Subjective Beliefs Recognition Using LSTM and Studying The Effect of Threshold Controlling

Siwei Ling
Research School of Computer Science, Australian National University
u7155777@anu.edu.au

Abstract. In this paper, I trained an LSTM model to predict subjective beliefs using a physiological signal dataset and proposed a simple method of combining threshold controlling and principle component analysis to reduce the number of false positive predictions while keeping the number of correct predictions as high as possible. I also ran a series of tests, compared the output of different neural network models and published results to prove that the neural network model and the method I proposed can effectively reduce the number of false positive predictions.

Keywords: Neural network, Deep Learning, LTSM, Classification, Threshold, Principle Component Analysis, False Positive Predictions

1 Introduction

Subjective beliefs plays an important role in human's daily activities, it can influence a person's mood and his or her decision making. To measure or predict it, however, proves to be rather difficult since its manifestation varies from person to person, and it is difficult to measure it with regular quantitative standards. In previous studies, many researchers tried to use neural networks to predict subjective beliefs given a person's physiological signals and achieved good results. But physiological signals are usually complicated and models often suffers from overfitting problems and yield a large percentage of false positive results.

In the case of a classification problem, a low percentage of false positive results is generally more desirable. It allows us to combine more evidence from multiple sources to achieve better results[1]. Also models can suffer from overfitting problems due to a large amount of features, by carefully adjusting the input data, we can also improve the overfitting problem, and further reduce the number of false positive predictions.

I would like to construct a neural network model and an LSTM model to classify subjective beliefs using the dataset gathered by Xuanying Zhu, Tom Gedeon and others[2] and use threshold controlling and principle component analysis to minimize false positive predictions while keeping correct prediction rate as high as possible in order to increase the reliability and accuracy of the neural network model.

2 Dataset

2.1 Dataset Overview

The datasets used in this paper was created by an experiment. The researchers first invited a few presenters and prepared some material for these researchers to present. For half of the presenters, the researchers told them that their material are bogus and asked them to still present them naturally in order to try and manipulate the presenter's subjective beliefs. The researchers recorded the presentation as videos and invited several observer to watch the videos. The observers are then asked to estimate whether the presenter's subjective belief was manipulated or not and some of their physiological signals such as Blood Volume Pulse (BVP), Galvanic Skin Response (GSR), Skin Temperature (ST) and Pupillary Dilation (PD) are recorded to form the dataset[2].

There are two datasets. The first dataset consists of sequential data gathered in the experiment. The dataset has three levels of directories. The first level of the directory is the participant's id and gender together with a csv file that contains labels for each participant-video pair, the second level of the directory are the ids of the videos that each participant watched, and the third level of the directory has csv files corresponding to BVP, GSR, ST and PD data along with the starting and ending time of each video. Each csv file has only one column and multiple rows. The first row is the initial time of the recording session expressed as unix timestamp in UTC and the second row is the sample rate expressed in Hz. The rest of the rows are data collected by sensors in chronological order, therefore, I will train an LSTM model to do predictions using this dataset.

The second dataset is a preprocessed version of the previous dataset, which constructs features using the sequential data in the previous dataset. It has 121 columns and 368 rows. Each row contains the physiological signals gathered from a specific observer when they are watching a specific video. The first column contains a string that records the participant id and the id of the video this participant watched. Columns 2 to 35 contains 34 features extracted from the participant's BVP . Columns 36 to 58 contains 23 features extracted from the participant's GSR. Columns 59 to 81 contains 23 features extracted from the participant's ST. Columns 82 to 120 contains 39 features extracted from the participant's PD. And the last column contains the label of whether the presenter in this video has doubted about their belief in the video. 1 represents we didn't manipulate presenters' subjective belief, and 0 means they doubted about what they presented as we manipulated their belief. I will train a neural network with only fully-connected layers and a softmax layer to do predictions using this dataset.

2.2 Data Preprocessing

We will apply two different data preprocessing methods on the two datasets. In the first dataset, pupillary data for participants 26 to participant 34 are missing, and the starting time and ending time of each video does not align with the the starting time and ending time of the sensors. The sensor for BVP value has a sample rate of 64Hz while the other sensors has a sample rate of 4Hz, so there are significantly Also, each csv file contains data collected from two videos instead of one, so some necessary separation has to be done.

