Classification of Static Facial Expressions in the Wild (SFEW) Using Neural Network and Network Reduction Techniques

Jianyang Li
Research School of Computer Science, Australian National University
u5787914@anu.edu.au
May 2020

Abstract. The goal of this study is to develop a feed-forward neural network for classifying face expressions from Static Facial Expressions in the Wild database [1]. In the past, a person independent training and testing protocol for expression recognition is proposed, however the performance accuracy is significantly low due to real word conditions in the SFEW database. Thus we experimented it with a distinctiveness network reduction technique proposed by T.D Gedeon and D.Harris (1999) [3] on a built feedforward network trained by backpropagation, and our result shows that the effectiveness in certain extent when considering detect unnecessary hidden units and reduce network size, without offsetting much prediction performance accuracy.

Keywords: Static facial expressions, Feed-forward neural network, Network reduction technique, Distinctiveness hidden neuron

1 Introduction

Facial expressions are the facial changes in response to a person's internal emotional states, intentions or social communications. Realistic face data nowadays plays a vital role in area of automatic facial expressios analysis. However, human facial expression database had been captured in controlled 'lab' environments, before the present of the Static Facial Expressions in the Wild (SFEW) database [1]. SFEW has been developed by selecting frames from AFEW [2]. The database covers unconstrained facial expressions, varied head poses, large age range, occlusions, different resolution of face and close to real world illumination. It contains 700 images and have been labelled for seven basic expressions: *angry, disgust, fear, happy, neutral, sad and surprise*. The datasets we experimented on contains first 5 principal components of Local Phase Quantization (LPQ) [6] features and first 5 principal components of Pyramid of Histogram of Gradients (PHOG) features [5]. In this paper, we experiment with a feed-forward networks to predict static facial expressions classes from SFEW database.

When applying Feed-forward Neural Network for predicting static facial expressions, the major disadvantages of back-propagation method are that it can be slow to train networks. A problem is also raised on how to optimize the number of hidden neurons in hidden layers. Large number of hidden neurons could cause network to overfit, while small number of hidden neurons in hidden layers may fail to capture data features. Thus, we applied distinctiveness network reduction technique by T.D. Gedeon and D. Harris [3] to identify and remove unnecessary hidden neurons in the network, so that reduces computation resources and remains prediction accuracy in the same time.

2 Methodology

2.1 Data preprocessing

The static database has been extracted from the temporal dataset Acted Facial Expressions in the Wild (AFEW) [2], which contains in total 700 images with seven labels for basic expressions *angry, disgust, fear, happy, neutral, sad and surprise*. The face-emotion datasets we used in this paper contains 675 lines (with removal of 25 disgust from origin dataset) includes the following attributes:

- Image id – unique id number of images
- Label – label of images from seven basic emotion expressions categories: 1=angry, 2=disgust, 3=fear, 4=happy, 5=neutral, 6=sad, 7=surprise
- Column 3-7 – First 5 principal components of Local Phase Quantization (LPQ) features [6]
- Column 8-12 - First 5 principal components of Pyramid of Histogram of Gradients (PHOG) features [5]

LPQ is based on computing the short-term Fourier transform on a local image window and is calculated on grids and then concatenated for an image. PHOG descriptor is an extension of the histogram of oriented gradients (HOG) descriptor [5], which counts occurrences of gradient orientation in localized portions of an image, which has been used extensively in computer vision.

The emotion classification problem uses the input features is the face-emotion expression data, which is the columns 3 to 12, includes principal components for LPQ as well as PHOG. The target output is represented using 7 neurons, expressing by numerical data 1-7 for each emotion classes quantitatively.

2.2 Feed-forward neural network implementation

A two-layer fully connected neural network is developed, with first input layer containing ten neurons representing to the 10 face-emotion image input features, one hidden layer with certain hidden units in and last output layer with seven output neurons corresponding to seven expression labels.

