

Audio Emotion Recognition with Noise Reduction

Yucheng Bian, Lei Gao, and Yuepeng Shen

University of California San Diego, La Jolla, CA 92093-0238

Abstract—In this paper, we analyze and predict the emotion of speech audios after noise reduction. We first reduce noise in speech signals by applying spectral subtraction. Then we extract Mel-Scale Frequency Cepstral Coefficients(MFCC) features, which is recently the dominant feature for speech recognition. We use a convolutional neural network(CNN) and a long short term memory network(CNN-LSTM) to train the audios. Finally, to make the model more efficient in practice, we customize Dense and Conv2D layers via pruning neurons. We show that the prediction time decreases while accuracy remains at the same level.

I. INTRODUCTION

Traditional approaches to distinguish emotions are usually involved with text-mining. Text-based method relies heavily on the content of the speech, including analyzing word frequency or TFIDF, etc. However, when dealing with audio-based data, text content is sometimes misleading. The emotion of an audio depends not only on text, but also on emotional intensity, intonation, and some other subtle patterns contained in the data. Therefore, we extract MFCC features from audio files as inputs for our neural network models. MFCC of a signal is a small set of features which concisely describe the overall shape of a spectral envelope. It has been proved to be efficient in audio recognition. Deep learning neural networks are needed as a supervised learning method to learn every pattern from the data.

In this paper, the input to our algorithm is MFCC features from an audio. We then use a CNN and a CNN-LSTM to output a predicted emotion.

The general trend of CNN has been to make deeper and more complicated networks in order to achieve higher accuracy [1]. However these advances to improve accuracy are not necessarily making networks more efficient respect to size and speed. The large sizes consume considerable storage, memory bandwidth and computational resources. For embedded mobile applications, these resource demands become prohibitive [2], as we could see from figure 1.

Thus, we use pruning techniques to reduce the energy required to run large networks so they can run in real time on mobile devices. The model size reduction from pruning also facilitates storage and transmission of mobile applications.

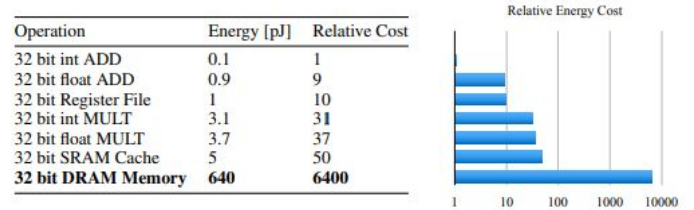


Figure 1: Energy table for 45nm CMOS process [2]. Memory access is 3 orders of magnitude more energy expensive than simple arithmetic.

II. RELATED WORK

A. Noise Reduction

Meaningless or corrupted data in the dataset is known as noise. Generally, noise can be categorized into additive noise and multiplicative noise. If it could not be handled well, noise could impact the model to some extent, for the model might think of noise as a pattern and try to generalize it. Therefore, it is essential to do noise reduction before next steps.

Noise reduction is the process of removing noise from a signal. For audio signals, there have been many ways to remove noise from the original signal. In 2013, Liu [7] and other researchers applied a least mean square adaptive filter to do noise reduction in audio files. After processing, the division of the variance of noise from noise canceled signal and noisy signal equals 0.008. In 2000 [8], M. Fujimoto proposed a noise reduction method based on the Kalman filter for noisy speech recognition. They carried out experiments to extract clean speech from noisy data and get over 90 SNR after processing. In 1996, Mwema [9] proposed a spectral subtraction method to reduce noise in speech signals. They evaluate their model by comparing the Euclidean distance of clean signal and noise canceled signal. And find the best model parameter to train the speech data. In our paper, we mainly used this method to do noise reduction before training our models.

B. Emotion Recognition

Emotion Recognition is a big area contained in audio recognition. To find out the emotion hidden in the audio, firstly, we have to find a way to gather the features which can represent the audio and help the model to understand what it expresses. MFCCs have been the dominant features used for

speech recognition. It has the ability to represent the speech amplitude spectrum in a compact form and has been proved that it is a good feature to explore the audio.

