Deep Learning for Emotion Recognition: Exploring Image and Audio Modalities

Oumaima Chater
CentraleSupélec
3 Rue Joliot Curie,91190, Gif-sur-Yvette
oumaima.chater@student-cs.fr

Abstract

Deep learning has been shown to be an effective approach for recognizing emotions, especially when applied to various modalities such as images and audio. This project aims to use separate deep learning models for image and audio data to achieve reliable emotion recognition. To recognize emotions from images, Convolutional Neural Networks (CNNs) are used to extract detailed facial features, capturing subtle emotional expressions. Meanwhile, for audio-based emotion recognition, Recurrent Neural Networks (RNNs) or Convolutional Neural Networks process spectrograms or raw audio signals. This captures intonation, pitch, and speech patterns that indicate emotional states. This modular approach allows for specialized processing tailored to the characteristics of each modality, optimizing performance for respective data types. Additionally, the project investigates techniques for integrating information from different modalities, such as multimodal fusion, to improve the accuracy and robustness of the emotion recognition system. Experimental evaluations demonstrate the effectiveness of the proposed methodology, showcasing its ability to accurately recognize diverse emotional states across various datasets. This project contributes to advancing emotion recognition research by employing separate models for image and audio data. The potential applications of this research include human-computer interaction, affective computing, and beyond.

1. Introduction

Emotion recognition is a crucial field in various domains, including human-computer interaction, affective computing, and psychology. The ability to discern and understand human emotions expressed through different modalities, such as images and audio, has significant implications for enhancing user experiences, providing personalized services, and developing support systems for mental health.

Deep learning techniques, specifically Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have revolutionized emotion recognition by extracting intricate features from complex data. CNNs are particularly effective in analyzing visual data, such as facial expressions, by capturing subtle nuances and spatial relationships. Conversely, Recurrent Neural Networks (RNNs) or Convolutional Neural Networks (CNNs) are proficient in processing audio data, capturing acoustic features such as intonation, pitch variations, and speech patterns that effectively convey emotional states.

This project aims to leverage deep learning models specifically tailored for image and audio data to enhance emotion recognition. By exploiting the unique characteristics of each modality, we aim to develop a more comprehensive and robust emotion recognition system. Separate models for image and audio data are used to enable specialized processing that is optimized for the unique characteristics of each modality.

The rationale for using separate models is based on the inherent differences between image and audio data, as well as the complexity of emotional cues embedded within each modality. By tailoring deep learning architectures for image and audio data, we aim to optimize the recognition process and improve the overall accuracy of the system.

In this paper, we present an overview of the significance of emotion recognition and outline the general approach adopted in this project. Subsequent sections will delve into the methodologies for image and audio-based emotion recognition, and experimental evaluations. Through this research, we aim to contribute to the advancement of emotion recognition technology and its applications across diverse domains.

2. Facial Emotion Recognition

2.1. Literature Review

Facial emotion recognition has received considerable attention in recent years due to its broad applications in fields such as human-computer interaction, affective computing, and psychology. This section provides a review

of the existing literature on facial emotion recognition, covering both traditional and deep learning-based approaches.

2.1.1 Traditional Approaches

Traditionally, facial emotion recognition has been tackled using various machine learning algorithms. In a Kaggle notebook [1], several methods were explored, including Support Vector Machine (SVM), Random Classifier, KNN Algorithms, Logistic Regression, among others. These methods have been widely used due to their simplicity and effectiveness in capturing facial features and classifying emotions.

2.1.2 Deep Learning Techniques

With the advent of deep learning, significant advancements have been made in facial emotion recognition. In a paper titled "Facial Emotion Recognition: A multi-task approach using deep learning" by Saroop et al. [2], a multi-task deep learning approach is proposed. The authors leverage deep neural networks to simultaneously recognize facial expressions and detect facial landmarks, achieving state-of-the-art performance in facial emotion recognition tasks.

2.2. Dataset

The dataset used in this study is the CK+ dataset, a widely used benchmark dataset for facial expression recognition research. Comprising 920 facial image sequences, the CK+ dataset captures various emotions under different lighting conditions and facial expressions. Each image sequence corresponds to a specific emotion class, including anger, contempt, disgust, fear, happiness, sadness, and surprise, with grayscale images having a resolution of 48×48 pixels. Additionally, each image sequence is labeled with the corresponding emotion class, providing ground truth annotations essential for training and evaluating convolutional neural networks for accurate facial emotion recognition.

