# Classifying Posed and Genuine Anger from Observers' Pupillary Response Using Distinctiveness-based Network Pruning

Ernest Kun Chi Kwan

Research School of Computer Science,
Australian National University
Acton ACT 2601 Australia
u6381103@anu.edu.au

**Abstract.** Recent studies have shown that physiological signals of observers can predict the veracity of emotions better than the verbal response of the same individuals. We construct a simple feedforward neural network to classify observed genuine or posed anger from the pupillary responses of the observers. Pruning techniques are then used to improve the performance of the network, by identifying hidden neurons with similar functionality. We consider two different pruning methods, one based on the behaviour of the hidden neurons and the other based on the trained weight matrix. We conclude that only the distinctiveness of neuron output activation vectors was appropriate for pruning the network, which increased the test accuracy to 75%. We were not able to achieve the same accuracy (95%) in Chen et al.'s study. Nonetheless, it is considerably better than human judgement (60%).

**Keywords:** emotion veracity, anger recognition, eye gaze, pupil dilation, physiological signals, machine classification, network pruning

## 1 Introduction

Facial expressions are important signals in face-to-face communication that provide impressions and feelings to observers. They can reveal the emotions of the displayer of the facial expressions and also affect the observer's mental state. These expressions can be real or posed. They are considered posed if the people were instructed to act and they do not carry genuine feelings. Creating a system that can classify posed and real emotions is valuable and could be applicable in different areas such as customer service, law enforcement and the health sector.

Recent studies [2] [3] have shown that observers' physiological signals can be used to classify real and posed smiles and anger. Physiological signals are particularly useful for machine classification because they are independent of conscious human control and are not affected by subjective biases. Data obtained in [2] was used for this study, where pupillary response patterns were used to predict the veracity of anger. In the experiment, videos of real and posed anger were displayed to 20 participants in a laboratory. While the videos were being played, a remote Eye Tribe eye gaze tracker was used to measure the eye gaze and pupil size. After each video, the participants were asked if the anger shown in the video was genuine. The mean accuracy of their verbal responses is 60%, not much higher than chance (50%).

With a simplified dataset from the previous study, we aim to replicate the results using a simple feedforward neural network. The goal is to reach the highest accuracy possible and at least 60%. A binary classifier is trained to predict genuine or posed emotion. We then attempt to improve the performance of the neural network by pruning neurons. The number of hidden neurons in a neural network may be more than necessary and a smaller neural network can significantly reduce computational complexity. In theory, neurons with similar functionality can be pruned with little to no negative impact on the model's performance and retraining will not be required. Gedeon and Harris [4] proposed a number of network reduction techniques and Gedeon [8] used the *distinctiveness* measure for pruning neurons successfully. Inspired by [1], this study adopts a similar analysis approach and examines pruning processes based on the behaviour of the hidden neurons and the trained weight matrix. The performance of the model after applying each pruning method is evaluated.

## 2 Methodology

### 2.1 Data Preparation

The dataset used for this study was a pre-processed dataset from [2]. There are 400 entries, as 20 videos (10 genuine, 10 posed) were displayed to 20 participants. Each row of data contains 9 features, which are as follows.

- Participant – 20 participants
- Video – 10 videos for the true angry emotion and another 10 videos for the false angry emotion.
- Mean – The mean of in pupillary response.
- Std – The standard deviation of in pupillary response.
- Diff1 – The change of left pupillary size after watching a video.
- Diff2 – The change of right pupillary size after watching a video.
- PCAd1 – An orthogonal linear transformation with first principal component.
- PCAd2 – An orthogonal linear transformation with second principal component.
- Label – Genuine or Posed emotion.

The first two columns were discarded since we are predicting using the numerical representations of pupillary responses only. The last column, Label, is the target output and the rest are the input features.

**Table 1.** Summary of the input features

| | **Mean** | **Std** | **Diff1** | **Diff2** | **PCAd1** | **PCAd2** |
|---|---|---|---|---|---|---|
| **count** | 400 | 400 | 400 | 400 | 400 | 400 |
| **mean** | 0.889090 | 0.102462 | 0.008421 | 0.209575 | 0.030703 | 0.121382 |
| **std** | 0.058538 | 0.083080 | 0.008153 | 0.097787 | 0.014715 | 0.029972 |
| **min** | 0.582885 | 0.007986 | 0.001081 | 0.021851 | 0.010325 | 0.061143 |
| **25%** | 0.852057 | 0.040081 | 0.002415 | 0.114244 | 0.019492 | 0.104173 |
| **50%** | 0.897453 | 0.068729 | 0.004292 | 0.234337 | 0.028206 | 0.117531 |
| **75%** | 0.934203 | 0.159266 | 0.013236 | 0.290046 | 0.035855 | 0.137517 |
| **max** | 0.983422 | 0.358368 | 0.044005 | 0.418234 | 0.083780 | 0.239318 |

Since the input features have different ranges, they must be rescaled before training the neural network. Min-max normalisation was applied to scale the data to [-1, 1], so the model will not be dominated by variables that use a larger scale. Since the data entries was sorted, they were shuffled before splitting into 80% training set and 20% test set.

