Reduction Techniques for Two-layer Neural Networks and Recurrent Neural Networks based on Similarity of Hidden Units

Weiwen Huang

Research School of Computer Science, Australian National University
U6726234@anu.edu.au

Abstract. Training feed-forward neural network of a few hidden units by back-propagation can be very time-consuming. The number of hidden units is difficult to decide so people tend to define excessive hidden units to complete a neural network. To pruning the network, some reduction techniques concentrate on removing these kind of units to simplify the structure, which is able to make a better estimate of the minimal size of hidden unit so as to decrease computing time and increase efficiency of the network. Distinctiveness by vector angles among hidden units provides the evaluation to identify those similar units and remove them to reduce the heavy parameters in neural network. The result of using this method shows that the reduced network produces similar performance without further retraining.

Keywords: Artificial neural network, recurrent neural network, distinctiveness, vector angle, hidden units

1 Introduction

An artificial neural network (ANN) is a computational paradigm based on biological nervous system. It is an interconnected group of natural artificial neurons that uses mathematical or computational model for information processing [1]. During the process, artificial neural network infers the internal pattern from input data and then this result can be applied into other practice. Each neuron in the network extracts specific features based on the content it receives and the position in the network. In most cases, ANN have a good performance on function approximation, regression analysis and classification when it is feed-forward and get trained by back-propagation algorithm [2]. Consequently, it becomes popular among relevant workers.

A recurrent neural network (RNN) is one class of artificial neural network which can model sequences of arbitrary length as inputs or outputs. Basically, anything that can be modelled as a sequence through time can be modelled by an RNN. A RNN model also has hidden layers and the process of feedforward computation and backward computation. Moreover, its recurrent connection of a unit from its output back to input can cycle through the same activation and weight connections for each time step of the sequence.

Although ANN has been widely used in solving a variety of problems, many people have realized that training such a network model at all or in a selected time scales is highly time-consuming and significantly relies on the computing ability of the hardware. Hidden units included in the training process are more than those appear to be required, which is exactly the case for many hand-crafted networks for some simple problems. Actually, these excess hidden units make little or no contributions to the final outcomes in the neural network [3]. The computation process of RNN is time-consuming and space-consuming, and a proper size of hidden layer is necessary for pruning the network and improve efficiency and accuracy [4]. A reduced RNN could have a better generalization performance on a completely new testing data set [5]. Consequently, these unnecessary hidden units can be removed with particular conditions to improve the utilization of the network.

2 Methodologies

2.1 Data processing

I used a dataset from paper [6] to build a two-layer neural network, which provides totally 192 patterns from 24 participants with 14 features and 8 groups of classes. All 14 features were extracted values from the physiological signals collected using devices with sensors. The signals were segmented according to the song length to get 8 samples from each participant. For the first group of class, it consists of 3 categories: classical, instrumental and modern pop music. Each categories contains 4 different songs. The other groups are feeling scales from the participants after they listen to the music. In this paper, I only used genres and one of subjective rating results (Depressing -> Neural -> Exciting) as output classes for two-layer neural networks.

To train a RNN model, I used time-series sequences as a dataset. This dataset is grouped by song numbers. Any 16 of 24 participants listened to one of the 12 song and some signals sequences could be received in this process. Each song was about 4-minute length and the sampling rate is 4 Hz so for each second there are 4 signal points and each sequence has a length of approximate 1000 data points. The total number of sequences is 16 * 12 = 192 and 75% of them are training data and the left are testing data. The output classes are the categories of the related songs.

To reduce the impacts from subjective movements in the first dataset, I used Min-Max normalization techniques to shift each feature into the range of minimum 0 to maximum 1 [7]. The equation for min-max normalization is:

$$value\_new = (value - value\_min)/(value\_\mathrm{max} - value\_\mathrm{min})\ . \qquad (1)$$

Where *value_new* corresponds the min-max normalized data in the range of 0 to 1, *value* is the raw data and *value_max* and *value_min* are the maximum and minimum value respectively of values through all patterns.

