Generating Human-Like Explanations for Shallow and Deep Neural Networks

Matthew Muscat,

Research School of Computer Science, Australian National University, Canberra, ACT 2600, Australia, u6988493@anu.edu.au

ABSTRACT

We have trained a shallow and deep feed-forward neural network using back-propagation to detect and distinguish perceived anger – either posed or genuine. The utilised dataset contains subjects' pupillary responses when shown a visual of an unknown person's face displaying anger. These facial displays of anger were either genuine or posed, allowing us to train two binary classification neural networks. From this, previous findings that pupillary response can be used to predict the validity of perceived anger were replicated.

Model outputs alone are not sufficient to uncover why subjects' pupillary responses can be used to predict the authenticity of facial anger more reliably than the subject's own verbal prediction. We have generated a reasonable explanation as to which input variables are most significant in providing predictive power for a shallow and deep neural network. Therefore, we have provided insight into how neural networks can predict anger authenticity more reliably than humans.

Keywords: Emotional authenticity, Anger, Feedforward, Network logic, Sensitivity analysis, Pruning

1 Introduction

Neural networks are loosely designed to operate like the human brain and provide us with a way to detect relationships between very large amounts of data. They are organised into layers; including an input, output and a determined number of hidden layers. Generally information flows forward through the network, commonly known as a feed-forward neural network, with back-propagation utilised to update the weights of the nodes within the network. These weights are updated based upon how well the networks previous prediction matches the ground-truth data, where the ground-truth data is the human-verified (or labelled data) supplied to the network. Neural networks are not exclusively a modern phenomenon, with research on the topic dating back to the 1940's. While powerful, neural networks do not generally offer detailed information as to how they are actually making their predictions. Black box predictions are sometimes viewed as untrustworthy, especially by the wider public. This distrust is warranted as without knowing how a neural network prediction is being made, it is impossible to understand all of the biases and potential out-of-sample errors that may occur while making predictions over real-life data. Research into the black box aspect of neural networks has allowed us to gain a general understanding of the features and rules set by neural networks which provide the models predictive power. Methods such as pruning, sensitivity analysis and a characteristic method, as discussed by T.D Gedeon Et al. [1], have been shown effective in providing insight into neural network's predictions. We will focus these methods onto a specific neural network, to try to identify logic used by the network when making predictions about the sincerity of facial emotions.

As demonstrated by Chen, L, et al. [2], the activity of a subject's pupils when viewing facial emotions can allow the authenticity of the facial emotion to be discerned. It was shown that while viewing facial displays of anger, the authenticity of the display is predicted by a neural network at a higher accuracy (when observing the viewer's pupillary response) (95 percent) than by listening to the human viewer's own prediction (60 percent). This was achieved by tracking eye gaze and pupil size of the participants in the experiment. The data obtained from this experiment included the mean and standard deviation of pupil response, two differences between the left and right pupil response, and the first and second principle components of the pupillary response [3]. We used this data to train both a shallow and deep neural network. By applying the methods discussed above to derive rule sets, we were able to explore the logic behind the predictions of neural networks.

2 Method

2.1 Model Architectures

Two models were developed for the purpose of this experiment. The first is a shallow feed-forward neural network with a model architecture similar to the initial model described by Chen, W [3]. The PyTorch library was used to create and train the network. There are 6 nodes in the input layer; these are described in more detail within section 2.2. There is one hidden layer with 10 nodes, and finally 2 output layers representing the genuine and posed labels to be predicted.

The neural network was trained using the Rprop optimizer (resilient backpropagation). As opposed to regular backpropagation, Rprop allowed for dynamic learning rates to adapt the learning process to the topology of the loss function [4] leading to higher accuracy predictions. Given that this problem is a classification task, the sigmoid activation function given below was used.

$$y = \frac{1}{1+\exp(-x)}. \tag{1}$$

As the problem is a binary classification task, the cross-entropy loss function given below was used to measure the error.

$$loss = \frac{1}{N}\sum_{1}^{N} y_i \log\big(p(y_i)\big) + (1-y_i)\log(1-p(y_i)). \tag{2}$$

The second network is a deep feed-forward neural network with two hidden layers. It was found that extending the number of hidden layers beyond this (with our small dataset of 400 rows) resulted in very quick overfitting (high training accuracy) and poor test results (less than 80 percent test accuracy). Overfitting could be somewhat controlled by reducing the number of hidden nodes in subsequent layers to very small numbers, however this was deemed insufficient to support further layers as the testing accuracy drop was still noticeable. Therefore the model was designed with a first hidden layer of 10 hidden nodes, and a second hidden layer of 6 hidden nodes. The model was once again trained using the resilient backpropagation method and using sigmoid activation functions over both layers. The output remained binary; therefore the cross-entropy loss function was used again.

