# Сравнительный анализ Random Forests, CNN, и RNN при распознавании речевых эмоций

*Ле Нгок Тхиен, Нгуен Тхи Ми Ту, Чинь Дык Тханг*

*https://github.com/kaitouz/Neurotechnologies-Project*

Университет ИТМО

## ABSTRACT

Speech emotion recognition is an important task in natural language processing with numerous applications, such as improving the user experience in virtual assistants or enhancing the effectiveness of mental health diagnosis. In this paper, we compare the performance of three popular machine learning techniques – random forest classifier (RFC), convolutional neural network (CNN), and recurrent neural network (RNN) - for speech emotion recognition using the RAVDESS dataset. We implement and train each model using appropriate optimization algorithms and evaluate their performance using various metrics, including accuracy, F1 score, and confusion matrix. Our results show that CNN and RNN outperform RFC in terms of overall performance, with CNN achieving the best results. We also observe that the models perform differently on different emotions, with some emotions being easier to recognize than others. Our findings provide insights into the relative strengths and limitations of these techniques for speech emotion recognition and suggest directions for future research.

## 1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Speech emotion recognition (SER) is a field of natural language processing that aims to identify and classify the emotional state of a speaker from their spoken words. It has a wide range of potential applications, including improving the user experience in virtual assistants, enhancing the effectiveness of mental health diagnosis, and improving customer service in call centers.

One of the main challenges in SER is the variability of emotions, which can be influenced by factors such as culture, individual differences, and context. As a result, developing robust and accurate SER systems requires the use of diverse and representative datasets, as well as robust machine learning techniques.

There are several approaches and methods that have been used to solve the existing problems in SER. One common approach is to extract features from the audio signal and apply machine learning algorithms (e.g., Random Forest, SVM,..) to classify the emotions. These features can include prosodic features such as pitch, frame energy, and duration, as well as spectral features such as spectral centroid, spectral rolloff, and spectral flatness.

Another approach is to use machine learning techniques that are capable of learning from raw audio data or audio spectrograms, such as deep learning models. These models, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), are able to learn complex patterns in the data and can achieve good performance on SER tasks. However, they require large amounts of labeled data and computational resources to train.

Other methods that have been used in SER include hybrid approaches that combine multiple techniques, as well as techniques that incorporate contextual information, such as speaker characteristics or the content of the spoken words.

Overall, SER is a complex and challenging task, but significant progress has been made in recent years thanks to advances in machine learning and the availability of large datasets. Future research in SER will likely focus on developing more robust and accurate SER systems, as well as exploring new applications and use cases for SER technology.

## 2. ОПРЕДЕЛЕНИЕ ЦЕЛИ И ЗАДАЧ РАБОТЫ

The purpose of this project is to compare the performance of three different methods for solving problems in speech emotion recognition (SER): random forest classifier (RFC), convolutional neural network (CNN), and recurrent neural network (RNN). The main objective of the project is to evaluate the relative strengths and limitations of these methods for SER and to identify which method performs the best on a given dataset. To achieve this objective, the project will involve implementing and training each of the three methods on a representative dataset for SER, such as the RAVDESS dataset, and evaluating their performance using various metrics, such as accuracy, F1 score, and confusion matrix. The results of the comparison will provide insights into the effectiveness of different approaches for SER and may inform future research in this field.

The work for this project has been divided as follows:

- Trinh Duc Thang was responsible for implementing and training the random forest classifier (RFC) model. This will involve extracting relevant features from the audio data, selecting hyperparameters, and evaluating the model's performance using various metrics.
- Nguyen Thi My Tu was responsible for implementing and training the recurrent neural network (RNN) model. This will involve designing and implementing a suitable RNN architecture, optimizing the model using appropriate techniques, and evaluating the model's performance.
- Le Ngoc Thien was responsible for implementing and training the convolutional neural network (CNN) model. This will involve designing and implementing a suitable CNN architecture, optimizing the model using appropriate techniques, and evaluating the model's performance.

