AUDIO EVENT DETECTION AND CLASSIFICATION

N Bhuvan
190521

Vinamra Shrivastava
190968

ABSTRACT

Audio Event Detection and Classification is an important but rather challenging problem.In this report we use a different kind of machine learning models for given task.We made ConvNet, Recurrent neural network and linear models and feed them MFCCs features extracted from spectrogram representation of given audio files.

1. INTRODUCTION

The ability of Neural Network and Probability Models to learn discriminative spectro-temporal patterns makes them well suited to acoustic event classification. The Project Problem Statement has been divided into 2 tasks. Where task-1 is given an audio file with silence in between audio frames we have to classify whether it contain music , speech or both music and speech. For task-2, given an audio file containing sequence of audio events we have to predict the onset and offset time of those events along with the class of those audio events. We basically focus on task-2 and it will let us solve both task through that as audio file for both are of same kind

2. DATA-SET COLLECTION

As one of the project's objectives, we have collected around 80 clips of 10 seconds for music and speech by handpicking each audio clip from the internet and converting it into a .wav file. But, after a few days of searching the internet, we stumbled across a beautiful dataset (GTZAN Music/Speech Collection) of 180 clips of .wav files for each class. We also made a silent clip by recording the audio using a smartphone.

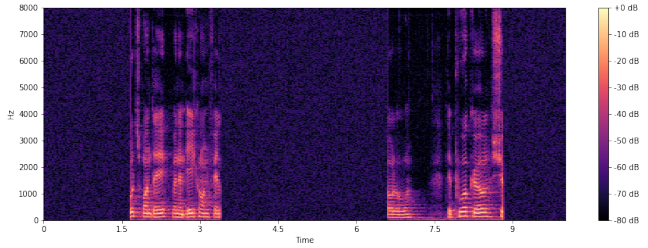
3. FEATURE EXTRACTION

We extracted Short-Time-Fourier-Transform from the audio file with sampling rate(sr/Fs) = 16000, window length = 1024, hop length = 512. The extracted STFT was a NumPy array of dimension (513) X (313).

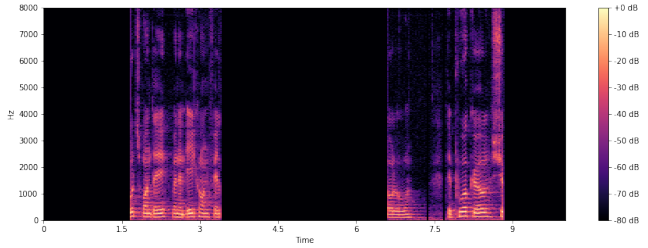
We know that the Mel-frequency spectrogram approximates the human auditory system, we then extracted the MFCC features from the spectrogram. Instead of finding the MFCC for the whole clip, we extract MFCC from every four consecutive frames of STFT. This step helped us get the MFCC feature for each clip of 4*10/313 = 0.127 seconds; we used a similar method to determine each class's Onset and Offset

4. NOISE REDUCTION WITH SILENCE SUPPRESSION

Dealing noise is expected in most of the signals. We tried to achieve a machine learning model of different types with the noisy STFT as the input; our attempt to achieve this was met with a model with very low accuracy. But, during this attempt, we found out that our model was working fine with inputs of just noisy music or noisy speech, but it failed to classify a silence with noise as silence. So, we made a noise reduction function that sets all the frames classified as noise with a constant value of -80dB. Using this noise reduction function, we suppressed the silence containing noise and got the STFTs from which we extracted MFCCs. Our model is achieved in a way by just silence suppression. By doing this, our model was able to classify each MFCC more accurately.