The normalized data then will be divided into 2 groups. 75% are training data and 25% are testing data, both of which has the same distribution of each class, so the variance of training model can be avoided and the network can receive stable outcomes in the testing data.

2.2 Implementation of a two-layer neural network

I generate a two-layer feed-forward neural network (hidden layer and output layer) of processing units. All units are connected from one layer to the subsequent one without lateral, backward or multilayer connections. The units in hidden layer receive the values from input layer and then their outcomes become the input for the output layer. In this paper, the fully-connected neural network is designed with 11 units in input layer (number of selected features by genetic algorithm), 30 units in hidden layer (as mentioned the best size of hidden layer) and 3 units in output layer (3 categories for each group of classes). The connection structure is shown in Fig. 1:

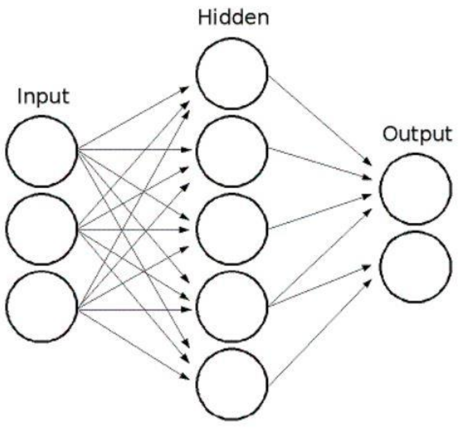

Fig. 1. Structure of a two-layer neural network

In the network, hidden units use Sigmoid activation function to implement nonlinear operation. The output layer uses softmax function to predict the classification results. This model also uses mean squared error (MSE) loss as loss function and Adam for optimizer. The model is trained with learning rate of 0.01 and epoch of 1000.

2.3 Implementation of a recurrent neural network

Basic recurrent neural networks are connected nodes into successive layers with a directed connection to every other node in the next time step layer. At each time step, each neuron has an activation function in hidden layer to achieve the property of nonlinearity. A proportion of information from the previous time step can be passed to the later time step within hidden layers. For classification tasks, many-to-one RNN model is widely used in practice. Supervisor-given target will be supplied in the last output as a final outcome of the network.

In this paper, hidden layers contains 50 units and use tanh activation function to achieve nonlinearity. The output layer is a vector with size of 3 * 1 for each pattern and uses log softmax function to predict the classification results. This model chooses cross entropy to calculate the loss of the network outcomes and Adam as an optimizer. In model training process, learning rate is 0.003 and epoch is 200.

The basic structure of many-to-one RNN model is shown in Fig. 2:

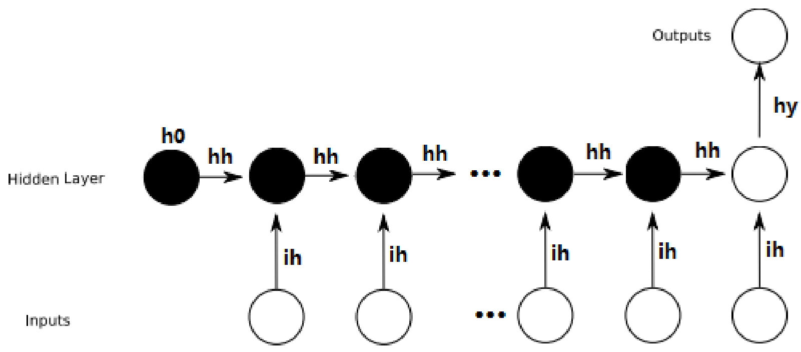

Fig. 2. Structure of a many-to-one recurrent neural network

2.4 Evaluation methods

In classification tasks, it is important to use suitable evaluation measures to evaluate the result of the network. Classification accuracy is the most common measure. What's more, I have used precision (fraction of the predicted labels matched), recall (fraction of the reference labels matched, also called sensitivity) and F1 score (harmonic mean of precision and recall, also called F score) to validate the classifier. Some needed definition is shown in Table 1:

Table 1. Definition for TP, TN, FP, FN

	Class=1 (Predicted)	Class = 0 (Predicted)
Class=1 (Real)	True Positive (TP)	False Negative (FN)
Class=0 (Real)	False Positive (FP)	True Negative (TN)

To calculate F1 score, we have to know the four parameters in Table 1:

- True Positive (TP): The positive number that is predicted correctly.
- True Negative (TN): The negative number that is predicted correctly.
- False Positive (FP): The positive number that is predicted falsely.
- False Negative (FN): The negative number that predicted falsely.

