Reduction Techniques for an Artificial Neural Network based on Similarity of Hidden Units

Wyman Wong

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 and redundant. The number of hidden units is difficult to decide so people trend 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 and increase the 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: Neural network, pruning techniques, 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.

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], so they are unnecessary for the utilization of the network and can be removed with particular conditions.

2 Methodologies

2.1 Data processing

I use a dataset from paper [4], which provides totally 192 patterns with 14 features and 8 groups of classes. For the first group of class, it consists of 3 categories: classical, instrumental and modern pop music. The other groups are feeling scales from the participants after they listen to the music.

To reduce the impacts from subjective movements, I used Min-Max normalization techniques to shift each feature into the range of minimum 0 to maximum 1 [5]. The equation for min-max normalization is:

$$value_{new} = \frac{value - value_{min}}{value_{\max} - value_{\min}}. \quad (1)$$

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 randomly spilt into 2 groups. 80% are training data and 20% are testing data. So there is probability that different data division may influence the stable outcomes of the network.

2.2 Implementation of an artificial neural network

I generate a feed-forward artificial neural network of three layers (input layer, 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 generic 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:

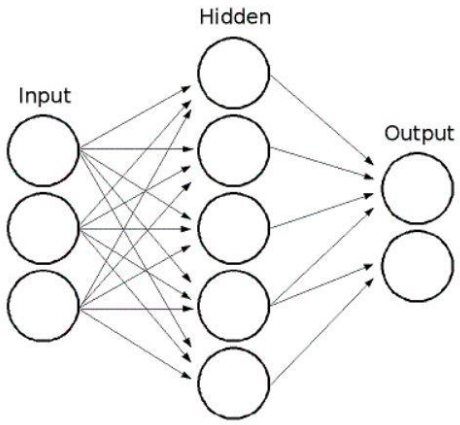

Fig. 1. Structure of an artificial neural network

In the network, hidden units use Sigmoid activation function to implement nonlinear operation, softmax to predict the classification results, mean squared error (MSE) loss as loss function and Adam for optimizer. The model is trained with learning rate of 0.01 and stop the training process once the training loss is less than 0.1000. The code for network is provided in Fig. 2:

```
# Hyper parameters.
features_num = 11
hidden_num = 30
classes_num = 3

# Definition of network structure.
class Net(torch.nn.Module):
    def init(self, input_size, hidden_size, output_size):
        super(Net, self).init()
        self.hidden = torch.nn.Linear(input_size, hidden_size)
        self.output = torch.nn.Linear(hidden_size, output_size)
        self.softmax = torch.nn.Softmax(dim = 1)

    def forward(self, input):
        z_hidden = self.hidden(input)
        a_hidden = torch.sigmoid(z_hidden)
        out = self.output(a_hidden)
        out = self.softmax(out)
        return out

# Create network, loss function and optimizer.
net = Net(features_num, hidden_num, classes_num)
criterion = torch.nn.MSELoss()
optimizer = torch.optim.Adam(model.parameters(), lr= learning_rate)
```

Fig. 2. Code for network

2.3 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:

$$precision = \frac{TP}{TP + FP}. \quad (2)$$

$$recall = \frac{TP}{TP + FN}. \quad (3)$$

$$F1 = \frac{2 * precision * recall}{prcision + recall}. \quad (4)$$

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:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}. \quad (5)$$

2.4 Improvement with pruning techniques

In 1991, Gedeon and Harris [6] propose that the hidden unit output activation vectors represent their functionality over input pattern and determine the distinctiveness of hidden units. Some of those similar 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 the vector angle in pairs over input space. 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. 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 °. Partition of the vector angels that I get from the trained model is shown in Table 2:

Table 2. Vector angles between the first five units

	…	**Unit 27**	**Unit 28**	**Unit 29**
…	…	…	…	…
Unit 23	…	39.01953	79.16623	72.38562
Unit 24	…	69.99639	57.78711	82.23487
Unit 25	…	56.63417	87.02668	4.69498

It can be seen from Fig.3 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. The value for unit 25 and unit 29 indicates a high similarity between the functionality of these two units, so one of them can be removed and add its weight values to the remained one. In this case, I remove unit 25.

