Machine Learning Project Report*

Amrutha Varshini Mandalreddy, Herat Patel, Raakesh Murugaian and Shweta Wahane

1 OVERVIEW

Classification of instruments in the sound recordings provided in the Nsynth dataset [?] is performed using two different deep neural network models. One of the models is a convolutional neural network (CNN) and the other is a CNN Long Term - Short Term(LSTM) model. The accuracy, precision, and other analysis of the models are performed, along with the visualizations of the results. Some of the methods that have been used for the visualizations are the learning curves for the training and validation data, 1-D audio waveforms with correct class probability and near decision class probability, confusion matrix, Histograms of correctly classified instruments, and Histograms of misclassfied instruments. The neural network models are modified so as to obtain the highest accuracy possible.

For the convolutional neural network model, we create a model which has 3 convolutional network layers and two dense layers. Each of the convolutional layers is followed by a max-pooling layer with an input size of 2. The first convolutional layer is given a dropout of 0.2 while the other two are given a dropout rate of 0.5. Both the dense layers are provided a dropout of 0.25. The filter size of the Convolutional layers starts at 32 and is doubled in further convolutional layers while halved in the dense layers with the output layer containing 11 classes. All of the layers except the output layer undergo a ReLU activation function, while the output layer has a softmax activation function.

For the Second neural network model, an LSTM Convolutional neural network has been used. The model's architecture involves two convolutional layers and two dense layers. The convolutional layers have a max-pooling layer with a size of two along with one of them being given a dropout rate of 0.2 and the other given a dropout rate of 0.4. We define the LSTM model after the convolutional layers and before the dense layers. The dense layers are provided a dropout rate of 0.2 and 0.25. The filter size of the layers is started at 32 and doubled further in the next convolutional layer and then halved in the dense layers. Similar to the previous model all the layers are provided the ReLU activation function, while the output layer has a softmax activation function.
The accuracy of our first model i.e. CNN is 57%. The accuracy of our seccond model i.e. CNN LSTM is 61%.

2 TRAINING

For both the models, in the dataset preprocessing the rows with instrument "synthlead" is filtered, so the models are never trained with "synth–lead" data. In the CNN model, the id of the "vocal" instrument is retained and the model is trained to classify 11 classes. In the CNN LSTM Model, the id of the "vocal" instrument is changed to vacant 9 in both test, train and validation data and the model is trained with to classify 10 classes.

For both the models, the instrument type synth has been removed from the dataset prior to the training and the data is downsampled to 32000 examples.

2.1 CNN

For the training of both models, different hyperparameters have been used. The learning rate of the convolutional neural network model is set to 0.0001. Adam optimizer has been selected to be used while the criterion function used is the Sparse Categorical Cross Entropy. The accuracy of the model for every epoch is printed and the number of epochs that have been used for this model is 10. First, the set of Convolutional layer, MaxPooling layer and Dropout is added to the model twice. This is followed by Flatten layer. Then, a set of Dense and Dropout layers are applied twice. Finally, we have a Dense layer with Softmax activation function.

(1) Conv1D: This creates a convolutional layer which takes the number of filters, kernel size and activation function as input.
(2) MaxPooling1D: This layer downsamples the input by taking the maximum value within the pool size.
(3) Dropout: This is used to prevent overfitting by randomly setting input to 0 with a frequency of rate given.
(4) LSTM: This layer selects different implementations of the model based on the hardware, in order to improve the performance of the model.
(5) Flatten: This layer converts the input into a one dimensional vector.
(6) Dense: This creates a fully connected layer.
It takes the dimension of output space and the activation function as input.
(7) ReLu Activation function: It performs the operation max(x, 0) where x is each element in the input.
(8) Softmax Activation function: This is used for predicting the probability distribution.

2.2 CNN LSTM

For the second model that has been used, which is the LSTM convolutional neural network model, a learning rate of 0.0003 has been selected to be used. The optimizer that has been implemented is the Adam optimizer. Similar to the previous model the Sparse Categorical Cross Entropy function has been implemented to determine the loss. The number of epochs that have been used for this model is 10 while the accuracy for each epoch is calculated.

