Logo

Description automatically generated with low confidence

PROJECT PROPOSAL

**Speech Emotion Recognition Project**

**Group 03:**

DPL302m

Summer Semester - 2025

- Ho Chi Minh City, July 2025 –

**TABLE OF CONTENTS**

[I. Introduction 3](#_1fob9te)

[1. Overview 3](#_3znysh7)

[1.1 Project Information 3](#_2et92p0)

[1.2 Project Team 3](#_l17hzdfbd3qv)

[Team Members 3](#)

[2. Abstract 3](#_1t3h5sf)

[3. Introduction 4](#_rcymh9fo6hx)

[III. Methodology 5](#_4i7ojhp)

[1. Dataset 5](#_oavezm99lfww)

[1.1 Overview 5](#_c42u2hp83dxs)

[1.2 Preprocessing 6](#_7rak813mtjlx)

[1.3 Data augmentation 7](#_x69lbfdqjhar)

[1.4 Feature Extraction 7](#_qwml8v297sv5)

[2. Model architecture 9](#_qfmvy52uaqa3)

[2.1 Overview 9](#_2n6ck9dtjy5y)

[2.2 Conv3D blocks 9](#_bjgyxdqql4xy)

[2.3 Adaptive pooling & fully-connected layers 13](#_s2ck6gfim76c)

[2.4 Hyperparameter optimization 13](#_81h5a2ntghg7)

[IV. Experiments 14](#_2atg90b2x7ql)

[1. Training process 14](#_wgiycovr6qzo)

[1.1 Data preparation 14](#_bly65kfd91s1)

[1.2 Train model 15](#_66h47tf7u6fu)

[2. Evaluation 17](#_3whwml4)

[V. System integration 20](#_aggbdw117ypk)

[1. Model integration 20](#_uijeks43z7wo)

[1.1 Overview 20](#_w75bqyizt9i)

[1.2 Methodology 20](#_290vrpxy0ty9)

[2. Simple Chatbot Demo 21](#_e6me7v51ms1t)

[2.1 Overview 21](#_k9s66vju2vxo)

[2.2 Methodology 21](#_lnyryo9uzmyx)

[VI. Discussion 22](#_u41778e25br)

[1. Overall effectiveness of the model 22](#_tzh7317awy73)

[2. Contribution of feature extraction techniques 22](#_px7omblaqcyl)

[3. Advantages of 3D CNN architecture 23](#_sqfnho5ayw42)

[4. Comparison between the initial model and the enhanced model 23](#_kb3apf2dk87d)

[5. Restrictions 26](#_4ft3uya8yq7j)

[6. Practical implementation and scalability 26](#_200nfbqvhebb)

[7. Development directions 26](#_dd89qwgys82y)

[VII. Conclusion 27](#_oxpdrkvq2ll)

[VII. Management 28](#_147n2zr)

[1. Project Schedule 28](#_hxmdptmaig73)

[REFERENCES 28](#_1hmsyys)

# **I. Introduction**

## **1. Overview**

### **1.1 Project Information**

* Project name: Speech Emotion Recognition Project
* Group name: Group 03

### **1.2 Project Team**

#### ***Team Members***

| **Full Name** | **Email** | **Student ID** | **Role** |
| --- | --- | --- | --- |
| Đặng Nguyễn Minh Duy | dangnguyenminhduy11b08@gmail.com | SE194216 | Leader |
| Nguyễn Kế Thịnh | ThinhNKSS181150@fpt.edu.vn | SS181150 | Member |
| Nguyễn Huỳnh Nguyên Phú | phunguyenvscode.py505205@gmail.com | SE196595 | Member |
| Bùi Tấn Phát | btfat3103@gmail.com | SE194343 | Member |
| Nguyễn Lâm Anh | nguyenla110505@gmail.com | SE194272 | Member |
| Trần Thoại Các | thoaicac2005@gmail.com | SE194405 | Member |
| Lê Quang Huy | leequanghuy2004@gmail.com | SE196862 | Member |

*Table 1. Team member contact information*

## **2. Abstract**

Speech Emotion Recognition—SER is one of the pivotal research directions in the field of artificial intelligence, with many potential applications in mental health care, virtual assistants, and human-machine communication [[6]](#l29yl8nn5zq9). In this study, we investigate a method using a 3D Convolutional Neural Network (3D CNN) model [[1]](#athshwwdnwns) to classify emotions in speech, based on the RAVDESS dataset comprising 1,440 audio files representing eight different emotions. The input data is processed and converted into image-like feature sequences using MFCC and Mel-Spectrogram feature extraction techniques. The 3D CNN model is designed to leverage both spatial and temporal information. The study aims to develop an effective emotion recognition system, contributing to the advancement of emotion-based interactive applications in artificial intelligence and intelligent systems.

## **3. Introduction**

Extracting and classifying emotions from speech requires not only language processing capabilities but also the ability to capture emotional acoustic features such as pitch, rhythm, vibrato, and duration [[2]](#drvpi762fbgc) [[7]](#14iosiarhmq0). This opens up many practical applications in fields such as virtual assistants, mental health care, social robots, user behavior analysis, and emotional education.

However, emotion recognition from audio signals remains a challenging problem due to the complexity and inconsistency of the data. An emotion can be expressed in various ways depending on the individual, language, region, or specific situation. Furthermore, emotions like “sad,” “calm,” or “neutral” have similar expressions, leading to confusion during classification [[8]](#pp7p3glot65v) [[9]](#wuvupvn87gsj). Meanwhile, traditional machine learning models often rely on manual features and lack the ability to learn abstractly from large datasets, resulting in limited effectiveness in real-world environments [[10]](#w0u4y1bjy4to).

In recent times, deep learning models, particularly convolutional neural networks (CNNs), have demonstrated superior performance in directly learning features from audio spectra such as MFCC or Mel-spectrograms [[3]](#mcn7qsgl7jvn) [[11]](#th5ckanuvrkc). However, most studies have been limited to using two-dimensional CNNs (2D CNNs), which only extract spatial features at each time point, without leveraging the temporal continuity—a crucial factor in speech signals.

In this study, we built a baseline model using a three-dimensional convolutional neural network (3D CNN) to extract spatial and temporal features from audio data simultaneously. The input signals are normalized in length, converted into MFCC feature sequences and Mel-spectrograms, and then reorganized into a 3D tensor format for training the 3D CNN model. [[4]](#xmasgrndu4gi)

The objective of this study is to build an efficient processing and training pipeline, expand the scope of deep learning applications in the field of affective computing, and lay the foundation for more emotionally intelligent human-machine interaction systems [[12]](#11cu247p7cna).

