ACOUSTIC/SOUND SCENE AND EVENT DETECTION

Aashish Patel, Prateek Jain

IIT Kanpur

ABSTRACT

Sound event detection (SED) is the task of detecting the **label, onset, and offset** of sound events speech and music in audio streams. **Audio tagging** is a task to predict the tags of audio clips. This project report presents the methods, results and comparisons on both the tasks by using **Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), Hybrid CRNN and Hidden Markov Model (HMM)**.

***Index Terms*—** CRNN, CNN, RNN,HMM, SED, Event Detection, Audio Tagging

1. INTRODUCTION

The goal of sound event detection (SED) is to recognize at what temporal instances different sounds classes are active within an audio signal. State of the art Convolutional Recurrent Neural Network (CRNN) based poly phonic SED systems use a frame-wise cost function for training. HMM model use probabilistic approach to predict framewise event labels. Sound event detection (SED) plays an important role in computational auditory scene analysis, with a specific purpose of detecting meaningful sounds, generally referred to sound events. Detecting sound events such as speech, music provides fundamental information for understanding the situation using acoustic signal. Furthermore, SED could be utilized in many applications, including automated surveillance systems, information retrieval, smart home systems and military applications.

2. PREPARATION OF TRAINING AND VALIDATION DATA

- Obtain music and speech data from here
- Each audio file is of 30 seconds
- Write Python script to break these 30s audio files into finite segments of audio having duration 1s to 5s
- Recorded silent audio files of duration 1s to 9s
- Using Python script, merge files randomly out of the above prepared sets and resulting audio files will be of 10s each

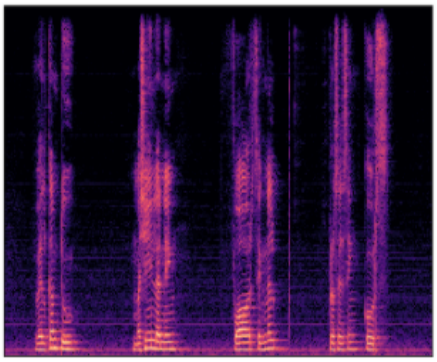

Fig. 1. Spectrogram

- Prepared a dataset of 1020 audio files with their labels

3. EXPERIMENTS

3.1. Feature Extraction

1. We use spectrograms of the 10s audio files as our input to the models described below.
2. Specifications for obtaining the required spectrograms from the audio files using Librosa library:-
   - Hop size - 512
   - Window Size - 1024
   - n_fft - 1024
   - Window Type - Hann Window
3. Spectrogram size for each audio file of 10s is (513, 313).
4. Here 313 are the no of frames into which the whole audio file of 10s if divided, and 513 if the no of features for each of these frame. The model classifies each of these 313 frames individually, and then some post processing is done.

3.2. Task 1 : Sound/Acoustic Event Detection With Onset And Offset Predictions

1. **Convolutional-Recurrent Neural Networks Model (CRNN)**

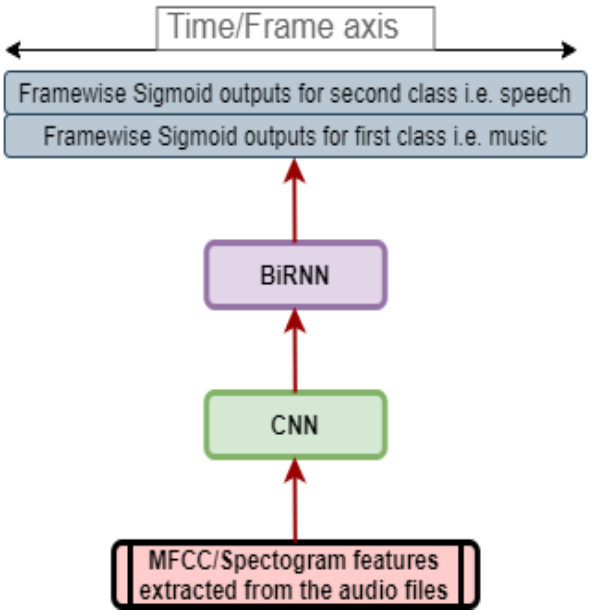

Fig. 2. CRNN Model

- Two CNN layers, followed by **Max Pooling Layers** and Reshaping Layers convert the (batch_size, 513, 313) input to (batch_size, 313, 1600), which is fed into the BiRNN.
- A BiRNN with size of 313 hidden units is exploited to work on input sequences that is 313 frames long. Converts (batch_size, 313, 1600) to (batch_size, 313, 32).
- This is followed by two **Dense Fully Connected** Layers to convert (batch_size, 313, 32) to (batch_size, 313, 2), followed by **Sigmoid** applied to the last dimension.
- **Loss Function:** Binary Cross-entropy as it is the case of Multi-Label Classification