## 3. ВЫБОР МЕТОДОВ И ТЕХНОЛОГИЙ ДЛЯ РЕАЛИЗАЦИИ ПРОЕКТА

### 3.1 Dataset

The dataset we have selected to train as well as evaluate the models is RAVDESS. The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS) is a widely used dataset for speech emotion recognition research. It consists of a collection of audio and video recordings of actors speaking and singing in a variety of emotional states, including neutral, happy, sad, angry, fearful, and disgusted.

The RAVDESS dataset was created to provide a diverse and representative dataset for the study of emotional speech and song. It includes a total of 1440 recordings, with each recording lasting approximately 10 seconds. The recordings are evenly balanced across the different emotions and gender of the actors, with each emotion being represented by 24 actors of each gender.

The RAVDESS dataset has been widely used in research on speech emotion recognition and has been found to be a useful benchmark for evaluating the performance of different machine learning algorithms on this task. It has also been used in a variety of other research projects, including those related to speech synthesis and speech recognition.

### 3.2 Random Forest

We explored the use of Random Forest classifier for emotion recognition (SER) on the RAVDESS dataset.

One of the main challenges of using this method was identifying and extracting appropriate features from speech. To overcome this challenge, we used the librosa package to extract relevant features from the audio recordings, including: MFCCs, spectral rolloff, zero-crossing rate, spectral centroid.

The Random Forest model was implemented in Python using the Scikit-Learn library. Since we’ve performed a classification task, we used the random forest classifier class, which is written as RandomForestClassifier in the Scikit-Learn's ensemble library.

### 3.3 RNN

As a second classifier, a bi-directional LSTM recurrent layers was implemented. It is a variety of recurrent neural networks (RNNs) that are capable of learning long-term dependencies, especially in sequence prediction problems. This allows them to learn patterns and relationships that may span multiple time steps, which is useful for tasks such as emotion recognition where the context and duration of the audio may be important factors in determining the emotion being expressed.

### 3.4 CNN

A convolutional neural network (CNN) was used as the final classifier in this project. Convolutional neural networks (CNNs) are a type of deep learning model that are commonly used for image classification and other tasks involving data with a grid-like structure, such as audio spectrograms. CNNs have been successful in many emotions recognition tasks, including those involving the RAVDESS dataset.

To use a CNN for emotion recognition on the RAVDESS dataset, we first used the librosa package to pre-process the audio data. Specifically, we used librosa to extract waveforms, spectrograms and other spectral features from the audio, which can then be used as input to a CNN.

## 3.5 Evaluation metrics

The performance of the above methods was evaluated using a variety of metrics, including accuracy, F1 score, and the confusion matrix. These metrics allowed us to assess the overall effectiveness of the methods and to compare their performance to each other and to baseline models.

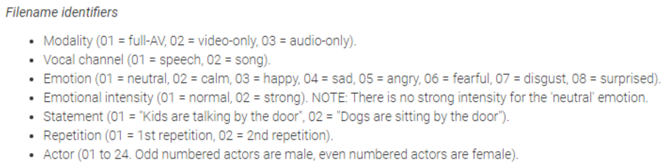
Accuracy – This metric measures the overall ability of a model to correctly identify the emotion expressed in a speech. It's the most commonly-used metric to evaluate SER, because it helps evaluate the model's performance by comparing predictions to the ground truth labels.

Confusion Matrix – This helps understand the performance of a model more specifically. It breaks down the model's accuracy into actual and predicted labels, enabling us to visually understand the model's performance. This can help identify the weaknesses of a model, which can then be improved on.

# 4. ПРОЕКТИРОВАНИЕ, РАЗРАБОТКА И ИССЛЕДОВАНИЕ

## 4.1 Data Preparation

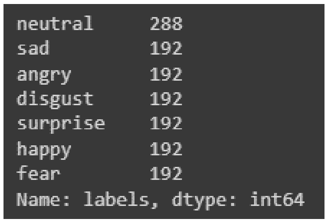
The RAVDESS dataset consists of 1440 audio files in .wav format. The audio files are 16-bit, mono, 44.1kHz .wav files and contain recordings of actors speaking in a variety of emotional states. The dataset is evenly balanced across the different emotions of neutral, happy, sad, angry, fearful, and disgusted, and the gender of the actors, with each emotion being represented by 24 actors of each gender. The audio files are organized into 24 folders, "Actor_01", "Actor_02", and so on. Each containing 60 audio files. And here is the filename identifiers as per the official RAVDESS website:



*Filename identifiers*

- Modality (01 = full-AV, 02 = video-only, 03 = audio-only).
- Vocal channel (01 = speech, 02 = song).
- Emotion (01 = neutral, 02 = calm, 03 = happy, 04 = sad, 05 = angry, 06 = fearful, 07 = disgust, 08 = surprised).
- Emotional intensity (01 = normal, 02 = strong). NOTE: There is no strong intensity for the 'neutral' emotion.
- Statement (01 = "Kids are talking by the door", 02 = "Dogs are sitting by the door").
- Repetition (01 = 1st repetition, 02 = 2nd repetition).
- Actor (01 to 24. Odd numbered actors are male, even numbered actors are female).

Because of the entire data was packaged, and the format of audio filename, there's a few more parsing steps required.



```
neutral     288
sad         192
angry       192
disgust     192
surprise    192
happy       192
fear        192
Name: labels, dtype: int64
```

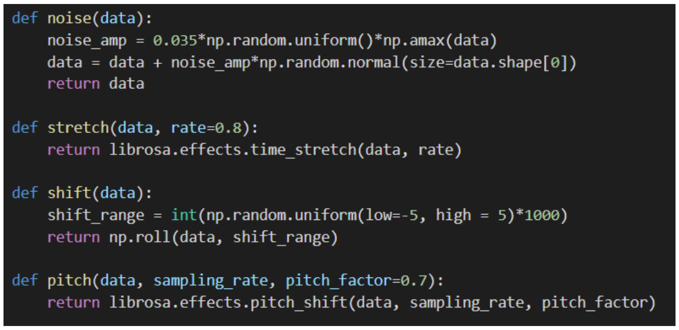
All emotion states have 192 files, except for neutral which only has 96. This class imbalance was considered when implementing the SVM, RNN, and CNN algorithms

In this project, we did not extensively explore the feature selection process to determine which features would be most effective for our dataset. Instead, we simply extracted five features for our analysis.:

- Zero Crossing Rate
- Chroma_stft
- MFCC
- MelSpectogram
- Spectral constrat

To evaluate the performance of our models, we divided the data into two sets: a training set and a test set. The test set comprised 20% of the total data and was used to assess the models' ability to generalize to unseen data.

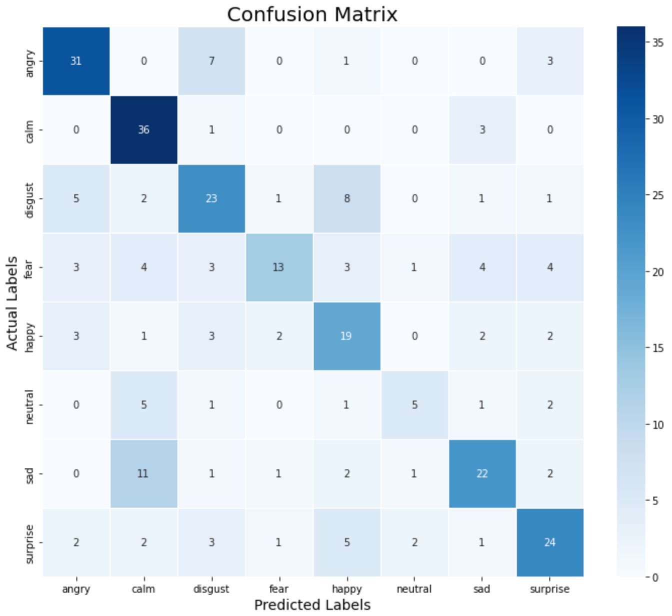
In order to improve the performance of deep learning algorithms such as CNN and RNN on our small training set, we employed a technique called data augmentation. Data augmentation involves creating new synthetic data samples by applying small perturbations to the original training set. For audio data, these perturbations can include adding noise, shifting the time, changing the pitch and speed, etc. The goal of data augmentation is to make the model more robust and able to generalize to new data by learning to be invariant to these types of perturbations. By augmenting our training data in this way, we were able to improve the performance of our deep learning models on the classification task.