Precision is the ratio of correctly predicted positive data to the total predicted positive data, and recall is the ratio of correctly predicted positive data to the actual positive data. Then the F1 score keep a balance of precision and recall to show the exactness and completeness of the classifier. The formulas are described as followed:

$$precision = TP\,/\,(TP + FP)\,. \tag{2}$$

$$recall = TP\,/\,(TP + FN)\,. \tag{3}$$

$$F1 = (2 * precision * recall)\,/\,(prcision + recall)\,. \tag{4}$$

Since there is a multi-classification task, the precision, recall and F1 score for each class should be calculated into average values to evaluate the whole performance for the network over this dataset. Accuracy is the ratio of correctly prediction to the total prediction. Its formula is described as followed:

$$Accuracy = (TP + TN)\,/\,(TP + FP + FN + TN)\,. \tag{5}$$

Space complexity and time complexity are both essential factors that should be considered in designing a neural network. Space complexity basically means the use of the memory in computation. A neural network contains a large quantity of parameter of weights and bias. Most of the network need to save temporary values for backward propagation or performance evaluation, so a more complex model requires more memory source to support its computation to achieve expected results. Time complexity mainly indicates with execution time of major function programs. The number of calculation parameters and the complexity of calculation methods determine the time spent in a model.

For classification tasks, the most time-consuming process is either training and testing computations, but a reduced network will not retrain in this work, so to evaluate the time complexity and space complexity of the reduced network, the execution time of testing process for each reduce network will be noted because of the lower number of hidden units in the neural network. Since whole structure of the neural network had been pruned, the memory used by the heavy parameters is bound to decrease and it doesn't need to observe.

2.5 Improvement with pruning techniques

In 1991, Gedeon and Harris [8] propose that the hidden unit output activation vectors represent their functionality over input pattern and determine the distinctiveness of hidden units. These units are recognized to perform insignificantly in the network and one of them can be removed without reducing the performance of the model. The way to identify the similarity of vectors is to calculate vector angle in pairs over input space. The formula on calculating the vector angle is:

$$angle(u, v) = \arccos(u \cdot v \,/\, (\parallel u \parallel \cdot \parallel v \parallel)) \ . \tag{6}$$

The angular separations less than 15 °can be seen effectively similar and one of them is removed. Then the weight of the removed unit should be added to the unit which remains. The angular separations greater than 165 °can be viewed contradictive and both of them are removed. Since Sigmoid function output values are constrained in the range 0 to 1, they are normalized to the range -0.5 to 0.5 so that the angular separation ranges from 0 °to 180 °rather than 0 °to 90 °. Tanh function outputs a range of -1 to 1, so it doesn't need any modification.

The network parameters were extracted from the model and the result of the angles between each pair of hidden unit vector is stored in a vector matrix, noting the indices of both unit vectors if their vector angle shows similarity of contradiction between these vectors. It can be seen that the result of vector angle matrix is symmetric matrix because the calculation of vector angle is commutative and the vector angle with a vector itself is 0 °.

To maintain the stability and normal performance of the neural network, I didn't remove too much units in the hidden layer and only considered the angles less than 10 °and greater than 170 °. All the angles satisfied with these conditions are sorted in descending order and operate the removals for the most contradictive vectors. If there is no angle greater than 170 °, then sort the angles in ascending order and remove the most similar vector. Either for similar vectors or contradictive vectors, those with unique association will be operate as priority. After each removal, the reduced model will be evaluated and recorded for analysis. This new neural network then repeats the vector angles analysis and reduction process, until there is no similarity or contradiction can be found in the model or the reduction work has reached an expected extend.