Therefore for the first dataset, I first selected the valid part of the BVP, GSR and ST values using the starting and ending time of the 2 videos. During this process, data in each csv file are separated into two parts based on the video, and all values that are collected by the senors outside of these time periods is discarded. The PT value is also discarded because pupillary data for participants 26 to participant 34 are missing. Then I duplicated the values in GSR and ST for 16 times due to the different sample rate, so that their shape matches the shape of the BVP data. In the next step, I removed several rows data that does not have a corresponding label. In the final step, I normalized the date so that all the values falls into the range between 1and 2. I did not normalize the data to 1 because the length of each data sequent is different and I later applied data padding with zeros.

For the second dataset, all the data except for the first and last column, are of type float and there is no missing data. Value within each column are pretty close to each other, and there is no significant outlier. Values between each column varied greatly, some are as big as 40, others are as small as $2x10^{-3}$. Therefore I applied min-max normalization to each column of the data. The preprocessed data is saved in a new csv file.

Also, the 119 columns of features can be categorized into 4 major types: blood volume pulse, galvanic skin response, skin temperature and pupillary dilation. Therefore, I applied 3 different principle component analysis process to each set of the4 columns, the 3 different principle component analysis process keeps 1, 2, and 3 components respectively.

During training, the most common cross-validation method would be to randomly shuffle the data and use k-fold cross-validation. But for human data, a continuous segment of physiological data with more than one data point can reflect a human's responses to a stimulus[2]. So training a classification model on random splits of data is not adequate, unless all data from one human is guaranteed to be within either the training set or the testing set for each run. Therefore, I used a method called "leave-one-participant-out" [2]. I used a slightly modified version which selects one presenter's

data as the test set, and use the rest of the data as the training set, repeated for all and averaging to calculate the final result reported. In comparison, the original researchers used each observer as the test set.[2]

3 Methodology

3.1 Previous Researches

Xuanying Zhu, Tom Gedeon and others used an ensemble of five artificial neural networks, each with a sigmoid hidden layer of size 100 and an output layer with two output neurons[2]. They trained the neural network with the Adam optimizer[3] using back propagation with the Cross Entropy loss function. L.K.Milne, T.D. Gedeon and A.K.Skidmore proposed that adjusting the threshold for the output layer can effectively manipulate the balance of false positive and false negative predictions[2].

3.2 Model Description

I will use 5 different artificial neural networks to do the classification, and then compare their results to each other. The baseline model uses the normalized data as input, it has 2 sigmoid hidden layers with 128 and 64 hidden neurons each and an output layer with 1 neuron and a sigmoid activation function. I used the Adam optimizer[3] and updated the weights using back propagation with Binary Cross Entropy loss function.

The LSTM model uses the preprocessed sequential data as input. It has a LSTM layer with 6 hidden neurons, and the output of the LSTM layer will be passed onto an output layer with 1 neuron and a sigmoid activation function. The output layer uses the output of the LSTM to do the prediction. I also used the Adam optimizer[3] and updated the weights using back propagation with Binary Cross Entropy loss function.

The other 3 models are the PCA_4 model, PCA_8 model and the PCA_12 model. PCA_4 model has 2 sigmoid hidden layers with 8 neurons and an output layer with 1 neuron and a sigmoid activation function. PCA_8 model has 2 sigmoid hidden layers with 16 neurons and an output layer with 1 neuron and a sigmoid activation function. PCA_12 model has 2 sigmoid hidden layers with 24 neurons and an output layer with 1 neuron and a sigmoid activation function. These 3 models also updates weights using back propagation, and uses the same optimizer and loss function as the baseline model.

There is a hyper parameter θ that controls the threshold of the 4 models' outputs. θ is a real number between 0 and 1, if the output is larger than θ, the models will predict 1, otherwise the models will predict 0. I tried several values between 0.4 and 0.7 for theta, and trained the model with different θs one by one.

The amount of data used in training the LSTM model is very large so I used a large batch size of 128 and trained the model for 10 epochs with learning rate a=0.005 and weight decay w=0.1. This is mainly due to the limited computing power of CPU and GPU, and the batch size and number of epochs should be adjusted if the computing power of CPU and GPU is better. I trained the other model for 30 epochs, with learning rate a=0.001 and weight decay w=0.1, and the batch size was set to 10.