Model Parameters:

(1) Conv1D: Just like in our previous model, this layer creates a convolutional layer which takes the number of filters, kernel size and activation function as input.
(2) MaxPooling1D: This layer downsamples the input by taking the maximum value within the pool size.
(3) Dropout: This layer randomly sets input to 0 with a frequency of given rate. prevent overfitting by .
(4) Flatten: This layer flattens the input into a 1D vector.
(5) Dense: This creates a fully connected layer.
It takes the dimension of output space and the activation function as input.

*

(6) ReLu Activation function: It performs the operation max(x, 0) where x is each element in the input.
(7) Softmax Activation function: This is used for predicting the probability distribution.

3 RESULTS

3.1 CNN

The visualizations of the signal wave for each of the classes is provided. While the increase in epochs is benefitting in improving the accuracy of the training set, it is observed that a large number of epochs can be detrimental to the accuracy when it comes to the validation dataset. This can be correlated to overfitting. Hence a moderate number of epochs is selected for both of the models.

If number of epochs are less then this may underfit the model and if it is more then chances are that it will overfit the model. Hence, it is crucial to have the perfect number of epochs.

We can observe that the classes whose instruments produce a high-pitched sound have a signal waveform probability that is higher for the higher frequencies. This is seen in instruments such as the flute and keyboard. Whereas the instruments which produce a lower-pitched sound have a waveform that has probabilities higher in the lower frequencies. This tells us that our model is successful in differentiating between the instruments.

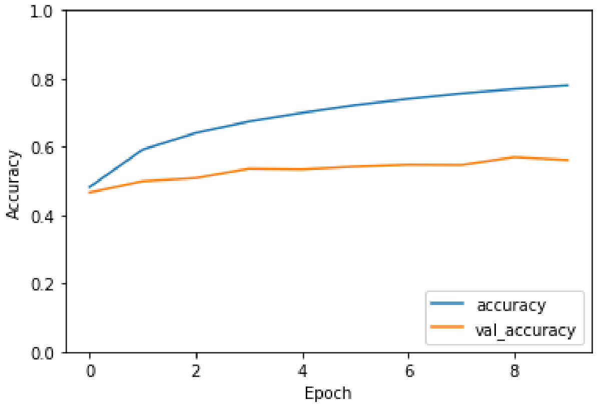

Figure 1: CNN:Training and Validation Learning Curves

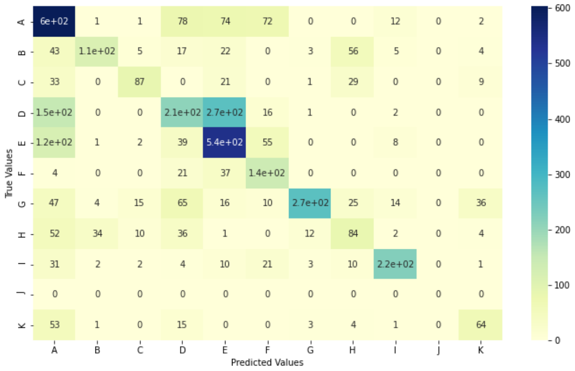

Figure 2: CNN:Confusion Matrix

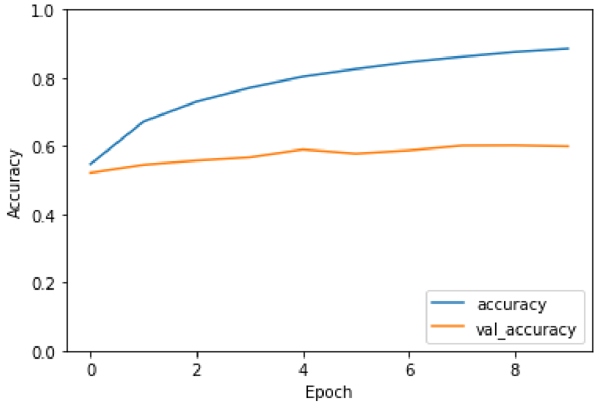

Figure 3: "CNN LSTM: Training and Validation Learning Curves"

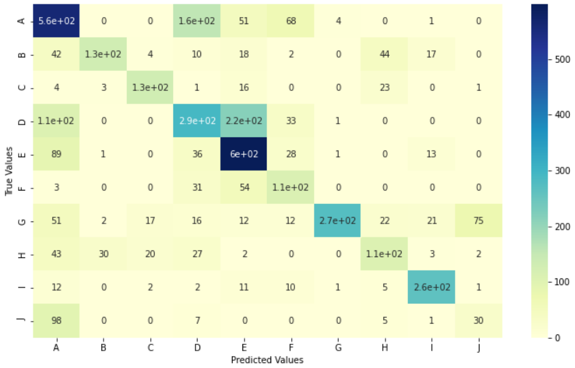

Figure 4: "LSTM: Confusion Matrix"

