NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

Improving CNN-based Speech Emotion Recognition with LSTM Variants and Unsupervised Representation

CZ4042 Neural Networks and Deep Learning

Final Group Project Report

School of Computer Science and Engineering

Li Bingzi (Matric No. U1722793H)

Zeng Yanxi (Matric No. U1722874L)

Zhang Yuehan (Matric No. U1722863L)

Project Aim-The aim of this project is to build a deep learning model that can classify the audios into several emotions.

Keywords: CNN, Classification, Audio Emotion Recognition, Long Short-Term Memory(LSTM), CNN with Attention, autoencoder

1 Introduction

Speech, as one of the primary instruments of emotion expression, serves as one paramount factor for understanding human emotion in a human machine interface. Recognizing the emotion carried by the speech is an important task for providing corresponding responses. [1] Deep learning has recently become an effective way for Speech Emotion Recognition tasks - also known as SER.

The idea of Speech Emotion Recognition (SER) comes from the fact that human voice usually reflects underlying emotion through tone and pitch. The voice can be represented by a data matrix (MFCC or Mel-spectrogram) and put into a neural network for training. Convolutional Neural Network is commonly used for SER, as it can capture the salient features of tone and pitch which allows it to classify an audio into the emotion categories. [2]

In this project, we will explore beyond the CNN based SER model. We aim to validate the improvement on basic 2D CNN with the proposed model variants, and to evaluate the usefulness of the unsupervised representation for SER. With a dataset containing short audios speaking some sentences from different speakers, we built SER models of several variants: the basic 2D-CNN model, the CNN + Long Short Term Memory (LSTM) model, the CNN + Attention based LSTM model. Further exploration is made on unsupervised representation learnt from an autoencoder, for which the encoder output would be combined with the intermediate layer output before the final model output.

The datasets used in this project are RAVDESS and CREMA-D. RAVDESS is used to perform the SER task while both RAVDESS and CREMA-D are used for the autoencoder training. The use of one dataset in SER task is to eliminate the interference brought by the drastic difference of the two datasets' sources and qualities. The use of two datasets in autoencoder training is to enhance the autoencoder with an increased sample size.

2 Literature Review

While the traditional machine learning methods for Speech Emotion Recognition tasks mainly focused on feature selections that does not require a large amount of data, it is hard to ensure that the features selected could perform across a variety of datasets. [3] Therefore, as the rise of deep learning is able to extract the a higher level feature representation, the deep learning techniques have been widely applied in the SER tasks.

The idea of using CNN on SER models came from Q Mao et al [4], who proposed that CNN models could learn salient features pertaining to affective expressions for SER. Using CNN also has robust performance under various complicated scenarios. Other than using CNN, Jinkyu Lee [5] proposed the use of RNN to extract the high-level features. His results has shown significant improvement from the DNN models. Along with these proposed models, H.

M Fayek [6] provided a method to augment training data. The dataset after augmentation could effectively reduce the extent of overfitting and in some cases improve the accuracy as well. What inspires this project the most is the work from George Trigeorgis et al [7] , who combined CNN with LSTM networks. The proposed model is able to capture the salient representation of speech signal, which combines the merits from both CNN and RNN model architecture.

3 Methodology

3.1 Data Pre-processing

In this project, the data consists of audio files in .wav format. Thus, data preprocessing is needed to convert the file into representations that are suitable to learn. Two methods are researched and Mel-spectrogram is chosen for the project.

3.1.1 Melspectrogram

One of the most common feature extraction is melspectrogram. The figure below shows the plot of an audio wave. The mel-spectrogram is done by separating the input into window size, performing Fourier transformation and generating a melscale. Based on the mel-scale, the corresponding mel-spectrogram is computed.

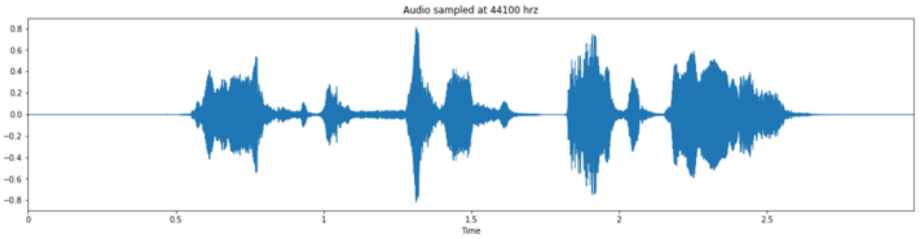


Figure 1. An audio wave

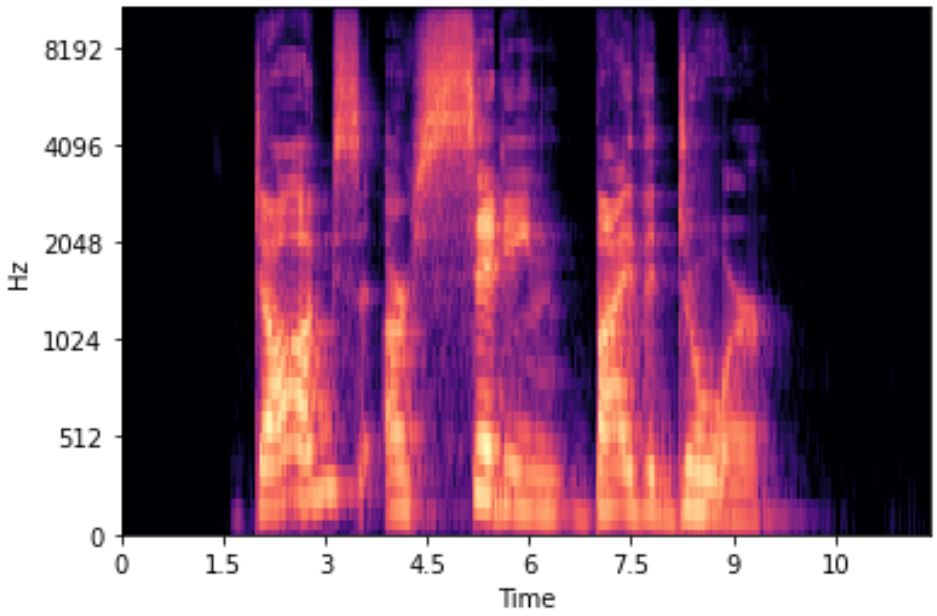


Figure 2. The same audio wave transformed into mel-spec

3.1.2 MFCC

Another common feature extraction is MFCC, which is a time sequence with MFCC bands on the y-axis. MFCC representation is from taking the logs and computing DCT (discrete cosine transform) from the mel-spectrogram. The figure below is the same audio wave extracted by MFCC. It is clear that the data is less complex than data extracted by mel-spec.

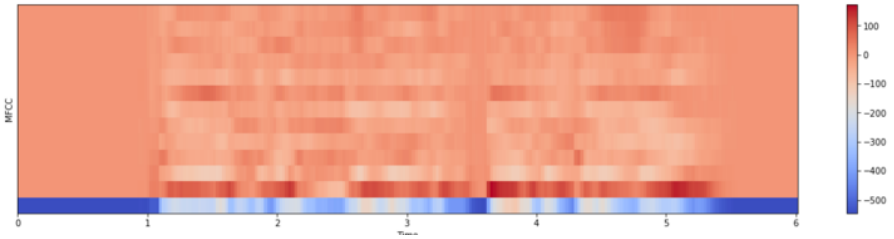


Figure 3. Audio wave transformed into MFCC (visualized by color bar)

3.1.3 Comparison between MFCC and Melspectrogram

MFCC is more compressed than melspectrogram, often with less than 20 coefficients in contrast to over 32 bands in melspectrogram. Therefore, MFCC best suits simpler linear models as it is more decorrelated. In this project, CNN network and its variants are trained on this set of data. Melspectrogram performs better on strong classifiers like CNN. Therefore, mel-spectrogram is selected for feature extraction.

3.2 Data Augmentation

In view of the limited number of audio clips from the RAVDESS dataset, data augmentation is used to increase the sample size and to avoid potential issues of overfitting.
The original size of the RAVDESS dataset consists of 1440 audio clips. Slight modified copies of the audio clips are synthesised and added to the existing data, and the new sample size after augmentation is 4320.

3.2.1 Methods used in data augmentation

We manage the data augmentation by basically generating a noisy signal and adding it to original mel-spectrogram data. The signal is generated step by step, i) we choose to generate White Gaussian noise first ii) then normalize the signal and noise and compute their power iii) random the SNR and compute K covariance matrix for each noise iv) and finally generate the noisy signal required to be later added to mel-spectrogram data.

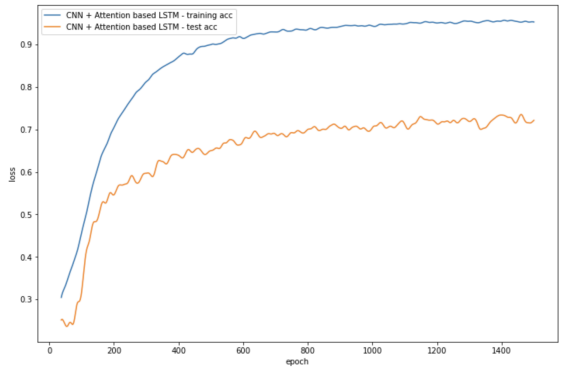
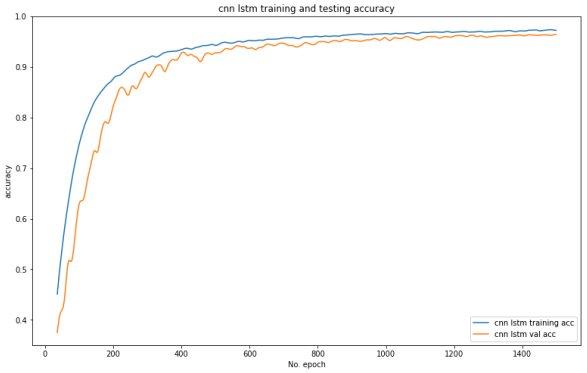
Overfitted model trained with original dataset	Model trained with dataset after data augmentation
	

Table 1. Comparison of the extent of overfitting without / with data augmentation

The improvement brought by data augmentation is significant. As a comparison, the same model is trained with the original dataset and the new dataset after augmentation. Their training accuracy and test accuracy are plotted in the plots below. As illustrated in Table.1, the accuracy of the model improves while the issue of overfitting has been avoided.

3.3 2D CNN

When processing datasets like images, ConvNet is able to capture the Spatial and Temporal dependencies in an image through relevant filters. In this experiment, a simple and commonly used architecture of ConvNet is designed: Conv blocks followed by fully connected layers.