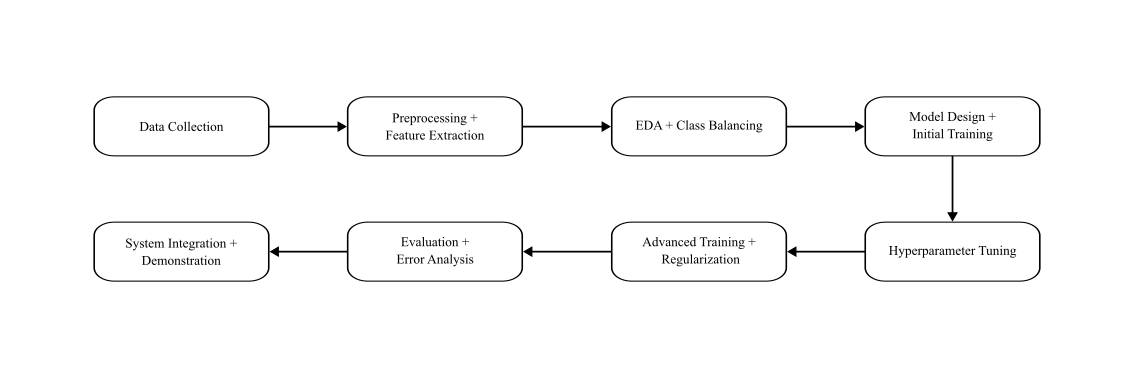
**II. Related Works**

Speech Emotion Recognition (SER) is all about getting emotional info from voice signals for stuff like virtual assistants, mental health care, and intelligent human-machine communication. Traditional approaches typically rely on manual feature extraction combined with classification models such as SVM or Random Forest. However, these methods are limited in their generalization ability and effectiveness on real-world data with high variability. The emergence of deep learning, particularly Convolutional Neural Network (CNN) architectures, has driven a new wave in SER by enabling models to learn directly from the audio spectrum without complex preprocessing. Although 2D CNNs have achieved many positive results, most studies have focused on exploiting spatial features at each time point, failing to take advantage of the continuous relationship over time.

To address this weakness, recent studies have proposed using three-dimensional convolutional networks to simultaneously learn features along three axes: frequency, time, and signal type. Combining multiple feature types and applying data augmentation techniques, such as adding noise, has been proven effective in improving model accuracy and generalization ability. Based on these directions, the current study implements a 3D CNN model optimized using Optuna, while combining MFCC and Mel-spectrogram into a 5-dimensional input tensor. Additionally, the system is practically integrated into a real-time response chatbot via the Flask API, contributing a significant step forward in bringing SER from the laboratory to practical applications—an area that most previous studies have not explored in depth.

# 

# **III. Methodology**

**

*Figure 1. Project pipeline*

## **1. Dataset**

### **1.1 Overview**

The dataset used in the study consists of 1,440 independent audio files, with 60 trials for each actor. There are a total of 24 actors, including 12 female and 12 male actors. The emotions expressed include calm, happy, sad, angry, fearful, surprised, and disgusted.

* Total number of samples: 1,440 .wav files
* Total number of emotion categories: 8 types
* Language: English (neutral content sentences but with varying emotional expressions)
* Source: [RAVDESS Emotional speech audio](https://www.kaggle.com/datasets/uwrfkaggler/ravdess-emotional-speech-audio) [[13]](#xdibeit9th76)

The emotions expressed include:

| **Number** | **Emotion** |
| --- | --- |
| 01 | neutral |
| 02 | calm |
| 03 | happy |
| 04 | sad |
| 05 | angry |
| 06 | fearful |
| 07 | disgust |
| 08 | surprised |

*Table 2. List of labeled emotion categories*

Each audio file is labeled with a 7-part numerical identifier encoding information about intonation, content, speaker, gender, and most importantly, the emotion code. In this study, we extracted the emotion code to serve as labels for the classification task.

### **1.2 Preprocessing**

During the preprocessing stage, the main objective is to ensure that all input audio files have a consistent format and appropriate length for feeding into a 3D convolutional neural network (3D CNN). Standardizing and preparing consistent audio data is a key factor in helping the model learn more stable and generalizable features from speech.

1. ***Load and normalize audio signals:***

First, audio files are loaded using the load\_audio method, which uses the librosa.load() function to extract the waveform and sampling rate from the .wav file.

Then, the waveform data is normalized in amplitude using librosa.util.normalize, which brings the audio amplitude values to the range from -1 to 1. This reduces the impact of volume levels on the learning model and improves stability during training.

1. ***Center Cropping***

Since neural networks require fixed-size inputs, it is necessary to ensure that all audio segments have the same length. To address this issue, we implement the center\_crop function, which operates in two ways:

* If the audio segment is shorter than the standard length, the function adds padding with a value of 0 at both ends, ensuring symmetry and preserving the main content in the middle.
* If the audio segment is longer, the function crops the middle segment to extract the central part of the signal, which typically contains emotionally rich information and has higher stability.

Cutting and padding in this way helps the model receive consistent input in terms of duration, thereby learning important audio features more effectively.

### **1.3 Data augmentation**

In this phase, we wanted our model's robustness and generalization to be improved, so we decided to use two approaches to augment our datasets [[14]](#l8rtvlathnqr). In the first approach, we implemented pitch\_shift\_audio() method to randomly select a pitch shift between -max\_steps and max\_steps semitones. Then, we apply the shift by using librosa.effects.pitch\_shift. This approach was used to increase dataset variety to improve model robustness [[14]](#l8rtvlathnqr) [[15]](#t4z9f71xt4jd).

In the second approach, we implemented the add\_noise() method in order to generate a random noise level between min\_level = 0.005 and max\_level = 0.007. Then, we created Gaussian noise with the same length as the input audio and added the scaled noise to the audio, and normalized the result to prevent clipping. This approach was applied to simulate real-world audio imperfections to enhance model generalization.

### **1.4 Feature Extraction**

During the feature extraction stage, the goal is to transform the audio signal into meaningful time-frequency features that are suitable for input into the 3D CNN model. The two main types of features used are Mel-Frequency Cepstral Coefficients (MFCC) and Mel-spectrogram:

* MFCC: simulates how humans hear sound, especially the ability to separate meaningful frequency bands in speech recognition.
* Mel-spectrogram: reflects intonation, energy, and frequency changes, which are important factors in emotional expression.

***Standardize input feature sizes***

Before feature extraction, it is necessary to ensure that all output feature matrices have the same number of time frames so that the model can learn consistent representations. The \_pad\_or\_truncate() function is designed to perform this task:

* If the time dimension T of the characteristic matrix is less than max\_len, the function will add columns of zeros to reach the standard length.
* If T is greater than max\_len, the function will truncate the excess to retain the specified number of time frames.