After that, people choose different models to classify the audio. At the beginning, researchers chose traditional speech recognition methods, like hidden Markov models. Recently, with the appearance of deep learning models, researchers turned to choose these models to explore the audio. A CNN-based SER method has been proposed that learns salient features of SER using semi-CNNs [3]. Besides, the SER system using RNNs was proposed in [4]. These researches proposed lots of ways to solve the problem. And we will propose the CNN-LSTM model based on their thoughts. The CNN-LSTM model is the state-of-the-art model, in this project, we plan to implement this model by ourselves and compare it to CNN models.

C. Pruning Techniques

In the paper [2], the author presented a method to prune network connections in a manner that preserves the original accuracy. After an initial training, the connections whose weights are lower than a threshold will be removed. In our paper, we implement this method by adding a mask variable to Conv2D and Dense layers and removing neurons that have lowest absolute weights.

III. DATASET AND FEATURES

In this part, we'll introduce the dataset we used and our preprocess steps in detail.

A. Dataset

Our dataset generally used "The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)" by Livingstone & Russo. This portion of RAVDESS contains totally 1440 files. There are 12 male and 12 female actors, and each actor performs 60 trials. Speech emotions include calm, happy, sad, angry, fearful, surprise, and disgust expressions. And each expression is produced at two levels of emotional intensity (normal, strong), with an additional neutral expression. In our next steps, we will use different methods to distinguish emotions between different files.

B. Noise reduction

In our paper, we applied two methods to reduce noise of speech signals.

- 1) Design low pass filters to reduce noise of high frequency. To be more detailed, as most of our the frequency of our signals lies beyond 4k Hz, we applied a 4k low pass filter to reduce high-frequency noise.
- 2) Applied spectral subtraction to our dataset, the detailed steps show as follows.
 - a) Transform the time-domain signal to frequency-domain by using FFT.

- b) Calculate the mean frequency of the first 10 frames of signal as noise frequency.
- c) Compare the frequency of each frame with the noise frequency, and then update both the frame value and noise value.
- d) Transform the frequency-domain signal to time-domain by using IFFT and get the data after noise reduction.

C. Feature extraction

Feature extraction is one of the most important steps in Recognition assignment. The audio signal is an 1-D array or a time sequence. If we want to explore the emotions within it, we have to use some methods to extract the proper feature to classify the emotions. The most common feature people used to explore audio signals is MFCC(Mel-scale Frequency Cepstral Coefficients).

To calculate the MFCC, we have to :

- 1) Frame the signal into short frames;
- 2) Calculate the FFT(Fast Fourier Transform) of each audio signal frame and get the spectrum;
- 3) Apply the Mel filter on the signal and get the Mel spectrum;
- 4) Take the logarithm and calculate DCT(Discrete Cosine Transform) of the signal to calculate the MFCC.

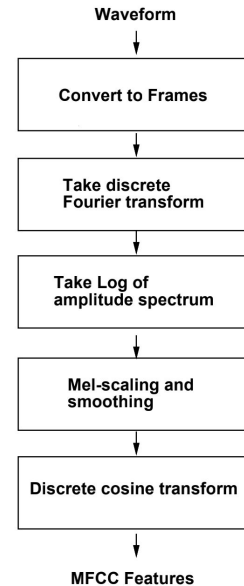


Figure 2: Process to create MFCC features

In this algorithm, we use FFT and DCT to process the original signal. The FFT (Fast Fourier Transform) is a Discrete Fourier Transform algorithm which reduces the number of computations needed for N points from $2N^2$ to $2N \lg N$. DCT (Discrete Cosine Transform) expresses a finite sequence of data points in terms of a sum of cosine functions oscillating at different frequencies.

IV. METHODS

In this part, we will describe the deep learning algorithms including CNN, RNN, the merge of CNN and RNN and so on. We will briefly present the architecture and how they work in this section.

A. CNN (Convolutional Neural Network)

CNN is a class of deep learning models and in most cases, it is used to analyze the image. Usually, it consists of input layers, convolutional layers, activation layers, pooling layers and dense layers. Figure 3 is a good example to show the architecture of CNN.

