overfitting and improve the prediction accuracy of the model

Figure 3. The training, validation, and test loss of the RNN model.

In order to verify the impact of BDR on the model performance, we also conducted experiments on the ANN and RNN models. It is worth noting that this time both models have applied learning rate decay to prevent overfitting. The experimental results are shown in Table 2, from which we can see that after using BDR, the training accuracy of both models has slightly decreased. This is because after removing outliers, the model is no longer trained on the entire training set, and the accuracy of the model's prediction on these outliers declines resulting in a decline in the overall training accuracy. However, the test accuracy of the models has not significantly improved or decreased, which may be caused by insufficient training data. Given the fact that there are only around 110 training data in the training set, the BDR algorithm could also remove some valuable data while removing the outlier. As a result, the average test accuracies of the models have not improved.

Table 2. The average training and test accuracy of the models with and without BDR.

Models	Learning rate decay	BDR	Average training accuracy	Average test accuracy
ANN	✓	✓	84.72%	70.01%
ANN	✓	-	87.63%	69.57%
RNN	✓	✓	85.49%	73.46%
RNN	✓	-	88.91%	73.74%

Conclusion and Future Work

In our experiments, the BDR method has been shown to have no significant impact on the performance of both the ANN and RNN model. In the prediction of student marks, RNN has better performance than conventional error back propagation ANN, which may be because the students' marks are obtained in chronological order. Both the ANN model and the RNN model developed in this work have higher training and test accuracies than the results reported in [20]. And by applying techniques like k-fold cross-validation and learning rate decay, the learning rate of the model can adjust according to the model's performance on the validation set, thereby helping the model avoid overfitting. Since the marks of students are distributed as sequences, in the future work, techniques such as bidirectional RNN and LSTM could be applied to further improve the model's prediction accuracy. And since the data set is slightly unbalanced, techniques like under-sampling and over-sampling can be used to further improve model's generalization.

Reference:

1. 1. Y.-Y. Chen, Y.-H. Lin, C.-C. Kung, M.-H. Chung, and I.-H. Yen, “Design and Implementation of Cloud Analytics-Assisted Smart Power Meters Considering Advanced Artificial Intelligence as Edge Analytics in Demand-Side Management for Smart Homes,” Sensors (Basel, Switzerland), 02-May-2019.

2. J. V. Tu, “Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes,” Journal of Clinical Epidemiology, 22-Mar-1999.

3. A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” ImageNet Classification with Deep Convolutional Neural Networks, 01-Jan-1970.

4. Mikolov, Tomáš / Karafiát, Martin / Burget, Lukáš / Černocký, Jan / Khudanpur, Sanjeev (2010): "Recurrent neural network based language model", In INTERSPEECH-2010, 1045-1048.

5. I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative Adversarial Nets,” Generative Adversarial Nets, 01-Jan-1970.

6. Chandra, R., & Zhang, M. (2012). Cooperative coevolution of elman recurrent neural networks for chaotic time series prediction. Neurocomputing, 86, 116-123. doi:10.1016/j.neucom.2012.01.014

7. Alex, “Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network,” arXiv.org, 31-Mar-2020. [Online]. Available: https://arxiv.org/abs/1808.03314.

8. Definition of "overfitting" at OxfordDictionaries.com: this definition is specifically for statistics.

9. 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.

10. Breiman, Friedman, Olshen, Breiman, Geisser, S.k., Kwok, Carter, J.l., McClelland, Rumelhart, Quinlan, Schaffer, J.w., Shavlik, Mooney, Towell, and Wolpert, “Selecting a classification method by cross-validation,” Machine Learning, 01-Jan-1984.

11. L. Prechelt, “Early Stopping – But When?”, SpringerLink, 01-Jan-1970.

12. P. Slade and T. D. Gedeon, “Bimodal Distribution Removal,” International Workshop on Artificial Neural Networks, pp. 249-254, 1993.

13. Patro, S. G. Krishna, Sahu, and K. Kumar, “Normalization: A Preprocessing Stage,” arXiv.org, 19-Mar-2015.

14. Xu, Bing, Wang, Naiyan, Chen, Li, and Mu, “Empirical Evaluation of Rectified Activations in Convolutional Network,” arXiv.org, 27-Nov-2015.

15. Lu, Shin, and G. Em, “Dying ReLU and Initialization: Theory and Numerical Examples,” arXiv.org, 12-Nov-2019.

16. C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, “Activation Functions: Comparison of trends in Practice and Research for Deep Learning,” Nov-2018.

17. S. Bruch, X. Wang, M. Bendersky, and M. Najork, “An Analysis of the Softmax Cross Entropy Loss for Learning-to-Rank with Binary Relevance,” Google Research, 01-Jan-1970.

18. Fushiki, T. Estimation of prediction error by using K-fold cross-validation. Stat Comput 21, 137–146 (2011). https://doi.org/10.1007/s11222-009-9153-8

19. Bowling, M., & Veloso, M. (2002). Multiagent learning using a variable learning rate. Artificial Intelligence, 136(2), 215-250. doi:10.1016/S0004-3702(02)00121-2

20. E. C. Y. CHOI and T. D. GEDEON, “Comparison of Extracted Rules from Multiple Networks,” University of New South Wales, 1995.