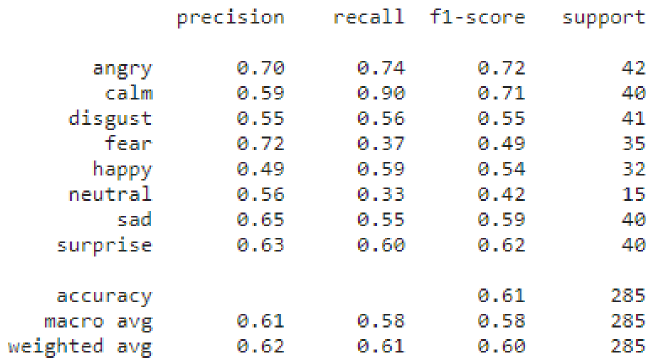
```
def noise(data):
    noise_amp = 0.035*np.random.uniform()*np.amax(data)
    data = data + noise_amp*np.random.normal(size=data.shape[0])
    return data

def stretch(data, rate=0.8):
    return librosa.effects.time_stretch(data, rate)

def shift(data):
    shift_range = int(np.random.uniform(low=-5, high = 5)*1000)
    return np.roll(data, shift_range)

def pitch(data, sampling_rate, pitch_factor=0.7):
    return librosa.effects.pitch_shift(data, sampling_rate, pitch_factor)
```

**Random Forest**

Accuracy of our model on test data :  60.70175438596491 %

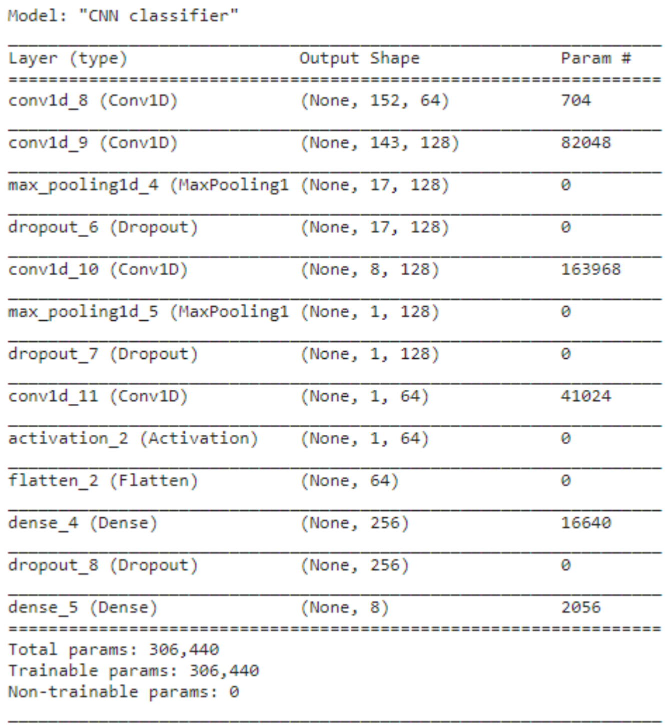


| | angry | calm | disgust | fear | happy | neutral | sad | surprise |
|---|---|---|---|---|---|---|---|---|
| angry | 31 | 0 | 7 | 0 | 1 | 0 | 0 | 3 |
| calm | 0 | 36 | 1 | 0 | 0 | 0 | 3 | 0 |
| disgust | 5 | 2 | 23 | 1 | 8 | 0 | 1 | 1 |
| fear | 3 | 4 | 3 | 13 | 3 | 1 | 4 | 4 |
| happy | 3 | 1 | 3 | 2 | 19 | 0 | 2 | 2 |
| neutral | 0 | 5 | 1 | 0 | 1 | 5 | 1 | 2 |
| sad | 0 | 11 | 1 | 1 | 2 | 1 | 22 | 2 |
| surprise | 2 | 2 | 3 | 1 | 5 | 2 | 1 | 24 |