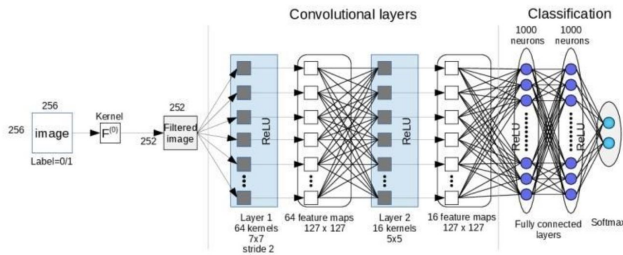


Figure 3: Architecture of CNN

When the image passes through a convolutional layer, the image is transferred to several feature maps. Convolutional layers convolve the image, a two-dimensional array, and pass it to the following layer. The convolution operation reduces the size of the parameters and it works well on extracting features from the input. Besides, Convolutional networks may include pooling layers to streamline the computation. This kind of layer can reduce the dimensions of the data by combining some output into a single neuron in the next layer. These two layers make CNN different from other kinds of deep learning networks and help CNN analyze the image. In our project, CNN is used to deal with the MFCC feature, which is a 2-D array.

B. RNN (Recurrent Neural Network)

RNN is also a class of deep learning networks where connections between nodes form a directed graph along a temporal sequence. It is commonly applied to exhibiting temporal sequences and calculating the result based on the order of the input. In this project, we used LSTM (Long Short-Term Memory). So we will introduce the architecture of LSTM in detail.

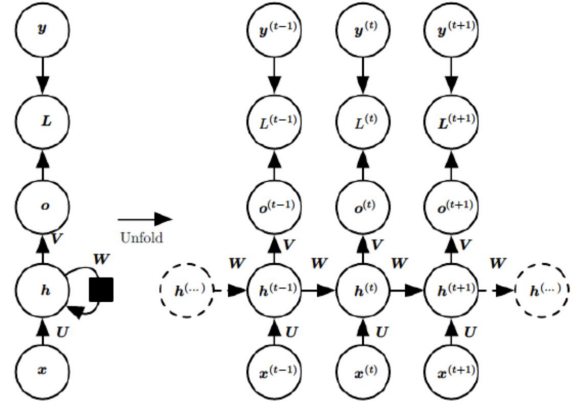


Figure 4: Architecture of RNN

Figure 4 shows the general architecture of RNN. The biggest difference between RNN and other deep learning models is that it has connections between the neurons in the same layer, which means that with the sequence input into the network, the output will also be calculated based on the past input.

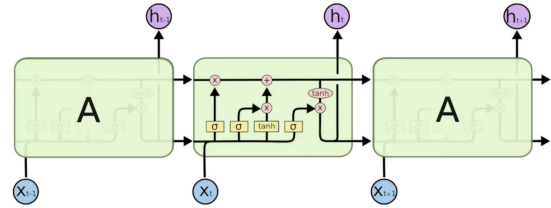


Figure 5: Architecture of LSTM neuron

Figure 5 shows the architecture of LSTM neurons. It is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell. In our project, LSTM can help us explore the order of the speech signal and help us get the emotion from it.

C. CNN & LSTM

Since the audio signal is a temporal sequence, we think that the order of the signal may be matters to the emotions within it. But CNN is a good model to analyze the MFCC feature, so we combine these two models. First we frame the original MFCC feature to several parts and it becomes a sequence of 2-D arrays. After that we use CNN to deal with each of them and get the sequence of the output. In the end, we use LSTM to process this sequence and finally get the classification result. If the order of the audio signal matters, we should get a better result.

D. Pruned Convolutional & Dense Layer.

As discussed in Section I, Pruning techniques are used to reduce the energy and storage. After pruning connections, neurons with zero input connections or zero output connections may be safely pruned. A neuron that has zero input connections or zero output connections will have no contribution to the final loss, leading the gradient to be zero for its output connection (or input connection) [2]. Thus, the dead neurons will be automatically removed during retraining. As we can see from Figure 6, Conv2D and Dense layers have the most parameters. In our paper, we add one more mask layer to each of the Conv2d and Dense layers, and set a parameter to decide the percentage of lowest neurons to be removed. Results are shown in the following section.