(a) Spectrogram Before Noise reduction



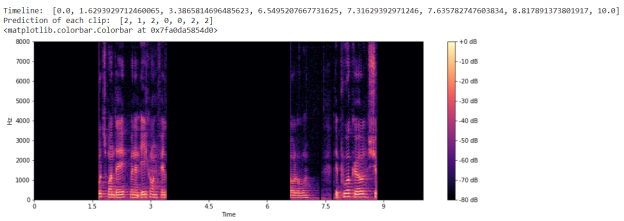
(a) Spectrogram After Noise reduction

Fig. 1. Example of an audio with Spectrogram before and after noise reduction

5. EVENT DETECTION

Using the noise reduction function, we suppressed the noise present in the silence between Music-Music, Music-Speech,

Speech-Speech, Speech-Music. We made a function that takes the noise-reduced STFT and returns a list consisting of the locations of the frame from the 313 frames where the audio changes from either silence to Music/Speech or from Music/Speech to speech. These frames indeed give us information regarding the onset and offset for Music and Speech. By getting this information, we were able to detect the occurrence and end of an event.



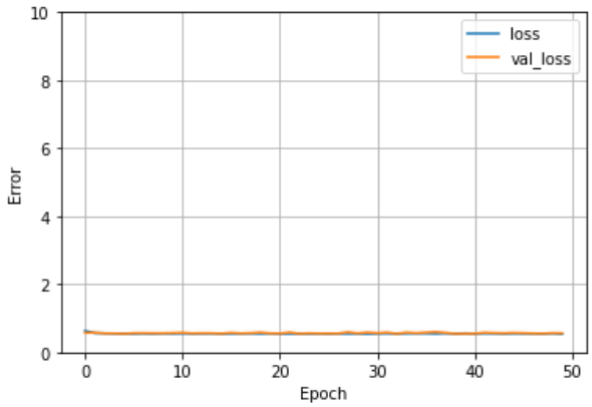
(a) Spectrogram with Onset and Offset

Fig. 2. Example of event detection for an audio file

In the above figure, for prediction: 0 represents Music, 1 represents Speech and 2 represents Silence. Here we can see that our model predicted and also detected audio events.

6. LINEAR MODEL

We trained a basic Linear model to classify the given MFCC as a one-hot vector. The maximum of the three is used to classify even though our linear model has an accuracy of 80

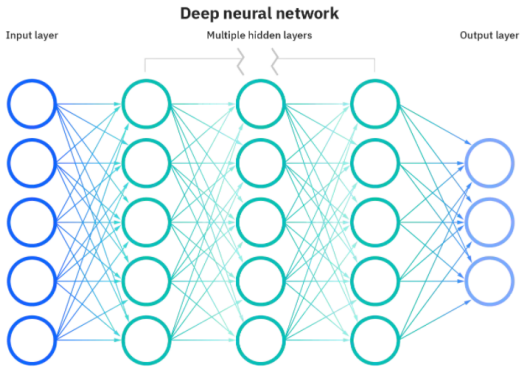


Error of our Linear Model while training

Fig. 3. Linear Model Loss for both Validation and Training Set

7. NEURAL NETWORK

Artificial neural networks (ANNs) are comprised of a node layers, containing an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network. Think of each individual node as its own



Neural Network

linear regression model, composed of input data, weights, a bias (or threshold), and an output. Ultimately, the goal is to minimize our cost function to ensure correctness of fit for any given observation. As the model adjusts its weights and bias, it uses the cost function and reinforcement learning to reach the point of convergence, or the local minimum. The process in which the algorithm adjusts its weights is through gradient descent, allowing the model to determine the direction to take to reduce errors (or minimize the cost function). With each training example, the parameters of the model adjust to gradually converge at the minimum. We have made our Neural Network model with dense and dropout layers passing MFCCs feature after flattening it into 1-D numpy array using Adam as optimizer and Categorical cross-entropy as loss function.

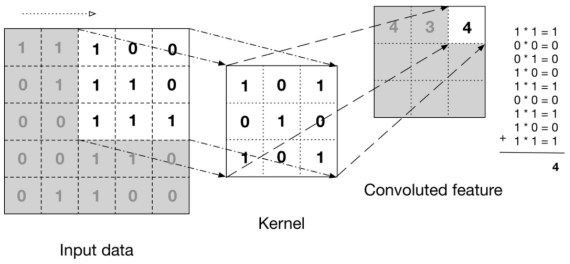
8. CONVOLUTION NEURAL NETWORK

In the past few decades, Deep Learning has proved to be a very powerful tool because of its ability to handle large amounts of data. One of the most popular deep neural networks is the convolutional neural network. CNN's were first developed and used around the 1980s. The most that a CNN could do at that time was recognize handwritten digits. The deep learning model requires a lot of data and computing resources. This was the major drawback of CNN at that period. ConvNet uses a special technique of convolution, bottom line is that role of ConvNet is two reduce higher dimension data into a form that is easier to process, without losing features that are critical for good prediction.

