Speech Emotion Recognition

Waseema Begum and **Praveena Kumari Silmala**

Abstract

This technical report presents a study on speech emotion recognition using Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) models. The analysis is conducted on the RAVDESS, TESS, CREMA-D, and SAVEE datasets, with additional details on data preprocessing, feature extraction, and model evaluation.

1 Introduction

Speech Emotion Recognition (SER) stands as a critical area within affective computing, aiming to decipher and interpret emotional states conveyed through speech signals. In this project, we explore the effectiveness of Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) architectures in the context of SER, leveraging datasets from RAVDESS, TESS, CREMA-D, and SAVEE. The initial evaluation showcases distinct performance outcomes, with the LSTM model achieving a test accuracy of 15%, while the CNN model excels with a notable 62% test accuracy.

To enhance the models' understanding of diverse emotional expressions, we implemented a robust preprocessing pipeline. Audio data from the selected datasets underwent augmentation techniques, including noise injection, time shifting, pitch variation, and speed changes. Following augmentation, we converted the audio data to feature vectors using established methods. The resulting feature vectors were organized into a structured pandas dataframe and served to deep learning models for the further study. This project aims to contribute insights into the nuanced realm of SER, exploring the strengths and limitations of LSTM and CNN models in capturing and classifying emotional cues within speech signals.

2 Data Preprocessing and Feature Extraction

Audio data from the RAVDESS, TESS, CREMA-D, and SAVEE datasets underwent a meticulous preprocessing pipeline to ensure the quality and diversity of the training data. The following steps outline the data preprocessing and feature extraction procedures:

2.1 Augmentation Techniques

To enhance the diversity of the dataset and improve model robustness, various augmentation techniques were applied to the raw audio data. These included:

- **Noise Injection:** Random noise was added to the audio signals to simulate real-world variations.
- **Time Shifting:** The audio samples were temporally shifted to introduce variability in timing.
- **Pitch Variation:** Changes in pitch were applied to account for pitch variations in natural speech.
- **Speed Changes:** Alterations in speed were made to capture variations in speech rate.

2.2 Feature Extraction Methods

The augmented audio data was then subjected to feature extraction techniques to convert the time-domain audio signals into meaningful feature vectors for machine learning models. The following feature extraction methods were employed:

- **Zero Crossing Rate (ZCR):** A measure of the rate at which the audio signal changes its sign, providing information about the frequency content.
- **Chroma Short-Time Fourier Transform (Chroma_stft):** Captures the energy distribution of pitches in the audio signal.

- **Mel-Frequency Cepstral Coefficients (MFCCs):** Represent the spectral characteristics of the audio signal, commonly used in speech and audio processing.

- **Root Mean Square (RMS) Value:** Represents the energy of the audio signal.

- **Mel Spectrogram:** Provides a visual representation of the spectrum of frequencies in the audio signal over time.

These feature extraction methods collectively provided a rich representation of the acoustic characteristics of the speech signals, capturing both temporal and frequency information essential for emotion recognition.

The resulting feature vectors from the augmented and extracted data were then organized into a structured pandas dataframe for further analysis and model training.

3 Data Representation

The resulting feature vectors, extracted through the aforementioned techniques (Zero Crossing Rate (ZCR), Chroma Short-Time Fourier Transform (Chroma_stft), Mel-Frequency Cepstral Coefficients (MFCCs), Root Mean Square (RMS) Value, and Mel Spectrogram), were meticulously organized into a structured pandas dataframe. The dataframe serves as the foundational data structure for our subsequent modeling and analysis.

Each row of the dataframe corresponds to an individual audio segment, and the columns capture various aspects of the extracted features. Specifically, the dataframe comprises 36486 rows and 163 columns. These columns encapsulate the following information:

- **Feature Columns:** Columns 1 to 162 correspond to the extracted features for each audio segment. These include numerical representations derived from ZCR, Chroma_stft, MFCCs, RMS Value, and Mel Spectrogram, capturing nuanced aspects of the speech signal indicative of emotional content.