2. **Convolutional Neural Networks Model (CNN)**

- Two CNN layers, followed by **Max Pooling Layers** and Reshaping Layers convert the (batch_size, 513, 313) input to (batch_size, 313, 1600), which is fed into the Dense Layers.
- This is followed by two **Dense Fully Connected** Layers to convert (batch_size, 313, 1600) to (batch_size, 313, 2), followed by **Sigmoid** applied to the last dimension.
- **Loss Function:** Binary Cross-entropy

3. **Recurrent Neural Networks Model (RNN)**

- Input is fed directly to the **BiRNN layers**(2 in no, stacked on top of each other, as shown in the RNN image).

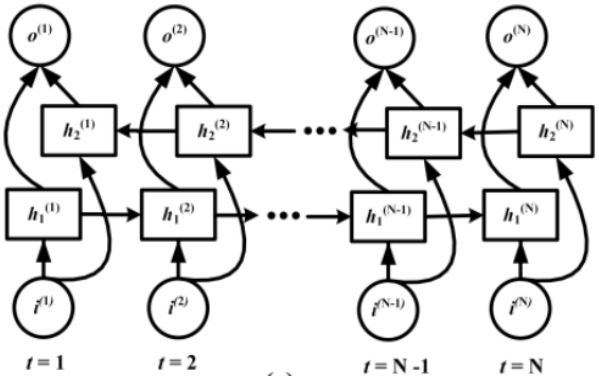

Fig. 3. RNN

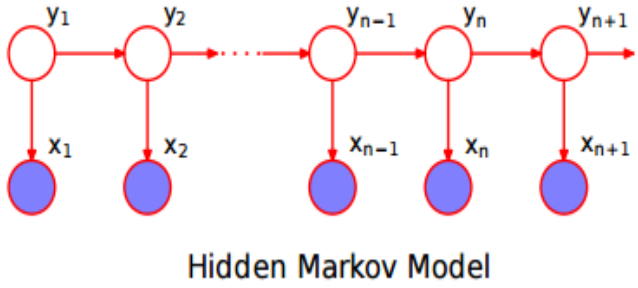

Fig. 4. HMM

- A BiRNN with size of 313 hidden units is exploited to work on input sequences that is 313 frames long. Converts (batch_size, 313, 1600) to (batch_size, 313, 32).
- This is followed by two **Dense Fully Connected** Layers to convert (batch_size, 313, 32) to (batch_size, 313, 2), followed by **Sigmoid** applied to the last dimension.
- **Loss Function:** Binary Cross-entropy

4. **Hidden Markov Model (HMM)**

- Define three classes

Classes = {speech,music, silence}

Observed States (x_n) (*continuos*) of the model are the spectrogram features vector of size (513,1) corresponding to each frame
Hidden States (z_n) (*discrete*) of the model are the classes

- **Input** of the model are spectrogram features of dimensions (513,313) for a 10s audio file. Size of

sequence_class	music	silence	speech
initial_classes			
music	0.989450	0.005048	0.005502
silence	0.003833	0.992683	0.003484
speech	0.004420	0.005092	0.990489

Fig. 5. State Transition Probability Matrix

each frame is 513. There are 1020 files in the training set.

- Calculate **State Transition Probability Matrix**(Fig. 5) of dimensions 3X3 from the training labels by counting all class-to-class transitions. Normalise all values so the probabilities of going from one chord to all others is always 1.
- Calculate **Initial State Probability Matrix** from the 1020 files

  music 0.236111
  speech 0.380952
  silence 0.382937

- Create three **Multivariate Gaussians/GMM**(with three components). Calculate the mean and their covariance by running a for loop on every class and their respective spectrogram

  (a) Mean Vector: Since we have 3 states, mu_array will be of shape [3 , 513], where 513 are the number of features from the spectrogram. It represents the average energy in each one of the 513 features for each of the class.

  (b) State Covariance Matrix: Matrix of shape [3, 531,531] for each class. It determines the variation of features among themselves in each class.

- **Prediction:** Decode the best sequence of classes given the test observations in the form of windowed spectrograms with the help of calculated initial state probability matrix, state transition matrix, means and covariances of GMM using **Viterbi Algorithm** under the HMM properties of (i) Output Independence and (ii) Markov Assumption.