This process standardizes the input size, making the data compatible with the requirements of multi-dimensional convolutional neural networks.

***Standardize characteristic intensity***

To reduce variance between samples and increase stability during training, the standardize\_clip() function is used to apply L2 normalization on the time axis of the feature matrix, via sklearn.preprocessing.normalize. This ensures that each audio segment after feature extraction has the same intensity value distribution.

***MFCC and Mel-spectrogram feature extraction***

* MFCC: Calculated using librosa.feature.mfcc with 40 coefficients. After extraction, the MFCC features are passed through two normalization steps: \_pad\_or\_truncate() and standardize\_clip().
* Mel-spectrogram: Calculated using librosa.feature.melspectrogram with 40 frequency bands (Mel bins), then converted to decibels using librosa.power\_to\_db. Similar to MFCC, Mel features are also processed using \_pad\_or\_truncate() and standardize\_clip().

***Combine features into input tensors for 3D CNNs***

The two features MFCC and Mel-spectrogram are stacked into a tensor of shape (2, 40, T):

* 2: the number of feature types.
* 40: the number of feature coefficients along the frequency axis.
* T: the number of time frames after normalization.

The result of this step is a 3-dimensional feature tensor representing each audio segment, ready to be fed into a deep learning model [[16]](#s3zw7uhnsegj).

## **2. Model architecture**

### **2.1 Overview**

Unlike 2D CNNs, which only extract spatial information, 3D CNN models are capable of learning both spatial and temporal information simultaneously:

* Each log-Mel image represents spectral information at a specific point in time.
* The third axis is timing — the evolution of the spectrum over segments of sound.

This is particularly important in sound analysis, where emotions are expressed through temporal evolution.

Our model is a 3D convolutional neural network (3D CNN) specifically designed for emotion recognition from speech, consisting of five consecutive Conv3D layers arranged in a spatio-temporal depth, along with GroupNorm, ReLU, Dropout3D, and MaxPool3D layers, followed by global average pooling and two fully-connected layers to make the final decision. This architecture aims to learn rich features across time, frequency, and data channels, effectively supporting the task of multi-class emotion classification.

A similar model has been demonstrated to be effective in recent studies, such as a SER system based on 3D CNN, which extracts important frames of the spectrogram as 3D tensors, achieving superior accuracy on various datasets like SAVEE and RML.

### **2.2 Conv3D blocks**

Input: Each input data sample has a 5-dimensional tensor format with dimensions: (batch\_size,1,2,130,40), where:

* batch\_size: the number of samples in each training batch.
* 1: audio channel - the data is mono audio, so there is 1 channel.
* 2: number of feature types - including MFCC and Mel-spectrogram, treated as 2 parallel “spectral channels”.
* 130: number of time frames.
* 40: number of feature coefficients (frequency coefficients) for each frame, corresponding to the number of dimensions after extraction.

By combining MFCC and Mel-spectrogram, we can take advantage of two complementary acoustic features, and the model can learn both horizontal (time) and vertical (frequency) information, thereby improving the model's learning ability.

***Each Conv3D block performs:***

1. Conv3d(in\_channels, out\_channels, kernel\_size=(3, 3, 3), padding=1)

* Function: Extract local features from spatially and temporally structured data, learn the relationship between acoustic features (depth), temporal progression, and frequency spectrum.
* Mechanism: A w kernel traverses the entire input tensor x across three dimensions: depth (features), time, and frequency, learning emotional changes across all three dimensions.

At each position (i,j,k), the kernel will:

1. Take a sub-block of the same size as the kernel from the input,
2. Multiply each element of the sub-block by the kernel weight,
3. Sum up to form an output value.

The result is a single number, representing the level of activation in that data region.

The iterative process forms the output feature map.

1. GroupNorm(num\_groups=4, channels=out\_ch)

* Function: Standardizing the output after each convolution layer plays an important role in stabilizing the training process, improving convergence speed, and reducing overfitting.
* Mechanism: Perform normalization for each channel group in per input instance, rather than in batches as in BatchNorm. Specifically:

1. Given an input tensor after each convolution layer of the form x∈RN×C×D×H×W , GroupNorm divides the total number of channels C into G groups, each group having C/G channels.
2. For each group, calculate the mean μ and variance σ2 across the entire space (D, H, W) and channel in the m-element group:
3. Standardize each element xi in the group:

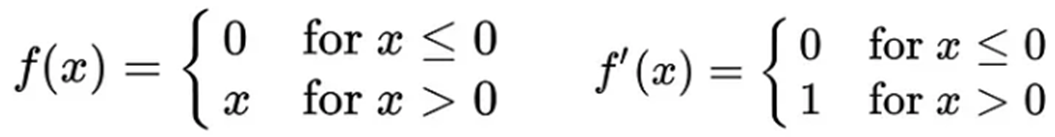
Where γ and β are parameters learned to scale and shift the output, and ϵ is a small constant to avoid division by zero.

In this study, processing 5-dimensional input tensors (batch, channel, depth, time, freq) with a limited batch size, GroupNorm is considered the optimal solution: as stable as BatchNorm but not limited by batch size.

* Comparison with BatchNorm:
  1. GroupNorm performs better and is more stable in cases of small batches (batch < 32) or non-uniform batches.
  2. Suitable for 3D audio data, enabling normalization by feature group without losing the time–frequency correlation.

1. ReLU

* Function: Activation functions play an essential role in deep neural networks by introducing nonlinearity, thereby allowing the model to learn complex and nonlinear relationships between inputs and outputs.
* Mechanism: For each input element x, the ReLU function keeps the value unchanged if x > 0 and returns 0 if x ≤ 0. This is a simple but effective piecewise nonlinear function that does not change the gradient when x > 0 and eliminates the gradient when x ≤ 0.



1. Dropout 3d(p=0.1)

* Function: A regularization technique that minimizes overfitting by preventing the model from becoming overly dependent on a specific feature subset, encouraging the network to learn more diverse and stable representations. This is particularly important in 3D CNNs, where each channel represents highly generalized information that directly influences the output.
* Mechanism: Operates on a channel-wise basis: for each sample in the minibatch, the entire output channel is randomly disabled with probability p.

In this study, with a probability of p=0.1, the model temporarily ignores 10% of the feature maps from Conv3D in each batch.

The value p=0.1 is chosen to maintain a balance between regularization and information retention. In convolutional models, using a high probability value can lead to the loss of important information, negatively impacting training performance. Therefore, the value 0.1 is used as a mild but effective reduction to prevent overfitting without disrupting the information flow in the model.