V. EXPERIMENT, RESULTS AND DISCUSSION

In this part, we will introduce the experiment we have done and use different figures and tables to present the result.

A. Traditional models

K-nearest-neighbors, an unsupervised learning algorithm, is served as a baseline model in our paper. an object is classified by a plurality vote of its neighbors with the object being assigned to the class most common along its K nearest neighbors. Here after testing, we use K=3 as our algorithms and get a baseline accuracy of 32% testing accuracy. We use this basic model to be compared by the following deep neural networks.

B. Deep learning models

In this project, we use three deep learning models, CNN and CNN-LSTM.

At the beginning, we'll give details about the parameters I chose for CNN because we use CNN in all three deep learning models. To compare their performance, we choose the same CNN parameters for them. We set the learning rate to 0.01 because under this learning rate, this model can converge in a short period of time and get a good result in the end. If lr is higher, the result would be worse. If lr is lower, the training time will be longer. Besides, we set the batch_size to ten. Since the size of deep learning models we use is too big, the server we used can't handle bigger batch_size, so we set it to ten. We do cross-validation by using 25% of the dataset to be the testing data and using the rest to train the model. In this way, we can validate the model. The architecture and parameters of CNN is shown in Figure 6. We use these parameters in all models.

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 30, 216, 32)	1312
batch_normalization_1 (Batch Normalization)	(None, 30, 216, 32)	128
max_pooling2d_1 (MaxPooling2D)	(None, 15, 108, 32)	0
dropout_1 (Dropout)	(None, 15, 108, 32)	0
conv2d_2 (Conv2D)	(None, 15, 108, 32)	40992
batch_normalization_2 (Batch Normalization)	(None, 15, 108, 32)	128
max_pooling2d_2 (MaxPooling2D)	(None, 7, 54, 32)	0
dropout_2 (Dropout)	(None, 7, 54, 32)	0
conv2d_3 (Conv2D)	(None, 7, 54, 32)	40992
batch_normalization_3 (Batch Normalization)	(None, 7, 54, 32)	128
max_pooling2d_3 (MaxPooling2D)	(None, 3, 27, 32)	0
dropout_3 (Dropout)	(None, 3, 27, 32)	0
conv2d_4 (Conv2D)	(None, 3, 27, 32)	40992
batch_normalization_4 (Batch Normalization)	(None, 3, 27, 32)	128
max_pooling2d_4 (MaxPooling2D)	(None, 1, 13, 32)	0
dropout_4 (Dropout)	(None, 1, 13, 32)	0
flatten_1 (Flatten)	(None, 416)	0
dense_1 (Dense)	(None, 64)	26688
dropout_5 (Dropout)	(None, 64)	0
batch_normalization_5 (Batch Normalization)	(None, 64)	256
dense_2 (Dense)	(None, 8)	520
Total params: 152,264		
Trainable params: 151,880		
Non-trainable params: 384		

Figure 6: Parameters of the CNN

With these parameters, we will show our experiment. Firstly, figure 7 shows the result of the CNN model on the original speech dataset. It's accuracy reaches about 77%. This is a really good result.