4 Results and Discussions

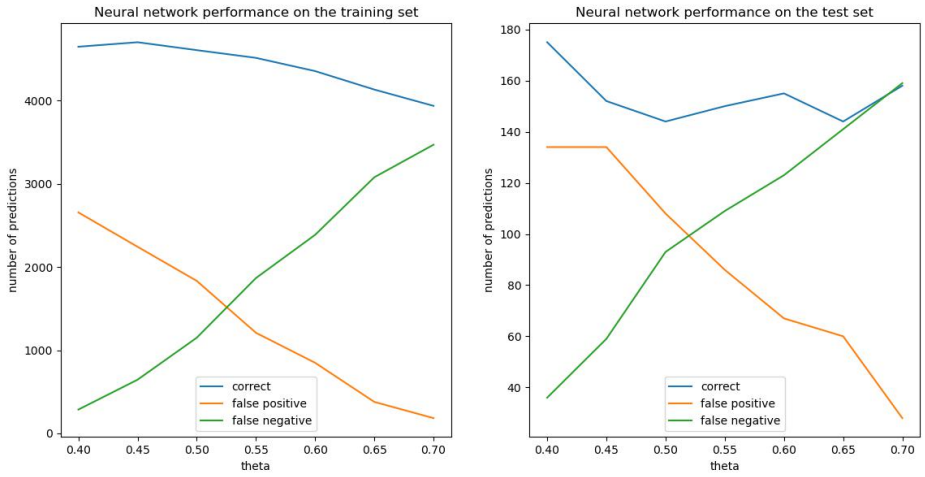
4.1 Effect of the Threshold

Figure 1 shows the training and testing accuracy of the baseline model using different thresholds. As θ gradually increases, the training accuracy gradually drops and the test accuracy goes up and down. The dataset has 119 features, but only 367 samples, therefore cause the model to overfit the training data, and the test accuracy becomes inconsistent.

theta	0.40	0.45	0.50	0.55	0.60	0.65	0.70
Training accuracy	61.24%	61.95%	60.70%	59.47%	57.38%	54.45%	51.87%
Test accuracy	50.73%	44.06%	41.74%	43.48%	44.93%	41.74%	45.80%

Fig. 1. Different training accuracy and test accuracy under different threshold conditions of the baseline model.

Figure 2 shows the number of correct predictions comparing to false positive predictions and false negative predictions of the baseline model. We could clearly see that as θ increases, the number of correct predictions and the number of false positive predictions both drops, and the model begins to produce more false negative predictions, which is exactly what we wanted.



Neural network performance on training set			
theta	correct	false positive	false negative
0.40	4648	2656	286
0.45	4702	2243	645
0.50	**4607**	**1833**	**1150**
0.55	4514	1209	1867
0.60	4355	848	2387
0.65	4133	378	3079
0.70	**3937**	**183**	**3470**

Neural network performance on test set			
theta	correct	false positive	false negative
0.40	175	134	36
0.45	152	134	59
0.50	**144**	**108**	**93**
0.55	150	86	109
0.60	155	67	123
0.65	144	60	141
0.70	**158**	**28**	**159**

Fig. 2. The figure above shows how the balance of correct prediction, false positive predictions and false negative predictions changes when θ changes for the baseline model. The figure below shows the exact number of each type of predictions for the baseline model.

We could see that on the training set, a threshold of 0.7 decreased the number of false positive predictions by almost 90%, while losing about 15% of accuracy comparing to a threshold of 0.5. A threshold of 0.6 decreased the number of false positive predictions by about 54%, while losing only about 6% of accuracy. This shows that if we choose a threshold smart enough, we could greatly reduce the number of false positive predictions while still keeping a relatively high accuracy.