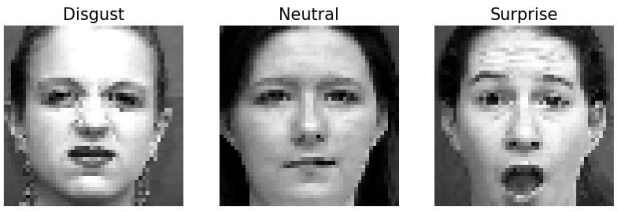

Figure 1: Facial expression samples from the CK+ dataset.

2.3. Approach

Our approach to facial emotion recognition relies on Convolutional Neural Networks (CNNs), a powerful class of deep learning models specifically designed for processing spatial data such as images. The architecture of our models is based on the fundamental principles of CNNs, comprising several layers that perform feature extraction and classification tasks.

2.3.1 Feature Extraction

The CNN architectures begin with convolutional layers, which use learnable filters to extract relevant features from input images. Each filter captures distinct spatial patterns in the input data, allowing for hierarchical feature representation. The resulting feature maps encode increasingly abstract representations of the input images, capturing essential facial expression characteristics.

The convolutional operation can be expressed mathematically:

$$Conv(I) = \sigma(W * I + b) \quad (1)$$

where I represents the input image, W denotes the filter weights, b is the bias term, $*$ denotes the convolution operation, and σ represents the activation function, typically ReLU.

2.3.2 Non-linear Activation

To introduce non-linearity into the model and enable complex feature mapping, rectified linear unit (ReLU) activation functions are applied after each convolutional operation. ReLU activation functions ensure that the output of each neuron is non-negative, effectively overcoming the vanishing gradient problem and accelerating convergence during training.

The ReLU activation function is defined as:

$$f(x) = \max(0, x) \quad (2)$$

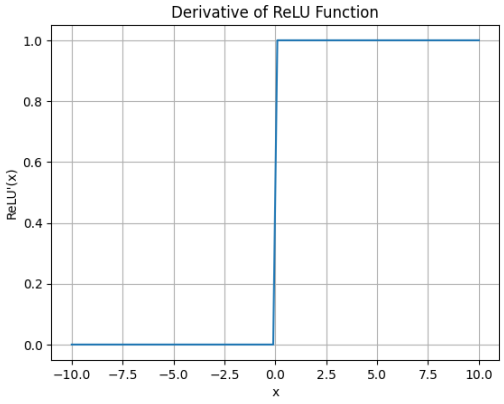

Figure 2: Derivative of ReLU function.

As depicted in the figure [2], the derivative of ReLU is 1 for input values greater than zero, indicating that the function is fully activated and allows gradients to flow unchanged during backpropagation. However, for input values less than or equal to zero, the derivative of ReLU is 0, effectively blocking the flow of gradients and preventing weight updates in the corresponding neurons. This behavior is characteristic of ReLU's sparsity-inducing property, where only neurons with positive activations contribute to the learning process, while others remain inactive.

2.3.3 Spatial Pooling

Subsequent to the convolutional layers, spatial pooling operations are employed to down-sample the feature maps, reducing their spatial dimensions while retaining the most salient information. Max-pooling, a commonly used pooling technique, selects the maximum value within each pooling region, effectively highlighting the most relevant features.

2.3.4 Flattening and Dense Layers

Following spatial pooling, the output feature maps are flattened into one-dimensional vectors to transition from the convolutional to the fully connected layers. Fully connected layers, also known as dense layers, enable global feature representation by connecting every neuron from one layer to every neuron in the subsequent layer. This dense connectivity facilitates high-level feature learning and abstraction, crucial for complex pattern recognition tasks.

2.3.5 Output Layer

The final layer of our CNN architectures comprises an output layer responsible for emotion classification. This layer typically consists of multiple neurons, each corresponding to a distinct emotion class. Softmax activation is employed to compute the probability distribution over the emotion classes, enabling multiclass classification by assigning the input image to the class with the highest predicted probability.

$$softmax(x)_i = \frac{e^{x_i}}{\sum_j e^{x_j}} \quad (3)$$

2.4. Experiments

In our experiments, we focused on utilizing a Deep Convolutional Neural Network (DCNN) for facial emotion recognition. Initially, we trained the DCNN without data augmentation and then proceeded to incorporate data augmentation techniques. We evaluated the performance of these models using both quantitative and qualitative measures to analyze their efficacy in facial emotion classification.

