Enhanced Speech Emotion Recognition: Leveraging Non-Linear Deep Learning Models (CNN, DNN, LSTM, Auto Encoders) with Data Augmentation

Rohith Reddy Vangala, Shashank Reddy Kandimalla, Yamini Muthyala and Swati

Department of Applied Data Science, San Jose State University

DATA 270: Data Analytics Processes

Dr. Eduardo Chan

December 10, 2023

Abstract

In the realm of human-computer interaction, accurately recognizing emotions in speech through deep learning presents a notable challenge. Speech emotion recognition (SER) plays a pivotal role in enhancing user experience and facilitating more intuitive and responsive human-computer interactions. Traditional methods, relying on Mel Frequency Cepstral Coefficients (MFCC) and chroma features with sigmoid and softmax activation functions, have typically achieved accuracy levels of around 70-75%. However, prior research in this field was often constrained by focusing on a narrow set of features and traditional activation functions, leading to suboptimal accuracy, and did not utilize data augmentation techniques. This study aims to substantially improve these results by integrating a broader range of features and advanced techniques, including root mean square, mel spectrograms, Zero-Crossing Rate (ZCR), and data augmentation techniques, alongside the standard MFCC and chroma features. Departing from conventional sigmoid and softmax functions, the Rectified Linear Unit (ReLU) activation function is adopted. Addressing these gaps, this research employs various deep learning architectures, including Long Short-Term Memory (LSTM), Convolutional Neural Network (CNN), Dense Neural Network (DNN), and Auto Encoders, each enhanced with a comprehensive feature set and extensive hyperparameter tuning. The Auto Encoders + 1-D CNN model, using optimized hyperparameters and ReLU, achieves the highest accuracy at 86%, significantly surpassing LSTM (83%), CNN (83%), and DNN (79%). These findings demonstrate the effectiveness of diverse feature sets, alternative activation functions, rigorous hyperparameter tuning, and data augmentation in speech emotion recognition, suggesting new paths for enhancing human-computer interaction technologies.

Introduction

1.1 Background and Executive Summary

Emotions are an important aspect of human communication and play an important role in daily relationships. When people speak, their tone, pitch, rhythm, and intonation convey not only information but also a plethora of emotional cues (Gupta et al., 2017). Recognizing these emotional indicators is critical for having courteous, empathic, and meaningful conversations with humans or with Machines. Several reasons contribute to the need for improved speech emotion recognition. For starters, it enables more natural and empathic human-computer interactions, which improves user experiences in applications such as voice assistants and customer support. Second, it has practical applications in healthcare, where tracking emotional states can dramatically improve patient care. Hence, accurate emotion recognition from speech is critical to provide a seamless and emotionally intelligent experience in customer service, healthcare, virtual assistants, and other Applications.

Traditional techniques for Speech Emotion Recognition (SER) typically rely on the manual extraction of handcrafted features and the utilization of conventional machine learning algorithms. Emotion classification in these predominantly depends on machine learning models like Support Vector Machines (SVMs) (Chen et al., 2006), Hidden Markov Models (HMMs), Gaussian Mixture Models (GMMs), k-Nearest Neighbors (k-NN) (Pao et al., 2007), and Decision Trees. These techniques have historically been the go-to methods for extracting relevant features and classifying emotions from speech. However, these conventional approaches face inherent limitations. They may struggle to fully encapsulate the intricate and nuanced patterns present in speech that correlate with different emotions. The reliance on hand-engineered features limits their ability to capture the high-level abstractions intrinsic to

emotional expression. Consequently, these traditional techniques often fall short in providing precise and robust emotion recognition, especially in scenarios with a wide range of emotions and diverse datasets.

Deep learning models, such as neural networks, have revolutionized the field by addressing these shortcomings, enabling the automatic extraction of features from raw audio data, modeling complex temporal dependencies, and providing a more data-driven approach to emotion recognition in speech. This undertaking is largely motivated by a set of critical goals. The quest of a significant improvement in SER (Joby & Praseetha, 2022) precision, providing the system with the extraordinary capacity to discern between a wide spectrum of emotions transmitted in speech, is foremost among them. Simultaneously, the study is focused on enhancing the versatility of these models, assuring their usefulness over a wide range of speakers and emotional expressions. To achieve these goals, the research attempts to push the frontiers of SER by combining non-linear deep learning models with data augmentation (Salamon & Bello, 2017) approaches seamlessly. These advanced models are built with the capacity to extract detailed features from raw audio input, providing a viable route for considerably improving SER accuracy and resilience in the ever-changing field of emotion recognition from speech.

The project takes a diversified approach to realizing its purpose. The implementation of non-linear deep learning models is a fundamental strategy used in this research. Convolutional Neural Networks (CNNs) (Aloysius & Geetha, 2017), Dense Neural Networks (DNNs), Long Short-Term Memory (LSTM) networks (Hochreiter & Schmidhuber, 1997), and Autoencoders (Cobos et al., 2020) are specifically used. These models were chosen for their ability to capture the subtle emotional nuances found in speech data. By harnessing the capabilities of these models, the aim is to enhance the ability to discern and interpret emotional cues present in

spoken language. Additionally, the ReLU activation function serves to alleviate training complexities like fostering efficient model training.

Convolutional Neural Network (CNN)

Convolutional Neural Networks (Aloysius & Geetha, 2017) are crucial in image recognition, but they're also quite good at recognizing spoken emotions. CNNs can discover and underline relevant patterns, such as spectral variations, that are important in conveying different emotions in the context of emotion recognition. By processing audio in small segments and learning from local patterns, CNNs enable the model to capture the variations in emotional expression present in different parts of speech.

Dense Neural Network (DNN)

DNNs (Kaushik et al., 2022) are useful for processing a wide range of audio descriptors to reveal subtle emotional correlations since they are essential in revealing complex relationships between features. DNNs utilize their voluminous modeling potential to uncover elaborated interconnections between diverse audio delineators, tendering an exhaustive discernment of emotional correlations. DNNs are characterized by densely linked layers, in which every neuron in one layer is connected to every other layer's neuron. This allows DNNs to learn complex patterns and representations from data. This architecture is flexible and can be used in a variety of fields, such as audio analysis, picture recognition, and natural language processing. It is especially well-suited for problems involving complicated interactions between features. DNNs can simulate extremely nonlinear mappings between input and output because they use non-linear activation functions to add flexibility and capture complex relationships.

Long Short-Term Memory Network (LSTM)

Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997), a type of recurrent neural network, are advantageous for speech emotion recognition because they can learn long-range dependencies in audio data. Standard RNNs struggle with vanishing gradients over long sequences, but LSTMs use specialized memory cells to retain information over time. This allows LSTMs to connect emotional cues from earlier parts of a spoken utterance to the present, capturing the contextual emotional trajectory. Their ability to remember distant cues makes LSTMs well-suited to model the complex, evolving emotional expressions in speech effectively.

Autoencoders

Autoencoders (Cobos et al., 2020) are primarily employed for feature learning and reduction. In speech emotion recognition, autoencoders can automatically discover useful lower-dimensional features from audio inputs. By encoding the speech data into a more compact latent space, the autoencoder focuses on the most salient parts of the audio for predicting emotions. This allows for more efficient models by reducing redundant and irrelevant information. The learned compact features from the autoencoder contain the core emotional cues needed for accurate emotion classification from speech.

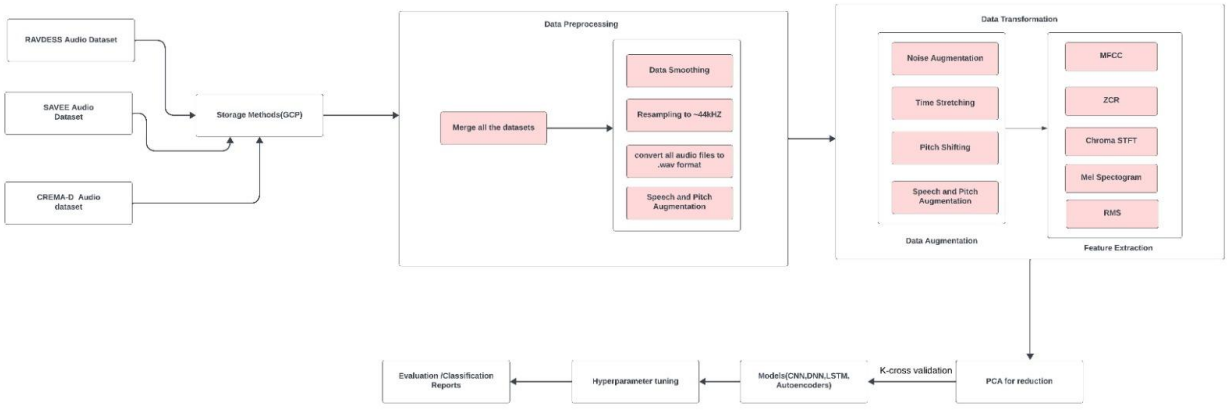
Each of these deep learning models contributes unique capabilities to the project, facilitating the recognition of subtle emotional nuances in speech.The data preprocessing prioritized audio quality and consistency, utilizing WAV format and approximately 44KHz sampling rate while minimizing background noise. Advanced techniques, such as white noise addition, pitch modification, time stretching, and speed/pitch alterations, were applied for further audio quality enhancement (Ayadi, Kamel, & Karray, 2011). A well-organized data frame

facilitated efficient access to files linked to their original audio paths. To address uneven emotion distribution, especially for less common emotions, strategic data augmentation methods were employed. Feature extraction encompassed various attributes like Zero Crossing Rate (ZCR), Root Mean Square (RMS), Mel Frequency Cepstral Coefficients (MFCC), Chroma Short-Time Fourier Transform (STFT) and Mel Spectrogram (Rezapour Mashhadi & Osei-Bonsu, 2023). Feature standardization involved row-wise Z-score transformation. Initially, 163 features were identified, later narrowed down to 30 principal components using Principal Component Analysis (PCA), capturing 95% of the dataset's variance, an essential step for effective machine learning model accuracy.

To test effectiveness in recognizing emotions in speech, the models are carefully examined using common evaluation measures such as accuracy, confusion matrix, precision, ROC curves, and F1-score (Hossin, 2015).These advanced techniques not only enhance accuracy and robustness but also set a precedent for future research in audio processing and emotion recognition, driving the evolution of SER systems. Figure 1 illustrates the data flow architecture of Speech Emotion Recognition.

Figure 1

Data Flow Architecture



This project is poised to make significant advancements in the field of Speech Emotion Recognition (SER) through its multifaceted approach. The research intends to expand speech emotion identification skills beyond traditional limits by incorporating nonlinear deep learning architectures such as CNNs (Aloysius & Geetha, 2017). Through rigorous empirical validation of these models, the project will refine state-of-the-art speech emotion recognition, setting a new bar for accuracy and robustness. The multifaceted strategies promise significant advancements in recognizing nuanced emotions from speech, laying groundwork to enable more emotionally intelligent systems in the future.

This project's improved SER system has enormous potential in a variety of applications. In the field of voice assistants and human-computer interaction, the technology can considerably increase virtual assistants' understanding and responsiveness to users' emotional states. This breakthrough has the potential to improve interactions by making them more natural, empathic, and user-centric (Gupta et al., 2017). Also, Industries that rely on customer service and sentiment analysis can also greatly benefit from the project's contributions. Businesses may use the improved SER accuracy to obtain deeper insights into consumer sentiment during interactions, enabling them to tailor responses and services more effectively to enhance customer satisfaction. The project's advances in emotion recognition will show promise for mental health monitoring in the field of healthcare. The system can be used to track and assess patients' emotional states, providing mental health providers with vital diagnostic and therapeutic assistance tools.

1.2 Project Requirements

Functional Requirements

The proposed speech emotion recognition system requires carefully implementing complex neural network architectures like CNNs, DNNs, LSTMs, and potentially autoencoders.

Convolutional Neural Networks (CNNs) (Aloysius & Geetha, 2017) for extracting features, Long Short-Term Memory networks (LSTMs) for managing time-series data, and potentially Autoencoders for noise reduction or data compression, to ensure meticulous audio data processing. The employment of the 1D CNN in Speech Emotion Recognition (SER) system ministers the particular function of deriving temporal attributes from speech statistics. Tailored to the analytics of audio signals with fluctuating templates over time, the 1D CNN (Aloysius & Geetha, 2017) efficiently processes time-series data. Its convolutional operations along the time axis are pivotal for identifying a range of emotions in speech, rendering it an indispensable component for SER applications. The 1D CNN's adaptability to wavering input lengths and its effectuality align with the system's objective of accurately sensing and recognizing emotions in speech.

Contrastingly, the Dense Neural Network (DNN) (Kaushik et al., 2022) undertakes a distinct approach to emotion identification. Utilizing fully affiliated networks, DNNs recognize elaborate templates by bonding each neuron across tiers. This architecture facilitates an exhaustive examination of emotional content in speech by coalescing data from the entire audio range. The DNN's capability to decode non-linear relationships enables the identification of minute particulars in emotional expressions, directing the system's particular goal of seizing nuanced emotional cues.

Additionally, the model enlisting Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) networks, a category of recurrent neural network, plays a critical role in processing sequential data. In the context of SER, where apprehending the evolution and background of emotional cues over time is paramount, LSTMs outperform in pinpointing long-range dependencies in time-series data. Maintaining a memory of previous data entrances

allows LSTMs to contextualize current data, providing a comprehensive understanding of the emotional trajectory in speech patterns. The exceptional memory and harnessing of past information render LSTMs particularly well-suited for SER tasks, aligning with the system's particular stipulation of precise emotion detection founded on temporal dynamics.

Buckets on Google Cloud Platform (GCP) are used for secure, scalable storage solutions. GCP Data Pipeline is only accessible by authorized team members. To maintain data security and integrity, strict procedures are in place, guaranteeing restricted access. Team members can retrieve and process data simultaneously with this limited access, which not only complies with legal and regulatory standards but also facilitates remote and collaborative work. IAM (Identity and Access Management) roles within GCP (Qian et al., 2009) are employed to manage access permissions, adding an additional layer of security. The project is exempt from any particular legal or regulatory obligations, including those related to HIPAA, IRB, or PHRP certifications. Similarly, CREMA-D (Cao et al., 2014), RAVDESS (Livingstone & Russo, 2018), SAVEE (Haq & Jackson, 2010) databases do not require licensing or authorization. The absence of legal restrictions allows for greater freedom in the use of data and the creation of models. Interactions with Google Cloud Platform are an important part of external interfaces, highlighting the significance of safe access and data management in that setting.

The system effectively preprocesses raw audio data into useful representations like MFCC features and ZCR (Rezapour Mashhadi & Osei-Bonsu, 2023).

Performing proper extract, transform, load (ETL) processes is vital to ready the data for model training. This includes critical normalization steps to ensure consistent, scalable data during training. Model training itself follows a rigorous methodology using established datasets like RAVDESS (Livingstone & Russo, 2018), SAVEE (Haq & Jackson, 2010), and CREMA-D

(Cao et al., 2014). Advanced optimization techniques are leveraged to maximize model performance. Training continues until optimal accuracy is reached or early stopping criteria are met. Careful emphasis on robust ETL pipelines and rigorous training procedures is crucial for enabling the system to learn effectively from the prepared datasets and achieve strong speech emotion recognition capabilities.

After training, the models are thoroughly evaluated using reliable metrics like accuracy, F1 score (Hossin, 2015), and confusion matrices on separate validation datasets. This produces comprehensive performance reports. To meet scalability needs, the models must capably handle datasets of different sizes. Thorough, multi-faceted evaluation and clear reporting ensures model integrity and usability, while scalability enables broad applicability across diverse dataset sizes. Together these factors lead to trustworthy, flexible, and understandable speech emotion recognition.

Implementing robust error handling and logging is vital to account for potential issues arising during data preparation, model training, or evaluation. This provides detailed logs that can be analyzed thoroughly to identify and troubleshoot any problems after the fact. With reliable error and exception capturing along with extensive logging at each stage, the system can detect and record discrepancies early on. The comprehensive logs enable tracing the root causes of failures through in-depth post hoc analysis. This error handling and logging infrastructure is key to debug, optimize, and improve the reliability of the speech emotion recognition system.

AI-powered Requirements

Speech emotion detection models are created, trained, and potentially deployed using a variety of complex libraries and tools. TensorFlow and Keras are two of the most important tools, as they enable the basic deep learning model implementations.

Each model architecture, including CNNs, DNNs/LSTMs, and autoencoders, is carefully implemented with TensorFlow or Keras. The models are tailored to handle the distinct properties of emotional speech data. For CNNs (Aloysius & Geetha, 2017), the AI solution must effectively extract time-based attributes from speech data. This necessitates adeptness in accommodating fluctuating input lengths. The AI system must demonstrate proficiency in processing time-series data, ensuring a thorough analysis of audio signals with wavering templates over time. The input prerequisites encompass raw audio data, while the anticipated output is a suite of derived temporal traits essential for identifying a range of emotions in speech.

Similarly, for Dense Neural Networks (DNNs) (Kaushik et al., 2022) and Long Short-Term Memory networks (LSTMs) (Hochreiter & Schmidhuber, 1997), the AI solution should be adept at seizing elaborate templates and processing sequential data, respectively. DNNs ask for the capability to integrate data from the entire audio range, permitting a comprehensive review of emotional content in speech. LSTMs, conversely, should outperform in pinpointing long-term dependencies in time-series data, contextualizing current data to provide a comprehensive discernment of the emotional trajectory in speech patterns. The input for DNNs encompasses audio data with intricate templates, while LSTMs (Liu et al., 2018) require sequential data as input, with both models yielding outputs indicative of nuanced emotional cues.

Dimensionality reduction and data selection techniques are prudently used to construct a varied, representative training dataset. Overall, the thoughtful application of core libraries like TensorFlow, Keras, and librosa, combined with customized model designs and data curation strategies, enables creating specialized deep learning models for speech emotion recognition.

Rigorous evaluation protocols leverage metrics from sklearn to align model performance with project goals. Training procedures, optimization strategies, and evaluation workflows are

significantly augmented by the computational resources of Google Colab. Overall, the thoughtful integration of core libraries like TensorFlow, Keras, librosa, and sklearn, together with Google Colab's computing capabilities, enables rigorous training, optimization, and evaluation of the models to meet the speech emotion recognition objectives.

This end-to-end framework meets the computational needs of the current speech emotion recognition project. Furthermore, it lays the groundwork for future initiatives, with the ability to scale up to real-time emotion detection applications. The combination of sophisticated libraries, tools, resources, and approaches results in a platform that is both expansive and extendable. This accomplishes current goals while also setting the framework for future advancements in emotion recognition capabilities.

Data Requirements

Meeting accurate data-related requirements is critical for developing an effective voice emotion recognition system. These stipulations govern the properties, quality, formatting, storage, preprocessing, and possible augmentation of the audio data, which is core to accurately detecting emotional signals from human speech. Strictly satisfying the data prerequisites is imperative for the system to reliably recognize emotions from voice inputs.

The primary data consists of audio files of human speech expressing various emotions, sourced from a wide demographic background. This diversity is key to capture nuanced tonal, pitch, accent, and emotional variations in human speech. Selected datasets for this undertaking CREMA-D (Cao et al., 2014), RAVDESS (Livingstone & Russo, 2018), SAVEE (Haq & Jackson, 2010) are renowned for their breadth and diversity. It is critical that these audio recordings be of high quality, with minimal background interference, in order for the proposed

models to perform reliable evaluations. Maintaining data integrity and coherence requires uniformity in audio formats (ideally WAV) and consistency in sampling rates.

The RAVDESS (Livingstone & Russo, 2018) dataset provides high-quality diversity, with 24 actors of different genders expressing various emotions through speech and song. The audio files are high-fidelity .wavs with 48kHz sampling rate and 24-bit depth, ensuring rich, clear data. The range of emotions - neutral, calm, happy, sad, angry, fearful, surprised, disgusted - delivered in speech and song enables thorough emotion analysis. RAVDESS's professional diversity, audio fidelity, and broad emotion spectrum make it an ideal data source for comprehensive speech emotion recognition.

The SAVEE (Haq & Jackson, 2010) dataset contains consistent data from four male British English speakers expressing seven different emotions. The audio is of excellent quality.Wav at a sample rate of 44.1kHz. The 1-2 second recordings effectively capture diverse emotions. Though smaller in scale, SAVEE's consistency, audio fidelity, and concise emotional utterances make it a valuable complementary dataset for the system.

The CREMA-D (Cao et al., 2014) dataset provides diverse data from 91 actors of different ages and ethnicities. This captures varied vocal profiles. It includes six key emotions plus a neutral state in high-quality 16-bit, 48kHz .wav audio. CREMA-D's diversity of voices and emotions with professional recording quality make it an important dataset for generalizable speech emotion recognition.

Large amounts of data are essential for the deep learning models to generalize across diverse voices and emotion nuances. Additionally, secure and scalable data storage is extremely important. Google Cloud Storage buckets are the optimal choice given their proven effectiveness and scalability for cloud-based data storage solutions. Overall, ample real or synthetic data along

with robust cloud storage are pivotal prerequisites to enable deep learning models to reliably recognize emotions from diverse speech.

Processing the raw audio data is crucial to transform it into representations suitable for model input. When data is scarce, the system utilizes data augmentation techniques (Salamon & Bello, 2017) to synthetically expand the dataset with realistic variations. This improves model robustness and accuracy. Careful data processing and augmentation ensures the models are trained on ample informative representations derived from the raw audio.

In conclusion, complying to these data requirements is critical in the engineering of emotion detection models, assuring their correctness, dependability, and scalability across a wide range of real-world applications.

1.3 Project Deliverables

Project Proposal

Description. The journey commences with the project proposal. This proposal acts as a guiding document, detailing work's aims and scope. The aim here is to enhance speech emotion recognition (Gupta et al., 2017) through the integration of non-linear deep learning models and data augmentation. The proposal is the foundation, defining the significance of the project and the research issues that will be addressed. It also provides an initial project timeline and introduces the technology and techniques that will be used, establishing expectations for the project's outcomes. The proposal helps to shape expectations for the project's outcomes.

Due Date. 25 September 2023

Work Breakdown Structure

Description. With the proposal completed, the next step was creating a Work Breakdown Structure, a comprehensive roadmap meticulously dividing the project into manageable

components. This facilitated resource allocation, task assignment, and overall project management. To ensure a structured, systematic approach, the Cross-Industry Standard Process for Data Mining methodology was selected. This widely used methodology guides the project through six clearly defined phases:

Business Understanding. Gained a full understanding of the business or research context by reviewing literature and defining clear objectives.

Data Understanding. Assessed the available datasets' quality, structure and content before collecting and preprocessing data to fully grasp the data's nature and anticipate challenges.

Data Preparation. Cleaned, formatted, and integrated the dataset, handling missing data and extracting essential audio features to prepare the data for subsequent model building.

Modeling. Designed and implemented non-linear deep learning models for speech emotion recognition after preparing the data, exploring architectures, hyperparameters, and optimization techniques to improve model performance.

Evaluation. Rigorously tested and validated the deep learning models using metrics like accuracy, precision, recall and F1 score (Hossin, 2015) to evaluate effectiveness in emotion recognition.

Deployment. Aimed to integrate the successfully developed and tested models into a working system or application.

JIRA has been used for project planning and management. This platform has made it easier to share the Work Breakdown Structure and set expectations for the project's outcomes.

Due Date. 4 October 2023

Gantt & PERT Charts

Description. Building on the WBS foundation, Gantt and PERT charts are constructed to

visually plan and manage the project. The Gantt chart maps out the timeline, surfacing task dependencies and deadlines for precise tracking. Meanwhile, the PERT chart sequences activities and estimates durations to identify critical paths and logical workflow. Together, these powerful project management tools enable robust scheduling, monitoring, and optimization for project execution.

Due Date. 16 October 2023

Data Management Plan

Description. The Data Management Plan is a document that provides a comprehensive work breakdown structure of the project into phases, deliverables, and work packages, ensuring clarity and systematic progress. Addressing Project Resource Requirements & Plan, hardware, software, tools, licenses, specifications, costs, and justifications are thoroughly outlined, ensuring a well-equipped environment for project execution. Project Schedule was presented through a Gantt Chart, offering a visual representation of tasks, timelines, responsible team members, and the status of deliverables, while a PERT Chart facilitated detailed project analysis, considering individual tasks and dependencies.

Due Date. 26 October 2023

Data Collection

Description. A comprehensive document has been created that describes in detail the sources, particular parameters, and the raw datasets CREMA-D (Cao et al., 2014), RAVDESS (Livingstone & Russo, 2018), SAVEE (Haq & Jackson, 2010) that are used to train models. This document serves as a comprehensive reference, ensuring clarity and transparency regarding the foundational datasets integral to the project. Moreover, the plan includes specific details on finding data, the procedure for collection, the duration of data collection, and the individuals

responsible for the task. In a collaborative effort, team members Swati, Shashank, Rohith, and Yamini all shared equal responsibility for data collection. Notably, Rohith took on the task of uploading the datasets to Google Colab, enhancing the efficiency and collaborative nature of data collection endeavors.

Due Date. 6 November 2023

Data Engineering

Description. Included within the comprehensive data engineering chapter, a detailed account outlines the utilization of key technologies such as TensorFlow, Keras, librosa (McFee et al., 2015), sklearn, and Google Colab**.** The project development process had a work breakdown structure that outlined phases, deliverables, and work packages. Project timelines, resource needs, and AI-powered solutions were all seamlessly integrated. Legal and regulatory considerations were described in order to guarantee transparent data practices. Data processing, transformation, preparation, and statistical summaries were carefully described, and samples and visualizations were carefully included in an extensive document.

Due Date. 13 November 2023

Presentation Slides

Description. The presentation slides demonstrated a thorough synopsis of the model selection procedure, updates, explanations, and perceptive comparisons, which successfully captured the complex features of models. Each slide was painstakingly created with a logical structure and clear presentation of the intricacies of models. The team provided a clear grasp of goals, needs, problems to be addressed, and deliverables by skillfully navigating through the project background, executive summary, and specifications. The presentations also included a comprehensive overview of technology and literature reviews, indicating a deep comprehension

of the current environment. The findings of the technological and literature survey were succinctly and thoroughly categorized, compared, and summarized. The presentation outlined project resource needs and plans comprehensively, including hardware, software, tools, licenses, costs, with explanations.

Due Date. 27 November 2023

Final Project Report

Description. The project culminates with an all-encompassing report summarizing the entire undertaking. It compiles the objectives, methodologies, results, and conclusions into one comprehensive document. The report encapsulates all aspects of the project from start to finish, consolidating the key activities and discoveries. Additionally, it may provide recommendations for future work and potential extensions of the project. Overall, the detailed report offers a holistic overview of the speech emotion recognition project, documenting the end-to-end journey and outcomes**.**

Due Date. 10 December 2023

Table 1 includes the detailed description of the Project Deliverables for Speech Emotion Recognition.

Table 1

Detailed table listing all project deliverables

Deliverable	Description	Due Date
Proposal of the Project	An initial report outlining the project's scope and objectives.	25 September 2023

Deliverable	Description	Due Date
WBS	WBS that delineates timelines and assigns responsibilities to team members.	4 October 2023
Gantt & PERT Charts	Gantt chart monitors progress across timeline, while the PERT chart highlights critical interdependencies.	16 october 2023
Data Management Plan	Addressing Project Resource Requirements & Plan, hardware, software, tools, licenses and costs are thoroughly outlined.	26 october 2023
Data Collection	Exploring RAVEDESS ,SAVEE and CREMA-D datasets.	6 November 2023
Data Engineering	Description of Data processing, transformation, preparation, and statistical summaries	13 November 2023
Presentation Slides	Comprehensive overview of technology and literature reviews, Models, Tools and Technologies Used	27 November 2023
Project Report	Final project report will include entire documentation in APA formatting.	10 December 2023

1.4 Technology and Solution Survey

In the fast-moving world of technology today, the field of Speech Emotion Recognition (SER) is seeing major improvements. This is because of new technologies and clever new ways of doing things. Looking closely at the current technologies shows that there are many different

approaches. Each one is designed to handle the complicated job of finding emotional hints in speech.

The work by Triantafyllopoulos et al. (2019) investigated utilizing deep residual networks (ResNets) for deep learning-based speech enhancement to improve robustness in speech emotion recognition systems. Their research employed scalable ResNet architectures for removing noise from speech signals while retaining emotional cues. By leveraging advanced deep learning and speech processing algorithms, they enabled more robust emotion recognition from speech under diverse noise conditions.The approach of combining deep residual networks, speech enhancement models, and integration into emotion recognition pipelines contributed significantly to the fields of deep learning, speech processing, and affective computing. The authors demonstrated deep ResNets can be trained to eliminate noise while preserving emotional content in speech. This improves resilience to various noise types and levels in speech emotion recognition. Their innovations in deep learning and speech processing advance the robustness and accuracy of real-world speech emotion recognition technologies.

Zhao et al. (2018) explored leveraging deep learning techniques for accurate speech emotion recognition. Their work focused on developing 1D and 2D CNN LSTM networks to capture both local features and long-term dependencies from speech and spectrograms. The dual networks aimed to combine CNNs' local learning with LSTMs' sequence modeling. Experiments demonstrated the 2D architecture's superior performance over traditional models like DBNs and CNNs on benchmark emotion databases. The paper provided insight into challenges in extracting emotional cues from speech, offering an overview of deep learning trends for speech processing. The authors highlighted the importance of hierarchical feature learning to capture nuanced emotions from speech. Their innovations in deep network design advanced emotion recognition

capabilities, with potential applications in healthcare, wellness, and human-computer interaction. Overall, the research contributed significantly to deep learning, speech analysis, and affective computing.

The study by Lim et al. (2016) explored speech emotion recognition using a combination of convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Their work focused on leveraging CNNs to extract features from 2D representations of speech signals via convolution and pooling layers. Additionally, they employed RNNs, specifically LSTM networks, to model sequential speech data and capture long-term dependencies related to emotional expression.The authors introduced time distributed CNNs, fusing deep CNN hierarchies with LSTM layers to improve recognition performance compared to standalone CNN or RNN models. Experiments demonstrated the advantages of their proposed architecture over traditional approaches relying on hand-crafted features for speech emotion recognition.This research highlighted the potential of deep learning techniques, through integrating CNNs and RNNs, to advance speech processing and affective computing capabilities without dependence on manual feature engineering. The authors contributed innovative modeling strategies for encoding emotional cues from speech using neural network architectures. Their work provides valuable insights into deep learning for sequential data analysis tasks such as speech emotion recognition.

The research focused on speech emotion recognition using spectrograms and deep convolutional neural networks (CNNs). Their work introduced a methodology leveraging spectrogram representations of speech data and CNNs to extract discriminative features for classifying emotions. The study highlighted the importance of robust feature learning to advance emotion recognition capabilities for applications like human-computer interaction.The authors

detailed their proposed framework involving spectrogram generation and a CNN architecture with convolutional and fully connected layers for feature extraction from speech. They described the model training process using a benchmark emotion dataset and computational resources. Experiments demonstrated accurate classification of emotions including anger, boredom, disgust, fear, happiness, neutrality, and sadness. The research also explored transfer learning techniques, revealing challenges in adapting pre-trained models for emotion recognition tasks Badshah et al. (2017).

Emphasis is laid on the need for advanced feature learning techniques to enable accurate speech emotion recognition. The authors contributed an effective deep learning methodology using spectrograms and CNNs to extract informative representations from speech for emotion classification. Their innovations provided valuable insights to progress robust machine analysis of emotional states from speech signals.

Trigeorgis et al. (2016) have explored end-to-end speech emotion recognition using deep convolutional recurrent neural networks, obviating the need for hand-crafted features. Their approach leverages convolutional layers for automatic feature learning directly from raw speech waveforms, combined with LSTM layers to model temporal context. The aim is to capture emotional nuances in speech through robust feature extraction that accommodates diverse speaking styles while ensuring adaptability. By integrating CNN and LSTM networks, their model demonstrates superior performance over conventional methods relying on engineered features for the emotion recognition task. Notably, they optimize model training using the concordance correlation coefficient as the objective, enhancing prediction accuracy.

Comprehensive analyses reveal their model learns discriminative features aligned with known acoustic and prosodic cues for conveying emotion through speech.This end-to-end

framework exemplifies a paradigm shift in feature extraction, moving from manual design to automated learning. Results on public datasets evidence the potential of deep convolutional recurrent networks to advance the state of the art in speech emotion recognition through robust feature learning. Their work provides valuable insights into the design and interpretability of such models, establishing a strong foundation for future research on deep learning for speech emotion analysis.

Li et al. (2021) explore speech emotion recognition using recurrent neural networks with directional self-attention. They address the need to extract relevant signals and increase informational diversity for accurate emotion recognition. Leveraging LSTM for learning long-term dependencies and BLSTM for robust structure handling via bidirectional processing, their approach also incorporates self-attention. This deals with speech frame autocorrelation and automatically annotates frame weights, selecting emotionally salient frames.Their study emphasizes distinguishing emotionally relevant and irrelevant frames, considering challenges from silent regions, short pauses, and phoneme transitions. Applying attention can automatically differentiate emotional/non-emotional frames. Directional self-attention determines frame autocorrelation within utterances. Integrating the bidirectional mechanism with self-attention alleviates performance decline from insufficient information. The proposed BLSTM-DSA architecture automatically annotates frame weights via self-attention, improving SER efficiency. Analyses demonstrate self-attention's effectiveness in assigning distinct frame weights and discovering inter-frame correlations. The paper provides comprehensive insights into the BLSTM-DSA framework, its decoding using Bi-LSTM, and incorporating directional self-attention. Detailing the algorithm and self-attention block operations illustrates the approach's applicability and advantages for enhancing speech emotion recognition accuracy.

Overall, their research showcases promising directions leveraging neural attention mechanisms to advance the state of the art in this field.

The Research "Speech Emotion Recognition Using Deep Neural Network and Extreme Learning Machine", Han et al. (2014) addresses the complex problem of recognizing emotions from speech signals. They propose a novel approach that leverages the capabilities of deep neural networks (DNNs) to extract high-level features from raw data. By utilizing DNNs, the authors demonstrate the effective learning of emotional information from low-level acoustic features, showcasing the potential of this technique in the field of emotion recognition. The proposed algorithm involves segment-level feature extraction, training the DNN for segment-level emotion recognition, and constructing utterance-level features from the segment-level outputs. Finally, an extreme learning machine (ELM) is employed for the utterance-level emotion classification. The experimental results conducted on the IEMOCAP database indicate a substantial 20% relative accuracy improvement compared to existing methods, showcasing the effectiveness of the proposed approach in enhancing emotion recognition from speech signals.The paper emphasizes the significant contribution of utilizing DNNs to learn emotional information from low-level acoustic features, offering promising results in the challenging task of speech emotion recognition. The authors highlight the potential of their proposed approach in effectively recognizing emotions in speech, with the use of ELM further enhancing the performance of utterance-level emotion classification. By demonstrating the substantial improvements achieved over existing methods, the paper underlines the value of leveraging advanced neural network techniques for addressing the challenging problem of emotion recognition, opening up new avenues for natural human-computer interaction and communication.

Zhang and Xue (2021) investigated utilizing data augmentation techniques to address the challenge of limited labeled data availability in speech emotion recognition (SER). Their research focused on augmenting the log magnitude spectrogram input representation to generate additional varied training samples. By applying deformations like time warping, frequency masking, and time masking to spectrograms, they successfully created more diversity in the training data.Their data augmentation approach targeting spectrogram inputs enabled the SER model to capture broader emotional nuances in speech. This improved generalization and recognition capabilities despite limited real training data. The authors' innovations in pragmatic data augmentation underscored its importance in enhancing model robustness and performance for SER with scarce labeled corpora. Zhang and Xue demonstrated how careful augmentation of spectrogram representations can address key data scarcity challenges in advancing speech emotion recognition research and applications.

The proposed method by Zhang et al. (2019) utilizes a multiscale deep convolutional LSTM framework for spontaneous speech emotion recognition. It integrates deep CNNs with LSTM at multiple spectrogram segment lengths, enabling extraction of high-level affective features from raw audio data. This improves the model's accuracy in recognizing spontaneous emotions.By leveraging deep CNNs, the approach can effectively capture intricate segment-level patterns from the spectrograms. The subsequent LSTM integration facilitates modeling of temporal dependencies within utterances, allowing comprehensive utterance-level representations. Notably, the multiscale spectrogram segments as CNN inputs enhance handling of diverse affective cues in spontaneous speech. Overall, integration of these advanced deep learning techniques demonstrates a promising solution to the challenging task of spontaneous

emotion recognition from speech. The multiscale strategy parallels data augmentation, improving model generalization despite limited data.

Etienne et al. (2018) in their research on speech emotion recognition have proposed a neural network architecture combining convolutional and recurrent layers. Their approach leverages deep learning for audio analysis, using a CNN to extract high-level spectrogram features followed by recurrent layers to model temporal dependencies. To tackle limited data and class imbalance in the IEMOCAP benchmark, they explore data augmentation techniques including vocal tract length perturbation, layer-wise optimizer adjustment, and recurrent batch normalization. The IEMOCAP dataset presents challenges due to its small size and skewed emotional class distribution. Vocal tract perturbation is used for data augmentation, yielding noticeable improvements. Batch normalization is examined to maintain signal structure, although its effectiveness is constrained by GPU memory. Adjusting optimizer parameters for convolutional and recurrent layers is also investigated, motivated by their different gradient scaling, but results are inconclusive. A 10-fold cross-validation methodology ensures rigorous evaluation. Overall, this work achieves competitive weighted and unweighted accuracy on IEMOCAP for recognizing four primary emotions. The research provides meaningful insights into data augmentation and neural network design for speech emotion recognition with limited imbalanced data. It makes valuable contributions towards advancing the state of the art in this domain.