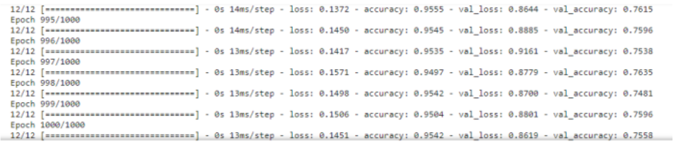


```
              precision    recall  f1-score   support

       angry       0.70      0.74      0.72        42
        calm       0.59      0.90      0.71        40
     disgust       0.55      0.56      0.55        41
        fear       0.72      0.37      0.49        35
       happy       0.49      0.59      0.54        32
     neutral       0.56      0.33      0.42        15
         sad       0.65      0.55      0.59        40
    surprise       0.63      0.60      0.62        40

    accuracy                           0.61       285
   macro avg       0.61      0.58      0.58       285
weighted avg       0.62      0.61      0.60       285
```

## CNN

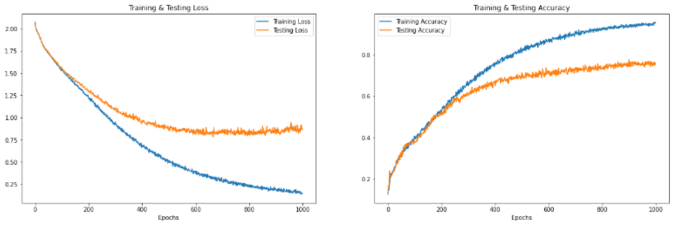


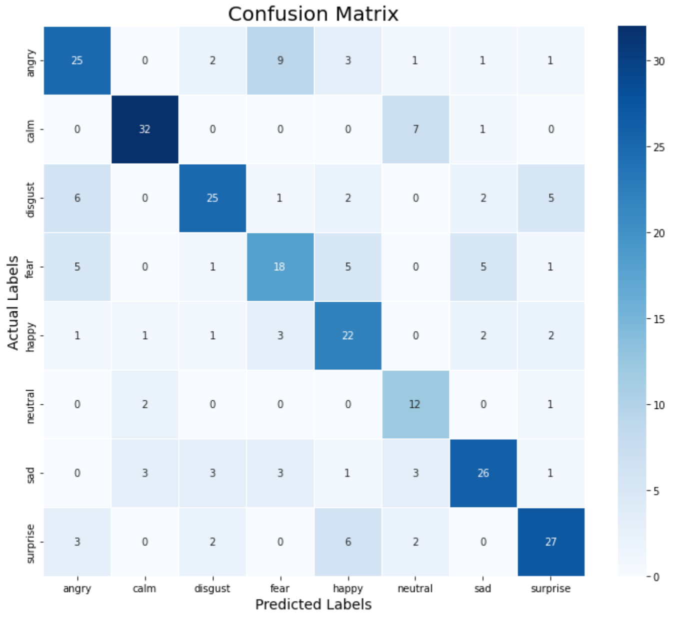
```
Model: "CNN classifier"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d_8 (Conv1D)            (None, 152, 64)           704
_________________________________________________________________
conv1d_9 (Conv1D)            (None, 143, 128)          82048
_________________________________________________________________
max_pooling1d_4 (MaxPooling1 (None, 17, 128)           0
_________________________________________________________________
dropout_6 (Dropout)          (None, 17, 128)           0
_________________________________________________________________
conv1d_10 (Conv1D)           (None, 8, 128)            163968
_________________________________________________________________
max_pooling1d_5 (MaxPooling1 (None, 1, 128)            0
_________________________________________________________________
dropout_7 (Dropout)          (None, 1, 128)            0
_________________________________________________________________
conv1d_11 (Conv1D)           (None, 1, 64)             41024
_________________________________________________________________
activation_2 (Activation)    (None, 1, 64)             0
_________________________________________________________________
flatten_2 (Flatten)          (None, 64)                0
_________________________________________________________________
dense_4 (Dense)              (None, 256)               16640
_________________________________________________________________
dropout_8 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_5 (Dense)              (None, 8)                 2056
=================================================================
Total params: 306,440
Trainable params: 306,440
Non-trainable params: 0
_________________________________________________________________
```