1. MaxPool3d(kernel\_size=(1,2,2)):

* Function: Retain the strongest features, reduce noise, reduce spatial size, save resources, and reduce computational costs.
* Mechanism: Operates by sliding a kernel over the input and retaining the maximum value in each block. With a kernel size (depth, height, width), this operation can be described by the formula: 

In which:

1. x: input tensor
2. y: output tensor
3. i, j, k: corresponding output space positions (depth, height, width),
4. Kd, Kh, Kw: kernel sizes in the depth, height, and width dimensions (typically corresponding to features/time/frequency in audio)
5. u,v,w: indices within the kernel region
6. max(⋅): the operation of selecting the maximum value within a small block of the input.

In this study, with a kernel size of (1,2,2), the pooling process only occurs on the last two dimensions of the tensor—corresponding to the 2D plane (time × frequency)—while preserving the integrity of the acoustic feature dimension (MFCC, Mel-spectrogram). Specifically:

* Depth dimension = 1: retained to preserve the semantic structure between acoustic feature channels (MFCC, Mel-spectrogram). Pooling along this dimension may lead to information mixing and loss.
* Time × freq dimension = 2 × 2: reduces spatial dimensions in local regions, helping to remove noise and highlight important emotional features while reducing computational cost.

The choice of (1, 2, 2) balances the preservation of information separation between features and the ability to generalize spatial representation in emotional expression over time. Larger kernel configurations may cause loss of detail; smaller ones do not provide significant downsampling benefits.

### **2.3 Adaptive pooling & fully-connected layers**

* After the Conv and Pool layers, the data passes through AdaptiveAvgPool3d((1,1,1)) to transform the output into a fixed size, making the model flexible with varying input lengths due to augmentation.
* Next is a Linear layer (flatten\_dim → fc\_dim=128), ReLU, and Dropout (p≈0.13), followed by a Linear layer (fc\_dim → num\_classes=8) for classification purposes.

### **2.4 Hyperparameter optimization**

During model training, selecting appropriate hyperparameters plays a crucial role in improving the performance and generalization of deep learning models. However, manually determining the optimal hyperparameter combination is often time-consuming and labor-intensive, without guaranteeing the most effective configuration.

To address this issue, we utilize Optuna—an efficient and automated hyperparameter optimization framework. Specifically, Optuna performs a search within the predefined hyperparameter space through 50 trials. Each trial corresponds to a specific model configuration, evaluated based on accuracy on the validation set.

The hyperparameter search space is defined as follows:

* Number of filters in Conv3D layers:

conv1 ∈ {16, 32, 64}

conv2 ∈ {32, 64, 128}

conv3 ∈ {64, 128}

conv4 ∈ {128, 256}

conv5 ∈ {128, 256, 384}

* Dropout rate: from 0.1 to 0.5 (continuous)
* Number of units in the fully-connected layer: {64, 128, 256}
* Learning rate: [1e-4, 1e-2]

After the optimization process, Optuna selected a set of hyperparameters that significantly improved the model's performance on the test set. The best model obtained was saved and used in the final evaluation.

# **IV. Experiments**

## **1. Training process**

### **1.1 Data preparation**

In the first step of the training process, we conducted a check on the distribution of samples among emotion classes to assess the degree of data imbalance—a common problem in speech emotion classification tasks. The statistical results are shown in the table below:

| **Class** | **Samples** | **Class** | **Samples** |
| --- | --- | --- | --- |
| 1 | 96 | 5 | 192 |
| 2 | 192 | 6 | 192 |
| 3 | 192 | 7 | 192 |
| 4 | 192 | 8 | 192 |
| 5 | 192 | 8 | 192 |
| 6 | 192 | 8 | 192 |
| 7 | 192 | 8 | 192 |
| 8 | 192 | 8 | 192 |

*Table 3. Class distribution*

It is easy to see that class 1 is the only class with a significantly lower number of samples than the other classes, only 50%. If left unaddressed, this discrepancy could cause the model to be biased and less effective in correctly identifying samples belonging to this class [[17]](#z54u4hig77d9).

To fix this, we applied data augmentation techniques to increase the number of samples in class 1 to match the other classes. Specifically, the pitch-shifting method was chosen to create new audio copies from the original samples by changing the pitch while keeping the overall content and emotion the same.

The use of pitch-shifting not only balances the data across classes but also increases the acoustic diversity in the training set, thereby contributing to improving the model's generalization ability [[18]](#96da0gd7f0j).

### **1.2 Train model**

In this phase, we trained a three-dimensional convolutional neural network (3D CNN) model to classify emotions from voice signals converted into spectral features (MFCC and Mel-spectrogram). The training process was rigorously designed, from hyperparameter configuration to model performance evaluation strategies.

The primary objective is to optimize emotion classification performance by constructing a 3D CNN architecture tailored to multi-dimensional input features, combining MFCC and Mel-spectrogram. The hyperparameter selection process is automated using the Optuna library. Simultaneously, techniques such as Early Stopping, Dropout, and Adaptive Pooling are applied to minimize overfitting and enhance the model's generalization capabilities.

***Hyperparameter***

The trial with the highest performance is Trial 13, achieving an accuracy of 0.8697, with the following hyperparameter configuration:

* 'conv1': 16,
* 'conv2': 128,
* 'conv3': 64,
* 'conv4': 256,
* 'conv5': 128,
* 'dropout': 0.2142,
* 'fc\_dim': 128,
* 'lr': 0.000137

***Training settings***

The training process is implemented on a GPU (CUDA) if available, ensuring high processing speed. If a GPU is not available, the system will automatically switch to using the CPU.

Loss function: CrossEntropyLoss, suitable for multi-class classification problems [[19]](#96da0gd7f0j).

Optimization algorithm: Adam with a learning rate set to 0.000137; this value was selected through hyperparameter optimization using Optuna [[5]](#kix.utjyb2z3z1gk) [[20]](#tbqqh0sl369k).

Number of training epochs: maximum 100 epochs, but will be adjusted flexibly via the early stopping mechanism.

Batch size: uses the default from DataLoader, set to 32 samples per training session.

Early stopping strategy: if the accuracy on the validation set does not improve over 5 consecutive epochs, the training process will stop to reduce the risk of overfitting and save computational resources.

***Training process***

Each training loop is divided into two main phases:

The model is set to train() mode, allowing gradient calculations and weight updates through the backpropagation algorithm. During this phase, the model learns to adjust its parameters to minimize the loss function.