To replicate the same implementation, 675 lines as dataset is randomly split into 346 lines as training set and the remaining as test data, to evaluate the network performance. In addition, we performed 10-fold cross validation method by randomly splitting data into 10 folds, to evaluate how accurate the network is when it comes to predict the expression labels on limited data sample.

By giving a new face-emotion image input data, such trained Neural Network is used to classify the face emotion into one of the seven expression categories.

Model Type	Unit	Learning Rate 0.01	Learning Rate 0.1	Learning Rate 0.2	Learning Rate 0.5
Classification	Test Accuracy	23.31%	23.20%	27.48%	22.52%
	Train Accuracy	34.63%	40.04%	44.57%	40.82%

Table. 1 Analysis of learning rate

Table 1 depicts the classification performance by a two-layer FNN with learning rate at different level. The local optimal number of hidden neurons is maximized at learning rate equals to 0.2 with test accuracy 27.48%. Therefore, we used the recommended learning rate 0.2.

2.3 Network reduction techniques and model design

Network reduction techniques by T.D. Gedeon and D. Harris [3], detect and remove redundant hidden units in the network. These excess units perform no real function in the final product and are unnecessary for the post-learning utilization of the network. Thus, such technique is applied in this experiment for two main objectives. Firstly, for increasing the efficiency of the network during actual use. Secondly, to observe and compare the network performance find in Section 2.2.

We implemented network reduction techniques – distinctiveness, that measures the significance and similarity of hidden units, and applied it on a trained the network with 20 hidden units as in Section 2.2. The following shows the main steps of implementation the *distinctiveness* technique for finding redundant hidden neurons:

- Step 1: For each hidden unit, a vector of the same dimensionality as the number of patterns in the training set is constructed. Such vector represents the functionality of the hidden unit in input pattern space, and each component of the vector corresponding to the output activation of the unit.
- Step 2: We applied method of 1.5 * IQR rule as a benchmark, to recognize insignificant hidden units. Each vector representation of hidden unit from Step 1 is transformed to its relative magnitudes. Therefore, by applying the rule, we calculate the first and third quantile of all the activation vector magnitudes, and units with activation vector magnitude below Q1-1.5 * (Q3 - Q1) are recognized as insignificant and can be removed.
- Step 3: The similarity of pairs of vectors is recognized by the calculation of the angle between them in pattern space. After removing insignificant hidden units by step 1 and 2, we compute the vector angles for all combination of two angle unit for the remaining hidden units. Vector angle are calculations are normalized to [-0.5, 0.5] by subtracting 0.5 from all activations, since all sigmoid activations are constrained to [0, 1]. Therefore, units with angular separations of up to about 15 degree are considered sufficiently similar, one of the unit is removed and the weight vector is added to the weight vector of the other unit. Similarly, units with angular separation over 165 degree will treated as complementary of each other, therefore both hidden units are remove

The effectiveness of the *distinctiveness* techniques is evaluated by comparing the performances from post-reduction model to the FNN model from Section 2.2 on both training and testing data.

2.4 Classification with genetic algorithm

We further applied genetic algorithm for training the FNN model. The genetic algorithm is applied here for estimating the optimal weights as an optimization problem, that would ideally help improve the predicting power of model built in

section 2.2. The genetic algorithm is associates with the size of population of random networks, initialisation operator, crossover probability and type, probability of mutation and selection operator. The implementation is explained below:

- Step 1: Initialize a population of random networks. With each individual network has 2 layers with 20 hidden units, 10 inputs neurons and 7 output neurons.
- Step 2: Fitness evaluation for each network in the population and replace least-fit population with new individuals.
- Step 3: Check if termination condition is reached. In this experiment we control the condition by number of evolutions.
- Step 4.1: If condition reached: return network with optimal weight.
- Step 4.2: If condition not reached: choose best fit of individual network for reproduction and breed new networks through crossover and mutation. Then back to Step 2 until condition is reached.

The above evaluation function is calculated using cross entropy loss function and returns the train accuracy to determine the fitness score. Then, the score is selected by highest test accuracy at evaluation in genetic algorithm.