All model training and data manipulation was performed using the PyTorch, Pandas, NumPy, Matplotlib and Scikit-Learn packages.

2.2 Data

The pupillary response data set was obtained from previous work by Chen, L, et al. [2]. The original data set includes a video and participant label; these were removed from the data set as we are interested only in the binary classification label column and the six input variable columns. These are summarised below [3].

- Label: Binary ground truth for the display of anger (genuine or posed)
- Mean: The mean pupillary response value over both eyes.
- Std: The standard deviation of the pupillary response over both eyes.
- Diff1: The changes in the size of the left pupil while watching the video.
- Diff2: The changes in the size of the right pupil while watching the video.
- PCAd1: The orthogonal linear transformation with first principal component.
- PCAd2: The orthogonal linear transformation with second principal component.

The label column was altered to display genuine as 0 and posed as 1, this was performed to transform the output into a numerical binary format. The variables in the dataset have been normalised to fit between 0 and 1, to place all input features on the same scale and allow the neural network to effectively determine which inputs are most important. There are 400 rows within the dataset, consisting of data points from 20 different videos shown to 22 participants [5]. 10 of the videos displayed genuine anger while the other 10 displayed posed anger; therefore there are 200 genuine and 200 posed labels within the dataset. The data was divided into an 80/20 split to form the training and testing data respectively, allowing us to train the model and produce unbiased testing accuracies.

2.3 Model Training and Analysis

An initial learning rate of 0.01 was used in both models, the same as used by Chen, W [3]. Note that the Rprop optimizer was used, and therefore this initial learning rate is not representative of the learning rate throughout training.

The number of epochs for the training cycle in the shallow model was limited to 3000, further training beyond 3000 epochs lead to exponentially smaller gains in model accuracy. This is demonstrated by figure 1 below showing the total loss value throughout a 5000 epoch training cycle. Further training the model after 3000 epochs would likely lead to overtraining and less generalizability in prediction, with little upside in accuracy.

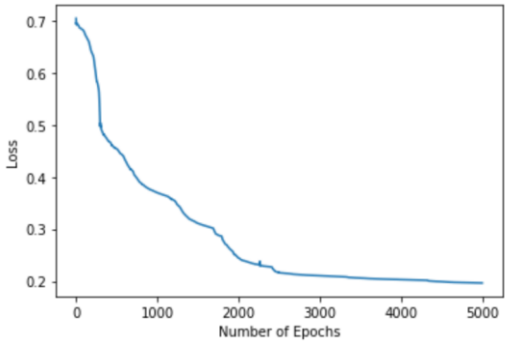

Fig. 1. Total loss value plotted during a training cycle of 5000 epochs.

At 3000 epochs a training accuracy of over 90 percent and a testing accuracy of over 80 percent are readily achievable. Due to the small size of the dataset and the non-deterministic nature of the model, these accuracies fluctuate slightly with each training cycle even if the training/test data split is held constant.

Similarly in the deep model, experiments were run with different numbers of nodes in the second layer to calculate the loss curves as shown in figure 2 below. When 10 hidden nodes (right plot in figure 2) were used in the second hidden layer, the model reached the point of zero loss (overfitting) by approximately epoch 3000 and reached testing accuracies of approximately 75 percent. When 6 hidden nodes (left plot in figure 2) were used in the second hidden layer, the model never reached a 0 loss and produced fluctuating testing accuracies over 80 percent (roughly similar to the shallow model). Given that overtraining is a real possibility with a more complex model, we held the number of epochs to 3000 for the deep neural network as well.