The standard Conv block includes: 2DConv layer, batch normalization, MaxPooling layer and dropout. Dropout is applied to each block,

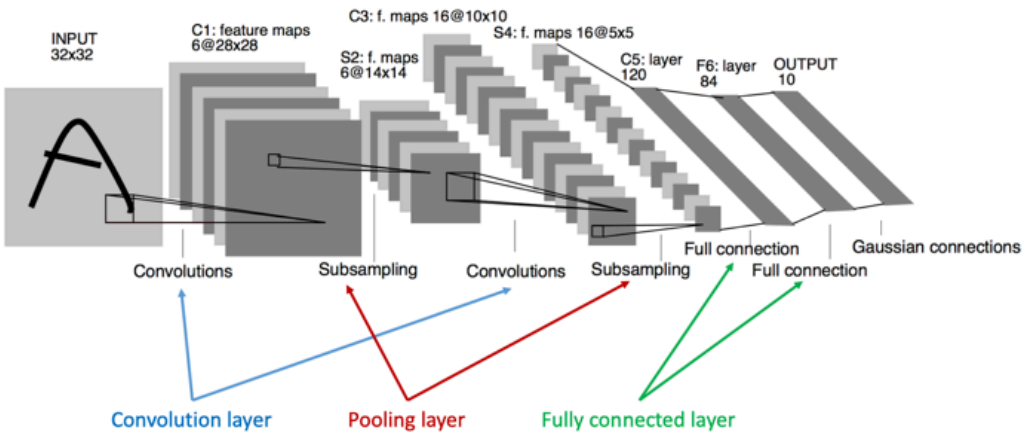


Figure 4. Convolutional Network [from lecture slides]

3.4 CNN + LSTM

For further improvement on simple 2D CNN, also in order to better fit our task, that is visualizing audio time series prediction problems, we experimented with a special architecture, the CNN Long Short-Term Memory Network (CNN LSTM), which is specially designed for sequence prediction problems.

“CNN LSTMs are a class of models that is both spatially and temporally deep, and has the flexibility to be applied to a variety of vision tasks involving sequential inputs and outputs.” (Long-term Recurrent Convolutional Networks for Visual Recognition and Description, 2015)

The CNN plus LSTM architecture can be considered as two sub-models, involving applying Convolutional Neural Network layers in front end for feature extraction on input data and combined with LSTM layer for interpreting the features across time steps, i.e. to support sequence prediction.

In our experiment, the CNN LSTM is defined by first defining four CNN layers, wrapping them in a TimeDistributed layer on the front, followed by an LSTM layer, and a softmax dense output layer.

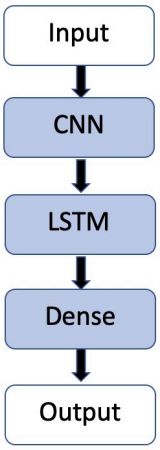


Figure 5. CNN Long Short-Term Memory Network (CNN LSTM) Architecture

3.5 CNN + Attention based LSTM

To further enhance the learning in the LSTM layer, we experimented with the attention mechanism. The attention mechanism is able to concentrate on different parts of the extracted features, which is important as the emotional saturation and contribution to the prediction outcome could vary across the audio clip. For example, the silent segments and less emotional segments of speech contain less emotional information. Hence, attention mechanism could be applied in the output of the LSTM layer to produce the weighted context vector that could pay particular attention to the more important parts of the inputs. [1]

In our work, the LSTM layer would be followed by an attention layer that produces the weighted outputs as the inputs for a dense layer before the output layer in order to extract the most useful information from the LSTM outputs for the emotion recognition.

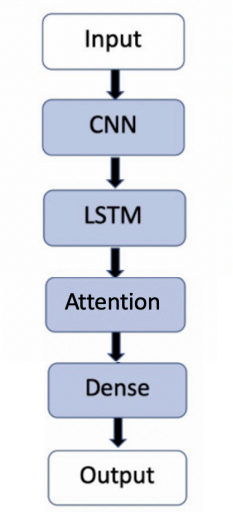


Figure 6. CNN Attention-based Long Short-Term Memory Network (CNN LSTM) Architecture

3.6 CNN + Unsupervised Representation Learning with Autoencoder

Other than representation of the features from supervised learning, unsupervised representation learnt from unlabeled could also improve the recognition accuracy. A similar approach has been adopted in [2] to improve a CNN-based Speech Emotion Recognition task, where an autoencoder is built on unlabeled audio clips and the learnt representation from the encoder is fed into the original CNN-based model to enhance recognition. In [3], similarly, different autoencoders were trained on different features to provide input features for emotion recognition.

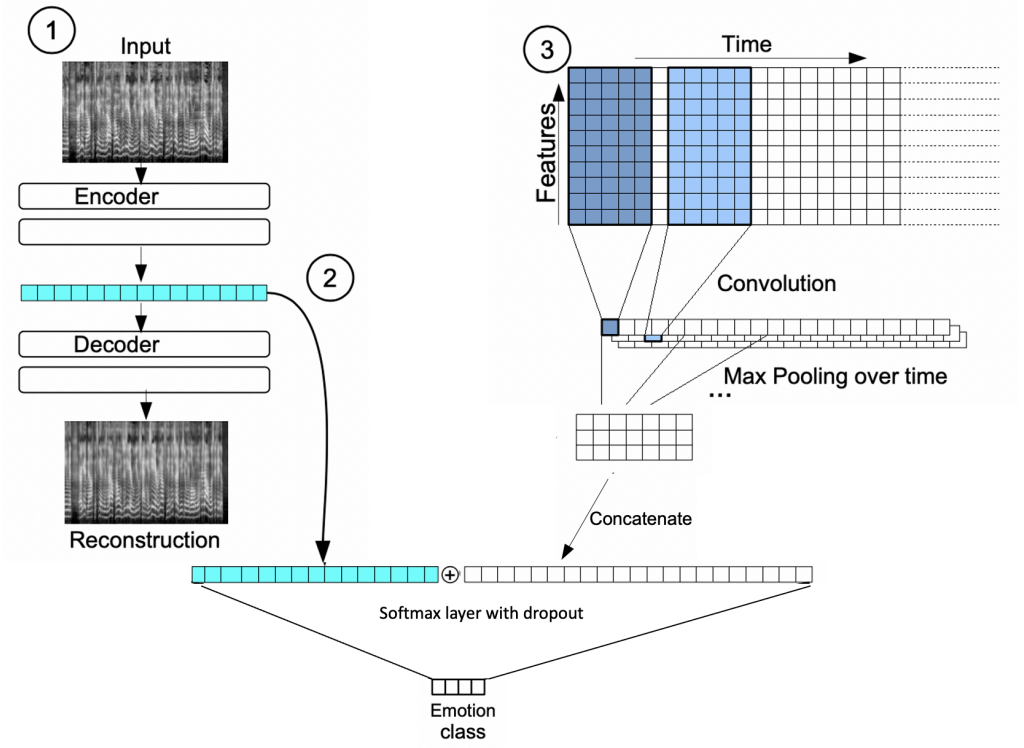


Figure 7. CNN + Autoencoder Architecture (Modified from [2])

In our work, an autoencoder would be trained from both the RAVDESS dataset and the CREMA-D dataset. Different model architectures and hyperparameters for the autoencoders would be experimented to find the optimal autoencoder architecture.

The learnt representation from the encoder of the autoencoder would then be concatenated to the intermediate outputs from the original recognition model. The combined representation would then be passed into a dense layer, and finally, the output layer.

4 Experiments, Results & Discussion

4.1 Data Exploration

4.1.1 RAVDESS

The recording of the RAVDESS[8] dataset is very clear. The upside of this dataset is that it includes both genders. The dataset has 8 commonly expressed emotions.

In theory, the emotions are recognized through pitch and tone. The common sense is that the pitches of male and female voices are different. To exclude the effect of gender during emotion recognition, each emotion label is attached with gender. In the end there are 8 labels.

The dataset includes:

neutral	96
calm	192
sad	192
happy	192
disgust	192
angry	192
fear	192
surprise	192

Table 2. Emotions in the RAVDESS dataset

There are half male speakers and half female speakers in each category. If trained on a dataset with only male or female speakers, the resulting neural network will be biased towards that gender. The neural networks in this project are trained on both genders so it can be gender-invariant when classifying the emotions.

Comparing sentences:

As shown from the figure below, different sentences uttered have different wave plots. However, the two save plots have similar features, which is a huge spike in the sound waves, which might be a prominent feature for angry emotion. Different contents uttered does not affect the feature.

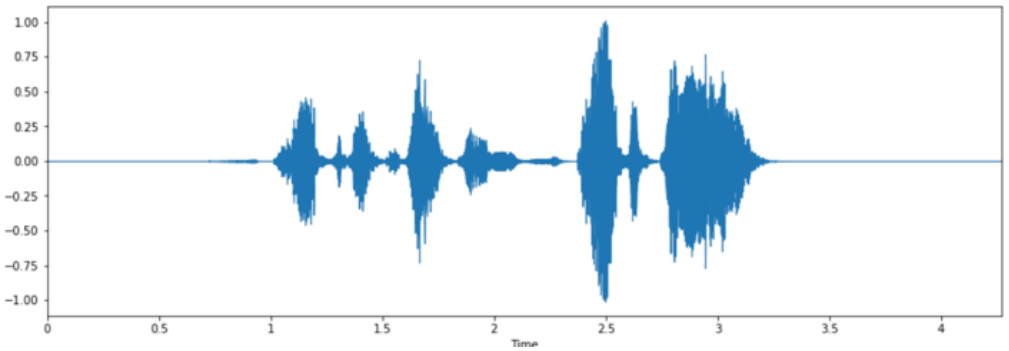


Figure 8. female, intense angry, speaking "Kids"

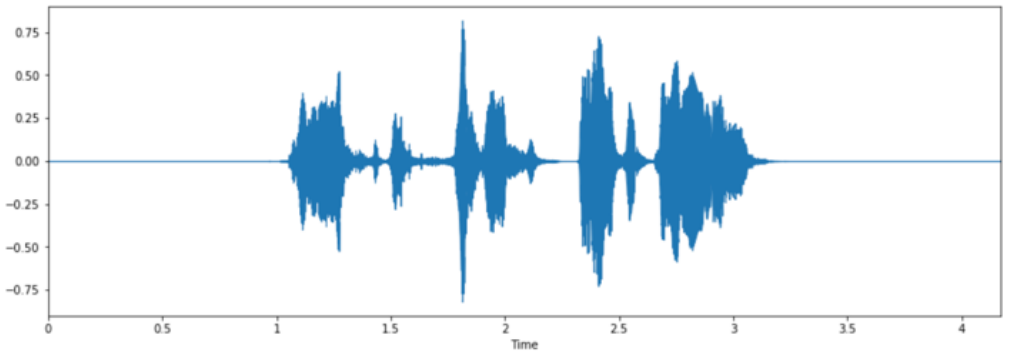


Figure 9. female, intense angry, speaking "Dogs"

Comparing emotions:

Comparing figure 9 and figure 10, same gender speaking same content in angry and neutral emotion. Clear features can be distinguished. For angry emotion, there are spikes and the volume is high. Neutral voice has low volume and a consistent pace.

