# EMOTION RECOGNITION FROM AUDIO FROM ANIMATED MOVIES USING RNN AND ITS VARIATIONS

Avantika Sivakumar 2019103007
Subhiksha Sai Subramanian 2019103067
Vasudha E 2019103073

*for the course*

CS 6301 – MACHINE LEARNING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING,
COLLEGE OF ENGINEERING, GUINDY, ANNA UNIVERSITY,
CHENNAI 600 025.

# ABSTRACT

Speech emotion detection is one of the most researched fields in ML, with uses in places such as the medical field, customer call centers, and emergency services. We have explored, compared and contrasted the differences and aim to find the most efficient algorithm out of simple RNN, bidirectional LSTM, and GRU.

Recurrent neural networks are one of the best algorithms for sequential data. It is even used by Apple's Siri and Google's voice search. It has internal memory to remember it's input, and is the first algorithm to do so. This makes their predictions for what comes next more accurate, and so they are used for audio datasets. They can understand sequences and contexts well. 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 usual RNN has a short-term memory, so we combine it with a LSTM so it also has long-term memory. It produces output, copies that output and loops it back into the network.

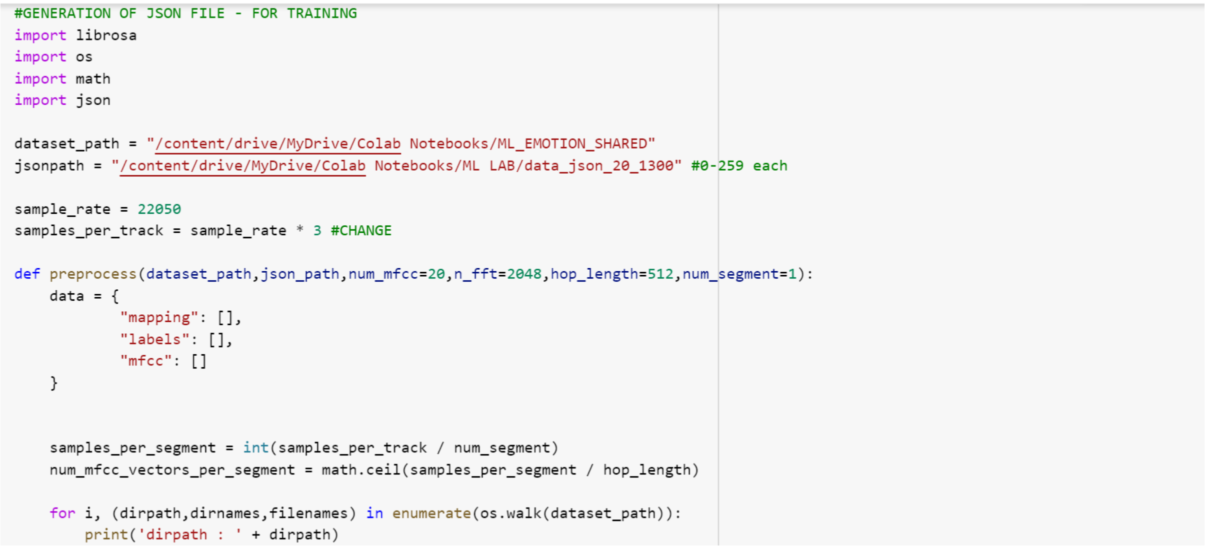
Bidirectional long-short term memory is the process of making any neural network o have the sequence information in both directions backwards (future to past) or forward (past to future). In bidirectional, our input flows in two directions, making a bi-lstm different from the regular LSTM. With the regular LSTM, we can make input flow in one direction, either backwards or forward.
However, in bi-directional, we can make the input flow in both directions to preserve the future and the past information. The input sequence is fed in normal time order for one network, and in reverse time order for another. The outputs of the two networks are usually concatenated at each time step.