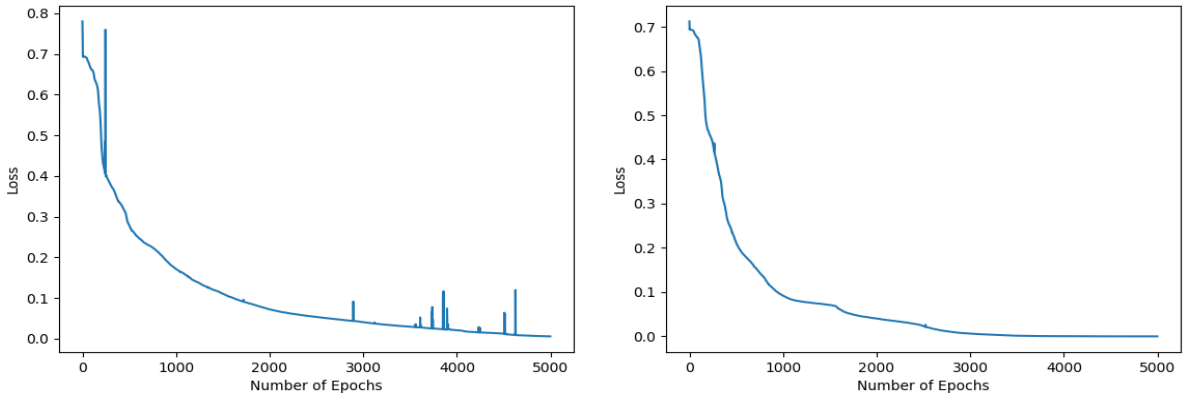

Fig. 2. Total loss value plotted during a training cycle of 5000 epochs for six nodes (left) and ten nodes (right) in hidden layer 2.

Given that the shallow and deep models performed somewhat similarly, this gave further evidence that increasing the number of layers further was likely creating an overly-complex architecture for the size of the dataset and the limited number of input variables. When attempted, these deeper models converged to 100 percent training accuracy very quickly (0 loss) and resulted in lower testing accuracies due to the overfitting.

Analysis was performed on the models using the state dictionary function in PyTorch to manipulate specific weights within the network. This is explained further within the results section.

3 Results

3.1 Rules Based Determination

In order to gain an understanding of the most influential input variables when creating rules for the network, a decision tree has been trained using scikit-learn's decision tree classifier in python. The decision tree was given a random 80/20 data split, and the max depth was set to 5 (increasing this further had deleterious effect). It was found that accuracies over 80 percent were achievable with the decision tree classifier, depending on the randomised choice of the train/test split. These simple trees relied very heavily on the PCAd1 and PCAd2 variables as shown in figure 3 below. Showing that the PCAd1 and PCAd2 input variables provided the most overall information and predictive power, while also indicating that the neural network would likely weight these variables with more importance.