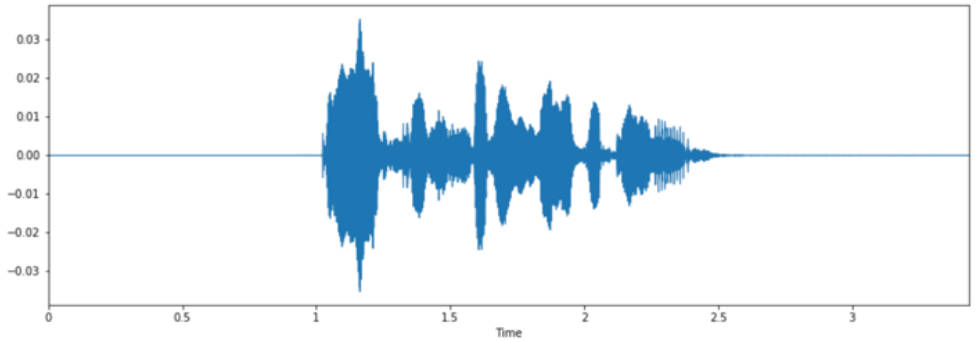


Figure 10. female, neutral, speaking "Dogs"

Comparing genders:

Comparing figure 10 and figure 12, female and male speaking the same sentence with the same emotion, similar patterns can be seen in the sharp spikes.

The feature which needs attention is that male usually has much lower pitch than female. And more interesting that females tend to express the emotion more strongly, which is shown later in the training as well.

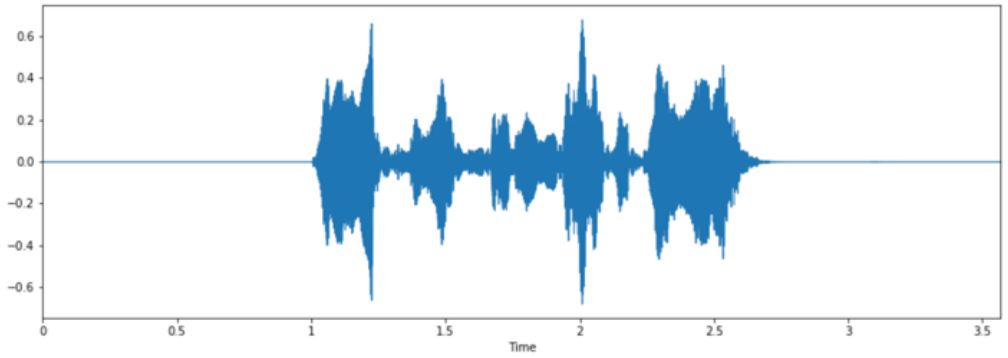


Figure 11. male, intense angry, speaking "Dogs"

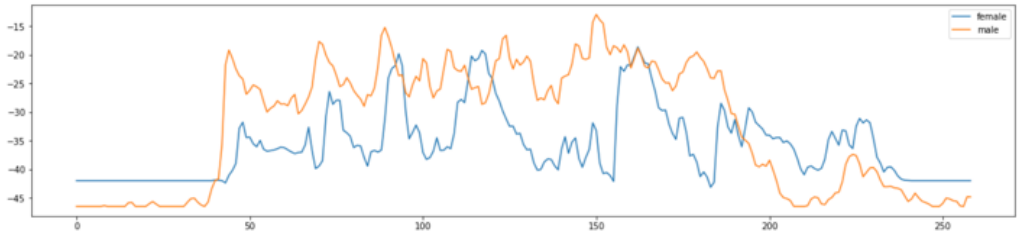


Figure 12. male vs female, intense angry, speaking "Dogs" (taking average across 30 mfcc bands)

4.1.2 CREMA-D

CREMA-D[9] dataset is much larger than RAVDESS.
The downside of this dataset is that the audios are taken from movies. They are not the best of quality.The upside is that this dataset covers various accents, genders, age. In this project, the CREMA-D dataset is used to train the autoencoder which later will be used to improve the CNN model.

Comparing with RAVDESS dataset:
Comparing figure 10 and figure 14, from different sentences uttered in these two dataset, the sharp spikes in the angry emotion are still present. This is the base for the classification model to be possible.

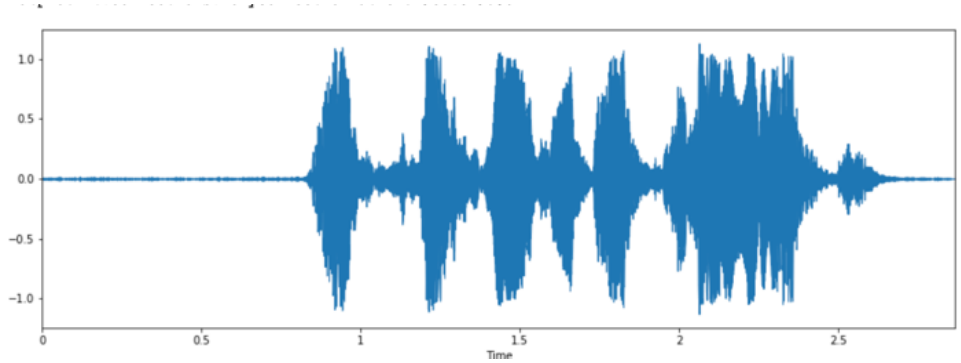


Figure 13. female, intense angry

female_neutral	512
male_neutral	575
female_sad	600
male_sad	698
female_happy	600
male_happy	698
female_disgust	600
male_disgust	698
female_angry	600
male_angry	698
female_fear	600
male_fear	698

Table 3. emotions in the CREMA-D dataset

4.2 Baseline 2D CNN

This baseline 2D CNN consists of:

1st LFLB	`Conv2D(64, kernel_size=(3, 3), strides=(1, 1), padding='same')`
	`BatchNormalization()`
	`Activation('elu')`
	`MaxPooling2D(pool_size=(2, 2), strides=(2, 2), padding='same')`
	`Dropout(0.3)`

2nd LFLB	`Conv2D(64, kernel_size=(3, 3), strides=(1, 1), padding='same')`
	`BatchNormalization()`
	`Activation('elu')`
	`MaxPooling2D(pool_size=(4, 4), strides=(4, 4), padding='same')`
	`Dropout(0.3)`

3rd LFLB	`Conv2D(128, kernel_size=(3, 3), strides=(1, 1), padding='same')`
	`BatchNormalization()`
	`Activation('elu')`
	`MaxPooling2D(pool_size=(4, 4), strides=(4, 4), padding='same')`
	`Dropout(0.4)`

4th LFLB	`Conv2D(128, kernel_size=(3, 3), strides=(1, 1), padding='same')`
	`BatchNormalization()`
	`Activation('elu')`
	`MaxPooling2D(pool_size=(4, 4), strides=(4, 4), padding='same')`
	`Dropout(0.4)`

Flatten layer	`Flatten()`

Dense layer	`Dense(64)`
	`Dropout(rate=0.5)`
	`BatchNormalization(`
	`Activation("relu")`
	`Dropout(rate=0.5)`

Output layer	`Dense(y_train.shape[1], activation='softmax')`

* LFLB refers to local feature learning block

Table 4. Baseline 2D CNN models design

Applying this model in training, we obtained the following results.