GRU or gated recurrent unit is variation of recurrent neural network, proposed by Kyunghyun Cho et al. in 2014. Although it is not as well-known as Long Short Term Memory Network (LSTM), It is equally effective. Unlike LSTM, it consists of only 3 gates and doesn't maintain an Internal Cell State. The information stored in the Internal Cell State in LSTM recurrent unit is incorporated into the hidden state of GRU. The basic flow of a GRU network is similar to that of a basic RNN when illustrated. The main difference between the two lies in the internal working within each recurrent unit.

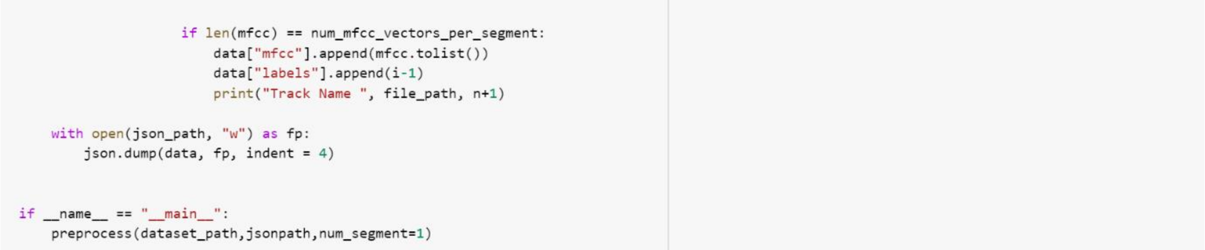
# METHODOLOGY

For each audio file 20 MFCC features are extracted and written into a JavaScript Object Notation (json) file with the labels and mappings.
The MFCC vector is then read from the json file and the train-test split is done. The neural Network is constructed with variations in hyperparameters and the model is compiled and trained with the train dataset.
The model is tested with the test dataset and the output class probabilities are recorded. Accuracy, precision, recall and confusion matrix are printed.

# CODE:



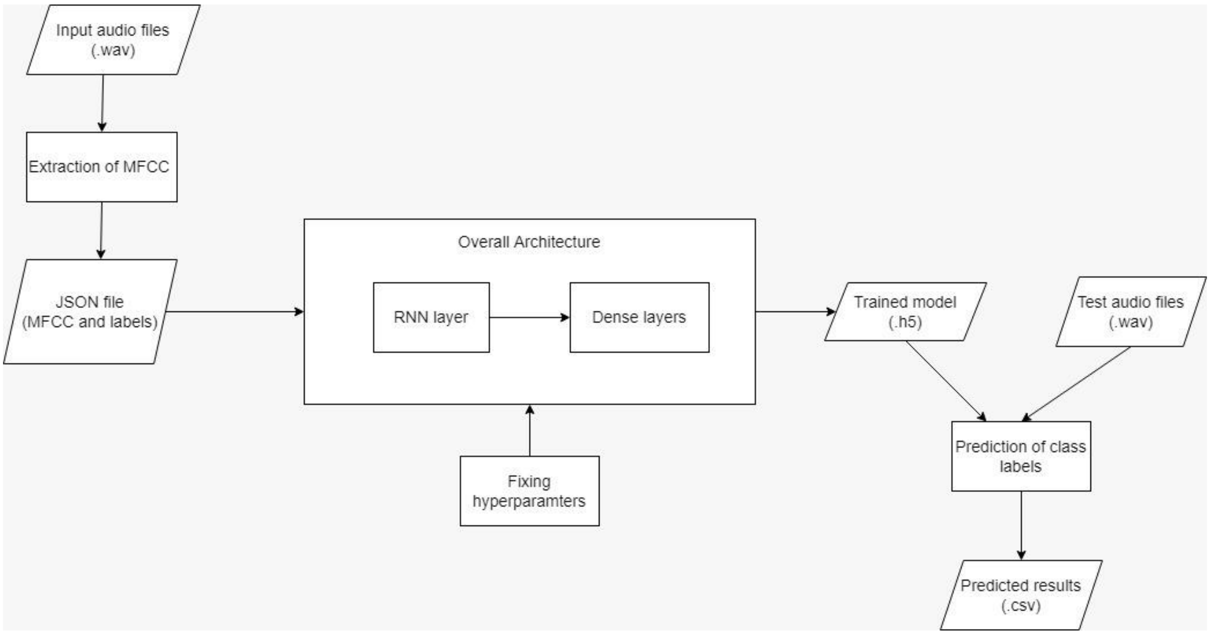
```
#GENERATION OF JSON FILE - FOR TRAINING
import librosa
import os
import math
import json

dataset_path = "/content/drive/MyDrive/Colab Notebooks/ML_EMOTION_SHARED"
jsonpath = "/content/drive/MyDrive/Colab Notebooks/ML LAB/data_json_20_1300" #0-259 each

sample_rate = 22050
samples_per_track = sample_rate * 3 #CHANGE

def preprocess(dataset_path,json_path,num_mfcc=20,n_fft=2048,hop_length=512,num_segment=1):
    data = {
            "mapping": [],
            "labels": [],
            "mfcc": []
    }


    samples_per_segment = int(samples_per_track / num_segment)
    num_mfcc_vectors_per_segment = math.ceil(samples_per_segment / hop_length)

    for i, (dirpath,dirnames,filenames) in enumerate(os.walk(dataset_path)):
        print('dirpath : ' + dirpath)
```