Padi et al. (2022) in their paper "Multi-window Data Augmentation Approach for Speech Emotion Recognition " present a novel multi-window data augmentation method (MWASER) to improve speech emotion recognition (SER) performance. Their approach focuses on two key aspects: designing the speech augmentation method and building the deep learning

model for recognizing emotions in audio signals. The primary objective is to address model overfitting by generating additional data samples. The paper outlines challenges in accurately recognizing speech emotions and emphasizes the importance of optimal window size selection during feature extraction. It discusses using CNNs for SER and techniques like dropout, regularization, batch normalization, and transfer learning to mitigate overfitting. Data augmentation is proposed as an effective strategy to provide more generalized training data, reducing overfitting.The MWASER approach leverages multiple window sizes during audio feature extraction to enable the model to analyze both shorter and longer emotional utterances. The proposed CNN architecture employs multiple convolutional, fully-connected, dropout, batch normalization, and max-pooling layers to learn robust SER features. Experiments on IEMOCAP, SAVEE, and RAVDESS showed MWASER significantly improved over a single-window baseline. Analyses reveal varying optimal window choices across datasets. Overall, the multi-window augmentation approach effectively addresses key SER challenges and enhances emotion recognition performance.

Lieskovská et al. (2021) provides a comprehensive overview of using deep learning and attention mechanisms for speech emotion recognition (SER). They discuss the common emotional speech databases used, including simulated, elicited, and spontaneous recordings, noting the challenges with each type. The authors explain how attention mechanisms can outperform traditional pooling methods like global max/average pooling in SER by learning to focus on emotionally salient parts of utterances. The use of attention in deep recurrent neural networks for SER is explored, showing how attention provides better temporal representations. The authors argue attention allows models to adapt to emotional changes in speech, improving accuracy. Different alignment scores for attention are analyzed, evaluating their impacts on

performance. Overall, the authors demonstrate how attention mechanisms are beneficial for SER, enabling models to concentrate on emotionally informative regions of speech and handle variations over time. Table 2 shows the Comparisons of various Models and their research.

Table 2

Comparison of solutions

Author/Year	Approaches and Models	Key Findings and Conclusions	Results
Triantafyllopoulos et al., 2019	Deep Residual Networks (ResNets) for speech enhancement in emotion recognition.	Showed ResNets' effectiveness in removing noise while preserving emotional content in speech,	84.3%
Zhao et al., 2018	1D and 2D CNN LSTM networks	Demonstrated the 2D architecture and emphasized the significance of hierarchical features.	86.36%
Lim et al., 2016	Combination of CNNs and LSTMs	Contributing to the advancement of SER without manual feature engineering.	86%

Author/Year	Approaches and Models	Key Findings and Conclusions	Results
Badshah et al., 2017	Spectrograms and deep CNNs.	Proposed an effective deep learning methodology for accurate emotion classification	84.3%
Trigeorgis et al., 2016	Deep Convolutional recurrent neural networks.	Performance of deep convolutional recurrent networks in advancing speech emotion recognition.	Improved signal to noise ratio of audio by 10%
Li et al., 2021	Recurrent neural networks with directional self-attention.	Provided insights into the BLSTM-DSA framework's operations.	88%
Han et al., 2014	Deep neural networks	Effectiveness of DNNs in extracting emotion cues.	20% relative accuracy improvement compared to the state-of-art-approaches

Author/Year	Approaches and Models	Key Findings and Conclusions	Results
Lieskovska et al., 2021	Review of SER using deep learning and attention mechanism,	Provided insights into different emotional speech databases and impact of attention mechanisms to improve recognition accuracy.	75% Accuracy
Zhang et al., 2019	Multiscale deep convolutional LSTM.	CNNs and LSTM integration, along with the multiscale strategy, in improving the model's accuracy.	10% Accuracy compared more than the state of arts
Etienne et al., 2018	Neural network architecture combining CNN, RNN	Achieved competitive accuracy in recognizing emotions providing insights into data augmentation.	64.5%
Padi et al., 2022	Multi-window data augmentation method	Effectiveness of the multi-window augmentation	88%

1.5 Literature Survey of Existing Research

Puri et al. (2022) introduces a hybrid Convolutional Neural Network (CNN) architecture for speech emotion recognition using the RAVDESS dataset. The model focuses on classifying emotions into positive, negative, neutral, and more specific categories like happy, sad, angry etc. The authors combine LSTM, HMMs, DNNs with CNNs for emotion classification. They also leverage MFCCs and Log Mel Spectrograms for audio feature extraction. The proposed 8-layer CNN model demonstrates improved performance over previous approaches. The paper details the data preprocessing, feature extraction, model architecture and algorithms used. Results show over 98% accuracy, highlighting the efficacy of their hybrid CNN approach for speech emotion recognition. This work has valuable applications in education, automotive, security, communication and healthcare by enabling systems to understand emotions from speech. While acknowledging challenges in accurately interpreting speech emotions, the study emphasizes the importance of feature selection and classification algorithms. It contributes to advancing speech emotion recognition research, underlining the need for robust, balanced datasets. Overall, the proposed CNN model provides useful insights for developing more sophisticated speech emotion recognition systems.

The research by Toyoshima et al. (2023) develops a multi-input deep learning model for speech emotion recognition using Mel spectrograms (MelSpec) and the Geneva minimalistic acoustic parameter set (GeMAPS). By integrating these complementary features into a unified architecture, their model aims to improve recognition accuracy. The study focuses on classifying four emotions - anger, happiness, sadness and neutrality - using the IEMOCAP speech database. A key contribution is addressing the issue of imbalanced emotional speech data by incorporating focal loss during training. Related works covering emotional speech data, audio features, and

deep learning models are surveyed. While prior multi-input models exist, the proposed integration of GeMAPS and focal loss represents a novel approach. The methodology outlines the dataset, feature extraction, model architecture, and training process with categorical cross-entropy and focal loss objectives. Experiments evaluate individual MelSpec and GeMAPS models, validating the benefits of the multi-input design and focal loss for imbalanced data. Overall, this research advances speech emotion recognition through an integrated model that merges complementary features and handles imbalanced training data.

The paper "A review of multimodal emotion recognition from datasets, preprocessing, features, and fusion methods", Pan et al. (2023) presents a comprehensive overview of affective computing, emphasizing the importance of understanding and replicating human emotions for advanced human-computer interaction. It categorizes emotion models into discrete and dimensional representations, highlighting multimodal emotion recognition through facial expressions, speech, text, behavior, and physiological signals. The paper outlines general procedures in multimodal emotion recognition - data collection, preprocessing, feature extraction, and fusion of modalities for accurate emotion identification. The authors also stress the significance of multimodal emotion datasets, analyzing both lab and real-world datasets including the eNTERFACE05 dataset collected through emotional stories. Distinguishing itself from existing surveys, the paper provides in-depth analysis of data preprocessing, feature extraction techniques, and novel perspectives on fusing modalities to improve emotion recognition. It aims to serve as a valuable resource by offering insights into current challenges and future research directions in this field.

Chen et al. (2018) discusses challenges in speech emotion recognition (SER) due to the complexity of emotions and influence of irrelevant factors. The authors propose a 3-D

attention-based convolutional RNN (ACRNN) architecture that integrates CRNN with an attention model to capture discriminative features for SER. The significance of emotional intelligence in human-machine interfaces is highlighted. The effectiveness of deep neural networks for feature extraction in SER is emphasized, but drawbacks of personalized features as input are noted. Inspired by positive results using MFCCs with deltas and delta-deltas, the authors propose calculating these for log Mel-spectrograms as input to the 3-D ACRNN. The attention mechanism focuses on emotionally relevant frames.Experiments on the IEMOCAP and Emo-DB databases demonstrate the proposed method's effectiveness, achieving superior performance over baseline methods in terms of unweighted average recall. The attention-based 3-D CRNN architecture represents an important advancement for improving SER performance.

The study introduces the Deep Canonical Time Warping (DCTW) method for learning non-linear representations of multiple time-series that are maximally correlated and temporally aligned. The authors highlight limitations of existing approaches in capturing complex non-linear representations for multi-modal data. DCTW extends traditional Dynamic Time Warping to handle varying dimensionalities and combines Canonical Correlation Analysis with DTW for efficient alignment. The significance of DCTW for signal analysis, speech, vision, and bioinformatics is emphasized. DCTW overcomes limitations by utilizing deep neural networks to capture non-linear transformations. The method is extended to a supervised setting using available labels to enhance alignment. Experiments on multiple datasets demonstrate DCTW's effectiveness in handling real-world scenarios with heterogeneous features and improving alignment over state-of-the-art techniques. The paper provides extensive discussion of related work and positions DCTW as a novel and promising approach for temporal alignment and representation learning challenges Trigeorgis et al. (2018).

The research by Jain et al. (2020) aimed to develop an efficient system to classify speech into four emotions: sadness, anger, fear, and happiness. They utilized two datasets - Linguistic Data Consortium (LDC) and University of Georgia (UGA) database - to train and evaluate the system. Key features including energy, pitch, Mel-Frequency Cepstral Coefficients (MFCCs), Linear Predictive Coding Coefficients (LPCCs), and speaking rate were extracted from speech samples to capture emotional characteristics. Emotion classification was done using Support Vector Machine (SVM) algorithms with two strategies: One Against All (OAA) and Gender Dependent Classification. Furthermore, comparative analysis between the classification strategies was conducted. The performance of MFCC and LPCC feature extraction methods was also evaluated. Results showed the SVM model achieved 85.085% overall accuracy, with LDC dataset having 90.08% and UGA dataset 65.95% accuracy. MFCC outperformed LPCC in terms of accuracy. Overall, the research demonstrated SVM's effectiveness as a robust classifier for speech emotion recognition by accurately categorizing diverse emotions based on extracted speech features.

The paper "Deep Learning Techniques for Speech Emotion Recognition", Abbaschian et al. (2021) provides a comprehensive review of Speech Emotion Recognition (SER) using deep learning techniques. The authors discuss the application of various emotional speech databases, including simulated, elicited, and spontaneous recordings, highlighting the challenges of each type. They also examine the use of different deep learning models and techniques like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Generative Adversarial Networks (GANs) for SER. The paper offers insights into the impact of these models and techniques on improving SER performance, emphasizing their ability to adapt to changes in emotional speech and increase recognition accuracy without manual feature

engineering. The authors also highlight the potential of these deep learning models and techniques to further advance SER capabilities. Overall, this review paper provides an in-depth analysis of deep learning approaches and emotional speech data for developing high-performance SER systems.

The Methodology compares two classification algorithms for speech emotion recognition: one based on Euclidean distances and the other on neural networks. Both algorithms use the same feature set and are evaluated on a database of emotional speech validated through subjective listening tests. The study emphasizes the difficulty of obtaining reliable emotional speech data. The dataset includes emotions of anger, sadness, happiness, and neutral speech, with acoustic feature extraction. The article discusses the classification methods in detail, providing insights into their architectures and training. Results are analyzed, showing distinct behaviors between the two algorithms in producing emotion indices. Detailed results and graphical representations of the classification decisions are presented for each method Amir et al. (2001).

The paper "Multi-Input Speech Emotion Recognition Model Using Mel Spectrogram and GeMAPS" introduces a novel multi-input deep neural network that leverages both Mel spectrogram and GeMAPS features for speech emotion recognition. This combined approach aims to address limitations of prior methods that relied on single audio features. A focal loss function is proposed to handle class imbalance. Experiments show the model significantly improves recognition of "happiness", a challenging emotion due to limited data. Comparisons demonstrate higher or comparable accuracy versus existing methods, highlighting the benefits of the joint feature approach. Limitations include needing more and better training data, enhancing multi-emotion recognition, and simplifying the model for practical applications. Overall, the

paper makes valuable contributions through the combined audio feature method and focal loss for imbalance, advancing speech emotion recognition and paving the way for future work.

The article "A Review of Emotion Recognition Methods Based on Data Acquired via Smartphone Sensors", Kołakowska et al. (2020) provides a comprehensive review of emotion recognition methods that leverage smartphone sensors. It emphasizes how the proliferation of smartphones with enhanced computational power and built-in sensors like cameras, microphones, accelerometers, and GPS has enabled new research into smartphone-based emotion recognition. The availability of cloud computing has also facilitated performing complex computational tasks on smartphones. The review systematically examines publications from the last decade, highlighting the different techniques used to extract meaningful features from smartphone sensor data. This includes an in-depth look at the unique characteristics of various sensors and the machine learning strategies and algorithms implemented to identify emotions based on the extracted data. Overall, the article underscores the importance of this emerging research area, given the dominance of smartphones as primary computing devices globally. By surveying the range of methodologies and sensor modalities explored thus far, the authors contribute to advancing emotion recognition systems for smartphones.

Basu et al. (2017) provides an in-depth review of emotion recognition using speech data. It highlights the key components needed to build an effective recognition system, including choosing appropriate speech corpora, identifying relevant features, and selecting suitable classification models. The authors examine different methodologies for collecting emotional speech data, underlining the complexities with each approach. They also thoroughly explore noise reduction techniques, emphasizing their importance in improving emotion detection accuracy from speech signals. Overall, the paper offers a comprehensive look at the intricacies

and critical factors involved in constructing high-performing emotion recognition systems using speech.

Zhang et al. (2017) tackles the challenging problem of speech emotion recognition. It aims to bridge the semantic gap between low-level features and subjective emotions by leveraging deep convolutional neural networks (DCNNs). Inspired by the success of DCNNs in visual tasks like image classification, the authors propose using DCNNs to close the affective gap in speech signals. Their approach extracts log Mel-spectrograms as input, similar to RGB images, then applies the pretrained AlexNet DCNN model from ImageNet. The segment-level features are aggregated via discriminant temporal pyramid matching to generate utterance-level representations. The authors show promising results using the proposed DCNN model and aggregation strategy, noting the pre trained DCNN's effectiveness in extracting affective speech features. They also demonstrate significant performance gains by fine-tuning the DCNN on emotional speech datasets specifically. Overall, the paper highlights DCNNs' potential for speech emotion recognition by bridaging low-level features and subjective emotions.

Ouyang et al. (2017) details an approach for classifying basic emotions and neutral expressions in the Emotion Recognition in the Wild 2017 video-based challenge. Their proposed solution uses deep network transfer learning for feature extraction, spatial-temporal model fusion to leverage different networks' strengths, and semi-auto reinforcement learning to optimize the fusion strategy based on organizer feedback. These advanced techniques address real-world emotion recognition challenges. Their approach achieved 57.2% accuracy on the test set, exceeding the 40.47% baseline, demonstrating the effectiveness of their transfer learning, model fusion, and reinforcement learning techniques for complex audio-visual emotion recognition.

Ma et al. (2018) introduce a novel deep neural network approach that accepts variable-length speech sentences directly as input for emotion recognition. By combining Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), the model extracts emotional information from spectrograms, improving accuracy over traditional methods that segment sentences. This variable-length approach aims to mitigate degradation caused by segmentation. Experiments on the IEMOCAP dataset demonstrate the superiority of the proposed model, with improved performance in distinguishing neutral and non-neutral emotions. The paper highlights the growing use of raw, unprocessed data and deep learning for speech processing. It outlines challenges in emotion recognition and the importance of extracting relevant acoustic features. The model uses CNNs to capture emotional patterns in spectrograms and RNNs to model temporal structure across sentences. Analysis of results and activations provides insights into the network's ability to reduce confusion between neutral and emotional speech. Overall, the variable-length approach showcases deep learning's potential for emotion recognition from spectrograms of unsegmented speech.

The study by Daneshfar et al. (2019) presents an innovative Speech Emotion Recognition (SER) system that uses a holistic approach to improve emotion classification accuracy. It starts by creating a high-dimensional rich feature vector from both the speech and glottal-waveform signals using techniques such as MFCC, PLPC, MVDR, and incorporating prosodic features derived from fundamental frequency (f0) contours. The main innovation of the system is the use of a modified quantum-behaved particle swarm optimization (QPSO) method known as pQPSO. This approach optimizes both the feature-vector dimension reduction and the Gaussian Mixture Model (GMM) classifier parameters. Importantly, the updated QPSO is intended to operate

effectively within a narrow parameter range, allowing for real-time application. Table 3 illustrates the comparison between relevant research papers.

Table 3

Comparison of Relevant Research Papers

Author/Year	Goal	Approach	Targeted Problem	Conclusion
Puri et al.,2022	Hybrid CNN architecture	Combine LSTM, HMMs, DNNs with CNNs	Highlighting the need for robust, balanced datasets.	Efficacy of the hybrid CNN approach showcasing 98% accuracy
Toyoshima et al., 2023	Develop a model using Mel spectrograms (MelSpec) and (GeMAPS).	Integrating GeMAPS and focal loss in a multi-input model.	Address the issue of imbalanced emotional speech data.	Integrated model that merges MelSpec and GeMAPS
Pan et al., 2023	Multimodal Emotion Recognition	In-depth analysis of data preprocessing, feature extraction technique.	Comprehensive understanding of multimodal emotion recognition	Innovative approaches in data preprocessing, feature extraction, and modalities.

Author/Year	Goal	Approach	Targeted Problem	Conclusion
Chen et al., 2018	Enhance SER performance by proposing an innovative architecture	3-D ACRNN architecture that integrates CRNN	Exploring log Mel-spectrograms with calculated MFCCs, deltas, and delta-deltas.	Superior performance over baseline methods, in terms of unweighted average recall.
Trigeorgis et al., 2018	Deep Canonical Time Warping (DCTW) method to address limitations in multi-modal data.	Dynamic Time Warping (DTW) by incorporating Canonical Correlation Analysis and deep neural networks.	Address challenges in capturing complex non-linear representations for multi-modal data	Highlight the effectiveness of DCTW through experiments on multiple datasets.
Jain et al., 2020	Efficient system for SER with classifying speech into four emotions: sadness, anger, fear, and happiness.	Employ (SVM) algorithms with two strategies, One Against All (OAA) and Gender Dependent Classification.	Focus on achieving high accuracy in emotion classification.	Demonstrate the effectiveness of SVM for SER by achieving an overall accuracy of 85.085%.
Abbaschian et al., 2021	(SER) using deep learning techniques.	Analyse (CNNs), (RNNs), and (GANs), for SER.	Improving SER performance without the need for manual feature engineering.	Development of high-performance SER systems.

Author/Year	Goal	Approach	Targeted Problem	Conclusion
Amir et al., 2001	Compare two classification algorithms for SER.	Implement two classification algorithms—one based on Euclidean distances and the other on neural networks.	Focus on comparing the performance of Euclidean distance-based and neural network-based algorithms using a consistent feature set.	Highlighting distinct behaviors between the two algorithms in producing emotion indices.
Basu et al., 2017	In-depth review of emotion recognition using speech data	Examine various methodologies for collecting speech data, explore complexities, analyze noise reduction	Challenge to choose appropriate speech corpora, identify relevant features, and select suitable classification models.	Explored noise reduction techniques and their role in enhancing emotion detection accuracy.

Data and Project Management Plan

2.1 Data management plan

Data Collection Approaches

The research utilized publicly accessible open-source datasets - RAVDESS, CREMA-D, and SAVEE - which promoted transparency and enabled reproducibility of the study's methodology and results.

The RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song) dataset contains audio files in WAV format of vocalizations expressing various emotions through speech and song, performed by 24 professional actors (Livingstone & Russo, 2018) is illustrated in Figure 2.

Figure 2

Ravdess dataset

	filename	gender	emotion	labels	source	sampling_rate	duration	fundamental_frequency
0	03-01-08-02-02-01-01.wav	male	surprise	male_8	RAVDESS	48000	3.503500	439.870880
1	03-01-08-01-01-01-01.wav	male	surprise	male_8	RAVDESS	48000	3.403396	399.554810
2	03-01-05-01-02-01-01.wav	male	angry	male_5	RAVDESS	48000	3.503500	433.684662
3	03-01-06-01-02-02-01.wav	male	fear	male_6	RAVDESS	48000	3.703688	439.910400
4	03-01-06-02-01-02-01.wav	male	fear	male_6	RAVDESS	48000	4.237563	1149.584595
5	03-01-05-02-01-01-01.wav	male	angry	male_5	RAVDESS	48000	4.104104	400.742096
6	03-01-07-01-01-01-01.wav	male	disgust	male_7	RAVDESS	48000	3.870521	429.941345
7	03-01-04-01-01-02-01.wav	male	sad	male_4	RAVDESS	48000	3.570229	398.068268
8	03-01-04-02-02-02-01.wav	male	sad	male_4	RAVDESS	48000	3.737062	517.057861
9	03-01-07-02-02-01-01.wav	male	disgust	male_7	RAVDESS	48000	4.037375	256.554382

The detailed naming conventions of the audio files provide metadata indicating the actor, gender, emotion portrayed, speech content, intensity, statement/song, repetition, and recording session details. This supports systematic preprocessing. The emotions encompassed in the

dataset include calm, happy, sad, angry, fearful, surprise, disgust, and neutral states, enacted at two intensity levels - normal and strong. The RAVDESS dataset offers a substantial compilation of emotional speech and singing samples for developing deep learning algorithms to recognize emotions from speech. The multi-speaker audio samples with labeled emotion categories and intensity levels allow training robust models that can generalize across different speaking styles. The dataset was downloaded from Zenodo open repository (Livingstone & Russo, 2018).

RAVDESS Dataset Filename Interpretation. RAVDESS filenames, such as 02-01-06-01-02-01-12.mp4, are highly structured, with each segment conveying different details.

Modality (02). Indicates the type of recording (audio, visual, or audio-visual).

Vocal Channel (01). Specifies whether the recording is speech or song.

Emotion (06). Denotes the type of emotion expressed, like "Fearful".

Intensity (01). Indicates the emotional intensity, with a separate coding for "neutral".

Statement (02). Refers to the specific content or sentence spoken.

Repetition (01). Denotes whether it's the first or second repetition of the statement.

Actor (12). Identifies the actor's number, with odd and even numbers indicating male and female actors, respectively.

The CREMA-D (Crowd-Sourced Emotional Multimodal Actors Dataset) dataset provides 7,442 audio clips of 91 amateur actors vocalizing emotions intensely and naturally in diverse spoken languages (Cao et al., 2014). The audio data is stored as WAV files annotated with

metadata specifying actor, language, emotion, and other details. Along with audio, video clips are also included. This multi-language, multi-speaker dataset supports developing cross-cultural speech emotion recognition systems using both audio and visual feeds as input. The varied samples enable training machine learning models to identify emotions reliably across different speaker demographics. The CREMA-D dataset was downloaded from the dataset's GitHub repository (Cao et al., 2014) and can be viewed in Figure 3 post pre-processing.

Figure 3

Crema-D Dataset

	Emotion	Filename	Sampling_Rate	Duration	Fundamental_Frequency
0	angry	1022_ITS_ANG_XX.wav	16000	2.435750	531.199768
1	angry	1037_ITS_ANG_XX.wav	16000	3.003000	260.872375
2	neutral	1060_ITS_NEU_XX.wav	16000	2.402375	380.431946
3	neutral	1075_ITS_NEU_XX.wav	16000	2.435750	263.769684
4	disgust	1073_IOM_DIS_XX.wav	16000	2.869562	533.592163
5	disgust	1066_IOM_DIS_XX.wav	16000	2.502500	454.303833
6	sad	1078_IWL_SAD_XX.wav	16000	2.435750	325.591675
7	fear	1029_TAI_FEA_XX.wav	16000	3.670313	368.304260
8	sad	1039_IEO_SAD_MD.wav	16000	2.202187	450.292328
9	happy	1008_TAI_HAP_XX.wav	16000	3.203187	568.321899

Crema-D Dataset Filename Interpretation. In the Crema-D dataset, filenames like 1001_DFA_ANG_XX.wav provided specific insights into the recording. Each segment of the filename offers different information

Actor Identifier (1001). Indicates the specific actor or participant who recorded the audio. This is key for identifying individual differences in emotional expression.

Session or Script Code (DFA). May refer to the particular script or recording session. This part is less standardized and might be specific to the dataset's internal organization.

Emotion (ANG). Stands for the emotional tone of the recording, in this case, "Anger". The Crema-D dataset categorizes emotions clearly in its filenames.

Take or Variation (XX). Could denote different takes or variations in the emotional expression or recording conditions.

The SAVEE (Surrey Audio-Visual Expressed Emotion) dataset contains recordings of 4 male actors voicing neutral and emotional sentences in British English . The audio data is provided as WAV files labeled with the actor identity and portrayed emotion. Video clips are also included, aligned with the audio. During dataset creation, rigorous subjective evaluations were conducted, achieving speaker-independent emotion recognition rates of 61% (audio), 65% (visual) and 84% (audio-visual). This validated dataset with audio, video, and labels was downloaded from the SAVEE website hosted by the University of Surrey and can be observed in Figure 4 post the preprocessing phase.

Figure 4

Savee dataset

	Emotions	Filename	Sampling_Rate	Duration	Fundamental_Frequency
0	sad	JK_sa01.wav	44100	4.511837	521.017761
1	sad	JK_sa15.wav	44100	6.058707	478.932922
2	neutral	DC_n13.wav	44100	2.788889	432.591309
3	surprise	DC_su09.wav	44100	3.433968	460.721375
4	neutral	DC_n07.wav	44100	4.051769	453.889893
5	neutral	JK_n20.wav	44100	3.328912	476.005829
6	neutral	JK_n08.wav	44100	3.544331	508.790924
7	sad	JE_sa08.wav	44100	4.133265	364.639862
8	fear	JK_f15.wav	44100	5.911950	576.237549
9	fear	JK_f01.wav	44100	4.140454	569.393311

SAVEE Dataset Filename Interpretation. In the SAVEE dataset, filenames such as DC_d03.wav encode information about the speaker, the expressed emotion, and the sentence Number:

Speaker Identifier (DC). Refers to one of the four male actors (DC, JE, JK, KL) in the dataset. Each actor's recordings provide unique vocal characteristics.

Emotion (d). Indicates the emotion being expressed, here "Disgust". The SAVEE dataset uses specific letters to denote different emotions.

Sentence Number (03). Represents the sequence of the sentence that was recorded,providing information about the script used in the expression of the emotion.

The summary of key features of three datasets are explained in Table 4.

Table 4

Summary of Key Features of Datasets

Dataset	Modality	Emotions	Number of Actors	Total Clips
RAVDESS	Audio	Calm, Happy, Sad, Angry, Fearful, Surprise, Disgust, Neutral	24	1440
CREMA-D	Audio-Visual	Anger, Disgust, Fear, Happy, Neutral, Sad	91	7442
SAVEE	Audio-Visual	Anger, Disgust, Fear, Happy, Neutral, Sad, Surprise	4	480

Data Management Methods

Throughout the project's lifecycle, a comprehensive approach to data management was implemented, emphasizing ethical standards and compliance with legal requirements. The RAVDESS, CREMA-D, and SAVEE datasets were utilized as foundational elements and stored separately in Google Cloud Platform buckets (Cao et al., 2014, Livingstone & Russo, 2018). Each dataset was equipped with distinct access control measures and encryption protocols to ensure strong data security and privacy. This storage approach not only demonstrated the project's dedication to transparency and reproducibility but also facilitated efficient preprocessing activities such as data cleaning, feature extraction, and normalization using advanced GPUs within the Google Colab environment.

In accordance with ethical standards and legal compliance, the project developed strong processes to manage the entire data lifecycle. Throughout the phases of acquiring, storing, and using data, close attention was paid to copyright and Intellectual Property Rights (IP/IPR) regulations. This involved thorough documentation of dataset origins, adherence to licensing agreements, and clear communication of usage rights. Team members responsible for metadata handling and data management played a vital role in ensuring compliance with ethical norms.

To maintain the quality of the collected data, the project established a thorough four-phase process for managing data. Team members were assigned specific tasks related to obtaining, handling metadata, storing, and integrating datasets. A detailed analysis of metadata was crucial in ensuring a deep understanding of data structure and format while adhering strictly to legal and ethical standards. Additionally, implementing a 5-fold cross-validation approach emphasized the dedication to achieving high accuracy in machine learning tasks by combining CRISP-DM methodology (Schröer, Kruse, & Marx Gómez, 2021) with Agile Scrum practices

for an efficient development cycle (Cao et al., 2014; Haq & Jackson, 2010; Livingstone & Russo, 2018).

The ownership of the data was closely linked to the open-source nature of the datasets, which were obtained from well-established repositories such as Zenodo and GitHub. Generally, the original contributors or institutions maintaining these repositories maintained ownership rights. It is crucial to note that this project did not entail gathering new personal or sensitive data; instead, it centered on existing datasets that had been carefully assessed for emotion recognition. Adhering strictly to legal and ethical guidelines while handling metadata also strengthened the responsible use of data in the project.

The research prioritized future data interpretability by focusing on complete documentation and metadata, according to detailed file name guidelines provided in the dataset description. A robust storage and backup plan was created while navigating through several stages of the data lifecycle, including raw to preprocessed data employing datasets such as RAVDESS, SAVEE, and CREMA-D. This entailed utilizing Google Cloud Platform buckets to assure the data's integrity and availability. The research attempted to ensure continuous usability and reliability during the project's entire lifecycle by combining thorough documentation and strategic storage strategies.

An integrated approach was employed by combining the CRISP-DM (Schröer, Kruse, & Marx Gómez, 2021) framework with Agile Scrum practices, ensuring a productive and well-organized development cycle. The development process unfolded in 14-day Sprints organized within broader Stories and Epics to facilitate efficient monitoring and successful completion of crucial project phases. Specifically, the Data Management phase was systematically divided into four primary stages, with each stage assigned to dedicated team

members which can be illustrated in Table 5. These tasks included obtaining datasets aligned with research objectives, carefully analyzing and documenting dataset metadata which is stored in GCP buckets, overseeing data access, and ensuring strict adherence to legal and ethical standards. A designated coordinator played a pivotal role in supervising the entire data management process as a liaison for alignment with project objectives and smooth progress.

Table 5

Details of the Data Management Stage

Data Management Phase	Team Member	Responsibilities
Data Acquisition	Rohith Reddy Vangala	Dedicated to sourcing and acquiring datasets suited for the project, ensuring alignment with research goals. Manages data access, focusing on ease of download and integration with Google Colab.
Metadata Handling	Shashank Reddy Kandimalla	Tasked with analyzing and documenting dataset metadata. Ensures understanding of data structure, format, and compliance with legal and ethical guidelines, facilitating usage in cloud environments.

Data Management Phase	Team Member	Responsibilities
Data Storage	Yamini Muthyala	Oversees the efficient and secure storage of datasets, leveraging cloud storage options, particularly Google Cloud Platform (GCP) for scalable data management. Manages data backup strategies.
Data Integration	Swati	Coordinates the integration and compatibility of various datasets in cloud-based platforms, focusing on seamless processing and modeling within Google Colab's collaborative environment.

The project team demonstrated proficient data management by effectively handling various tasks. This involved thorough examination of metadata, detailed documentation of data structure and format, secure storage of datasets in separate Google Cloud Platform buckets, and careful supervision of data processing within Google Colab. To ensure a secure environment, access control measures were put in place for the datasets. As the project progressed, the previously distinct datasets were smoothly combined into a unified dataset surpassing 3GB. Considering the open-source nature of the datasets prompted thoughts about depositing them in public repositories. Essential resources crucial to the success of this endeavor included cloud storage in GCP, advanced GPUs for preprocessing tasks, and collaborative platforms like Google Colab throughout the development process. This project exemplified a comprehensive and

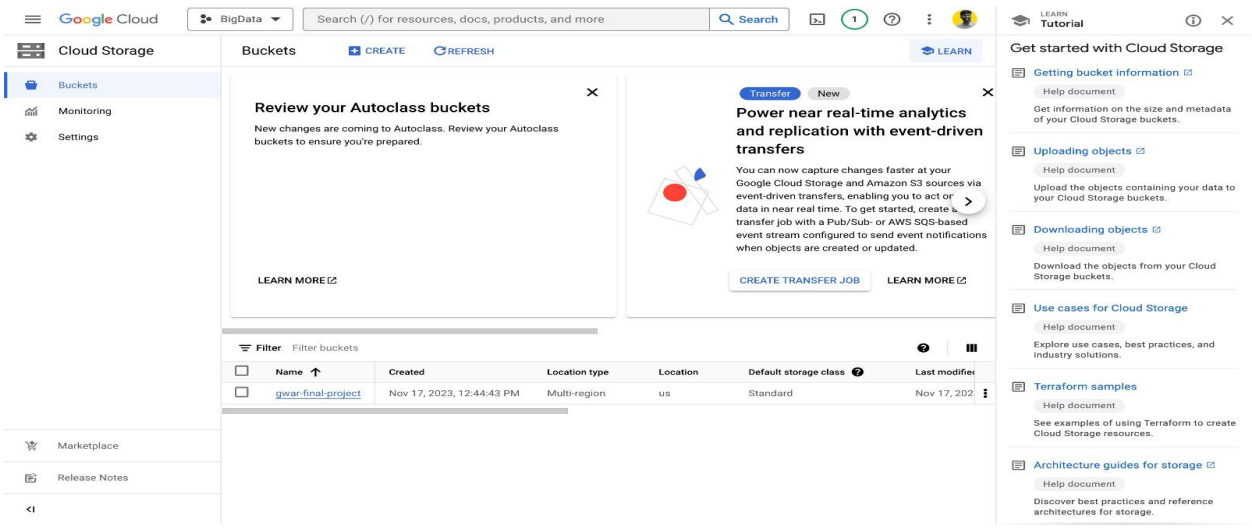
transparent approach to data management that ensured effectiveness and consistent adherence across its entire lifecycle.

Data Storage Methods

Throughout the project's lifespan, a thorough strategy for data storage was executed, in line with ethical standards, legal requirements, and the overall research objectives (Cao et al., 2014; Haq & Jackson, 2010 ; Livingstone & Russo, 2018). The RAVDESS, CREMA-D, and SAVEE datasets were used as foundational elements and were stored separately in dedicated Google Cloud Platform buckets and can be seen in Figure 5.

Figure 5

Google Cloud Bucket



This segregation of datasets within individual GCP buckets ensured unique access control measures and encryption protocols to prioritize data security and privacy.The isolation of each dataset not only showcased a commitment to transparency and reproducibility but also set the stage for efficient preprocessing activities like data cleaning, feature extraction,and normalization utilizing advanced GPUs within the Google Colab environment (Bisong, 2019).

As the project progressed, the individual datasets were seamlessly combined into one comprehensive dataset of over 3GB in size. This consolidation was essential for transitioning from separate storage units to a unified dataset suitable for machine learning applications. The storage methods used, along with advanced preprocessing capabilities, played a crucial role in ensuring the overall success of the project. Additionally, given that the datasets are open-source, there was consideration of depositing the unified dataset into public repositories as part of commitment to transparency and collaboration.

Efficient storage on the GCP played a crucial role in the data storage techniques, providing a secure and accessible environment for datasets. The use of cloud storage supported collaborative preprocessing activities within Google Colab, emphasizing effectiveness and security in managing large-scale audio datasets. The strategic arrangement of folders within GCP buckets contributed to a well-organized structure, potentially grouped by dataset name or project phase. Each folder contained files related to specific tasks or stages, ensuring clarity and accessibility in data organization. Importantly, during the merging process, a ‘Path’ column was incorporated into the dataset to store file paths leading to the original audio data. This deliberate inclusion allowed direct access to raw audio data, streamlining subsequent analysis phases while maintaining a connection between processed and raw data (Cao et al., 2014; Haq & Jackson, 2010; Livingstone & Russo, 2018).

In conclusion, the project's data storage methods showcased a strategic blend of secure cloud storage, thorough organization, and open-source collaboration considerations. These procedures not only guaranteed the integrity and accessibility of the datasets but also facilitated the smooth shift from individual storage units to a unified dataset prepared for advanced deep

learning activities. Adhering to ethical standards and open-data principles was fundamental to these methods, fostering transparency and cooperation within the research community.

