A Comparative Evaluation of the Effectiveness of Mel Frequency Cepstral Coefficients and Difference Files for Audio Effect Identification Using Convolutional Recurrent Neural Networks

Patrick Sneyd
CCT College Dublin

Problem Identification

- Music information retrieval (MIR): computational systems that help humans better make sense of the processing, searching, organizing, and accessing of music related data.
- Takes in disciplines such as music theory, computer science, psychology, neuroscience, library science, electrical engineering, and machine learning.
- Mahana & Singh (2015) - audio classification has applications in fields such as:
  - Speech recognition.
  - Automatic Bandwidth Allocation.
  - Audio Database Indexing - useful for large audio collections in broadcasting facilities, the movie industry or music content providers.

Problem Identification - MIR Classifications

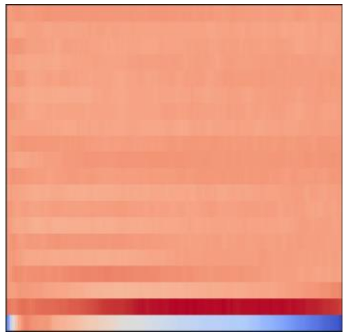
- Previous work uncovered on topic such as
  - Classifying audio into musical genres (Nasrullah and Zhao 2019)
  - Identifying individual instruments in musical snippets (Li et al. 2015)
  - Identifying the individual components of a drum kit (Jacques & Roebel 2018)

- No studies uncovered on identifying the various audio effects that can be applied to music.

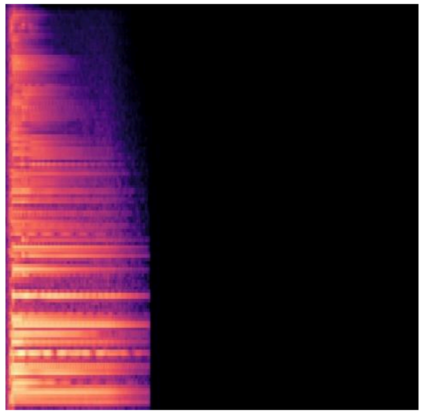
Problem Identification - Models and Image Inputs

- Early audio classification studies used SVM or KNN - neural networks specifically CRNN now predominant approach.

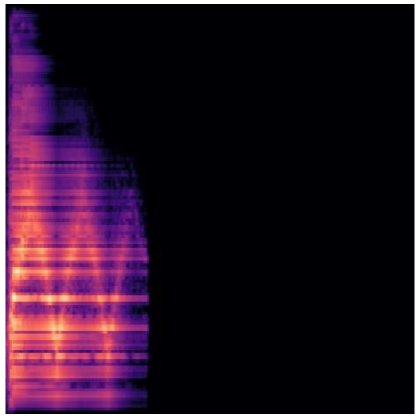
- Mel Frequency Cepstral Coefficient (MFCC) primary image input format.



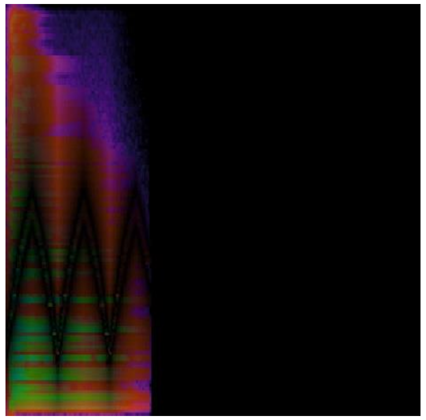
- This study also examined effectiveness of difference files (image displaying the absolute value of the pixel - by - pixel differences between two images) as image inputs.



Original File



Processed File



Difference File

Hypothesis

- Hypothesis One states that a CRNN can successfully utilise MFCC input images of effected audio files to classify the audio effect that has been applied to the audio file.

- Hypothesis Two states that a CRNN can successfully utilise difference file input images of effected audio files to classify the audio effect that has been applied to the audio file.

- Hypothesis Three states that a CRNN can successfully utilise both difference files and MFCC input images of effected audio files to classify the audio effect that has been applied to an audio file and that the model run using the difference files will outperform the model using the MFCCs based on the accuracy scores returned.

Data

- Freesound website (https://freesound.org) or came from the community edition of the Versilstudios VSCO 2 sound library (https://vis.versilstudios.com/vsco-community.html).