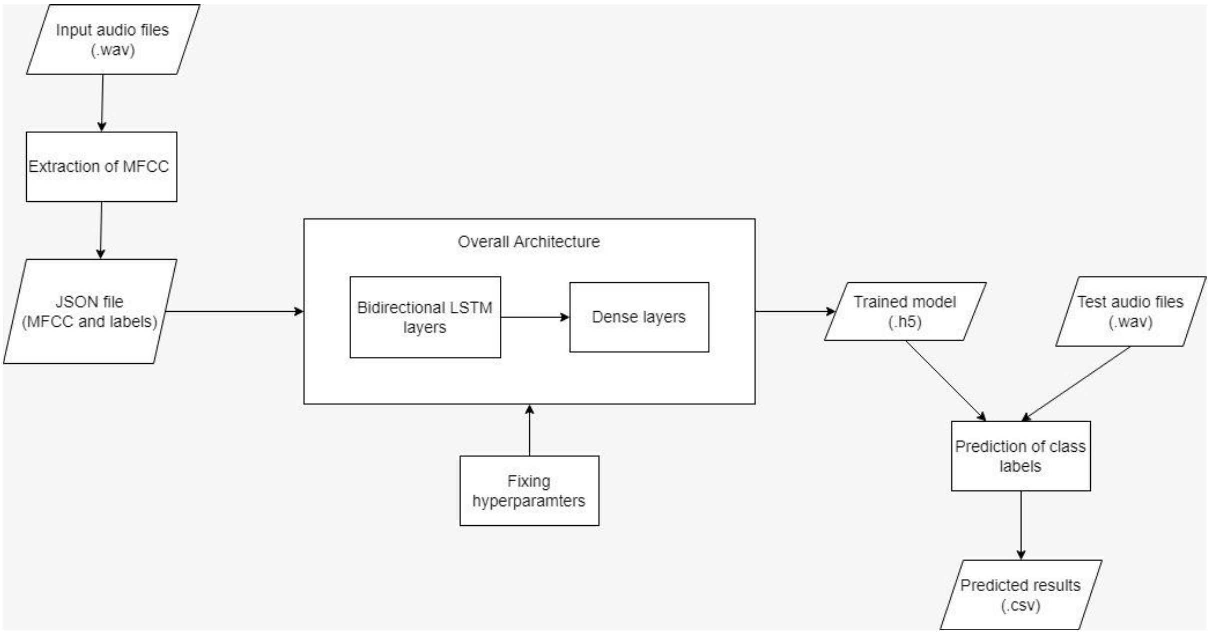
```
                    if len(mfcc) == num_mfcc_vectors_per_segment:
                        data["mfcc"].append(mfcc.tolist())
                        data["labels"].append(i-1)
                        print("Track Name ", file_path, n+1)

    with open(json_path, "w") as fp:
        json.dump(data, fp, indent = 4)


if __name__ == "__main__":
    preprocess(dataset_path,jsonpath,num_segment=1)
```