- **Label Column:** The last column in the dataframe is designated as the 'label' column. This categorical attribute signifies the emotional expression associated with each audio segment. There are eight unique labels representing different emotions: happiness, sadness, anger, fear, surprise, disgust, and neutral expressions.

This structured representation enables a clear association between the extracted features and the corresponding emotional labels, forming the basis for our supervised learning approach. The comprehensive nature of the dataframe ensures that the model is exposed to a diverse range of features, facilitating a nuanced understanding of the emotional content embedded in the speech signals.

This detailed organization of the data is crucial for both transparency in the modeling process and the interpretability of the subsequent results.

4 Class Imbalance Treatment

Class imbalance is a common challenge in emotion recognition datasets, where certain emotion classes may have significantly fewer instances than others. To address this issue and ensure that the model learns from all classes effectively, we applied class weights during the training phase.

4.1 Class Weighting Strategy

Class weights were assigned inversely proportional to the class frequencies, aiming to penalize misclassifications of under-represented classes more heavily. Let w_c be the weight assigned to class c and N_c be the number of instances in class c, the class weight w_c was calculated as:

$$w_c = \frac{totalnumberofinstances}{numberofclasses \times N_c}$$

This weighting strategy allowed the model to give more importance to minority classes, preventing them from being overshadowed by the majority classes during training.

4.2 Effect on Model Training

By incorporating class weights into the training process, we aimed to mitigate the impact of class imbalance on the model's performance. The weighted loss function provided a more balanced optimization landscape, encouraging the model to learn representative features from all emotion classes.

This strategy proved crucial in enhancing the model's ability to generalize across different emotions, particularly for those with limited instances in the dataset. The resultant model exhibited improved accuracy in predicting under-represented

emotions, contributing to a more comprehensive and equitable emotion recognition system.

5 Data Splitting

Effective evaluation of the speech emotion recognition models necessitates a careful division of the dataset into training, validation, and test sets. The following protocol was adopted to ensure robust model assessment:

5.1 Training Set

The training set, comprising a substantial portion of the dataset, served as the primary source for model parameter learning. The models learned patterns and features associated with various emotional expressions from this set. Specifically, 80% of the data, randomly selected, was allocated to the training set.

5.2 Validation Set

To fine-tune model hyperparameters and prevent overfitting, a separate validation set was established. This set, constituting 10% of the total data, was employed during the training phase. Models were evaluated on the validation set after each epoch, and hyperparameter adjustments were made accordingly.

5.3 Test Set

The final evaluation of model generalization and performance occurred on the test set, which remained unseen during the training and validation stages. This set, encompassing the remaining 10% of the data, provided an unbiased assessment of the models' ability to recognize emotions in unseen instances.

The rationale behind this division was to ensure that the models neither memorized the training data nor overfit to specific patterns. By maintaining a clear distinction between training, validation, and test sets, the models were encouraged to generalize well to unseen instances, promoting their utility in real-world applications.

It is crucial to note that this data splitting strategy was consistently applied across experiments involving both the LSTM and CNN models, ensuring fair and comparable evaluations of their respective performances.

6 Callback Functions

During the training phase, employing callback functions is crucial for optimizing model performance, preventing overfitting, and ensuring efficient convergence. Two essential callback functions used in this study are 'EarlyStopping' and 'ReduceLROnPlateau'.

6.1 EarlyStopping

The 'EarlyStopping' callback is employed to monitor the validation loss during training. It halts the training process if the validation loss fails to decrease for a specified number of consecutive epochs, determined by the 'patience' parameter. The model's weights are restored to the best performing configuration, preserving the optimal state.

For both the LSTM and CNN models, the training process is configured to halt if there is no improvement in validation loss for 20 consecutive epochs. This helps prevent overfitting and ensures that the model is trained effectively without unnecessary iterations.

6.2 ReduceLROnPlateau

The 'ReduceLROnPlateau' callback is utilized to dynamically adjust the learning rate during training. It monitors the validation loss, and if the validation loss plateaus for a specified number of epochs ('patience'), the learning rate is reduced by a factor ('factor'). This adaptive learning rate reduction helps navigate the model through challenging regions of the loss landscape and enhances convergence.