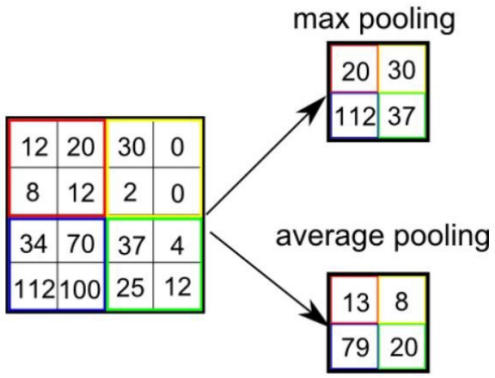
The Above image shows what a convolution is we take a filter/kernel(3x3) matrix and apply it to input image to get

```
Model: "sequential_15"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_6 (Conv1D)            (None, 62, 128)           6272

max_pooling1d_6 (MaxPooling  (None, 31, 128)           0
1D)

dropout_10 (Dropout)         (None, 31, 128)           0

flatten_9 (Flatten)          (None, 3968)              0

dense_18 (Dense)             (None, 64)                254016

dropout_11 (Dropout)         (None, 64)                0

dense_19 (Dense)             (None, 3)                 195

=================================================================
Total params: 260,483
Trainable params: 260,483
Non-trainable params: 0
_________________________________________________________________
```

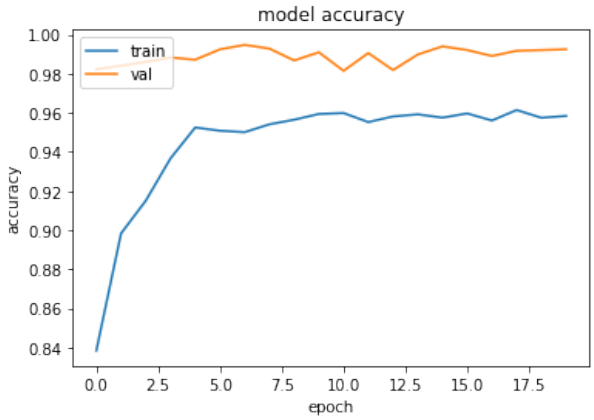
Model architecture



Convolution Operation



Pooling Operation



CNN-Model performance plot

the convolved feature. This convolved feature is passed on to next layer.

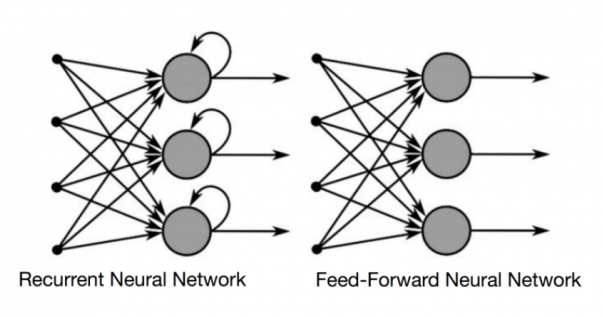
Pooling layer : Similar to the Convolutional Layer, the pooling layer is responsible for reducing the spatial size of convolved feature. This is to decrease the computational power required to process the data by reducing the dimensions. There are two type of pooling average pooling and max pooling. Here in our project we used Max pooling in between the convolutional layers. So what we do in Max Pooling is we find the maximum value of a pixel from a portion of the image covered by the kernel. Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction.

So what we are doing is after extraction of MFCCs feature from STFT feature we pass those MFCC features into ConvNet our model consist of convolution , pooling , dropout and Dense layers.