# BLOCK DIAGRAMS

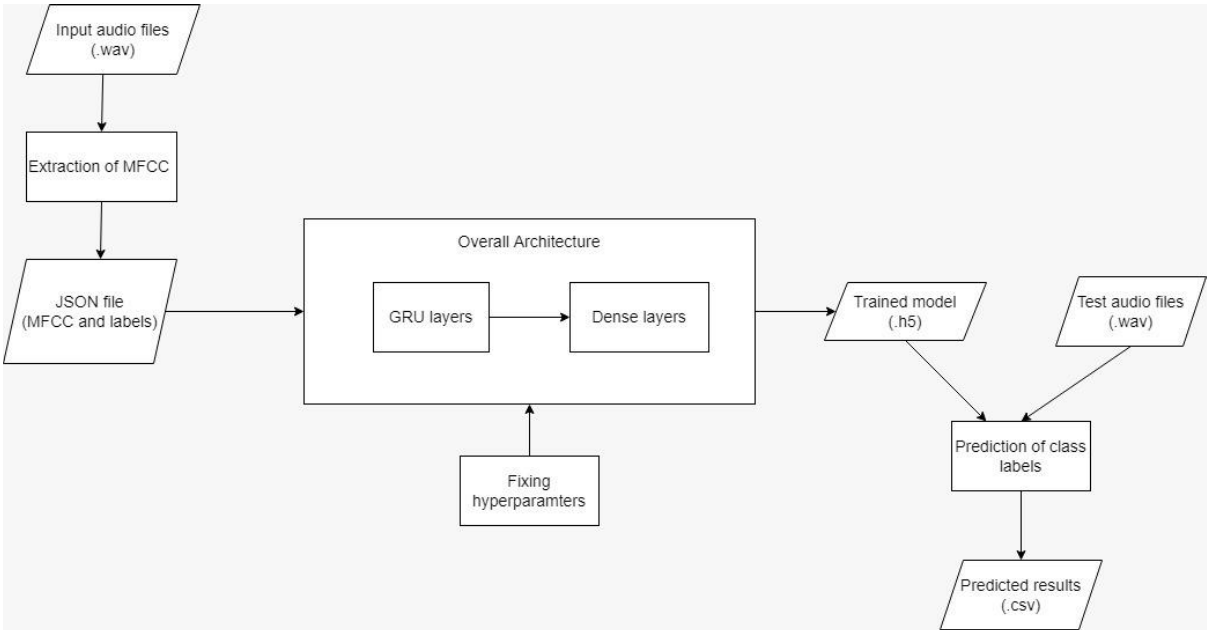
## RNN Algorithm:



## Bidirectional LSTM Algorithm:



### GRU Algorithm:



## DATASET

Audio files from animated movies were manually extracted and split based on each dialogue and labelled each one as Happy, Sad, Anger, Disgust, or Shock. The duration of each file was set to 3 seconds.