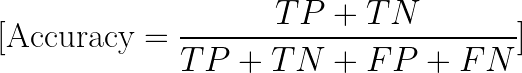
The model is switched to eval() mode, where calculations are performed without updating the weights. The goal is to evaluate the model's performance on unseen data, using metrics such as accuracy and loss.

After each epoch, if the accuracy on the validation set improves, the model weights at that point are saved. If there is no improvement after 5 consecutive epochs, the training process is stopped early to ensure overall efficiency and prevent overfitting.

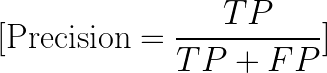
## **2. Evaluation**

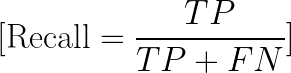
Following the training process, the 3D CNN model was evaluated on the test set to measure its generalization ability and emotion classification performance. The evaluation was conducted using standard metrics for multi-class classification problems.

The predicted labels are compared with the actual labels to calculate three groups of standard evaluation indices for the multi-class classification problem:

Accuracy: measures the ratio of correctly predicted samples to the total number of samples. This is a general indicator that reflects the overall recognition ability of the model, but it can be affected when the dataset is unbalanced.

Precision: reflects the ability to predict correctly when the model selects a specific class. High precision means less confusion with other classes.



Recall: evaluates the ability to correctly detect all samples belonging to a class. High recall indicates that the model misses few samples.

Confusion Matrix: a visual tool that helps analyze confusion between emotion classes. Each row represents the actual label, and each column represents the predicted label. From this, it is possible to identify emotion pairs that are easily confused.

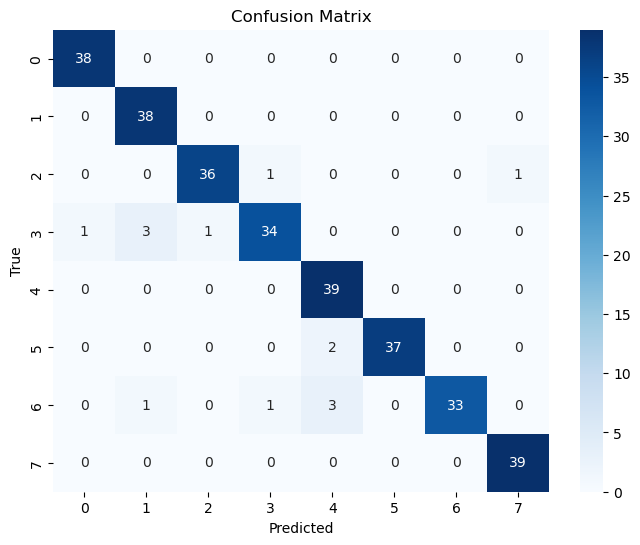
***Evaluation results***

After applying the best model, we obtained the following results:

Test Accuracy: 0.9545

| **Emotion** | **Precision** | **Recall** | **F1-score** | **Support** |
| --- | --- | --- | --- | --- |
| Neutral | 0.9744 | 1.0000 | 0.9870 | 38 |
| Calm | 0.9048 | 1.0000 | 0.9500 | 38 |
| Happy | 0.9730 | 0.9474 | 0.9600 | 38 |
| Sad | 0.9444 | 0.8718 | 0.9067 | 39 |
| Angry | 0.8864 | 1.0000 | 0.9398 | 39 |
| Fearful | 1.0000 | 0.9487 | 0.9737 | 39 |
| Disgust | 1.0000 | 0.8684 | 0.9296 | 38 |
| Surprised | 0.9750 | 1.0000 | 0.9873 | 39 |

*Table 4. Class distribution*



*Figure 2. Confusion Matrix*

***General assessment***

The evaluation results show that the 3D CNN model achieves good classification performance on the test set, with an overall accuracy of up to 95.45%. This indicates that the model has good learning and generalization capabilities from the training data to new data.

Emotions with clear expressions such as neutral, surprised, fearful, and happy achieve F1-scores > 0.96, indicating that the model has learned distinct intonation features that clearly distinguish between these classes.

***Classes that are more easily confused***

Sad (F1 = 0.9067) and Disgust (F1 = 0.9296) have lower F1 values than the other classes. This reflects some overlap in prosody (intonation, rhythm, intensity) between negative emotions. The confusion matrix shows that some cases of disgust were mistaken for fearful or sad, which has been noted.

**V. System integration**

# **1. Model integration**

### **1.1 Overview**

Training and creating a neural network is not enough; in reality, the neural network must be integrated into a system or application to have commercial value [[21]](#a4jg47ddgv2g). Therefore, we decided to integrate this neural network into an application with the goal of creating a chatbot capable of predicting emotions based on human voice.

We integrated the neural network into the system with the following idea: the neural network will take the user's recorded voice at the time of using the chatbot as input, and after processing, it will output the corresponding emotion based on their voice. To enable communication between the application and the neural network, we used the Flask module to create an API, utilizing it as an intermediary tool for interaction between the application and the neural network.

Thus, the results predicted by the neural network will be displayed on the user interface layer with pre-generated response segments based on the labels.

### **1.2 Methodology**

The main method we used here is to create a local server API based on the Flask module I mentioned earlier. The directory file of our local API will include several subdirectories such as models, utils, and unit\_test [[22]](#eovgjc9zlyc7). The models directory is used to store pre-trained models for the purpose of reuse in predictions. The unit\_test directory is where we conduct testing for the APIs we have created.

Regarding the utils directory, for the architecture module, this is the module containing the architecture of the neural network that our team previously created and is reused for prediction. The model module, on the other hand, is designed to perform predictions as well as process user input. Additionally, whenever we receive a user's recorded file via the API, we temporarily store this file on the server and use it for prediction purposes. After the prediction is complete, we use the file\_manager module to delete these files to avoid duplication.

Finally, the API communication methods will be written in the main program in the app.py file. The user's voice, after being recorded at the interface layer, will be sent to the local API using the POST method. At this point, the server will receive the file sent by the user and process it to return the prediction result.

## **2. Simple Chatbot Demo**

### **2.1 Overview**

The simple chatbot application was developed based on the idea of creating a conversation between the user and the bot. However, the key difference is that this bot is trained to perform a specific task, which is to recognize emotions from the user's voice [[23]](#1dm8jrqkj5w). The application was built and developed using the C# .NET language and WPF technology. As it was created for a course assignment demo, its features and interface are still quite basic. The application includes three main features: a button to allow users to view usage instructions, a button for users to record and send their voice, and a final button to clear the current conversation history.

### **2.2 Methodology**

The primary language used in the application is C#, with several libraries integrated into the project, including NAudio, Newtonsoft, and RestSharp. The NAudio library is used for general audio file processing in C#. We utilized this library to support features such as recording the user's voice, reading, writing, and saving .wav audio files.