For both models, if the validation loss remains stagnant for three consecutive epochs, the learning rate is reduced by a factor of 0.2. The minimum learning rate is capped at 1e-6. This dynamic adjustment contributes to the model's adaptability and robustness during training.

6.3 Training with Callbacks

The LSTM and CNN models are trained using these callback functions during the fitting process

These callback functions, in conjunction with the validation set, play a crucial role in optimizing model training, preventing overfitting, and improving the generalization of the models.

7 Model Architecture

Long Short-Term Memory (LSTM) architecture, known for its proficiency in capturing sequential

```
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, num_timesteps, 128) 84480
lstm_1 (LSTM)                (None, 128)               131584
dense (Dense)                (None, 128)               16512
dropout (Dropout)            (None, 128)               0
dense_1 (Dense)              (None, num_classes)        1032
=================================================================
Total params: 233,608
Trainable params: 233,608
Non-trainable params: 0
```

Figure 1: LSTM Model Architecture

dependencies in time-series data. The Keras library is utilized for the model implementation.

7.1 Input Layer

The input layer of the LSTM model is tailored to accommodate the shape of the input data, crucial for ensuring compatibility between the data and the model. The input shape is defined as `(num_timesteps, num_features)`, effectively capturing the temporal dynamics of the speech signals.

7.2 LSTM Layers

The core of the model comprises two LSTM layers stacked sequentially. The first LSTM layer consists of 128 memory units and is configured to return sequences, facilitating the propagation of temporal information to subsequent layers. This sequential information is further processed by a second LSTM layer with 128 memory units.

7.3 Dense Layers

Following the LSTM layers, two fully connected (dense) layers are added to capture higher-level representations of the temporal features. The first dense layer consists of 128 units and utilizes the rectified linear unit (ReLU) activation function, introducing non-linearity to the model. This layer is succeeded by a dropout layer with a dropout rate of 0.5, serving as a regularization mechanism to mitigate overfitting. The final dense layer employs the softmax activation function to produce class probabilities for the eight distinct emotion categories.

7.4 Model Compilation

The model is compiled using the categorical cross-entropy loss function, suitable for multi-class classification tasks. The Adam optimizer is employed to adaptively adjust learning rates during training, and the model's performance is evaluated based on accuracy.

The LSTM model exhibits a trainable parameter count of 233,608, making it a reasonably sized model for the given task. The architecture is designed to effectively learn temporal patterns within the speech data.

7.5 Convolutional Neural Network (CNN) Model Architecture

The Convolutional Neural Network (CNN) is a widely adopted architecture for image and sequential data processing, and in this work, it is harnessed for Speech Emotion Recognition (SER). The CNN model is implemented using the Keras library.

7.5.1 Model Definition

The CNN architecture is designed to automatically learn hierarchical features from the input spectrogram representations of speech signals.

7.5.2 Convolutional Layers

The CNN model starts with a series of convolutional layers aimed at extracting essential features from the input spectrogram. Three convolutional layers are utilized with increasing filter sizes (64, 128, and 256), each followed by max-pooling layers to downsample the learned features.

7.5.3 Flatten Layer

Following the convolutional layers, a flatten layer is introduced to transform the 3D output into a 1D vector, preparing the data for processing by fully connected layers.