5. **Post Processing of the output vector of Models for the SED task**

   - The final output of each model if of shape (batch_size, 313, 2) where 313 is the no of frames.
   - **Binarization Step:** We apply threshold of 0.5 along the last dimension, can convert them to 1 if value is greater than 0.5 otherwise 0.
   - **Clubbing Event Frames for Each Class:** We pick each of the two vectors(two classes) of size 313 for each file, and club the event frames into one.
   - Then we remove events which have duration less than 0.5
   - Then we combine all the events of the same class for each file, which are separated by a distance less than 0.5
   - The two of the above were fixed by experimenting with different thresholds and then going with the one which gives best results.
   - Finally these events are written to the CSV file.

6. **Results**

Table 1. DCASE based Segment-Based Metrics for SED

Model	F1 Score	Binary Accuracy	ER
CRNN	0.7515	0.821	0.41
CNN	0.58	0.710	1.34
RNN	0.2128	0.7063	1.0571
HMM	0.9074	0.9355	0.18

3.3. Task 2: Acoustic Event Tagging in Audio

1. **Modelling:** Use all the four models described in the previous sections with the same layer structure and different inputs and outputs for the audio tagging task

2. Postprocess the output CSV file of the SED task to extract the event labels present in the audio file

3. **Results**

Table 2. Audio Tagging Event Metrics

Model	F1 Score	Accuracy
CRNN	0.537	0.34
CNN	0.484	0.29
RNN	0.40	0.25
HMM	0.641	0.44