2.4.1 Without Data augmentation

In this subsection, we present the implementation and evaluation of a Deep Convolutional Neural Network (DCNN) architecture for facial emotion recognition. The DCNN model consists of several convolutional layers followed by max-pooling layers, aimed at extracting essential features from input facial images. The architectural layout is depicted in the figure below.

Layer (type)	Output Shape	Param #
Conv2D (32)	(None, 46, 46, 32)	320
MaxPooling2D	(None, 23, 23, 32)	0
Conv2D (64)	(None, 21, 21, 64)	18496
MaxPooling2D	(None, 10, 10, 64)	0
Conv2D (128)	(None, 8, 8, 128)	73856
MaxPooling2D	(None, 4, 4, 128)	0
Flatten	(None, 2048)	0
Dense (512)	(None, 512)	1049088
Dropout	(None, 512)	0
Dense (256)	(None, 256)	131328
Dropout	(None, 256)	0
Dense (7)	(None, 7)	1799

Figure 3: DCNN's architecture.

The DCNN model is trained using the Adam optimizer and categorical cross-entropy loss function. Training is conducted on the training set and evaluated on the validation set over 40 epochs, with a batch size of 64. Throughout the training process, the model's performance is monitored by recording accuracy metrics and loss values for each epoch. Additionally, plots illustrating the accuracy trends on both the training and validation sets are generated to visualize the model's learning progress.

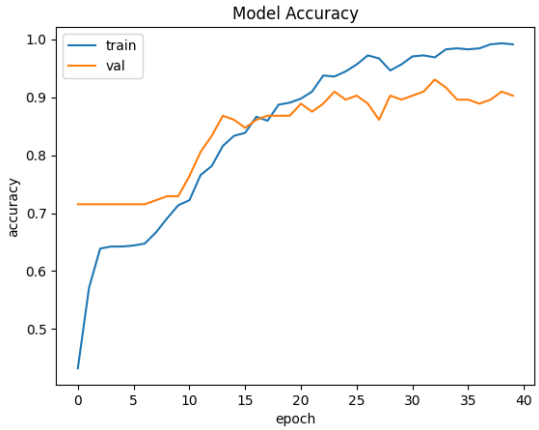

Figure 4: Model accuracy on training and validation sets across epochs.

The training process demonstrates a remarkable improvement in accuracy, starting from an initial accuracy of **43.23%** and reaching a peak accuracy of **99.13%** by the end of the 40 epochs. This substantial increase in accuracy underscores the effectiveness of the model in learning and capturing the underlying patterns in the training data. However, despite achieving high accuracy on the validation set, with a final accuracy of **90.28%**, the model's performance on the unseen test data is notably lower, with an accuracy of **64%**. This discrepancy between the validation and test accuracies suggests potential challenges in generalizing the model's learned features to unseen data, indicating the need for further exploration and optimization to enhance the model's robustness and generalization capabilities.

2.4.2 With Data augmentation

To resolve the problem of overfitting observed in the previous subsection, we employ data augmentation techniques. By augmenting the dataset through methods such as rotation and horizontal flipping, we aim to increase the diversity of the training samples. This approach not only expands the dataset but also introduces variability in the training images, which can help prevent the model from memorizing specific patterns and improve its generalization performance. In this subsection, we describe the application of data augmentation and evaluate its impact on the performance of our facial emotion recognition model.

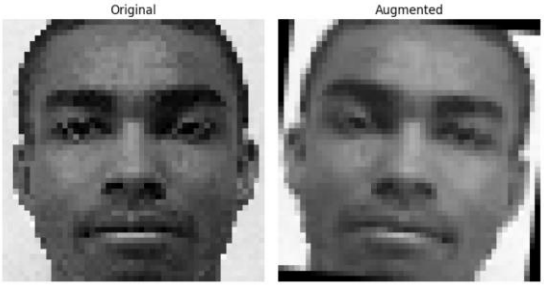

Figure 5: Comparison between an augmented image and its original counterpart.