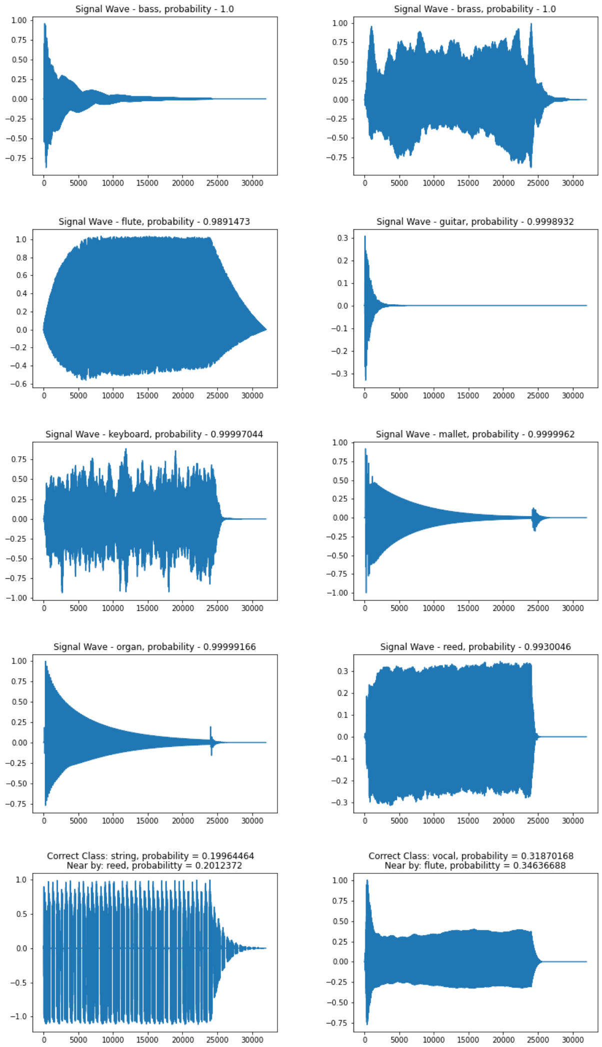

Figure 5: CNN: Images of each instrument's correct class probability

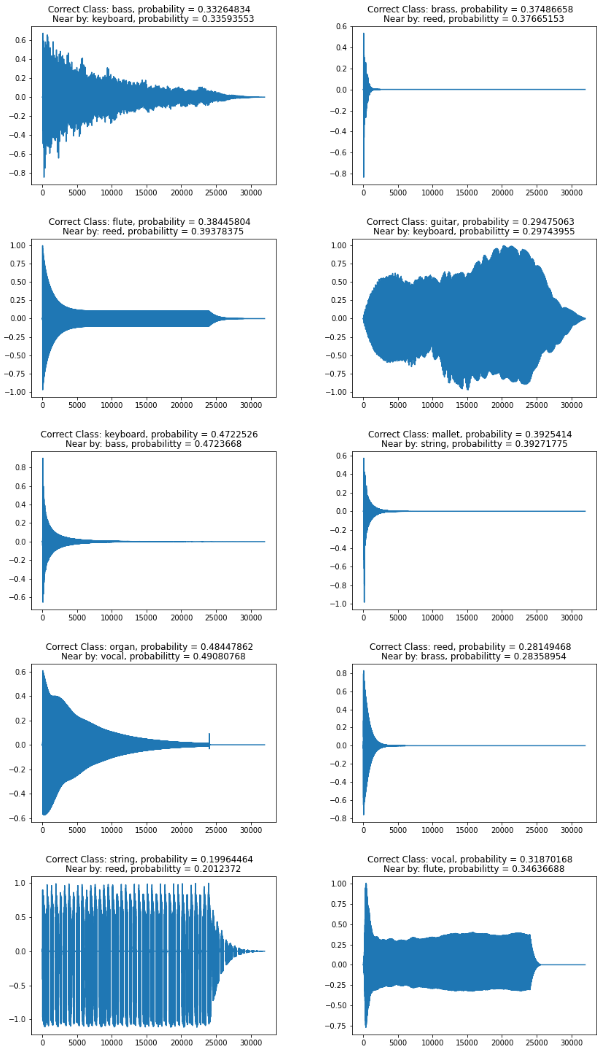

Figure 6: CNN: Images of each instrument's near the decision class probability

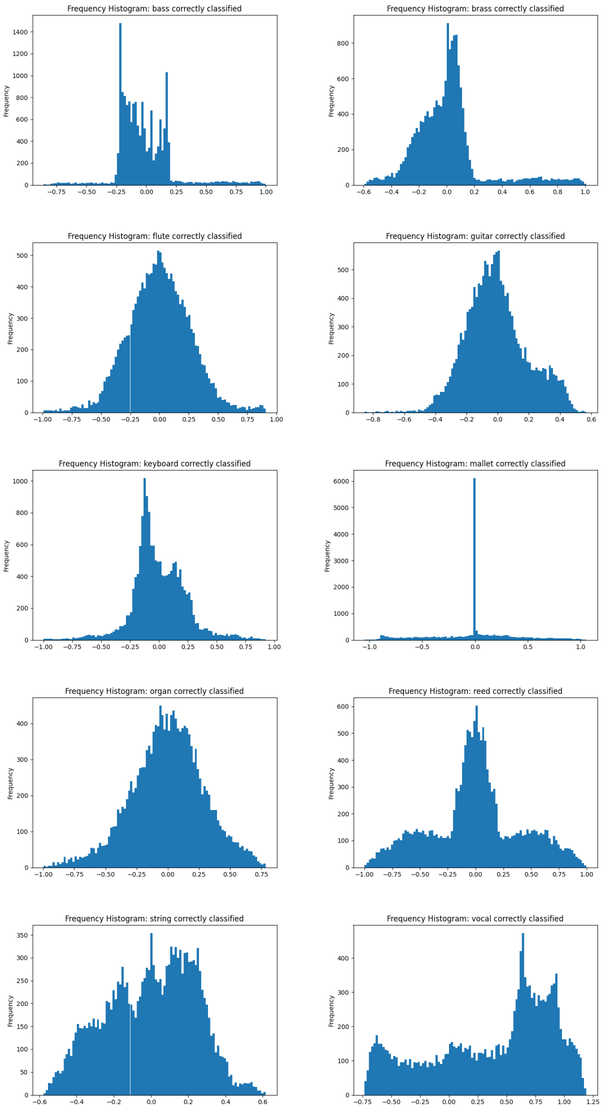

Figure 7: CNN: Histograms of each correctly classified instrument

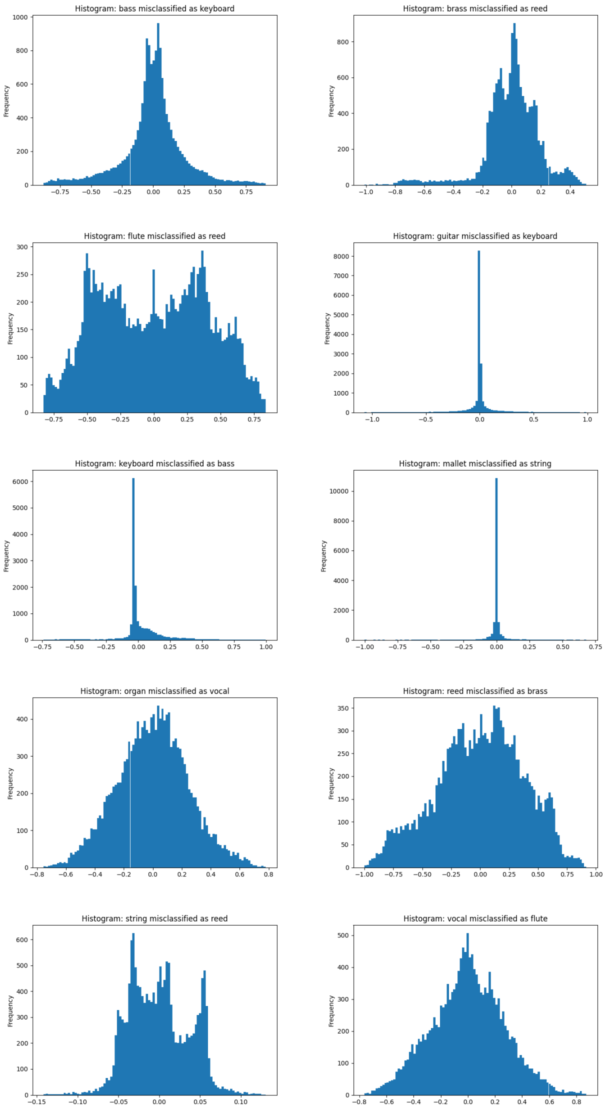

Figure 8: CNN: Histograms of each misclassified instrument

3.2 CNN LSTM

There are a few differences in the signal waveform of probabilities. The probabilites of keyboard is the most different as the waveform for the LSTM model suggests that higher frequencies might have lesser probability for it. This is the opposite of the result obtained in the CNN model. Other instruments such as bass are mostly, in line with the results obtained in the CNN model.

The acuracy of the LSTM model is then compared with the number of epochs. It is observed that the accuracy of both the training and validation dataset is higher than that of the CNN model. This can be attributed to some of alterations made to the model in comparison to the previous CNN model. However, similar to the CNN model, it is found that having an epoch value that is too high might cause a decrease in the accuracy. Therefore in order to prevent the overfitting, a reasonable number of epochs are used for the model. It can be deduced that with further modifications of the model an even better accuracy might be obtained. While the accuracy obtained for the validation dataset in the CNN model is 0.56962, the accuracy for the LSTM is increased due to above changes to a value of 0.6118.