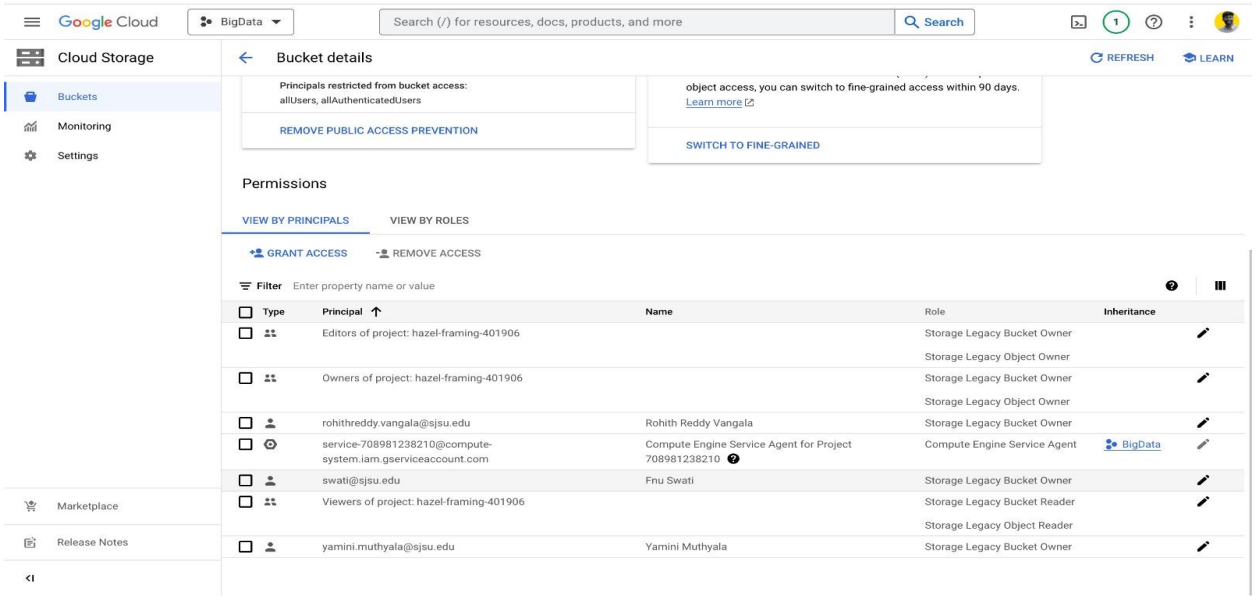
Data Usage Mechanisms

Selection and Preservation. The project prioritizes the meticulous choice and conservation of datasets. Following data collection, the obtained datasets are securely stored in a Google Cloud bucket. This decision is based on scalability, security, and accessibility considerations. Google Cloud storage is selected for its capacity to efficiently handle large datasets, ensuring scalability tailored to the project's requirements. This selection aligns with preservation best practices by offering a trustworthy and resilient infrastructure to protect the integrity and longevity of the gathered data.

Data Sharing. The regulation of entry to Google Cloud storage is a crucial part of the data sharing process. Approved team members have unique access to the cloud storage, guaranteeing a secure and limited environment which is illustrated in Figure 6.

Figure 6

Access Privileges

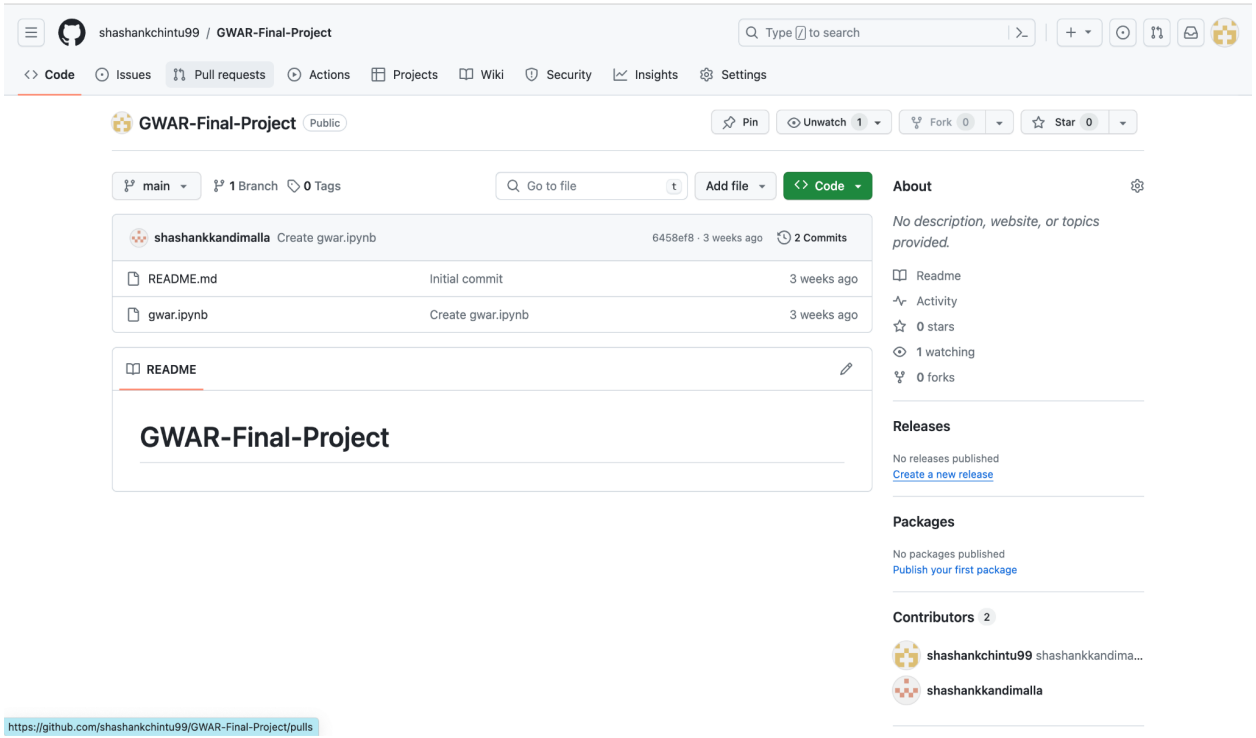


Stringent regulations exist to uphold data integrity and security, which contributes to responsible practices in sharing data. This controlled entry not only protects the information but also enables remote and collaborative work. Team members can simultaneously access and analyze the data, fostering effective collaboration while upholding data security.

Responsibilities and Resources.The project takes a methodical approach by assigning specific team members responsibilities for different data management phases. This organizational tactic guarantees a structured and systematic process for managing the data. While Google Cloud Platform is crucial for scalable data storage and management, GitHub supports the team's work by providing a centralized platform for version control and collaborative coding which can be seen in Figure 7.

Figure 7

GitHub



The use of advanced GPUs in Google Colab demonstrates the commitment to allocating appropriate resources for efficient data processing and there is no need for virtual machine instances. Integrating GitHub with cloud resources represents a comprehensive strategy that combines version control and collaborative development tools with strong data management practices.

2.2 Project Development Methodology

Analytics of Data with Intelligent Systems Development Cycle

This project utilizes a hybrid development methodology that strategically integrates the structured Cross-Industry Standard Process for Data Mining (CRISP-DM) (Schröer, Kruse, & Marx Gómez, 2021)with the adaptive Agile Scrum framework. Development progresses through a meticulous 14-day Sprint cycle seamlessly woven into larger Stories and Epics that offer overarching direction. This fusion establishes an iterative yet systematic process, leveraging Scrum's continuous feedback-driven approach to enable responsiveness alongside CRISP-DM's structured organization of analytics and intelligent systems cycles (Cao et al., 2014; Livingstone & Russo, 2018). The project's development backbone thereby balances efficient tracking of completion status with the flexibility to adjust based on emerging requirements. This unique methodology integration lends robustness and adaptability, ensuring timely delivery of project goals and high-quality outputs that align with evolving needs (Sutherland & Schwaber, 2021).

The structured development cycle, which combines CRISP-DM and Agile Scrum (Sutherland & Schwaber, 2021), offers a comprehensive framework for navigating the complexities of data analytics. The focus on 14-day Sprints creates an agile and flexible development environment, enabling the project team to effectively address challenges and integrate feedback promptly. This approach is crucial not only for achieving technical milestones

but also for aligning the project with broader goals. By incorporating Scrum practices, transparency, collaboration, and accountability among team members are ensured, creating an environment conducive to innovation and problem-solving. Overall, this analytics-driven intelligent systems development cycle follows CRISP-DM and Agile Scrum practices to position the project for success in meeting its objectives and delivering impactful outcomes.

Planned development processes and activities

CRISP-DM Consists of Six Key Phases. CRISP-DM is made up of six important phases that provide an organized approach to data mining and data science initiatives. The phases are as follows: Business Understanding for determining project goals, Data Understanding for exploring and assessing available data, Data Preparation for processing and formatting data for modeling, Modeling for applying data mining algorithms and techniques, Evaluation of model results and performance metrics, and finally Deployment for integrating the final analytical models into business processes and applications for organizational impact. Through methodical planning, development, and deployment centered on end goals, this phased technique encourages effectively transforming business requirements into data-driven solutions.

Business Understanding. This emotion recognition research project aimed to demonstrate the potential of artificial intelligence in interpreting human expression nuances from voice data. The main goals included using multiple datasets to train deep neural networks for accurately categorizing emotional states from audio inputs. Potential applications ranged from enhancing interactive voice agents with emotional awareness to providing scalable tools for evaluating mental health through speech pattern analysis. Success metrics highlighted high accuracy in multi-class emotion classification as well as feasible real-time performance. The trained models were intended for integration within conversational interfaces and telehealth

platforms utilized by major industry participants. Competitive benchmarking analysis verified that combining extensive emotional speech datasets with cloud-based GPU infrastructure for advanced deep learning was an effective approach.

Data Understanding.This phase was crucial and involved a careful analysis of three different datasets—CREMA-D, SAVEE, and RAVDESS. These datasets each provided unique insights into human emotions, contributing to a comprehensive understanding of variations in emotional expression, intensity, and linguistic accents (Cao et al., 2014; Haq & Jackson, 2010; Livingstone & Russo, 2018) The focus of this phase was on meticulously scrutinizing the datasets to reveal their intrinsic characteristics, which guided subsequent project steps. An extensive analysis considered recording quality, emotional range, and potential biases inherent in the datasets. This thorough understanding enabled the research team to make well-informed decisions during data preparation and modeling stages, establishing a strong foundation for developing an advanced emotion recognition system.

Data Preparation.Several tasks were performed to transform the collected raw datasets into a polished collection suitable for training effective deep learning models. Following recommended guidelines, the various datasets were organized into a consistent structure while preserving links to original media files for tracking purposes. After ensuring integrity and formats as emphasized in CRISP-DM (Schröer, Kruse, & Marx Gómez, 2021), specific preprocessing and refining methods were applied, including noise reduction, normalization, and augmentation - all crucial for uncovering important patterns. Another essential step involved extracting informative audio features presented in visualizations that captured emotional cues such as pitch, tempo, and intensity to be used for subsequent modeling. Moreover,a strategic partitioning of samples was performed for training validation,and testing,in accordance with

CRISP-DM directives,to rigorously evaluate model reliability.The methodical approach taken during the data preparation phase laid a strong foundation for impactful modeling by creating an enhanced dataset focused on only the most influential characteristics of emotional speech.

Modeling. The modeling phase of the project is where theoretical concepts and data preparation converge into practical application, harnessing the power of various deep learning architectures. This phase commences with the design and development of Convolutional Neural Networks (CNNs)(Aloysius & Geetha, 2017), which are adept at extracting spatial features from spectrograms. The spectrograms, with their rich visual cues about frequency and amplitude changes over time, provide a fertile ground for CNNs to identify patterns indicative of different emotional states. Then, the project taps into the potential of Dense Neural Networks (DNNs)(Kaushik et al., 2022) and Long Short-Term Memory networks (LSTMs). These models excel in interpreting time-series data, capturing the temporal dynamics and nuances of speech that convey emotional undertones. Their ability to remember and learn from long sequences of data makes them particularly suited for audio where the context and progression of sound waves play a crucial role in emotion recognition.

Another key component of the modeling phase is the use of Autoencoders. By learning to encode input data into a compressed representation and then reconstructing it back to the original format, Autoencoders are instrumental in dimensionality reduction and feature extraction. This process helps in distilling the essence of the audio data, reducing noise, and focusing on the most salient features necessary for emotion classification.

Throughout this phase, the models are trained, tested, and tuned using Google Colab's cloud-based GPUs, which provide the necessary computational power for handling large datasets and complex neural network architectures. Hyperparameter tuning is a critical aspect of this

phase, involving iterative experimentation with various configurations of learning rates, layer sizes, and regularization techniques to find the most effective model settings. The use of tools like Keras Tuner or Hyperopt in this environment facilitates systematic exploration of parameter spaces, ensuring that the models are not just fitting well to the training data, but are also generalizable and efficient.

Evaluation.The project reorients focus towards the initial business goals, critically assessing whether the developed analytics solutions fulfill the intended real-world objectives. This phase employs robust techniques like k-fold cross-validation to evaluate model performance, but more importantly, examines if the accuracy, efficiency and capabilities of the models adequately address target applications and use cases (Cao et al., 2014).

Utilizing Agile Scrum practices, the project incorporates continuous feedback loops for course corrections, adaptability, and accountability to business outcomes (Sutherland & Schwaber, 2021). Performance is judged not merely through metrics, but also by the degree of alignment to dynamic industry needs.

The evaluation phase bridges theoretical model development with practical deployment - providing the analytical basis for selections and iterations aligned to success criteria. By employing Agile techniques that orient progress to business priorities, this stage focuses intelligence solutions to where accuracy and real-world impact intersect . The phase culminates with models transformed into user-centric tools - evaluated and refined to better propel competitive advantage.

Key performance metrics such as accuracy, precision, recall, F1 score, confusion matrix and the Receiver Operating Characteristic (ROC) curve are meticulously calculated for each fold. Accuracy provides a general sense of the model's overall correctness, while precision and

recall offer insights into its ability to correctly identify and capture all relevant instances of emotions.

The F1 score, as a harmonic means of precision and recall, becomes critical in scenarios where the balance between false positives and false negatives is crucial. The ROC curve and the associated Area Under the Curve (AUC) provide an aggregate measure of performance across all possible classification thresholds, highlighting the model's ability to distinguish between different emotional states.

This rigorous evaluation process, supported by the computational capabilities of GCP, is crucial not just for selecting the best-performing model but also for understanding and improving the models' behaviors under different conditions. The insights gained from this phase guide further refinements in the modeling phase, ensuring that the final deployed models are not only accurate and efficient but also robust and reliable in various real-world applications. The evaluation phase thus acts as a bridge between the theoretical development of models and their practical, impactful deployment.

Deployment. The deployment phase marks the culmination of the project, where the refined and tested models are transformed into practical tools for end-users. A key aspect of this phase is packaging the models into production-ready services that can be easily integrated into target applications.

Another crucial element of this phase is the use of GitHub for managing and sharing the project's codebase, documentation, and other relevant materials. GitHub provides a collaborative platform that supports version control, issue tracking, and code review, ensuring that the project remains organized, up-to-date, and open for future enhancements or contributions from the community. The repository includes detailed documentation on the project's methodology, data

preprocessing steps, model architectures, and evaluation results, making it a comprehensive resource for anyone interested in the field of emotion recognition or audio processing. By hosting the project on GitHub, it also becomes accessible to a wider audience, ranging from academic researchers to industry practitioners, promoting transparency, reproducibility, and continuous learning.

2.3 Project Organization Plan

The integration of the Cross-Industry Standard Process for Data Mining (CRISP-DM) methodology with Agile principles forms the foundation of approach in managing the speech emotion recognition project (Sutherland & Schwaber, 2021). This integration provides a structured yet flexible framework for handling the complexities of data mining tasks. CRISP-DM's systematic phases, ranging from business understanding to deployment, offer a robust structure for the project. Concurrently, Agile principles, characterized by iterative and incremental processes, inject adaptability into the strategy. This amalgamation creates a methodical and responsive approach, crucial in the dynamic landscape of speech emotion recognition. Agile's collaborative nature complements CRISP-DM's systematic process, fostering a balanced, responsive, and efficient project execution.

In the initial phase of Business Understanding, we define the project's requirements and objectives. This stage lays the groundwork for meticulous audio data processing and serves as the cornerstone for subsequent phases. Moving into Data Understanding, raw audio data is transformed into meaningful formats such as time-series signals, Mel-Frequency Cepstral Coefficient (MFCC) features, and spectrogram images. Tools like libROSA facilitate this transformation, emphasizing the need to understand these data forms and their impact on the recognition process.

During Data Preparation, audio data undergoes preprocessing and transformation, utilizing Google Cloud Platform (GCP) buckets for secure and scalable storage. Extract, transform, load (ETL) processes ensure data normalization, essential for consistent model training.

The Modeling phase employs various neural network architectures, including Convolutional Neural Networks (CNNs) for feature extraction from spectrograms and Dense Neural Networks (DNNs) or Long Short-Term Memory networks (LSTMs) for managing time-series data. Training utilizes established datasets like RAVDESS, SAVEE, and CREMA-D, emphasizing optimization techniques to enhance model performance.

Following training, the models undergo rigorous evaluation employing metrics such as accuracy, F1 score, and confusion matrices on separate validation datasets. This evaluation is pivotal for assessing performance and ensuring model integrity. In the final phase of deployment, robust error handling and logging mechanisms are implemented. This infrastructure is crucial for detecting and recording discrepancies, enabling in-depth post hoc analysis to troubleshoot and optimize the system. The deployment phase ensures the long-term effectiveness and efficiency of the speech emotion recognition system, supported by continuous monitoring and maintenance.

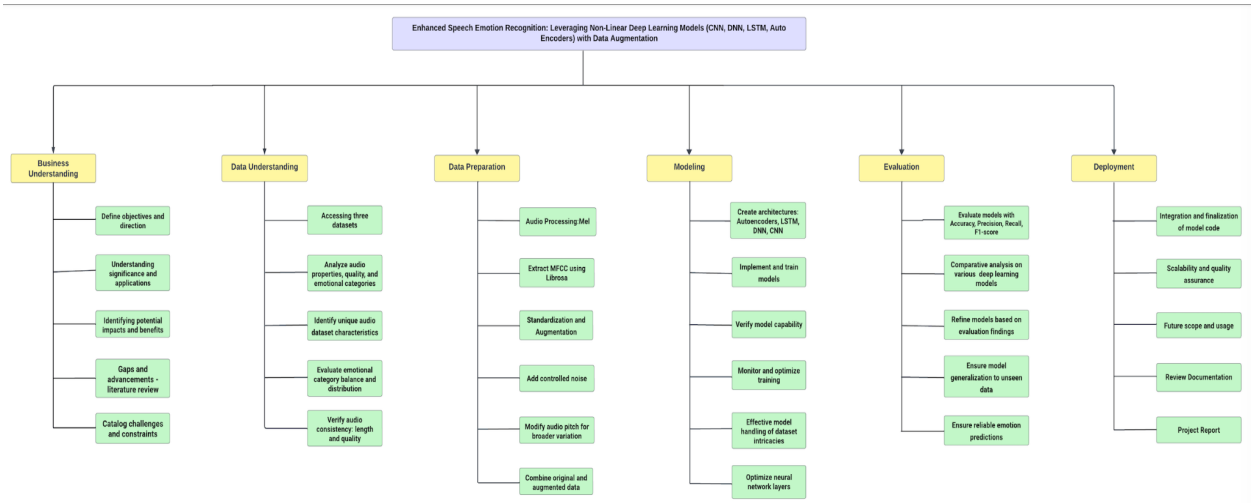
Work Breakdown Structure

The Work Breakdown Structure (WBS) for this research initiative provided a structured overview of the project's stages, each encapsulating specific work packages and deliverables. The systematic execution of the research initiative unfolded through distinct phases, starting with the Business Understanding epic. In this phase, the project's primary aim was to develop a sophisticated speech emotion recognition system, utilizing complex neural network architectures such as CNNs, DNNs, LSTMs, and potentially autoencoders (Hinton & Salakhutdinov, 2006).

Key tasks included defining requirements, focusing on meticulous audio data processing, and laying the foundation for subsequent phases. The Work Breakdown Structure can be illustrated in Figure 8.

Figure 8

Work Breakdown Structure



Transitioning to the Data Understanding phase, the system converted raw audio data into meaningful formats such as time-series signals, MFCC features, and spectrogram images(McFee et al., 2015).Tools like libROSA were essential in enabling this transformation and highlighting the vital understanding of these data forms and their influence on the recognition process. The related tasks included examining dataset structures (RAVDESS, SAVEE, and CREMA-D), evaluating audio properties, and maintaining recording consistency.

Audio data was preprocessed and transformed as part of the Data Preparation epic, with Google Cloud Platform (GCP) buckets used for safe and scalable storage (Qian et al., 2009). During this phase, extract, transform, and load (ETL) techniques were used to ensure data normalization for consistent model training. Key audio processing ideas, feature extraction using

libROSA (McFee et al., 2015), data cleaning, augmentation, and the synthesis of realistic spectrograms were all part of the work packages.

Diverse neural network models were deployed as part of the Modeling epic, including CNNs , DNN, Autoencoders and LSTMs (Greff et al., 2017). These models were trained using existing datasets such as RAVDESS, SAVEE, and CREMA-D, with an emphasis on optimization utilizing sophisticated techniques. The linked job items included developing architectures, implementing and training models, monitoring the training process, and optimizing for spatial and sequential data.

The Evaluation and Optimization phase guaranteed that the performance of trained models was thoroughly assessed. Tasks included testing models against validation and test datasets, pinpointing areas for enhancement, and ensuring dependable emotion predictions in various scenarios. The focus was on evaluating the overall system's ability to correctly recognize emotions from speech. Additional testing was conducted to guarantee real-world reliability.

The Deployment phase was implemented by strong error handling and logging systems critical for detecting and documenting inconsistencies. This phase emphasized continual monitoring and maintenance to ensure the speech emotion recognition system's long-term effectiveness and efficiency. Work packages included merging code and model components, documenting execution stages, using GitHub for version control, and providing comprehensive documentation for future development and use.

2.4 Project Resource Requirements and Plan

The speech emotion recognition (SER) project required robust hardware infrastructure and sophisticated software tools to facilitate a seamless workflow. Local machines equipped with

high-end GPUs served as the computational backbone for training complex neural network models, especially those involving convolutional and recurrent architectures (Sarma et al., 2018). Primary storage, managed through Google Cloud Platform (GCP), ensured secure and scalable storage for both raw audio files and the generated data (Qian et al., 2010). Python 3.9 served as the foundational programming language, complemented by deep learning libraries such as TensorFlow and PyTorch for advanced model development (Abadi et al., 2016; Paszke et al., 2019).

Specialized audio processing libraries like Librosa and PyDub were leveraged for effective feature extraction, enhancing the precision of the emotion recognition system (McFee et al., 2015). Additionally, collaborative coding was facilitated through the utilization of Jupyter Notebooks via Google Colab, enabling efficient experimentation and model development (Bisong, 2019).

The project's management and communication aspects were streamlined using tools like JIRA, GitHub, and Google Meet, fostering seamless collaboration among team members (Rausch et al., 2021). The meticulously justified cost estimation covered essential hardware components, cloud services, and software licenses, ensuring a balanced allocation of resources essential for the project's successful execution.

Hardware Requirements

The successful execution of a speech emotion recognition project necessitated a carefully curated blend of robust hardware, cutting-edge software, and efficient collaboration tools (Lee & Tashev, 2015). Table 6 outlined the critical hardware prerequisites, including a powerful local machine equipped with a minimum of 16 CPUs, NVIDIA GeForce RTX 40 series GPUs, and 64 GB RAM. These specifications were essential, particularly when developing complex deep

learning models, especially those involving convolutional and recurrent architectures (Sarma et al., 2018). Ensuring computational prowess was pivotal for the effective training of models, guaranteeing accurate and nuanced speech emotion recognition.

Table 6

Hardware and Specifications

Hardware	Specifications	Cost
Local Machine	Minimum 16 CPUs, NVIDIA GeForce RTX 40 series GPUs, 64 GB RAM	$2000
Total		$2000

Software Requirements

The speech emotion recognition project necessitated specialized software libraries illustrated in Table 7 tailored to deep learning tasks. Python 3.9 served as the fundamental programming language, providing a versatile and accessible coding environment (Reback et al., 2022). Deep learning models, such as DNNs, CNNs, LSTMs, and autoencoders, were constructed using TensorFlow and PyTorch, which provide advanced frameworks for neural networks (Abadi et al., 2016; Paszke et al., 2019). The Keras interface integrated these two libraries, enabling intricate neural network architectures for speech emotion recognition.

Audio processing tasks were executed utilizing Librosa for feature extraction and PyDub for precise audio manipulation, critical for representing speech data (McFee et al., 2015). Pandas and NumPy enabled efficient data handling while Matplotlib and Seaborn facilitated visualization (Harris et al., 2020).

Google Colab offered a collaborative platform for development, experimentation and model building (Bisong, 2019). This tailored suite of libraries and tools ensured an efficient approach to speech emotion recognition.

Table 7

Recommended Softwares

Software Category	Required Libraries and Versions	Cost
Programming Language	Python 3.9	$0.00
Deep Learning Libraries	TensorFlow 2.14.0 or higher, PyTorch 2.1.0 or higher	$0.00
Neural Network Architectures	Keras (with TensorFlow backend version 2.6.0)	$0.00
Data Augmentation Libraries	TensorFlow-GAN 2.0.0, PyTorch-GAN 0.0.3	$0.00
Audio Processing Libraries	Librosa 0.8.1, PyDub 0.25.1	$0.00
Data Handling Libraries	Pandas 1.3.3, NumPy 1.21.2	$0.00
Visualization Libraries	Matplotlib 3.4.2, Seaborn 0.11.2	$0.00
Development Environment	Jupyter Notebooks via Google Colab (Pro or Pro+ subscription for enhanced resources	$0.00
Total		$0.00

Tools and Licenses

Table 8 summarized the essential tools and licenses crucial for seamless project management and collaboration within the project (Rausch et al., 2021). JIRA, a free and user-friendly project management platform, provided robust task tracking capabilities, serving as the primary tool. GitHub, also free, enabled version control for efficient code management and collaborative development (Rausch et al., 2021).

Google Meet, an accessible free platform, facilitated real-time communication and collaboration within the team (Bisong, 2019). For documentation purposes, a student subscription of Microsoft 365 and free Google Docs were utilized, allowing comprehensive records and communication channels (Bisong, 2019).

This carefully selected set of technologies ensured effective coordination, control, teamwork, and documentation, aligning with project needs while enabling communication (Rausch et al., 2021). Table 8 comprises tools and their purposes.

Table 8

List of the Tools and its Purposes

Tool	Configuration	Cost
Project Management	JIRA (free tier)	$0.00
Version Control	GitHub (free tier)	$0.00
Collaboration	Google Meet (free)	$0.00
Documentation	Microsoft 365 (Student subscription), Google Docs (free)	$0.00
Total		$0.00

Project Cost Estimation and Justification

A substantial portion of the project's budget, totaling $2,000, has been allocated to a local workstation. This investment covers both hardware and software components necessary for the successful implementation of the speech emotion recognition project. The local workstation acted as the computational foundation, featuring at least 16 CPUs, NVIDIA GeForce RTX 40 series GPUs, and 64 GB RAM. This powerful hardware is crucial for managing intricate tasks, particularly in relation to advanced models such as DNNs, CNNs and Autoencoders. The workstation provided significant computational ability that contributes to precise model training and dependable outcomes.

The budget allocated around $240 for cloud services over three months. These charges included the use of Google Colab Pro as well as additional cloud storage. Although the specific split of services was not disclosed, it included resources necessary for model creation and deployment. This incorporated the flexibility and scalability provided by cloud services to fulfill the project's developing needs. It's worth mentioning that the project stored both raw audio recordings and generated data in Google Cloud Platform (GCP) storage buckets, which were secure and scalable.

The $2,000 designated for the local workstation did not necessarily need to be split among team members, as it was likely an individual expense covering a single powerful machine. In contrast, costs associated with cloud services could be distributed among team members based on their usage and requirements.

In summary, the budget allocation for the local workstation covered both hardware and software components, emphasizing its importance in handling complex computational tasks. The cloud services budget included expenses for Google Colab Pro and additional storage, supporting

the project's flexibility and scalability. The specific services and their allocation among team members could be further detailed based on individual needs and usage patterns can be viewed in Table 9.

Table 9

Cost Estimation

Type	Cost Category	Estimated Cost
Software	Cloud Services	Approximately $80/month x 3 months = $240
Software	Project Management Tools	Mostly free (with optional paid upgrades if needed)
	Total	$240

2.5 Project Schedule

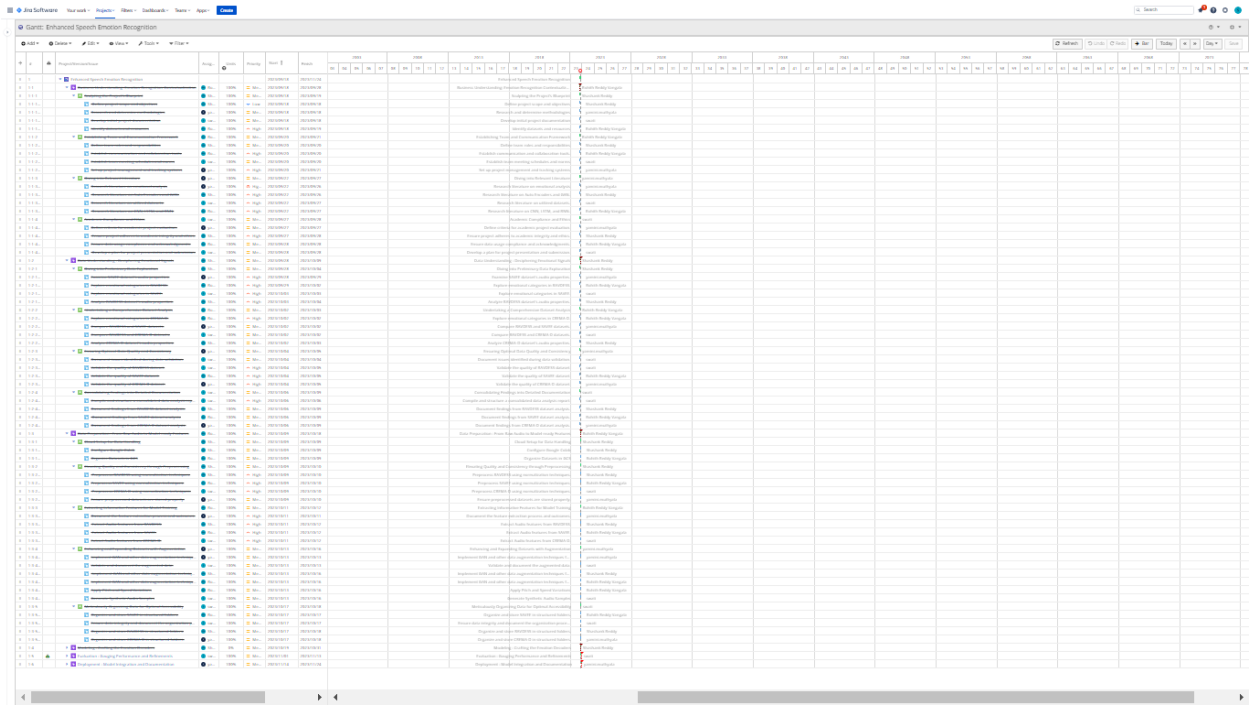
Gantt Chart

A Gantt chart was a popular and commonly used project management tool that visually depicted a project timetable. It was named after its creator, Henry Gantt (Geraldi & Lechter, 2012). The vertical axis of these charts was used to depict jobs or activities, and the horizontal axis was used to display a time scale. The graphic depicted the start and ending dates of various project aspects, allowing for a quick view of the entire project timeline, task dependencies, and progress.

Figure 9 illustrates the Gantt Chart of this study

Figure 9

The Entire Gantt chart



Each task was represented by a horizontal bar indicating its duration, schedule, and location within the project timeline.Charts could assist in optimizing schedules by showing overlaps or connections between jobs by visualizing them on a shared timeline.

By delegating responsibilities to specific individuals, they increased responsibility. Overall, Gantt charts were crucial for successful project planning and management due to their clear, visual representation of scheduled tasks over time.

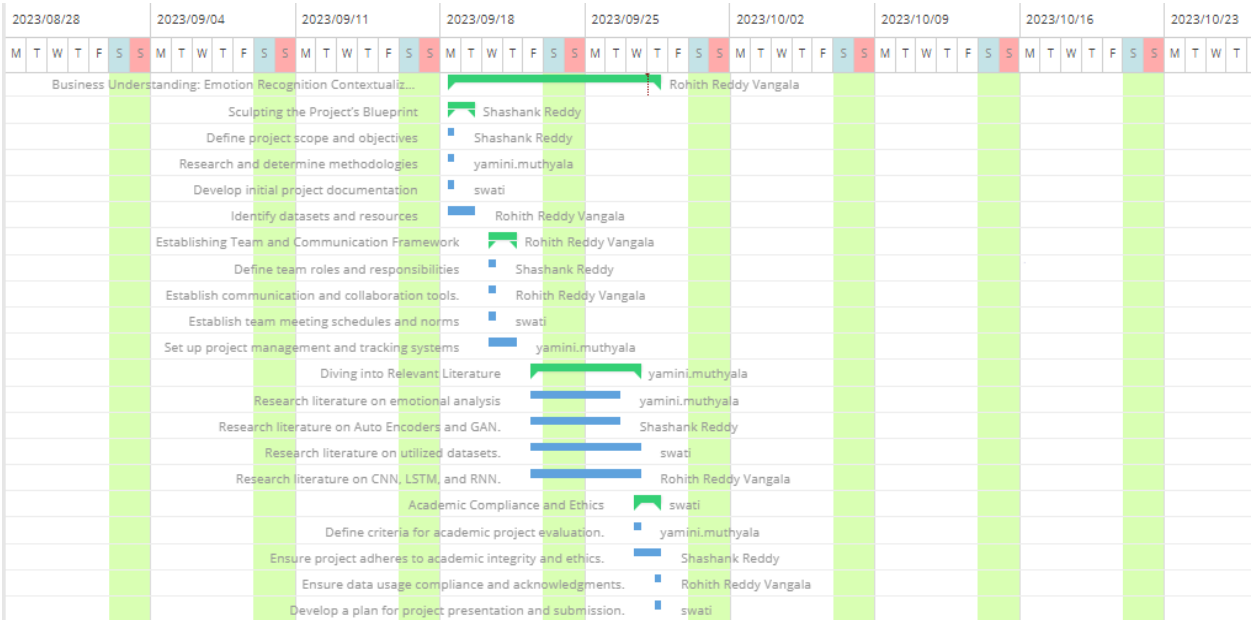
The systematic workflow began with the Business Understanding phase and progressed to the Deployment phase. Weekends were categorized as non-working days in accordance with typical organizational work schedules.Activity distribution was rigorously maintained and altered based on the intricacy of the given activity.The team's approach to task assignment and

overall project management was distinguished by a methodical and professional methodology that employed effective instruments to assure the project's success. Each stage was planned in compliance with all of the tasks.

Business Understanding . The Business Understanding phase focused on thoroughly evaluating and defining the project's goals, objectives, and stakeholder requirements. It involved gathering and comprehending the full scope of the project, including the purpose, intended outcomes, and diverse needs. Within this phase, Gantt chart creation centered on mapping the initial project plan by pinpointing major deliverables, delineating the project scope, and determining the overall timeline and can be seen in Figure 10.

Figure 10

Representation of the Tasks in Business Understanding



It also entailed allocating resources and identifying potential constraints or risks impacting the schedule.

Overall, this phase laid the groundwork for the entire project by enabling a clear understanding of what must be achieved which is seen in Table 10.

Table 10

Business Understanding

Task	Timeline	Team Member	Status of the deliverables
Sculpting the project's Blueprint	09/18/2023-09/19/2023	Shashank Reddy	Completed
Establishing Team and Communication Framework	09/20/2023-09/21/2023	Rohith Reddy Vangala	Completed
Diving into Relevant Literature	09/22/2023-09/27/2023	Yamini Muthyala	Completed
Academic Compliance and Ethics	09/27/2023-09/28/2023	Swati	Completed

In this phase, a literature review was conducted on emotional analysis, datasets, Auto Encoders , CNN, LSTM, and DNN, ensuring academic compliance and ethics. This established a structured, well-defined plan to guide subsequent phases in the Gantt chart and the project's progression. The below figure gives the detailed representation of the tasks that were focused upon during this phase.

Data Understanding. This phase focused on thoroughly exploring and analyzing available data sources to comprehend the information's nature, quality, and properties for the project. This crucially informed Gantt chart creation by preparing for subsequent stages. It involved evaluating data sources, relevance, limitations, and biases. Data collection strategies

were defined, and initial assessments of completeness and quality conducted which can be observed in Table 11.

Table 11

Data Understanding

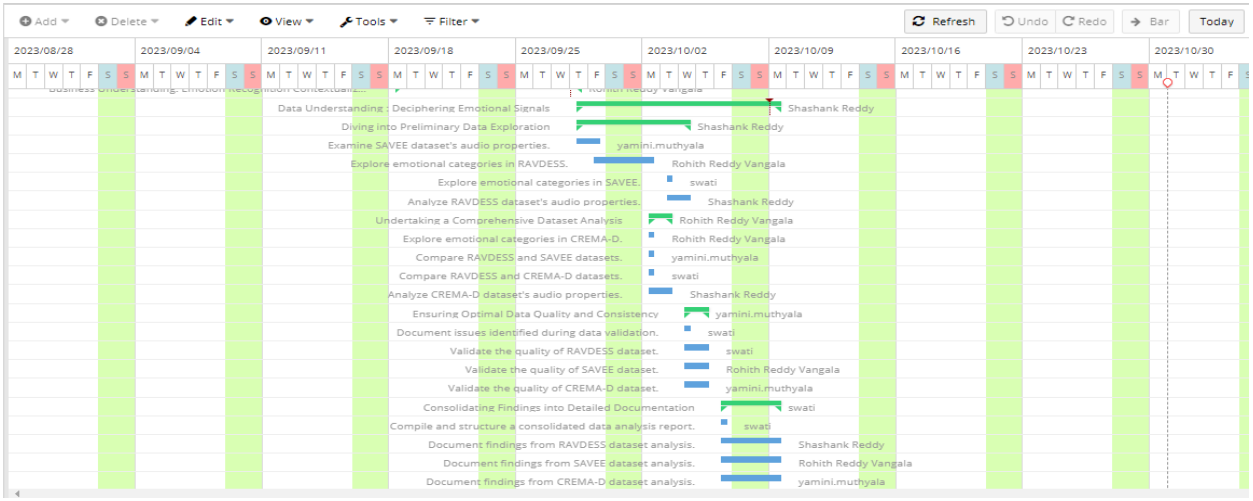
Task	Timeline	Team Member	Status of the deliverables
Dividing into preliminary Data Exploration	09/28/2023-10/04/2023	Shashank Reddy	Completed
Undertaking a Comprehensive Dataset Analysis	10/02/2023-10/03/2023	Rohith Reddy Vangala	Completed
Ensuring Optimal Data Quality and Consistency	10/04/2023-10/05/2023	Yamini Muthyala	Completed
Consolidating Findings into Detailed Documentation	10/06/2023-10/09/2023	Swati	Completed

The Gantt chart outlined timelines and tasks for data collection - identifying sources, acquiring data, preliminary reviews, It allocated time for potential cleaning, transformation, and preparation to ensure data is accurate, reliable, and ready for upcoming analysis stages. Overall, this phase enabled grasping the data landscape to establish well-informed plans through the Gantt chart for the data-centric activities necessary to ready the information for the project.