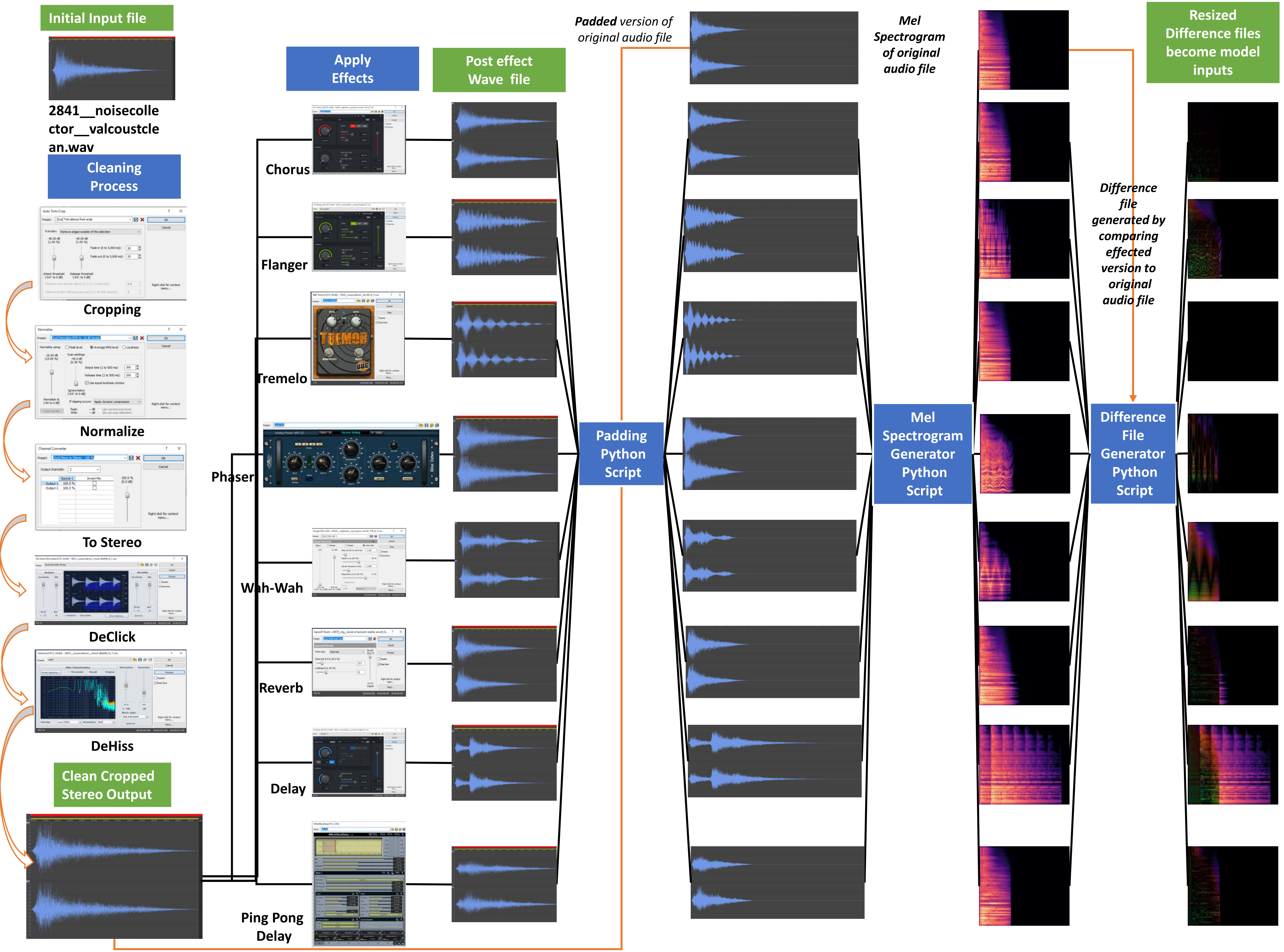
- The data set consisted of 8,920 samples when initially assembled and 7,648 of these samples were deemed suitable.

- All released under various creative commons licences.

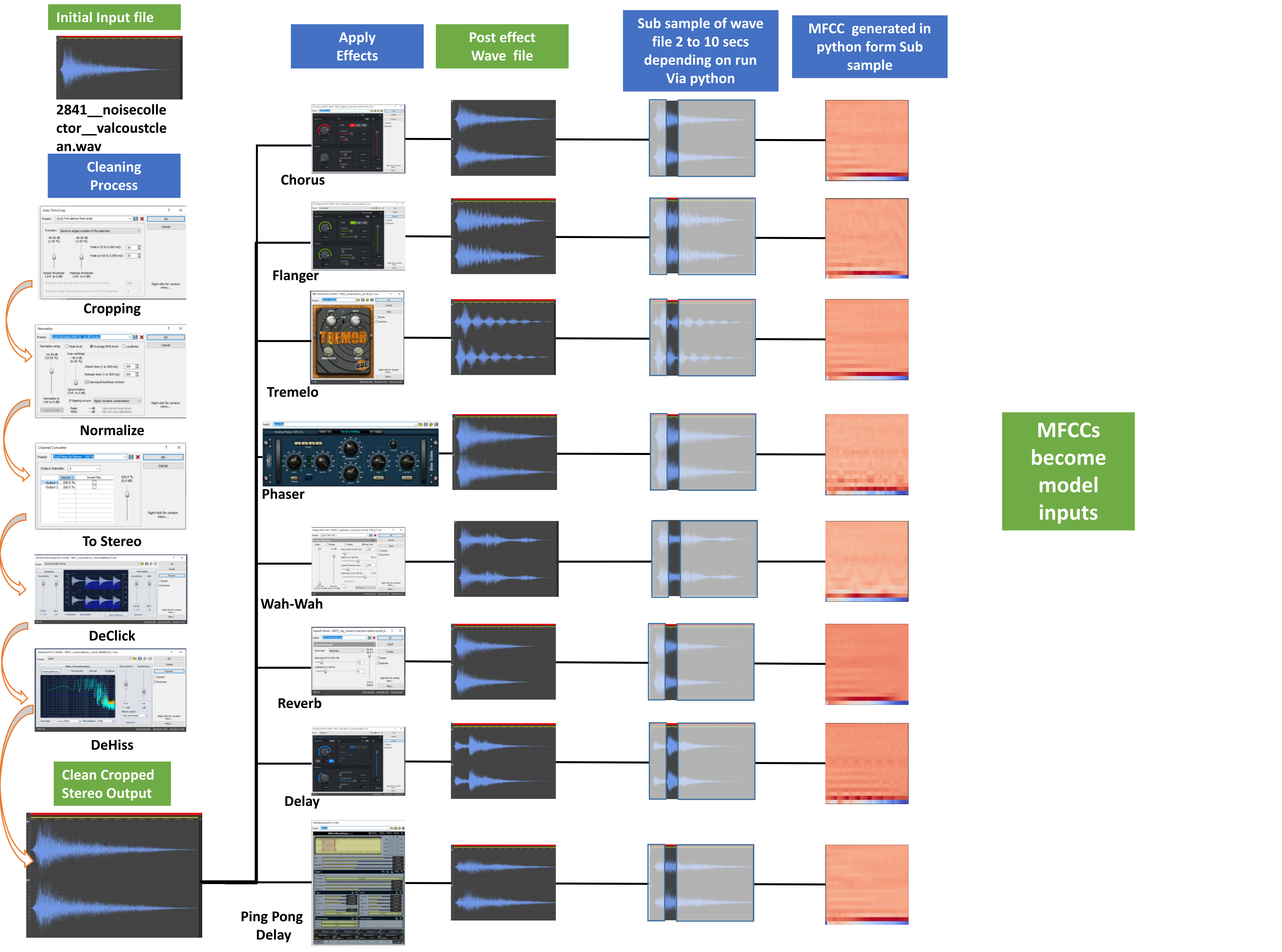
- Each processed through 8 different audio effects – final data set consisted of 68,832 audio files.