The Newtonsoft library is used to process JSON strings returned from APIs and convert them into corresponding objects in C#. This library was employed to handle the JSON strings returned when posting the user's voice to the server.

The RestSharp library is used to perform API communication methods such as GET, POST, and others. In our project, this library was used to send the user's audio file to the server and receive the results for processing.

**VI. Discussion**

## **1. Overall effectiveness of the model**

The 3D Convolutional Neural Network (3D CNN) model proposed has shown impressive performance in the task of classifying emotions from speech. After training and optimization using the Optuna algorithm with 50 trials, the model was trained on the training set of the RAVDESS dataset and evaluated on an independent test set.

The results on the test set show that the model achieves:

* Overall accuracy: 95.45%
* Macro F1-score: 0.9543
* Weighted F1-score: 0.9542

The macro and weighted F1-score metrics are nearly equivalent, reflecting that the model performs uniformly across all emotion classes without bias toward more common classes. This demonstrates the effective feature learning and good generalization capabilities of the 3D CNN architecture on spectrogram sequences representing speech signals.

| **Emotion** | **Precision** | **Recall** | **F1-score** | **Support** |
| --- | --- | --- | --- | --- |
| Accuracy |  |  | 0.9594 | 308 |
| Macro Average | 0.9572 | 0.9545 | 0.9543 | 308 |
| Weighted Average | 0.9572 | 0.9545 | 0.9542 | 308 |

*Table 5. Class distribution*

## **2. Contribution of feature extraction techniques**

Sound feature extraction plays a key role in the effectiveness of speech emotion recognition models. The simultaneous combination of two popular feature methods—MFCC (Mel-Frequency Cepstral Coefficients) and Mel-spectrogram—has resulted in an information-rich representation that supports the model in learning both the content and tone of speech.

Both features are normalized in length and stacked along the channel axis to form an input tensor of size (2, 130, 40), corresponding to three axes: feature modality, temporal, and frequency. To align with the 3D CNN architecture, this tensor is expanded into a 5-dimensional tensor with shape (batch\_size, 1, 2, 130, 40), where the depth of 2 represents the combination of MFCC and Mel-spectrogram.

The integration of the channel axis not only enables the model to process two independent representation spaces simultaneously—MFCC (focused on speech content) and Mel-spectrogram (emphasizing intonation and emotional energy), but also facilitates the network's learning of intermodal correlations between feature modalities. This is a significant advantage over traditional 2D CNN models, which only process single-channel 2D tensors and cannot leverage the multi-dimensional combination of signal representations.

The addition of a third dimension in the input allows the 3D CNN model to extract emotional features more comprehensively, thereby improving classification performance, especially in cases where emotions are easily confused, such as between fearful and sad, or calm and neutral. The results from the confusion matrix clearly reflect this improvement, showing that the model is better able to distinguish between similar emotion classes thanks to its multi-dimensional feature representation architecture.

## **3. Advantages of 3D CNN architecture**

Compared with traditional architecture:

| **Method** | **Limitation** |
| --- | --- |
| 2D CNN | Learns only spatial features from individual frames and cannot capture the temporal relationships between consecutive audio frames, leading to a loss of information about the emotional progression over time. |
| SVM / Random Forest | Fully dependent on handcrafted features. Cannot learn deep representations directly from raw data, resulting in lower performance on complex audio signals. |

*Table 6. Traditional architecture*

Advantages of the 3D convolutional neural network (3D CNN) model:

Learn continuous emotional context information over time

Extract features directly from spectral data

Use GroupNorm instead of BatchNorm to make the model more stable with small batch sizes

Dropout3D, EarlyStopping, and Adaptive Pooling help increase generalization and avoid overfitting.

## **4. Comparison between the initial model and the enhanced model**

***a. Initial model***

The initial 3D CNN model demonstrated rapid learning capabilities in the early stages of training, but quickly fell into overfitting and ceased to improve after early convergence.

Convergence process:

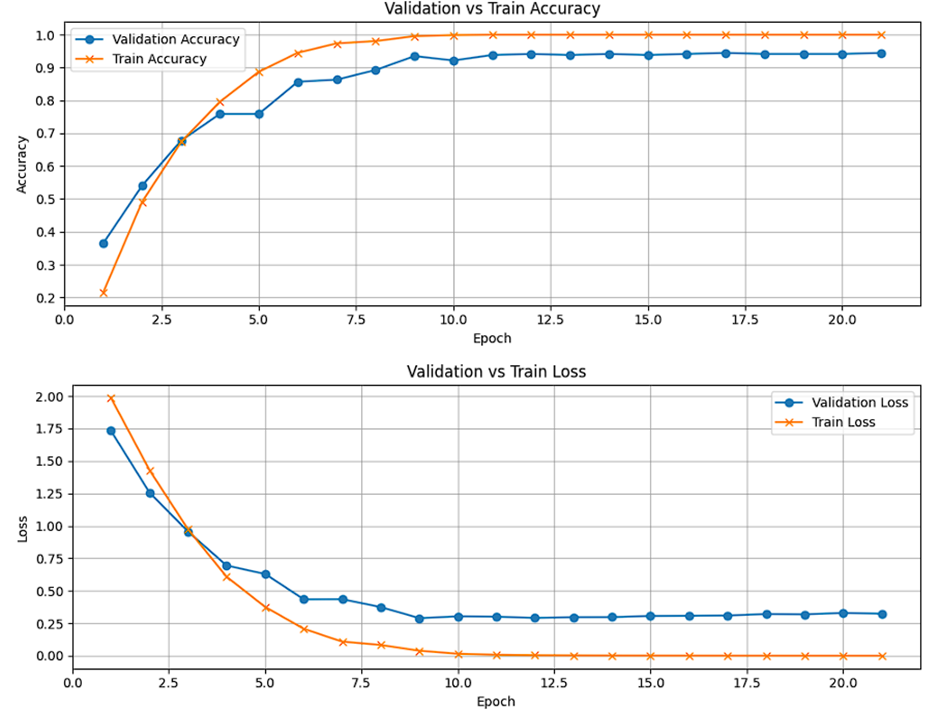
* Epoch 1–6: The model reduces loss significantly and improves accuracy on the validation set, increasing from 36% to 85% after only the first 6 epochs.
* Epoch 9: The model achieves 93.5% validation accuracy with the lowest val loss of 0.2899. This is considered the convergence point of the model.

Post-convergence (epochs 10–21):

* Although training accuracy reaches 100% from epoch 10, validation metrics do not show significant improvement:
* Validation acc only increases from 93.5% to 94.5%.
* Validation loss fluctuates slightly around 0.29–0.33, without further decrease.
* Train loss decreases very slightly, reaching nearly 0.0008 at epoch 21, indicating that the model has almost memorized the entire training set.