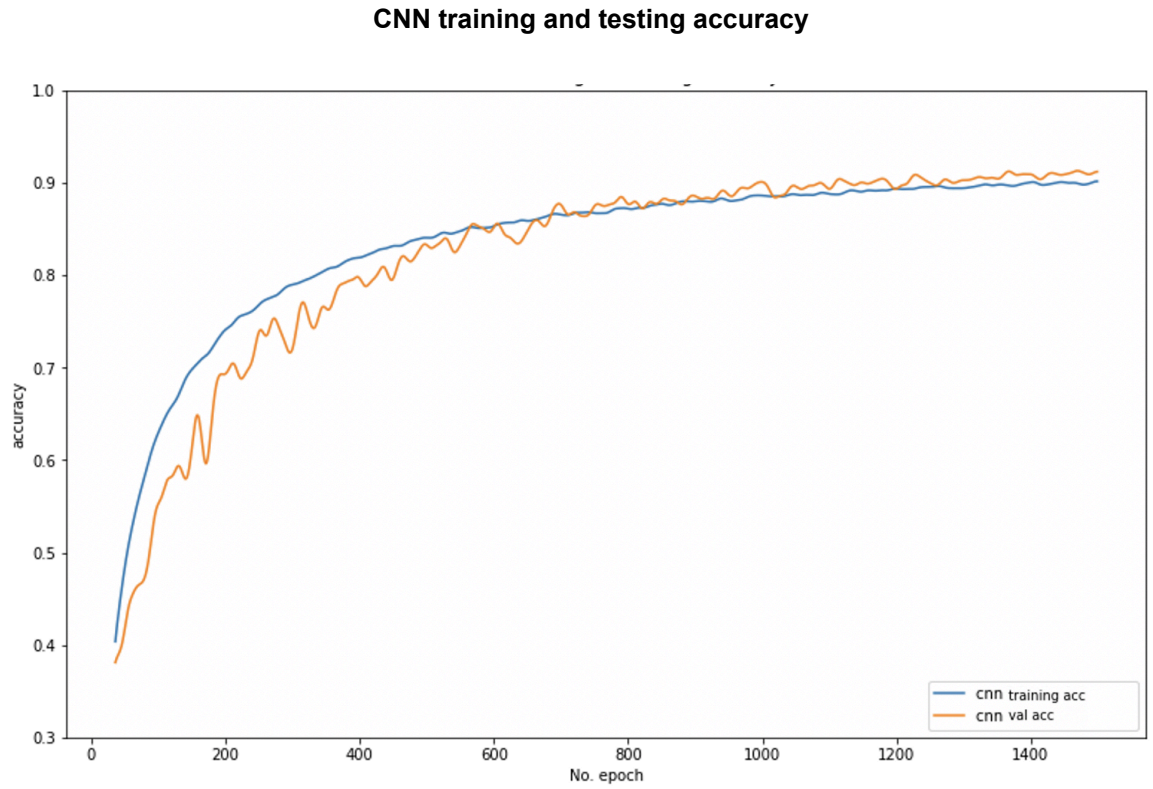


Figure 14. RAVDESS , melspectrogram, train & test acc

Model testing accuracy: 92.31%

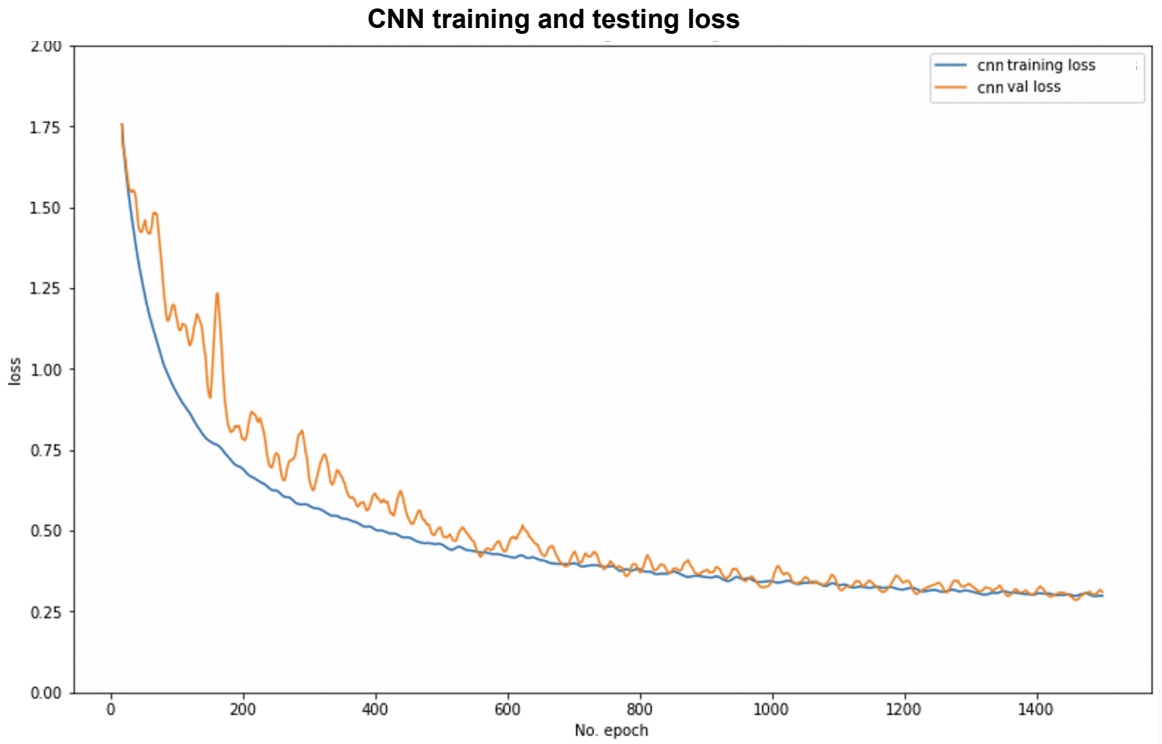


Figure 15. RAVDESS, melspectrogram, train & test loss

Observed from above plots of training testing accuracy and loss, we achieved a rather good result using this CNN model design, with test accuracy 92.31% and well avoided overfitting problem, by observing testing accuracy converged at the same value of training accuracy. It is also found that at the last hundreds epochs, testing accuracy go beyond training a bit, guess is due to our random data splitting, the testing data are easier to train compared to training, and this shall be able to be ignored since the difference was less than 0.2%, this model still is considered as a rather well performed one, hence will be later used as base for adding more useful techniques to achieve further improvement.

4.3 CNN + LSTM

Here as mentioned in the methodology section, in order to improve performance of previous baseline 2D CNN, an additional LSTM block is added after convolutional layers before the softmax output layer. This design of adding an additional LSTM layer is for interpreting the features across time steps, i.e. to better support sequence prediction.
In this experiment, the CNN LSTM is defined by first defining four CNN layers, wrapping them in a TimeDistributed layer on the front, followed by an LSTM layer, and a softmax dense output layer.

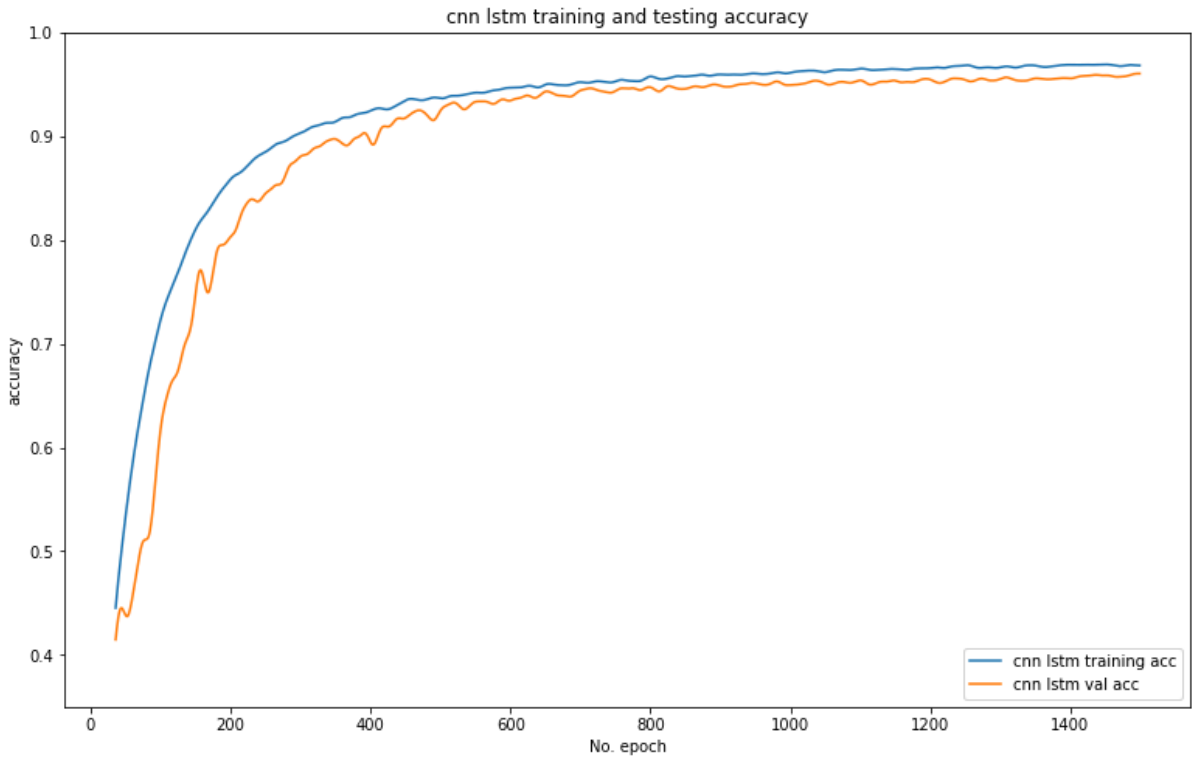


Figure 16. RAVDESS dataset, Mel-spectrogram, CNN LSTM training & testing accuracy

Model accuracy:95.83%

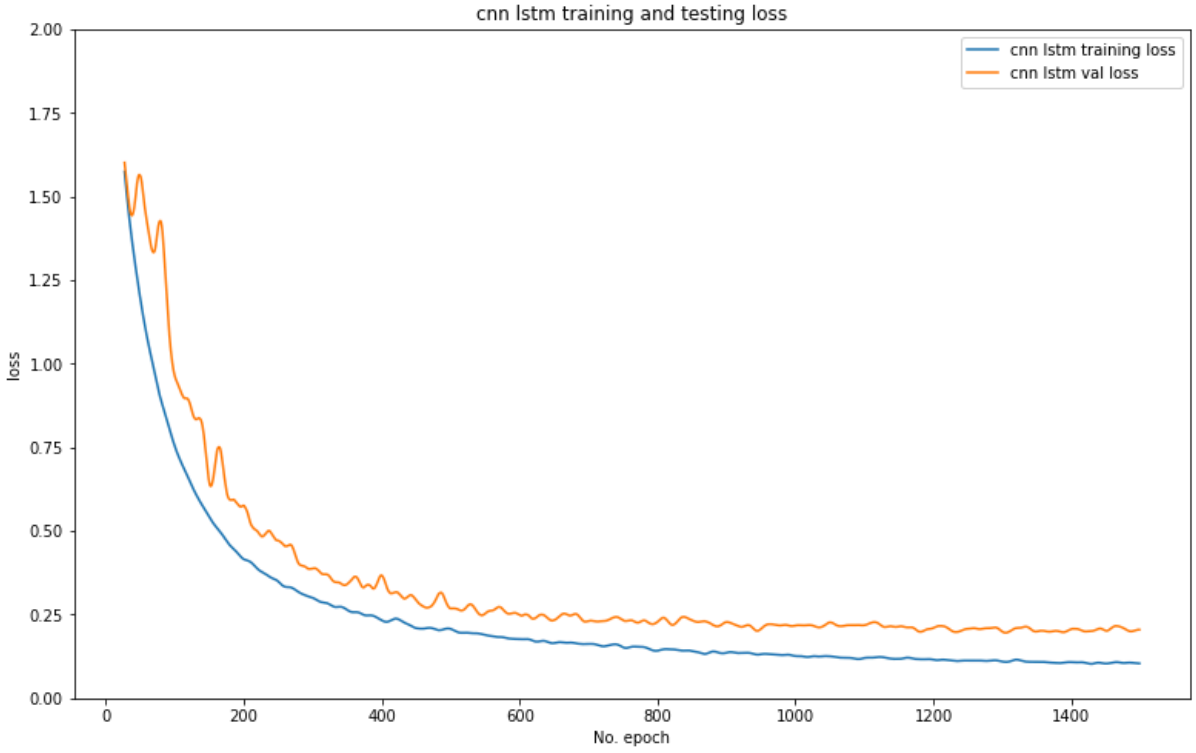


Figure 17. RAVDESS dataset, Mel-spectrogram, CNN LSTM training & testing loss

Figure 16 and Figure 17 above shows the training test accuracy and loss respectively, from which we observed that CNN with an additional LSTM layer added did have a good impact on increasing model performance, a higher accuracy was reached.

This shows by taking into consideration that our experiment was targeted on audio, i.e. sequential dataset, adding LSTM can be beneficial.