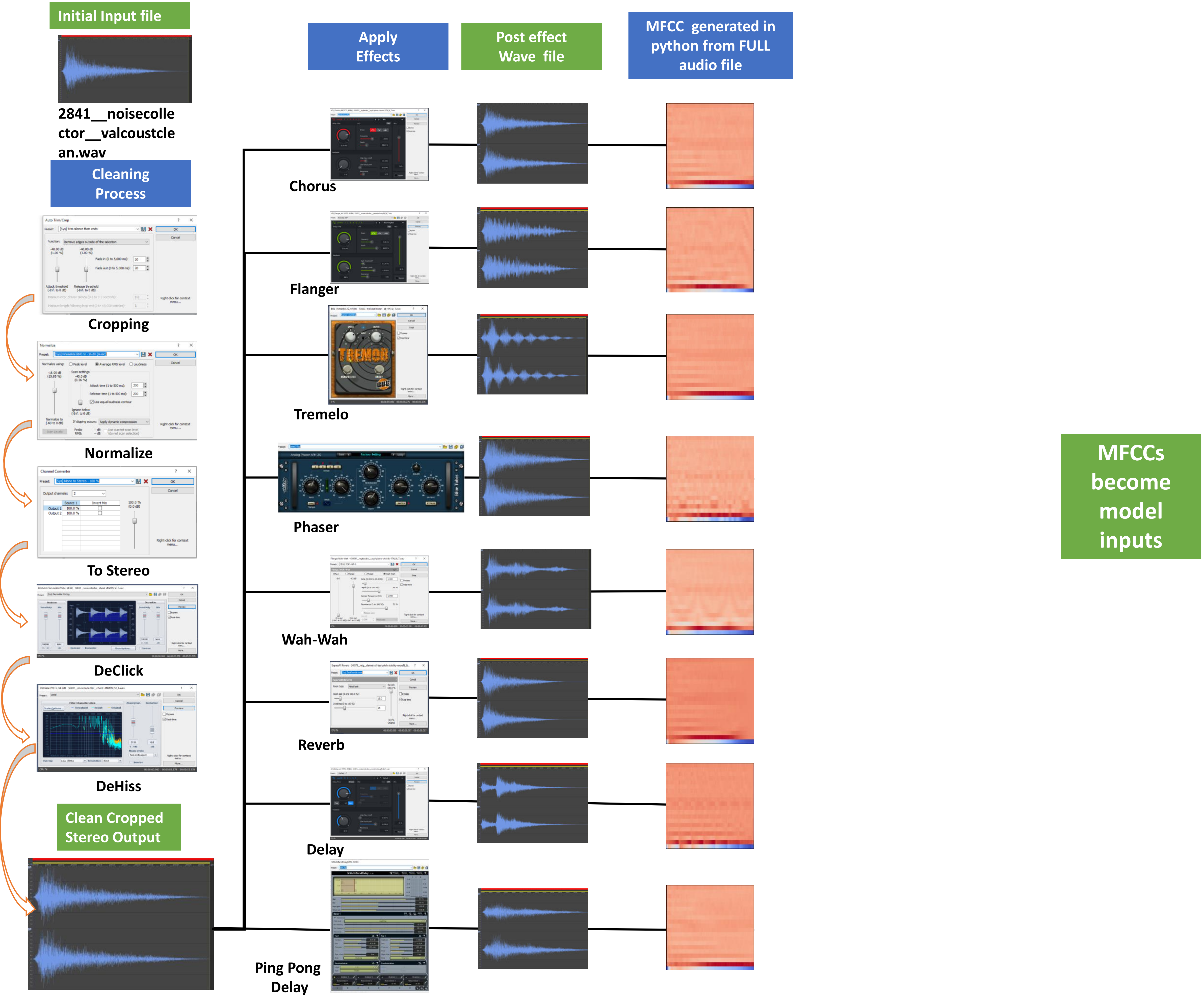
Model Inputs

- Three sets of inputs
  - Full length MFCC files.
  - Full Length difference files.
  - Sub Samples as MFCC files.



Initial Input file
2841__noisecollector__valcoustclean.wav
Cleaning Process
Cropping
Normalize
To Stereo
DeClick
DeHiss
Clean Cropped Stereo Output
Apply Effects
Chorus
Flanger
Tremelo
Phaser
Wah-Wah
Reverb
Delay
Ping Pong Delay
Post effect Wave file
Padded version of original audio file
Padding Python Script
Mel Spectrogram of original audio file
Mel Spectrogram Generator Python Script
Difference file generated by comparing effected version to original audio file
Difference File Generator Python Script
Resized Difference files become model inputs