The Figure 11 explains the detailed outline of the data exploration on SAVEE, RAVEDESS, and CREMA-D Datasets.

Figure 11

Representation of the Tasks in Data Understanding



Data Preparation. This phase focused on tasks to handle missing data, fix inconsistencies, address outliers, standardize formats, data normalization, scaling, and transformation which can be viewed in Table 12.

Table 12

Data Preparation

Task	Timeline	Team Member	Status of the deliverables
Cloud Setup for Data Handling	10/09/2023-10/09/2023	Shashank Reddy	Completed
Elevating Quality and Consistency	10/10/2023-10/10/2023	Shashank Reddy	Completed

Task	Timeline	Team Member	Status of the deliverables
Extracting Informative Features for Model Training	10/11/2023-10/12/2023	Rohith Reddy Vangala	Completed
Enhancing and Expanding Datasets with Augmentation	10/13/2023-10/16/2023	Yamini Muthyala	Completed
Meticulously Organizing Data for optimal Accessibility	10/17/2023-10/18/2023	Swati	Completed

Gantt chart creation entailed scheduling tasks like cleaning, integrating sources, formatting, and executing transformations. This phase involved meticulous work on resolving issues, selecting features, normalizing, scaling, etc. Figure 12 explained the preprocessing done on the three datasets using normalization techniques and the detailed implementation of data augmentation techniques.

Figure 12

Representation of the Tasks in Data Preparation

Modeling. The Modeling phase applied analytical techniques and algorithms to the prepared data to develop models that met project objectives as seen in Table 13 .

Table 13

Modeling

Task	Timeline	Team Member	Status of the deliverables
Crafting Efficient Models Through Strategic Design	10/19/2023-10/23/2023	Shashank Reddy	Completed
Implementing Models for Emotion Recognition Tasks	10/23/2023-10/24/2023	Rohith Reddy Vangala	Completed
Facilitating Knowledge Transfer through Model Training	10/26/2023-10/27/2023	Yamini Muthyala	Completed
Refining and Elevating Model Performance	10/30/2023-10/31/2023	Swati	Completed

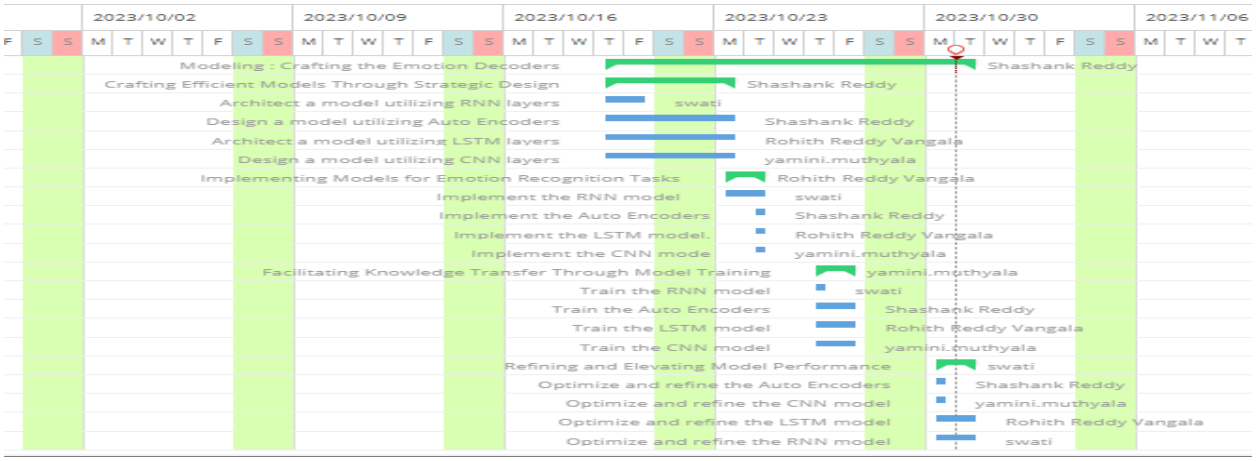
Models were implemented using Auto Encoders, LSTM, CNN, and DNN layers. These models were trained and optimized to achieve accurate emotional analysis. The team focused on architecting, implementing, training, and refining the models.

The Gantt chart outlined tasks like choosing techniques, splitting data, model building, and iteratively refining and testing models to achieve the most accurate, effective solution. It scheduled activities around selecting modeling approaches, training/testing data splits, iterative

development, parameter tuning, and testing to build optimal analytical models aligned with project goals and can be seen in Figure 13.

Figure 13

Representation of the Tasks in Modeling



Evaluation.The Evaluation phase focused on assessing model performance and accuracy against project objectives. It involved evaluating models using metrics, validation, and comparison to predefined criteria from initial project stages.

The Gantt chart outlined tasks like performance evaluation, model comparisons, tuning for improved performance, and documentation which can be observed on Table 14.

Table 14

Evaluation

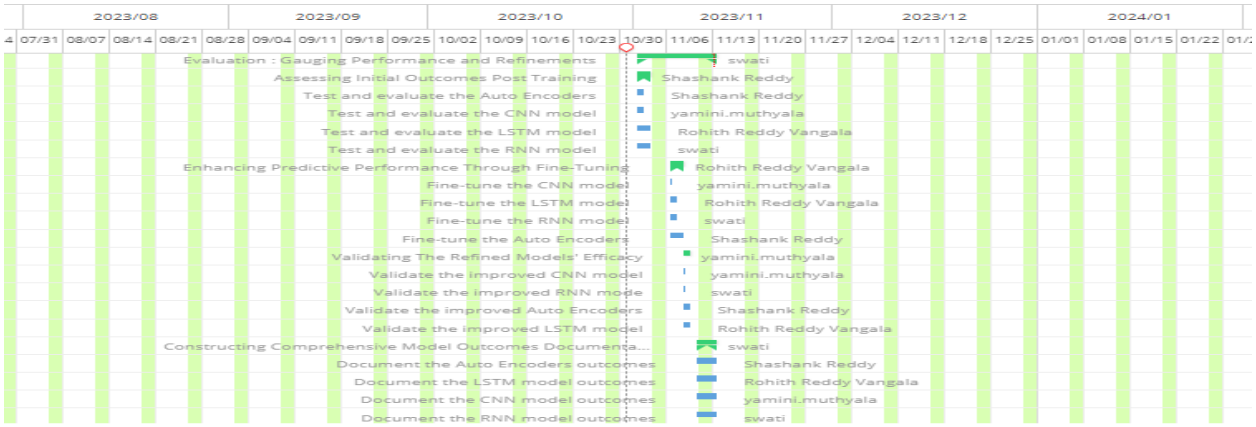
Task	Timeline	Team Member	Status of the deliverables
Assessing Initial Outcomes Post Training	11/01/2023-11/03/2023	Shashank Reddy	Completed

Task	Timeline	Team Member	Status of the deliverables
Enhancing Predictive Performance through Fine-Tuning	11/06/2023-11/08/2023	Rohith Reddy Vangala	Completed
Validating the Refined Models' Efficacy	11/08/2023-11/09/2023	Yamini Muthyala	Completed
Constructing Comprehensive Model Outcomes Documentation	11/10/2023-11/13/2023	Swati	Completed

It also scheduled evaluating if models meet requirements defined during Business Understanding. Key activities involved testing and fine-tuning the models developed in previous stages. It included comprehensive model validation and documentation of outcomes. Also, readiness of the models was assessed for deployment by testing, fine-tuning, and validating them. From the below Figure 14, detailed tasks can be observed for fine-tuning the four models, validating them, and documenting them.

Figure 14

Representation of the Tasks in Evaluation



Deployment

The Deployment phase focused on transitioning successful models into practical application. It entailed preparing models for deployment, integrating them into operational systems as seen in Table 15.

Table 15

Deployment

Task	Timeline	Team Member	Status of the deliverables
Executing Model Deployment and Verification	11/14/2023-11/15/2023	Shashank Reddy	Completed
Conducting Real-time testing and Accumulating Feedback	11/16/2023-11/20/2023	Rohith Reddy Vangala	Completed
Crafting the Final Presentation and Model Overview	11/20/2023-11/21/2023	Yamini Muthyala	Completed
Finalizing Documentation and Preparing for Project Submission	11/22/2023-11/24/2023	Swati	Completed

The Gantt chart scheduled tasks like implementation, documentation, monitoring strategies, and feedback loops for continuous improvement. Key activities involved deploying models for real-world usage along with guidance, training, and monitoring. This phase ensured comprehensive documentation of model outcomes and prepared for the project's final presentation and submission and can be viewed in Figure 15.

Figure 15

Representation of the Tasks in Deployment

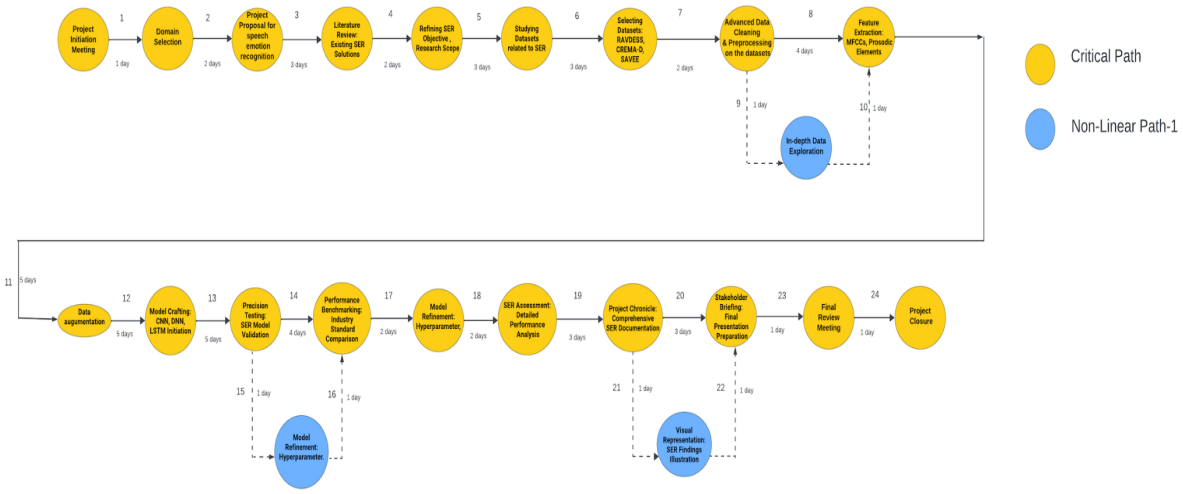
PERT

Planners and schedulers utilized PERT (Program Evaluation Review Technique) charts to manage a project's tasks and activities. In contrast to Gantt charts, which indicated a project's timetable, PERT charts highlighted task interdependencies and the critical path (Evdokimov, 2018). Every milestone along the critical path was important and directly related to how quickly the project was completed since it established the longest possible path. PERT charts employed arrows to depict the links between tasks, and nodes to indicate tasks and milestones.

The critical route was first established by creating a PERT chart with jobs depicted in Figure 16, their dependencies, and the time of each job. Following that, it was combined at the Milestone level for improved comprehension and detail.

Figure 16

PERT Chart (Milestone Level)



In the speech emotion recognition (SER) project, the PERT (Program Evaluation and Review Technique) chart is a crucial tool for overseeing project progression, with a focus on 24 key activities. The chart is tailored to avoid the common pitfalls of micromanagement, instead providing a macroscopic view of the project's evolution.

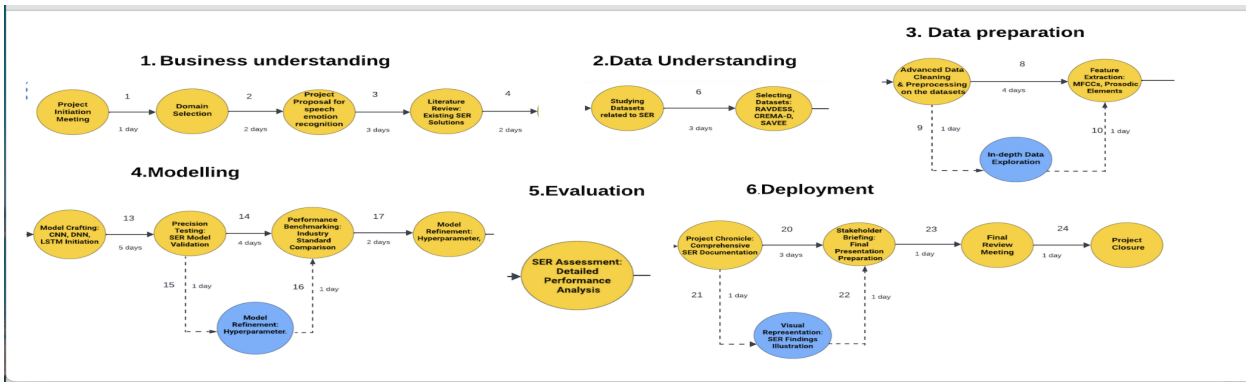
The critical path, comprising 19 of these activities (1, 2, 3, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14, 17, 18, 19, 20, 23, and 24), is a meticulously curated sequence that defines the project's most crucial and time-sensitive elements.

Beginning with the "Project Initiation Meeting," this path sets a foundational tone for the entire project, ensuring alignment among stakeholders and a clear consensus on the project's direction.

Pert chart illustrated below in Figure 17 according to CRISP-DM methodology.

Figure 17

Pert chart according to CRISP-DM



As the project advances, activities such as "Domain Selection," "Project Proposal for Speech Emotion Recognition," and "Refining SER Objective and Research Scope" anchor the critical path. These stages are pivotal in crystallizing the project's goals, ensuring they are specific, measurable, achievable, relevant, and time-bound. The trajectory further includes essential processes like "Selecting Datasets," "Advanced Data Cleaning & Preprocessing," and "Feature Extraction," where the emphasis is on preparing and refining the data for in-depth analysis can be seen in the Figure 18.

Figure 18

Pert Activities with Description

PERT ACTIVITY	PERT ACTIVITY DESCRIPTION
1	Finalising domain after Project Initiation meeting
2	define project objectives, Decide on critical metrics and KPIs
3	Sharpening project focus and expected outcomes.
4	Analyzing existing studies, solutions in emotion recognition,taking the reference of literatures in selecting the datasets
5	Detailed study and comparison of potential datasets.
6	Finalizing datasets post-analysis forproject suitability.
7	Preparing data for CNN, RNN, LSTM , autoencoder models.
8	Utilizing audio characteristics for emotion detection.,
9	Delving into data, identifying key patterns.
10	More optmised and processed data for the feature engineering
11	Expanding data scope to enhance training.
12	Building foundational models with specified algorithms.
13	Ensuring model accuracy in emotion detection.
14	Evaluating models against recognized SER benchmarks
15	Adjusting models for optimal emotion recognition.
16	more refined model for performance testing
17	Iterative improvements based on performance analysis.
18	Comprehensive model evaluation using robust metrics.
19	Systematic recording of methodologies, outcomes, insights.
20	Crafting and rehearsing conclusive project presentation.
21	Developing graphical content to depict results.
22	Reviewing project results with stakeholders
23	Concluding meeting to review project results.
24	Project Conclusion: SER Documentation

The culmination of these efforts is marked by the advanced stage of "Model Crafting: CNN, DNN, LSTM," where theoretical concepts are translated into practical models. This journey, charted through the PERT chart, underlines the significance of each phase in building a robust and accurate SER system.

The PERT chart's structured approach is balanced with flexibility, allowing for adjustments and adaptations in response to evolving project needs or unexpected challenges. This combination of structure and adaptability is essential to managing a project as complex and dynamic as speech emotion recognition. The critical path's sequence, from initial planning to project closure, highlights the interdependencies and importance of each activity. The critical path of the project encompasses a 51-day duration, indicating the sequence of tasks with the longest time to completion..By meticulously following this path and being mindful of potential delays, the project aims to stay on track, ensuring timely completion and the achievement of its ambitious goals. This strategic blend of precision and flexibility in the PERT chart serves not only as a guide for project management but also as a testament to the project’s commitment to quality, efficiency, and innovation in the field of speech emotion recognition.

Data Engineering

3.1 Data Processing

Data Acquisition and Preliminary Processing

The research utilizes three publicly available, open-source datasets: RAVDESS (Livingstone & Russo, 2018), CREMA-D(Cao et al., 2014), and SAVEE (Haq & Jackson, 2010). These datasets enable transparency and reproducibility of the methodology and findings through their open access nature. Their selection provides the foundation for rigorous, verifiable research practices.

The RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song) dataset (Livingstone & Russo, 2018) is a comprehensive dataset consisting of audio files in WAV format. It contains recordings of vocalizations by 24 professional actors, encompassing various emotions expressed through speech and song. The dataset provides detailed metadata, including information about the actor's gender, emotion category, intensity level, and more. The RAVDESS dataset was obtained from the Zenodo open repository and features emotions such as calmness, happiness, sadness, anger, fearfulness, surprise, disgust,and neutral states at different intensity levels.

The CREMA-D (Crowd-Sourced Emotional Multimodal Actors Dataset) dataset (Cao et al., 2014) contains 7,442 audio clips from 91 amateur actors. These vocalizations encompass a range of emotions and are in different languages. Each audio file is stored in WAV format and includes metadata about the actor, language, emotion, and other details. The dataset also includes video clips to support the development of cross-cultural speech emotion recognition systems. It was obtained from the GitHub repository dedicated to this dataset.

The SAVEE (Surrey Audio-Visual Expressed Emotion) dataset (Haq & Jackson, 2010) consists of recordings of 4 male actors speaking neutral and emotional sentences in British English. It includes audio data in WAV format, as well as video clips labeled with actor identity and portrayed emotion. Renowned for its thorough subjective evaluations and high rates of emotion recognition, the dataset was obtained from the University of Surrey's SAVEE website.

Storage and Infrastructure Setup

In the infrastructure setup for the research, Google Cloud Platform's storage services were utilized to host the datasets securely. The datasets were stored in a Google Cloud bucket, a solution chosen for its robust security features and its ability to integrate smoothly with Google Colab. Google Colab was then employed for computational tasks, providing a platform with powerful data processing capabilities. This setup facilitated the use of advanced deep learning libraries and tools without the need for local resources, thereby optimizing the research workflow.

Ensuring Quality and Standardizing Data

The integrity and completeness of the datasets were ensured by utilizing the Pandas library in Python to perform a meticulous check for missing values, confirming the absence of such discrepancies. The uniformity of audio file formats was verified through consistent WAV format adherence, a process facilitated by the PyDub library. For advanced visualization of the audio data, the LibROSA library (McFee et al., 2015) was used to generate waveforms and spectrograms, with Matplotlib aiding in their graphical representation. These Python libraries were integral to the quality assurance and preprocessing phases of the research.

Data Integration and Preprocessing Techniques

Merging the datasets from CREMA-D (Cao et al., 2014), RAVDESS (Livingstone & Russo, 2018), and SAVEE (Haq & Jackson, 2010) was a crucial step in creating a unified dataframe. The original audio files were kept as references during this process. Background noise reduction techniques using the noisereduce library (Virtanen et al., 2020) were applied to enhance data clarity and quality by reducing unwanted noise in the audio files. Additionally, augmentation techniques like noise augmentation, time stretching, and pitch shifting were employed to address data imbalance issues specifically for emotions that had fewer representations (Salamon & Bello, 2017).

Extracting and preparing features

A crucial part of preparing the data for emotion recognition was extracting important audio features such as Zero Crossing Rate, Chroma STFT, Mel Frequency Cepstral Coefficients, Root Mean Square Value , and Mel Spectrogram (Rezapour Mashhadi & Osei-Bonsu, 2023). These features were incorporated into the dataset to enhance it for training the model. This step played a vital role in capturing a comprehensive range of data attributes essential for recognizing emotions.

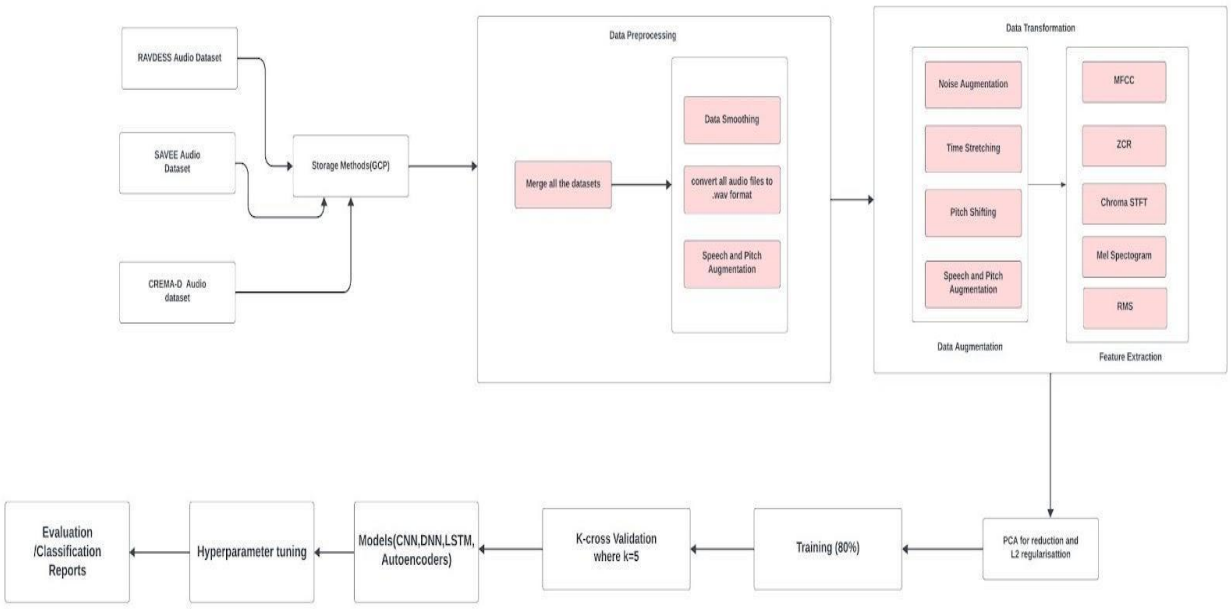
Normalization, Encoding, and Dataset Splitting

To standardize the feature scales, the data was normalized through row-wise Z-score normalization (Fei, Gao, Lu, & Xiang, 2021). Categorical data such as gender and emotions were converted into numerical formats using label encoding. The dataset was then split into separate sets for training (60%), testing (20%), and validation (20%). This division allows for effective model training, testing, and performance validation.

The outlined steps ensure that the datasets are well-prepared, standardized, and suitable for subsequent stages of model development. Paying attention to data quality, feature richness, and balanced representation is essential in constructing robust and accurate emotion recognition models. Data process overview is illustrated in figure 19 below for the whole research.

Figure 19

Data Process Overview



Above Figure shows a sequential approach starting with the storage of RAVDESS (Livingstone & Russo, 2018), CREMA-D (Cao et al., 2014), and SAVEE (Haq & Jackson, 2010) on Google Cloud Platform. The preprocessing phase involved merging the datasets, converting files to WAV format, and applying speech and pitch augmentations. Transformation techniques like noise augmentation, time stretching, and pitch shifting were used to enhance data variety(Salamon & Bello, 2017). Features like MFCC, ZCR, Chroma STFT, Mel Spectrogram, and RMS were extracted(Rezapour Mashhadi & Osei-Bonsu, 2023). The dataset was then split into separate sets for training (60%), testing (20%), and validation (20%). Models such as CNN,

DNN, LSTM, and Autoencoders were developed, fine-tuned, and validated, leading to comprehensive evaluation reports.

3.2 Data Collection

The research utilizes three public, open-source datasets - RAVDESS (Livingstone & Russo, 2018), CREMA-D (Cao et al., 2014), and SAVEE (Haq & Jackson, 2010) - to promote transparency and reproducibility of the methodology and findings. Specifically, the RAVDESS dataset from the Zenodo repository contains 1440 WAV audio files of emotional speech and songs performed by 24 actors(12 male actors,12 female actors) as illustrated in Figure 20.

Figure 20

Ravdess Dataset

	Filename	Emotion	Gender	Dataset	Path
0	03-01-03-02-01-01-16.wav	Happy	female	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...
1	03-01-02-01-02-01-16.wav	Calm	female	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...
2	03-01-06-01-02-02-16.wav	Fearful	female	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...
3	03-01-03-02-01-02-16.wav	Happy	female	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...
4	03-01-05-01-02-02-16.wav	Anger	female	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...
...	...	...	...	...	...
1435	03-01-06-02-02-02-03.wav	Fearful	male	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...
1436	03-01-07-02-02-01-03.wav	Disgust	male	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...
1437	03-01-06-02-01-01-03.wav	Fearful	male	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...
1438	03-01-08-01-01-02-03.wav	Surprise	male	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...
1439	03-01-07-02-02-02-03.wav	Disgust	male	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...

1440 rows × 5 columns

with metadata like actor, gender, emotion, speech content, and intensity to enable systematic processing and analysis. By relying on publicly available data like RAVDESS with detailed labels, the research aims to have an accessible and verifiable approach.

The CREMA-D dataset (Cao et al., 2014), sourced from its GitHub repository, provides 7,442 audio clips in WAV format from 91 amateur actors as well as corresponding video clips. These files contain extensive metadata labels. CREMA-D is critical for building cross-cultural speech emotion recognition systems that leverage both audio and visual cues, owing to its cultural diversity and multimodal nature with aligned video. Figure 21 illustrates the dataset.

Figure 21

Crema-d Dataset

	Filename	Emotion	Gender	Dataset	Path
0	1079_TIE_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
1	1080_DFA_NEU_XX.wav	Neutral	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
2	1080_DFA_SAD_XX.wav	Sad	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
3	1079_TIE_HAP_XX.wav	Happy	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
4	1079_TSI_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
...	...	...	...	...	...
7437	1006_IEO_HAP_LO.wav	Happy	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
7438	1006_ITH_ANG_XX.wav	Anger	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
7439	1006_ITH_SAD_XX.wav	Sad	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
7440	1006_ITS_NEU_XX.wav	Neutral	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
7441	1006_ITH_NEU_XX.wav	Neutral	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...

7442 rows × 5 columns

The SAVEE dataset, sourced from the University of Surrey's website, provides audio recordings and corresponding video clips of 4 male actors uttering neutral and emotional sentences in British English. Each file contains labels for the speaker's identity and emotion displayed. Owing to its speaker diversity and emotional richness, SAVEE has been effectively utilized for benchmarking speaker-independent emotion recognition systems, achieving high performance (Haq & Jackson, 2010).

Figure 22 illustrates the SAVEE Dataset with all the corresponding columns

Figure 22

Savee Dataset

	Filename	Emotion	Gender	Dataset	Path
0	DC_a01.wav	Anger	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
1	DC_a09.wav	Anger	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
2	DC_a07.wav	Anger	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
3	DC_a02.wav	Anger	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
4	DC_a05.wav	Anger	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
...	...	...	...	...	...
475	KL_n06.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
476	KL_n05.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
477	KL_n26.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
478	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
479	KL_n11.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...

480 rows × 5 columns

Table 16 below presents a comparative overview of key characteristics of three datasets: RAVDESS (Livingstone & Russo, 2018), CREMA-D (Cao et al., 2014), and SAVEE (Haq & Jackson, 2010).

Table 16

Summary of Key Features of Datasets

Dataset	Modality	Emotions	Number of Actors	Total Clips
RAVDESS	Audio	Calm, Happy, Sad, Angry, Fearful, Surprise, Disgust, Neutral	24	1440

Dataset	Modality	Emotions	Number of Actors	Total Clips
CREMA-D	Audio-Visual	Anger, Disgust, Fear, Happy, Neutral, Sad	91	7442
SAVEE	Audio-Visual	Anger, Disgust, Fear, Happy, Neutral, Sad, Surprise	4	480

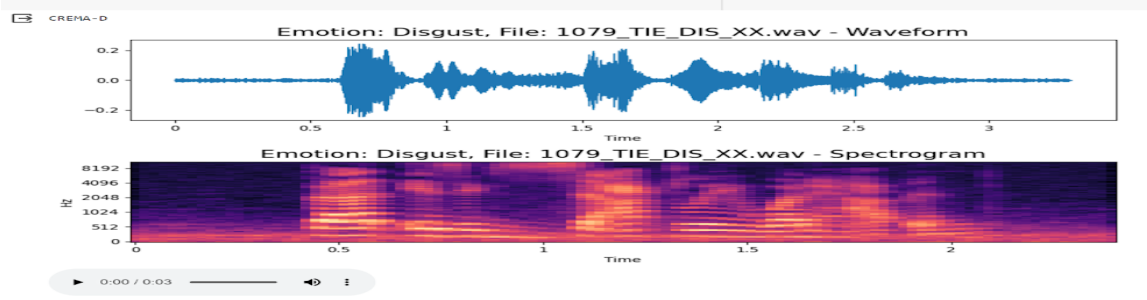
It details their modality, range of emotions, number of actors involved, and total number of clips, providing a comprehensive perspective on the datasets' diversity and scope, essential for the research's robustness in speech emotion recognition.

Raw Data Samples

This section presents an in-depth examination of the raw data samples collected in the study. The data, consisting primarily of audio recordings, captures three distinct emotions. To facilitate a comprehensive analysis, this data is presented in two formats: waveforms and spectrograms. Figure 23 presents a detailed audio sample from the CREMA-D dataset, specifically capturing the emotion of disgust.

Figure 23

Crema-D Dataset

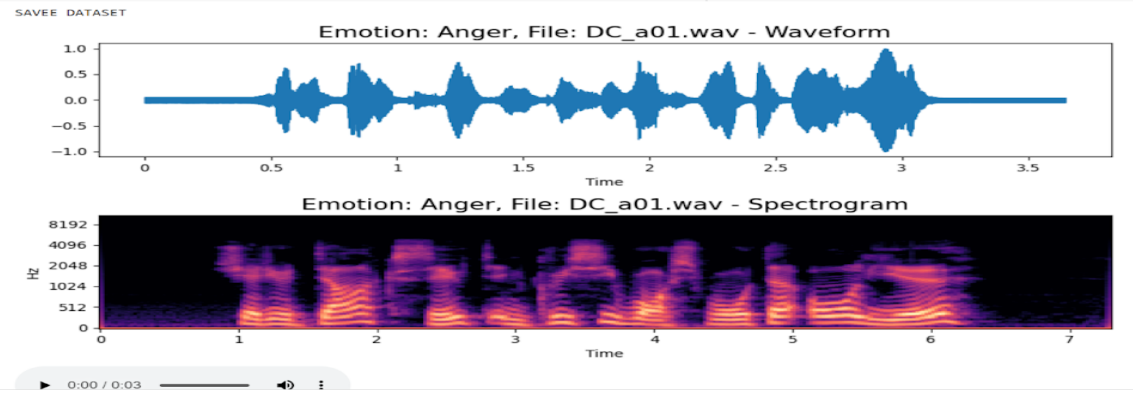


In the waveform representation, variations in amplitude over time are evident, corresponding to vocal intensity and pitch fluctuations characteristic of the expression of disgust. These variations align with vocal cues often associated with the emotion of disgust, such as heightened intensity and potentially lower pitch. In the spectrogram , the visualization reveals the spectrum of frequencies over time, highlighting unique acoustic features indicative of the emotion of disgust. These distinct patterns in the audio data offer valuable insights for recognizing and analyzing the disgust emotion in speech.

Figure 24 is a visual analysis of an audio sample from the SAVEE dataset, designated to represent the emotion of anger.

Figure 24

Savee Dataset



The waveform and spectrogram together offer insights into the acoustic features that characterize the expression of anger in speech.

In the waveform displayed at the top, the amplitude variations suggest a higher and more variable intensity level, which is often associated with anger. The peaks and troughs indicate a forceful enunciation, a common characteristic in angry speech. The variability in the amplitude

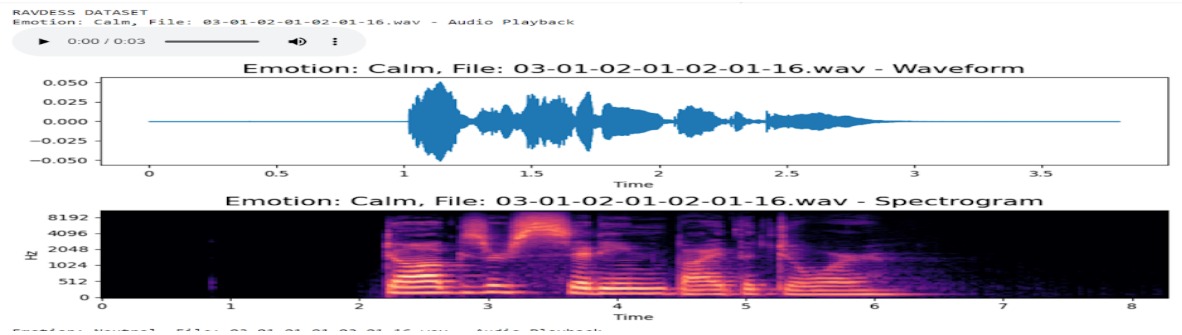
and the abrupt changes in the waveform are indicative of the volatility and intensity typically present in an angry tone of voice.

The spectrogram, shown in the lower half of the image, further details the frequency components over time. Anger is often conveyed through a higher frequency range and increased energy in the speech signal, which can be visualized as brighter and more intense colors in the spectrogram. Additionally, the pronounced bands of energy that appear throughout the spectrogram correlate with the emphatic articulation often found in angry speech.

Figure 25 exhibits a waveform and spectrogram corresponding to a 'calm' emotion audio file from the RAVDESS dataset.

Figure 25

Ravdess Dataset



The waveform shown in the upper section is characterized by its relatively flat and uniform shape, indicating a smooth and steady vocal delivery that one would expect from a calm speech pattern. The minimal variability in amplitude reflects a controlled volume and pitch, which are acoustic signatures commonly associated with a calm emotional state.

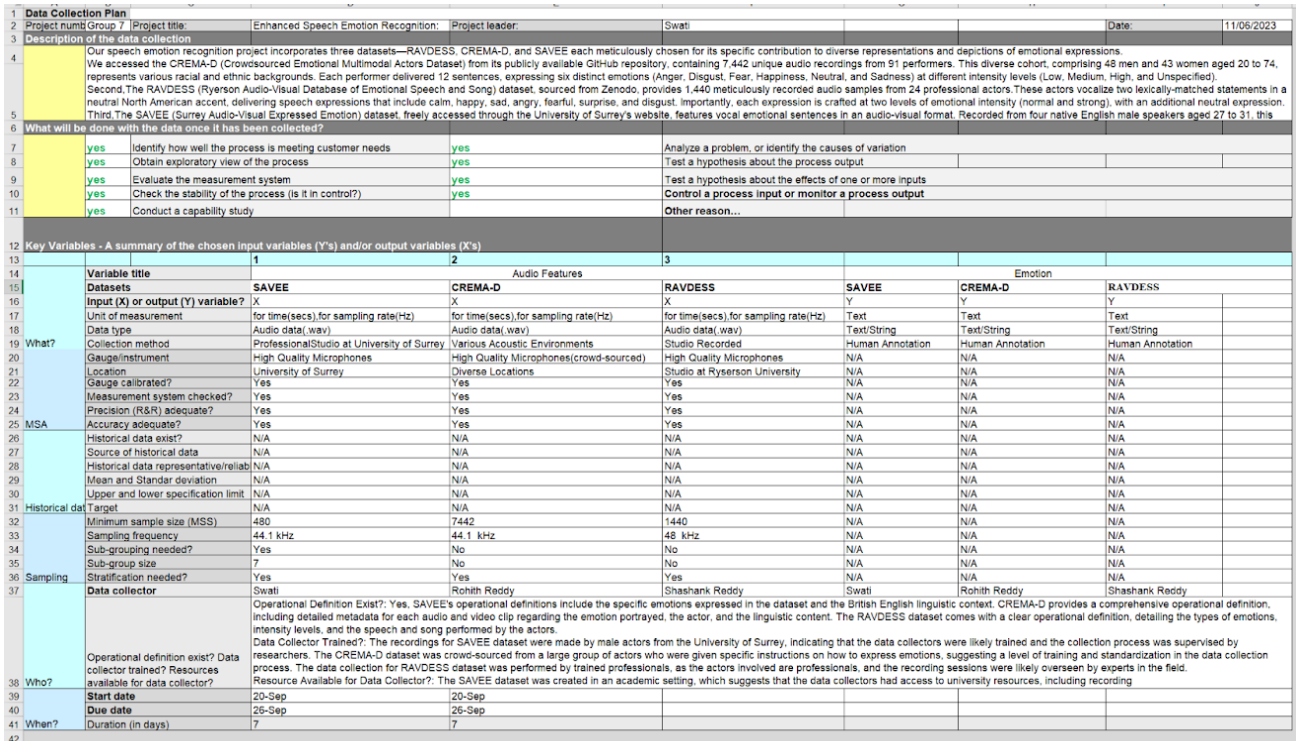
The spectrogram in the lower section provides a visual representation of the sound's frequency content over time, with cooler colors signifying less intense frequency activity. This spectral analysis reveals a lack of abrupt energy bursts across the frequency bands, further

reinforcing the notion of calmness. The uniform distribution of energy and the absence of dramatic fluctuations in the spectrogram underscore the consistent and tranquil nature of the audio sample.