Model	Unit	Crossover 0.5	Crossover 0.8	Hidden Unit 1.0
Classification	Accuracy	26.41%	28.07%	27.79%

Table. 2 Analysis of Crossover rate

Model	Unit	Mutation 0.01	Mutation 0.1	Mutation 1.0
Classification	Accuracy	25.90%	27.12%	26.02%

Table. 3 Analysis of Mutation rate

Random crossover operator is set, that each parent has an equal probability to pass on its corresponding weight to its child. We have used the recommended crossover probability at 0.8. And uniform mutation is adopted where the probability of mutating the child weights is constant for each evolution step, where we used mutation rate at 0.1.

2.5 Parameters and pruning process

There are other hyper-parameters that would affect the network performance, which would also have impact on the performance of network reduction technique. Networks from 2 hidden neurons to 20 hidden neurons are trained under different learning rate, to further investigate and compare the network reduction performance in terms of increasing efficiency as well as remaining prediction power. To measure the performance, we adopted 10-fold cross validation method by randomly splitting data into 10 folds, to evaluate how accurate each network is when it comes to classification power that to predict the expression labels on limited data sample.

3 Result and Discussion

3.1 Classification performance of feed-forward neural network

In this paper, we applied a two-layer FNN with 15 hidden units. As shown in Fig.1, both test accuracy and train accuracy increase rapidly in first 1000 epoch and then, it steadily further improves as the training and test loss decrease. Overall result reflects that given a new 10-input data on unseen face-emotion images, this FNN is 23.48% accurate in classifying the emotion into the correct expression category.

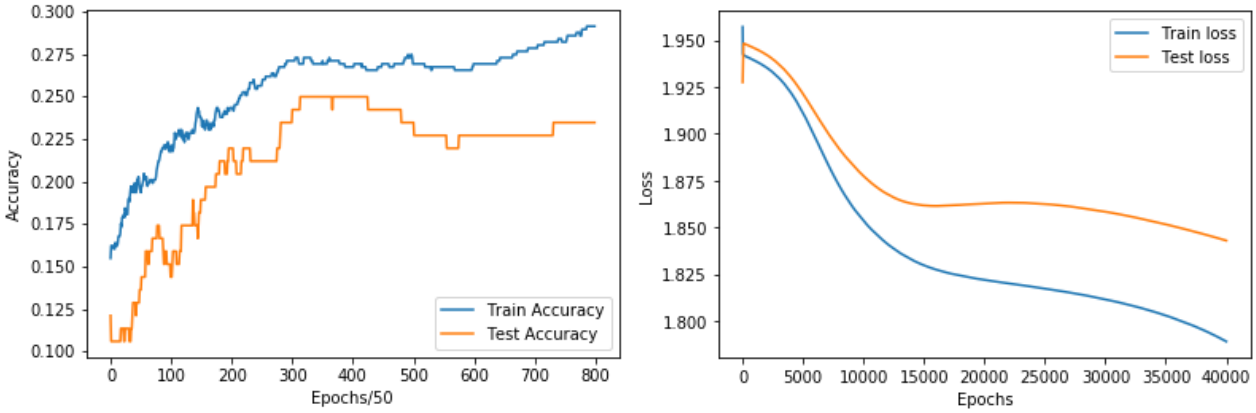

Fig. 1. Left: Train accuracy and test accuracy with number of Epochs. Right: Train loss and test loss with respect to number of Epochs.

Both results turn out to be lower than the result in the research by Dhall et al [1]. In their research, they mentioned that that such low accuracy may be due to the close to real world conditions in the SFEW database, which contains both high and very low-resolution faces that added to the complexity of classification problem. We could expect the classification accuracy improve for experiment on lab-controlled data.

3.2 Analysis of network reduction techniques

We also analyzed on to what extent network reduction technique is useful in reducing the network size without decrease its modelling power as comparing to the result from section 3.1.