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 160, 64)           256
max_pooling1d (MaxPooling1D) (None, 80, 64)            0
conv1d_1 (Conv1D)            (None, 78, 128)           24704
max_pooling1d_1 (MaxPooling1 (None, 39, 128)           0
conv1d_2 (Conv1D)            (None, 37, 256)           98560
max_pooling1d_2 (MaxPooling1 (None, 18, 256)           0
flatten (Flatten)            (None, 4608)              0
dense_2 (Dense)              (None, 128)               589952
dropout_2 (Dropout)          (None, 128)               0
dense_3 (Dense)              (None, 64)                8256
dropout_3 (Dropout)          (None, 64)                0
dense_4 (Dense)              (None, num_classes)        520
=================================================================
Total params: 723,248
Trainable params: 723,248
Non-trainable params: 0
```

Figure 2: CNN Model Architecture

7.5.4 Dense Layers

The flattened output is then fed into densely connected layers. The first dense layer consists of 128 units with the rectified linear unit (ReLU) activation function. To prevent overfitting, a dropout layer with a dropout rate of 0.5 is applied. This process is repeated with another dense layer comprising 64 units and a dropout layer. The final dense layer utilizes the softmax activation function to produce class probabilities for the eight distinct emotion categories.

7.5.5 Model Compilation

The CNN model is compiled using the categorical cross-entropy loss function, suitable for multi-class classification tasks. The Adam optimizer is employed to adaptively adjust learning rates during training, and model performance is evaluated based on accuracy.

The CNN model is trained with class weights to address class imbalance. Early stopping and learning rate reduction on plateau callbacks are applied during training to enhance convergence and prevent overfitting.

8 Model Training and Evaluation

8.1 Long Short-Term Memory (LSTM) Model

The LSTM model was trained on the preprocessed speech emotion recognition dataset. The LSTM model was designed to capture temporal dependencies in the audio sequences, making it well-suited for analyzing speech data. The training process involved the use of class weights to account for the imbalanced distribution of emotion classes. Additionally, early stopping and learning rate reduction on plateau callbacks were employed to enhance the model's convergence and prevent overfitting.

8.1.1 Training Configuration

The LSTM model was trained for 20 epochs with a batch size of 32 on the training set. The validation set was used to monitor the model's performance during training. The resulting training history, including accuracy and loss curves, was recorded for subsequent analysis.

The training process involved minimizing the categorical crossentropy loss using the Adam optimizer. To address the class imbalance in the dataset, class weights were applied during training. The LSTM model underwent an extensive training regimen on the 'Train' dataset, with periodic validation on the 'Validation' set to monitor generalization performance.

The LSTM model training consumed approximately 2 hours to complete the 20 epochs, underscoring the computational intensity of training deep learning models on the provided dataset. Due to the inherent challenges in modeling sequential data, the LSTM model achieved only a test accuracy of 15%.

8.1.2 Overall Performance

The training history reveals intricate details of the model's learning process. Notably, the initial accuracy stands at 18.1%, and the validation accuracy at 10.14% for the first epoch. As training progresses, the model achieves an accuracy of 37.01% on the training set and 34.48% on the validation set by the end of the 20th epoch.

The model achieved an overall accuracy of 35% on the evaluation dataset, comprising 3649 instances.

8.1.3 Emotion-Specific Metrics

- **Angry:** Precision (64%), Recall (55%), F1-score (59%)
- **Calm:** Precision (20%), Recall (74%), F1-score (31%)
- **Disgust:** Precision (28%), Recall (8%), F1-score (13%)
- **Fear:** Precision (49%), Recall (14%), F1-score (22%)
- **Happy:** Precision (37%), Recall (21%), F1-score (27%)
- **Neutral:** Precision (24%), Recall (71%), F1-score (36%)
- **Sad:** Precision (51%), Recall (26%), F1-score (34%)
- **Surprise:** Precision (31%), Recall (79%), F1-score (45%)

8.1.4 Macro and Weighted Averages

- **Macro Average:** Precision (38%), Recall (44%), F1-score (33%)
- **Weighted Average:** Precision (41%), Recall (35%), F1-score (32%)

8.1.5 Insights