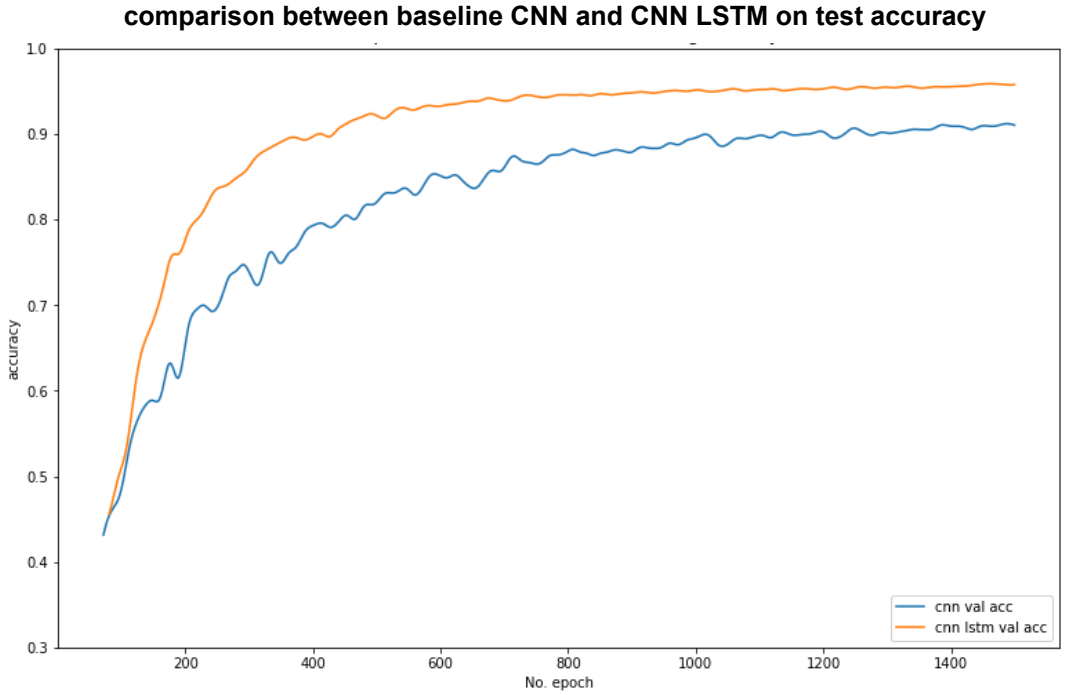


Figure 18. RAVDESS dataset, Mel-spectrogram, comparison between baseline CNN and CNN LSTM on test accuracy

By comparing test accuracy of baseline CNN and CNN LSTM models as shown in Figure 18, it is clear that with LSTM added model accuracy reaches a rather high value of 95.83%, nearly 4% higher than the previous result we obtained in baseline CNN.

Besides, by looking at Figure 16 train and test accuracy plotted above, this model has a good performance in the aspect of overfitting condition also, with training acc curve and testing acc curve converging at around the same value.

This shows the advantage of applying additional the LSTM layer on baseline CNN.

4.4 CNN + Attention based LSTM

To further enhance the model performance, we have applied the Attention mechanism on the LSTM layer. Due to the increase in model complexity, we have also increased the dropout to avoid overfitting.

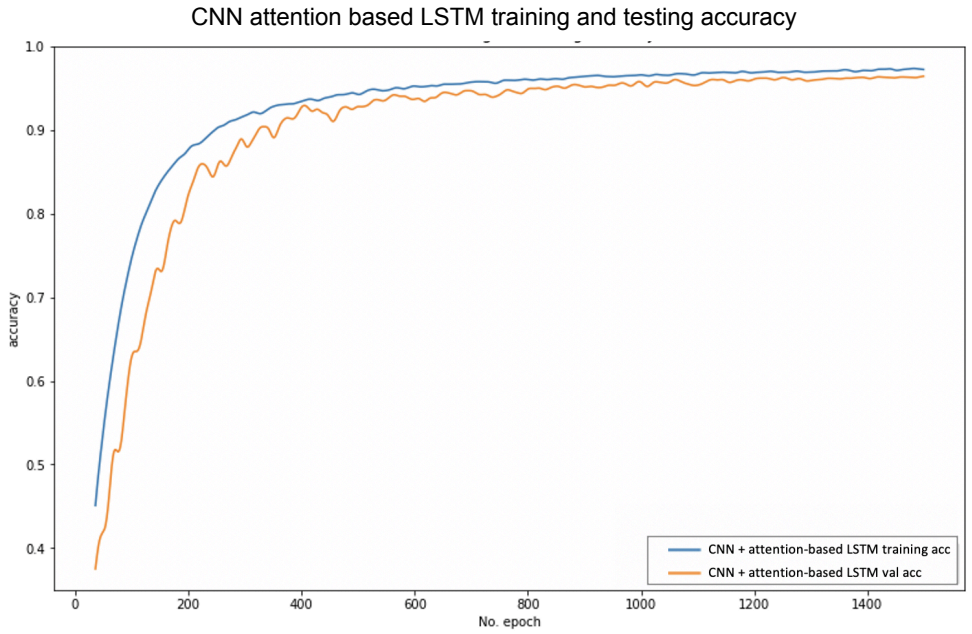


Figure 19. RAVDESS dataset, Mel-spectrogram, CNN attention based LSTM, training and testing acc

Model accuracy: 96.30%

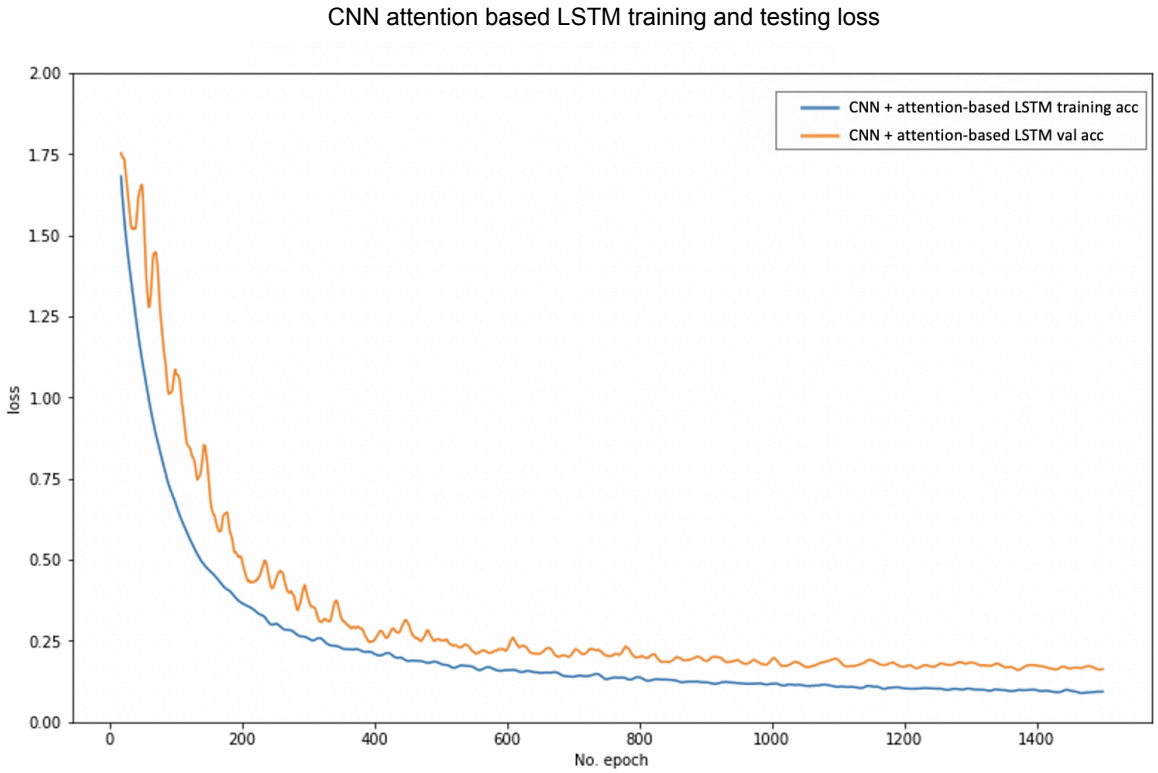


Figure 20. RAVDESS dataset, Mel-spectrogram, CNN attention based LSTM, training and testing loss

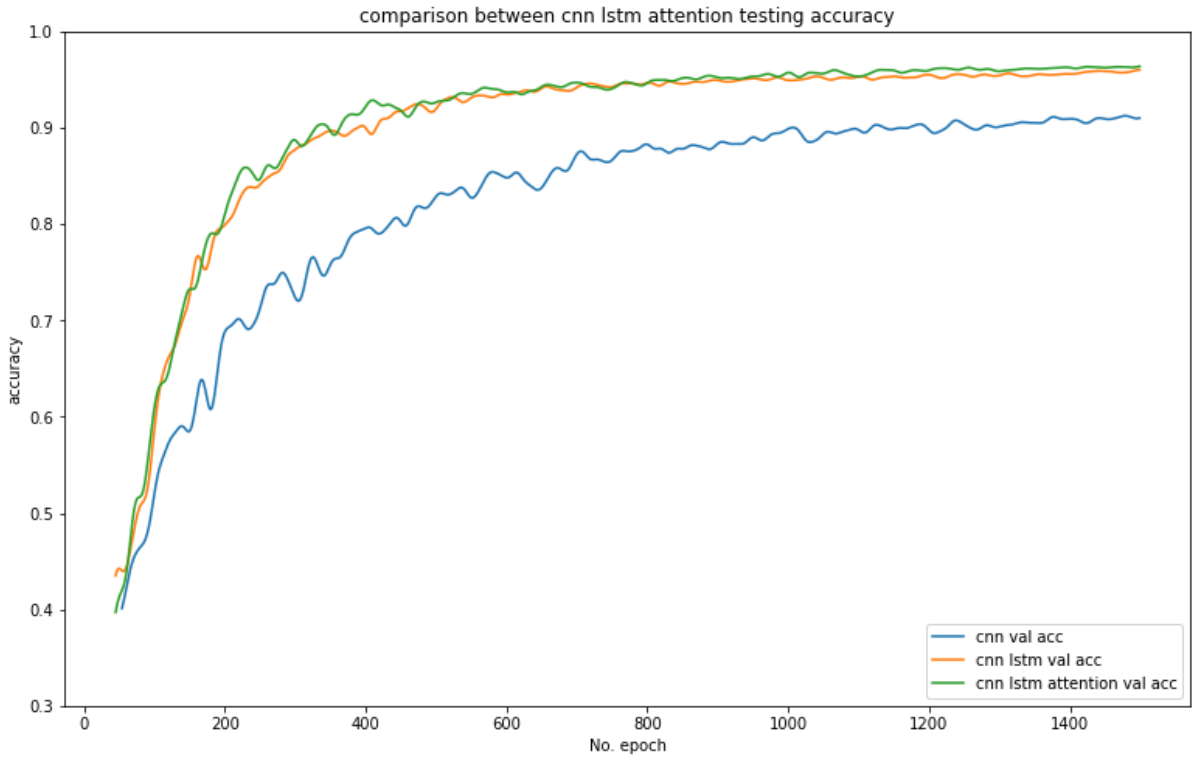


Figure 21. RAVDESS dataset, Mel-spectrogram, comparison between baseline CNN, CNN LSTM and CNN LSTM Attention on test accuracy

As shown in the above test accuracy comparison between CNN + LSTM and CNN + Attention based LSTM, we can arrive at the following observations:

1. The two models have comparable performance before the 800th epoch. They also have comparable speed of convergence.
2. After the 800th epoch, the CNN + Attention based LSTM has shown better accuracy than CNN + LSTM. Although at the end of the 1500 epochs, both models are having similar test accuracy, the attention based model has shown generally a higher accuracy as well as a higher maximum accuracy. We can conclude that, at this feature dimension level, adding attention mechanisms does improve the model.