As a result, the number of removed hidden units generally increases when the number of hidden units used in network increases. This reflects that the network reduction techniques indeed remove the redundant units. However, for most of the network, the application of network reduction techniques is not obvious, as there are little hidden units removed through the process. This could be a limitation of the implementation, that more hidden units being similar or complementary are not detected.

Network Reduction on No. Hidden Neurons and Learning Rate								
	0.001	0.005	0.01	0.05	0.1	0.2	0.5	Total
2	0	0	0	0	0	1	0	1
3	1	0	0	0	0	0	0	1
4	1	0	0	0	0	0	0	1
5	1	1	0	1	0	0	0	3
6	0	1	0	0	0	0	0	1
7	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0
9	1	1	0	0	0	0	0	2
10	0	1	0	0	0	0	0	1
11	1	0	0	0	0	1	0	2
12	1	0	0	0	0	0	0	1
13	0	0	0	1	0	0	0	1
14	0	1	0	0	0	0	0	1
15	1	0	0	0	0	0	0	1
16	1	1	1	1	0	0	0	4
17	1	1	1	0	0	0	1	4
18	0	1	1	0	0	0	0	2
19	1	0	0	1	1	0	0	3
20	1	0	1	0	0	0	0	2
Total	11	8	4	4	1	2	1	

Table. 4 Number of hidden units removed for network with 2 to 20 hidden neurons at different learning rate.

To further investigate the network reduction technique and network performance, we applied the network reduction technique on networks with different number of hidden neurons at each learning rate. As shown in Table.4, the total number of hidden units removed generally increases as the learning rate turns lower. The learning rate is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated. Such result may be explained by the fact that the network trained using small learning rate allows the model to learn a more optimal or even globally optimal set of weights but may take significantly longer to train. And for networks with larger learning rate, models are learned faster, but at the cost of arriving on a sub-optimal final set of weights. Therefore, smaller training rate could associate with over-fitting problem, and it may appear to have more redundant hidden units. Furthermore, we discovered that network reduction performance is highly dependent on the final weights assigned between hidden units.

To compare the network performance before and after applying the network reduction technique, we computed the percentage of increase to average training accuracy and average testing accuracy using the performance data before and after the network reduction. The test and train accuracy remain same when there are no hidden units removed from the origin network. For cases where hidden units were removed, Fig.2 illustrates that the median of percentage improvement of train and test accuracy is 0, which implies the success of network reduction techniques that manages to reduce the number of hidden units and preserve the prediction power at the mean time.

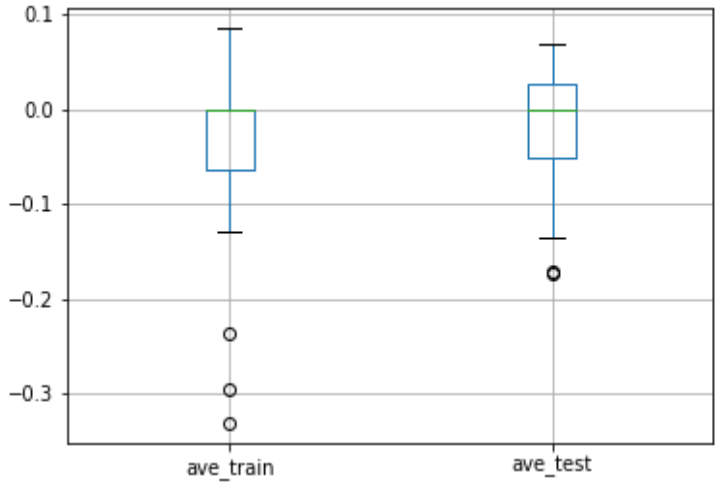

Fig. 2 % increase of average training and testing accuracy after network reduction.

However, it is worth mention that there are cases that the post-reduction network test accuracy is significantly smaller, which are depicted as in Fig.2 as outliers. All such cases appeared when learning rate is large (above 0.1), as shown in Table.5, where the average training accuracy is down by almost 18% in average, and 7% for testing accuracy. This could imply some limitation and restriction of such technique implementation that hidden units with great prediction power was mistakenly removed during the network reduction process.