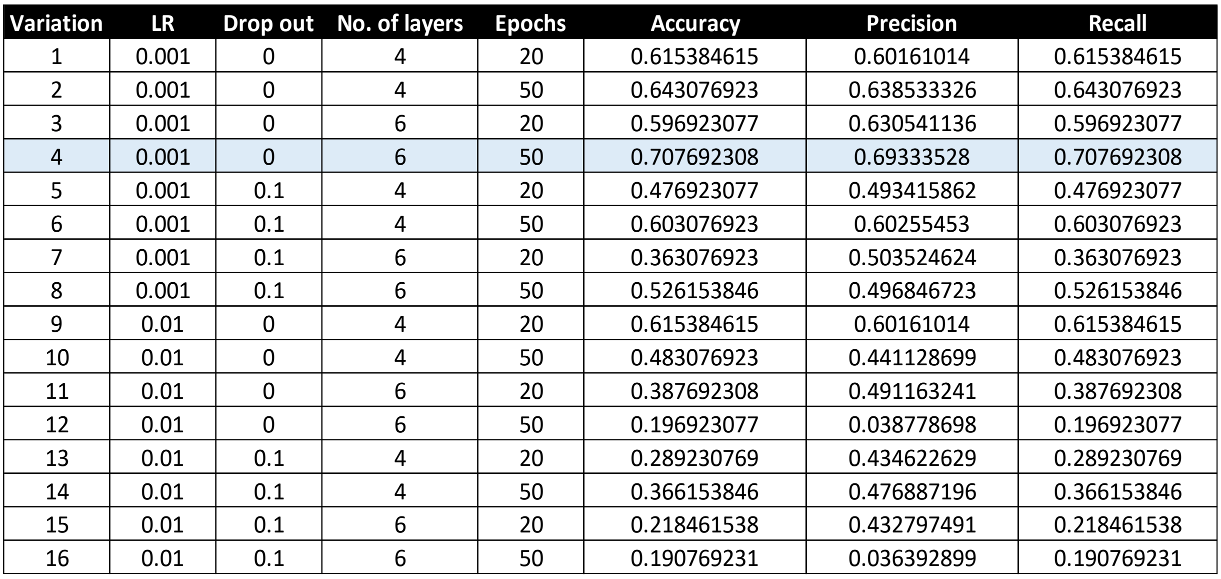
Total number of instances: 1300

### LINK:

https://drive.google.com/drive/folders/1rGX2LOjr8Cs1tqMnPzGZxafYZteUKWRc?usp=sharing