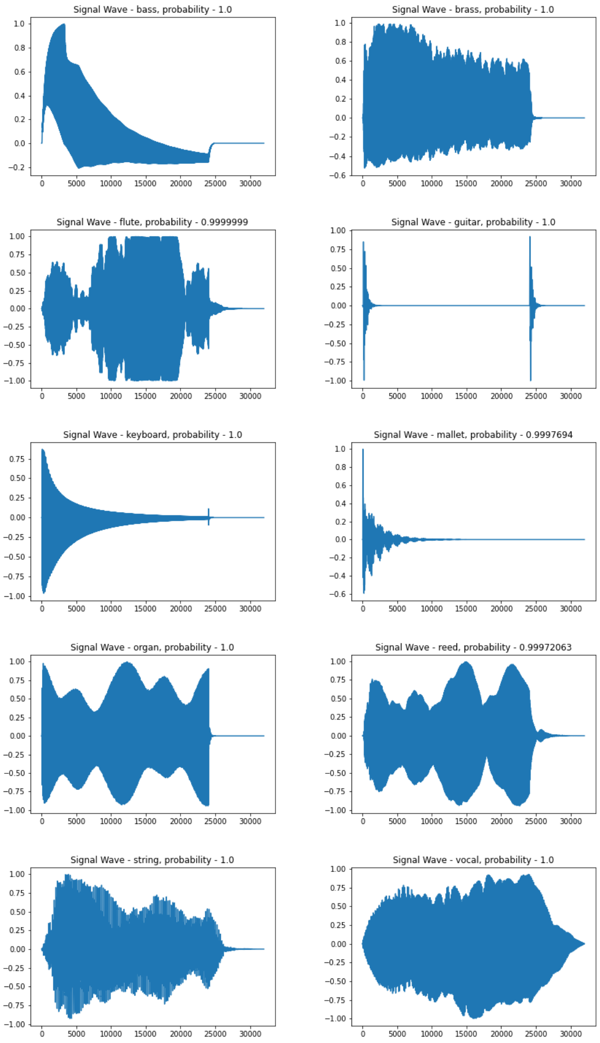

Figure 9: CNN LSTM: Images of each instrument's correct class probability

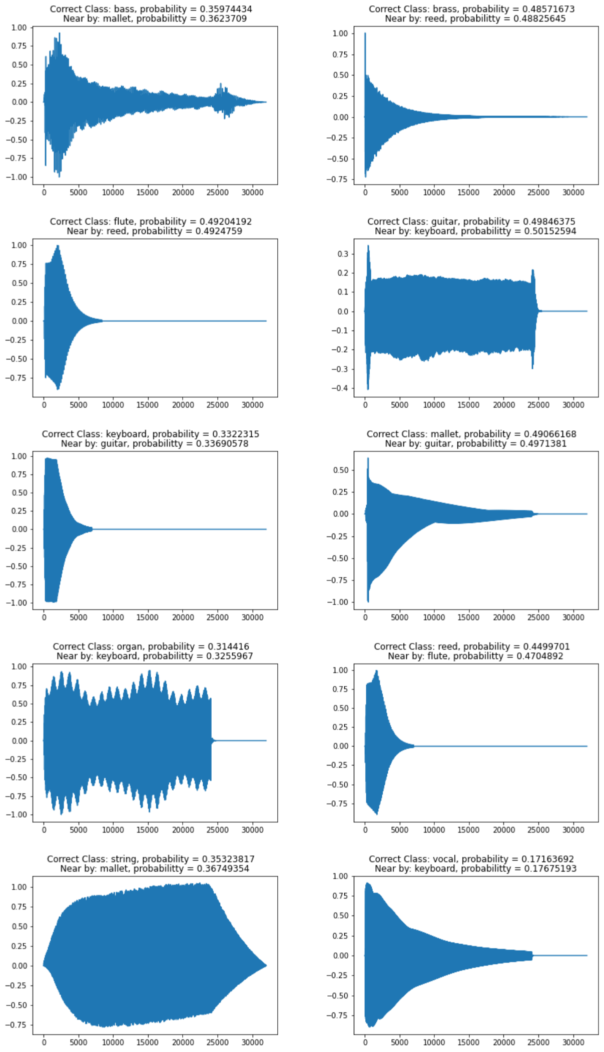

Figure 10: CNN LSTM: Images of each instrument's near decision probability

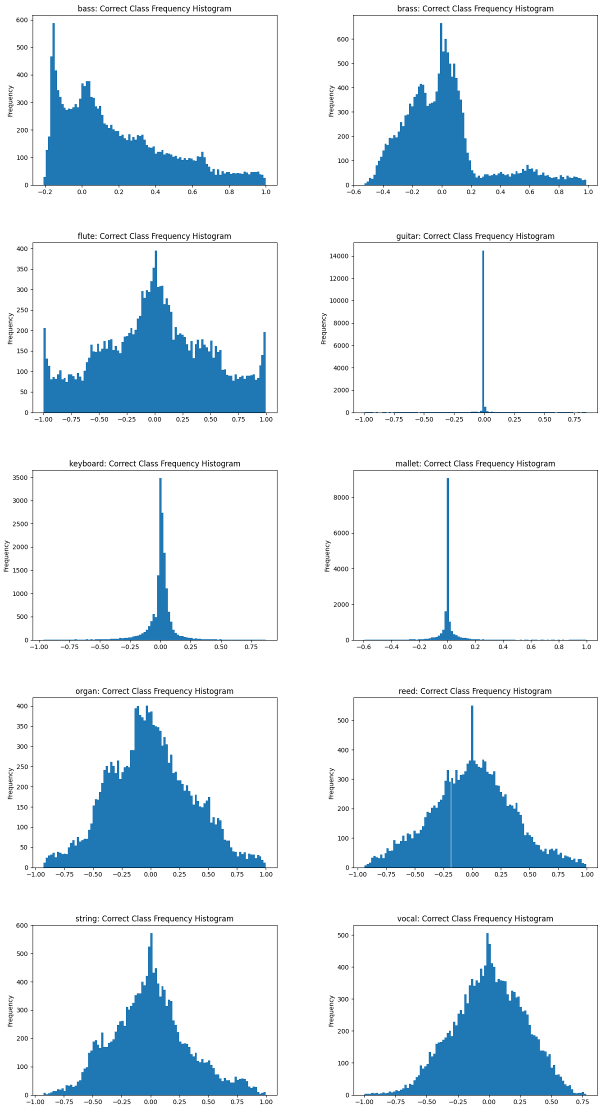

Figure 11: CNN LSTM: Histogram of each correctly classified instrument

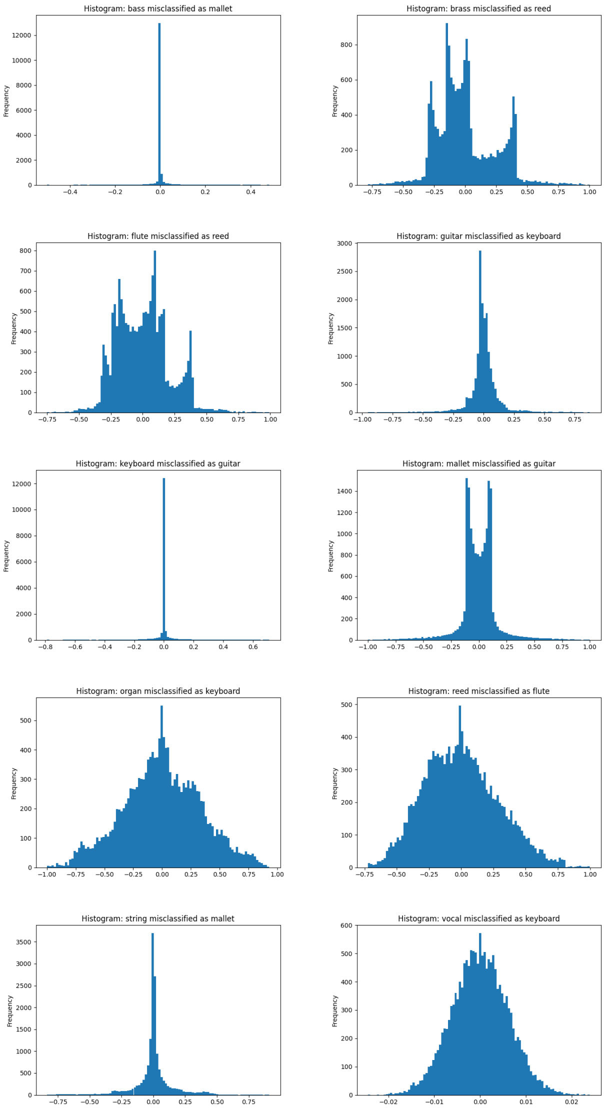

Figure 12: CNN LSTM: Histogram of each misclassified instrument

4 DISCUSSION

Our initial Model was the CNN Model. Initially, we worked on a more complex network and received lesser accuracy. The simpler we made the network by decreasing the layers which further reduced the number of trainable parameters, the better accuracy we received. The training for this model took 15 minutes for each epoch i.e 150 minutes in total for 10 epochs.

Our next model was the CNN LSTM Model. The training for this model took 25 minutes for each epoch i.e 250 minutes in total for 10 epochs. In comparison, the second one took almost twice the time taken for the CNN model. Our goal was to improve the accuracy provided by the CNN model i.e 57