### 2.2 Neural network structure and hyperparameters

A simple feedforward neural network was constructed using Pytorch, which consists of one input layer, one hidden layer and one output layer. Backpropagation was used to train the model. Sigmoid activation function and cross entropy loss were used since this is a binary classification problem. We used the Adam optimiser [7] since it performs better than SGD. The learning rate starts off as 0.01 and reduces to 0.001 as the model is being trained. The hidden layer has a size of 20 neurons. It was intended to be larger so that there would be redundant neurons for us to apply pruning.

### 2.3 Neuron pruning by distinctiveness

Once the neural network was fully trained, two different pruning methods were applied to the network. A simple and computationally cheap method is pruning by *distinctiveness*. The *distinctiveness* of hidden neurons is determined from the neuron output activation vector over the pattern set [4]. Then cosine similarity is used to calculate the angle between the vectors formed by the neuron's output activation values. Thus, we get a vector for every neuron and angles between all pairs of neurons. If similar neurons are found (<15°), one of the neurons is pruned and its weight is added to the other. If complementary neurons are found (>165°), both can be pruned. If the angles are low, it means that the neurons are very similar and the resulting network after pruning should not require further training.

**Table 2.** Example of angles between vectors

| Pairs | Angle | |
|---|---|---|
| 6 7 | 21.9915 | |
| 6 8 | 44.9817 | |
| 6 9 | 109.1802 | |
| 6 10 | 83.2963 | |
| 6 11 | 104.8257 | |
| 6 12 | 146.2123 | |
| 6 13 | 101.7704 | |
| 6 14 | 1.2196 | similar |
| 6 15 | 93.716 | |
| 6 16 | 91.3762 | |
| 6 17 | 175.7510 | complementary |
| 6 18 | 3.2649 | similar |
| 6 19 | 43.1632 | |

These are the steps of the pruning process in detail.

1. Get the activation output vectors from the model's hidden layer
2. Normalise the vectors to [-0.5, 0.5] to use the angular range of 0-180°. Since a sigmoid activation function was used, the values had range [0, 1]. So, we simply subtract 0.5 from all vectors.
3. Calculate the cosine similarity for each neuron pair and convert them to angles in degrees.
4. Find the neuron pair with min(angle, 180 - angle).
5. If the neurons are similar, prune one of them. If they are complementary, prune both.
6. Train the neural network until convergence.

In step 5, pruning is achieved by setting the weight and bias of the corresponding neuron to zero, removing the neuron in effect. In practice, the neuron will be deleted so that space and time can be saved. We train the neural network for additional epochs in step 6 because the neurons removed are not completely identical, and we want to find the best accuracy possible after pruning. In order to evaluate the performance of different numbers of neurons, steps 1-6 were repeated until one neuron is left. The minimum angle between the vectors and the validation loss were recorded for each neuron pruned. We also observed the number of epochs in step 6 were also to compare how long it takes to retrain the network. Thus, we can see how the pruning process has affected the model performance over time.

In [1], Gedeon found that the static weight matrix was coarsely approximating the behaviour of hidden neurons. Hence, our second method is to calculate angles between the weights connecting the hidden neurons to the output neurons, using the *distinctiveness* approach as above. The process is very similar to the steps above. We get the weight matrix vectors from the model's second layer. In step 2, the vectors need to be normalised between the global maxima and minima. Again, steps 1-6 were repeated until one neuron is left.

Finally, we try to train the final model, pruning similar and complementary neurons only, and compare the results.