When there were more than two units similar with one unit in the network, I removed each hidden unit to build a new reduced neural network individually. Assumed that one of the hidden units show similarity with another n-1 units, the pruning step got n new model and I compared the performance of each network. The best one was selected as the new one and continued the next pruning.

When finishing building a new neural network, use the same testing data as input and expected outcomes to evaluate the model. Theoretically the value of accuracy, precision and F1 score are maintained in a small range of oscillation and the computation time is accordingly reduced.

3 Results and discussion

3.1 Electric signals based classification with Two-layer neural network

The results and testing time for two-layer neural network on electric signals are shown in Fig. 3 and Fig. 4:

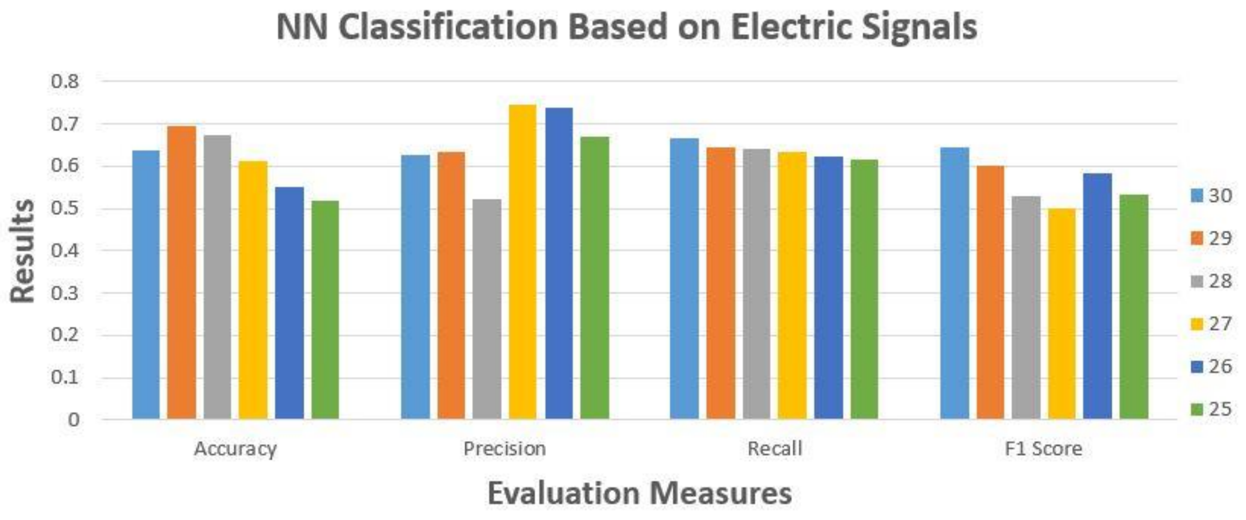

Fig. 3. Two-layer NN Classification Based on Electric Signals

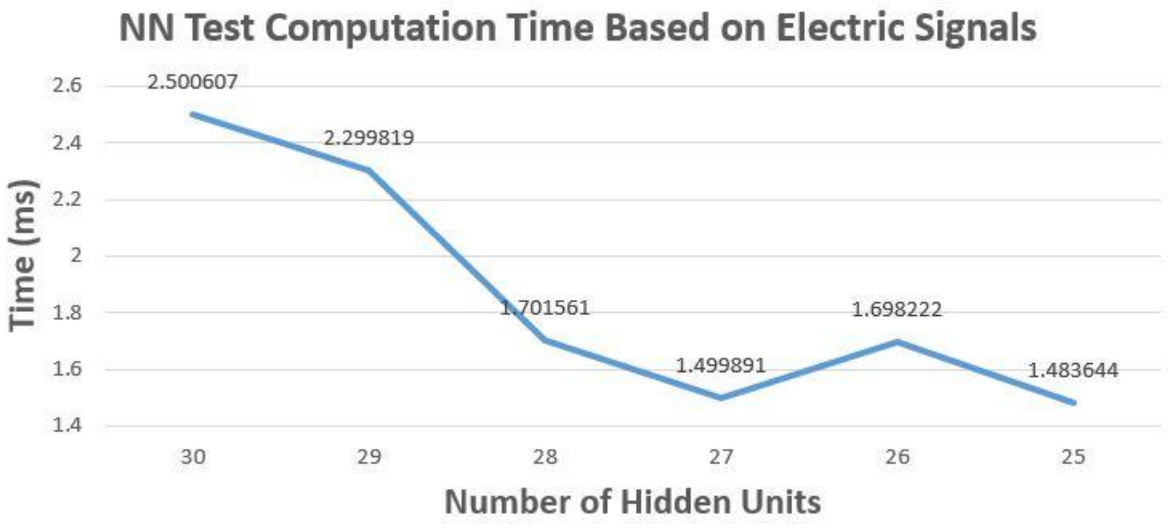

Fig. 4. Two-layer NN Test Computation Time Based on Electric Signals

The Fig. 3 shows the classification accuracy and the results of three evaluation measures, which are the average of precision, recall and F1 scores among 3 music genre categories using the 11 features selected by generic