Data collection approach for the CREMA-D, RAVDESS, and SAVEE datasets encompasses a variety of methodologies, providing a comprehensive overview of the data acquisition process. It is illustrated in Figure 26.

Figure 26

Data Collection Plan For All Datasets



Data Collection Plan									
Project numb	Group 7	Project title:	Enhanced Speech Emotion Recognition:	Project leader:	Swati			Date:	11/06/2023
Description of the data collection									
	Our speech emotion recognition project incorporates three datasets—RAVDESS, CREMA-D, and SAVEE each meticulously chosen for its specific contribution to diverse representations and depictions of emotional expressions. We accessed the CREMA-D (Crowdsourced Emotional Multimodal Actors Dataset) from its publicly available GitHub repository, containing 7,442 unique audio recordings from 91 performers. This diverse cohort, comprising 48 men and 43 women aged 20 to 74, represents various racial and ethnic backgrounds. Each performer delivered 12 sentences, expressing six distinct emotions (Anger, Disgust, Fear, Happiness, Neutral, and Sadness) at different intensity levels (Low, Medium, High, and Unspecified). Second,The RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song) dataset, sourced from Zenodo, provides 1,440 meticulously recorded audio samples from 24 professional actors.These actors vocalize two lexically-matched statements in a neutral North American accent, delivering speech expressions that include calm, happy, sad, angry, fearful, surprise, and disgust. Importantly, each expression is crafted at two levels of emotional intensity (normal and strong), with an additional neutral expression. Third,The SAVEE (Surrey Audio-Visual Expressed Emotion) dataset, freely accessed through the University of Surrey's website, features vocal emotional sentences in an audio-visual format. Recorded from four native English male speakers aged 27 to 31, this								
What will be done with the data once it has been collected?									
	yes	Identify how well the process is meeting customer needs	yes	Analyze a problem, or identify the causes of variation					
	yes	Obtain exploratory view of the process	yes	Test a hypothesis about the process output					
	yes	Evaluate the measurement system	yes	Test a hypothesis about the effects of one or more inputs					
	yes	Check the stability of the process (is it in control?)	yes	**Control a process input or monitor a process output**					
	yes	Conduct a capability study		**Other reason...**					
Key Variables - A summary of the chosen input variables (Y's) and/or output variables (X's)									
			1	**2**	**3**				
	Variable title		Audio Features			Emotion			
	Datasets		**SAVEE**	**CREMA-D**	**RAVDESS**	**SAVEE**	**CREMA-D**	**RAVDESS**	
	Input (X) or output (Y) variable?		X	X	X	Y	Y	Y	
	Unit of measurement		for time(secs),for sampling rate(Hz)	for time(secs),for sampling rate(Hz)	for time(secs),for sampling rate(Hz)	Text	Text	Text	
	Data type		Audio data(.wav)	Audio data(.wav)	Audio data(.wav)	Text/String	Text/String	Text/String	
What?	Collection method		ProfessionalStudio at University of Surrey	Various Acoustic Environments	Studio Recorded	Human Annotation	Human Annotation	Human Annotation	
	Gauge/Instrument		High Quality Microphones	High Quality Microphones(crowd-sourced)	High Quality Microphones	N/A	N/A	N/A	
	Location		University of Surrey	Diverse Locations	Studio at Ryserson University	N/A	N/A	N/A	
	Gauge calibrated?		Yes	Yes	Yes	N/A	N/A	N/A	
	Measurement system checked?		Yes	Yes	Yes	N/A	N/A	N/A	
	Precision (R&R) adequate?		Yes	Yes	Yes	N/A	N/A	N/A	
MSA	Accuracy adequate?		Yes	Yes	Yes	N/A	N/A	N/A	
	Historical data exist?		N/A	N/A	N/A	N/A	N/A	N/A	
	Source of historical data		N/A	N/A	N/A	N/A	N/A	N/A	
	Historical data representative/reliab		N/A	N/A	N/A	N/A	N/A	N/A	
	Mean and Standar deviation		N/A	N/A	N/A	N/A	N/A	N/A	
	Upper and lower specification limit		N/A	N/A	N/A	N/A	N/A	N/A	
Historical dat	Target		N/A	N/A	N/A	N/A	N/A	N/A	
	Minimum sample size (MSS)		480	7442	1440	N/A	N/A	N/A	
	Sampling frequency		44.1 kHz	44.1 kHz	48 kHz	N/A	N/A	N/A	
	Sub-grouping needed?		Yes	No	No	N/A	N/A	N/A	
	Sub-group size		7	No	No	N/A	N/A	N/A	
Sampling	Stratification needed?		Yes	Yes	Yes	N/A	N/A	N/A	
	Data collector		Swati	Rohith Reddy	Shashank Reddy	Swati	Rohith Reddy	Shashank Reddy	
Who?	Operational definition exist? Data collector trained? Resources available for data collector?		Operational Definition Exist?: Yes, SAVEE's operational definitions include the specific emotions expressed in the dataset and the British English linguistic context. CREMA-D provides a comprehensive operational definition, including detailed metadata for each audio and video clip regarding the emotion portrayed, the actor, and the linguistic content. The RAVDESS dataset comes with a clear operational definition, detailing the types of emotions, intensity levels, and the speech and song performed by the actors. Data Collector Trained?: The recordings for SAVEE dataset were made by male actors from the University of Surrey, indicating that the data collectors were likely trained and the collection process was supervised by researchers. The CREMA-D dataset was crowd-sourced from a large group of actors who were given specific instructions on how to express emotions, suggesting a level of training and standardization in the data collection process. The data collection for RAVDESS dataset was performed by trained professionals, as the actors involved are professionals, and the recording sessions were likely overseen by experts in the field. Resource Available for Data Collector?: The SAVEE dataset was created in an academic setting, which suggests that the data collectors had access to university resources, including recording						
	Start date		20-Sep	20-Sep					
	Due date		26-Sep	26-Sep					
When?	Duration (in days)		7	7					

The integrity of the datasets was meticulously verified through various checks. This included gauge calibration to ensure the accuracy of recording equipment and measurement system checks to validate the data collection process. The precision and accuracy of the data

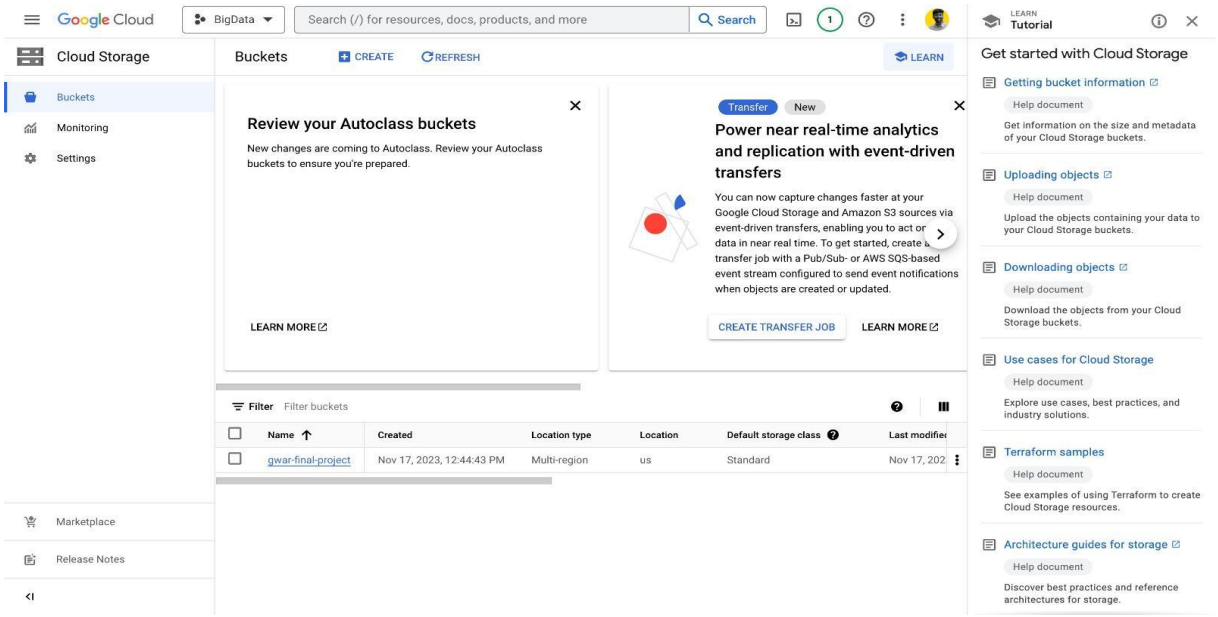
were assessed, although the spreadsheet indicates that historical data and statistical measures such as mean and standard deviation were not applicable or provided.

The thoroughness of the data collection process, combined with the quality checks and the academic sources of the data, suggests that the datasets meet the necessary standards for use in developing reliable speech emotion recognition models. This level of detail underscores the datasets' suitability for rigorous research and application in the field of emotion recognition.

After the data collection, stored the acquired datasets in a secure Google Cloud bucket as illustrated in Figure 27.

Figure 27

Google Cloud Bucket

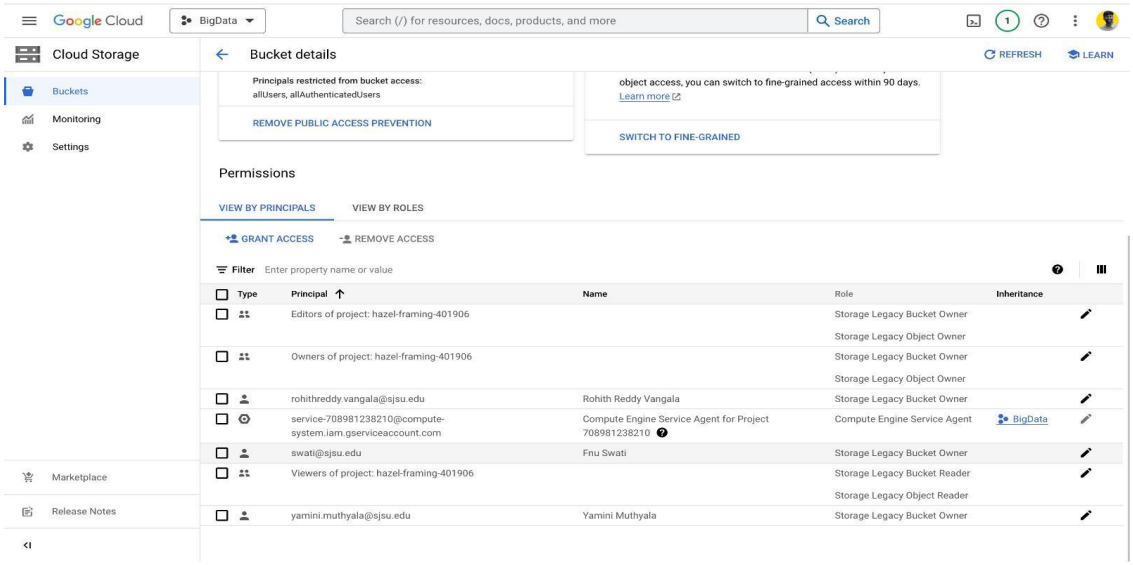


For their scalability, security, and ease of access. This choice was made because Google Cloud storage is well-suited for managing large datasets and ensures scalability, security , ease of access and efficient data retrieval during subsequent processing stages.

It's important to note that only authorized team members as illustrated in Figure 28, have exclusive access to this cloud storage, with strict policies in place to maintain data integrity and security.

Figure 28

Access Privileges



This controlled access allows for remote and collaborative work by enabling team members to retrieve and process the data concurrently.

3.3 Data Preprocessing

To ensure the reliability and effectiveness of deep learning applications, essential steps were taken during the data pre-processing stage for the RAVDESS (Livingstone & Russo, 2018), CREMA-D (Cao et al., 2014), and SAVEE (Haq & Jackson, 2010) datasets. This thorough process played a crucial role in converting raw data into a usable format suitable for advanced analytical techniques.

Firstly, a comprehensive evaluation was performed to identify any instances of missing data within each dataset using pandas library from python. This process was essential in ensuring

that the data was reliable and complete. For CREMA-D (Cao et al., 2014), the analysis confirmed that key variables such as Filename, Emotion, Gender, Dataset, and Path had no missing values. The RAVDESS and SAVEE datasets were also carefully examined and found to have no missing values in their respective variables as illustrated in Figure 29.

Figure 29

Checking Missing Values

```
Missing values in CREMA-D dataset:
 Filename    0
Emotion      0
Gender       0
Dataset      0
Path         0
dtype: int64

Missing values in RAVDESS dataset:
 Filename    0
Emotion      0
Gender       0
Dataset      0
Path         0
dtype: int64

Missing values in SAVEE dataset:
 Filename    0
Emotion      0
Gender       0
Dataset      0
Path         0
dtype: int64
```

Verifying the integrity of the data through this pre-processing step established a solid foundation for subsequent processing stages by guaranteeing that all datasets were thorough and ready for further analysis.

As part of the pre-processing, it was important to make sure that all datasets had consistent data formats. It was verified that all audio files were consistently in the WAV format using pyDub library from python as illustrated in Figure 30.

Figure 30

Checking Format

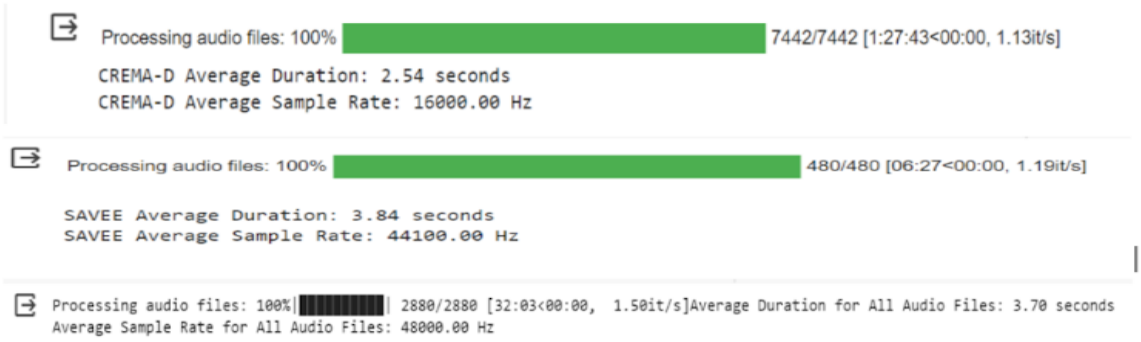
```
Non-WAV files in CREMA-D: []
Non-WAV files in RAVDESS: []
Non-WAV files in SAVEE: []
Count of non-WAV files in CREMA-D: 0
Count of non-WAV files in RAVDESS: 0
Count of non-WAV files in SAVEE: 0
```

This consistency is essential for ensuring standardized data processing, particularly when working with multiple datasets within a single study.

Analyzing the sampling rates and average durations of audio files in the datasets was an important part of pre-processing. This analysis provided insights into the temporal characteristics of the audio data and ensured that they aligned with the intended feature extraction and analysis methods as illustrated in Figure 31.

Figure 31

Average Duration And Sampling Rates



Having a clear understanding of these aspects enabled tailoring subsequent processing steps to suit each dataset's unique characteristics.

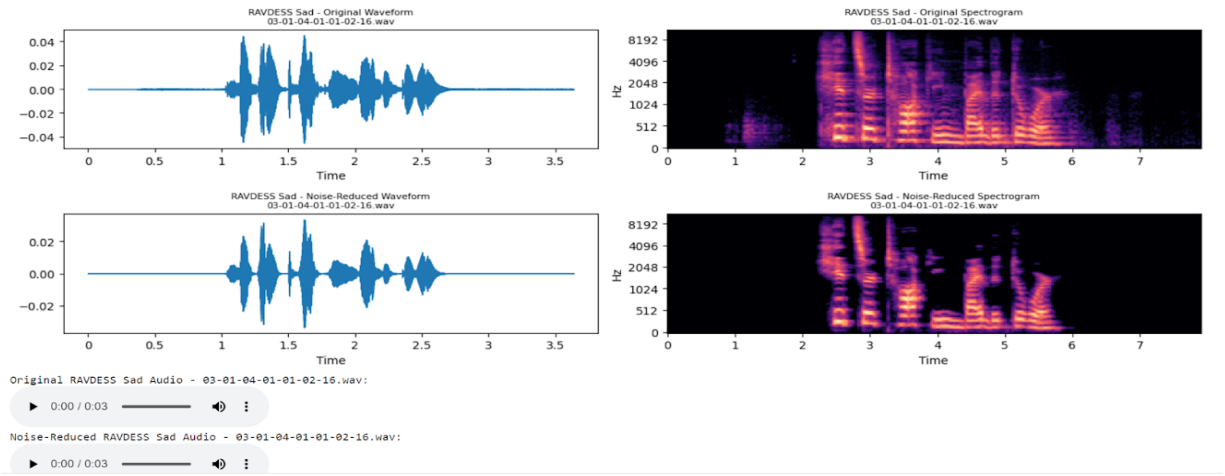
To improve the quality of the audio files, data smoothing was conducted using a Python library called noisereduce (Virtanen et al., 2020). This technique played a crucial role in reducing background noise and enhancing overall audio quality. In speech and emotion recognition tasks, minimizing noise is vital as it can greatly impact the performance of deep learning models. By utilizing the capabilities of the noisereduce library, finally able to create a dataset that is clear and free from unwanted noise, ensuring accurate emotion recognition results.

To demonstrate the effectiveness of these cleaning and validation steps, carefully chosen samples from the pre-processed datasets were utilized. For example, in RAVDESS (Livingstone & Russo, 2018), an audio file named "03-01-04-01-01-02-16.wav" underwent both format consistency and metadata accuracy checks.

This particular file depicted an actor exhibiting a sad emotion and showcased successful application of the pre-processing steps as illustrated in Figure 32.

Figure 32

Ravdess Data Smoothing



The Figure presents a comparative analysis of audio data before and after noise reduction from the RAVDESS dataset, specifically demonstrating the 'sad' emotional state. On the top, an original waveform and spectrogram of the audio file are displayed, followed by their noise-reduced counterparts below.

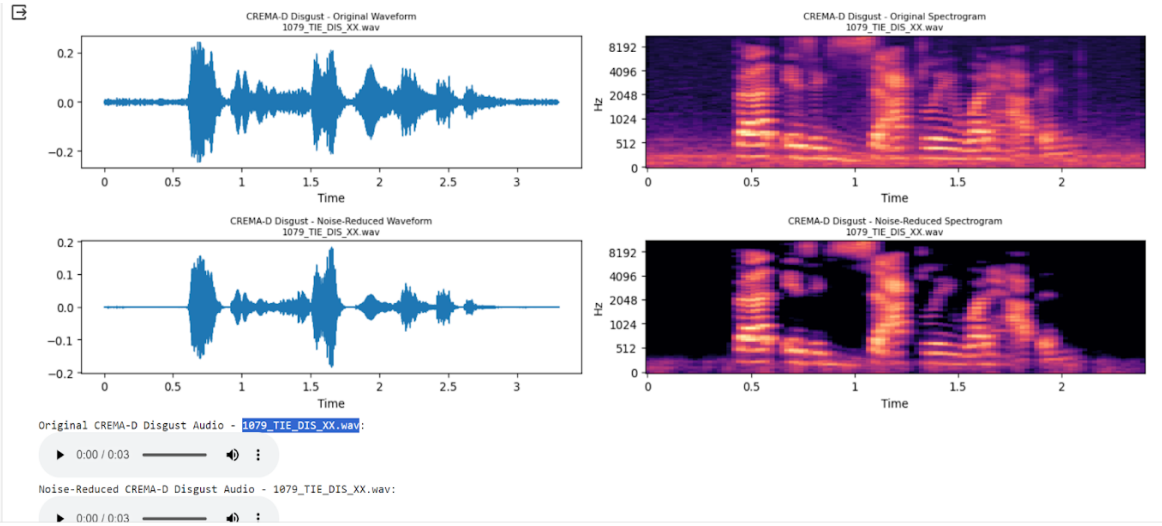
The upper waveform exhibits a raw audio signal with the typical fluctuations that represent the vocal expression of sadness, which often involves a softer, more subdued tone and slower speech. The accompanying original spectrogram shows the frequency distribution over time, with varying intensities and energy concentrations that are inherent to the unfiltered audio.

The lower part of the image shows the audio data after the application of noise reduction. The waveform appears smoother, with less erratic spikes, indicating that extraneous noise has been filtered out. Correspondingly, the noise-reduced spectrogram depicts a cleaner spectral representation with fewer artifacts, enhancing the clarity of the emotional speech features.

Similarly, a CREMA-D (Cao et al., 2014) sample titled "1079_TIE_DIS_XX.wav" was employed to exemplify an instance where the actor expresses disgust; this audio file was validated for format and enhanced for optimal audio quality as illustrated in Figure 33.

Figure 33

Crema-D Data Smoothing



The Figure showcases a comparative visual analysis of an audio signal from the CREMA-D dataset, specifically an utterance labeled to convey 'disgust'. The visualization includes both the waveform and spectrogram before and after the application of a noise reduction process.

The top row of the image displays the original waveform and spectrogram. The waveform shows the characteristic fluctuations in amplitude, which may correspond to the

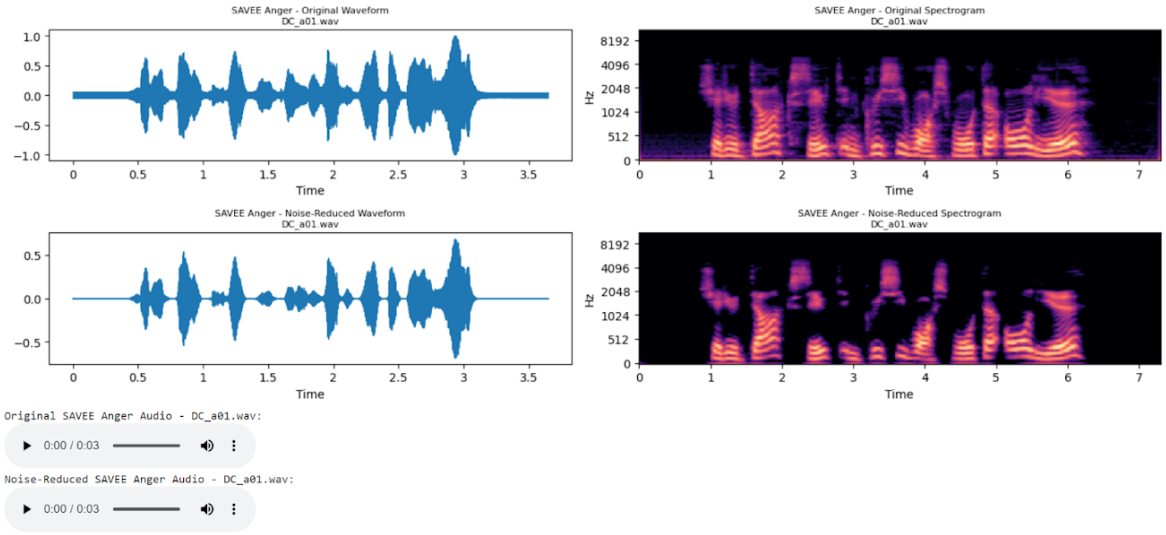
expressive modulation often found in the vocalization of disgust. The spectrogram illustrates the frequency content over time, with bright areas representing higher energy at specific frequencies, which could be indicative of the tonal modulations of a disgusted speech.

The bottom row reveals the changes post noise reduction. The waveform appears smoother, indicating that transient noise has been filtered out, leaving a clearer signal that presumably retains the important emotional content. The spectrogram also reflects this noise reduction, displaying a less cluttered image that allows for a more precise analysis of the frequency components pertinent to the emotion of disgust.

Lastly, a SAVEE dataset file like "DC_a01.wav" provided confirmation that the dataset is prepared for advanced analysis by displaying an actor conveying anger with complete metadata, correct format adherence, and minimized background noise as illustrated in Figure 34.

Figure 34

Savee Data Smoothing



The Figure provided exhibits a set of visual representations for an audio sample from the SAVEE dataset, which is annotated to express anger. It compares the original audio data with a noise-reduced version, using both waveform and spectrogram visualizations.

The top left panel shows the original waveform of the audio file "DC_a01.wav", where the variations in amplitude likely correspond to the vocal intensity and pitch variations typical of an angry speech. The waveform's pronounced peaks and troughs suggest a strong dynamic range and abrupt vocal changes, common in expressions of anger.

On the top right, the original spectrogram provides a visual mapping of the frequency spectrum over time. The bright, densely packed areas indicate a high level of energy across various frequencies, which can be associated with the emphatic articulation and raised pitch that often accompany anger.

The bottom panels display the noise-reduced versions of the same audio sample. In the waveform, the smoothing out of spikes and reduction of background fluctuations are evident, highlighting the primary vocal expression without ambient interference. The noise-reduced spectrogram shows a cleaner distribution of frequencies, with less scattered noise, thus allowing for a more focused analysis of the emotional content in the speech.

To ensure accuracy and quality, the datasets underwent extensive pre-processing steps including data cleaning, validation, format consistency checks, and noise reduction. The meticulousness in the pre-processing phase highlights a commitment to maintaining data integrity and reliability. These factors are vital for successful data-driven research, especially in emotion recognition using deep learning models.

Strategies for Dealing with Possible Data Irregularities If Discovered

In the pre-processing stage, no missing values or inconsistent formats were found in the CREMA-D (Cao et al., 2014), RAVDESS (Livingstone & Russo, 2018), and SAVEE (Haq & Jackson, 2010) datasets. However, it is essential to discuss the potential methods that would have been utilized if any irregularities had been discovered. These strategies play a crucial role in guaranteeing data quality and dependability – especially within a project that heavily relies on accurate audio data for emotion recognition purposes.

Dealing with Missing Data

Techniques for Filling in Values. In the event that any of the datasets contained missing values, would have used suitable methods to fill them in. When it comes to categorical data such as Emotion or Gender, the mode (most commonly occurring value) could be utilized for filling in missing values. For numerical data, either mean or median imputation might be applied depending on how the data is distributed.

Data Deletion. If a large percentage of data is missing and imputation could introduce bias, the affected records would be considered for removal to mitigate this.

Establishing a Uniform Audio Format. When encountering audio files in various formats, it would be necessary to convert them into a standardized format (such as WAV) using audio processing libraries such as librosa or ffmpeg. Maintaining consistency in the audio format is vital for ensuring uniformity during processing and feature extraction.

Batch Processing. To efficiently convert large datasets into different formats, will make use of batch processing scripts.

Addressing Sampling Rate Variations

Resampling. Dealing with differing sampling rates in datasets requires the implementation of resampling techniques to achieve a consistent rate. This involves using audio processing tools to ensure that all audio files have an identical sampling rate, which is essential for conducting comparative analysis and extracting relevant features.

Quality Assurance. After the resampling process, quality checks will be conducted to ensure that the audio quality has not been negatively affected.

These theoretical approaches exemplify the typical techniques employed in data science to address prevalent problems found in datasets. By incorporating these methods can ensure that the data utilized in deep learning models is of exceptional quality, which ultimately enhances the reliability and precision of results. Within this project, the absence of such issues within datasets showcases their strong foundation and dependability for Enhanced Speech Emotion Recognition.

After completing the thorough pre-processing of each dataset, the next important phase was to merge them into a cohesive structure. The RAVDESS (Livingstone & Russo, 2018), CREMA-D (Cao et al., 2014), and SAVEE (Haq & Jackson, 2010) datasets were combined into one dataframe after being cleansed, validated, and standardized as illustrated in Figure 35.

Figure 35

Combined Data Frame

```
df_final
```

	Filename	Emotion	Gender	Dataset	Path
0	1079_TIE_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
1	1080_DFA_NEU_XX.wav	Neutral	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
2	1080_DFA_SAD_XX.wav	Sad	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
3	1079_TIE_HAP_XX.wav	Happy	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
4	1079_TSI_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
...	...	...	...	...	...
9357	KL_n06.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
9358	KL_n05.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
9359	KL_n26.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
9360	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
9361	KL_n11.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...

9362 rows × 5 columns

```
df_final.to_csv("./final")
```

The development of this consolidated dataset was a major achievement for the project, as it enabled a more streamlined and effective analysis process. By merging the datasets, not only increased the amount of data available but also expanded the range of samples to work with. This enhanced the reliability of future deep learning models that will be built.

After the combined data frame was prepared, it was saved as a CSV file. This file contained all of the processed and merged data from the three datasets and was stored in a bucket on the Google Cloud Platform.

The datasets were successfully combined into one CSV file in the GCP bucket, laying the foundation for subsequent stages of the project. These phases involve extracting features, augmenting data, and training deep learning models. This crucial step ensured that the data was not only carefully pre-processed but also organized in a way that facilitates efficient and effective data analysis.

3.4 Data Transformation

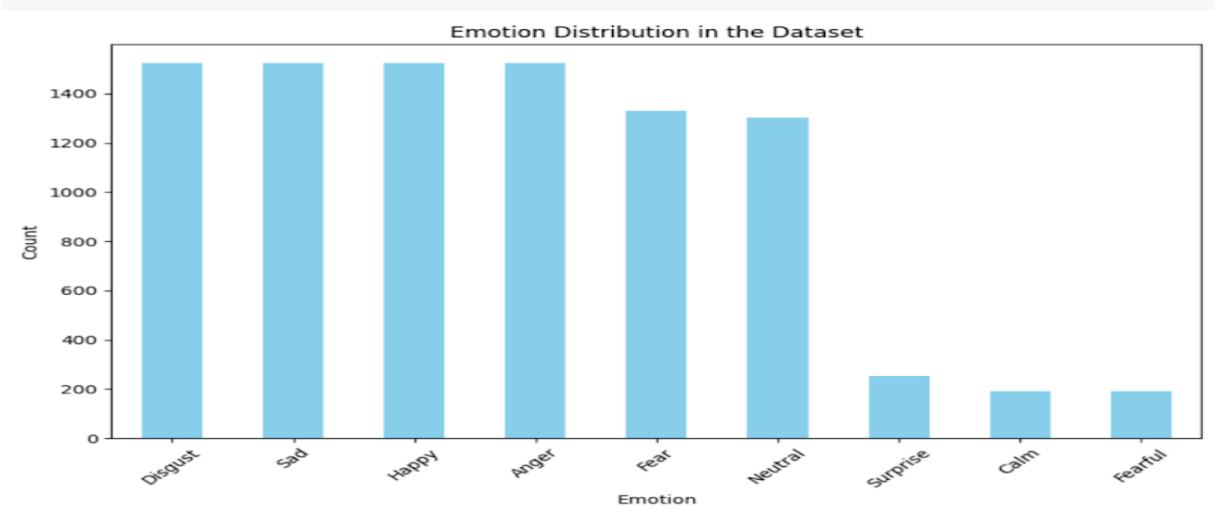
Data Augmentation

Originally, the collated dataset showed class imbalance with lower sample counts for three emotions: Surprise (n = 252), Calm (n = 192), and Fearful (n = 192) as illustrated in Figure 18. There was disproportionately scarce representation of audio samples conveying these affective states relative to other categories, based on the aggregated corpus from the RAVDESS (Livingstone & Russo, 2018), CREMA-D (Cao et al., 2014), and SAVEE (Haq & Jackson, 2010) datasets. This skew potentially hampers modeling, as minority classes are typically harder to learn effectively in speech emotion recognition. Rectifying this imbalance was thus important before leveraging the multimodal dataset for robust analyses.

Figure 36 shows the dataset emotions distribution before data augmentation, there is a clear imbalance of the data that is evident.

Figure 36

Emotions Before Augmentation



The class imbalance in the aggregated dataset posed modeling hurdles (Livingstone & Russo, 2018; Haq & Jackson, 2010; Cao et al., 2014). To mitigate this, data augmentation methods were utilized to synthesize additional samples by transforming the source data, thereby diversifying the corpus. These techniques enlarge the training distribution through input variations, improving generalization capability. Accordingly, augmented data provided crucial balanced representations across originally underrepresented emotions like Surprise, Calm, and Fearful. Enriching rare classes through augmentation addressed classification challenges and enhanced overall robustness of the imbalanced collated dataset.

Augmentation Techniques Employed

Noise Augmentation. In the project, controlled levels of noise were added to audio clips following the approach described by (Salamon & Bello, 2017). This technique involves the

addition of noise to simulate real-world environmental noise during speech. The purpose of noise augmentation was to enhance the robustness of the models to noisy input data. The introduction of various noise levels aimed to improve the model's performance in the presence of noisy speech recordings, making it suitable for real-world applications where environmental noise is common

Time Stretching Augmentation. Time stretching augmentation was applied to the audio clips as part of data augmentation. This technique involved modifying the duration of the audio clips while preserving their pitch and content. By altering the time aspect, variations in speech speed were created. This augmentation method contributed to diversifying the training dataset, allowing the model to learn to recognize emotions expressed at different speech speeds. It also made the model adaptable to variations in speech tempo encountered in real-life scenarios(Salamon & Bello, 2017).

Pitch Shifting Augmentation. Different voice pitches were simulated by adjusting the pitch of audio clips. Pitch shifting augmentation introduced variations in vocal pitch, expanding the range of vocal characteristics present in the dataset. This augmentation method enhanced the model's ability to recognize emotions expressed with variations in vocal pitch, improving its generalization to different speakers and emotional expressions(Salamon & Bello, 2017).

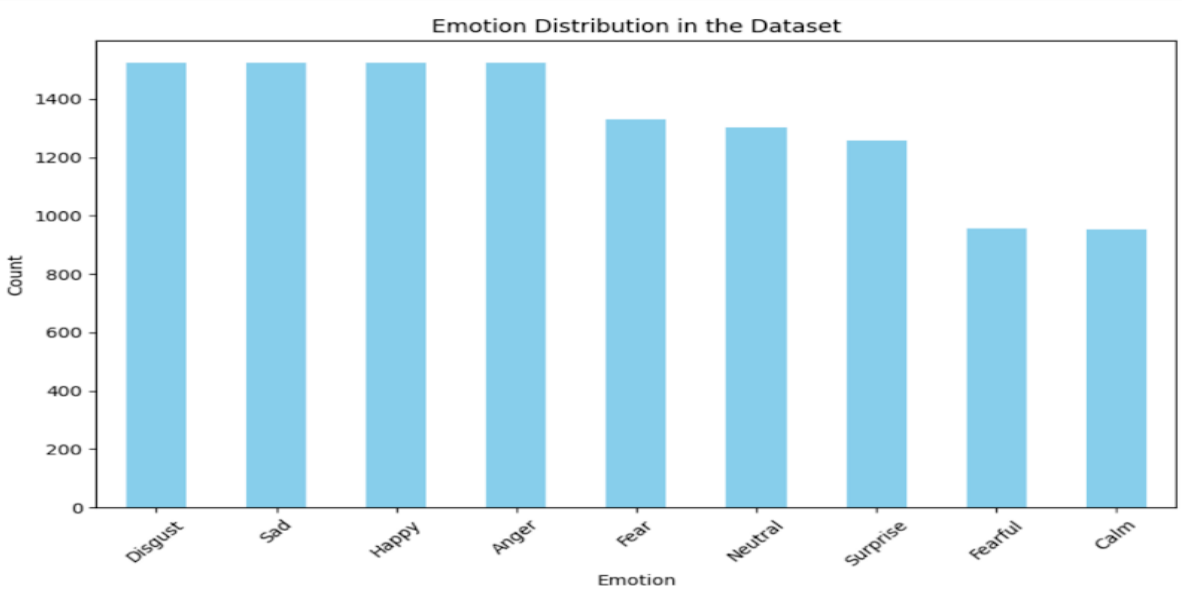
Speech and Pitch Augmentation. This comprehensive augmentation technique combined modifications in both speech speed and pitch. By altering both speech rate and pitch simultaneously, a broader range of variations was introduced into the dataset. The combination of time stretching and pitch shifting provided an extensive set of training examples for the model. It ensured the model's effective capture and differentiation of emotions expressed with

varying speech speeds and vocal pitches, enhancing its versatility in emotion recognition tasks(Salamon & Bello, 2017)..

Data augmentation significantly increased the samples for the minority classes - Surprise (n = 1,256), Calm (n = 952), and Fearful (n = 956) as illustrated in Figure 37.