4. TRAINING GRAPHS

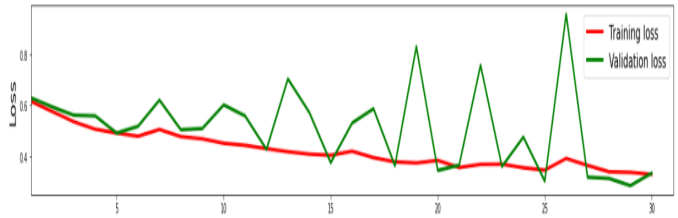

Fig. 6. CRNN Tranining and Validation Loss

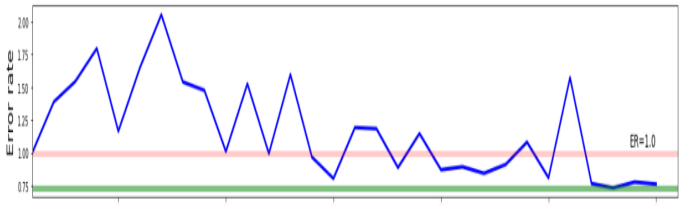

Fig. 7. CRNN Validation Segment Based Error Rate

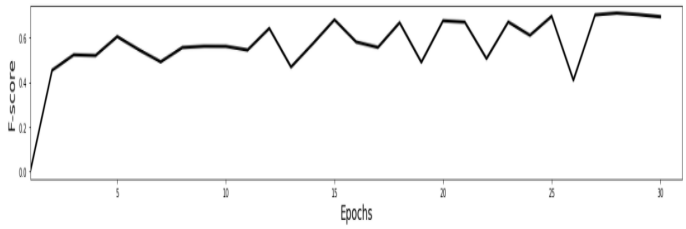

Fig. 8. CRNN F-score Segment Based Error Rate

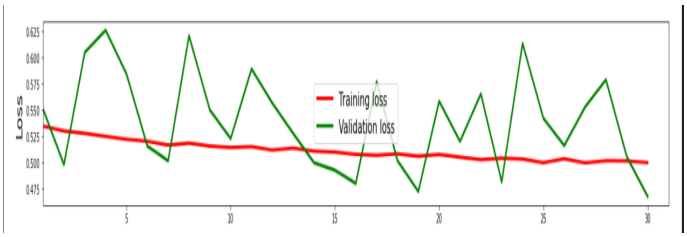

Fig. 9. CNN Tranining and Validation Loss

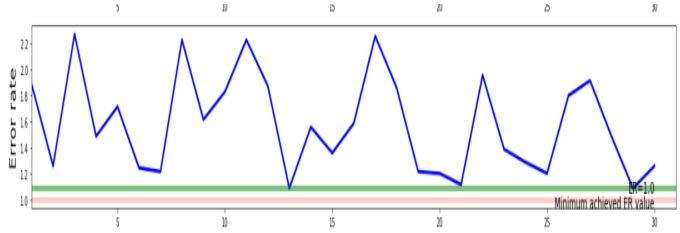

Fig. 10. CNN Validation Segment Based Error Rate

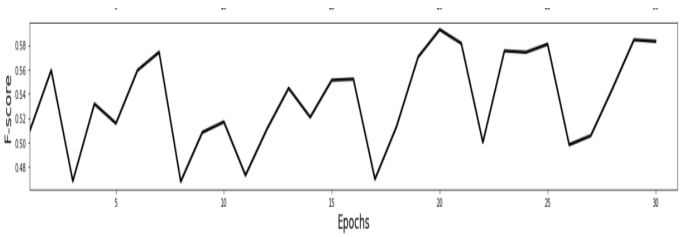

Fig. 11. CNN F-score Segment Based Error Rate

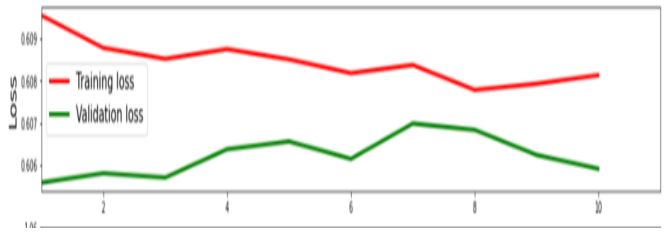

Fig. 12. RNN Tranining and Validation Loss

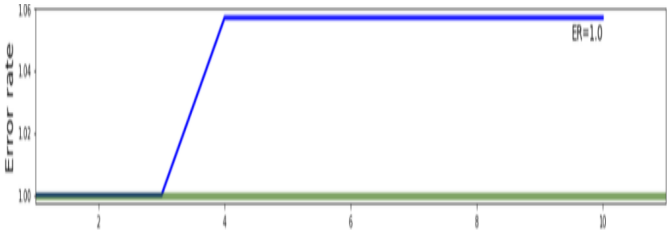

Fig. 13. RNN Validation Segment Based Error Rate

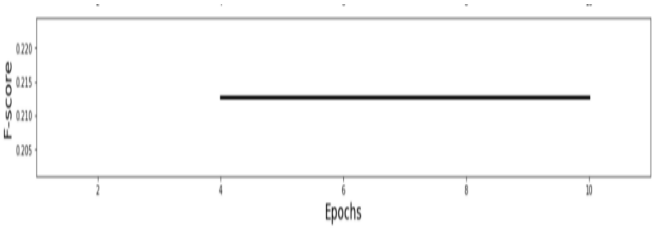

Fig. 14. RNN F-score Segment Based Error Rate

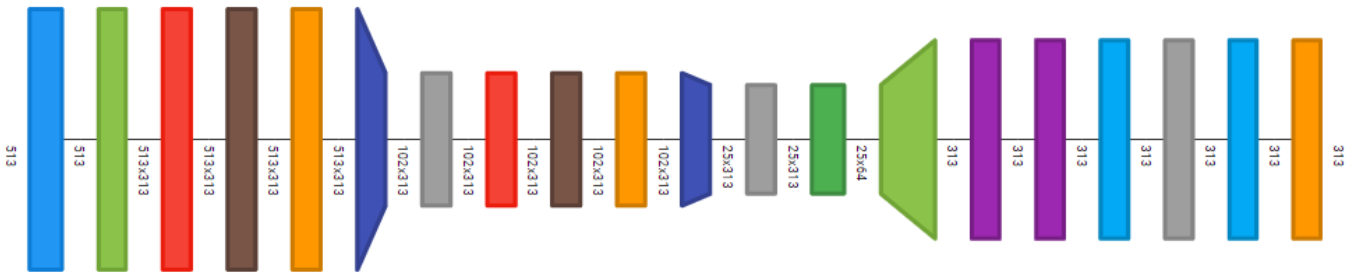

Fig. 15. CRNN Layers Model

5. REFERENCES

[1] Inkyu Choi and Kisoo Kwon, Soo Hyun Bae and Nam Soo Kim, *DNN-BASED SOUND EVENT DETECTION WITH EXEMPLAR-BASED APPROACH FOR NOISE REDUCTION*, DCASE 2016.

[2] Tomoki Hayashi, Shinji Watanabe, Tomoki Toda1, Takaaki Hori, Jonathan Le Roux, Kazuya Takeda1, *BIDIRECTIONAL LSTM-HMM HYBRID SYSTEM FOR POLYPHONIC SOUND EVENT DETECTION*, DCASE 2016.

[3] Annamaria Mesaros1, Toni Heittola1, Tuomas Virtanen1, Mark D. Plumbley, Tampere University, Finland, *Sound Event Detection: A Tutorial* .

[4] Sharath Adavanne and Archontis Politis and Joonas Nikunen and Tuomas Virtanen *Sound Event Localization and Detection of Overlapping Sources Using Convolutional Recurrent Neural Networks*, IEEE 2018