The classification report offers insights into the model's strengths and weaknesses across diverse emotion classes. Notably, the model excels in recognizing 'Calm,' 'Neutral,' and 'Surprise' emotions while facing challenges in accurately identifying 'Disgust' and 'Fear' emotions.

This nuanced analysis facilitates a deeper understanding of the model's behavior, paving the way for future improvements and refinements.

8.2 Convolutional Neural Network (CNN) Model

The CNN model was designed to leverage local patterns and spectral information within the feature vectors. The input to the CNN consisted of the same preprocessed feature vectors, organized into a tensor suitable for convolutional operations. The architecture included convolutional layers, max-pooling layers, and fully connected layers to capture hierarchical features. It underwent a meticulous training regimen over 20 epochs. The training details, key metrics, and insights are presented below.

8.2.1 Training Configuration

The CNN model was trained with a batch size of 32 and over 20 epochs. Class weights were applied to address class imbalance, and the training process incorporated early stopping and learning rate reduction on plateau callbacks.

8.2.2 Overall Performance

The training history showcases the model's progression. Notably, the initial accuracy stands at 34.57%, and the validation accuracy at 47.05% for the first epoch. As training progresses, the model achieves an accuracy of 66.46% on the training set and 61.77% on the validation set by the end of the 20th epoch. The CNN model training consumed approximately 10 minutes to complete the 20 epochs, demonstrating efficiency in convergence.

The overall accuracy of the model on the evaluation dataset is 62%, involving a total of 3649 instances.

8.2.3 Emotion-Specific Metrics

- **Angry:** Precision (76%), Recall (76%), F1-score (76%)
- **Calm:** Precision (51%), Recall (97%), F1-score (67%)

- **Disgust:** Precision (65%), Recall (42%), F1-score (51%)
- **Fear:** Precision (70%), Recall (43%), F1-score (53%)
- **Happy:** Precision (56%), Recall (57%), F1-score (57%)
- **Neutral:** Precision (44%), Recall (72%), F1-score (55%)
- **Sad:** Precision (62%), Recall (61%), F1-score (62%)
- **Surprise:** Precision (76%), Recall (95%), F1-score (84%)

8.2.4 Macro and Weighted Averages

- **Macro Average:** Precision (63%), Recall (68%), F1-score (63%)
- **Weighted Average:** Precision (63%), Recall (61%), F1-score (60%)

8.2.5 Insights

The updated classification report indicates improvements in several emotion classes, with notable enhancements in precision, recall, and F1-scores for 'Calm,' 'Surprise,' and 'Angry.' The overall macro and weighted averages reflect a balanced performance across diverse emotions.

This refined analysis provides valuable insights into the model's capabilities, offering a comprehensive understanding of its strengths and areas for further refinement.

9 Conclusion

The journey of exploring Speech Emotion Recognition (SER) has been both insightful and challenging. The integration of Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) models provided a comprehensive approach to deciphering emotional nuances from audio data. While the LSTM model faced challenges, the CNN model showcased promising results. The iterative exploration has not only shed light on the complexities of speech emotion recognition but has also laid the groundwork for future enhancements and innovations in the realm of audio-based emotion analysis. As we continue to refine our models and explore novel approaches, the quest for more accurate and nuanced speech emotion recognition remains an exciting and evolving journey.

References

[1] Javier de Lope, Manuel Graña, *Title of the Fourth Article*, *Neurocomputing*, Volume 528, 1 April 2023, Pages 1-11, https://www.sciencedirect.com/science/article/pii/S0925231223000103.

[2] F. Andayani, L. B. Theng, M. T. Tsun, C. Chua, *Title of the Third Article*, *IEEE Access*, 10 (2022), pp. 36018-36027, https://ieeexplore.ieee.org/document/9745599.

[3] B. J. Abbaschian, D. Sierra-Sosa, A. Elmaghraby, *Title of the First Article*, *Sensors*, 21 (2021), p. 1249, https://www.mdpi.com/1424-8220/21/4/1249.

[4] M. B. Akçay, K. Oğuz, *Title of the Second Article*, *Speech Communication*, 116 (2020), pp. 56-76, https://www.sciencedirect.com/science/article/pii/S0167639319302262.

[5] Margaret Lech, Melissa Stolar, Christopher Best, Robert Bolia, *Title of the Fifth Article*, *Human-Media Interaction*, Volume 2 - 2020, 26 May 2020, https://www.frontiersin.org/articles/10.3389/fcomp.2020.00014/full.