Clear signs of overfitting:

The large gap between train accuracy and validation accuracy, as well as between train loss and validation loss, clearly reflects overfitting, meaning that the model has learned too closely to the training data and is no longer able to generalize well to new data.



*Figure 3. Training Progress of Initial 3D CNN Model*

***b. Enhanced model***

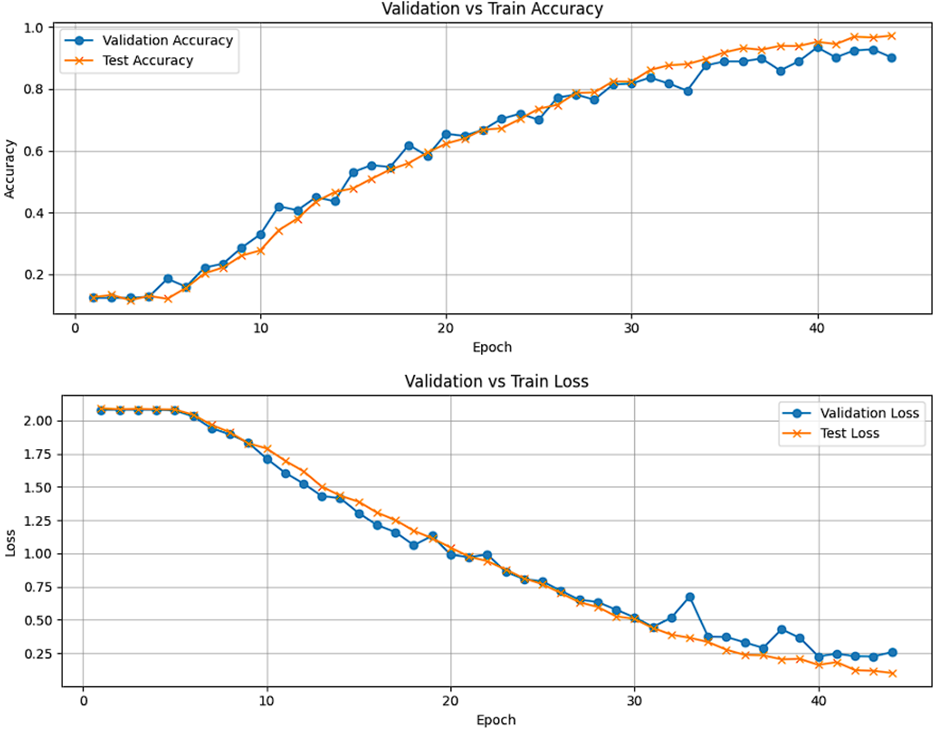
In contrast to the original model, which learned very quickly but overfitted early, the improved model was optimized with regularization strategies and more appropriate hyperparameter configurations, resulting in more stable learning and better generalization.

Learning and convergence process:

* Epoch 1–10: The model learns slowly, with validation accuracy increasing from 12.3% to 32.9%, demonstrating a cautious start without the sudden increase seen in the initial model.
* Epoch 20: Validation accuracy reaches 65.5%, validation loss = 0.9900
* Epoch 27–31: The model begins to improve more rapidly, with validation accuracy exceeding 80%
* Epoch 40: The model converges with validation accuracy = 93.5%, validation loss = 0.2248

After convergence (epochs 40–44):

* Train accuracy fluctuated between 95% and 97%, not reaching 100%.
* Validation accuracy remained stable between 92% and 93%.
* Validation loss remains at 0.22–0.25, without a significant increase
* Train loss at epoch 44 is 0.0999, not yet reaching 0



*Figure 4. Training Progress of Enhanced 3D CNN Model*

The distance between the indices is reasonable, reflecting that the model does not memorize the training data but maintains good generalization ability. This is a positive sign that the model has learned effectively from the data without overfitting.

| **Criteria** | **Initial Model** | **Final Model** |
| --- | --- | --- |
| Convergence Epoch | 9 | 40 |
| Final Train Accuracy | 100% | 97% |
| Final Val Accuracy | 94.5% | 93.5% |
| Train Loss | ~0.0008 | ~0.0999 |
| Val Loss | ~0.29-0.33 | ~0.22-0.25 |
| Overfitting status | Yes | Trivial |

*Table 7. Comparative Summary of Initial and Final 3D CNN Models*

Although the improved model requires more epochs to converge, the learning process is more stable and efficient. Notably, there is no obvious overfitting phenomenon as in the original model. This model retains good generalization capabilities and is more suitable for practical implementation.

## **5. Restrictions**

Some classes are easily confused: Sad, Disgust, and Fearful—these classes share the same low-pitched phonetic characteristics and are relatively similar in rhythm.

Although the data for the Neutral class has been augmented, there is still potential bias due to the initial imbalance in distribution.

The RAVDESS dataset is limited in diversity: it only contains neutral sentences of a fixed length.

## **6. Practical implementation and scalability**

The model was successfully implemented via Flask API and a chatbot utilizing C# (.NET + WPF), demonstrating its practical application potential.

Near real-time response speed, scalable for virtual assistants, mental health support, and emotional education.

## **7. Development directions**

***a. Expand and diversify training data***

To continue improving the performance of emotion recognition from speech and expanding the scope of application, the following potential development directions are proposed:

Current models are exclusively trained on the RAVDESS dataset, which has a relatively balanced and standardized structure. However, expanding the dataset by integrating additional emotionally rich datasets such as:

* CREMA-D (Crowd-sourced Emotional Multimodal Actors Dataset),
* TESS (Toronto Emotional Speech Set),
* SAVEE (Surrey Audio-Visual Expressed Emotion)

This will enhance diversity in voice, expression, gender, and style of expression. This will enable the model to generalize better when deployed in real-world environments with non-standardized variations.

***b. Upgrade model architecture***

Currently, the model uses a pure 3D CNN. To further exploit the temporal context, the following approaches can be applied:

* Attention Mechanism: Helps the model focus on important audio segments related to emotions, improving interpretation and classification performance.
* Hybrid 3D CNN + LSTM: Combines the spatial and temporal feature extraction capabilities of 3D CNN with the long-term memory capabilities of LSTM to enable the model to better understand the flow of emotions.
* Transformer Encoder: With its ability to model long-term dependencies and learn representations via self-attention, the transformer encoder has the potential to replace RNN/LSTM for sequence-based emotion analysis tasks.