Figure 37

Emotions After Augmentation

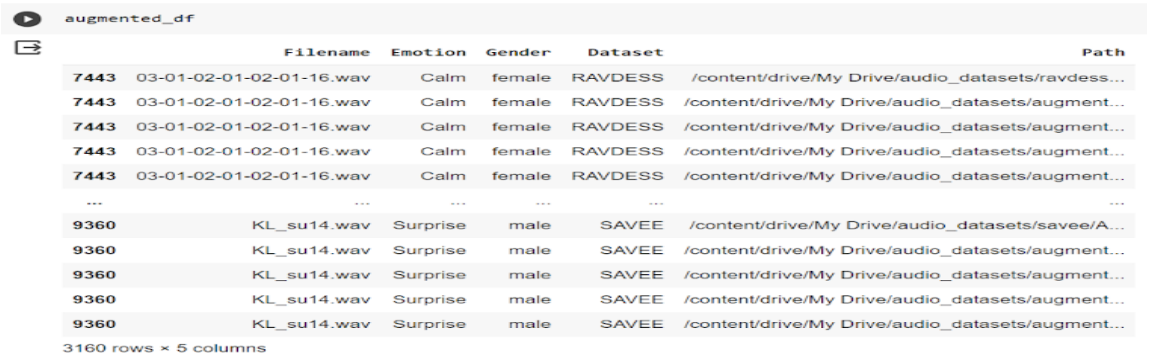


The synthetic audio clips were stored separately in an “augmented data” folder, retaining the original emotion labels for integration. By appending the file paths into the combined CSV metadata as illustrated in Figure 20, the augmented representations could be seamlessly utilized together with the original dataset. Obtaining such balanced distributions across categories (Cao et al., 2014; Haq & Jackson, 2010; Livingstone & Russo, 2018;) is imperative when training nonlinear deep learning architectures for optimal speech emotion recognition performance. The enriched corpus could now effectively leverage complex models to improve classification capabilities.

Figure 38 shows the Augmented data after performing all the techniques

Figure 38

Augmented Data



augmented_df

	Filename	Emotion	Gender	Dataset	Path
7443	03-01-02-01-02-01-16.wav	Calm	female	RAVDESS	/content/drive/My Drive/audio_datasets/ravdess...
7443	03-01-02-01-02-01-16.wav	Calm	female	RAVDESS	/content/drive/My Drive/audio_datasets/augment...
7443	03-01-02-01-02-01-16.wav	Calm	female	RAVDESS	/content/drive/My Drive/audio_datasets/augment...
7443	03-01-02-01-02-01-16.wav	Calm	female	RAVDESS	/content/drive/My Drive/audio_datasets/augment...
7443	03-01-02-01-02-01-16.wav	Calm	female	RAVDESS	/content/drive/My Drive/audio_datasets/augment...
...	...	...	...	...	...
9360	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
9360	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...
9360	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...
9360	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...
9360	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...

3160 rows × 5 columns

Feature Extraction

With class imbalance addressed through data augmentation methods, the next data preparation stage involved feature extraction (Cao et al., 2014; Haq & Jackson, 2010; Livingstone & Russo, 2018;). As a vital preprocessing step before deep learning, multiple acoustic descriptors were derived from the audio samples to represent emotional cues. Extraction helps condense the core affective details, facilitating efficient modeling. The computations produced an extensive set of features encapsulating various speech attributes relevant for recognition.

Methods Used

Zero Crossing Rate (ZCR). The ZCR, as implemented in the project, calculated the speed at which the audio signal transitioned between positive and negative values or vice versa. This feature provided insights into the level of noise in the audio. Specifically, it quantified how frequently the signal changed its polarity, which was indicative of abrupt changes in the audio

waveform, such as consonant sounds or noisy segments(Rezapour Mashhadi & Osei-Bonsu, 2023).

Chroma from Short-Time Fourier Transform (Chroma_stft). Chroma features, derived from the Short-Time Fourier Transform, depicted how energy was distributed across various pitch classes in the audio signal. They offered a description of the tonal content of the audio. Chroma features were valuable for capturing musical characteristics, making them useful for tasks like music genre classification(Rezapour Mashhadi & Osei-Bonsu, 2023).

Mel Frequency Cepstral Coefficients (MFCCs). MFCCs, as widely utilized in speech and audio processing (Rezapour Mashhadi & Osei-Bonsu, 2023), represented the power spectrum of sound over short time intervals. These coefficients effectively captured phonetic characteristics and were used to model the spectral envelope of audio. MFCCs were particularly useful for speech-related tasks, such as speech recognition, as they focused on the spectral details of the audio.

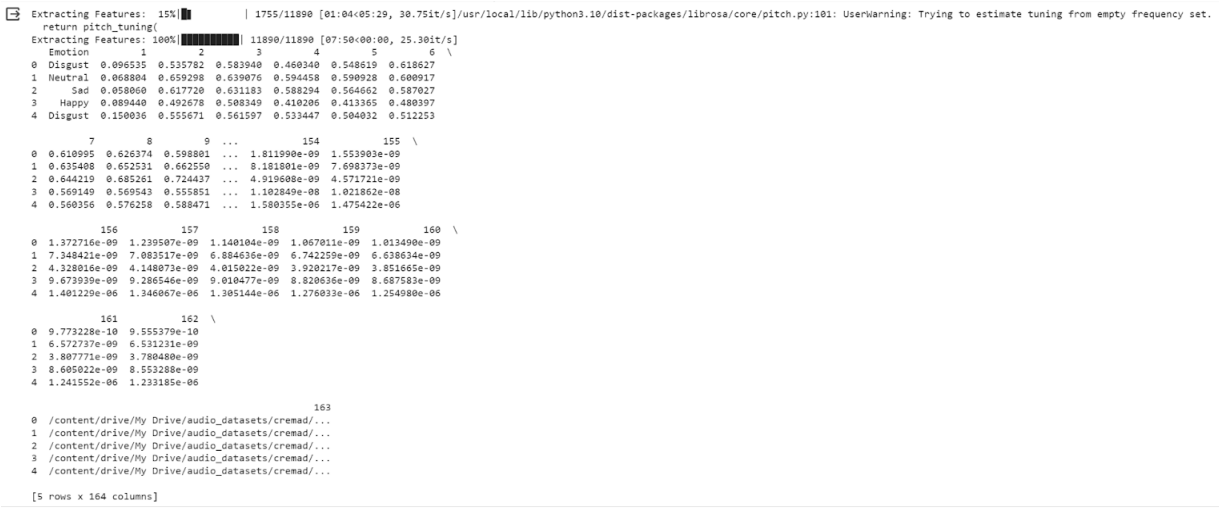
Root Mean Square (RMS). Root Mean Square, implemented in the project according to (Rezapour Mashhadi & Osei-Bonsu, 2023), quantified the energy present in the audio signal. It provided information about the amplitude of the audio. RMS was valuable for assessing the overall intensity or loudness of the audio. Higher RMS values suggested louder segments in the audio, while lower values indicated quieter segments.

Mel Spectrogram Bands. These features documented how power was spread across different frequency bands, similar to what was observed in a Mel spectrogram. This information helped capture the distribution of energy in different frequency regions of the audio spectrum. It was particularly useful for understanding the tonal characteristics and spectral content of the audio(Rezapour Mashhadi & Osei-Bonsu, 2023).

A set of 163 features was calculated for each audio sample, incorporating both temporal and spectral attributes as illustrated in Figure 39. These features are well-suited for training deep learning models that recognize emotions in speech.

Figure 39

Extracted Features



```
Extracting Features:  15%|█▌          | 1755/11890 [01:04<05:29, 30.75it/s]/usr/local/lib/python3.10/dist-packages/librosa/core/pitch.py:101: UserWarning: Trying to estimate tuning from empty frequency set.
  return pitch_tuning(
Extracting Features: 100%|██████████| 11890/11890 [07:50<00:00, 25.30it/s]
   Emotion         1         2         3         4         5         6  \
0  Disgust  0.096535  0.535782  0.583940  0.460340  0.548619  0.618627
1  Neutral  0.068804  0.659298  0.639076  0.594458  0.590928  0.600917
2      Sad  0.058060  0.617720  0.631183  0.588294  0.564662  0.587027
3    Happy  0.089440  0.492678  0.508349  0.410206  0.413365  0.480397
4  Disgust  0.150036  0.555671  0.561597  0.533447  0.504032  0.512253

          7         8         9  ...           154           155  \
0  0.610995  0.626374  0.598801  ...  1.811990e-09  1.553903e-09
1  0.635408  0.652531  0.662550  ...  8.181801e-09  7.698373e-09
2  0.644219  0.685261  0.724437  ...  4.919608e-09  4.571721e-09
3  0.569149  0.569543  0.555851  ...  1.102849e-08  1.021862e-08
4  0.560356  0.576258  0.588471  ...  1.580355e-06  1.475422e-06

           156           157           158           159           160  \
0  1.372716e-09  1.239507e-09  1.140104e-09  1.067011e-09  1.013490e-09
1  7.348421e-09  7.083517e-09  6.884636e-09  6.742259e-09  6.638634e-09
2  4.328016e-09  4.148073e-09  4.015022e-09  3.920217e-09  3.851665e-09
3  9.673939e-09  9.286546e-09  9.010477e-09  8.820636e-09  8.687583e-09
4  1.401229e-06  1.346067e-06  1.305144e-06  1.276033e-06  1.254980e-06

           161           162  \
0  9.773228e-10  9.555379e-10
1  6.572737e-09  6.531231e-09
2  3.807771e-09  3.780480e-09
3  8.605022e-09  8.553288e-09
4  1.241552e-06  1.233185e-06

                                                 163
0  /content/drive/My Drive/audio_datasets/cremad/...
1  /content/drive/My Drive/audio_datasets/cremad/...
2  /content/drive/My Drive/audio_datasets/cremad/...
3  /content/drive/My Drive/audio_datasets/cremad/...
4  /content/drive/My Drive/audio_datasets/cremad/...

[5 rows x 164 columns]
```

The feature generated split is illustrated in Table 17.

Table 17

Summary of Key Features of Datasets

Feature	Feature No
ZCR (Zero Crossing Rate)	0
Chroma_stsft (Chroma from STFT)	1-12
MFCC (Mel Frequency Cepstral Coefficients)	13-25
RMS (Root Mean Square	26
Mel Spectogram Bands	27 - 163

After the completion of feature extraction, the feature vectors for each individual were combined and appended to an existing CSV file as illustrated in Figure 40.

Figure 40

Finalized Dataset With Features

```
[ ] final_df
```

	Filename	Emotion	Gender	Dataset	Path	1	2	3	4	5	...	153	154	155	156	157	158	
0	1079_TIE_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...	0.096535	0.535782	0.583940	0.460340	0.548619	...	2.194198e-09	1.811990e-09	1.553903e-09	1.372716e-09	1.239507e-09	1.140104e-09	1.067
1	1080_DFA_NEU_XX.wav	Neutral	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...	0.068804	0.659298	0.639076	0.594458	0.590928	...	8.833827e-09	8.181801e-09	7.698373e-09	7.348421e-09	7.083517e-09	6.884636e-09	6.742
2	1080_DFA_SAD_XX.wav	Sad	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...	0.058060	0.617720	0.631183	0.588294	0.564662	...	5.419312e-09	4.919608e-09	4.571721e-09	4.328016e-09	4.148073e-09	4.015022e-09	3.920
3	1079_TIE_HAP_XX.wav	Happy	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...	0.089440	0.492678	0.508349	0.410206	0.413365	...	1.226272e-08	1.102849e-08	1.021862e-08	9.673939e-09	9.286546e-09	9.010477e-09	8.820
4	1079_TSI_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...	0.150036	0.555671	0.561597	0.533447	0.504032	...	1.728195e-06	1.580355e-06	1.475422e-06	1.401229e-06	1.346067e-06	1.305144e-06	1.276
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
11885	KL_su07.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...	0.020717	0.597684	0.496242	0.552453	0.607318	...	5.453797e-07	3.058611e-07	1.354868e-07	8.483688e-08	3.766702e-08	2.079202e-08	1.246
11886	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...	0.270788	0.725905	0.759456	0.760370	0.752060	...	9.132795e-04	8.649249e-04	8.881587e-04	8.247387e-04	8.497087e-04	8.944159e-04	9.001
11887	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...	0.041477	0.605494	0.564465	0.526253	0.524834	...	8.205189e-08	3.732608e-08	1.332046e-08	9.244797e-09	9.041458e-09	9.110384e-09	8.722
11888	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...	0.046305	0.560680	0.509567	0.563484	0.551319	...	3.014327e-06	2.679694e-06	1.792788e-06	8.230818e-07	4.088227e-07	1.677608e-07	1.189
11889	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...	0.038113	0.578048	0.549836	0.592483	0.545703	...	7.380036e-07	5.112161e-07	3.984859e-07	2.603008e-07	2.007248e-07	1.812377e-07	1.787

11890 rows × 167 columns

This file contains details about the audio files such as their emotions and paths, along with the newly added audio features. This comprehensive dataset, consisting of both metadata and extracted features, was then utilized for additional data processing and training models.

Normalization and Label Encoding

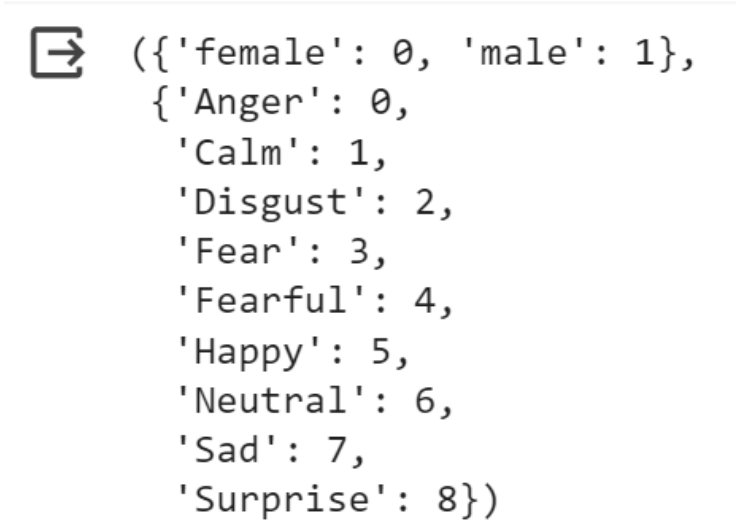
Following the feature extraction process, proceeded with two crucial data preprocessing steps to prepare the dataset for deep learning models (Cao et al., 2014; Haq & Jackson, 2010; Livingstone & Russo, 2018). The first step involved row-wise Z-score normalization(Fei, Gao, Lu, & Xiang, 2021) , a fundamental technique for ensuring that a dataset's features are consistently scaled. By standardizing the features through this normalization procedure, eliminate potential biases and variations arising from differences in magnitudes among different features. This standardized format plays a crucial role in deep learning models by facilitating faster convergence and enhancing overall model performance.

The second task was label encoding, applied to categorical columns such as 'Gender' and 'Emotions' in the dataset. Label encoding is a method used to convert categorical data into numerical format, a prerequisite for feeding data into deep learning algorithms. In the case of 'Gender,' the encoding was straightforward, with 'female' assigned the value 0 and 'male' assigned 1. It is illustrated in Figure 41.

However, the 'Emotions' column required more extensive label encoding to represent the various emotional categories effectively. Each emotion category was assigned a numerical label as follows: 'Anger' (0), 'Calm' (1), 'Disgust' (2), 'Fear' (3), 'Fearful' (4), 'Happy' (5), 'Neutral' (6), 'Sad' (7), and 'Surprise' (8). These numerical representations of gender and emotions enable the seamless integration of these features into architectures for speech emotion recognition. Together, the normalization and encoding preprocessings significantly enhance robustness and performance when detecting affect in speech.

Figure 41

Label Encoding



```
({'female': 0, 'male': 1},
 {'Anger': 0,
  'Calm': 1,
  'Disgust': 2,
  'Fear': 3,
  'Fearful': 4,
  'Happy': 5,
  'Neutral': 6,
  'Sad': 7,
  'Surprise': 8})
```

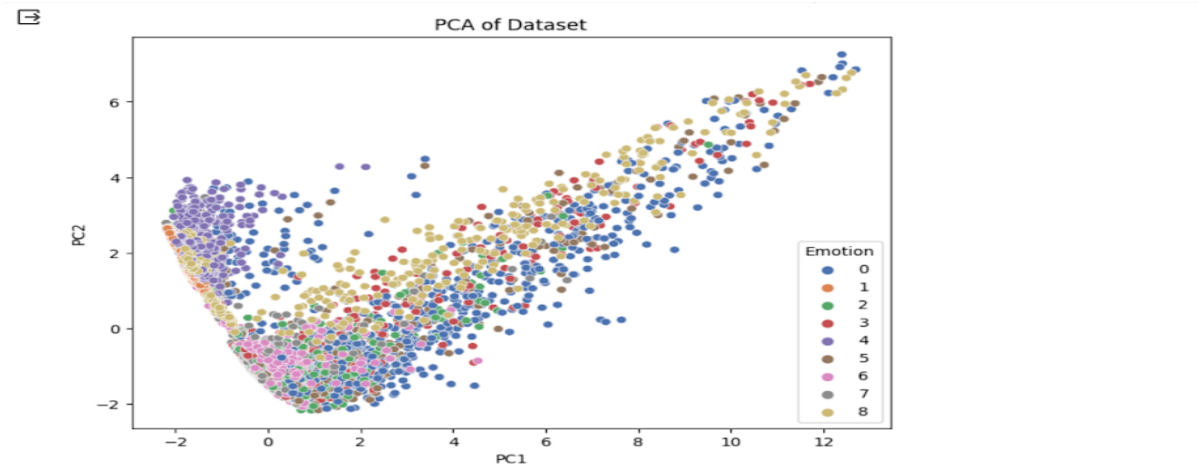
PCA

After encoding, Principal Component Analysis (PCA) was utilized as a dimensionality reduction technique to transform the high-dimensional dataset into a lower-dimensional feature space while retaining maximal variance. As PCA simplifies complex data and mitigates multicollinearity, it boosts model performance. The PCA module from scikit-learn was leveraged(Tsalera, Papadakis, & Samarakou, 2021).

First, the target emotion variable and features were segregated, excluding the encoded gender column. PCA was applied to the features, reducing to two principal components for visualization through a scatter plot. Each point represented an audio sample, colored by emotion category. This allowed understanding PCA's ability to cluster the data by affective labels in the reduced space. Specifying two components enabled insightful visualization for preliminary analysis while maximally preserving variance from the initial feature set as illustrated in Figure 42.

Figure 42

PCA Of Dataset

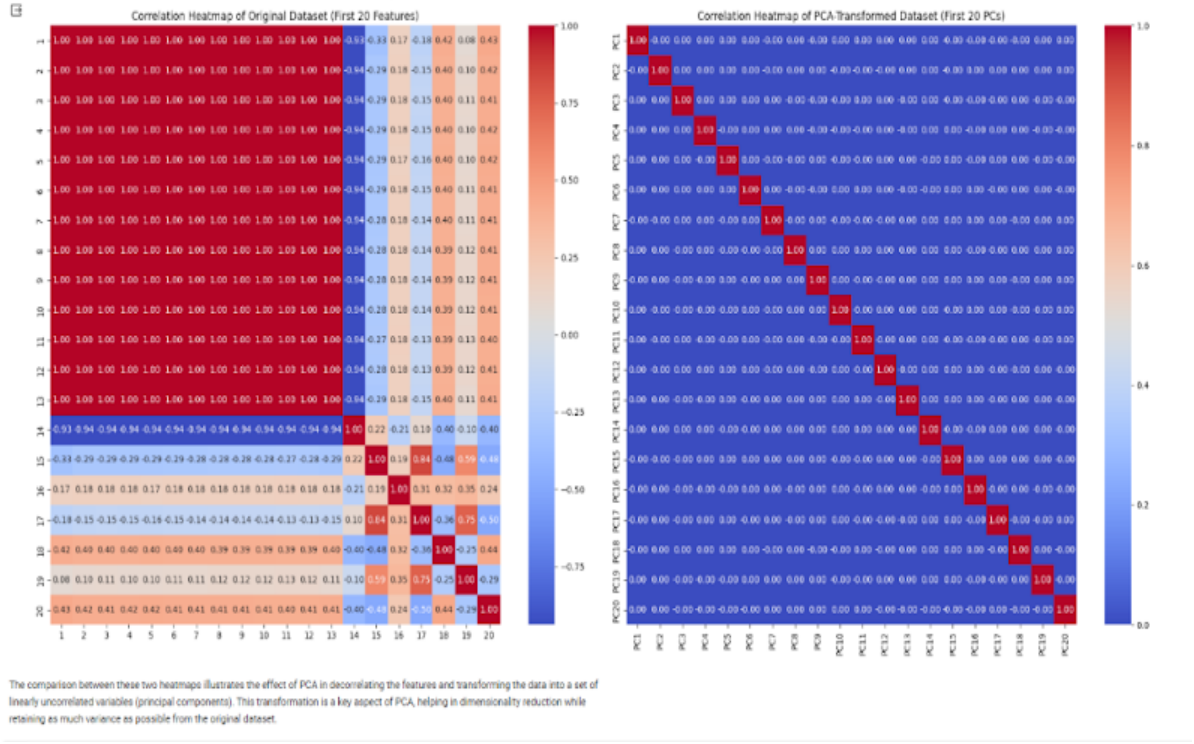


The scatter plot displays the data after applying PCA, reducing the dimensionality to two principal components. This plot shows how the samples are distributed in the transformed space, with colors still representing the 'Emotion' category.

To evaluate the efficacy of PCA, a comparative examination was conducted by analyzing correlation heatmaps prior to and post transformation. The initial heatmap depicted the correlation pattern of the original dataset, emphasizing the first 20 characteristics. Conversely, the secondary heatmap exhibited correlations among the first 20 principal components within the dataset after being transformed by PCA which is illustrated in Figure 43.

Figure 43

Correlation Heatmap For Original And PCA Dataset



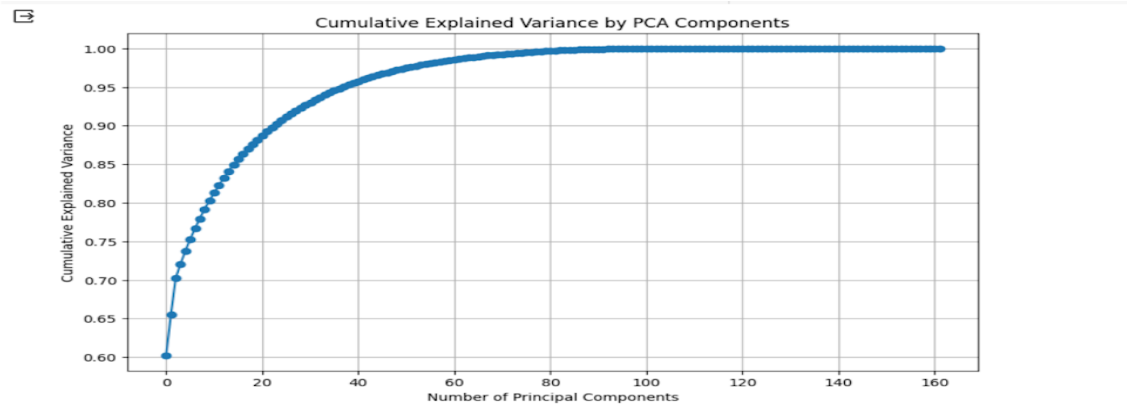
This analysis enabled observations on how PCA efficiently reduced dimensionality by decorrelating features and converting them into linearly uncorrelated variables as a crucial aspect(Tsalera, Papadakis, & Samarakou, 2021).

The analysis also considered the cumulative explained variance by the principal components.

A graph in Figure 44 was used to illustrate how the number of principal components relates to the cumulative explained variance.

Figure 44

Number Of Components And The Amount Of Information (variance) Retained From The Original Dataset.



This visualization provides a clear picture of the trade-off between the number of components and the amount of information (variance) retained from the original dataset. For instance, if the curve flattens out quickly, it indicates that most of the variance can be captured by a relatively small number of components. This would imply that PCA has effectively reduced the dimensionality of the data while retaining most of its information

This visual representation highlighted the trade-off between reducing dimensions and retaining information. If a small number of components captured most of the dataset's variance quickly, it indicated that dimensionality reduction effectively preserved crucial information. This understanding guided the decision on how many principal components to keep for subsequent modeling, ensuring a balance between reduced dimensions and retained data (Tsalera, Papadakis, & Samarakou, 2021).

PCA has been instrumental in decreasing the dataset's dimensionality without compromising vital information. By decorrelating features, it enables more effective training of deep learning models and improves performance in speech emotion recognition systems (Tsalera, Papadakis, & Samarakou, 2021).

L2 REGULARIZATION

In the next step of the data preprocessing pipeline, implemented L2 regularization on the feature set. Regularization is an essential technique used to prevent overfitting in deep learning models, particularly linear regression models. Overfitting occurs when a model fits the training data too closely, including noise, leading to poor generalization on unseen data. L2 regularization, also known as Ridge Regression, introduces a penalty term into the standard linear regression loss function. This term penalizes large coefficients and effectively limits their magnitudes (Hastie et al., 2009).

Linear regression models without regularization can generate coefficients that are significantly large, resulting in complex models that incorporate noise from the training data. Although they may perform well on the training data, their ability to effectively generalize and predict new, unseen data is often limited.

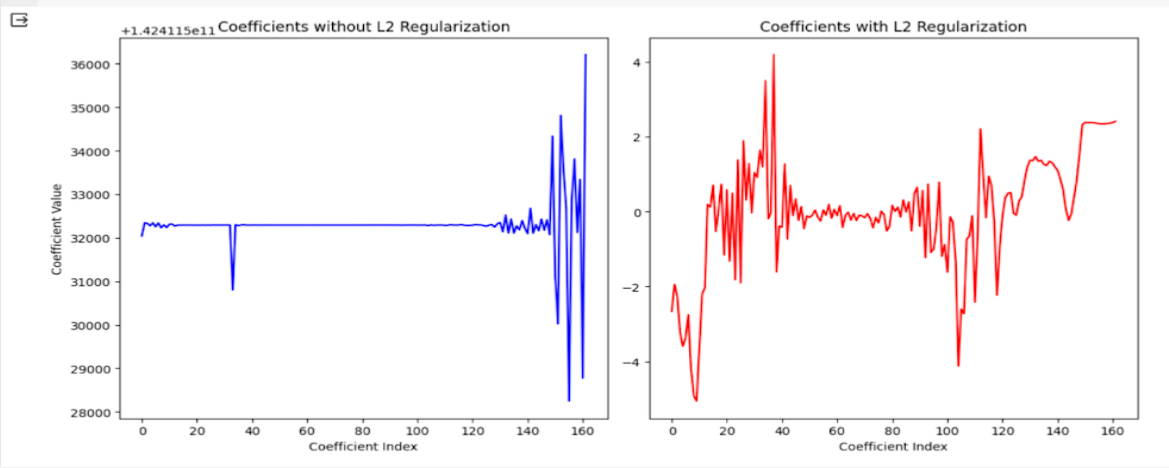
Ridge Regression tackles this problem by incorporating L2 regularization, which involves adding a penalty term to the loss function. The strength of the penalty is determined by the hyperparameter alpha (α), with higher values leading to stronger regularization. By encouraging smaller coefficients, Ridge Regression prevents them from growing excessively large. Consequently, Ridge Regression models find a balance between accurately fitting the training data and controlling model complexity. These models are highly valuable in deep learning due to their ability to improve generalization and reduce overfitting (Hastie et al., 2009).

An experiment was conducted to demonstrate the effect of L2 regularization using Ridge Regression. The alpha (α) value used for this experiment was set at 1.0. By comparing the obtained coefficients with and without regularization, the impact can be observed.

Two plots are provided to visualize these coefficients as illustrated in Figure 45.

Figure 45

With And Without L2 Regularization



The first plot shows the coefficients without any form of regularization, emphasizing their potential for large values. In contrast, the second plot showcases how applying L2 regularization results in moderated magnitudes for these coefficients

The use of this regularization method greatly improves the ability of the model to generalize, ensuring accurate detection of emotional speech in real-life situations.

3.5 Data Preparation

In the subsequent important phase of data preparation, a split was implemented between the training, validation, and test datasets derived from the processed dataset. This systematic approach establishes a framework for effectively training and evaluating speech emotion recognition models. The division of data plays a crucial role in assessing model performance,

mitigating overfitting risks, and ensuring reliable generalization to new data. The training dataset holds the majority share with 7134 samples (60% of the total) as illustrated in Figure 46. This significant portion is utilized for deep learning model training to enable them to grasp underlying patterns and characteristics within the speech data.

Figure 46

Train Data

```
[ ] train
```

	Emotion	Gender	1	2	3	4	5	6	7	8	...	153	154	155	156	157	158	159	160	161	162
11263	8	0	0.083723	0.095257	0.094646	0.094185	0.093809	0.093561	0.094598	0.095515	...	0.079088	0.079087	0.079087	0.079086	0.079086	0.079088	0.079088	0.079088	0.079087	0.079084
6086	0	0	-0.063815	-0.046107	-0.043569	-0.041709	-0.046247	-0.046004	-0.046272	-0.047128	...	-0.066630	-0.066630	-0.066630	-0.066630	-0.066630	-0.066630	-0.066630	-0.066630	-0.066630	-0.066630
1108	3	1	0.040568	0.058620	0.056955	0.054936	0.054544	0.056179	0.056718	0.059390	...	0.039047	0.039047	0.039047	0.039047	0.039047	0.039047	0.039047	0.039047	0.039047	0.039047
11432	4	1	0.079417	0.090018	0.091723	0.092232	0.091355	0.090291	0.089395	0.088697	...	0.075532	0.075531	0.075530	0.075530	0.075530	0.075529	0.075529	0.075529	0.075529	0.075528
8225	1	1	0.069623	0.081826	0.082337	0.082648	0.082995	0.082383	0.080354	0.079225	...	0.066825	0.066825	0.066826	0.066826	0.066826	0.066826	0.066827	0.066826	0.066826	0.066825
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7808	5	0	0.095076	0.102212	0.102538	0.103153	0.102793	0.101325	0.101315	0.101576	...	0.090056	0.090053	0.090053	0.090050	0.090047	0.090045	0.090045	0.090044	0.090044	0.090044
7414	3	0	0.043391	0.058768	0.059424	0.057517	0.056983	0.057918	0.057887	0.057835	...	0.041714	0.041714	0.041714	0.041714	0.041714	0.041714	0.041714	0.041714	0.041714	0.041714
404	7	0	0.043673	0.058165	0.059850	0.057194	0.057180	0.056699	0.058109	0.058458	...	0.042663	0.042663	0.042663	0.042663	0.042663	0.042663	0.042663	0.042663	0.042663	0.042663
113	5	1	0.035674	0.052727	0.052193	0.054615	0.054806	0.054482	0.051815	0.051566	...	0.034322	0.034322	0.034322	0.034322	0.034322	0.034322	0.034322	0.034322	0.034322	0.034322
2088	2	0	0.022672	0.041469	0.040571	0.041211	0.038752	0.042083	0.044380	0.041272	...	0.019699	0.019699	0.019699	0.019699	0.019699	0.019699	0.019699	0.019699	0.019699	0.019699

7134 rows × 164 columns

During the training process, a validation dataset is used as an intermediate checkpoint. This dataset, which represents 20% of the validation data with 2378 samples as illustrated in Figure 47.

Figure 47

Validation Data

```
[ ] validate
```

	Emotion	Gender	1	2	3	4	5	6	7	8	...	153	154	155	156	157	158	159	160	161	162
526	7	1	0.039045	0.054515	0.055472	0.054143	0.053431	0.054224	0.054876	0.055825	...	0.038206	0.038206	0.038206	0.038206	0.038206	0.038206	0.038206	0.038206	0.038206	0.038206
9847	1	0	0.085501	0.091375	0.091610	0.092281	0.093455	0.092254	0.092182	0.093158	...	0.081595	0.081595	0.081595	0.081596	0.081595	0.081595	0.081595	0.081595	0.081595	0.081595
6943	2	0	0.040250	0.054475	0.054325	0.053533	0.054943	0.055665	0.055076	0.056284	...	0.038198	0.038198	0.038198	0.038198	0.038198	0.038198	0.038198	0.038198	0.038198	0.038198
8408	6	1	0.063844	0.073195	0.073346	0.072997	0.072493	0.073109	0.072962	0.071968	...	0.060215	0.060215	0.060215	0.060215	0.060215	0.060215	0.060215	0.060215	0.060215	0.060215
3720	5	0	0.025608	0.046766	0.046469	0.042353	0.046044	0.043981	0.043728	0.046818	...	0.023226	0.023226	0.023226	0.023226	0.023226	0.023226	0.023226	0.023226	0.023226	0.023226
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
5390	0	1	-0.206567	-0.190536	-0.188511	-0.193100	-0.193549	-0.191883	-0.190334	-0.194356	...	-0.208764	-0.208764	-0.208764	-0.208764	-0.208764	-0.208764	-0.208764	-0.208764	-0.208764	-0.208764
1318	7	1	0.037868	0.051519	0.052422	0.051943	0.053184	0.053500	0.054398	0.055815	...	0.036156	0.036156	0.036156	0.036156	0.036156	0.036156	0.036156	0.036156	0.036156	0.036156
1961	7	0	0.041326	0.054347	0.054388	0.058071	0.055634	0.051986	0.053101	0.055146	...	0.039854	0.039854	0.039854	0.039854	0.039854	0.039854	0.039854	0.039854	0.039854	0.039854
6497	2	0	0.041858	0.057366	0.059317	0.056617	0.054833	0.056250	0.055165	0.055017	...	0.040345	0.040345	0.040345	0.040345	0.040345	0.040345	0.040345	0.040345	0.040345	0.040345
8726	7	1	0.076964	0.085789	0.085814	0.086034	0.086314	0.084522	0.083837	0.084414	...	0.072109	0.072108	0.072108	0.072108	0.072108	0.072108	0.072108	0.072108	0.072108	0.072108

2378 rows × 164 columns

helps in adjusting hyperparameters, monitoring model performance, and identifying signs of overfitting. Its role is crucial in ensuring that models achieve both accuracy and generalization.

On the other hand, there is also a separate test dataset consisting of 2378 samples (also representing 20% of the total data) as illustrated in Figure 48.

Figure 48

Test Data

test

	Emotion	Gender	1	2	3	4	5	6	7	8	...	153	154	155	156	157	158	159	160	161	162
1476	5	0	0.032398	0.048557	0.046277	0.045948	0.047807	0.052208	0.051210	0.049711	...	0.030082	0.030082	0.030082	0.030082	0.030082	0.030082	0.030082	0.030082	0.030082	0.030082
2961	6	1	0.037644	0.053160	0.054087	0.054813	0.054048	0.057037	0.056212	0.057909	...	0.036193	0.036193	0.036193	0.036193	0.036193	0.036193	0.036193	0.036193	0.036193	0.036193
9132	7	1	-0.001118	0.016673	0.018208	0.016559	0.015293	0.016166	0.017986	0.016642	...	-0.001505	-0.001505	-0.001505	-0.001505	-0.001505	-0.001505	-0.001505	-0.001505	-0.001505	-0.001505
9683	8	1	0.074808	0.083877	0.084786	0.085330	0.084303	0.083718	0.085078	0.085350	...	0.071333	0.071333	0.071333	0.071332	0.071332	0.071333	0.071333	0.071332	0.071332	0.071332
2074	7	0	0.041783	0.060050	0.057738	0.056869	0.056789	0.058022	0.057085	0.057919	...	0.040721	0.040721	0.040721	0.040721	0.040721	0.040721	0.040721	0.040721	0.040721	0.040721
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
7106	7	0	0.056145	0.074636	0.075630	0.073454	0.075085	0.074778	0.075035	0.074568	...	0.053963	0.053963	0.053963	0.053963	0.053963	0.053963	0.053963	0.053963	0.053963	0.053963
1756	6	0	0.040463	0.053648	0.054013	0.054297	0.053512	0.052503	0.055092	0.057292	...	0.038628	0.038628	0.038628	0.038628	0.038628	0.038628	0.038628	0.038628	0.038628	0.038628
9233	7	1	-0.197357	-0.180776	-0.182331	-0.183617	-0.183842	-0.180507	-0.178545	-0.179397	...	-0.199194	-0.199196	-0.199196	-0.199196	-0.199196	-0.199197	-0.199197	-0.199197	-0.199197	-0.199197
6679	2	1	0.044975	0.060728	0.060877	0.060287	0.060068	0.059783	0.058304	0.058088	...	0.040831	0.040831	0.040831	0.040831	0.040831	0.040831	0.040831	0.040831	0.040831	0.040831
3073	6	1	0.040353	0.060168	0.061551	0.059317	0.057946	0.057792	0.057925	0.058998	...	0.038913	0.038913	0.038913	0.038913	0.038913	0.038913	0.038913	0.038913	0.038913	0.038913

2378 rows × 164 columns

Unlike the validation dataset, this test dataset remains untouched during training and validation phases. Its purpose is solely to provide an unbiased evaluation of the trained and fine-tuned model's performance. By testing on completely unseen data simulating real-world scenarios, it accurately assesses how well the model can recognize emotions in speech.

Moreover, examples from each of these datasets have been included to offer a glimpse into the emotional distribution within them. This transparent approach in showcasing dataset samples underscores the diversity and significance of the data subsets, which play a crucial role in ensuring robust model training and evaluation.

With thorough preparation of these datasets and presentation of their samples, have paved the way for subsequent phases of this project. focus will be on training and testing deep learning models such as Convolutional Neural Networks, Recurrent Neural Networks, Long Short-Term Memory networks, and Autoencoders in speech emotion recognition.

3.6 Data Statistics

This section provides a summary of the data preparation outcomes, which include transforming raw datasets into pre-processed and prepared datasets. The findings are presented using statistical visualization techniques to facilitate comprehension of how the data has evolved.