algorithm. It indicates the comparisons among the original two-layer neural network and the reduced networks based on the genre. The quantity of hidden units decreased from 30 to 26, and the evaluation results are generally similar for each neural network. The recall kept a steady level during the reduction, but the accuracy and F1 score decreased about 10%. The precision of the network rose again when it lowered with 28 hidden units.

In the Fig. 4, the time consumptions in testing step with the same testing data in each neural network were listed in line chart. The original neural network spent about 2.5 ms to complete testing computation, while the other reduced network all cost less time in testing. Fewer hidden units consumed less time, though the model with 26 hidden units spent a little bit more time than the previous one. The figure showed the trend of decreasing time consumption for the reduced neural networks.

3.2 Subjective rating based classification with Two-layer neural network

The classification results and testing time consumption for two-layer neural network on subjective rating (Depressing -> Neural -> Exciting) are shown in Fig. 5 and Fig. 6:

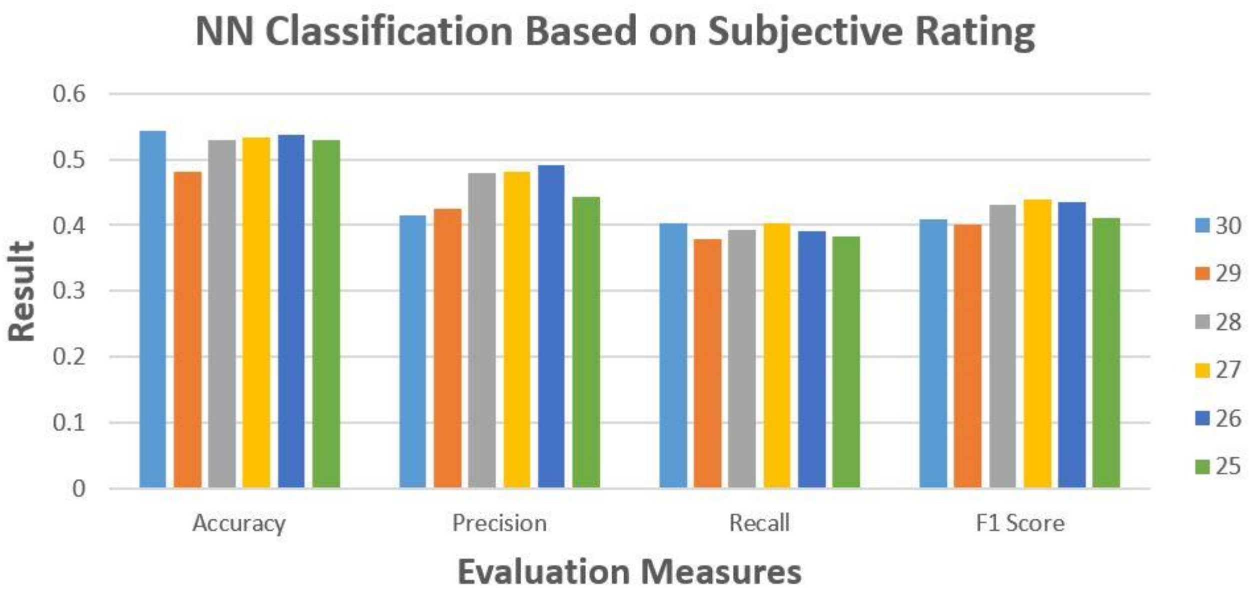

Fig. 5. Two-layer NN Classification based on subjective rating (Depressing -> Neural -> Exciting)