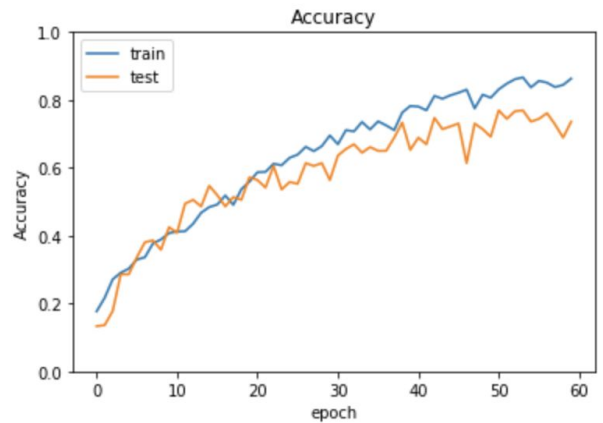


Figure 7: Accuracy vs epoch of CNN on original dataset

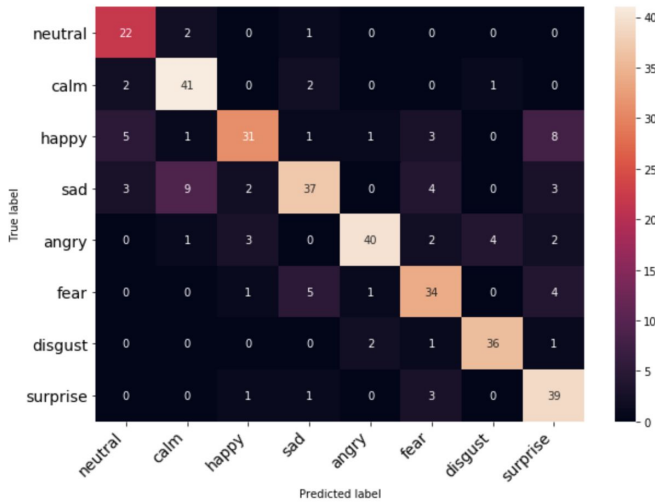


Figure 8: Confusion matrix of CNN on original dataset

Then, we add some noise on the original dataset and use the method stated above to reduce the noise and form the new dataset. The accuracy and the confusion matrix of CNN model on this dataset will be presented in figure 9 and figure 10. We can see that the accuracy of CNN on this dataset reaches about only 60%, which is worse than the CNN on the original dataset. This is due to the noise we add to our dataset. It affects the original audio signal and our noise reduction method can't reduce the noise totally. So the accuracy of CNN drops.

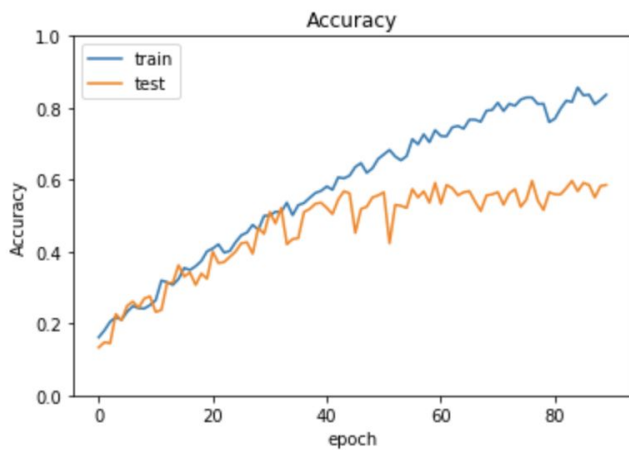


Figure 9: Accuracy vs epoch of CNN on new dataset

In addition to the CNN model, we also implemented CNN-LSTM model. If the order of the audio signal sequence has some relation with the emotion, we can get a better result than just the CNN model. The result is shown in figure 11 and figure 12. The accuracy of this model reaches about 70%, which is lower than the CNN model. We think that the order of the audio doesn't matter. We can get the emotion just through the tune of the audio without the order of the audio or speech.