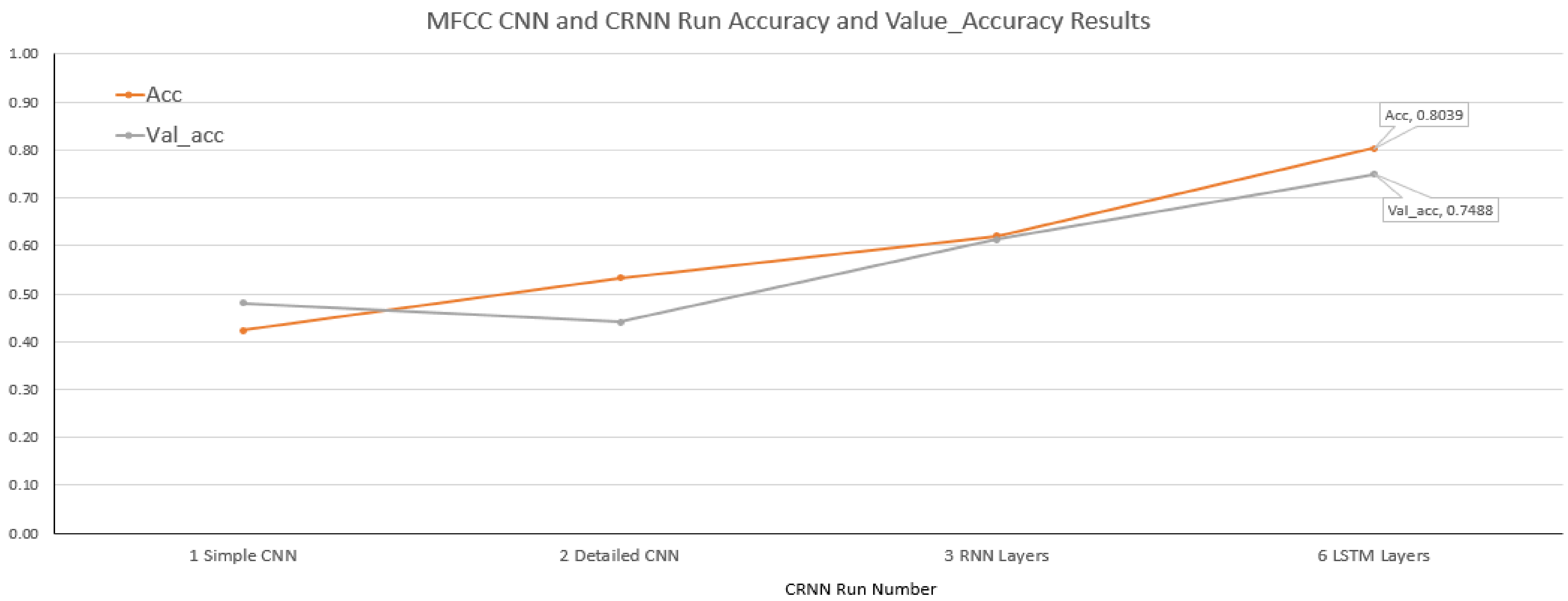

Initial Input file
2841__noisecollector__valcoustclean.wav
Cleaning Process
Cropping
Normalize
To Stereo
DeClick
DeHiss
Clean Cropped Stereo Output
Apply Effects
Chorus
Flanger
Tremelo
Phaser
Wah-Wah
Reverb
Delay
Ping Pong Delay
Post effect Wave file
Sub sample of wave file 2 to 10 secs depending on run Via python
MFCC generated in python form Sub sample
MFCCs become model inputs



Initial Input file
2841__noisecollector__valcoustclean.wav
Cleaning Process
Cropping
Normalize
To Stereo
DeClick
DeHiss
Clean Cropped Stereo Output
Apply Effects
Post effect Wave file
MFCC generated in python from FULL audio file
Chorus
Flanger
Tremelo
Phaser
Wah-Wah
Reverb
Delay
Ping Pong Delay
MFCCs become model inputs

Modelling

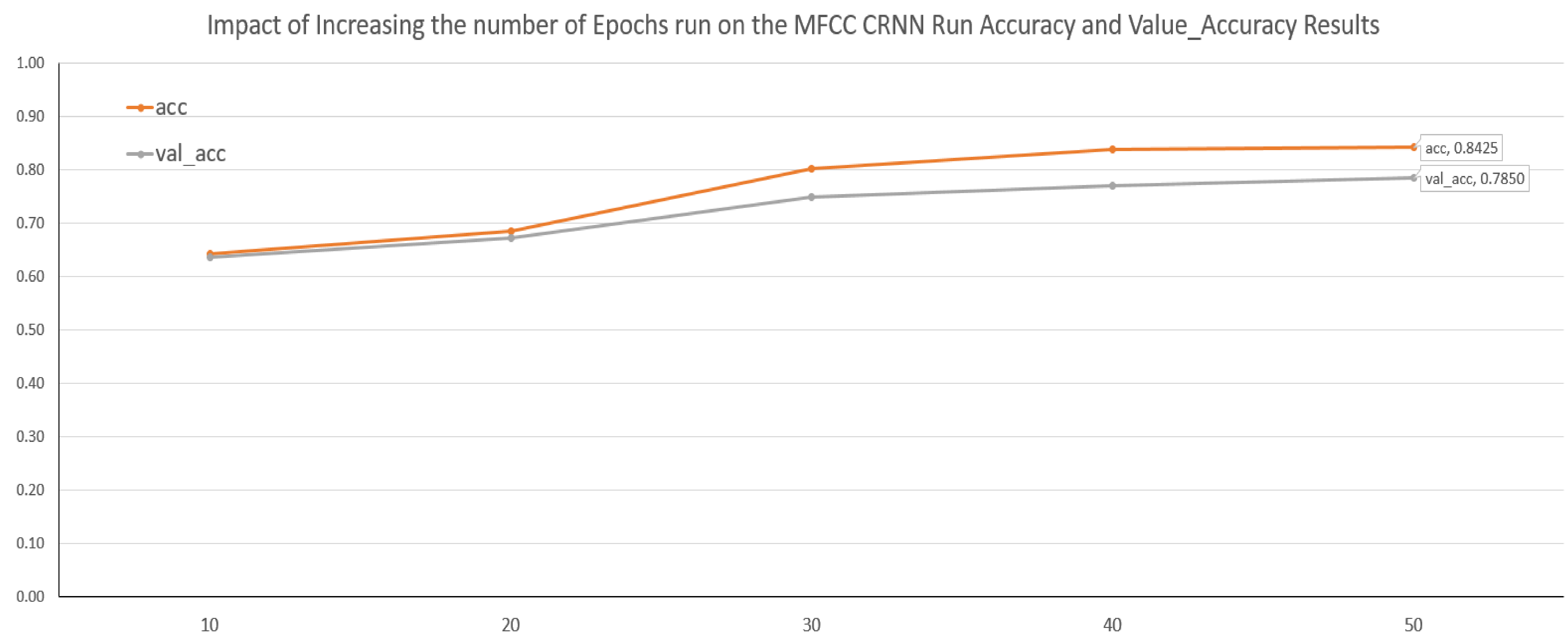
- Started with a relatively simple two convolutional layer CNN
  - Softmax activation function
  - Categorical crossentropy loss function
  - ADAM optimizer
- Next moved on to a more detailed four convolution layer CNN model to ensure that it returned an improvement in performance.
- Introduced the recurrent network elements via the use of simple RNN layers.
- Simple RNN layers were then replaced with LSTM RNN layers.