Raw Dataset

The CREMA-D dataset comprises 7442 samples, while the RAVDESS dataset contains 1440 samples, and the SAVEE dataset includes 480 samples (Cao et al., 2014; Haq & Jackson, 2010; Livingstone & Russo, 2018;).These datasets exhibit variations in the distribution of emotions, reflecting the diversity of emotional expressions captured in the raw data.

The Raw DataFrame, as presented in Figure 49, is a structure from the pandas library within Python, characterized by a RangeIndex spanning 9,362 entries from 0 to 9,361.

Figure 49

Raw Data Frame

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9362 entries, 0 to 9361
Data columns (total 5 columns):
 #   Column    Non-Null Count  Dtype 
---  ------    --------------  ----- 
 0   Filename  9362 non-null   object
 1   Emotion   9362 non-null   object
 2   Gender    9362 non-null   object
 3   Dataset   9362 non-null   object
 4   Path      9362 non-null   object
dtypes: object(5)
memory usage: 365.8+ KB
None
```

It encompasses a total of five columns, each populated with non-null entries indicative of a fully occupied dataset. The columns are denoted as 'Filename', 'Emotion', 'Gender', 'Dataset', and 'Path', all stored as object data types—a common representation for textual data in pandas. This structure consumes approximately 365.8+ KB of memory, reflecting the substantial volume of data contained within. The information provided by the DataFrame is crucial for further data manipulation and analysis tasks, offering insights into the dataset's composition and the nature of the stored variables.

Pre-processed Datasets

After preprocessing, all the datasets remained intact without any missing values. The data was standardized and basic analysis was conducted to understand the characteristics of the dataset. For example, in the CREMA-D dataset, there were 7442 samples with a distribution of emotions including Disgust, Sadness, Happiness, Anger (1523 samples each), Fear (1331 samples), and Neutral (1303 samples) which is illustrated in Figure 50.

Figure 50

Pre Processed CREMA Dataset

```
CREMA-D Dataset Analysis:
Total files: 7442
Emotion counts:
 Disgust    1271
Sad         1271
Happy       1271
Anger       1271
Fear        1271
Neutral     1087
Name: Emotion, dtype: int64
```

Similarly, in the RAVDESS dataset comprising 1440 samples is illustrated in Figure 51.

Figure 51

Pre Processed SAVEE Datset