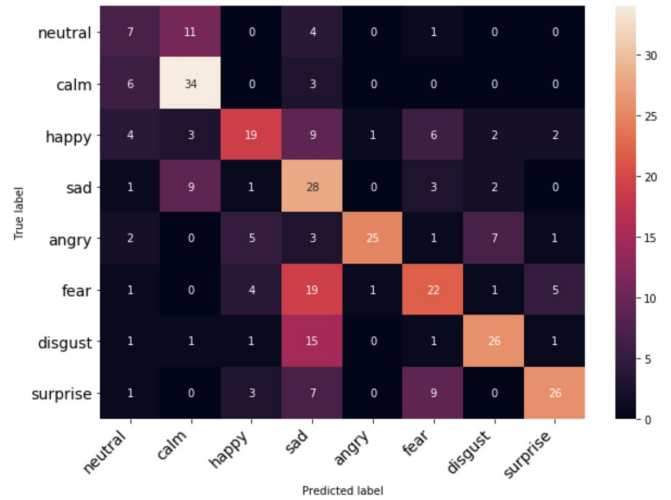


Figure 10: Confusion matrix of CNN on new dataset

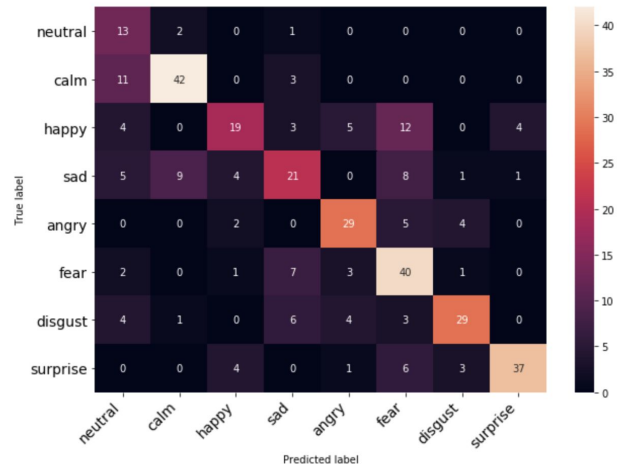


Figure 11: Accuracy of CNN-LSTM on original dataset

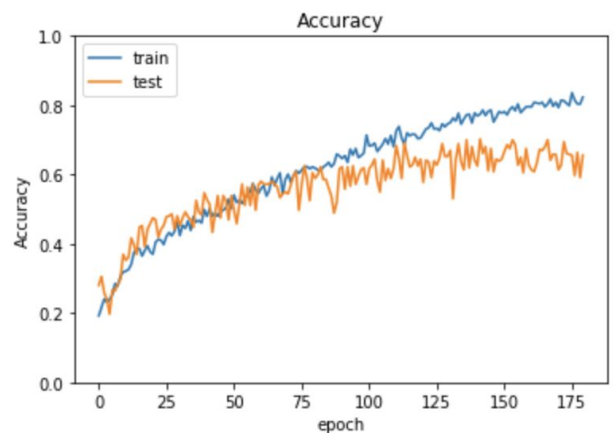


Figure 12: Confusion matrix of CNN-LSTM

By default, the mask parameter that adds to each Conv2D and Dense layer is set to 1, in which case the model (each weight in the layers) and the testing classification accuracy does not change. We can test by comparing the test accuracy of the original trained neural network and the one of the pruned model.

The choice of percentage to be removed depends on the complexity of models and accuracy. Thus, we could iterate different percentages and see the accuracy respectively.

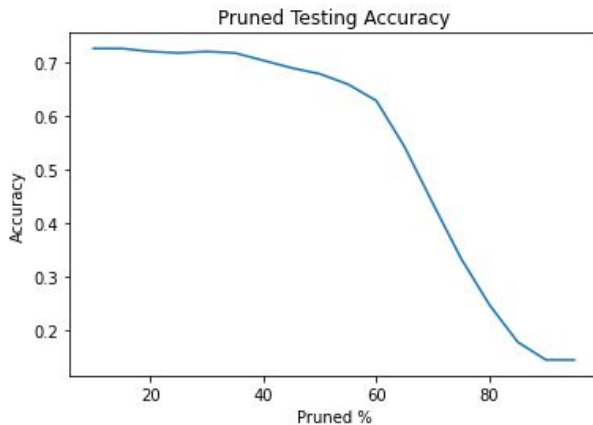


Figure 13: Testing set accuracy versus percentage of removed neurons (lowest absolute weight neurons).

From figure 13, we can see that after pruning, we could still get an accuracy over 70% and remove 40% of the neurons.

VI. CONCLUSION/ FUTURE WORK

From the above, we use audio files as inputs, add noise via spectral subtraction, extract MFCC features and train with CNN and LSTM models. Meanwhile, we use prune techniques to reduce the energy required to run large networks so they can run in real time on mobile devices. We can see from the results that CNN outperforms other models. Also, 40% percent of neurons can be removed while still keeping an accuracy of around 70%.

In the future, if we had more time and computational resources, we will try to combine text data and audio as our features for training. Also, from our results, we see that LSTM does not perform better than CNN. Thus we infer that chronological relations do not affect too much when we are using MFCC features to do classification. We would do more experiments to test this in the future.