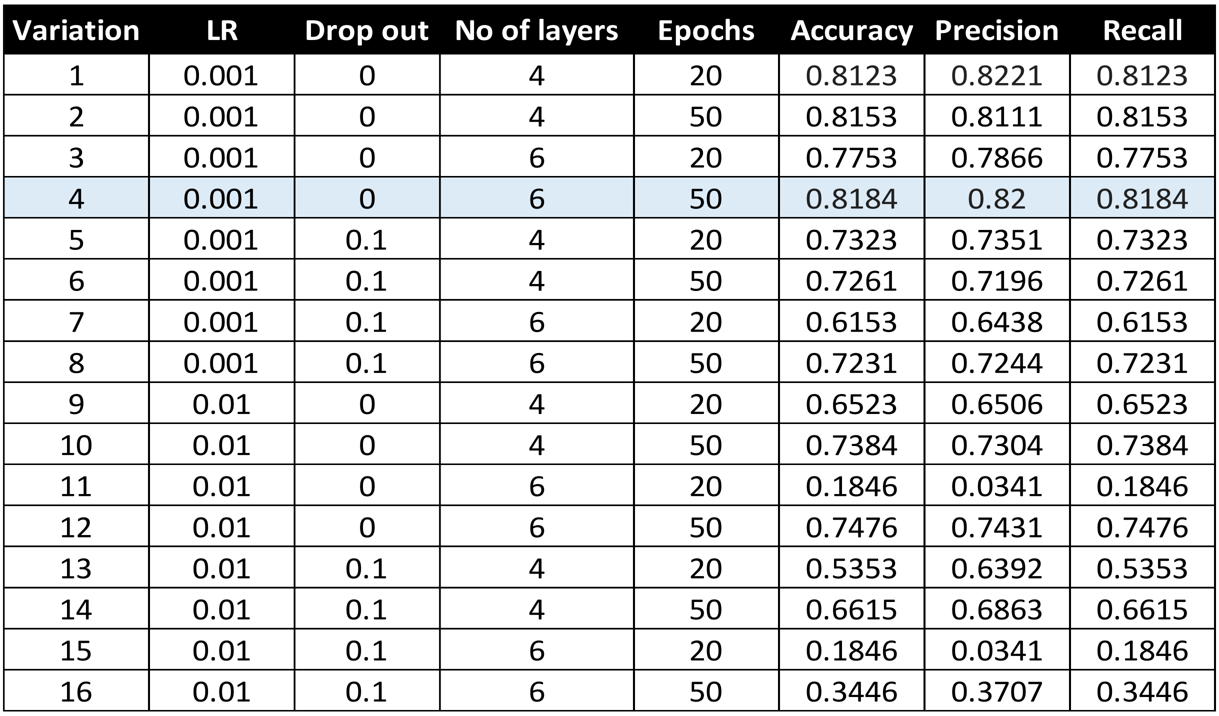
# RESULTS AND DISCUSSION

## RNN ALGORITHM



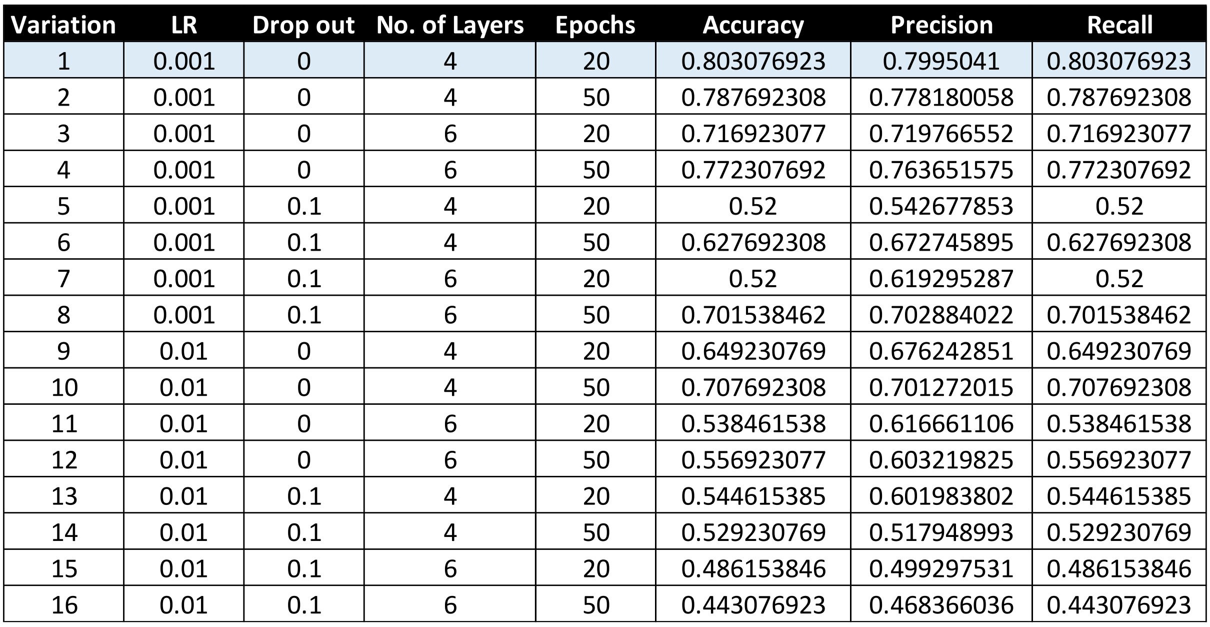
| Variation | LR | Drop out | No. of layers | Epochs | Accuracy | Precision | Recall |
|---|---|---|---|---|---|---|---|
| 1 | 0.001 | 0 | 4 | 20 | 0.615384615 | 0.60161014 | 0.615384615 |
| 2 | 0.001 | 0 | 4 | 50 | 0.643076923 | 0.638533326 | 0.643076923 |
| 3 | 0.001 | 0 | 6 | 20 | 0.596923077 | 0.630541136 | 0.596923077 |
| 4 | 0.001 | 0 | 6 | 50 | 0.707692308 | 0.69333528 | 0.707692308 |
| 5 | 0.001 | 0.1 | 4 | 20 | 0.476923077 | 0.493415862 | 0.476923077 |
| 6 | 0.001 | 0.1 | 4 | 50 | 0.603076923 | 0.60255453 | 0.603076923 |
| 7 | 0.001 | 0.1 | 6 | 20 | 0.363076923 | 0.503524624 | 0.363076923 |
| 8 | 0.001 | 0.1 | 6 | 50 | 0.526153846 | 0.496846723 | 0.526153846 |
| 9 | 0.01 | 0 | 4 | 20 | 0.615384615 | 0.60161014 | 0.615384615 |
| 10 | 0.01 | 0 | 4 | 50 | 0.483076923 | 0.441128699 | 0.483076923 |
| 11 | 0.01 | 0 | 6 | 20 | 0.387692308 | 0.491163241 | 0.387692308 |
| 12 | 0.01 | 0 | 6 | 50 | 0.196923077 | 0.038778698 | 0.196923077 |
| 13 | 0.01 | 0.1 | 4 | 20 | 0.289230769 | 0.434622629 | 0.289230769 |
| 14 | 0.01 | 0.1 | 4 | 50 | 0.366153846 | 0.476887196 | 0.366153846 |
| 15 | 0.01 | 0.1 | 6 | 20 | 0.218461538 | 0.432797491 | 0.218461538 |
| 16 | 0.01 | 0.1 | 6 | 50 | 0.190769231 | 0.036392899 | 0.190769231 |