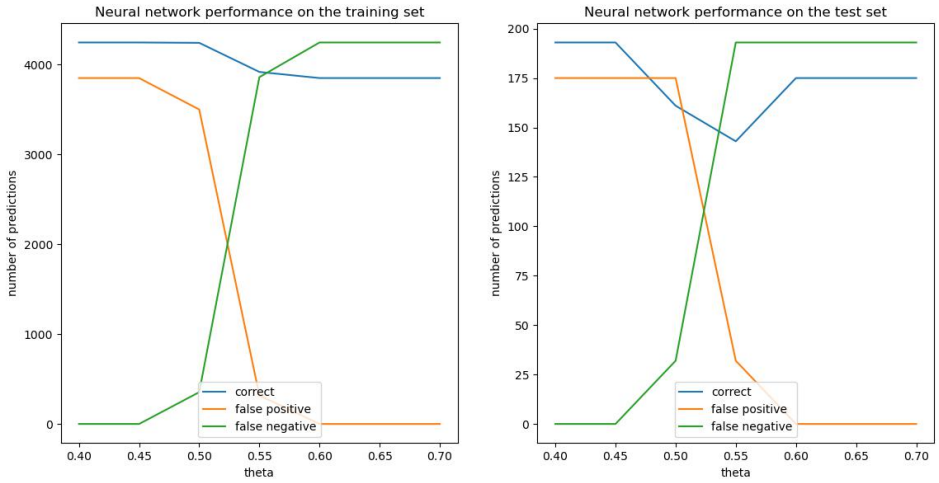
Figure 3 shows the training and testing accuracy of the LSTM model using different thresholds. As θ gradually increases, the training and testing accuracy gradually increased, reached a peak value when θ is 0.5 and then gradually drops. This is because when θ is larger or smaller than 0.5, we are actually using the threshold method to

control the number of false positive and false negative predictions, therefore we will lose some accuracy. The exceptionally high test accuracy when θ is 0.4 is probably caused by noise.

theta	0.40	0.45	0.50	0.55	0.60	0.65	0.70
Training accuracy	59.81%	60.24%	60.50%	58.92%	57.11%	55.35%	52.61%
Test accuracy	50.14%	48.21%	49.78%	47.42%	46.14%	47.46%	45.80%

Fig. 3. Different training accuracy and test accuracy under different threshold conditions of the LSTM model.

Figure 4 shows the number of correct predictions comparing to false positive predictions and false negative predictions of the LSTM model. As θ increases, the number of correct predictions and the number of false positive predictions both drops, and the model begins to produce more false negative predictions, similar to the results of the baseline model. When θ is larger than or equal to 0.6, the number of false positive predictions dropped to 0, as desired.



Neural network performance on training set			
theta	correct	false positive	false negative
0.40	4246	3850	0
0.45	4246	3850	0
0.50	4242	3500	354
0.55	3918	318	3860
0.60	**3850**	**0**	**4246**
0.65	3850	0	4246
0.70	3850	0	4246

Neural network performance on test set			
theta	correct	false positive	false negative
0.40	193	175	0
0.45	193	175	0
0.50	161	175	32
0.55	143	32	193
0.60	**175**	**0**	**193**
0.65	175	0	193
0.70	175	0	193

Fig. 4. The figure above shows how the balance of correct prediction, false positive predictions and false negative predictions changes when θ changes for the LSTM model. The figure below shows the exact number of each type of predictions.

By comparing the LSTM model and the baseline model, we can prove that the threshold method can work on both a simple neural network model and other more complicated deep learning models. Also, the threshold controlling method is much more effective on the LSTM model, since the number of false positive predictions can be successfully reduced to zero when minor changes are applied to the threshold.