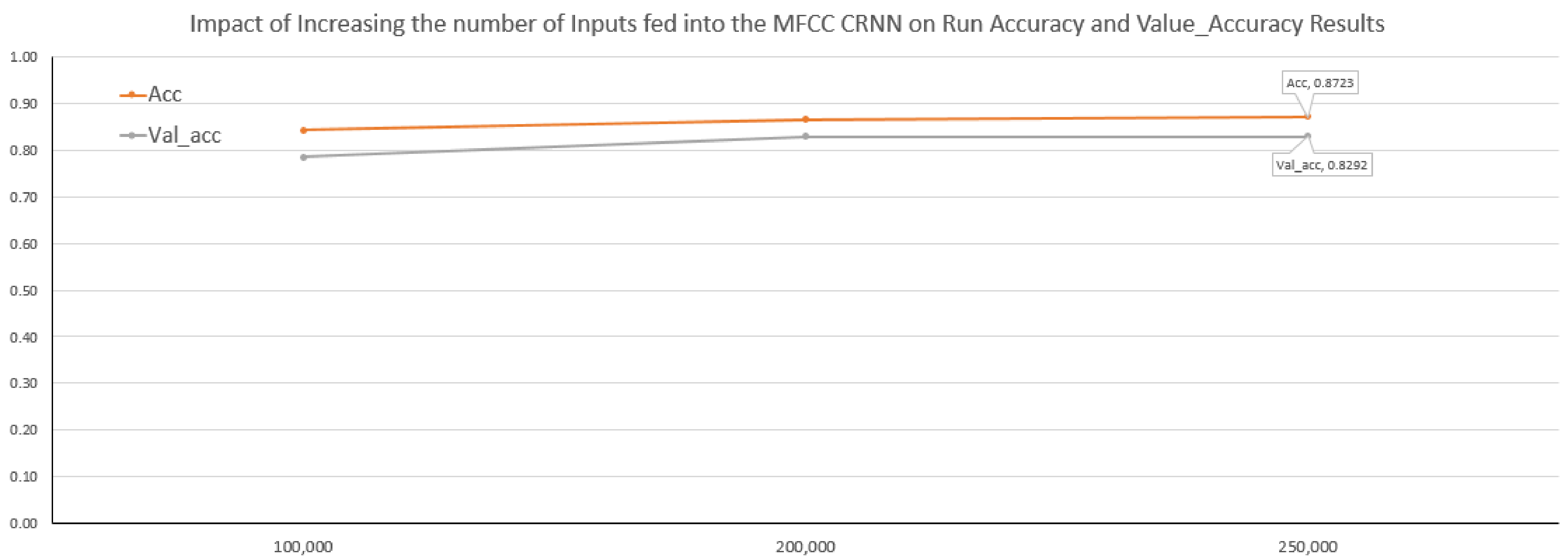


Selected Model

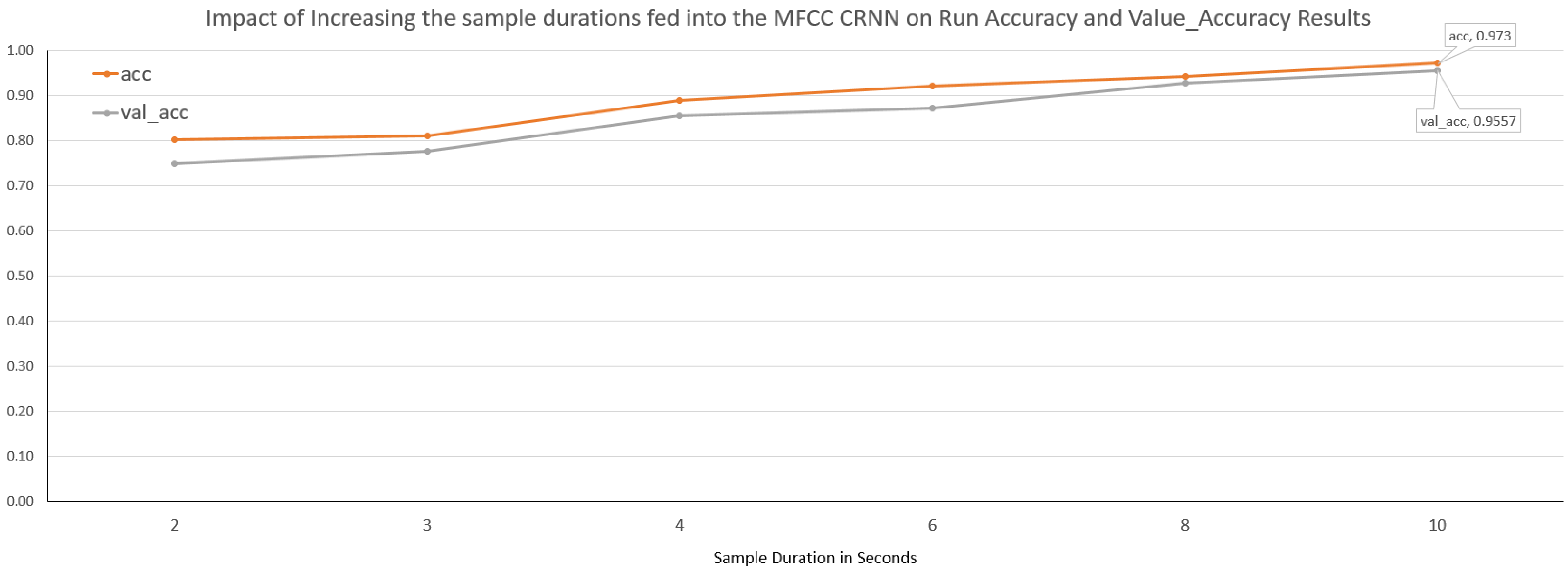
```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_44 (Conv2D)          (None, 275, 13, 32)       320

 max_pooling2d_36 (MaxPoolin  (None, 138, 7, 32)       0
 g2D)

 conv2d_45 (Conv2D)          (None, 138, 7, 64)        18496

 max_pooling2d_37 (MaxPoolin  (None, 69, 4, 64)        0
 g2D)

 conv2d_46 (Conv2D)          (None, 69, 4, 128)        73856

 max_pooling2d_38 (MaxPoolin  (None, 35, 2, 128)       0
 g2D)

 conv2d_47 (Conv2D)          (None, 35, 2, 256)        295168

 max_pooling2d_39 (MaxPoolin  (None, 18, 1, 256)       0
 g2D)

 lambda_5 (Lambda)           (None, 18, 256)           0

 lstm_6 (LSTM)               (None, 18, 64)            82176

 lstm_7 (LSTM)               (None, 18, 32)            12416

 lstm_8 (LSTM)               (None, 32)                8320

 dense_10 (Dense)            (None, 8)                 264

=================================================================
Total params: 491,016
Trainable params: 491,016
Non-trainable params: 0
_________________________________________________________________
```