## BIDIRECTIONAL LSTM ALGORITHM



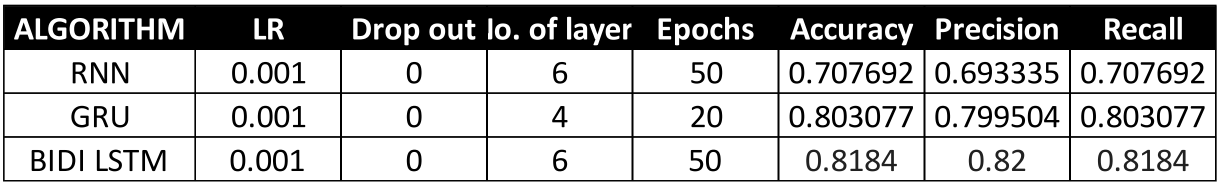
| Variation | LR | Drop out | No of layers | Epochs | Accuracy | Precision | Recall |
|---|---|---|---|---|---|---|---|
| 1 | 0.001 | 0 | 4 | 20 | 0.8123 | 0.8221 | 0.8123 |
| 2 | 0.001 | 0 | 4 | 50 | 0.8153 | 0.8111 | 0.8153 |
| 3 | 0.001 | 0 | 6 | 20 | 0.7753 | 0.7866 | 0.7753 |
| 4 | 0.001 | 0 | 6 | 50 | 0.8184 | 0.82 | 0.8184 |
| 5 | 0.001 | 0.1 | 4 | 20 | 0.7323 | 0.7351 | 0.7323 |
| 6 | 0.001 | 0.1 | 4 | 50 | 0.7261 | 0.7196 | 0.7261 |
| 7 | 0.001 | 0.1 | 6 | 20 | 0.6153 | 0.6438 | 0.6153 |
| 8 | 0.001 | 0.1 | 6 | 50 | 0.7231 | 0.7244 | 0.7231 |
| 9 | 0.01 | 0 | 4 | 20 | 0.6523 | 0.6506 | 0.6523 |
| 10 | 0.01 | 0 | 4 | 50 | 0.7384 | 0.7304 | 0.7384 |
| 11 | 0.01 | 0 | 6 | 20 | 0.1846 | 0.0341 | 0.1846 |
| 12 | 0.01 | 0 | 6 | 50 | 0.7476 | 0.7431 | 0.7476 |
| 13 | 0.01 | 0.1 | 4 | 20 | 0.5353 | 0.6392 | 0.5353 |
| 14 | 0.01 | 0.1 | 4 | 50 | 0.6615 | 0.6863 | 0.6615 |
| 15 | 0.01 | 0.1 | 6 | 20 | 0.1846 | 0.0341 | 0.1846 |
| 16 | 0.01 | 0.1 | 6 | 50 | 0.3446 | 0.3707 | 0.3446 |