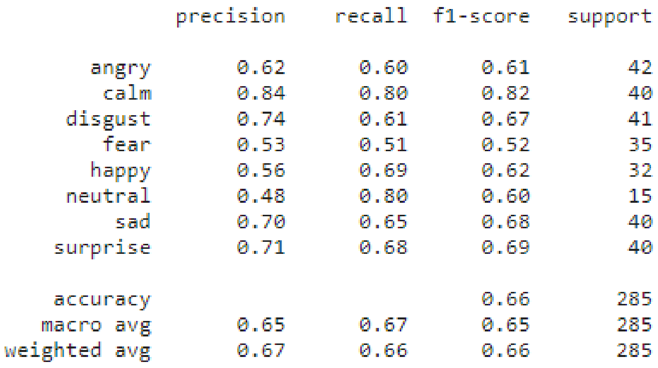
```
12/12 [==============================] - 0s 14ms/step - loss: 0.1372 - accuracy: 0.9555 - val_loss: 0.8644 - val_accuracy: 0.7615
Epoch 995/1000
12/12 [==============================] - 0s 14ms/step - loss: 0.1450 - accuracy: 0.9545 - val_loss: 0.8885 - val_accuracy: 0.7596
Epoch 996/1000
12/12 [==============================] - 0s 13ms/step - loss: 0.1417 - accuracy: 0.9535 - val_loss: 0.9161 - val_accuracy: 0.7538
Epoch 997/1000
12/12 [==============================] - 0s 13ms/step - loss: 0.1571 - accuracy: 0.9497 - val_loss: 0.8779 - val_accuracy: 0.7635
Epoch 998/1000
12/12 [==============================] - 0s 13ms/step - loss: 0.1498 - accuracy: 0.9542 - val_loss: 0.8700 - val_accuracy: 0.7481
Epoch 999/1000
12/12 [==============================] - 0s 13ms/step - loss: 0.1506 - accuracy: 0.9504 - val_loss: 0.8801 - val_accuracy: 0.7596
Epoch 1000/1000
12/12 [==============================] - 0s 13ms/step - loss: 0.1451 - accuracy: 0.9542 - val_loss: 0.8619 - val_accuracy: 0.7558
```

Accuracy of our model on test data : 65.61403274536133 %



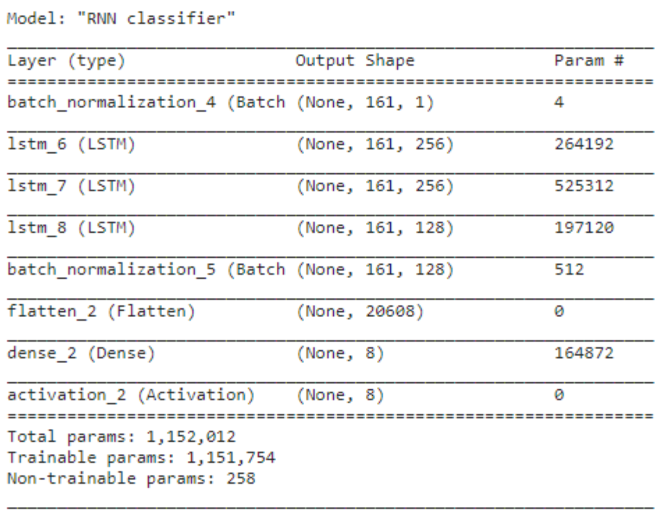


| | angry | calm | disgust | fear | happy | neutral | sad | surprise |
|---|---|---|---|---|---|---|---|---|
| angry | 25 | 0 | 2 | 9 | 3 | 1 | 1 | 1 |
| calm | 0 | 32 | 0 | 0 | 0 | 7 | 1 | 0 |
| disgust | 6 | 0 | 25 | 1 | 2 | 0 | 2 | 5 |
| fear | 5 | 0 | 1 | 18 | 5 | 0 | 5 | 1 |
| happy | 1 | 1 | 1 | 3 | 22 | 0 | 2 | 2 |
| neutral | 0 | 2 | 0 | 0 | 0 | 12 | 0 | 1 |
| sad | 0 | 3 | 3 | 3 | 1 | 3 | 26 | 1 |
| surprise | 3 | 0 | 2 | 0 | 6 | 2 | 0 | 27 |