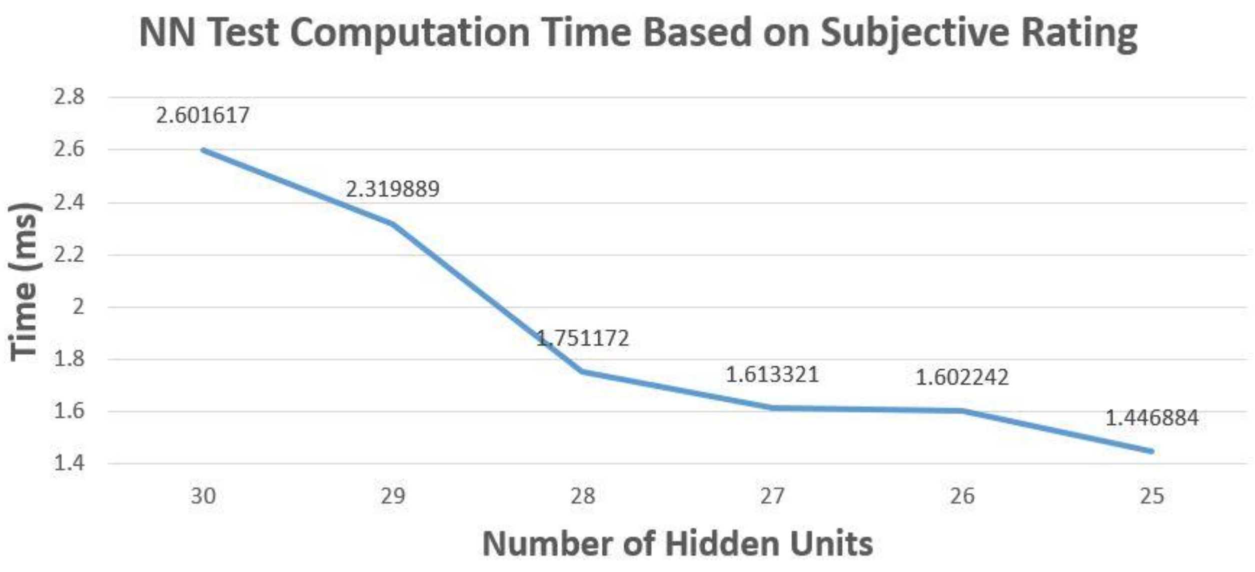

Fig. 6. Two-layer NN Test Computation Time Based on subjective rating (Depressing -> Neural -> Exciting)

As seen in Fig. 5, most new networks gave better measures than those of the original network. All new networks indicated a better precision than the original network and they gave similar results on recall and F1 score.

One situation should be pointed out that some reduced networks seemed to have good performance. According to the results above, no apparent pattern can be found to evaluate the use of pruning. The differences between these networks are partly relevant to the unbalanced distribution of classes in subjective rating. Most participant chose Exciting but few rated on Depressing or Neural; as a consequence, the models' behaviour is very unstable and unreliable.

In the Fig. 6, the time consumptions in testing step also showed an obvious trend of decrease with the reduction of the number of hidden unit. The original neural network spent about 2.6 ms to complete testing computation, while the other reduced network all spent significantly less time with fewer hidden units.