## 3 Results and Discussion

### 3.1 Performance of the neural network before and after pruning

**Table 3.** Test accuracy of neural network classifier

| | Number of hidden neurons | Test accuracy |
|---|---|---|
| Without pruning | 20 | 68.75% |
| Pruning (neuron activation output) | 6 | 75% |
| Pruning (weight matrix) | 5 | 73.75% |
| Final model | 13 | 72.5% |

The test accuracy of the simple neural network without pruning was 68.75%, which is the lowest in the 4 cases. The highest accuracy achieved was 75% when there were 6 hidden neurons during the pruning process based on neuron output activation vector, which is a 6.25% increase in accuracy. This shows that the pruning method based on neuron behaviour can increase the performance of the neural network. The final model has more neurons than the best models during the evaluation of pruning methods and obtained a lower accuracy. This indicates that the initial choice of 20 hidden neurons was way too high, as a high accuracy can be achieved with 5-6 hidden neurons. Since the final model stopped at 13 hidden neurons, it may be beneficial to continue pruning neurons with degrees higher than 15°. Overall, the neural network was better than chance (50%) and the observers' verbal response in [2] which is 60%. However, it is still far away from the accuracy reached in [2] using trained machine classifiers, which is 95%.

Note that the accuracy of the neural network before and after pruning are subject to randomness. Since the partitioning of training set and test set and the weights of the neural network are both random, the state of the hidden neurons can be very different in different occasions. It is possible that pruning the network may lower its performance. Therefore, it is important to save the model every time before pruning a neuron.