As for the situation that more than two units perform similarly in the network, n-1 units can be removed and add their weight values to the remained one if n units are found similarity, an instance is shown in Table 3:

Table 3. Similarity among more than two units

	…	**Unit 15**	**Unit 16**	**Unit 17**	**Unit 18**
Unit 1	…	35.92233	22.67898	14.93795	24.99183
Unit 2	…	32.07097	10.32072	34.55152	36.80578
Unit 3	…	40.80959	39.57852	31.37539	1.36616
Unit 4	…	40.86298	40.03368	30.40262	12.79382
Unit 5	…	36.01481	27.29467	22.24233	0

In this case, unit 3, 4 can be removed and add the values to unit 18. However, to keep the stability of the network, the smaller values of vector angles will be operated as a priority, rather than removing all possible units. In this case, since the vector angle between unit 3 and unit 18 is 1.36616 degrees, I remove unit 3.

To pruning the network, here are several groups of removals for hidden units. Each group is indexed with sequence numbers and the form of (a -> b) indicates that remove unit a and add its weight values to those of unit b. All groups for music genre classification and subjective rating classification are listed in Table 4 and Table 5:

Table 4. Strategies of pruning network on music genre classification

Indexing	**Pruning strategies**
(I)	(No pruning)
(II)	(3 -> 18)
(III)	(29 -> 25)
(IV)	(25 -> 29) (16 -> 2)
(V)	(25 -> 29) (16 -> 2) (3 -> 18)
(VI)	(25 -> 29) (16 -> 2) (17 -> 1)

Table 5. Strategies of pruning network on subjctive rating classification

Indexing	**Pruning strategies**
(I)	(No pruning)
(II)	(24 -> 6)
(III)	(23 -> 9)
(IV)	(17 -> 3)
(V)	(17 -> 3) (23 -> 9)
(VI)	(17 -> 3) (23 -> 9) (24 -> 6)

After pruning the network based on the distinctiveness of hidden units, the structure of the neural network become 11 * (30 – n) * 3, since n hidden units have been removed. The number of hidden units has decreased but the joint effects remain in the neural network. So the reduced neural network requires no further retrain and can have a similar or better performance on given pattern spaces.

3 Results and discussion

3.1 Genre based classification

From the result of the vector angles, it can be seen that all values are within the range of 0 degree to 90 degrees, because the outputs from sigmoid function are non-negative values, which means no contradiction can be found in this method. The classification results can be seen in Table 6 and Fig. 3:

Table 6. Classification results based on music genre

Indexing	Pruning strategies	Accuracy	Precision	Recall	F1 Score
(I)	(No pruning)	0.6471	0.3722	0.5278	0.4038
(II)	(3 -> 18)	0.6176	0.3509	0.5093	0.3800
(III)	(29 -> 25)	0.6471	0.3722	0.5278	0.4038
(IV)	(25 -> 29) (16 -> 2)	0.6176	0.4394	0.5093	0.3866
(V)	(25 -> 29) (16 -> 2) (3 -> 18)	0.5294	0.2576	0.4500	0.3250
(VI)	(25 -> 29) (16 -> 2) (17 -> 1)	0.6471	0.3098	0.4306	0.3172