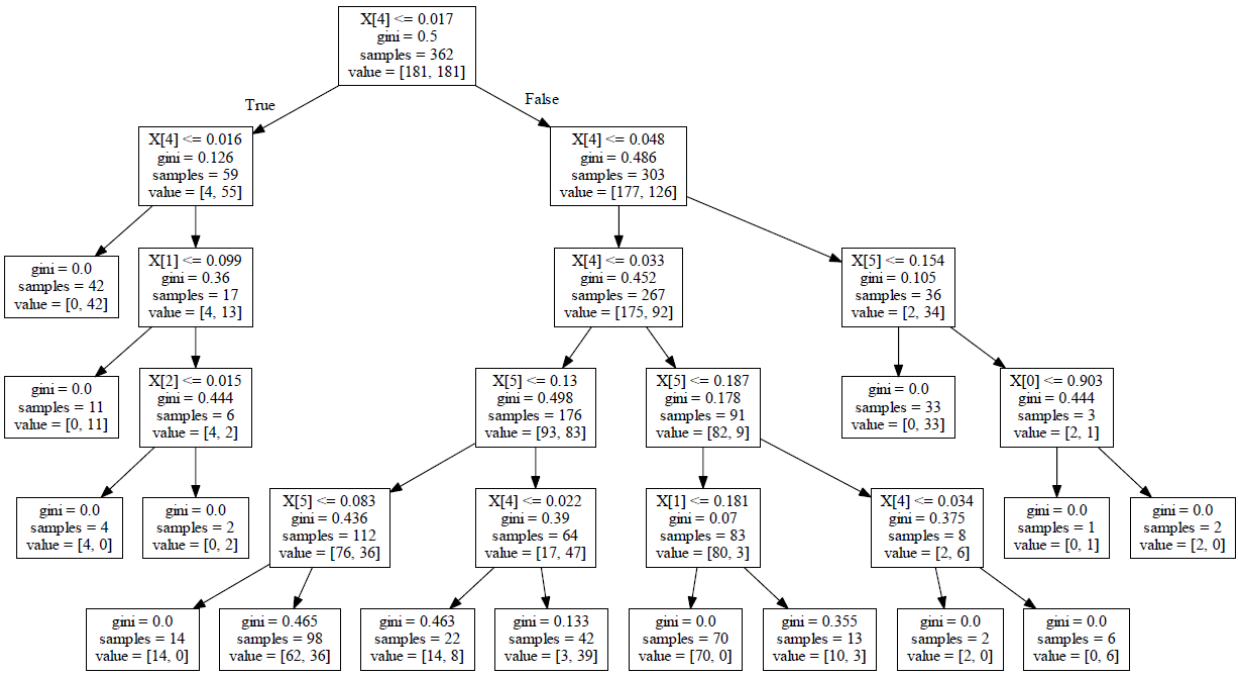

Fig. 3. Simple decision tree generated in Python using Scikit-Learn with a max depth of 5.

The diff1 and mean input variables also appeared in the decision tree rulesets once, and the std input variable was used twice. This shows that these other features in the dataset should offer finer predictive power as the model becomes more complex. It should be noted that logical derivation of rules like this has limits in producing compact rule sets, as complexity grows exponentially for the number of inputs [1]. Therefore, we may not be able to detect rulesets to the same degree of complexity as a neural network here.

3.2 Sensitivity Analysis and Pruning

3.2.1 Shallow Network

As shown in 3.1, PCAd1 and PCAd2 were identified as clear candidates for sensitivity analysis to gain insight into the neural network's logic. Pruning the network at the hidden layer to output layer interface would give us little information as to which inputs were influencing the network most greatly. Therefore sensitivity analysis and pruning efforts were focused onto the input layer to hidden layer interface.

The model was trained as described in the method with a test accuracy of 84.15 percent. By altering the weights in the network and utilising the test data, we were able to gain insight into the network's logic without retraining the model. The weights in the network corresponding to a specific input variable were set to zero, the testing accuracy and confusion matrix were measured again, and the weights were reset (to the original state after training) before repeating the process for the next input variable. The results are displayed in the table 1 below where a true positive (TP) indicates a correctly predicted genuine anger emotion, a true negative (TN) indicates a correctly predicted posed anger

emotion, a false positive (FP) indicates an incorrectly predicted genuine anger emotion and a false negative (FN) indicates an incorrectly predicted posed anger emotion.

Input Variable (weights to zero)	Test Accuracy	TN	FN	TP	FP
N/A (Baseline)	84.15	34	6	35	7
Mean	60.98	34	25	16	7
Std	80.49	29	4	37	12
Diff1	65.85	33	20	21	8
Diff2	60.98	23	14	27	18
PCAd1	50.00	0	0	41	41
PCAd2	50.00	41	41	0	0

Table 1. Summarised shallow network results after setting specified input variable's weights to zero. Test accuracy and counts for true negatives (TN), false negatives (FN), true positives (TP) and false positives (FP) are included for each input variable.

We saw a strong dependence of the network on the input variables PCAd1 and PCAd2, which reinforced previous findings. It appeared that if the network was trained on the whole dataset and subsequently lost either the PCAd1 or PCAd2 input information during testing, the network would simply predict an output of all genuine emotion or all posed emotion respectively. This demonstrated that the first custom rule required in explaining the logic of the network would be to observe the PCAd1 or PCAd2 value and set a particular threshold. The mean and diff2 appeared to be the next input variables which indicated a role of significance within the networks' logic, both having a test accuracy of 60.98 percent. Diff1 and std appeared to be least important with test accuracies of 65.85 and 80.49 respectively. In order to explore sensitivity analysis further, the weights were altered instead of being pruned (set to zero).

When attempting to alter weights for the individual input variables, it was found that simply reducing the weights by specific numbers was ineffective. The result was that reducing the weights for the mean and diff2 inputs had the most effect on the network accuracy. However upon inspection, this was due to the magnitude of weights being lower in those inputs and therefore disproportionately affecting the network accuracy when reduced by a constant number. Instead it was chosen to divide the weights by two for each input variable. A similar experiment was undertaken as in table 1, where the weights were instead halved. This further distinguished the significance of each input variable. The results are summarised in table 2 below.