9. RECURRENT NEURAL NETWORK

Recurrent Neural Networks: The beauty of recurrent neural networks lies in their diversity of application. When we are dealing with RNNs they have a great ability to deal various input and output types. Lets see how the architecture of RNN looks like. Recurrent neural networks can form a much deeper understanding of a sequence and its context compared to other algorithms. Feed Forward neural network have no memory of the input they receive and are bad at predicting what's come next that's where RNN comes into play,In a RNN the information cycles through a loop. When it makes a decision, it considers the current input and also what it has learned from the inputs it received previously. A feed-forward neural network assigns, like all other deep learning algorithms, a weight matrix to its inputs and then produces the output. Note that RNNs apply weights to the current and also to the previous input. Furthermore, a recurrent neural network will also tweak the weights for both through gradient descent and backpropagation through time (BPTT). RNN has two issues Exploding Gradients and Vanishing Gradient Exploding gradients are when the algorithm, without much reason, assigns a stupidly high importance to the weights. Fortunately, this problem can be easily solved by truncating or squashing the gradients.

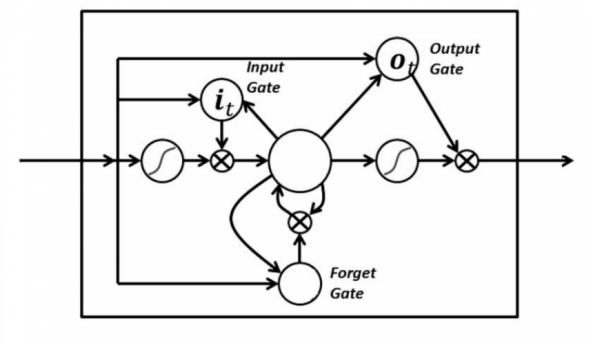
Vanishing gradients occur when the values of a gradient



RNN & NN

are too small and the model stops learning or takes way too long as a result. Fortunately, it was solved through the concept of LSTM by Sepp Hochreiter and Juergen Schmidhuber.

LSTM: Long short-term memory networks (LSTMs) are an extension for recurrent neural networks, which basically extends the memory. Therefore it is well suited to learn from important experiences that have very long time lags in between. LSTMs contain information in a memory, much like the memory of a computer. This memory can be seen as a gated cell, with gated meaning the cell decides whether or not to store or delete information (i.e., if it opens the gates or not), based on the importance it assigns to the information. The assigning of importance happens through weights, which are also learned by the algorithm. This simply means that it learns over time what information is important and what is not.

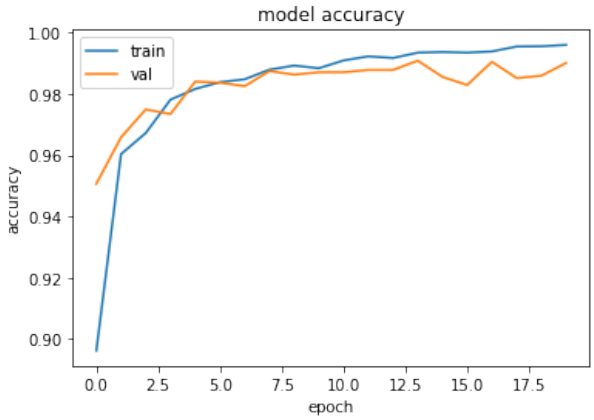


LSTM

So what we are doing is we pass our 2-D features data into LSTM layer with return sequence = True then add some more LSTM layers along with dense and dropout layers(To prevent overfitting) . We trained it using ADAM as optimizer and categorical cross entropy as loss function.

10. CONCLUSION

In this report, we made a model which detects events with considerable precision and predicts which event has occurred



RNN-Model performance plot

using a combination of Convolutional Neural Network and Linear Model. Even though this model only works for audio which contains speech and music divided apart by silence, our model can be improved upon to predict when the audio changes suddenly from music to speech or from speech to music.

11. ACKNOWLEDGMENT

We thank Prof. Vipul Arora for the course he has taught us on Machine Learning in Signal Processing. Beyond theoretical understanding in this course, working on a project helped us in getting an insight as to how Machine Learning can have significant real world applications in the field of signal processing.

12. REFERENCES

[1] Ruilin Xu, Rundi Wu, Yuko Ishiwaka, Carl Vondrick, Changxi Zheng, Listening to Sounds of Silence for Speech Denoising