CNN LSTM uses dilation and creates long term patterns. A certain number of modifications were made to the second model in order to obtain a more accurate model. One of the change made to the LSTM model is the decreased dropout rate that has been applied to both the convolutional and dense layers. This helps the model in detecting more features by not ignoring the outputs produced by a lot of nodes. We also observed that the learning rate of the model when set to 0.0003 instead of the value used for the CNN model (0.0001) yielded better accuracy in the results. The rest of the hyperparamters such as the activation functions and the loss functions that are implemented are kept similar in both the models, however, the number of layers is changed. The third convolutional layer is removed in the LSTM model and the definition of the LSTM is provided in its place. These changes to the model has resulted in an increase of 0.4 to the overall accuracy for the validation dataset from 0.56 in the CNN model to 0.61 in the CNN LSTM model.

The last second of the audio data is just the release of the note and contains only zeros. We tried training the models by removing the last 16000 samples which decreased the trainable parameters. It didn't help and further decreased the accuracy and did not even speed up the training process. If we do zero padding(i.e. padding='same'), those zeros at the end will anyways not matter [5].

In the visualizations, the audio files which are near the decision boundary have the similar 1-D waveform to the actual decisions and thus receive the similar probability. In addition to the 1-D waveforms, the histograms are plotted too. These histograms for the incorrect ones and the correct ones do not have much of a visible relationship, as the time matters for the audio. To plot the histograms the last 16000 samples which are only zeros are omitted to give a clear idea of the histograms.

REFERENCES

[1] Adrian Yijie Xu. Urban sound classification using convolutional neural networks with keras: Theory and implementation. February 2019.

[2] Muhammad Ardi. Musical instrument sound classification using cnn. August 2020.

[3] Swanand Mhalagi. The quest of higher accuracy for cnn models. May 2019.

[4] Michael Grogan. Cnn-lstm: Predicting daily hotel cancellations. September 2020.

[5] B W Shirlbert. Fft zero padding. February 2018.