4.2 Effect of Principle Component Analysis

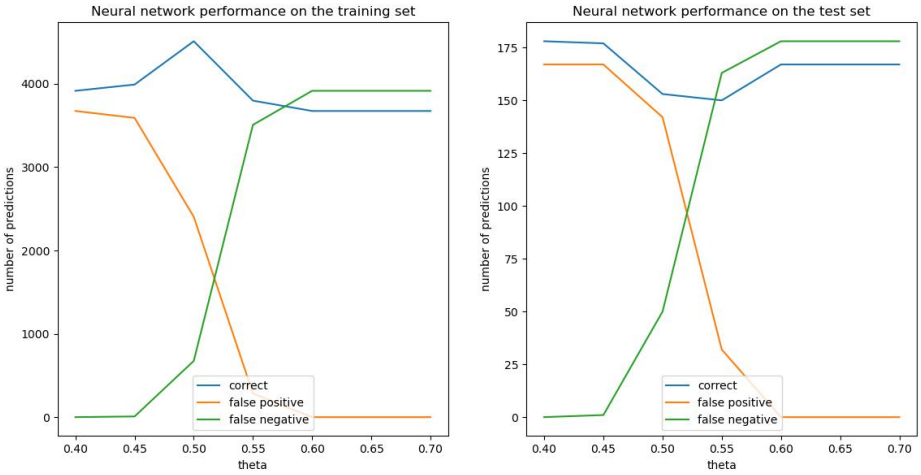
Figure 5 shows the the training and testing accuracy of the PCA_8 model using different thresholds. Figure 6 shows the number of correct predictions comparing to false positive predictions and false negative predictions of the PCA_8 model.

Comparing to the result of the baseline model, the overall training accuracy dropped significantly, but the test accuracy increased slightly, especially when θ is greater than or equal to 0.6. This is because principle component analysis reduces the number of features in the original dataset, and therefore the model is less likely to suffer from overfitting problems.

Also, notice that when $\theta = 0.6$, the number of false positive predictions in the test set dropped to 0, and the number of correct predictions only dropped by about 6% comparing to $\theta = 0.4$. Even greater improvements could be seen when comparing $\theta = 0.6$ to the baseline model. The number of false positive predictions dropped significantly, and the number of correct predictions increased slightly. This could suggest that when dealing with dataset that has a relatively large amount of features, principle component analysis can enhance the effect of threshold control, making the model even more reliable and accurate.

theta	0.40	0.45	0.50	0.55	0.60	0.65	0.70
Training accuracy	51.59%	52.58%	59.43%	50.04%	48.41%	48.41%	48.41%
Test accuracy	51.59%	51.30%	44.35%	43.48%	48.41%	48.41%	48.41%

Fig. 5. Different training accuracy and test accuracy under different threshold conditions of the PCA_8 model.



Neural network performance on training set			
theta	correct	false positive	false negative
0.40	3916	3674	0
0.45	3991	3591	8
0.50	4511	2404	675
0.55	3798	284	3508
0.60	**3674**	**0**	**3916**
0.65	3674	0	3916
0.70	3674	0	3916

Neural network performance on test set			
theta	correct	false positive	false negative
0.40	178	167	0
0.45	177	167	1
0.50	153	142	50
0.55	150	32	163
0.60	**167**	**0**	**178**
0.65	167	0	178
0.70	167	0	178

Fig. 6. The figure above shows how the balance of correct prediction, false positive predictions and false negative predictions changes when θ changes for the PCA_8 model. The figure below shows the exact number of each type of predictions.

4.3 Effect of Different PCA results

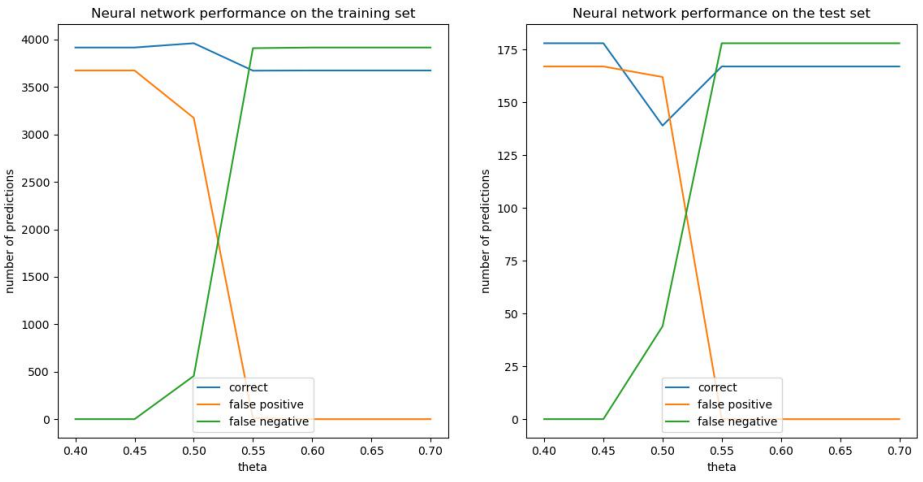
Further more, I studied the effect of different PCA results on the output of the model. Figures 7 and 8 shows the result of the PCA_4 model which uses only 1 principle component, and Figures 9 and 10 shows the result of the PCA_12 model which uses 3 principle components. In comparison, PCA_8 model uses 2 principle components and the baseline model used all the components(without PCA).