3.3 Electric signals based classification with recurrent neural network

The results and testing time for recurrent neural network on electric signals are shown in Fig. 7 and Fig. 8:

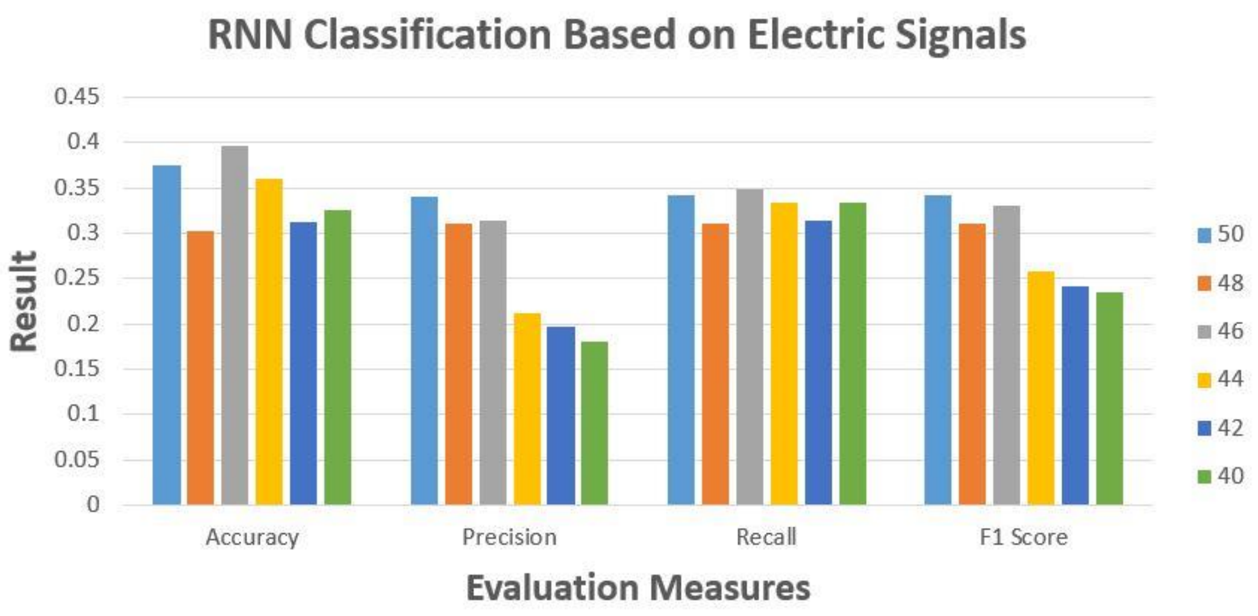

Fig. 7. RNN Classification Based on Electric Signals

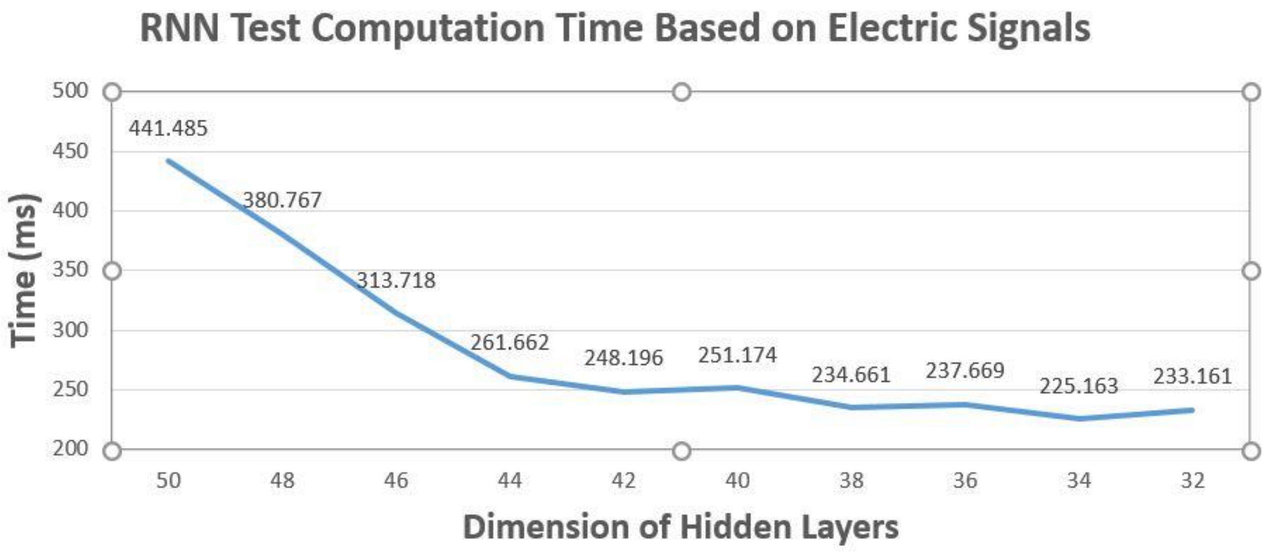

Fig. 8. RNN Test Computation Time Based on Electric Signals

In the Fig. 7, it can be observed that the first two reduced networks were capable to maintain similar performance as the original RNN model with similar evaluation results. When the models were pruned until a hidden layer dimension was less than 46, the precisions and F1 scores dropped rapidly. The recalls are stable on the level near 0.33, which means the model didn't work very well on classification and outcomes for each categories might occur equally for a specific class. In the Fig. 8, the line chart still shows the decreasing trend of testing time on each reduced RNN model. The computation time significantly dropped with the removals of contradictive hidden units.

This results demonstrate that a slight reduction on RNN make sense, but too much pruning may influence the normal functionality and stability of the network models. Time consumption is also reduced as what the models above showed.

4 Conclusion and Future Work

To pruning artificial neural network, I illustrated the use of vector angles on evaluating distinctiveness among hidden units. The experience conducted above validated the performance of observing the similarity among hidden units and removing some of them for network reduction. More hidden units often mean more parameters, and the improvement of neural network with reduction may not influence the accuracy, precision and recall, sometimes even become better without retaining the models.

The main benefit of network reduction should be the decrease of computation time and parameter space. A proper number of hidden units can avoid model redundancy and excessive training time. Rather than saving time on computing testing results, the fewer hidden units also reduce the variance of the network without training with too much parameters, so this reduction method can be used as a pre-experience to determine some potential numbers of hidden units in an artificial neural network.