Limitation:

Attention mechanism aims to help the LSTM layer focus on certain parts of the sequence data, which in turns enhance the accuracy by extracting the more useful information. In the feature dimension level adopted in this project, the advantage may not be significant as it might get more obvious when the dimension of features gets higher. Due to the time and resource constraints, we only discuss the extent of improvement from attention mechanism in the current level of feature dimension.

4.5 Autoencoder

To select the optimal set of hyperparameters for the autoencoder model, we consider the following factors:

Modification	Options
Model architecture	1. Fully Connected (FC) DNN with Dense layer 2. RNN with GRU layer
Model complexity (inclusive of input layer)	1. 3-layer encoder for FC DNN 2. 5-layer encoder for FC DNN 3. 2-layer encoder for RNN
Regularization	1. None 2. Dropout = 0.2
Encoding dimension	1. 100 2. 300

Table 5. Structures and Hyperparameters of Autoencoder

Due to constraints on the computation resources, the training of RNN autoencoders with more than 2 layers would exceed the maximum time allowed for one user in the school GPU server provided for this project. Therefore, we restricted our search space of the optimal model architecture to RNN with 2 layers (inclusive of the input layer).

A batch size of 64, the Adam optimizer and the Mean Square Error loss function are used for all candidate autoencoder models.

4.5.1 DNN Autoencoder with Dense Layers

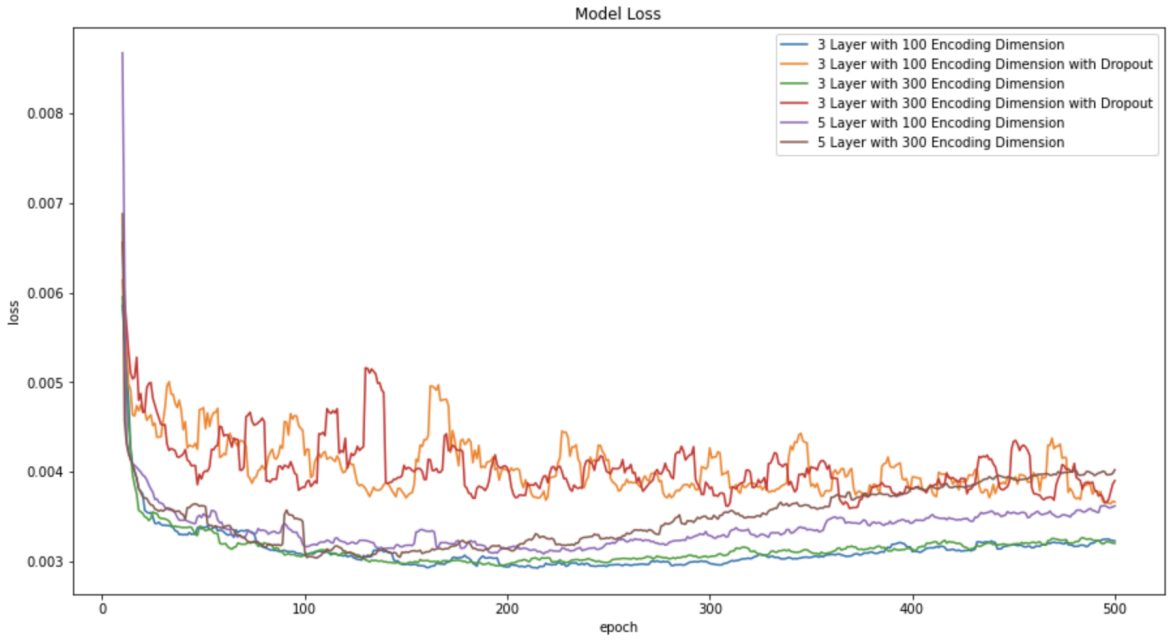


Figure 22. FC DNN Autoencoders Test Loss

From the above plotting of FC DNN autoencoders' test losses, the 5 layers FC DNN with dropouts have been unable to perform. Therefore, we eliminated the 5 layers FC DNN with dropouts for a better comparison amongst the remaining models.

From the above test loss plotting, we can observe that:

1. The 3-layer FC DNN with dimensions of both 100 and 300 have comparable performance regardless of the addition of dropout. While there is a slight increase in test loss while the epoch increases above 400, adding dropout to the 3-layer FC DNN models does eliminate the slight rise in loss but also hurts the model performance.
2. The 5-layer FC DNN faces more severe overfitting. However, adding dropouts to the model has severely hurt the model with high and unstable test losses. It could suggest that adding dropouts to the autoencoders is risky even if it is only added after the first dense layer of encoder and before the last layer of decoder. Hence, the overfitting in the 5-layer FC DNN is difficult to be resolved by adding dropouts and simple model architecture has to be used.

The above plots used only 500 epochs due to constraint of time and resources. To further investigate the outcome of adding dropout, we use the 3-layer FC DNN with dimension 100 as an example and trained the model for 1000 epochs:

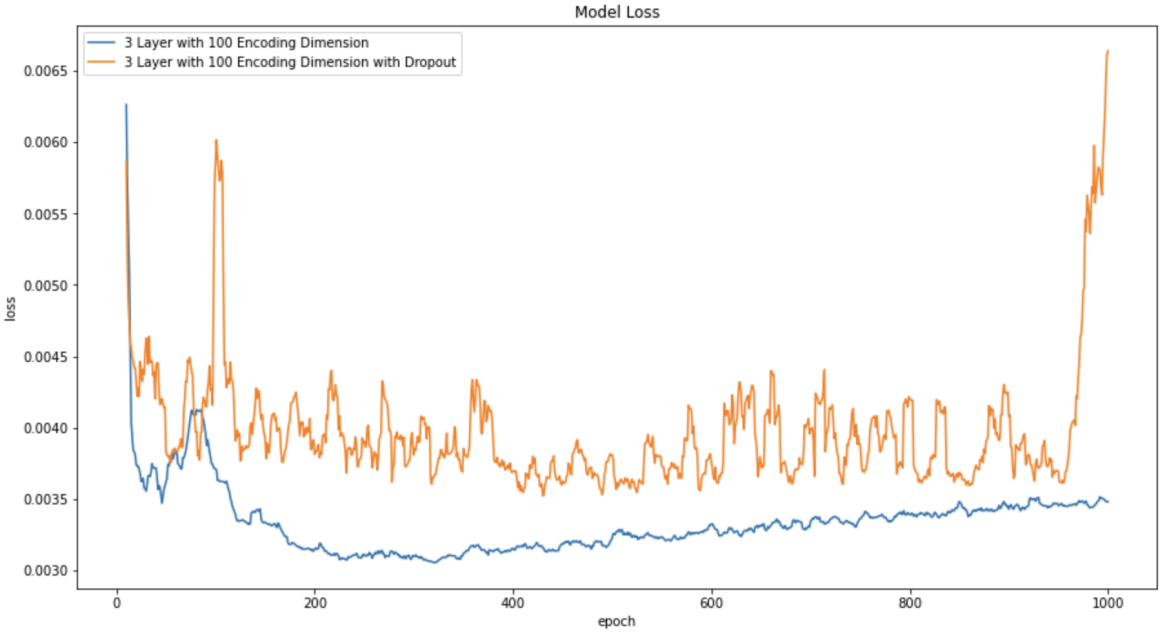


Figure 23. 3 layer FC DNN with / without dropout

The above comparison has shown that the model overfits without adding dropout. However, the model with dropout added also has unstable performance.

Due to the constraints of time and computation resources, we were unable to investigate further with more epochs. Hence, we tentatively select the 3-layer FC DNN autoencoder without dropout as the optimal autoencoder with dense layers. Both dimensions of 100 and 300 would be tested in the following section for classification accuracy.

4.5.2 RNN Autoencoder with GRU Layers

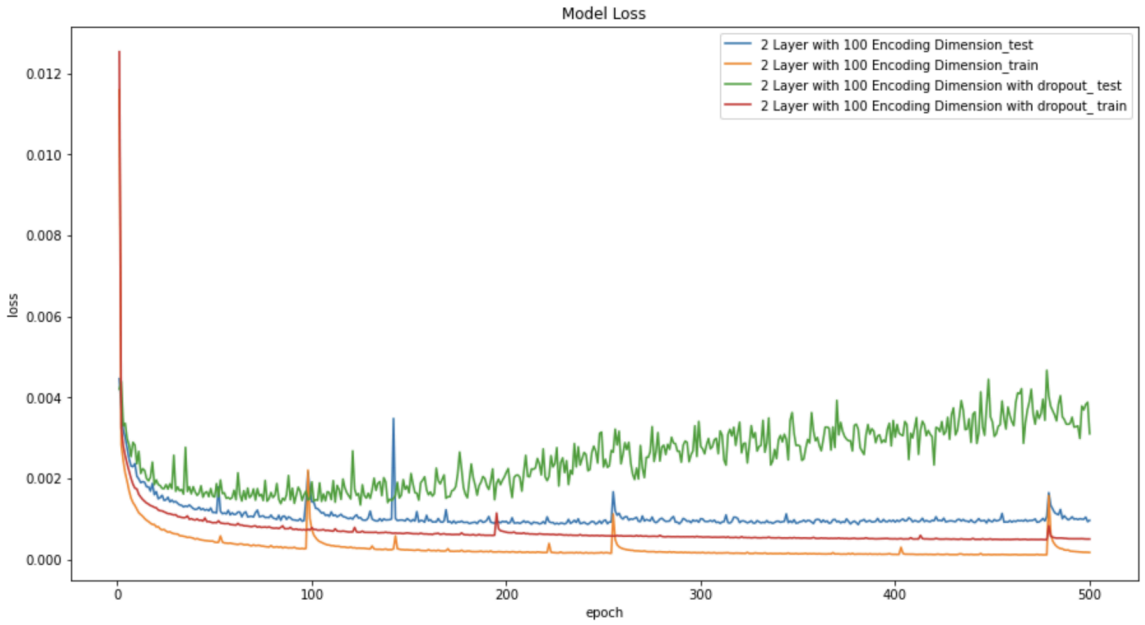


Figure 24. 2 layer GRU autoencoder with / without dropout

From the above plot on the GRU autoencoders' training and test loss, we can arrive at the following observations: The autoencoder with dropout has worse performance than the autoencoder without dropout in both training and test losses. Especially for the test loss, the model's performance is hurt and there is a rise in the test loss as the epochs increase. It suggests that the model is yet to overfit, as there is also no sign of overfitting in the test loss from the autoencoder without dropout.