Input Variable (weights halved)	Test Accuracy	TN	FN	TP	FP
N/A (Baseline)	84.15	34	6	35	7
Mean	56.10	33	28	13	8
Std	80.49	33	8	33	8
Diff1	75.61	33	12	29	8
Diff2	85.37	37	8	33	4
PCAd1	50.00	5	5	36	36
PCAd2	51.22	41	40	1	0

Table 2. Summarised shallow network results after halving the specified input variable's weights. Test accuracy and counts for true negatives (TN), false negatives (FN), true positives (TP) and false positives (FP) are included for each input variable.

When focusing on the positive to negative ratio (or vice versa) of PCAd1 and PCAd2, it is apparent that the model reverted to predicting all posed labels (negatives) faster than it moves to predicting all genuine labels (positives). In other words, halving the weights for PCAd2 moved the network more quickly to a state of predicting a single label compared to halving PCAd1. This indicates that the most important rule for the network is to look at the PCAd2 value, as the most information was shown to be lost. PCAd1 remained the next most valuable indicator. Interestingly, the mean proved to be far more valuable than the remaining input variables reducing the test accuracy to 56.10 percent. Diff1 and Diff2 showed the conflicting evidence as to their relative significance (compared to table 1), and the std was shown still to offer the least significance to the network.

3.2.2 Deep Network

The deep neural network was trained and produced a test accuracy of 85.37 percent using the randomised test dataset. The same process was applied to this model, first removing the weights from each input variable completely and then halving the weights.

Input Variable (weights to zero)	Test Accuracy	TN	FN	TP	FP
N/A (Baseline)	85.37	37	8	33	4
Mean	68.29	34	19	22	7
Std	74.39	39	19	22	2
Diff1	79.27	41	17	24	0
Diff2	60.98	28	19	22	13
PCAd1	50.00	1	1	40	40
PCAd2	50.00	41	41	0	0

Table 3. Summarised deep network results after setting specified input variable's weights to zero. Test accuracy and counts for true negatives (TN), false negatives (FN), true positives (TP) and false positives (FP) are included for each input variable.

When the weights were set to zero for each variable independently, the same relationship was shown as in the shallow network. The deep network depended heavily on the input variables PCAd1 and PCAd2, reinforcing findings that the network would not operate at above random accuracy without PCAd1 or PCAd2. Std and Diff1 appeared to be the least important input variables here again, albeit with the test accuracy being affected less for Diff1 instead of Std this time. This demonstrates that the network logic previously deduced also holds for a more complex neural network, and is likely vital to any producible model over the dataset. Interestingly, the deep neural network may be attributing a greater importance on the right eye as compared to the left eye (when looking at Diff1 and Diff2).

Input Variable (weights halved)	Test Accuracy	TN	FN	TP	FP
N/A (Baseline)	85.37	37	8	33	4
Mean	70.73	36	19	22	5
Std	78.05	36	13	28	5
Diff1	78.05	39	16	25	2
Diff2	76.83	36	14	27	5
PCAd1	54.88	10	6	35	31
PCAd2	56.10	39	34	7	2

Table 4. Summarised deep network results after halving the specified input variable's weights. Test accuracy and counts for true negatives (TN), false negatives (FN), true positives (TP) and false positives (FP) are included for each input variable.

When halving the weights for each input variable, we can again see that the deep neural network moved more quickly to a state of predicting a single label (negative or positive) when the PCAd2 variable was halved. This Indicates that the PCAd2 input still takes slight precedence over PCAd1. However, it is clear that the PCAd1 and PCAd2 variables are slightly less important here as the test accuracies do not drop all the way to approximately 50 percent. Therefore, the more complex deep neural network is likely leveraging the other input variables to a fractionally higher extent than the simple shallow neural network. Again the mean proved to be the next most valuable input variable to the network. The major difference between the shallow and neural network was the Std, Diff1 and Diff2 inputs all showing a very similar level of contribution to the network.

3.3 Characteristic Input Patterns

As there is one characteristic input pattern produced from a network's training set per distinctive output [1], there were only two characteristic input patterns available for us to identify. As performed by Gedeon and Turner, compressed representations of the dataset were created to aid in identifying the characteristic input patterns [1]. Given that we had the test set available to the same model trained in section 3.2 and an observed hierarchy of input variable significance (PCAd2, PCAd1 and mean (in order)), we could create compressed representations with the goal of further deducing the logical patterns of the network and improving the test accuracy.