- [1] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", *arXiv preprint arXiv:1611.10012*, 2017. 1
- [2] Song Han, Jeff Pool, John Tran, William J. Dally, "Learning both Weights and Connections for Efficient Neural Networks", *arXiv preprint arXiv:1506.02626*, 2015. 1 2 4
- [3] Huang, Zhengwei, et al. "Speech emotion recognition using CNN." Proceedings of the ACM International Conference on Multimedia. ACM, 2014.
- [4] Lee, Jinkyu, and Ivan Tashev. "High-level Feature Representation using Recurrent Neural Network for Speech Emotion Recognition." Sixteenth Annual Conference of the International Speech Communication Association. 2015.
- [5] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740-741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
- [6] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.
- [7] Liu, Yang, Mingli Xiao, and Yong Tie. "A Noise Reduction Method Based on LMS Adaptive Filter of Audio Signals." (2013).
- [8] Fujimoto, Masakiyo, and Yasuo Ariki. "Noisy speech recognition using noise reduction method based on Kalman filter." 2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100). Vol. 3. IEEE, 2000.
- [9] Mwema, W N., A spectral Subtraction Method for Noise reduction in Speech Signals, AFRICON, 1996., IEEE AFRICON (24-27 Sept. 1996), P. 382 - 385 vol.1

Individual Contributions

- Yucheng Bian

He worked on pruned Conv2D/Dense, background/ motivation, report, coordination.

- Lei Gao

He worked on noise reduction, related work, methods and reports.

- Yuepeng Shen

He worked on Conv2D, CNN-LSTM model, experiment and report.

Replies to Critical Reviews

Critical review from team 12:

- 1. What is the reason to use the same parameter throughout CNN models? Can you try different parameters within or on another CNN model?**

When we compare the performance of different models, in most cases, we control the parameters of the model. In this way, we can find the better model.

- 2. Does pruned CNN produce a better result than without pruned CNN? What is the accuracy of pruned CNN? How about a pruned Decision Tree?**

Pruned CNN is used to save storage and transmission time especially in mobile devices. As presented in the slides, when using pruned CNN, the default mask value is 1 and accuracy will remain the same. And if we drop 40% lowest neurons, the accuracy still remains at the same level, around 70%.

- 3. What variance is it when adding noises to the dataset?**

I set the variance to the max amplitude of signal. And we add this noise to signal by times 0.05.

Critical review from team 13:

- 1. Which dataset was the baseline (KNN, decision tree) tested on?**

We test the baseline model on all dataset and we only show the result on the original dataset.

- 2. We can understand that MFCC was used for feature extraction but how exactly is it incorporated with CNN?**

MFCC is a two-dimensional array feature. So we can use 2-D CNN to deal with it.

3. **How LSTM is used since LSTM works on temporal while CNN usually serves as an spatial feature extractor.**

How CNN-LSTM model is constructed needs more detailed explanations.

We frame the MFCC in several frames and see this as a sequence. In this way, we can use CNN on each of them and use LSTM to handle the sequence.

Critical review from team 40:

1. **There is a typo on slide page 11 about the MFCC feature extraction process. For step 4, DCT stands for Discrete Cosine Transform instead of Discrete Fourier Transform.**

Yes. We have corrected it and will pay more attention to this kind of problem.

2. **The 8 accuracy plot for CNN and CNN-LSTM can be made more intuitive by labeling the testing accuracy for each plot or using the same y-axis scaling. Also the accuracy for CNN-LSTM model with noise-reduction dataset is still increasing and not converging after 120 epochs. Maybe it can have better results if it is trained for more epochs as the current plot suggests it is a good model and shows very few signs of overfitting.**

We change the epochs and re-run the code.

3. **Since the main motivation for using pruned CNN is to reduce computational time by dropping certain amount of parameters without much loss of accuracy, it could be better if they could provide a comparison of running time between the original CNN and pruned CNN.**

Since our model is complex and data is insufficient, we indeed did experiments on different dropping percentages versus time but could hardly see the decreasing trends. Also, the saved computation time is better to reveal when using mobile devices, because pruning techniques are mostly used to reduce the energy required to run large networks so they can run in real time on mobile devices.