After employing the same model architecture and training methodology as before, without incorporating data augmentation techniques, we observed the following results: the training accuracy steadily increased throughout the epochs, reaching approximately **98.96%** by the end. Similarly, the validation accuracy showed consistent improvement, peaking at around **90.97%** after 40 epochs. These results indicate that the model successfully learned the patterns present in the training data, demonstrating high accuracy not only on the training set but also on the unseen validation set. Additionally, we evaluated the model on a separate test set and achieved an accuracy of approximately **86.02%,** confirming the robustness of the trained model in classifying facial expressions.

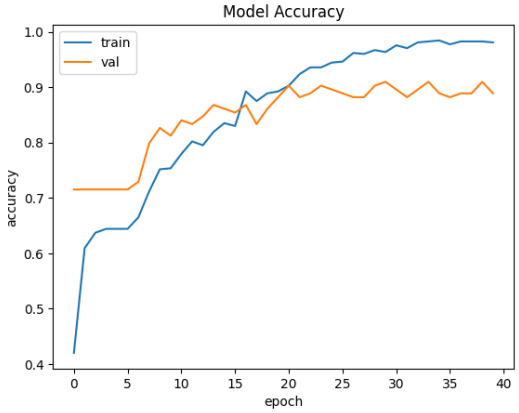

Figure 6: Model accuracy on training and validation sets.

2.5. Next steps

To enhance the emotion recognition model from images and potentially achieve higher accuracy, several strategies can be explored:

1. **Architecture Selection**: Experimenting with different convolutional neural network (CNN) architectures such as ResNet, VGG, or DenseNet can help identify the architecture that best captures spatial features in the images. Additionally, exploring variations in the number of layers, filter sizes, and activation functions can improve the model's ability to extract relevant features associated with emotions.

2. **Data Augmentation**: Augmenting the image dataset with techniques such as rotation, flipping, scaling, and adding noise can increase the diversity of the training data and improve the model's generalization ability. Additionally, utilizing advanced augmentation methods like CutMix or MixUp can further enhance the robustness of the model to variations in input images.

3. **Transfer Learning:** Leveraging pre-trained CNN models trained on large-scale image datasets (e.g., ImageNet) and fine-tuning them on the emotion recognition task can expedite the training process and enable the model to learn discriminative features relevant to emotions. Transfer learning can be particularly beneficial when the available dataset is limited.

4. **Ensemble Learning**: Combining predictions from multiple CNN models trained with different architectures or initializations can often lead to improved performance compared to individual models. Ensemble techniques

such as bagging, boosting, or stacking can be employed to leverage the diversity of multiple models and enhance overall accuracy.

5. **Attention Mechanisms:** Incorporating attention mechanisms within the CNN architecture can enable the model to focus on salient regions of the input images relevant to recognizing emotions. Attention mechanisms can improve the model's interpretability and help it better capture subtle facial expressions associated with different emotions.

6. **Regularization Techniques**: Applying regularization methods such as dropout, batch normalization, or weight decay can prevent overfitting and improve the generalization capability of the model. Regularization techniques help the model learn more robust representations of emotions from the training data.

3. Emotion recognition from audios

3.1. Literature review

Emotion recognition from audio recordings represents a dynamic field at the intersection of sound theory, acoustics, and cutting-edge technology. A series of insightful articles provide valuable insights and methodologies for implementing emotion recognition systems using Python. Tal Baram's exploration on Towards Data Science delves into the application of sound and acoustics theory in developing advanced emotion recognition technologies, offering a foundational understanding for further research [3]. Additionally, Pipe Runner's work on Project Heuristics offers a comprehensive examination of audio signal feature extraction and clustering techniques, providing a robust framework for analyzing and categorizing emotions embedded within audio data [4]. These resources not only showcase the potential of machine learning in understanding and interpreting emotions but also serve as inspirations for future endeavors in the field.

3.2. Datasets

For this project, two primary datasets were utilized: CREMA-D and SAVEE.

3.2.1 CREMA-D (Crowd-sourced Emotional Multimodal Actors Dataset)

The CREMA-D dataset is a rich collection of audio recordings featuring a diverse range of emotional expressions. It consists of audio samples from professional actors who were instructed to portray specific emotions. CREMA-D includes expressions such as happiness, sadness, anger, and fear, among others. These recordings are valuable for emotion recognition tasks due to their authenticity and the controlled environment in which they were captured.