3.3.1 Shallow Network

With the existing test dataset, we altered the data to determine the effect on test accuracy. As expected, removing data based on extreme values of PCAd2 had a noticeable effect on accuracy. Removing PCAd2 values less than 0.096 (14 genuine and 7 posed) and greater than 0.16 (1 genuine and 4 posed) from the test data set, reduced testing accuracy by 2.18 and 1.03 percent respectively while maintaining a large sample size in the test dataset. This supported the idea that the network looks at PCAd2 early on for logical deduction.

Removing PCAd1 values less than 0.019, resulted in the identification of 1 genuine and 9 posed labels. However when removed from the test dataset, the accuracy of the network rose by 0.28 percent indicating little change. This is potentially due to some of these data points being removed via earlier rules in the network's logic, and therefore not greatly affecting the network's accuracy when removed. When applying a greater than threshold to PCAd1, it was

difficult to obtain a non-symmetrical return of posed and genuine labels. However, it was found that they had a very large impact on the testing accuracy. For example, removing PCAd1 values greater than 0.027 (24 genuine and 20 posed labels) reduced the testing accuracy by 7.83 percent. Initially this was a surprising outcome, potentially flagging PCAd1 as higher in the logic hierarchy than PCAd2. However when combined with the resultant values of the rule: PCAd2 less than 0.096 (shown earlier), this rule set for PCAd1 and PCAd2 was able to uncover 8 genuine and 0 posed labels. This exemplifies a potential set of chain logic applied by the network, where PCAd2 is highest in the hierarchy.

The mean value was also tested as it was the next most influential variable on the network. Removing mean values at the lower extremes made little difference to testing accuracy and did not provide meaningful difference in the number of posed or genuine labels. There also remained to be a fairly even number of genuine and posed labels when removing mean values at the upper extreme; however the testing accuracy dropped noticeably. For example, at mean values greater than 0.95 we obtained 7 genuine and 9 posed labels with a fall in accuracy of 2.33 percent. Using this, the following rules were generated: mean > 0.95 AND PCAd2 > 0.096 AND PCAd1 > 0.027 (returned 7 posed and 3 genuine labels), as well as mean > 0.95 AND Using PCAd2 < 0.096 AND PCAd1 > 0.027 (returned 1 genuine label).

3.3.2 Deep Network

Removing PCAd2 values less than 0.096 (14 genuine and 7 posed) and greater than 0.16 (1 genuine and 4 posed) from the test data set, reduced testing accuracy by 4.97 and increased testing accuracy by 1.64 percent respectively while maintaining a large sample size in the test dataset. This further supported the idea that either network will look at PCAd2 early on for logical deduction.

When removing PCAd1 values less than 0.019 (1 genuine and 9 posed labels) from the test dataset the test accuracy was reduced by 3.83 percent, and removing PCAd1 values greater than 0.027 (24 genuine and 20 posed labels) reduced the test accuracy by 33.7 percent. This was a very similar outcome to the shallow network whereby restricting the upper tail of PCAd1 had a very large unexpected effect on the test accuracy. Offering increased support to the chain logic of PCAd2 less than 0.096 and PCAd1 greater than 0.027 (giving 8 genuine and 0 posed labels), and showing that PCAd2 remains the more useful first input here.

Testing the deep neural network while removing the mean values at the low extremes offered little insight into the model, with testing accuracies varying by 0 – 0.6 percent. When removing mean values at the high extreme, accuracy reductions up to 2 percent were seen. This falls in line with deductions from the shallow neural network, and strengthens the hypothesis of the rule sets formed previously.

3.4 Applying Generated Rules to Dataset

The rule sets deducted in section 3.3 were applied to the entire dataset. The results are summarised in the table below. We see that the bulk of the predictions existed at approximately 85 percent accuracy (as measured for the most returned label), with an approximately equal amount on either side (36 predictions at 100 percent accuracy and 35 predictions at 66 percent accuracy).

Rule Set	Genuine Count	Posed Count	Accuracy %
PCAd1 < 0.019	14	78	85
PCAd1 > 0.027 AND PCAd2 < 0.096	33	0	100
Mean > 0.95 AND PCAd2 < 0.096 AND PCAd1 > 0.027	3	0	100
Mean > 0.95 AND PCAd2 > 0.096 AND PCAd1 > 0.027	12	23	66

Table 3. Summarised rule sets generated for both neural networks, including label counts and accuracy.