# 

# **VII. Conclusion**

In this study, we successfully built a speech emotion recognition (SER) pipeline using a three-dimensional convolutional neural network (3D CNN) on the RAVDESS dataset. The system includes audio preprocessing stages, feature extraction (MFCC and Mel-spectrogram), normalization, and conversion to 3D input tensors, combined with an optimized model architecture using Optuna to achieve the highest performance.

Experimental results show that the model achieves high accuracy up to 95.45% on the test set, with precision, recall, and F1-score metrics maintaining stable levels across emotion classes. The simultaneous combination of two acoustic features with the 3D CNN's spatio-temporal learning capability is a key factor in enabling the model to effectively capture emotional variations in speech.

In particular, by comparing the initial model with the improved model, we demonstrated that stable training and overfitting control play an important role in the generalization ability of the system. The final model not only achieved high performance but was also successfully integrated into a demo chatbot application, demonstrating its practical feasibility.

However, the system still has several limitations, primarily due to the nature of the data and the overlap between emotions with similar expressions. This points to future research directions such as: integrating attention mechanisms, adding more datasets, and expanding the ability to recognize transitional emotional states.

Overall, the study has laid the foundation for a reliable, scalable emotion recognition system from speech that can be applied to various fields such as virtual assistants, mental health care, emotional education, and intelligent human-machine communication.

# **VIII. Management**

## **1. Project Schedule**

| **Project phase** | **Objectives** | **Timeframe** |
| --- | --- | --- |
| Prepare Data | Understand about dataset Extract feature | 14/5-21/5 |
| Data Analyzing | EDA Visualizing Class Balancing | 24/5-28/5 |
| Model building | Research Model Build Model | 1/6 - 21/6 |
| Hyperparameter tuning | Fine-tune the model’s hyperparameters  Adjust model Evaluate the model’s results | 23/6-12/7 |
| System demonstration | Build an app to deploy the system and model | 14/7-18/7 |
| Report | Design slide Write report | 14/7 - 19/7 |

*Table 8. Project Schedule*

# **REFERENCES**

**[1 ]** M. R. Falahzadeh, E. Z. Farsa, A. Harimi, A. Ahmadi, and A. Abraham, "3D convolutional neural network for speech emotion recognition with its realization on Intel CPU and NVIDIA GPU," *IEEE Access*, vol. 10, pp. 112460–112471, 2022.

**[2]** M. El Ayadi, M. S. Kamel, and F. Karray, “Survey on speech emotion recognition: Features, classification schemes, and databases,” *Pattern Recognition*, vol. 44, no. 3, pp. 572–587, 2011.

**[3]** V. Sareen and K. R. Seeja, "Speech emotion recognition using Mel spectrogram and convolutional neural networks (CNN)," *Procedia Computer Science*, vol. 258, pp. 3693–3702, 2025.

**[4]** H. Ismail, S. Nahid, M. Z. Hasan, M. P. Hossain, and M. A. Rahaman, "A 3D CNN model with multi-feature fusion for enhancing human emotion recognition from speech," in *Proc. Int. Conf. Communication and Computational Technologies*, Singapore: Springer Nature Singapore, Jan. 2024, pp. 289–300.

**[5]** T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, "Optuna: A Next-generation Hyperparameter Optimization Framework, " *arxiv:1907.10902,* 2025*.*

**[6]** K. Hegde and H. Jayalath, "Emotions in the Loop: A Survey of Affective Computing for Emotional Support," *arXiv preprint arXiv:2505.01542*, 2025.

**[7]** B. W. Schuller and A. Batliner, *Computational Paralinguistics: Emotion, Affect and Personality in Speech and Language Processing*. John Wiley & Sons, 2013.

**[8]** L. Devillers, L. Vidrascu, and L. Lamel, "Challenges in real-life emotion annotation and machine learning based detection," *Neural Networks*, vol. 18, no. 4, pp. 407–422, 2005.

**[9]** Y. Deng, X. Tang, M. Zhang, C. Han, and P. Zhang, "Speech Emotion Recognition Incorporating Relative Difficulty and Labeling Reliability," *Sensors*, vol. 24, no. 13, art. 4111, 2024.

**[10]** K. Han, D. Yu, and I. Tashev, "Speech emotion recognition using deep neural network and extreme learning machine," in *Proc. 15th Annu. Conf. Int. Speech Communication Association*, 2014.

**[11]** A. Alarifi and A. Alowaidi, "Enhancing Speech Emotion Recognition Using Deep Convolutional Neural Networks," in *Proc. Int. Conf. Machine Learning Technologies (ICMLT 2024)*, 2024.

**[12]** T. Dhanasingh, "Speech emotion recognition for human–computer interaction," *Int. J. Speech Technology*, vol. 27, no. 3, pp. 817–830, 2024.

**[13]** S. R. Livingstone and F. A. Russo, "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*, vol. 13, no. 5, art. e0196391, 2018.

**[14]** T. Ko, V. Peddinti, D. Povey, and S. Khudanpur, "Audio augmentation for speech recognition," in *Proc. 16th Annu. Conf. Int. Speech Communication Association*, 2015.

**[15]** J. Han and Y. Wang, "Data Augmentation for Speech Emotion Recognition," in *Proc. 12th Int. Conf. Information Technology and Multimedia (ICIM)*, IEEE, 2020, pp. 1–5.

**[16]** J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Machine Learning Research*, vol. 13, pp. 281–305, Feb. 2012.

**[17]** S. Zhang, D. Wu, D. Zhang, and X. Wei, "Speech Emotion Recognition Based on Selective Interpolation Synthetic Minority Over-Sampling Technique in Small Sample Environment," *Computational Intelligence and Neuroscience*, vol. 2024, art. 5534789, 2024.

**[18]** Y. Zang and R. Wei, "Data augmentation methods for imbalance datasets in acoustic event classification," *Applied Sciences*, vol. 14, no. 11, art. 4452, 2024.

**[19]** T. M. Cover and J. A. Thomas, *Elements of Information Theory*, 2nd ed. John Wiley & Sons, 2006.

**[20]** D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

**[21]** S. Amershi et al., "Guidelines for Human-AI Interaction," in *Proc. 2019 CHI Conf. Human Factors in Computing Systems*, 2019, pp. 1–12.

**[22]** S. Zhang et al., "A Deep Learning Model Deployment Method Based on Flask and RESTful API," in *Proc. 7th Int. Conf. Image, Vision and Computing (ICIVC)*, IEEE, 2022, pp. 1162–1165.

**[23]** G. Lugaresi et al., "Emotional Chatbots: A Review," in *Proc. 5th Int. Conf. Applied and Theoretical Computing and Communication Technology (iCATccT)*, IEEE, 2023, pp. 112–117.