3.2.2 SAVEE (Surrey Audio-Visual Expressed Emotion)

The SAVEE dataset contains audio recordings of actors speaking short sentences with various emotional expressions. Similar to CREMA-D, SAVEE provides a broad spectrum of emotions, including happiness, sadness, anger, and surprise, among others. Each actor in SAVEE was directed to convey specific emotions while uttering the sentences, resulting in a diverse and well-labeled dataset for emotion recognition research.

3.2.3 Dataset Combination and Emotion Distribution

To enhance the diversity and richness of emotional expressions in our dataset, we combined the CREMA-D and SAVEE datasets. This merging process allows for a broader range of emotional contexts, improving the model's ability to generalize across different scenarios. After merging the datasets, we analyzed the distribution of emotions to ensure balance and representation across various emotional states. The plot below illustrates the counts of different emotions in the combined dataset, providing insights into its composition and balance.

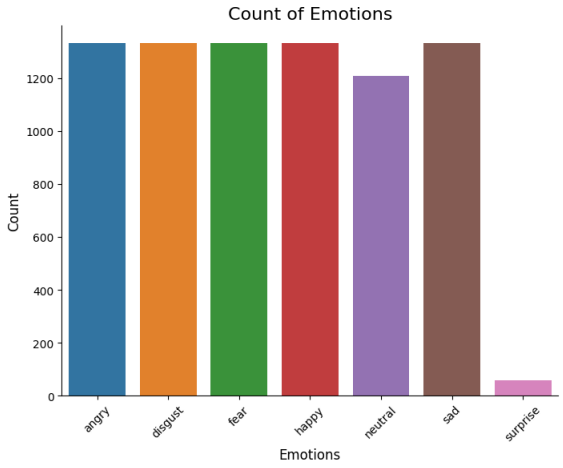

Figure 7: Count of different emotions in the combined dataset

3.3. Approach

In this section, we outline the methodology employed to tackle the emotion recognition task from audio signals. This approach combines data augmentation techniques, feature extraction methods, and deep learning models to effectively capture and classify emotional expressions in audio recordings.

3.3.1 Data augmentation

In the context of audio data augmentation, several techniques can be employed. We will start by using the methods below:

- Stretching and Pitching: This technique involves altering the temporal characteristics of the audio signal. Stretching the audio signal slows down or speeds up the playback rate without affecting the pitch, while pitching modifies the pitch without altering the playback speed.

- Noise Injection: Noise is added to the audio signal to simulate real-world environmental conditions and increase robustness to background noise.

3.3.2 Feature extraction

In feature extraction for audio signal processing, several techniques [4] are commonly used to capture relevant information from the raw audio waveform. Here, we'll delve deeper into some of these techniques:

- **Zero Crossing Rate :**

ZCR measures the rate at which the audio waveform changes its sign, indicating the number of times it crosses the zero axis. Mathematically, it is computed as:

$$ZCR = \frac{1}{N-1}\sum_{n=1}^{N} |x(sign[n]) - x(sign[n-1])| \quad (4)$$

Where N is the length of the audio signal, and x[n] represents the audio sample at time n.

- **Chroma Short-Time Fourier Transform:**

Chroma feature extraction is used to represent the pitch content of an audio signal. It computes the energy distribution of different pitches over time using Short-Time Fourier Transform.

- **Mel-frequency cepstral coefficients:**

MFCCs [5] are widely used for capturing the spectral characteristics of an audio signal. The process involves computing the Discrete Fourier Transform of short overlapping frames of the audio signal, followed by applying a Mel filter bank to obtain the Mel-scale filter outputs.

- **Root Mean Square:**

RMS measures the average power of the audio signal over a short window. It is calculated as:

$$RMS = \sqrt{\frac{1}{N}\sum_{n=1}^{N} x^2[n]} \quad (5)$$

- **Mel Spectrogram:**

Mel spectrogram [6] computes the power spectral density of the audio signal in Mel frequency bins. It provides a visual representation of the frequency content of the audio signal over time, with the frequency axis warped to the Mel scale.

3.4. Experiments

3.4.1 First experiment

The CNN architecture utilized in this project is designed to process audio data for emotion recognition. With a series of convolutional and pooling layers, the model learns hierarchical features from the input audio signals. The architecture begins with Conv1D layers to extract low-level features, followed by max-pooling layers to down sample the feature maps. Subsequent convolutional layers further abstract the learned features, leading to more complex representations. Dropout layers are incorporated to prevent overfitting, and dense layers at the end of the network perform classification tasks.