4 Discussion

As shown in section 3.2, the sensitivity analysis provided an observed hierarchy of input variables when trying to approximate logical rules the network was following. The hierarchy found was found to be (in order of highest to lowest influence): PCAd2 > PCAd1 > mean > diff1 ~ diff2 > std. Using the order of the hierarchy, compressed representations of the testing dataset were created (while maintaining the same trained network weights) to generate rules for the model in section 3.3. These rules are seen above in table 3. While two of these rules identified genuine labels with 100 percent accuracy, they did so with small sample sizes (3 and 33 respectively out of 400 data points). This demonstrates a limitation of the experiment, in that a larger dataset is required to generate more complex rule sets

that return a meaningful number of results. The small size of the dataset also affects the reproducibility of this experiment, as the results may differ slightly if the network is retrained. Note that a much larger dataset would potentially require a more complex neural network architecture, therefore reproducing the manual parts of this experiment would become exponentially more difficult. An automated or algorithmic based approach would be more suitable to larger datasets. Overall it was shown that rules could be generated to represent close to half of the dataset with approximately 85 percent accuracy. This result is similar to the test accuracies generated by the shallow and deep neural networks developed (84.15 and 85.37 percent respectively), indicating that the ruleset partially represents the network's logic. Given that we are able to predict the validity of facial displays of anger at above 80 percent accuracy with rule sets, it is clear as to how neural networks are able to outperform humans (60 percent accuracy) in predicting genuine or posed facial displays of emotion. This suggests that humans are incapable of picking out all of the facial details, which provide strong predictive power over genuine or posed emotion.

We can also see from the results that the shallow and deep neural network largely agreed upon which input variables were most important in predicting the authenticity of the emotional display. There was a subtle difference within the deep neural network which showed a slightly lesser reliance on PCAd1 and PCAd2 as the go to input variables. This indicates that the model complexity is linked to the ability of the network to draw information from less obvious input variables; this is an expected and widely accepted trend. Extrapolating from this, one would assume that more hidden layers would allow us to gain insight into the model at levels beyond what we have achieved here. As shown in the results this is incorrect for a small dataset, we began to rapidly over fit to the data when introducing three or more hidden layers and produced lower test accuracies. This is supported by the literature, a study by Shuo Feng, et al. [6] showed that using deep neural networks over a small data set of 487 data points (similar size to that of our own) generated predictions at a lower accuracy than that of shallow neural networks when trained from scratch. However, by utilising pre-trained deep neural networks (over similar data), they were able to fine tune the deep models with the small dataset to outperform the accuracy of shallow neural networks [6]. This method in turn could be applied to our problem, if a suitable deep neural network (with a large number of hidden layers) pre-trained on detecting emotional authenticity could be sourced for future work. Additionally, an interesting find was that the deep neural network was beginning to attribute a larger focus on the responses of the pupil in the right eye over the left. This finding could be explored separately in future studies.

In the results by Gedeon and Turner [1], rule sets produced were capable of predicting the correct outcome in 94 percent of total cases. However comparing quantitative results over different experiments may not provide much insight, as different datasets allow for different prediction accuracies. Comparing the result of the rule set to the result of the model on the same data (as shown above) likely provides more information. Another study by Chen, R, et al. [7] provided both quantitative and qualitative results to explain exactly which features an image classifier is using to classify an image. This paper utilises larger neural networks, namely multilayer Convolutional Neural Networks, and investigates the neural activations of each filter separately, resulting in a highly accurate explanation of the networks logic throughout the layers. An approach similar to this would allow for significant improvements to the work performed in this paper, however this would require to extending the size of the dataset and require imagery to allow us to examine individual layers of the network logic. This would include imagery capturing pupillary responses and eye gaze.

Other studies have shown that neural networks can be effectively pruned to 10-20% of the original size and maintain equal or higher accuracy [8], lending some credit to the idea that simple rule sets can readily explain network logic for small models. However, it has also been shown that pruning or model compression is highly dependent on the data itself. In a study by Hooker, S, et al. [9] it was shown that while models with radically different numbers of weights have comparable accuracy after compression, the accuracies can diverge greatly when focusing on a narrow subset of the data. This demonstrated that while deep neural networks are generally highly tolerant of pruning, doing so may negatively impact prediction accuracy on the underrepresented tail ends of the data distribution. This potentially indicates that the sensitivity analysis and pruning approach taken for this experiment may not scale well to larger datasets with more output classes (non-binary output), especially if some of the desired outputs are not well represented within the dataset.