We could see that if we use more principle components, the training and testing accuracy of the model with smaller θ s will tend to increase, but it requires larger θ s to effectively reduce the number of false positive prediction. If we use less principle components, the training and testing accuracy of the model with smaller θ s will tend to decrease, but we could more effectively reduce the number of false positive predictions while maintaining the number of correct predictions.

Also, note that when applying PCA and reduce the number of features to a limited number, if θ reaches a value that is large enough, the number of false positive predictions will eventually reach 0 and we will get a fixed accuracy (in this case it is 48.41%). This means that we should not always aim for large θ s or large number of features, instead we should be more flexible and adaptive when choosing the appropriate PCA result and θ.

theta	0.40	0.45	0.50	0.55	0.60	0.65	0.70
Training accuracy	51.59%	51.59%	52.19%	48.41%	48.41%	48.41%	48.41%
Test accuracy	51.59%	51.59%	40.29%	48.41%	48.41%	48.41%	48.41%

Fig.7. Different training accuracy and test accuracy under different threshold conditions of the PCA_4 model.



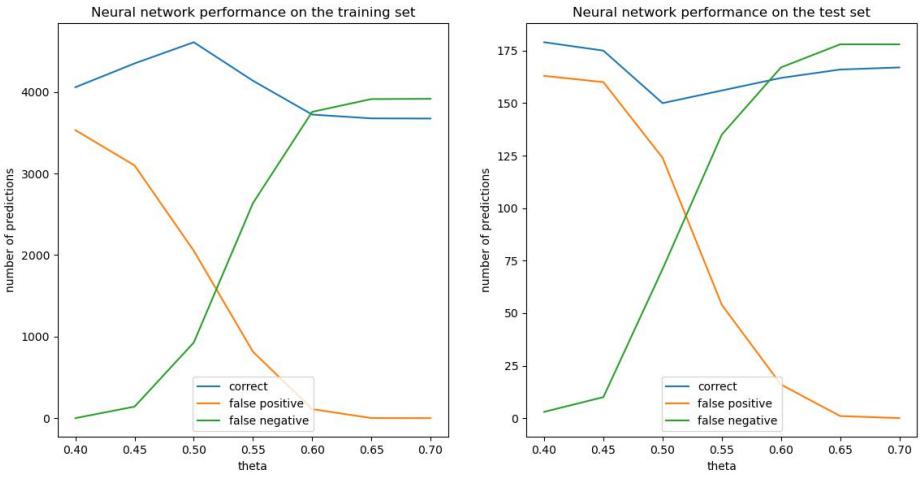
Neural network performance on training set			
theta	correct	false positive	false negative
0.40	3916	3674	0
0.45	3916	3674	0
0.50	3961	9	455
0.55	**3672**	**0**	**3909**
0.60	3674	0	3916
0.65	3674	0	3916
0.70	3674	0	3916

Neural network performance on test set			
theta	correct	false positive	false negative
0.40	178	167	0
0.45	178	167	0
0.50	139	162	44
0.55	**167**	**0**	**178**
0.60	167	0	178
0.65	167	0	178
0.70	167	0	178

Fig. 8. The figure above shows how the balance of correct prediction, false positive predictions and false negative predictions changes when θ changes for the PCA_4 model. The figure below shows the exact number of each type of predictions.

theta	0.40	0.45	0.50	0.55	0.60	0.65	0.70
Training accuracy	53.47%	57.31%	60.74%	54.52%	49.05%	48.43%	48.41%
Test accuracy	51.88%	50.73%	43.48%	45.22%	46.96%	48.12%	48.41%

Fig.9. Different training accuracy and test accuracy under different threshold conditions of the PCA_12 model.