Tuning Epochs and Input Numbers



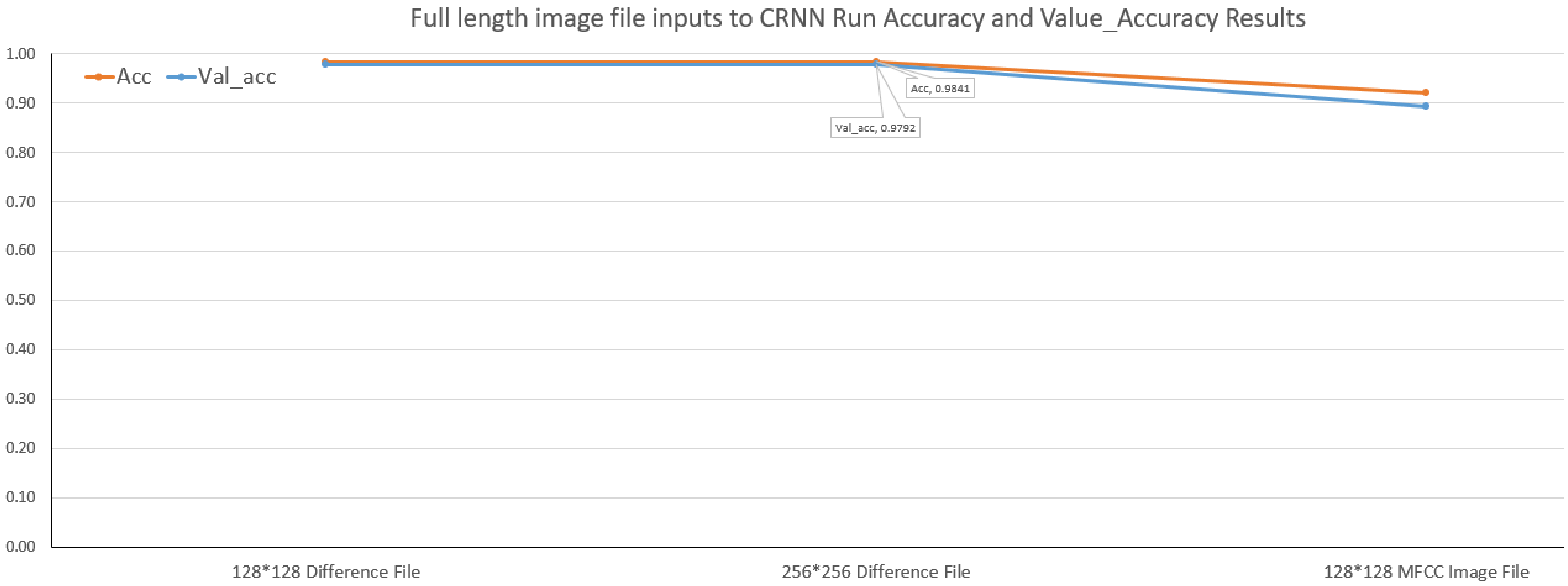


Tuning Sample Durations & Subsample results



Run No.	File	Details	Minimum sample length	Number of input files	Number of Samples	Number of Epochs	Acc	Val_acc
1	Run1_SimpleCNN.ipynb	Simple CNN	2	42,284	100,000	30	0.4228	0.4804
2	Run2-DetailedCNN.ipynb	Detailed CNN	2	42,284	100,000	30	0.5326	0.4407
3	Run3_Initial_CRNN-RNN layers_30_Epochs.ipynb	CRNN RNN layers	2	42,284	100,000	30	0.62	0.6128
4	Run4_Initial_CRNN_LSTM_10_Epochs.ipynb	CRNN LSTM Layers	2	42,284	100,000	10	0.6434	0.6373
5	Run5_Initial_CRNN_LSTM_20_Epochs.ipynb	CRNN LSTM Layers	2	42,284	100,000	20	0.6866	0.6742
6	Run6_Initial_CRNN_LSTM_30_Epochs.ipynb	CRNN LSTM Layers	2	42,284	100,000	30	0.8039	0.7488
7	Run7_Initial_CRNN_LSTM_40_Epochs.ipynb	CRNN LSTM Layers	2	42,284	100,000	40	0.8389	0.7703
8	Run8_Initial_CRNN_LSTM_50_Epochs.ipynb	CRNN LSTM Layers	2	42,284	100,000	50	0.8425	0.785
9	Run9_Initial_CRNN_LSTM_50_Epochs_200k.ipynb	CRNN LSTM Layers	2	42,284	200,000	50	0.8667	0.829