Number of Hidden Unit	Train Accuracy	Train Accuracy with reduction	Test Accuracy	Test Accuracy with reduction
5	40.52%	27.11%	23.78%	17.17%
10	28.91%	20.35%	23.05%	15.77%
15	45.22%	26.67%	27.48%	20.06%

Table. 5 Average training accuracy and average testing accuracy before and after the reduction technique with network trained at specific learning rate.

Overall, when we applied the same network reduction technique to our model, it is shown that the classification accuracy drops significantly with each hidden neuron pruned. Thus, we can conclude that this pruning technique should not be applied to the SFEW dataset with the LPQ and PHOG descriptors as features for our model.

3.3 Analysis of network reduction techniques with genetic algorithm

The result shows that when we applied genetic algorithm to train the FNN model, it reduces error in learning which in turn improve the performance of the model at the training stage. The graph below outlines that convergence is improved by introducing genetic algorithm.

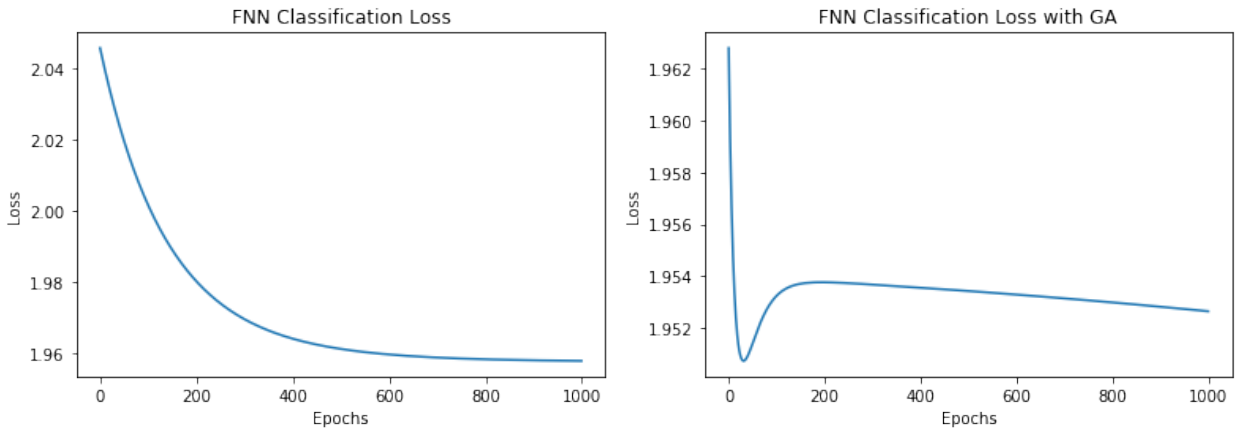

Fig. 3 Performance of classification loss in two models for training

We found that the *distinctiveness* network reduction technique works well when applying on networks learned by genetic algorithm, especially using small learning rate, which to some extent is useful in pruning redundant hidden units while maintain the network prediction power. The networks with learning rate ranged from 0.001 to 0.05, the trend for average change testing accuracy increases with decrease in change of testing accuracy. One possible reason could be that the hidden units which cause the overfitting problem due to small learning rate, was removed during the network reduction process, which causes the training accuracy to decrease.

Test Accuracy	Without Reduction	With Reduction
FNN	27.48%	20.06%
FNN with GA	30.91%	29.37%

Table. 6 Comparing test accuracy under different models with and without network reduction technique.

When model is trained using genetic algorithm, the local optimal average testing accuracy across different number of neurons is 30.91%, at learning rate set as 0.2. The reduction technique on this network shows a slight decrease in accuracy of 1.54%. This result implies that network classification power is decreased on a new face-emotion data when applying network reduction technique on modal trained using genetic algorithm, however its less than the percentage decrease in test accuracy as compared to model trained using FNN, meaning that to some extent it is useful in pruning redundant hidden units and improves over-fitting problem. Thus, we can conclude that the model trained under genetic algorithm outperforms the original model. An in either case, this pruning technique shows a negative impact on classification power therefore should not be applied to the SFEW dataset with the LPQ and PHOG descriptors as features for our models.