However, some unreasonable results after pruning indicate that the stability and robustness of a neural network are quietly reliable on the distribution of dataset. A high quality dataset can promote the whole efficiency in network designing. To address this issue, more research on biased data processing and network unit observation are required to improve the implements. Additionally, which of similarity and contradiction demonstrated by vector angles should be operated as a priori is another factor that need to be considered in future work. A reasonable strategy is required to maintain functionality of original network as much as possible, which is quite associated with the application of the reduction technique.

References

[1] M. T. Manry, "Neural networks: Algorithms, applications, and programming techniques," *Neural Networks,* vol. 7, no. 1, pp. 209-212, 1994.

[2] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[3] O. Fujita, "Optimization of the hidden unit function in feedforward neural networks," *Neural Networks,* vol. 5, no. 5, pp. 755-764, 1992.

[4] M. Sheikhan, Z. Jadidi, and A. Farrokhi, "Intrusion detection using reduced-size RNN based on feature grouping," *Neural Computing and Applications,* vol. 21, no. 6, pp. 1185-1190, 2012.

[5] C. L. Giles and C. W. Omlin, "Pruning recurrent neural networks for improved generalization performance," *IEEE transactions on neural networks,* vol. 5, no. 5, pp. 848-851, 1994.

[6] J. Rahman, T. Gedeon, S. Caldwell, R. Jones, M. Hossain, and X. Zhu, "Melodious Micro-frissons: Detecting Music Genres From Skin Response," in *In 2019 International Joint Conference on Neural Networks (IJCNN)*, 2019: IEEE, pp. 1-8.

[7] S. Patro and K. K. Sahu, "Normalization: A preprocessing stage," *arXiv preprint arXiv:1503.06462,* 2015.

[8] T. Gedeon and D. Harris, "Network reduction techniques," in *Proceedings International Conference on Neural Networks Methodologies and Applications*, 1991, vol. 1, pp. 119-126.