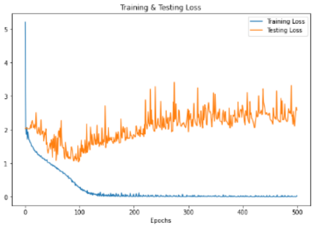
```
              precision    recall  f1-score   support

       angry       0.62      0.60      0.61        42
        calm       0.84      0.80      0.82        40
     disgust       0.74      0.61      0.67        41
        fear       0.53      0.51      0.52        35
       happy       0.56      0.69      0.62        32
     neutral       0.48      0.80      0.60        15
         sad       0.70      0.65      0.68        40
    surprise       0.71      0.68      0.69        40

    accuracy                           0.66       285
   macro avg       0.65      0.67      0.65       285
weighted avg       0.67      0.66      0.66       285
```

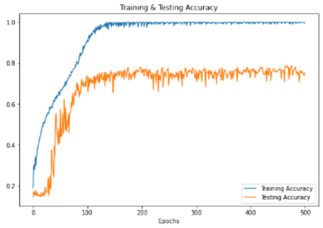
## RNN

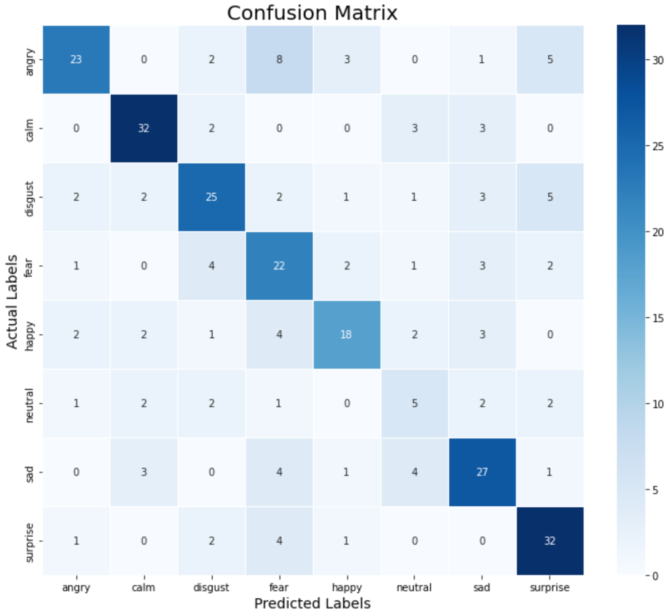


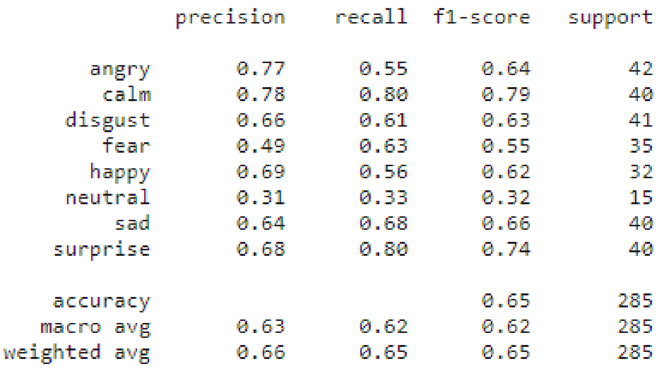
```
Model: "RNN classifier"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
batch_normalization_4 (Batch (None, 161, 1)            4
_________________________________________________________________
lstm_6 (LSTM)                (None, 161, 256)          264192
_________________________________________________________________
lstm_7 (LSTM)                (None, 161, 256)          525312
_________________________________________________________________
lstm_8 (LSTM)                (None, 161, 128)          197120
_________________________________________________________________
batch_normalization_5 (Batch (None, 161, 128)          512
_________________________________________________________________
flatten_2 (Flatten)          (None, 20608)             0
_________________________________________________________________
dense_2 (Dense)              (None, 8)                 164872
_________________________________________________________________
activation_2 (Activation)    (None, 8)                 0
=================================================================
Total params: 1,152,012
Trainable params: 1,151,754
Non-trainable params: 258
_________________________________________________________________
```