Due to the limit from time and computation resources, only autoencoders with a 2 layer GRU encoder (inclusive of input layer) are trained. If resources allowed, autoencoders with a 3 layer GRU encoder and a dimension of 300 would have been experimented. In view of this limitation, the 2 layer GRU encoder is tentatively selected as the optimal RNN model for unsupervised representation learning.

4.5.3 Adding Unsupervised Representation to the CNN Model

With the optimal encoders each consisting of dense layers and GRU layer, the melspec feature extracted as described in the previous section is passed into these encoders to generate unsupervised representation.

To add the unsupervised representation to the supervised models before the output layer, two parallel models are trained. Note that the outputs from the two parallel models are not from the conventional output layer with softmax activation. One model consists of the basic CNN model as described in the previous section. Another model takes the unsupervised representation as the input, which is followed by a fully connected layer and a batch normalisation layer. The output from both models are concatenated together and serve as the input for a final model. In the final model, the input is followed by a fully connected layer and the output layer with softmax activation.

Comparing Test Accuracies with/without Unsupervised Representation:

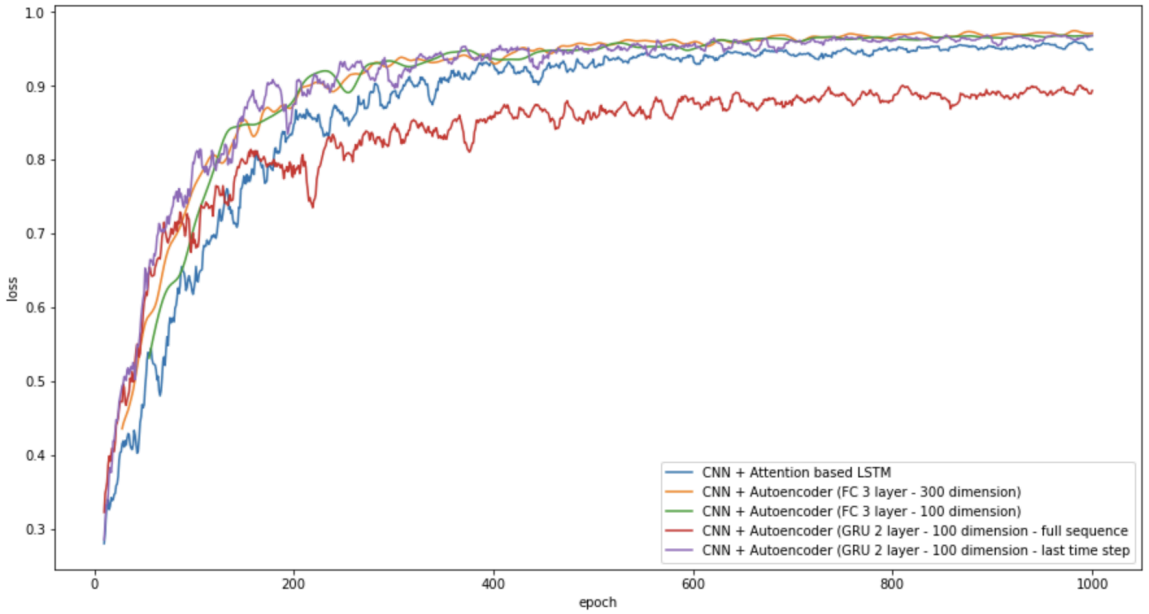


Figure 25. Test accuracies of CNN models with unsupervised representation from different autoencoders

Model	Best Accuracy
CNN + Attention based LSTM	96.574%
CNN + Autoencoder (FC 3 layer - 300 dimension)	97.545%
CNN + Autoencoder (FC 3 layer - 100 dimension)	96.866%
CNN + Autoencoder (GRU 2 layer - 100 dimension - full hidden states)	96.879%
CNN + Autoencoder (GRU 2 layer - 100 dimension - last hidden state)	90.064%

Table 6. Best accuracies of CNN models with unsupervised representation from different autoencoders

In the above plot, we made comparisons on test accuracy among:

1. The CNN + attention based LSTM model, as described in the previous section
2. The CNN model in parallel with the unsupervised representation from the FC 3 layer autoencoder with a dimension of 300
3. The CNN model in parallel with the unsupervised representation from the FC 3 layer encoder with a dimension of 100
4. The CNN model in parallel with the unsupervised representation from the GRU 2 layer encoder with a dimension of 100, output all hidden states

5. The CNN model in parallel with the unsupervised representation from the GRU 2 layer encoder with a dimension of 100, output the the final state

From the above comparison, we can arrive at the following observation:

1. Compared to CNN + LSTM, CNN + unsupervised representation from FC encoders has achieved higher accuracy and faster convergence. This has validated the effectiveness of unsupervised representation learning in assisting the supervised classification task at this feature dimension level.
2. For the CNN GRU 2 layer encoder's performance, using the full hidden states returned by the GRU layer has interfered with the model instead of improving it. Using the final hidden state has shown a better outcome, though the performance is still comparable to CNN + Attention based LSTM model despite showing a faster convergence.
3. For the models with unsupervised representation from FC encoders, the performances from dimensions of 100 and 300 do not vary much. It has shown that the dimension on a scale between 100 and 300 has little impact on the ability of representation learning for autoencoders.

Limitation:

On a side note, the performance of CNN + Attention LSTM may improve significantly on features extracted with higher dimension. To prove that the CNN + unsupervised representation could outperform CNN + Attention based LSTM, experiments with features of higher dimension need to be conducted. Due to time and resource constraints, this project only discusses the effectiveness of unsupervised representation at the current feature dimension level.

4.5.4 Effectiveness of Adding Unsupervised Representation to CNN + Attention based LSTM model

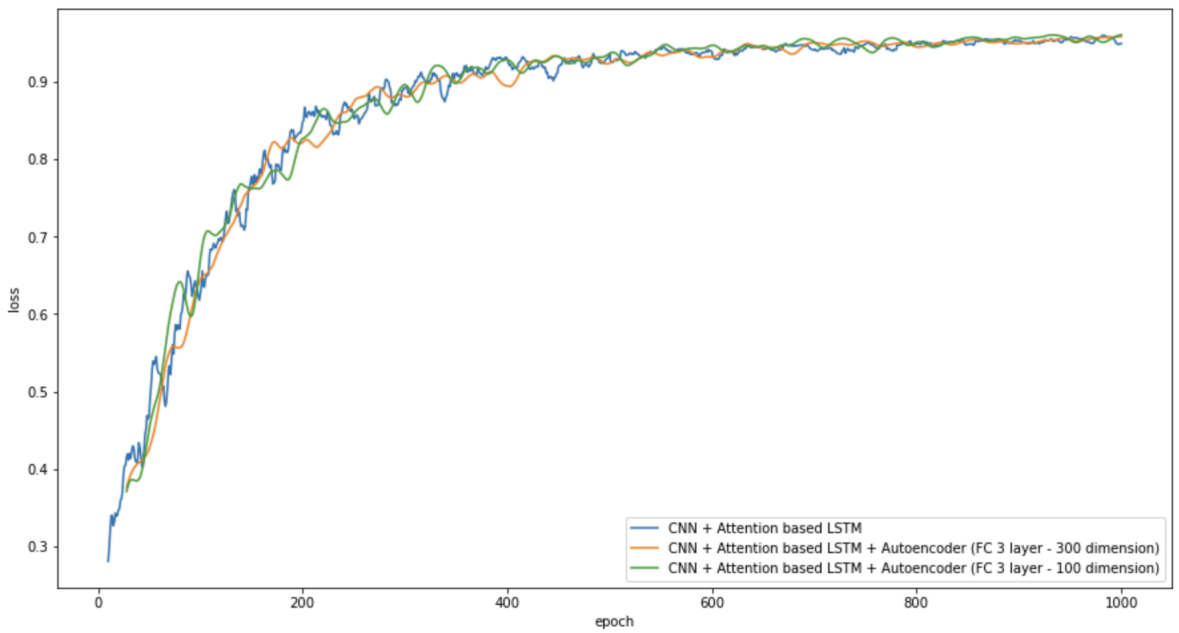


Table 7. Test accuracies of CNN + Attention based LSTM models with unsupervised representation from different autoencoders

Following the previous experiment, we continue to add the unsupervised representation to the CNN + Attention based LSTM, with the same parallel model architecture as the previous segment.

From the above test accuracy comparison, we found that the improvement on CNN + Attention based LSTM model is very little. Due to the increase of model complexity, overfitting might occur on the current feature dimension level. Therefore, the model with unsupervised representation is unable to generalise its prediction on the unseen data well.

Limitation:

To conclude how much improvement the unsupervised representation could potentially bring to the mode complex SER models, experiments with features extracted with higher dimension need to be conducted. Due to time and resources constraints, this project only discusses the effectiveness on the current feature dimension level.

4.6 Evaluation

4.6.1 Improvements on CNN based Model

From the above experiments, we have listed the best accuracies for each of the model architectures below:

Model	Best Accuracy
2D CNN	92.31%
CNN + LSTM	95.83%
CNN + Attention Based LSTM	96.30%
Optimal CNN + Unsupervised Representation (FC 3-layer Encoder with 300 dimension)	97.54%

Table 8. Best Accuracies for All Model Architectures

With the above data, it is observed that with each modification on the CNN based models, some improvement is achieved. As a summary of the discussion in the above sections, the results suggest that:

1. As suggested by previous works, CNN based models are capable of processing the audio spectrogram and extract features for SER tasks.