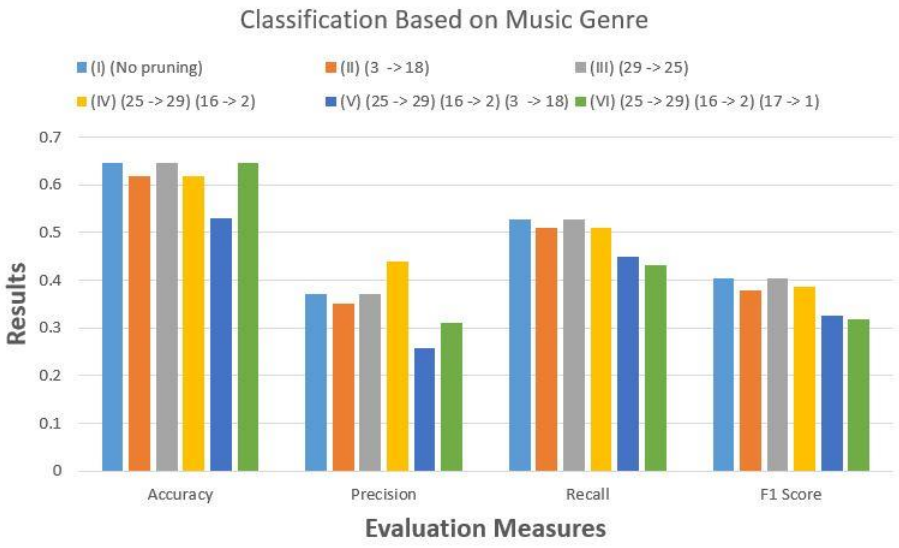

Fig. 3. Classification based on 3 music genres

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 can be observed that the pruning strategies (III) and (VI) can give the same accuracy as the network without pruning. (III) also gives a completely same result of other evaluation measures while (VI) shows a slightly lower result. (II) and (IV) give a slightly lower result in all measures, except for the precision of (IV), which is even greater than the original network by about 0.06.

The reason why (V) gives a worse result than that of the original network is probably that unit 3 and unit 18 respectively has complicated relationship with other units. Removing any one of them can significantly influence the performance of some of these units. It may also loss some important information from the abandoned unit.

3.2 Subjective rating based classification

The classification results on subjective rating (Depressing -> Neural -> Exciting) can be seen in Table 7 and Fig. 4:

Table 7. Classification results based on subjective rating

Indexing	Pruning strategies	Accuracy	Precision	Recall	F1 Score
(I)	(No pruning)	0.4324	0.2034	0.2013	0.1761
(II)	(24 -> 6)	0.2162	0.1529	0.1644	0.1429
(III)	(23 -> 9)	0.5135	0.3071	0.2923	0.1667
(IV)	(17 -> 3)	0.5405	0.2955	0.3013	0.1987
(V)	(17 -> 3) (23 -> 9)	0.5135	0.3071	0.2923	0.1667
(VI)	(17 -> 3) (23 -> 9) (24 -> 6)	0.5135	0.3775	0.2976	0.1966

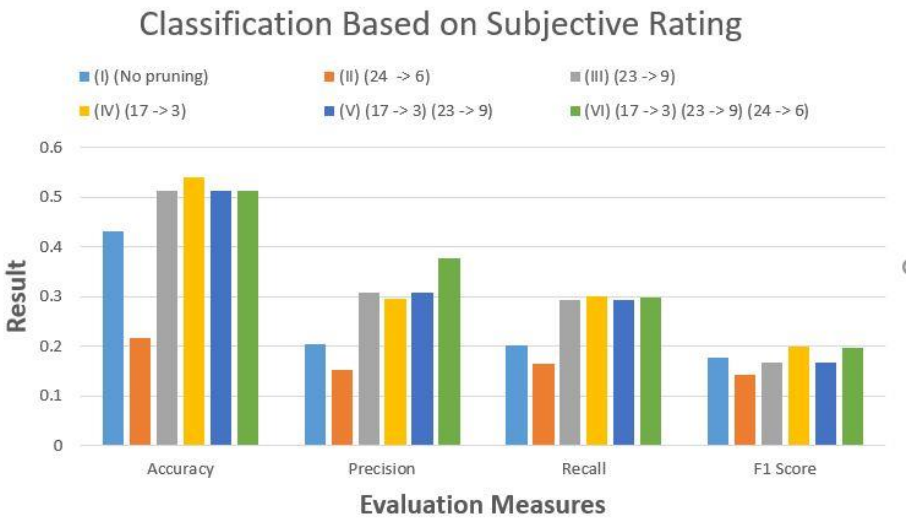

Fig. 4. Classification based on subjective rating (Depressing -> Neural -> Exciting)

As seen in Fig. 4, most strategies give better measures than those of the original network, except for (II), which produces a worse performance on classification. (III), (IV) and (VI) indicate a better performance than the original network and they give similar results on accuracy and recall. Actually, (VI) is observed that it gives an average level of measures and the highest precision among all strategies.

One situation should be pointed out that although (III) and (V) seem to have good performance, (IV) give a higher accuracy by about 0.03 than (III) or (V), which removed 2 hidden units and one of them is the same as (IV). 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.

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. Based on the best choice of hidden unit numbers, different strategies removed some redundant units from the network and basically gave the same or even better results on a series of evaluation measures without retraining the models.

However, some unreasonable results after pruning indicates 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.

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] 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.

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

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