Layer (type)	Output Shape
Conv1D	(None, 162, 256)
MaxPooling1D	(None, 81, 256)
Conv1D	(None, 81, 256)
MaxPooling1D	(None, 41, 256)
Conv1D	(None, 41, 128)
MaxPooling1D	(None, 21, 128)
Dropout	(None, 21, 128)
Conv1D	(None, 21, 64)
MaxPooling1D	(None, 11, 64)
Flatten	(None, 704)
Dense	(None, 32)
Dropout	(None, 32)
Dense	(None, 7)

Figure 8: CNN architecture

The model was trained using a batch size of 32 and for 100 epochs. Throughout the training process, the model's performance was evaluated on the validation set.

As evident from the plot [9] below, the accuracy achieved by the model is relatively low, with an accuracy of approximately **46.7%** on the test data. This performance indicates suboptimal model effectiveness in accurately classifying the data. Therefore, there is a clear need for improvement to enhance the model's performance.

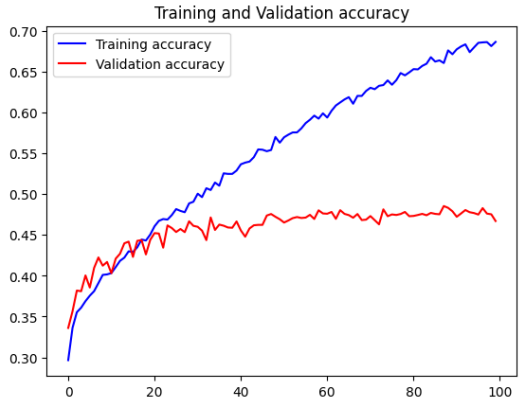

Figure 9: CNN's accuracy on the training and validation set

3.4.2 Second experiment

In addition to the previously implemented augmentation techniques, I've introduced new methods to enhance the diversity of the training data. This includes shifting. Additionally, pitching and stretching have been separately incorporated. These additions aim to broaden the dataset's variability, thereby improving the model's ability to generalize and perform effectively across various audio inputs.

We utilize the identical CNN model employed in the initial experiments and present the training and validation set accuracies in the figure below.

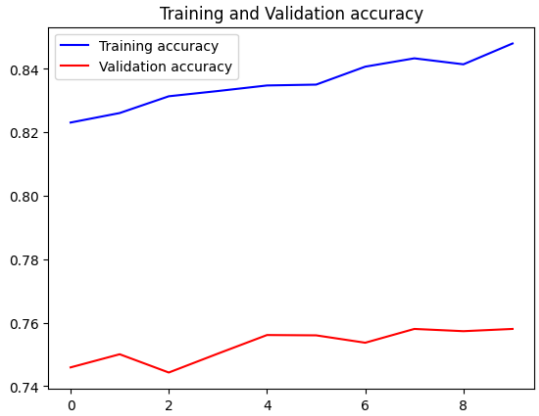

Figure 10: CNN's accuracy on the training and validation set

The model's performance has significantly improved with an accuracy of **63.53%** on the test data compared to the previous accuracy of **46.7%**. This enhancement underscores the effectiveness of the added augmentation techniques, including shifting, pitching, and stretching, which contributed to a more robust and adaptable model. The increased accuracy demonstrates the model's improved ability to generalize and make accurate predictions on unseen data, indicating its enhanced suitability for real-world applications.

3.4.3 Third experiment

To improve the accuracy of our predictions, we will be transitioning to a different model architecture: Long Short-Term Memory. LSTM is a type of recurrent neural network (RNN) that is well-suited for sequence prediction tasks, making it a promising candidate for our experimentation. By leveraging the unique memory cell structure of LSTM networks, we aim to capture long-term dependencies in the input data and enhance the model's ability to learn complex patterns. This shift in architecture presents an opportunity to explore new avenues for improving accuracy and uncovering insights that may have been overlooked with our previous model.