5 Conclusion and Future Work

In this paper a list of rules has been generated to approximate the logic of both a shallow and deep feedforward neural network. The rule sets generate predictions similar in accuracy to the neural networks' testing accuracy (84.15 percent for the shallow network and 85.37 percent for the deep network). These results demonstrate the hierarchy of importance placed on the input features by the network, and at which values the network may be identifying thresholds for clustering. This indicates that neural networks can, in part, be described by human-like rules and reasoning. It is also shown that simple rule sets are able to outperform human prediction accuracy (60 percent), therefore it is not surprising that neural networks are able to predict posed or genuine emotion more effectively than humans.

The experiment performed in this report could be improved upon by increasing the size of the dataset. This would allow for a larger test data set, and therefore more conclusive rule sets can be developed that include a greater number of examples. Other work has found that highly accurate qualitative and quantitative explanations of deep neural networks (specifically convolutional neural networks) are possible using imagery [7], therefore including imagery in future work may allow deduction of the distinct eye movements responsible for activating specific nodes of the network. It is also clear that utilising pre-trained deep neural networks to fine tune a model over small datasets, as performed by Shou Feng, et al. [6], could allow us to utilise deeper neural network architectures in future work. On top of this, deeper neural networks may allow us to further explore the highlighted importance of the pupillary response in the right eye over the left eye. Other techniques for sensitivity analysis such as Local Relative Sensitivity Index (LRSI) and modern pruning techniques including VNP, Xing-HU and N2PS [10] could be implemented, this may allow further rule sets to be uncovered but will also likely require a larger dataset to be effective.

References

1. Gedeon, T.D., Turner, H.S.: Explaining student grades predicted by a neural network. In: 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan), pp. 609-612 vol.1, Nagoya, Japan (1993). DOI 10.1109/IJCNN.1993.713989.
2. Chen, L., Gedeon, T.D., Hossain, Z., Caldwell, S.: Are you really angry? detecting emotion veracity as a proposed tool for interaction. In: 29th Australian Conference on Computer-Human Interaction (OZCHI 2017). Association for Computing Machinery, New York, NY, USA, 412–416 (2017). DOI https://doi.org/10.1145/3152771.3156147.
3. Chen, W.: Neural Networks to the Question: Are You Really Angry?. School of Computer Science and Engineering, Australian National University.
4. Riedmiller, M.: Rprop – Description and Implementation Details. University of Karlsruhe, Germany (1994).
5. Zhu, Q.: Predicted the authenticity of anger through LSTMs and three-layer neural network and explain result by causal index and characteristic input pattern. In: 3rd ANU Bio-inspired Computing conference (ABCs 2020), paper 83, 7 pages, Canberra (2020).
6. Feng, S., Zhou, H., Dong, H.: Using Deep Neural Network with Small Dataset to Predict Material Defects. In: Materials & Design, Volume 162, 15 January 2019, Pages 300-310. University of Leicester, Leicester, UK (2018). DOI https://doi.org/10.1016/j.matdes.2018.11.060
7. Chen, R., Che, H., Ren, J., Huang, G., Zhang, Q.: Explaining Neural Networks Semantically and Quantitatively. In: 2019 International Conference on Computer Vision (ICCV). Shanghai Jiao Tong University, China (2019).
8. Frankle, J., Carbin, M.: The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In: International Conference on Learning Representations (ICLR) 2019. Massachusetts Institute of Technology, United States (2019).
9. Hooker, S., Courville, A., Clack, G., Dauphin, Y., Frome, A.: What Do Deep Neural Networks Forget?. Google Brain, United States (2020). DOI https://doi.org/arXiv:1911.05248v2.
10. Augasta, M., Kathirvalavakumar, T.: Pruning algorithms of neural networks — a comparative study. In: Open Computer Science, vol. 3, no. 3, 2013, pp. 105-115. Department of Computer Applications, Sarah Tucker College, Tirunelveli, India (2013). DOI https://doi.org/10.2478/s13537-013-0109-x