4 Future Work and Conclusion

Feed-forward Neural Network is used for predicting static facial expressions, the major disadvantages of back-propagation method are that it can be slow to train networks. To optimize the number of hidden neurons in hidden layers, distinctiveness network reduction technique managed to identify and remove unnecessary hidden neurons in the network and remains classification power in the same time.

The two-layer neural network is proved not sufficient due to low testing accuracy. This may be due to the close to real world conditions in the SFEW database, which contains both high and very low-resolution faces that added to the complexity of classification problem. On the other hand, the *distinctiveness* network reduction technique is helpful to a certain extent for removing unnecessary hidden units without sacrificing model's ability for our classification tasks.

It is also shown that the network reduction techniques on genetic algorithm trained model is slightly better than the network reduction techniques on backpropagation trained model in terms of classification accuracy. To further improve the network technique, a different mutation and crossover operations could be used to generate a better network.

As for future improvement on network model, only number of hidden units and learning rate is examined in this network model, while there are other factors could affect the overall performance of network reduction technique on the model, such as batch size. Thus, for future investigation, more hyper-parameters could be applied such as including batch size and more hidden layers. Especially for image classification, could build convolutional neural network through tens or hundreds of hidden layers, in order to make the network reduction technique more robust.

References

1. A. Dhall, R.Goecke, S.Lucey, and T.Gedeon. Static Facial Expression Analysis in Tough Conditions: Data, Evaluation Protocol and Benchmark, 2011.
2. A. Dhall, A. Asthana, R. Goecke, and T. Gedeon. Emotion recognition using PHOG and LPQ features. In Proceedings of the Ninth IEEE International Conference on Automatic Face Gesture Recognition and Workshops (FG'2011), Facial Expression Recognition and Analysis Challenge Workshop (FERA), pages 878–883, 2011
3. T.D. Gedeon and D. Harris. (1991). Network reduction techniques. Proc. International Conference on Neural Networks Methodologies and Applications, 2:25–34.
4. A. Dhall, R. Goecke, S. Lucey, and T. Gedeon. Acted Facial Expressions in the Wild Database. In Technical Report, 2011.
5. N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recogni- tion, CVPR'05, pages 886–893, 2005.
6. V. Ojansivu and J. Heikkil. Blur Insensitive Texture Classi- fication Using Local Phase Quantization. In Proceedings of the 3rd International Conference on Image and Signal Pro- cessing, ICISP'08, pages 236–243, 2008.
7. Han, J., Kamber, M., & Pei, J. (2011). In Data Mining: Concepts and Techniques (ch.8 Classification: Basic Concpets). The Morgan Kaufmann Series in Data Management Systems Morgan Kaufmann Publishers.
8. Cui, X., Zhang, W., Tüske, Z., & Picheny, M. (2018). Evolutionary stochastic gradient descent for optimization of deep neural networks. In Advances in neural information processing systems (pp. 6048-6058).
9. Sietsma,J, Dow, RF, "Neural net pruning - why and how," IJCNN, vol. I, pp. 325-333, 1988.
10. S. Lucey, I. Matthews, C. Hu, Z. Ambadar, F. de la Torre, and J. Cohn. AAM Derived Face Representations for Robust Facial Action Recognition. In Proceedings of the IEEE Inter- national Conference on Automatic Face and Gesture Recog- nition, FG'2006, pages 155–162, 2006.
11. Dalal, N. and Triggs, B., 2005, June. Histograms of oriented gradients for human detection. In 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05) (Vol. 1, pp. 886-893). IEEE.

12. Ojansivu, V. and Heikkil, J., 2008, July. Blur insensitive texture classification using local phase quantization. In International conference on image and signal processing (pp. 236-243). Springer, Berlin, Heidelberg.