## 3.2 Analysis of distinctiveness pruning

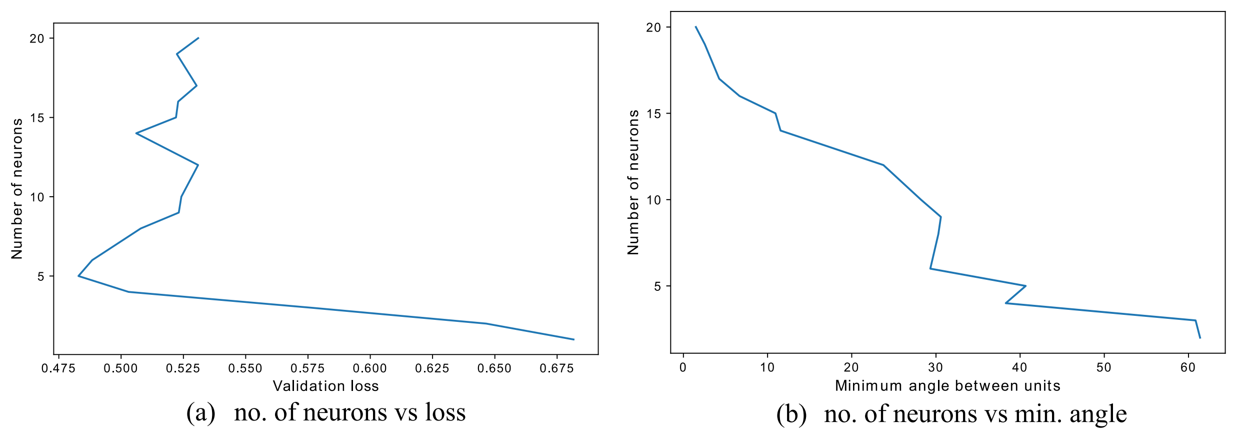

**Fig. 1.** Validation loss and minimum angle between vectors for pruning based on neuron behaviour

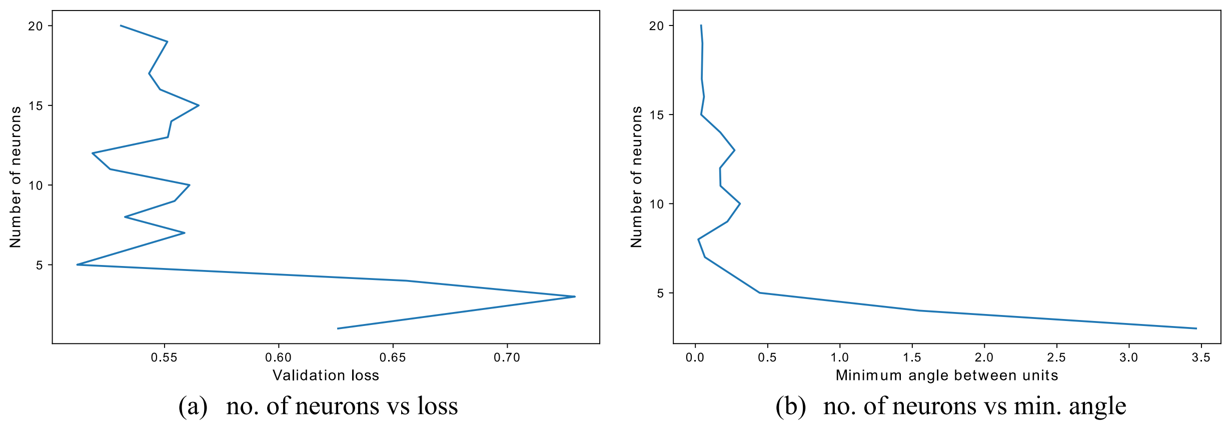

**Fig. 2.** Validation loss and minimum angle between vectors for pruning based on weight matrix

The curves for both pruning methods have similar shapes, which means they follow similar trends. The validation loss for both methods fluctuates from 5 to 20 neurons but gets much higher when the hidden neurons drop below 5, as shown in Fig. 1(a) and 2(a). This suggests that the 5 neurons were all important so at least 5 neurons are needed to obtain a good accuracy for the classification task. When pruning the first few neurons based on neuron behaviour, the network was able to obtain a better performance than the initial network, as shown in Fig. 1(a). This is due to the fact that the network is finetuned for some addition epochs after each removal, and it confirms that there were many unnecessary hidden neurons. Although a small angle indicates similar functionality, the validation loss sometimes increases when a 'similar' neuron is removed, which is why we always retrain for a bit whenever we prune a neuron.

The minimum angle between vectors increases as the number of neuron decreases, which is expected. However, the angles of the second method were all extremely small, which indicates that the vectors were all similar. That is not the case, as the angles are much larger in neuron output vectors. The reason is that the neural network had only 2 output neurons, that is, the weight matrix vectors were only 2-dimensional. Since the range of the weights was also fairly small, this made the vectors extremely similar when the *distinctiveness* approach was used. Thus, the second pruning method was not suited for this task. Even though it was able to reduce the validation loss and obtain a high test accuracy, it was simply by coincidence. If this was not a binary classification problem, the method might have been able to produce meaningful results.

As the minimum angle between units increases, the number of epochs needed to retrain until convergence also increases. This is expected as more significant neurons were pruned, the network will require more retraining.

## 4 Conclusion and Future Work

We demonstrated the impacts of the two pruning methods on our neural network. The first pruning method based on neuron behaviour was able to increase the accuracy of the classifier while reducing its computational complexity. The second pruning method based on weight matrix is not applicable to binary classification. We showed the effectiveness of a simple measure used to identify neurons with similar functionality. We also showed that a simple feedforward neural network is capable of learning to identify genuine or posed anger which outperforms human judgement, although not as high as the previous study.

There are rooms for improvement as we can evaluate other parameters for pruning in more detail. For example, we can analyse the results of pruning using different learning rates, different threshold for the angles to be considered similar and different number of epochs after the removal of a neuron. We can also prune using other properties in [4], such as sensitivity and badness.

In addition, we can conduct a study on the original time series dataset in [2] instead of the pre-processed dataset. Sophisticated neural networks with more hidden layers may be used to improve the performance.

## References

1. Gedeon, T.D.: 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. 26–29 (1995). https://doi.org/10.1109/annes.1995.499431.
2. Chen, L., Gedeon, T., Hossain, M.Z., Caldwell, S.: Are you really angry? Detecting emotion veracity as a proposed tool for interaction. Proceedings of the 29th Australian Conference on Computer-Human Interaction. 412–416 (2017). https://doi.org/10.1145/3152771.3156147.
3. Hossain, M.Z., Gedeon, T.: Classifying posed and real smiles from observers' peripheral physiology. Proceedings of the 11th EAI International Conference on Pervasive Computing Technologies for Healthcare. 460–463 (2017). https://doi.org/10.1145/3154862.3154893.
4. Gedeon, T.D., Harris, D.: Network reduction techniques. Proceedings International Conference on Neural Networks Methodologies and Applications. 1, 119–126 (1991).
5. Wong, P.M., Gedeon, T.D., Taggart, I.J.: An improved technique in porosity prediction: a neural network approach. IEEE Transactions on Geoscience and Remote Sensing. 33, 971–980 (1995). https://doi.org/10.1109/36.406683.
6. Sietsma, J., Dow, R.J.F.: Neural net pruning - why and how. IEEE International Conference on Neural Networks. 1, 325–333 (1988). https://doi.org/10.1109/icnn.1988.23864.
7. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980. (2014).
8. Gedeon, T.D., Harris, D.: Progressive image compression. [Proceedings 1992] IJCNN International Joint Conference on Neural Networks. 4, 403–407 (1992). https://doi.org/10.1109/ijcnn.1992.227311.