2. LSTM layer has boosted the CNN model's capability of extracting salient features from the audio signal.
3. With attention mechanism, the LSTM layer could focus on the parts of data that provides more useful information and hence its capability of extracting salient features is further improved.
4. The unsupervised representation learnt from an autoencoder could boost the basic CNN model's performance further.

4.6.2 Correlation Analysis on Emotion Intensity and Gender

Background

After evaluating the performance of above models we experimented with, it is observed that the model accuracy did improved by adding additional techniques on baseline 2D CNN, e.g. LSTM, Attention, the final accuracy we achieved increased around 4%.

Purpose

In this part, with this better model adjusted, CNN LSTM plus Attention, combining with real life experience and human beings intuition on audio emotion recognition, we did some evaluation and made comments on relations between data input features and classification result, including prediction result of emotion classification, certain input features correlation with classification correctness.

Model

We use CNN LSTM with Attention model here for evaluation, since it achieved the highest model accuracy among three, can better support our conclusion.

Dataset

For the dataset we still use RAVDESS, and Mel-spectrogram for feature extraction.
The number of chunks that Mel-spectrogram data was divided into is 2. And bands number to be 60, results in a data with shape to be 2*60*216. per data input. This is used as our input data to model in this evaluation. Also to increase data volume, augment signals (by adding AWGN) was being done before we calculated mel-spectrograms, thus the number of inputs increased from previous 1440 to 4320.
We randomly split data into 1:3, for testing and training, the following evaluation was being conducted based on 1080 testing data.
Hence the final shape of inputs is (1080, 2, 1, 60, 216).

Discussion

Figure 23 below shows the confusion matrix plotted based on testing results.

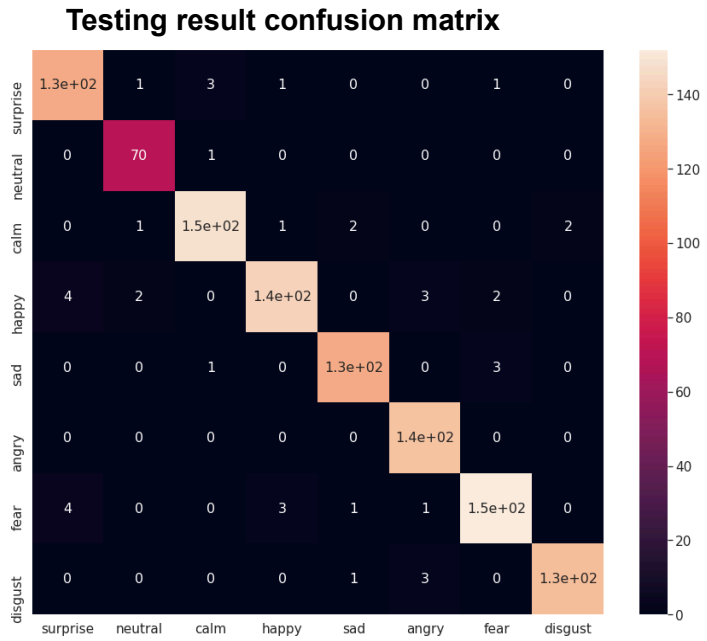


Figure 26. RAVDESS dataset only, Mel-spectrogram, Testing result confusion matrix

As shown in the confusion matrix in Figure 23, it is found that those in real life we considered as rather contrastive emotions, like angry, happy, calm were rather well classified. Emotions like neutral and sad are not well classified as compared with above ones.
There are some other points which deserve notice: surprise is often misclassified as happy or fear, this result is quite reasonable if we consider our emotions in real life: surprise in someway does have a common feeling as happy, e.g. birthday surprise, while it also often being hard to distinguish with fear, e.g. when people are fearful, they are shocked.

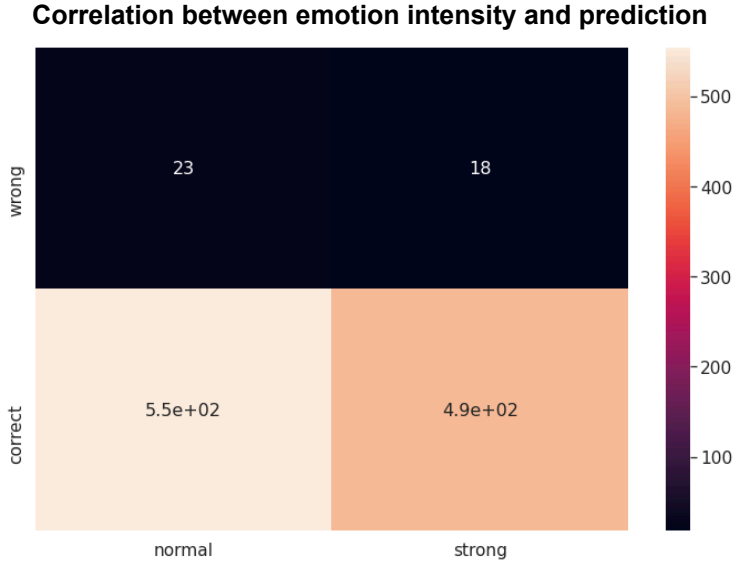


Figure 27. RAVDESS dataset only, Mel-spectrogram, Correlation between emotion intensity and prediction

Correctness of Normal Intensity: 95.99%
Correctness of Strong Intensity: 96.46%

By evaluation correlation between emotion intensity, i.e. normal emotion or strong one, and prediction result we obtained using test data, it is found that correctness of strong intensity is 96.46%, higher than that of correctness of normal intensity, which is only 95.99%. This result obtained was considered rather reasonable, for it is actually consistent with our real life experience. For example, the same emotion, angry, in real life someone with a very strong angry emotion will definitely be captured by us much easier and quicker than someone with a lower intensity of angry emotion. Actually in that case, just normal or a bit angry, others may not even realize it was a signal of anger.

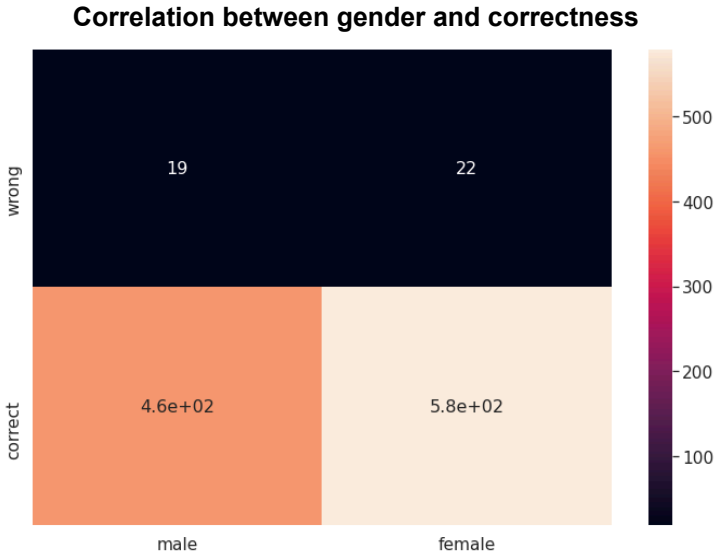


Figure 28. RAVDESS dataset only, Mel-spectrogram, Correlation between gender and correctness

Correctness of Male: 96.03%

Correctness of Female: 96.35%

As shown in the above figure, comparing correctness of male and female, it is clear that female emotion is easier to be correctly classified, this might be due to females usually expressing the emotions much more strongly than males in real life. This feature different from genders can also be observed from audio wave plots of male and female with the same emotion shown in previous 4.1 data exploration section.

In order to better illustrate and verify our findings mentioned above, we actually tried to listen to some audios of the same emotion but with different gender or with different emotion intensity levels. As human beings, we do feel i) it is easier to classify the emotion with stronger intensity label ii) females tend to have stronger emotion and vary more in pitch or volume in the same sentence compared to male. Hence the above evaluation was again verified and shows our model has a rather good learning performance.

5 Conclusion

In this project, we have conducted experiments on Speech Emotion Recognition with the basic 2D CNN models and its variants. We have shown improvements on the basic 2D model by separately adding an LSTM layer, an attention-based LSTM layer and unsupervised representation from autoencoders. Further work could be done to increase the

dimension of features extracted and to train more autoencoders with more complex structures and higher encoding dimensions. Inclusion of other speech datasets could also be done to validate the conclusion obtained in this project.

References

[1] S. Mirsamadi, E. Barsoum, C. Zhang.Ieeexplore.ieee.org. 2020. *Automatic Speech Emotion Recognition Using Recurrent Neural Networks With Local Attention - IEEE Conference Publication*. [online] Available at: <https://ieeexplore.ieee.org/abstract/document/7952552> [Accessed 23 November 2020].

[2] reasearchgate. 2020. *Speech Emotion Recognition From Spectrograms With Deep Convolutional Neural Network*. [online] Available at: <https://www.researchgate.net/publication/315638464_Speech_Emotion_Recognition_from_Spectrograms_with_Deep_Convolutional_Neural_Network> [Accessed 23 November 2020].

[3] Niu, Y., Zou, D., Niu, Y., He, Z. and Tan, H., 2020. *A Breakthrough In Speech Emotion Recognition Using Deep Retinal Convolution Neural Networks*. [online] arXiv.org. Available at: <https://arxiv.org/abs/1707.09917> [Accessed 23 November 2020].

[4] Mao, Q., et al. "Learning Salient Features for Speech Emotion Recognition Using Convolutional Neural Networks." Multimedia IEEE Transactions on 16.8(2014):2203-2213

[5] Lee, Jinkyu, and I. Tashev. "High-level Feature Representation usingRecurrent Neural Network for Speech Emotion Recognition." INTERSPEECH 2015.

[6] Fayek, H. M., M. Lech, and L. Cavedon. "Towards real-time Speech Emotion Recognition using deep neural networks." International Conference on Signal Processing and Communication Systems 2015:1-5.

[7] Trigeorgis, George, et al. "Adieu Features? End-to-end Speech Emotion Recognition using a Deep Convolutional Recurrent Network." ICASSP 2016.

[8] Kaggle.com. 2020. *RAVDESS Emotional Speech Audio*. [online] Available at: <https://www.kaggle.com/uwrfkaggler/ravdess-emotional-speech-audio> [Accessed 23 November 2020].

[9] Kaggle.com. 2020. *CREMA-D*. [online] Available at: <https://www.kaggle.com/ejlok1/cremad> [Accessed 23 November 2020].