10	Run10_Initial_CRNN_LSTM_50_Epochs_250k.ipynb	CRNN LSTM Layers	2	42,284	250,000	50	0.8724	0.8292
11	Run11_Initial_CRNN_LSTM_30Epochs_3Secs.ipynb	CRNN LSTM Layers	3	36,257	100,000	30	0.8106	0.7774
12	Run12_Initial_CRNN_LSTM_30Epochs_4Secs.ipynb	CRNN LSTM Layers	4	28,913	100,000	30	0.8898	0.8555
13	Run13_Initial_CRNN_LSTM_30Epochs_6Secs.ipynb	CRNN LSTM Layers	6	15,739	100,000	30	0.923	0.8734
14	Run14_Initial_CRNN_LSTM_30Epochs_8Secs.ipynb	CRNN LSTM Layers	8	7,693	100,000	30	0.9433	0.9282
15	Run15_Initial_CRNN_LSTM_30Epochs_10Secs.ipynb	CRNN LSTM Layers	10	4,219	100,000	30	0.973	0.9557

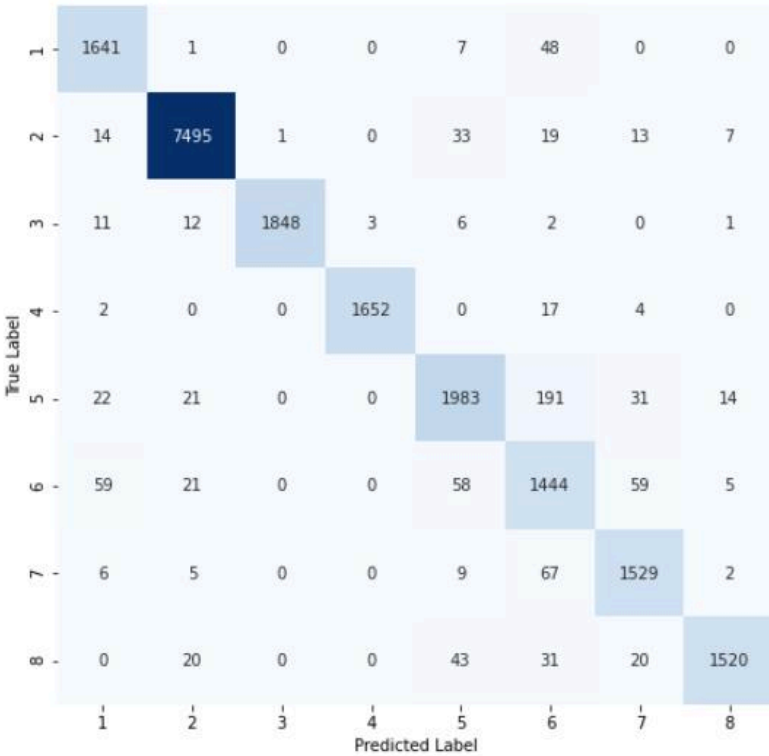
Full Length Image Inputs



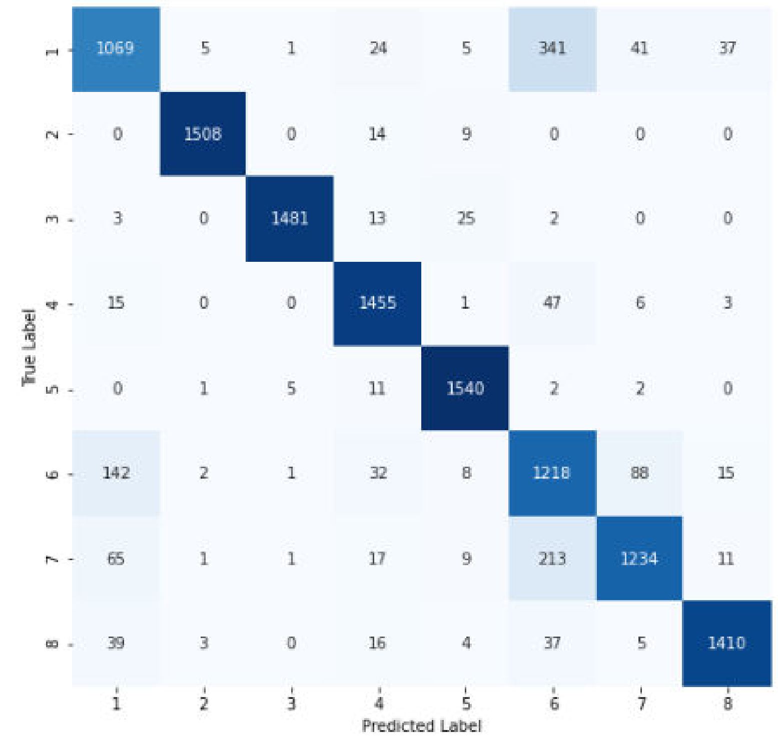
File	Details	Number of input files	Number of Epochs	Acc	Val_ acc
Run16-1_DiffFiles.ipynb	128*128 Difference File	61,184	10	0.9822	0.9783
Run17-2_DiffFiles.ipynb	256*256 Difference File	61,184	10	0.9841	0.9792
Run18-3_MFCCsImageInputs10Epochs.ipynb	128*128 MFCC Image File	61,184	10	0.9209	0.892