Accuracy of our model on test data : 64.56140279769897 %









```
              precision    recall  f1-score   support

       angry       0.77      0.55      0.64        42
        calm       0.78      0.80      0.79        40
     disgust       0.66      0.61      0.63        41
        fear       0.49      0.63      0.55        35
       happy       0.69      0.56      0.62        32
     neutral       0.31      0.33      0.32        15
         sad       0.64      0.68      0.66        40
    surprise       0.68      0.80      0.74        40

    accuracy                           0.65       285
   macro avg       0.63      0.62      0.62       285
weighted avg       0.66      0.65      0.65       285
```

### Final result

| **Metrics** | | **Random Forest** | **CNN** | **RNN (LSTM)** |
|---|---|---|---|---|
| **accuracy** | | 0.61 | 0.66 | 0.65 |
| **f1-score** | angry | 0.72 | 0.61 | 0.64 |
| | calm | 0.71 | 0.82 | 0.79 |
| | disgust | 0.55 | 0.67 | 0.63 |
| | fear | 0.49 | 0.52 | 0.55 |
| | happy | 0.54 | 0.62 | 0.62 |
| | neutral | 0.42 | 0.60 | 0.32 |
| | sad | 0.59 | 0.68 | 0.66 |
| | surprise | 0.69 | 0.69 | 0.74 |
| | Macro avg | 0.58 | 0.65 | 0.62 |

In this study, we compared the performance of three methods for speech emotion recognition on the RAVDESS dataset: a random forest classifier (RFC), a convolutional neural network (CNN), and a recurrent neural network (RNN) with long short-term memory (LSTM) cells.

The results of the comparison showed that the CNN had the highest accuracy, with a score of 0.66, followed closely by the RNN with an accuracy of 0.65. The difference in accuracy between the CNN and RNN was not large, indicating that both methods performed similarly well on this task. The random forest classifier had the lowest accuracy, with a score of 0.61.

In terms of the f1-score macro average, the CNN had the highest score of 0.65, followed by the RNN with a score of 0.62 and the random forest classifier with a score of 0.58. This suggests that the CNN was able to achieve a good balance between precision and recall, and was therefore able to make more accurate predictions overall.

These results suggest that deep learning algorithms can outperform traditional machine learning methods for speech emotion recognition, at least when the number of samples is not too small. However, it is important to note that the relative performance of different algorithms can depend on many factors, including the specific characteristics of the dataset and the complexity of the task. In general, it seems that deep learning algorithms are not always superior to machine learning algorithms, but if the number of samples is sufficient, deep learning methods may have an advantage.

## REFERENCES

Livingstone SR, Russo FA (2018) The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. PLoS ONE 13(5): e0196391. doi:10.1371/journal.pone.0196391.

Huang, Z., Dong, M., Mao, Q., & Zhan, Y. (2014). Speech Emotion Recognition Using CNN. Proceedings of the ACM International Conference on Multimedia - MM '14. doi:10.1145/2647868.2654984

Livingstone, S. R., & Russo, F. A. (2018). The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. PLOS ONE, 13(5), e0196391. doi:10.1371/journal.pone.0196391

Puri, Tanvi & Soni, Mukesh & Dhiman, Gaurav & Khalaf, Osamah & Alazzam, Malik & Khan, Ihtiram. (2022). Detection of Emotion of Speech for RAVDESS Audio Using Hybrid Convolution Neural Network. Journal of Healthcare Engineering. 2022. 1-9. 10.1155/2022/8472947.

Bhavan, Anjali & Sharma, Mohit & Piplani, Mehak & Chauhan, Pankaj & Hitkul, & Shah, Rajiv Ratn. (2020). Deep Learning Approaches for Speech Emotion Recognition. 10.1007/978-981-15-1216-2_10.