## GRU ALGORITHM



| Variation | LR | Drop out | No. of Layers | Epochs | Accuracy | Precision | Recall |
|---|---|---|---|---|---|---|---|
| 1 | 0.001 | 0 | 4 | 20 | 0.803076923 | 0.7995041 | 0.803076923 |
| 2 | 0.001 | 0 | 4 | 50 | 0.787692308 | 0.778180058 | 0.787692308 |
| 3 | 0.001 | 0 | 6 | 20 | 0.716923077 | 0.719766552 | 0.716923077 |
| 4 | 0.001 | 0 | 6 | 50 | 0.772307692 | 0.763651575 | 0.772307692 |
| 5 | 0.001 | 0.1 | 4 | 20 | 0.52 | 0.542677853 | 0.52 |
| 6 | 0.001 | 0.1 | 4 | 50 | 0.627692308 | 0.672745895 | 0.627692308 |
| 7 | 0.001 | 0.1 | 6 | 20 | 0.52 | 0.619295287 | 0.52 |
| 8 | 0.001 | 0.1 | 6 | 50 | 0.701538462 | 0.702884022 | 0.701538462 |
| 9 | 0.01 | 0 | 4 | 20 | 0.649230769 | 0.676242851 | 0.649230769 |
| 10 | 0.01 | 0 | 4 | 50 | 0.707692308 | 0.701272015 | 0.707692308 |
| 11 | 0.01 | 0 | 6 | 20 | 0.538461538 | 0.616661106 | 0.538461538 |
| 12 | 0.01 | 0 | 6 | 50 | 0.556923077 | 0.603219825 | 0.556923077 |
| 13 | 0.01 | 0.1 | 4 | 20 | 0.544615385 | 0.601983802 | 0.544615385 |
| 14 | 0.01 | 0.1 | 4 | 50 | 0.529230769 | 0.517948993 | 0.529230769 |
| 15 | 0.01 | 0.1 | 6 | 20 | 0.486153846 | 0.499297531 | 0.486153846 |
| 16 | 0.01 | 0.1 | 6 | 50 | 0.443076923 | 0.468366036 | 0.443076923 |

## DISCUSSION



| ALGORITHM | LR | Drop out | lo. of layer | Epochs | Accuracy | Precision | Recall |
|---|---|---|---|---|---|---|---|
| RNN | 0.001 | 0 | 6 | 50 | 0.707692 | 0.693335 | 0.707692 |
| GRU | 0.001 | 0 | 4 | 20 | 0.803077 | 0.799504 | 0.803077 |
| BIDI LSTM | 0.001 | 0 | 6 | 50 | 0.8184 | 0.82 | 0.8184 |

The above table shows the best variation of each algorithm.

As we can see, bidirectional LSTM gives the highest accuracy, precision, and recall, with GRU as a close second.

# REFERENCES

1. Yoon, S., Byun, S., & Jung, K. (2018, December). Multimodal speech emotion recognition using audio and text. In *2018 IEEE Spoken Language Technology Workshop (SLT)* (pp. 112-118). IEEE.

2. Cho, J., Pappagari, R., Kulkarni, P., Villalba, J., Carmiel, Y., & Dehak, N. (2019). Deep neural networks for emotion recognition combining audio and transcripts. *arXiv preprint arXiv:1911.00432*.

3. Weninger, F., Ringeval, F., Marchi, E., & Schuller, B. W. (2016, July). Discriminatively Trained Recurrent Neural Networks for Continuous Dimensional Emotion Recognition from Audio. In *IJCAI* (Vol. 2016, pp. 2196-2202).

4. Baird, A., Amiriparian, S., Milling, M., & Schuller, B. W. (2021, January). Emotion recognition in public speaking scenarios utilising an lstm-rnn approach with attention. In *2021 IEEE Spoken Language Technology Workshop (SLT)* (pp. 397-402). IEEE.