Selected Confusion Matrices

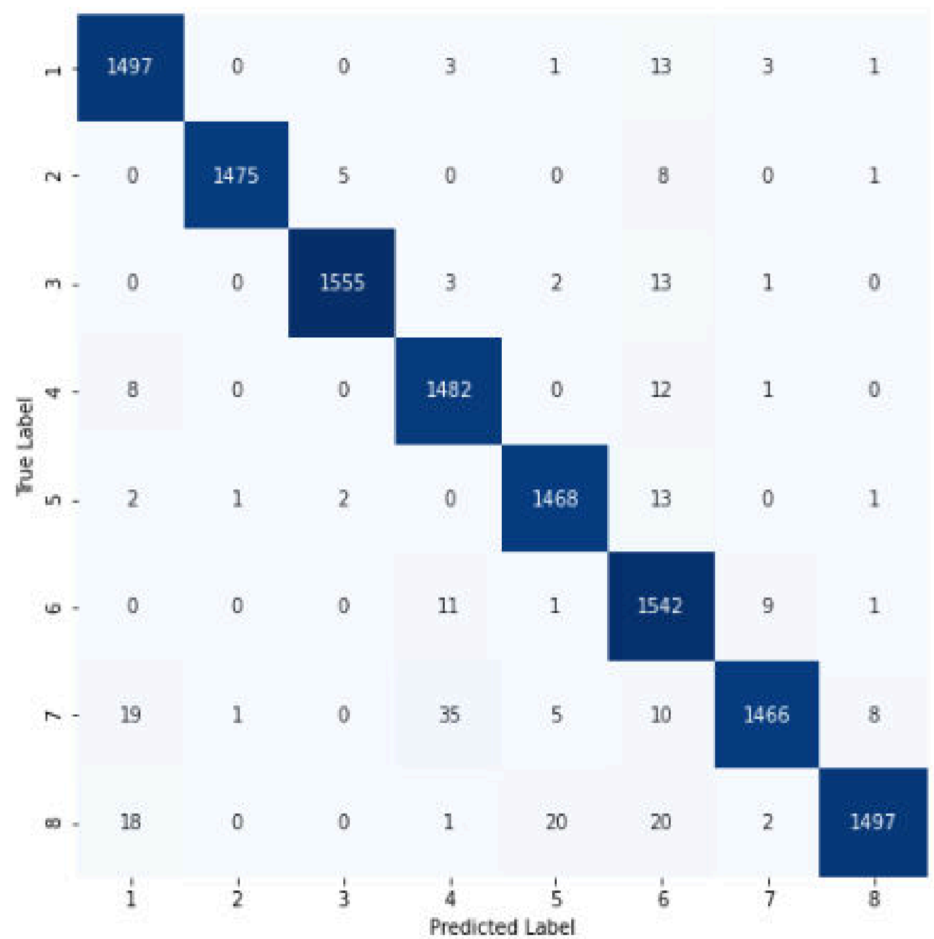
LSTM_30Epochs_10Secs - 97.3



128*128 MFCC Image File- 92

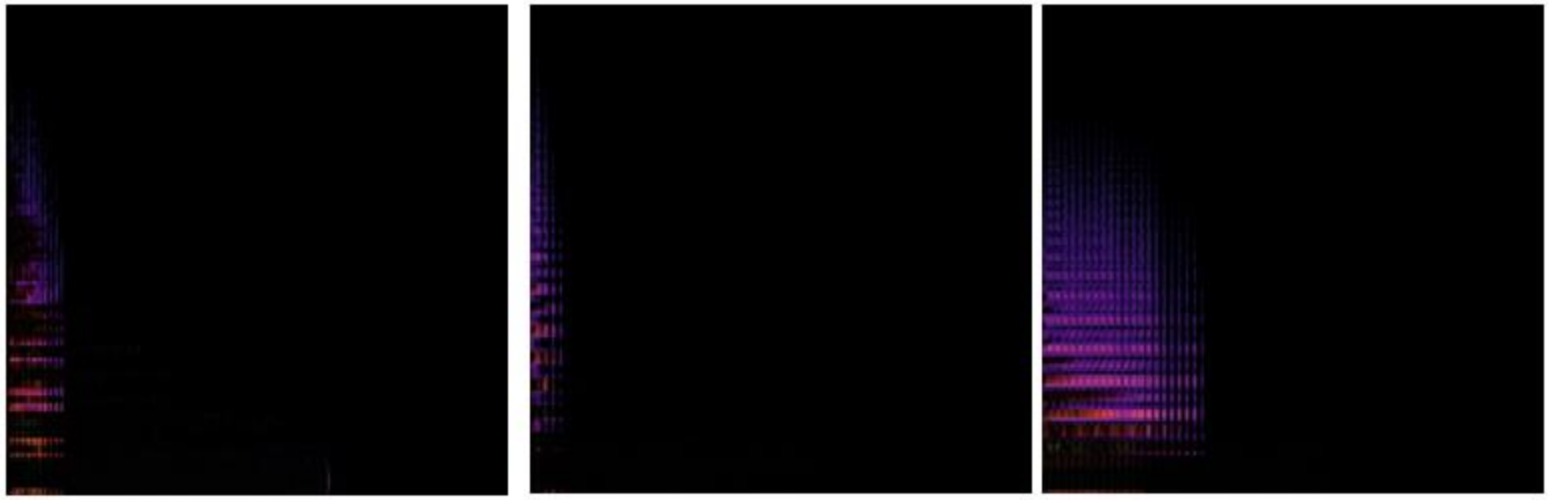


256*256 Difference File - 98.4



Conclusions

- All hypothesis accepted.
- Importance of the length of the audio file input provided to the model with longer inputs returning higher accuracies- importance of the LSTM layers effectiveness in learning long-range patterns.
- Success of difference file due to sound effects signature pattern.



Potential Areas for future study

- Potential for work on identifying multiple effects applied to multiple instruments in the one audio file.

- Investigate how the base image format used to generate the difference file e.g. Spectrogram, Mel Spectrogram, Scalograms, MFCC etc. affects the accuracy scores.

- There is also the potential to examine difference files effectiveness for classification activities in other fields where
  - An image is obtained of an item in one state
  - The item changes state in some way
  - An image of the item is obtained in this second state and thus a difference file can be generated by comparing image 1 to image 2 eg cancer growths .