Layer (type)	Output Shape	Param #
LSTM	(None, 162, 256)	264192
Dropout	(None, 162, 256)	0
LSTM	(None, 162, 256)	525312
Dropout	(None, 162, 256)	0
LSTM	(None, 162, 128)	197120
Dropout	(None, 162, 128)	0
LSTM	(None, 162, 64)	49408
Dropout	(None, 162, 64)	0
Flatten	(None, 10368)	0
Dense	(None, 32)	331808
Dropout	(None, 32)	0
Dense	(None, 7)	231

Figure 11: LSTM's architecture

During the training process of the LSTM model, we conducted ten epochs using a batch size of 32. The training process was monitored using both training and validation datasets, allowing us to track the model's performance on unseen data. The training accuracy gradually increased from **82.31%** to **84.80%** over the epochs[12], while the validation accuracy ranged between

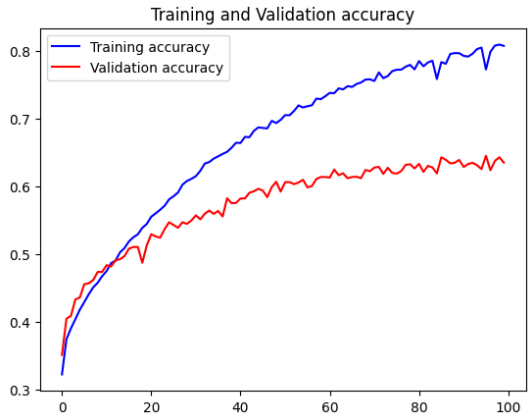

Figure 12: Accuracy of the LSTM on the training and validation set

74.43% and **75.81%**. Despite some fluctuations, the model demonstrated consistent improvement in accuracy, indicating its ability to generalize well to unseen data. The accuracy of our LSTM model on the test data is **75.81%**. This achieved accuracy demonstrates a notable improvement in model performance compared to previous experiments, indicating the effectiveness of the LSTM architecture in capturing temporal dependencies within the data.

3.5. Next steps

To further enhance the model's performance and potentially achieve even higher accuracy, several strategies can be considered:

1. **Hyperparameter Tuning**: Fine-tuning hyperparameters such as learning rate, batch size, number of LSTM units, and dropout rates can significantly impact model performance. Utilizing techniques like grid search or random search can help identify the optimal set of hyperparameters.

2. **Model Architecture Exploration**: Experimenting with different LSTM architectures, including variations in the number of layers, units per layer, and activation functions, can offer insights into the model's capacity to capture complex patterns in the data.

3. **Data Augmentation**: Augmenting the training data with additional techniques such as time shifting, pitch shifting, and adding noise can help diversify the dataset and improve the model's robustness to variations in input signals.

4. **Regularization Techniques**: Implementing regularization methods such as L1 and L2 regularization or applying dropout at different layers can prevent overfitting and enhance the generalization capabilities of the model.

5. **Ensemble Learning**: Combining predictions from multiple LSTM models trained with different initializations or architectures can often lead to superior performance compared to individual models. Ensemble techniques like bagging or boosting can be explored to leverage the diversity of multiple models.

6. **Transfer Learning**: Leveraging pre-trained LSTM models on similar tasks or domains and fine-tuning them on the target dataset can expedite the training process and potentially yield better results, especially when limited labeled data is available.

7. **Data Preprocessing**: Further preprocessing the input data, such as normalization, standardization, or feature scaling, can aid in reducing the impact of noise and irrelevant information, thereby improving the model's ability to extract meaningful features.

By systematically exploring these strategies and iteratively refining the model, it's possible to achieve significant improvements in accuracy and overall performance on the given task.

3.6. Conclusion

In conclusion, our study highlights the effectiveness of deep learning models in emotion recognition tasks using both audio and image data. By refining model architectures and incorporating advanced techniques, we achieved significant improvements in accuracy. These findings underscore the potential of deep learning in advancing affective computing applications and enhancing human-machine interactions.

References

[1] CK+ Pixel Facial Emotion Recognition | Kaggle

[2] A. Saroop, P. Ghugare, S. Mathamsetty, V. Vasani, "Facial Emotion Recognition: A multi-task approach using deep learning", Department of Computer Engineering, K. J. Somaiya College of Engineering, Mumbai, Maharashtra.

[3] Classifying emotions using audio recordings and Python | by Tal Baram | Towards Data Science

[4] Audio signal feature extraction and clustering | by Pipe Runner | Project Heuristics | Medium

[5] MFCC (Mel Frequency Cepstral Coefficients) for Audio format (opengenus.org)

[6] Understanding the Mel Spectrogram | by Leland Roberts | Analytics Vidhya | Medium

[7] FirstName LastName. Frobnication tutorial, 2014. Supplied as supplemental material `tr.pdf`.