Neural network performance on training set			
theta	correct	false positive	false negative
0.40	4058	3532	0
0.45	4350	3099	8
0.50	4610	2053	675
0.55	4138	815	3508
0.60	3723	111	3916
0.65	**3676**	**1**	**3916**
0.70	3674	0	3916

Neural network performance on test set			
theta	correct	false positive	false negative
0.40	179	163	3
0.45	175	160	10
0.50	150	124	71
0.55	156	54	135
0.60	162	16	167
0.65	**166**	**1**	**178**
0.70	167	0	178

Fig. 10. The figure above shows how the balance of correct prediction, false positive predictions and false negative predictions changes when θ changes for the PCA_12 model. The figure below shows the exact number of each type of predictions.

5 Conclusion and Future Work

I have studied the effects of applying different thresholds on the output layer and applying principle component analysis when using sigmoid activation function to do classification problems. In this paper I demonstrated that by applying an appropriate threshold on the output layer, we can significantly reduce the number of false positive predictions, while maintaining the number of correct predictions to be as high as possible. I have also found evidence that applying principle component analysis on dataset with large number of features can prevent the model from overfitting and enhance the effect of threshold controlling.

This method can also be used in some other tasks such as cancer detection. If the result of a cancer detection is false negative, it will delay the best treatment time, and the cancer will develop from the early stage to the late stage, which is unacceptable. In this case we can use the threshold controlling method to reduce the number of false negative predictions and increase the number of false positive prediction.

Some improvements can also be made to the LSTM model. In this paper, I only used the output of the LSTM model after all the sequential data has been processed, but we can actually make use of the outputs at each hidden state to continuously keep track of the presenter's subjective belief. For example, we can extend the dataset by including the specific parts of the presentation where the presenter is most suspicious about and use the output of each hidden state in the LSTM to try and predict the presenter's subjective belief at each time step. This model can be used to dynamically track a person's subjective belief over time.

There are also several drawbacks on this method that needs to be pointed out. The overall accuracy of the LSTM model and the other neural network model are all pretty low, at around 48% - 50%. One possible explanation is that the model I used in this paper is relatively simple and hyper-parameters are not optimized enough. A way to achieve a higher prediction accuracy is that we can use a pre-trained model and fine tune the model using our dataset, and then we can test the threshold controlling method on the new model. Also, if the prediction accuracy increases by a lot, the effectiveness of the threshold controlling method could possibly decrease, and further research on this needs to be conducted.

References

1. Milne, LK & Gedeon, Tom & Skidmore, AK. (1995). Classifying Dry Sclerophyll Forest from Augmented Satellite Data: Comparing Neural Network, Decision Tree & Maximum Likelihood.
2. Zhu, Xuanying & Qin, Zhenyue & Gedeon, Tom & Jones, Richard & Hossain, Md & Caldwell, Sabrina. (2018). Detecting the Doubt Effect and Subjective Beliefs Using Neural Networks and Observers' Pupillary Responses: 25th International Conference, ICONIP 2018, Siem Reap, Cambodia, December 13-16, 2018, Proceedings, Part IV. 10.1007/978-3-030-04212-7_54.
3. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv Preprint arXiv1412.6980 (2014)
4. Chow, C., Gedeon, T.: Classifying document categories based on physiological measures of analyst responses. In: 2015 6th IEEE International Conference on Cognitive Info communications (CogInfoCom), pp. 421–425 (2015)
5. Wu, Danyang & Zhang, Han & Nie, Feiping & Wang, Rong & Yang, Chao & Jia, Xiaoxue & Li, Xuelong. (2020). Double-Attentive Principle Component Analysis. IEEE Signal Processing Letters. 27. 1814-1818. 10.1109/LSP.2020.3027462.
6. Ghaffari, Mehdi & Farrokhi, E.. (2008). Principle component analysis as a reflector of combining abilities.
7. Itoh, Hayato & Imiya, Atsushi & Sakai, Tomoya. (2015). Low-Dimensional Tensor Principle Component Analysis. 9256. 10.1007/978-3-319-23192-1_60.
8. Gers, Felix A., Jurgen Schmidhuber, and Fred Cummins. "Learning to Forget: Continual Prediction with LSTM." Neural Computation 12.10 (2000): 2451-2471.
9. Rao, Guozheng, et al. "LSTM with sentence representations for document-level sentiment classification." Neurocomputing (2018): 49-57.