```
SAVEE Dataset Analysis:
Total files: 480
Emotion counts:
 Neutral     120
Anger        60
Disgust      60
Happy        60
Fear         60
Sad          60
Surprise     60
Name: Emotion, dtype: int64
```

and in the SAVEE dataset consisting of 480 samples illustrated in Figure 52.

Figure 52

Pre Processed RAVDESS Dataset

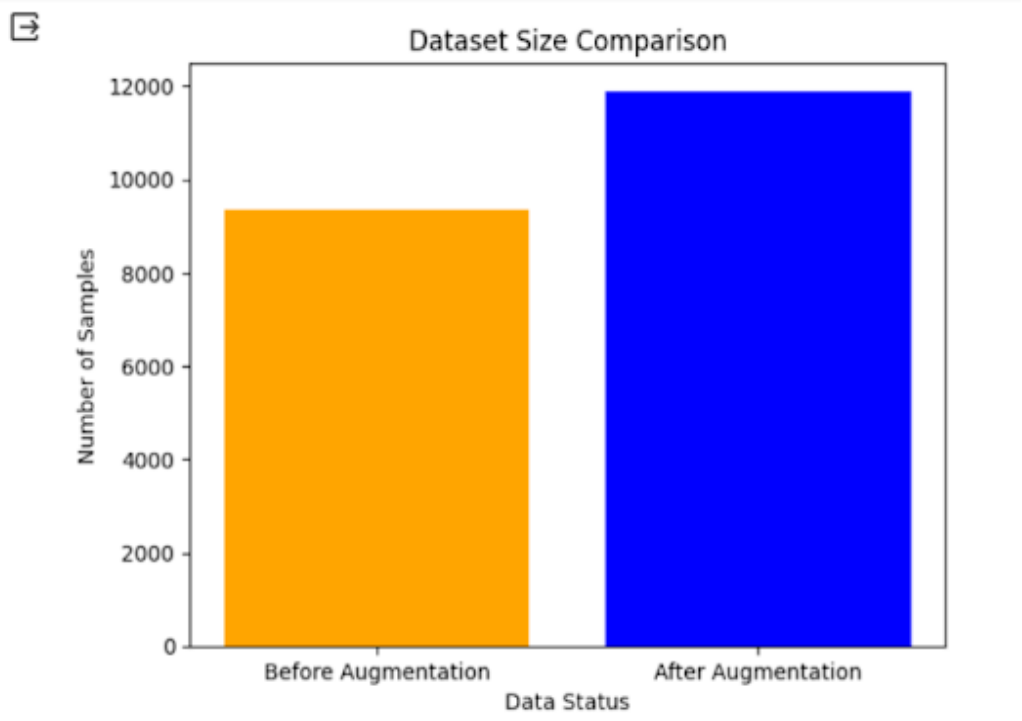
```
RAVDESS Dataset Analysis:
Total files: 1440
Emotion counts:
 Happy       192
Calm         192
Fearful      192
Anger        192
Sad          192
Disgust      192
Surprise     192
Neutral       96
Name: Emotion, dtype: int64
```

Transformed Dataset

Initially, the dataset showed an evident disparity in sample distribution across emotion categories. The original counts were 252 for Surprise, 192 for Calm, and 192 for Fearful. After applying data augmentation techniques, there was a substantial increase in the number of samples: Surprise now had a count of 1256, Calm with 952 samples, and Fearful at 956. This approach successfully addressed the imbalance issue in the dataset by ensuring a more equitable representation of emotions. Prior to augmentation, there were a total of 9,362 samples present in the dataset. However, after implementing augmentation techniques it expanded to contain a total of 11,890 samples which significantly increased available data volume as shown in Figure 53.

Figure 53

Transformed Dataset Comparison



Double-click (or enter) to edit

The datasets were transformed by extracting various audio features such as Zero Crossing , Chroma from Short-Time Fourier Transform, Mel Frequency Cepstral Coefficients, Root Mean Square, and Mel Spectrogram bands(Rezapour Mashhadi & Osei-Bonsu, 2023). This feature extraction process resulted in a dataset consisting of 163 features for each sample and have been added to the finalized csv data as illustrated in Figure 54.

Figure 54

Transformed Dataset

	Filename	Emotion	Gender	Dataset	Path	1	2	3	4	5	...	153	154	155	156	157	
0	1079_TIE_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...	0.096535	0.535782	0.583940	0.460340	0.548619	...	2.194198e-09	1.811990e-09	1.553903e-09	1.372716e-09	1.239507e-09	1.14
1	1080_DFA_NEU_XX.wav	Neutral	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...	0.068804	0.659298	0.639076	0.594458	0.590928	...	8.833827e-09	8.181801e-09	7.698373e-09	7.348421e-09	7.083517e-09	6.88
2	1080_DFA_SAD_XX.wav	Sad	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...	0.058060	0.617720	0.631183	0.588294	0.564662	...	5.419312e-09	4.919608e-09	4.571721e-09	4.328016e-09	4.148073e-09	4.01
3	1079_TIE_HAP_XX.wav	Happy	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...	0.089440	0.492678	0.508349	0.410206	0.413365	...	1.226272e-08	1.102849e-08	1.021862e-08	9.673939e-09	9.286546e-09	9.01
4	1079_TSI_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...	0.150036	0.555671	0.561597	0.533447	0.504032	...	1.728195e-06	1.580355e-06	1.475422e-06	1.401229e-06	1.346067e-06	1.30
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
11885	KL_su07.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...	0.020717	0.597684	0.496242	0.552453	0.607318	...	5.453797e-07	3.058611e-07	1.354868e-07	8.483688e-08	3.766702e-08	2.07
11886	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...	0.270788	0.725905	0.759456	0.760370	0.752060	...	9.132795e-04	8.649249e-04	8.881587e-04	8.247387e-04	8.497087e-04	8.94
11887	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...	0.041477	0.605494	0.564465	0.526253	0.524834	...	8.205189e-08	3.732608e-08	1.332046e-08	9.244797e-09	9.041458e-09	9.11
11888	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...	0.046305	0.560680	0.509567	0.563484	0.551319	...	3.014327e-06	2.679694e-06	1.792788e-06	8.230818e-07	4.088227e-07	1.67
11889	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...	0.038113	0.578048	0.549836	0.592483	0.545703	...	7.380036e-07	5.112161e-07	3.984859e-07	2.603008e-07	2.007248e-07	1.81

11890 rows × 167 columns

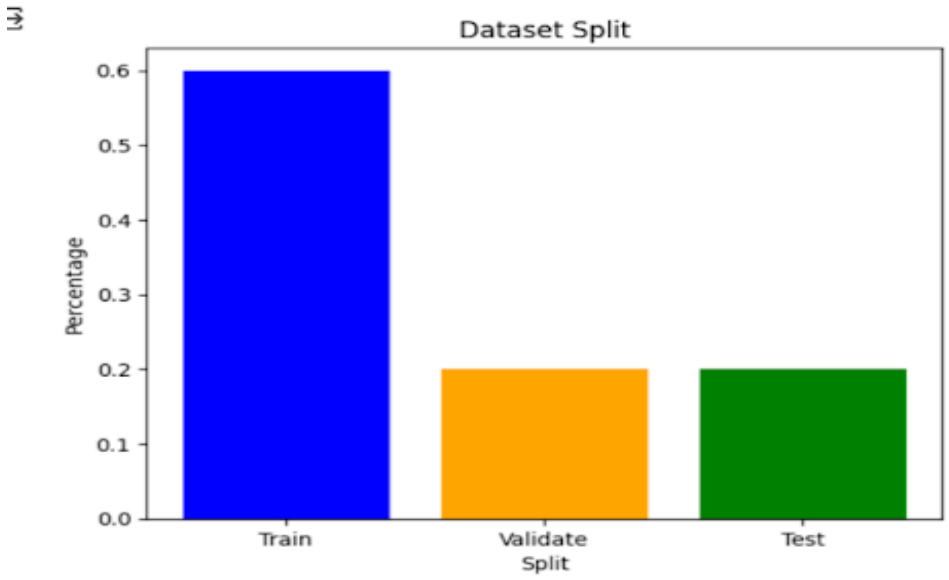
Prepared Dataset

The dataset was divided into training, validation, and test sets with sample sizes of 7134, 2378, and 2378.

Figure 55 provides a visual representation of the dataset distribution with a specific allocation for training, validation, and testing subsets.

Figure 55

Transformed Dataset Comparison



The dataset is divided with 60% dedicated to the training set, as indicated by the prominent blue bar. This majority segment is utilized for the initial development of the model, allowing the algorithm to learn and identify patterns. The validation and testing sets each comprise 20% of the dataset, depicted by the orange and green bars, respectively. The validation set is used to fine-tune the model parameters and to provide an unbiased evaluation of a model fit during the training phase. The testing set is reserved for the final evaluation, assessing the model's performance on new, unseen data to gauge its predictive accuracy and generalization to real-world scenarios.

3.7 Data Analytics Results

This section showcases a range of data analytics findings utilizing different visualizations for big data. These results provide valuable insights into the Enhanced Speech Emotion Recognition project.

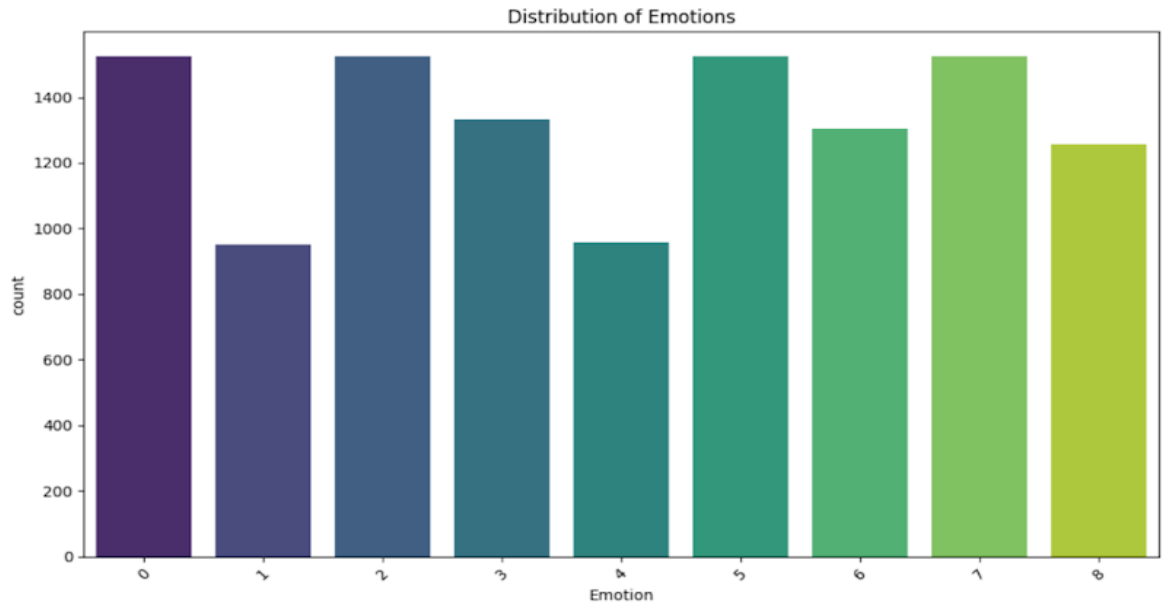
Emotional Distribution

A key focus of the project is comprehending the distribution of emotions within the dataset. Following the process of data transformation, an analysis was conducted on how emotions were distributed in Figure 56. The visualization displayed a more equal representation of emotions, indicating significant progress compared to the initial unbalanced dataset .

This even distribution is illustrated in Figure 56 and holds great importance in training reliable models.

Figure 56

Distribution Of Emotions

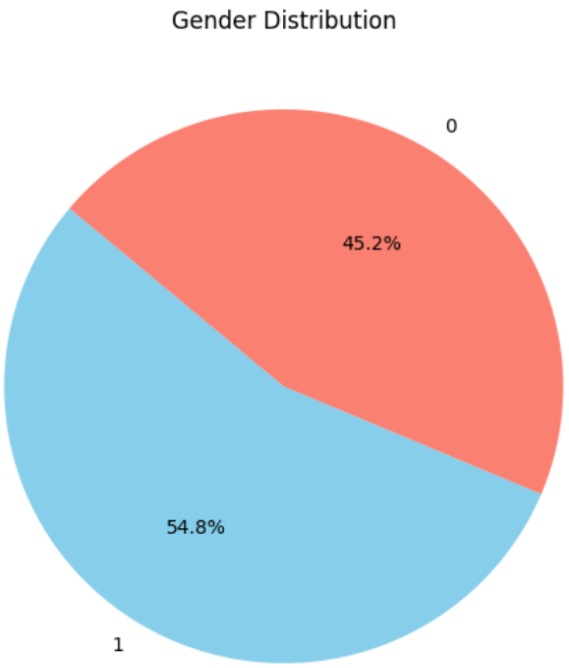


Gender distribution

The dataset includes a diverse representation of genders, which can influence speech patterns and emotions. analysis revealed that among the samples, 54.8% were male and 45.2% were female. Examining the gender distribution is important for uncovering potential gender-related trends in speech emotion recognition. It is illustrated in Figure 57.

Figure 57

Distribution of Gender



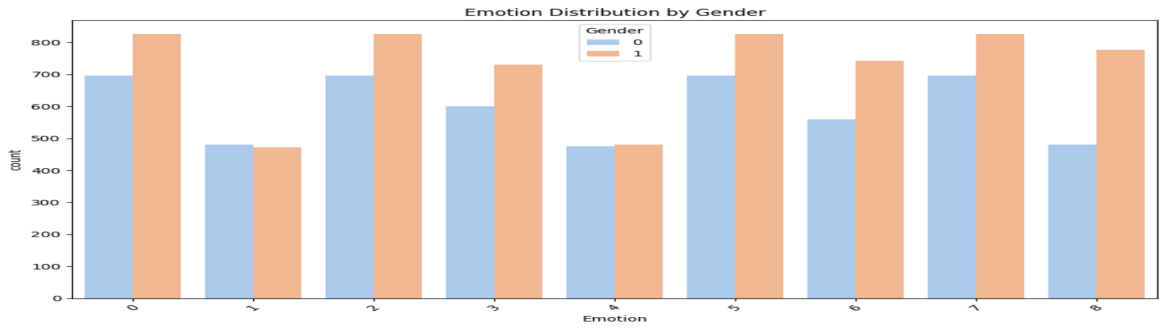
Stacked Bar Chart - Emotion Distribution by Gender

To gain further insights, created a visual representation using a stacked bar chart to illustrate how different emotions are distributed across males and females. This visualization provides valuable information on how specific emotions vary between genders, allowing for

more tailored approaches in emotion recognition models based on these nuances. It is illustrated in Figure 58.

Figure 58

Emotion Distribution By Gender

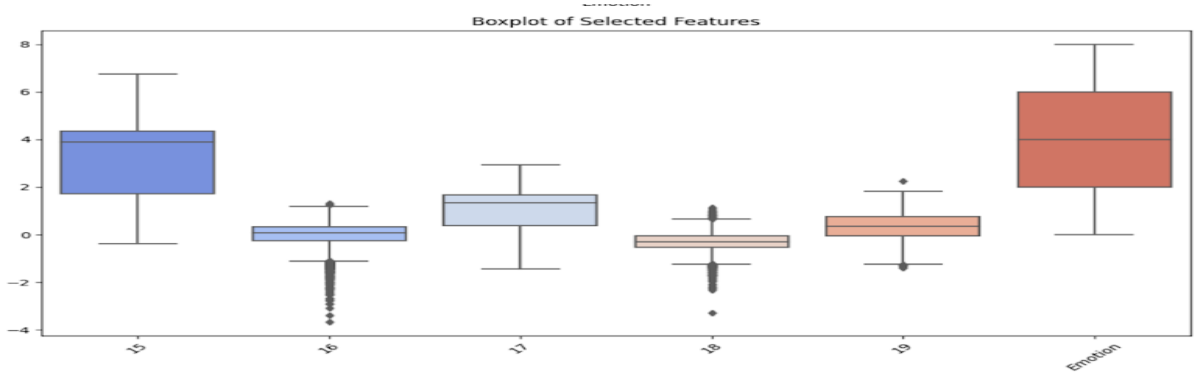


Box Plot of Chosen Features

In order to gain a better understanding of the dataset, generated box plots for select features. These visual representations reveal the distribution and potential anomalies in the data, which can be helpful when selecting features and developing models. It is illustrated in Figure 59.

Figure 59

Box Plot Of Selected Features

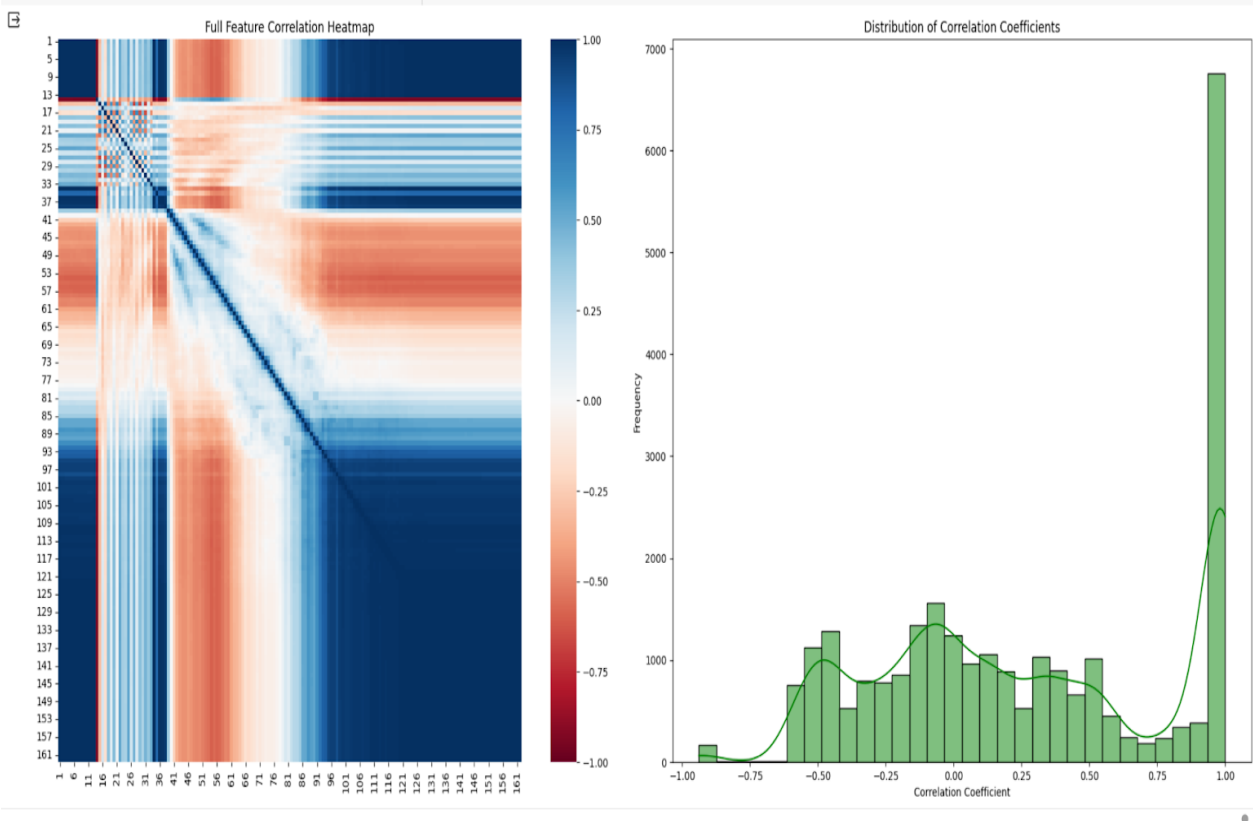


Full Feature Correlation Heatmap

A heatmap illustrating the correlation between dataset features is in Figure 60.

Figure 60

Full Feature Correlation Heatmap And Distribution Of Correlation Coefficients



with blue to red indicating strong positive to strong negative correlation, and a histogram showing the distribution of these correlation coefficients, most clustering around zero which implies little to no linear relationship between most feature pairs.

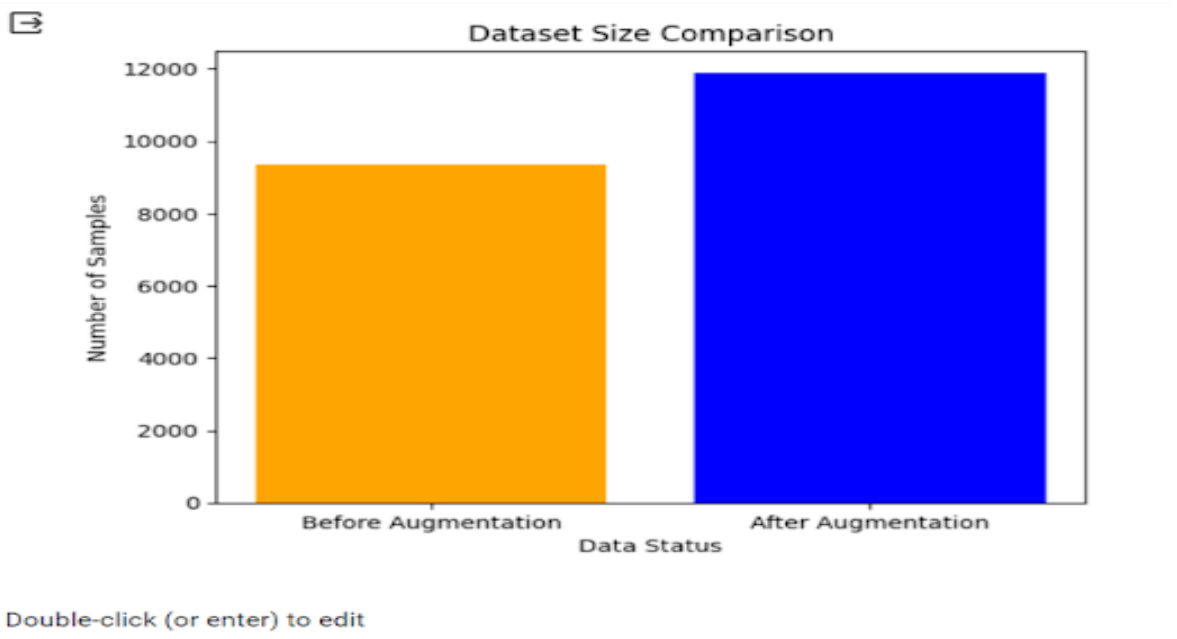
Comparison of Dataset Sizes - Before and After Augmentation

Lastly, compared the sizes of the dataset before and after augmentation. The process of augmentation significantly expanded the dataset, resulting in a larger and more diverse set of

samples. This expansion is essential for effectively training deep learning models. It is illustrated in Figure 61.

Figure 61

Before And After Augmentation



Modeling

4.1 Model Proposals

In the realm of Speech Emotion Recognition (SER), the fundamental framework relies on the application of data engineering techniques to extract crucial acoustic features, including Mel Frequency Cepstral Coefficients (MFCCs), Chromatograms, and Mel-scaled spectrograms. These techniques serve as the initial step in the conversion of raw audio data into meaningful representations that encapsulate the pertinent acoustic characteristics associated with emotional expression in speech.

After an initial phase of data preprocessing, the research project incorporated various deep learning models, selected based on insights gained from an extensive literature review. Firstly, 1D Convolutional Neural Networks (CNNs) were seamlessly integrated, drawing inspiration from the work of Yulan Li et al. (2019). This choice was rooted in their effectiveness in accurately capturing the intricate nuances and complexities of emotional expression within spoken language. Notably, 1D CNNs excel at discerning localized patterns within audio data, making them indispensable for identifying subtle acoustic cues that differentiate various emotions. Consequently, their inclusion significantly bolstered the accuracy and overall performance of the research project.

Shifting away from CNNs, the research was guided by the work of Aouani and Ayed (2020), which delved into the application of Autoencoders (AEs) for feature dimension reduction in speech emotion recognition. Given the inherently high-dimensional nature of audio data, particularly concerning features like Mel-Frequency Cepstral Coefficients (MFCCs), the integration of AEs played a pivotal role in simplifying and condensing this wealth of information. This dimensionality reduction not only streamlined subsequent classification

models but also harmonized seamlessly with the latest advancements in deep learning, ultimately resulting in an elevated level of accuracy across the research project (Rezapour Mashhadi & Osei-Bonsu, 2023).

Continuing the journey, the capabilities of Long Short-Term Memory (LSTM) networks were harnessed, as demonstrated by Kumbhar and Bhandari (2019). The rationale behind this choice was deeply rooted in the temporal and sequential characteristics inherently present in speech data, both of which are essential for capturing the dynamic nuances of emotional expression. Through the use of MFCCs as input features, LSTMs showcased their adeptness in learning from sequential data and retaining temporal information. This rendered them especially well-suited for the recognition of varying emotional states in speech, underscoring their suitability for intricate tasks like emotion recognition.

Finally, Deep Neural Networks (DNNs) found their place within the research project, inspired by the work of Md. Shah Fahad et al. (2021). DNNs were chosen for their effectiveness in modeling complex, nonlinear relationships within data, a trait that resonated well with the nuanced speech patterns associated with different emotions. Leveraging the IEMOCAP dataset, the study validated their significance by showcasing how DNNs substantially improved the performance of conventional Hidden Markov Models (HMMs) within the domain of emotion detection. This reaffirmed the pivotal role of DNNs in the research project, underlining their proficiency in enhancing the accuracy and robustness of emotion recognition models, thereby marking a significant stride forward in the field.

Four neural network models, namely autoencoders, CNN, DNN with multiple layers, and LSTM are explained below.

Convolutional Neural Networks

Literature Survey. Abbaschian et al. (2021) conducted a comprehensive review of Speech Emotion Recognition (SER) methods, focusing on both conventional and deep learning techniques and examining various emotional speech datasets. The study emphasizes the importance of precise and nearly real-time SER for improving human-computer interactions. Initially, the paper discusses the historical reliance on traditional machine learning methods like hidden Markov models, Gaussian mixture models, and support vector machines for SER, which required extensive preprocessing and feature engineering. However, the advent of deep learning, particularly neural networks, has significantly improved accuracy, surpassing 90% in controlled settings. Furthermore, the paper delves into the crucial role of training datasets in SER, categorizing them as natural, semi-natural, or simulated. Lastly, the survey highlights recent research that employs recurrent neural networks, long short-term memory networks, autoencoders, and generative adversarial models to address the complexities of speech emotion recognition.

In a specific focus on Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) in Speech Emotion Recognition (SER), Lim et al. (2016) explore recent advancements in deep learning approaches. Deep learning has become widely adopted in various research areas, including pattern recognition and signal processing, with CNNs excelling in computer vision tasks and RNNs being adept at handling sequential data. This paper aims to advance SER by proposing an algorithm that utilizes concatenated CNN and RNN architectures, eliminating the need for traditional hand-crafted features. This approach aligns with the trend of leveraging deep learning to improve classification accuracy. The proposed SER methodology is

evaluated on an emotional speech database and outperforms traditional classification methods, achieving superior performance without manual feature engineering.

Badshah et al. (2017) introduce an innovative Speech Emotion Recognition (SER) approach that employs spectrograms and a deep Convolutional Neural Network (CNN) architecture. Their deep CNN architecture uses spectrograms extracted from speech samples as input data. The model is designed to extract distinctive features from spectrogram images and predict seven different emotions. The study focuses on training this model using spectrograms from the Berlin emotional dataset and explores transfer learning by incorporating a pre-trained AlexNet architecture for emotion classification. Initial results indicate that the newly trained model outperforms the fine-tuned model in terms of accuracy and efficiency in emotion prediction. This pioneering spectrogram-based approach underscores the potential of deep learning techniques in SER, particularly when processing spectrogram representations within neural network models.

In their study titled "Multi-window Data Augmentation Approach for Speech Emotion Recognition," Padi et al. (2022) contribute to the field of SER by introducing a novel technique called MWASER (Multi-Window Augmentation Approach for Speech Emotion Recognition). Their main objectives include advancing SER capabilities through specialized speech augmentation and effective deep neural network design. They address the issue of model overfitting by generating additional data samples to improve emotion recognition accuracy. The paper emphasizes the importance of selecting optimal window sizes during feature extraction and advocates the use of Convolutional Neural Network (CNN) architectures for SER. Techniques such as dropout, regularization, batch normalization, and transfer learning are discussed to mitigate overfitting in the proposed CNN. A key innovation is MWASER's utilization of multiple

window sizes during audio feature extraction, enabling the analysis of both shorter and longer emotional expressions. Experiments conducted on datasets such as IEMOCAP, SAVEE, and RAVDESS demonstrate that MWASER significantly outperforms single-window baseline methods, effectively addressing core SER challenges and improving emotion recognition accuracy.

Ma et al. (2018) present an innovative deep neural network method that can directly handle variable-length speech sentences, leading to improved accuracy in emotion recognition from speech. By integrating Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), the model effectively extracts emotional cues from spectrograms without requiring sentence segmentation. This variable-length approach mitigates performance degradation associated with segmentation, especially enhancing the differentiation between neutral and non-neutral emotions. The study emphasizes the increasing use of deep learning techniques and raw, unprocessed data in speech processing for accurate emotion recognition. The suggested model combines RNNs to capture temporal structures between phrases and CNNs to extract emotional patterns from spectrograms.

Convolutional Neural Network

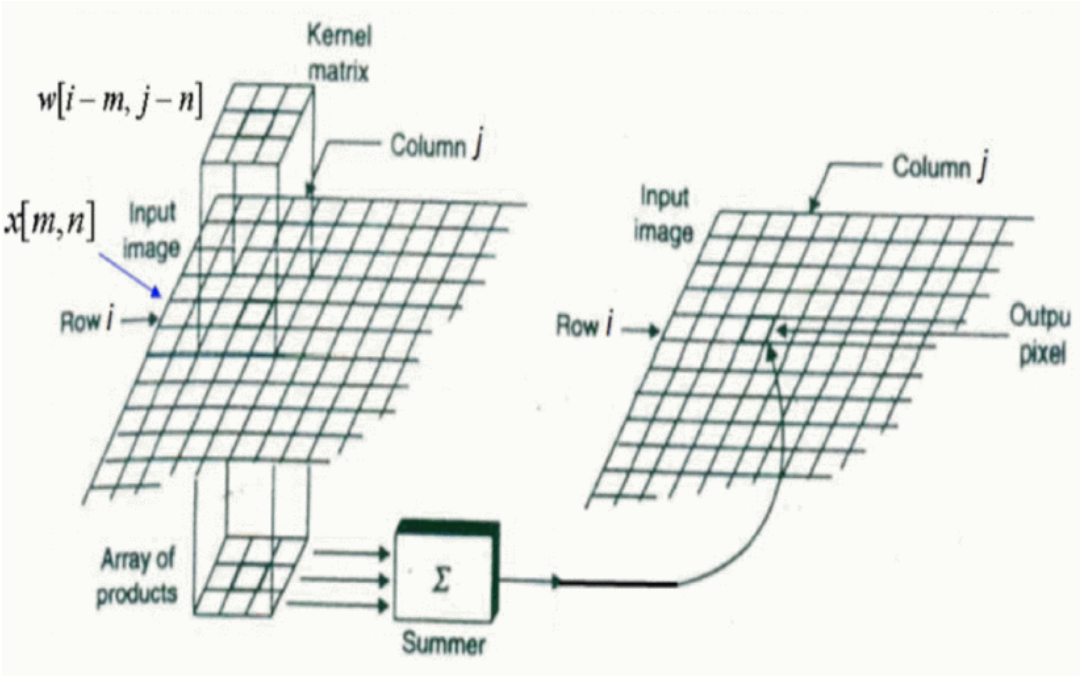
A Convolutional Neural Network (CNN) model has been created to employ deep learning to categorize emotions expressed in speech. This CNN structure incorporates several convolutional and pooling layers, designed to autonomously acquire intricate characteristics from audio spectrograms. The ultimate goal is to facilitate the recognition and sorting of diverse emotional states within speech recordings.

Input Layer. In CNNs, the initial layer typically corresponds to the unprocessed input data, typically in the form of an image. Images are typically displayed as a grid of pixel values, with each pixel potentially conveying information about color channel intensity.

Convolutional Layer. The core element of a CNN is the convolutional layer, which employs filters or kernels to perform convolution operations on the input image. These filters are small windows that scan across the input image in search of patterns such as textures, edges, or finer features (Kwon et al., 2020). You can refer to Figure 62 for a detailed illustration.

Figure 62

Convolutional Layer Details



Note: This Image is from a paper by (Albawi et al., 2017)

Activation Function. To introduce non-linearity into the model, the Rectified Linear Unit (ReLU) is applied element-wise after the convolution process, as depicted in equations 1 and 2.

$$\text{ReLU}(x) = \max(0,x) \quad (1)$$

$$\frac{d}{dx}(x) = \{1 \text{ if } x>0;\ 0 \text{ otherwise}\} \quad (2)$$

Pooling Layer. Pooling layers like MaxPooling or AveragePooling are employed to reduce the spatial dimensions of the input volume. Pooling, as discussed in (Albawi et al., 2017), helps in reducing computational demands, decreasing the number of parameters, and incorporating translation invariance. It retains the most important information while discarding less significant details.

Flattening Layer. The result obtained from the convolutional and pooling layers takes the form of a 3D tensor. Before being fed into the fully connected layers, this data must undergo flattening, transforming it into a 1D vector.

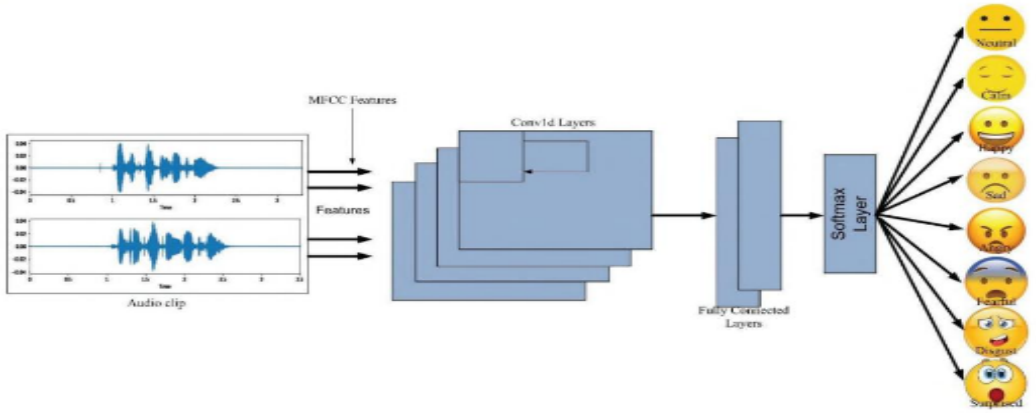
Fully Connected Layer. In a fully connected layer, the neurons are organized in a way that each node is directly linked to every other node in both the preceding and succeeding layers. Figure 2 illustrates this, where it's clear that each node in the last stage of the pooling layer is connected in a vector-like manner to the initial layer of the fully connected layer (Albawi et al., 2017).

Output Layer. The ultimate predictions are generated by the output layer, and the number of neurons in this layer is determined by the number of classes in the classification task.

In the context of the Convolutional Neural Network (CNN) utilized in this project, the architectural details are illustrated in Figure 63.

Figure 63

1D-CNN Architecture



Note: This Image is from paper by (Goel et al., 2020)

Additionally, Table 18 provides the pseudo-code outlining the CNN model's structure and operations, offering a comprehensive view of its implementation.

Table 18

Pseudocode of 1D-CNN

Layer Type	Description
Input Layer	Represents input data, often acoustic features like MFCCs extracted from audio.
Convolutional Layers	Conv1D layers apply convolutional operations for spatial feature extraction. ReLU activation introduces non-linearity. Batch Normalization stabilizes and accelerates training Multiple Conv1D layers may be used.

Layer Type	Description
MaxPooling Layers	Follow each Conv1D layer for spatial downsampling. Preserve important features and reduce computational load and apply MaxPooling Id with specified pool size and strides.
Flatten Layer	Flatten output from convolutional layers to a 1D vector.
Fully Connected Layers	Densely connected layers with ReLU activation. Capture high-level features from flattened representation. Batch Normalization for improved convergence.
Output Layer	Softmax activation converts scores to probability distributions. Produces final prediction for the emotion class.
Training	Setting the number of epochs and batch size. Dynamic learning rate adjustment and early stopping based on validation accuracy.
Evaluation	Assesses model performance on a test set. Reports test loss and accuracy metrics.

Model Optimization. The combination of autoencoders with 1D CNN in an ensemble model presents an innovative approach to speech emotion recognition, leveraging the synergies

between these two architectures. Tuning the CNN involves adjusting the learning rate of the optimizer for faster convergence, selecting epoch counts to strike a balance between underfitting and overfitting, and optimizing batch sizes for efficiency. Augmenting data diversifies the dataset, while fine-tuning dropout addresses overfitting issues. Varying kernel sizes and filter counts in convolutional layers captures diverse spatial patterns, and techniques like Batch Normalization enhance stability. Effective hyperparameter optimization, using the Hyperband tuning approach, has been crucial, improving classification accuracy from 80.2% to 83.6%. This systematic tuning of hyperparameters, facilitated by Hyperband, reinforces model robustness and adaptability while enhancing accuracy, underscoring the pivotal role of precise hyperparameter tuning in developing CNN architectures for speech emotion recognition.

Dense Neural Networks

Helen et al. (2021) investigated the application of Convolutional Neural Networks (CNNs) to predict diabetes, with a focus on the impact of hidden dense layers on classification performance. Their study aimed to enhance early diabetes detection through effective CNN predictive models, utilizing the Pima Indian Diabetes Database. They constructed three deep classification models, featuring 3, 4, and 6 hidden dense layers, and observed that increasing the number of layers improved classification performance. Notably, the 6-layer model achieved 99% accuracy, representing a 19% increase compared to the 3-layer model. The findings highlight the potential benefits of deeper models with additional dense layers, up to a certain saturation point.

In another study, Farnaz et al. (2021) compared the outcomes of Dense Neural Network (DNN) and Convolutional Neural Network (CNN) models for predicting a building's operational energy use based on its shape. Using a synthesized dataset generated from parametric building models and EnergyPlus simulations, they found that the DNN model outperformed the CNN

model in terms of prediction accuracy, simplicity, and computation time. The DNN model achieved a lower Mean Squared Error (MSE) value (0.008) compared to the CNN model (0.022) and demonstrated faster computation, making it more efficient for predicting building energy performance. However, the CNN model, despite its performance lag, offered the advantage of using architectural graphics as input, potentially enhancing design communication. This study underscores the suitability of the DNN model for precise mathematical predictions, while the CNN model provides visual prediction advantages.

In a separate investigation, Mohammed et al. (2018) addressed challenges in recognizing speech emotions using the MSP-Podcast corpus, a comprehensive emotional speech database. Their research explored various training techniques for Deep Neural Networks (DNNs), including data size, network depth, activation functions, batch normalization, and residual networks. They implemented the Exponential Linear Unit (ELU) activation function for the DNN model, highlighting its effectiveness over Rectified Linear Units. ELU is represented by Equation 3 in their study.

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha(exp(x) - 1) & \text{if } x < 0 \end{cases} \tag{3}$$

In another study, Zhen et al. (2019) present FDNet, a fully dense neural network designed for semantic segmentation, addressing challenges related to spatial information loss and training optimization. Unlike conventional encoder-decoder structures, FDNet utilizes an adaptive aggregation structure that incorporates feature maps from all previous blocks in the decoder module. This innovative approach enhances spatial boundary reconstruction and improves the

efficiency of gradient backpropagation. To tackle the labeling issue of "hard examples," the authors propose a boundary-aware loss function, represented as Equation 4 in their paper.

$$loss(L, L^{gt}) = -\frac{1}{N}\sum_{j=1}^{K}\sum_{I_i \in S_j}\sum_{c=1}^{C}\alpha_j L_{i,c}^{gt} w(L_{i,c}) log L_{i,c} \quad (4)$$

The traditional encoder-decoder structure for semantic segmentation often faces issues related to information loss during spatial reduction, resulting in coarse masks and optimization difficulties due to depth. FDNet addresses these limitations by efficiently integrating feature maps, yielding superior performance on standard datasets like PASCAL VOC 2012 and NYUDv2. It competes favorably with established models such as RefineNet and SDN, demonstrating notable memory efficiency. Experiments highlight FDNet's superiority, particularly in boundary localization, thanks to its boundary-aware loss function. In conclusion, FDNet's fully dense connected structure and boundary-aware approach offer improved spatial information reconstruction and efficient gradient propagation, making it a promising solution for semantic segmentation challenges (Zhen et al., 2019).

Pau et al. (2018) tackle the challenge of predicting 3D chromatin folding, crucial for gene regulation, based on the distribution of bound chromatin factors. They introduce a backward dense neural network (DNN) approach that models the relationship between chromatin structure and bound factor sequences. The DNN architecture incorporates convolutional filters to compress information from multiple factors into a single 1D sequence representation optimized for predicting chromatin structure. The study involves training networks in both directions, demonstrating the feasibility of this model for speech-emotion recognition. The results provide

insights into the physical mechanisms governing chromatin folding, highlighting the importance of chromatin contexts and neighborhoods in long-range contact regulation.

The figure for the pseudo code of the Dense Neural Network is provided below in Figure 64, offering a clear reference to the location of the code explanation.

Figure 64

Dense Neural Network Pseudo Code

Step	Description	Code

1	Initialize parameters (weights and biases) randomly.	`W1 = random_initialize(input_size, hidden_size)`
		`b1 = zeros(hidden_size)`
		`W2 = random_initialize(hidden_size, output_size)`
		`b2 = zeros(output_size)`

2	Define the activation function (e.g., sigmoid).	`def sigmoid(x):` `  return 1 / (1 + exp(-x))`

3	Forward pass through the network.	`def forward_pass(input_data):`
		`  z1 = input_data @ W1 + b1`
		`  a1 = sigmoid(z1)`
		`  z2 = a1 @ W2 + b2`
		`  a2 = sigmoid(z2)`
		`  return a2`

4	Training loop with gradient descent.	`for epoch in range(num_epochs):`
		`  for batch in iterate_over_batches(training_data, batch_size):`
		`  predictions = forward_pass(batch.inputs)`
		`  loss = compute_loss(predictions, batch.labels)`
		`  # Backward pass (compute gradients)`
		`  # Update weights and biases using gradient descent`

Model Optimization. Hyperparameter tuning played a pivotal role in optimizing dense neural network for speech emotion recognition, increasing categorical accuracy from 74.6% to 76.6%. Utilized the Hyperband method to fine-tune crucial hyperparameters like the number of hidden layers (1 to 3), units per layer (e.g., 64, 128, 256 neurons), activation functions (e.g., ReLU or Sigmoid), learning rates (e.g., 0.01, 0.001, 0.0001), and the number of training epochs. This systematic adjustment, guided by Hyperband, not only improved accuracy but also

reinforced the model's adaptability, underscoring the importance of precise hyperparameter tuning in advancing dense neural networks for speech emotion recognition.

LSTM

Literature Survey. The introduction of the Long Short-Term Memory (LSTM) network, a specialized variant of recurrent neural networks, by Hochreiter and Schmidhuber in 1997, represented a significant breakthrough. It addressed the notorious vanishing gradient problem that plagued traditional RNNs. This advancement empowered LSTMs to excel in capturing long-term dependencies within sequential data, a pivotal capability for tasks like speech and emotion recognition. (Hochreiter & Schmidhuber, 1997).

The distinctive architecture of LSTMs, encompassing input, output, and forget gates, received further enhancements from Gers et al. (2000), resulting in improvements in the model's learning and generalization capabilities. Additionally, Graves and Schmidhuber (2005) provided empirical evidence of LSTM's effectiveness in phoneme classification.

In subsequent research within the domain of machine translation, Cho et al. (2014) emphasized the proficiency of LSTMs in sequence-to-sequence learning. Furthermore, investigations by Greff et al. (2017) and Zaremba et al. (2014) delved into the architectural subtleties and regularization techniques associated with LSTMs. Additionally, Chung et al. (2014) conducted empirical assessments of gated RNN architectures, including LSTMs. Lipton et al. (2015) conducted a critical review of RNNs, including LSTMs, in the context of sequence learning.

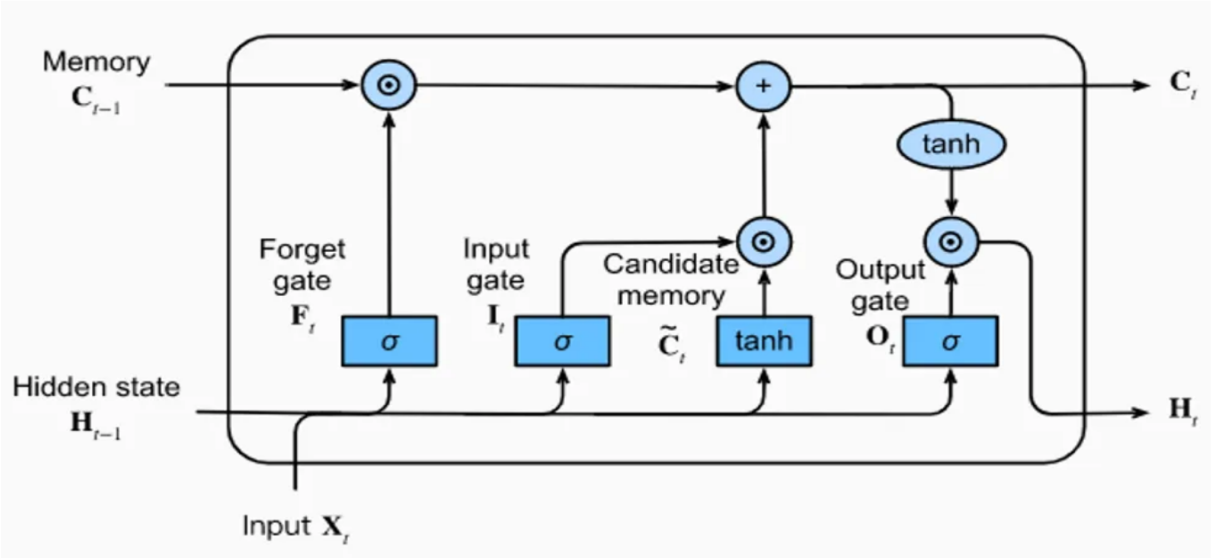
Long Short-Term Memory (LSTM) networks are renowned for their adaptability, robustness, and efficiency, firmly establishing them as a fundamental element in modern sequence learning, including applications like speech emotion recognition. LSTMs are tailored

for sequential data processing, excelling in capturing long-term dependencies due to their specialized architecture. Comprising units with three gates (input, forget, and output gates), LSTMs govern information flow, deciding what to retain or discard. The input gate updates data, the forget gate selects what to omit, and the output gate determines the output based on current input and unit memory. The cell state at the core acts as a conveyor belt for relevant information across the sequence, with each gate using the sigmoid activation function to control information flow (Hochreiter & Schmidhuber, 1997; Gers, Schmidhuber, & Cummins, 2000; Graves & Schmidhuber, 2005).

Figure 65 offers a detailed illustration of an LSTM cell, purposefully designed to process sequential data by selectively retaining or discarding information.

Figure 65

LSTM Model Architecture



Note. The structure has been adapted from the work by Hochreiter and Schmidhuber (1997)

At each time step, the LSTM cell receives the current input (x(t)) and the previous hidden state (h(t-1)). The forget gate (f(t)) decides what parts of the previous cell state (c(t-1)) to retain for the future, while the input gate determines which new information from the candidate memory ($\tilde{c}(t)$) should be added to the cell state. These decisions update the cell state (c(t)), preserving essential information throughout the sequence. Finally, the output gate (o(t)) combines the updated cell state and generates the new hidden state (h(t)), which is used in subsequent steps.

LSTMs address issues like the vanishing gradient problem common in traditional RNNs. During training, LSTMs adjust their weights through backpropagation, similar to other neural networks. However, their intricate gating mechanisms require precise parameter tuning (Cho et al., 2014). These gate interactions enable LSTMs to make nuanced decisions about what information to retain, making them effective for handling long-term dependencies, such as in speech emotion recognition.

Refer to Table 19 for the pseudocode illustrating how the LSTM layer functions.

Table 19

Formulae for Gates and Cell Updates in LSTM

Gate/Update	Equation
Forget Gate (f(t))	$f(t) = \sigma(W_f \cdot [h(t-1), x(t)] + b_f)$
Output Gate (o(t))	$o(t) = \sigma(W_o \cdot [h(t-1), x(t)] + b_o)$
Input Gate (i(t))	$i(t) = \sigma(W_i \cdot [h(t-1), x(t)] + b_i)$

Gate/Update	Equation
Candidate Memory Cell ($\tilde{c}(t)$)	$\tilde{c}(t) = \tanh(W_C \cdot [h(t-1), x(t)] + b_C)$
Cell State Update ($c(t)$)	$c(t) = f(t) * c(t-1) + i(t) * \tilde{c}(t)$
Hidden State Update ($h(t)$)	$h(t) = o(t) * \tanh(c(t))$
Hidden State Update ($h(t)$)	$h(t) = o(t) * \tanh(c(t))$

Note. The formulae has been adapted from the work by Hochreiter and Schmidhuber (1997)

Table 20 contains the pseudocode for implementing an LSTM network, detailing the initialization of weights, biases, and states.

Table 20

LSTM Model pseudocode

Step	Pseudocode
Initialization	Initialize weights: W_f, W_i, W_C, W_o Initialize biases: b_f, b_i, b_C, b_o Initialize h(0), c(0) to zero
Input Loop	for each time step t from 1 to T: Read input x(t)

Step	Pseudocode
Forget Gate Computation	$f(t) = \sigma(W_f \cdot [h(t-1), x(t)] + b_f)$
Input Gate Computation	$i(t) = \sigma(W_i \cdot [h(t-1), x(t)] + b_i)$ $\tilde{c}(t) = \tanh(W_C \cdot [h(t-1), x(t)] + b_C)$
Cell State Update	$(t) = f(t) * c(t-1) + i(t) * \tilde{c}(t)$
Output Gate Computation	$o(t) = \sigma(W_o \cdot [h(t-1), x(t)] + b_o)$
Hidden State Update	$h(t) = o(t) * \tanh(c(t))$
Output	Output: Hidden states h(1), h(2), ..., h(T)

Note. The pseudocode has been adapted from the work by Hochreiter and Schmidhuber (1997)

Model Optimization. Optimizing the LSTM model for speech emotion recognition, as demonstrated by Greff et al. (2017) and Zaremba et al. (2014), entails refining the network's architecture and employing regularization techniques. Critical hyperparameter adjustments, including layer count and learning rate, significantly impact model performance, especially when dealing with intricate emotional cues in speech. The careful selection of relevant features, following guidelines from McFee et al. (2015) and Virtanen et al. (2020), is pivotal for improving the model's efficiency. These optimizations ensure that the LSTM can effectively learn and generalize from the subtle emotional nuances within speech data, ultimately enhancing the accuracy of emotion recognition.

Auto Encoders

Literature Survey. Díaz and Fonnegra (2018) proposed a deep learning strategy for speech emotion recognition using paralinguistic features. They transformed these features into

higher-level representations using a deep convolutional stack auto-encoder network. Their research achieved an overall accuracy of 91.4% in testing and 61.1% in validation on the eNTERFACE'05 Database, demonstrating the effectiveness of combining acoustic features with deep-learning auto-encoders for emotion recognition.

Aouani et al. (2018) introduced a two-stage system for emotion recognition from speech, incorporating auto-encoders and a Deep Support Vector Machine (DSVM). They tested their system on the SAVEE audio database, highlighting the effectiveness of using MFCC features in conjunction with sophisticated classification methods like auto-encoders. Their study showed that the auto-encoder method and the DSVM method outperformed the standard SVM, achieving classification rates of 73.01% and 69.84%, respectively.

Deng et al. (2017) presented a semi-supervised autoencoder (SS-AE) model for speech emotion recognition, outperforming traditional supervised methods. Their research showed significant performance gains, particularly with smaller labeled datasets, emphasizing the effectiveness of semi-supervised approaches in situations with limited labeled data.

Ying et al. (2021) focused on enhancing speech-emotion recognition (SER) using unsupervised feature learning based on autoencoders. They implemented denoising and adversarial autoencoders for feature extraction and achieved classification accuracies of 78.67% (WA) and 76.89% (UA) on the IEMOCAP dataset, slightly outperforming state-of-the-art results with supervised learning methods, highlighting the potential of unsupervised learning in SER.

Fei et al. (2016) employed deep neural networks (DNNs) to address the challenge of emotion recognition in speech. Their research achieved a notable accuracy of 85% by focusing on high-level features such as Mel-Frequency Cepstral Coefficients (MFCCs) and prosodic features, surpassing traditional methods like Gaussian Mixture Models (GMM) and Support

Vector Machines (SVM). These findings underscored the superiority of DNNs in improving emotion recognition accuracy from speech.

The Pseudo-code for Autoencoders is illustrated in Figure 66 below.

Figure 66

Pseudocode of Autoencoder

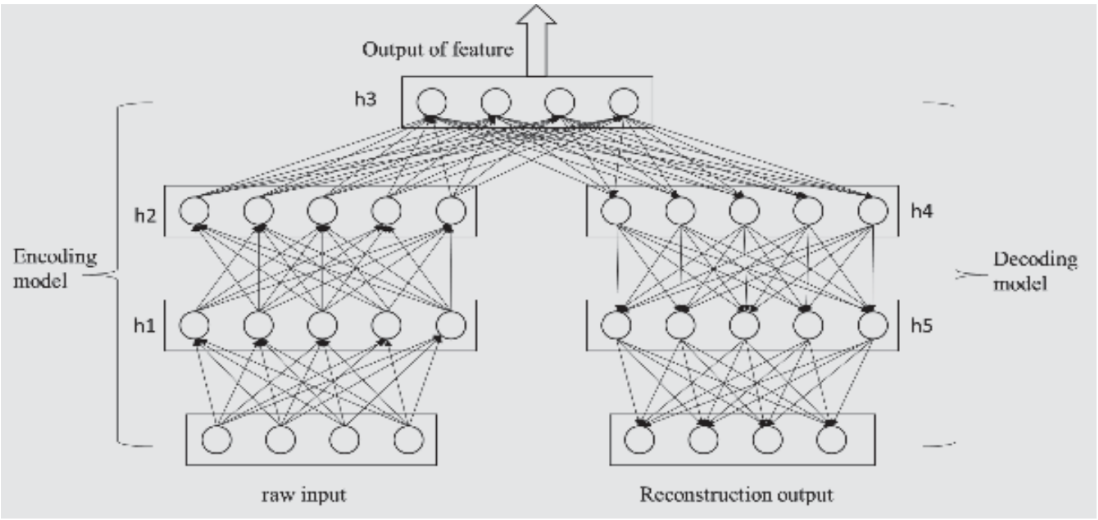
Step	Description
1: Initialize the Autoencoder	- Define an Encoder. - Define a Decoder.
2: Define the Encoder Process	For each layer l in the Encoder: - Input: x_l (input data to layer l). - Apply a linear transformation: $z_l = W_l \times x_l + b_l$. - Apply a non-linear activation function: $h_l = f(z_l)$. - Output: h_l (encoded data for layer l).
3: Define the Decoder Process	For each layer l' in the Decoder: - Input: $h_{l'}$ (encoded data to layer l'). - Apply a linear transformation: $z_{l'} = W_{l'} \times h_{l'} + b_{l'}$. - Apply a non-linear activation function: $x_{l'} = f'(z_{l'})$. - Output: $x_{l'}$ (reconstructed data for layer l').
4: Define the Loss Function	- Mean Squared Error (MSE): $L(x, x') = \|x - x'\|^2$. - Alternatively, Binary Cross-Entropy (for binary inputs): $L(x, x') = -\sum(x \cdot \log(x') + (1 - x) \cdot \log(1 - x'))$.
5: Training Process	While the training is not complete: - Pass input data x through the Encoder to get h. - Pass h through the Decoder to get x'. - Calculate the loss $L(x, x')$. - Update weights and biases to minimize the loss.
6: Feature Extraction	- Pass input data x through the Encoder to get h.
7: Data Reconstruction	- Pass h through the Decoder to get x'.

Autoencoders excels in various applications such as noise reduction, data augmentation, data compression, and feature extraction due to their proficiency in minimizing reconstruction loss. The method involves iterative adjustments of encoder and decoder parameters using backpropagation and gradient descent optimization, as depicted in Figure X. The primary objective is to closely align the reconstructed output with the original input. The training cycle initiates with forward propagation, encompassing data encoding and decoding, followed by loss evaluation and parameter refinement. Through these iterative stages, the autoencoder progressively enhances its accuracy in reproducing input data. Notably, there is no need for

Principle Component Analysis (PCA) integration within this framework, as autoencoders inherently possess feature extraction capabilities.

Figure 67

Transformation of raw input to reconstructed output in an autoencoder is shown.



Note: The image has been taken from the paper by Fei et al. (2016)

The autoencoder is utilized by the system in this advanced speech emotion recognition framework due to its superiority in unsupervised feature extraction and dimensionality reduction . Using a sequence of ReLU (Rectified Linear Unit) activation functions, the autoencoder effectively compresses speech input into a lower-dimensional space, facilitating non-linear learning of data representations. Essential elements that capture the speech's subtle emotional undertones are extracted during this encoding process. These prominent, emotionally charged features are then fed into a custom-tuned 1D Convolutional Neural Network (1D CNN) for time-series data analysis. With its specialized layers engaged by ReLU functions, the 1D CNN goes deeper to unearth intricate, higher-level characteristics that are essential for accurately classifying speech emotions. This model is trained using a loss function—mean squared error for

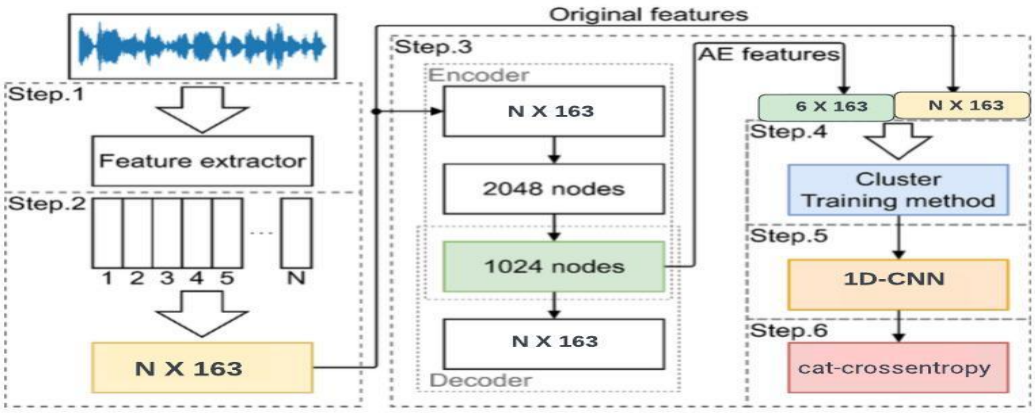
the autoencoder to minimize reconstruction discrepancy, and categorical cross entropy for the 1D CNN to handle the multi-class classification—thereby ensuring high fidelity in emotion prediction. This synergistic workflow, beginning with the autoencoder's reduction and abstraction of data and culminating with the 1D CNN's nuanced feature refinement and classification, establishes the core methodology of this sophisticated emotion recognition system.

A comprehensive explanation of the architecture and pseudocode of CNN can be found in the figure and Table 18, which were previously detailed in the preceding section on CNN.

Figure 69 illustrates an implementation that successfully harnesses the complementary strengths of autoencoders and 1D Convolutional Neural Networks (1D CNNs). This integrated approach capitalizes on autoencoders' proficiency in compressing and encoding input data, seamlessly followed by the utilization of 1D CNNs for advanced feature extraction and classification. This synergy elevates the model's overall capacity, enabling it to efficiently process and reduce data dimensionality while also extracting vital features and performing precise classification tasks, as delineated in the sequential steps depicted in the figure.

Figure 69

Autoencoder-1D CNN method architecture diagram



Note. The image has been referred from the paper by (Chen et al., 2023, p. 10)

Model optimization. In optimizing a speech emotion recognition model, the significant impact of hyperparameter tuning on performance improvement was evident, leading to a rise in categorical accuracy from 83.6% to 85.6%. Employing the Hyperband method, the focus was directed towards critical hyperparameters, including the number of convolutional layers, filter sizes, kernel sizes, and learning rates. Variations in the number of convolutional layers (ranging from 1 to 3) influenced the depth of feature extraction, while diverse filter sizes (32, 64, 128) enabled a more intricate capture of emotional subtleties in speech. The selection of kernel sizes (4 and 5) played a pivotal role in processing temporal patterns, vital for deciphering emotions in speech sequences. Additionally, the adjustment of learning rates (0.01, 0.001, 0.0001) struck a balance between optimization speed and stability, a crucial factor for ensuring the model converges to optimal weights. This systematic hyperparameter tuning, orchestrated through the Hyperband tuner, not only enhanced accuracy but also bolstered the model's resilience and adaptability, underscoring the paramount importance of precise hyperparameter optimization in advanced deep-learning models.

4.2 Model Supports

Environment, Platform, and Tools

All models for speech emotion recognition were developed using a system with an Intel Core i7 CPU, 16GB RAM, an NVIDIA GeForce RTX 3080 GPU, running Windows 11, and utilizing Google Colab Pro's Tesla V100 GPU for computational tasks. Data storage and management were handled through the Google Cloud Platform.

For every model, extensive feature extraction was carried out using Librosa, a Python library designed for audio analysis. Librosa functions like librosa.feature.mfcc() and librosa.feature.chroma_stft() were pivotal in extracting Mel-frequency cepstral coefficients

(MFCCs) and chroma features from the audio data, providing valuable insights into the emotional content within the speech. Additionally, PyRubberband was employed for audio data augmentation, leveraging functions like pyrb.time_stretch() and pyrb.pitch_shift() to introduce variability into the training data, thus enhancing the model's robustness.

For data splitting and model evaluation, Scikit-learn, a fundamental Python library for deep learning, was utilized. Functions such as train_test_split() and StratifiedKFold() were instrumental in partitioning the dataset into training, validation, and testing sets, establishing a robust evaluation framework. Scikit-learn also offered a diverse range of metrics, including confusion_matrix, accuracy_score, precision_score, recall_score, and f1_score, facilitating a comprehensive assessment of the model's performance in accurately recognizing emotions in speech.

Pandas and NumPy played crucial roles in data manipulation within the LSTM. Pandas, with functions like DataFrame, read_csv, and data manipulation methods (e.g., merge, drop, concat), provided an efficient means of handling and preprocessing datasets. NumPy supported essential numerical operations for feature engineering and data processing. Furthermore, Matplotlib and Seaborn were employed for data and result visualization, utilizing functions like Matplotlib.pyplot.plot and seaborn. heatmap to depict metrics such as confusion matrices and ROC curves, offering intuitive insights into the model's performance.

Table 21 below provides a summary of the essential libraries, methods, and their respective usages in the project.

Table 21

Required Libraries, Method/Functions in Libraries, and Usage of these Libraries

Library	Method	Usage
Scikit-learn	train_test_split(), StratifiedKFold(),sklearn.model_selection.GridSearchCV(),confusion_matrix(), accuracy_score(), precision_score(), recall_score(), f1_score()	Data splitting into training, validation, and testing sets; Cross-validation,Hyperparameter tuning using GridSearchCV,Model evaluation metrics for assessing performance
Pandas	DataFrame(), read_csv(), merge(), drop(), concat()	Efficient data handling, preprocessing, and manipulation
NumPy	Numerical operations	Support for numerical operations in feature engineering and data processing
Matplotlib	matplotlib.pyplot.plot()	Data visualization and plotting of metrics
Seaborn	seaborn.heatmap()	Data visualization, particularly for plotting confusion matrices and ROC curves
tqdm	tqdm()	Real-time progress visualization during lengthy processes
Keras with TensorFlow	Model(), Sequential(), LSTM(), Dense(), Dropout(), Activation(), compile(), fit()	Model construction, training, and evaluation in the LSTM project
noisereduce	noisereduce.reduce_noise()	Reducing noise based on audio signal
timeit	timeit.timeit()	Measuring the execution time of code

Library	Method	Usage
Librosa	librosa.feature.mfcc(), librosa.feature.chroma_stft()	Feature extraction for MFCCs and chroma features from audio data
PyRubberband	pyrb.time_stretch(), pyrb.pitch_shift()	Audio data augmentation for time-stretching and pitch-shifting

Data Workflow and Architecture

The initial preprocessing and data augmentation phase play a crucial role in creating a high-quality and diverse dataset, forming the foundation for robust emotion recognition capabilities. Various datasets, including RAVDESS (Livingstone & Russo, 2018), CREMA-D (Cao et al., 2014), and SAVEE (Haq & Jackson, 2010), are used, encompassing a range of emotional expressions performed by actors. These datasets undergo rigorous preprocessing to maintain audio fidelity, ensuring compliance with standardized WAV format and a sampling rate of approximately 44KHz. Techniques like data smoothing are applied to reduce background noise and enhance recording clarity. Augmentation methods, such as introducing white noise, pitch shifting, and time stretching, are employed to simulate diverse acoustic conditions and speaker variations, enriching the dataset and addressing class imbalances, especially for less represented emotions like fear and surprise.

Following preprocessing, the data is divided into training, testing, and validation sets in a 60-20-20 percent ratio. This partitioning supports a comprehensive model evaluation process, covering training, hyperparameter optimization during validation, and performance assessment on unseen data. Additionally, the training data undergoes 5-fold cross-validation within the autoencoder and 1D CNN framework to improve model generalization and prevent overfitting.

The autoencoder plays a crucial role in condensing high-dimensional data into a 1024-dimensional space (Schuller et al., 2013). This encoding is then decoded to restore the data to its original size using a 'sigmoid' function in the final convolutional layer reconstruction. The encoded representation feeds into a 1D CNN classifier, featuring multiple convolutional layers with 512, 256, and 128 filters, along with batch normalization and max pooling (Livingstone & Russo, 2018). The model is optimized using a 'categorical_crossentropy' loss function and concludes with a 'softmax' activated layer for emotion classification (Cao et al., 2014). For a detailed insight into the data flow and architecture.

In the LSTM architecture, two LSTM layers with 'relu' activation functions capture sequential patterns and temporal dependencies, utilizing 'mean_squared_error' loss for regression tasks like speech emotion recognition. In contrast, the Dense Neural Network (DNN) is tailored for tasks without sequential dependencies. It incorporates two dense layers with 'relu' activation functions to capture complex patterns and includes dropout layers for regularization. The DNN is optimized with 'categorical_crossentropy' loss, making it effective for classification tasks.

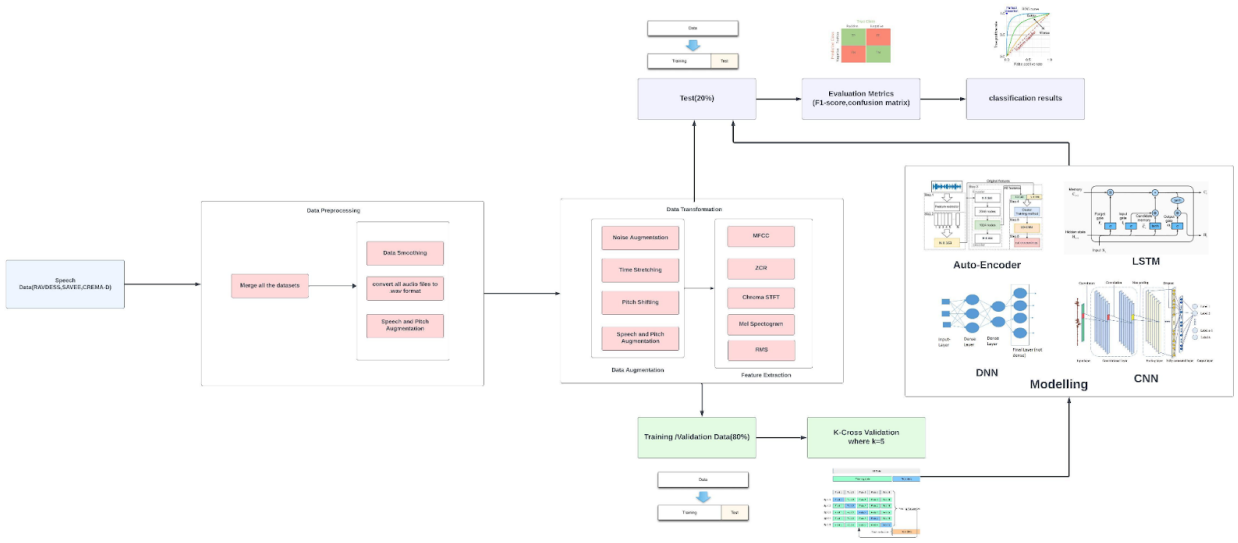
For model evaluation, a suite of metrics including precision, recall, and the F1-score are employed to provide a nuanced view of the model's classification prowess and its handling of class imbalances (Schuller et al., 2013). The confusion matrix elucidates the model's classification accuracy across emotions, while ROC curves—generated with and without the implementation of PCA and data augmentation—visually represent the model's diagnostic efficacy (Cao et al., 2014; Livingstone & Russo, 2018).

These ROC curves are essential for assessing the model's discriminative performance across various feature sets and preprocessing methodologies, ensuring a thorough evaluation of

its emotion recognition capabilities (Jackson & Vincent, 2003). For a detailed view of their architecture, please refer to Figure 70.

Figure 70

Data Flow and the Model Architecture of the Research Project



4.3 Model Comparison and Justification

In the realm of Speech Emotion Recognition (SER), four advanced deep learning algorithms have been employed to enhance the accuracy and efficiency of emotion detection from speech data. These algorithms include 1D Convolutional Neural Networks (1D CNN), Dense Neural Networks, Long Short-Term Memory Networks (LSTM), and a hybrid approach that combines Autoencoders with 1D CNN. To ensure data integrity and prevent leakage, row-wise normalization has been applied consistently across datasets. Additionally, data smoothing and augmentation techniques were utilized to enrich the dataset and address class imbalances, thereby establishing a robust foundation for the deep learning models to effectively extract and interpret emotional cues from speech.

The implementation of 1D CNN is crucial for extracting temporal features from speech data. Given its proficiency in processing time-series data, this architecture is well-suited for analyzing audio signals with evolving patterns over time. The 1D CNN efficiently captures relevant features from raw audio by applying convolutional operations along the time axis, a pivotal step in recognizing a wide range of emotions in speech (Hershey et al., 2017). Due to its adaptability in handling varying input lengths and its effectiveness in learning from spectrograms, the 1D CNN stands as the optimal choice for SER applications.

The second model employs a distinct approach in emotion identification, utilizing Dense Neural Networks. These fully connected networks excel in recognizing intricate patterns by establishing connections between each neuron in one layer and every other neuron in subsequent layers. This architectural design facilitates the integration of data across the entire audio spectrum, leading to a comprehensive analysis of the emotional content present in speech. Dense Neural Networks' capacity to decode non-linear relationships further enables the detection of subtle details within emotional expressions (Hinton et al., 2012).

Additionally, the study incorporates a model based on Long Short-Term Memory (LSTM) networks. LSTMs, categorized as recurrent neural networks, demonstrate remarkable proficiency in processing sequential data. In the context of Speech Emotion Recognition (SER), where comprehending the evolution and context of emotional cues over time is paramount, LSTMs' ability to identify long-term dependencies in time-series data proves to be crucial. These LSTM networks retain a memory of previous data inputs to contextualize current data, providing a holistic understanding of the emotional trajectory within speech patterns. LSTMs are exceptionally well-suited for SER tasks, given their exceptional memory and adeptness at

leveraging past information to capture the temporal dynamics essential for precise emotion detection (Hochreiter & Schmidhuber, 1997).

The selection of these models took into account several factors, including the characteristics of the data, the risk of overfitting or underfitting, preprocessing requirements, and computational demands. LSTMs excel in processing sequential data and retaining long-term memory but come with significant computational resource requirements. 1D CNNs are efficient at feature extraction but may be susceptible to overfitting. DNNs exhibit adaptability to diverse data types but necessitate extensive preprocessing and are prone to overfitting. The hybrid model combining Autoencoders and 1D CNNs provides comprehensive data analysis capabilities but demands substantial computational resources which is shown in table 22.

Table 22

The following table presents a comparison of the proposed models about specific aspects

Characteristic	**1D-CNN**	**Dense Neural Network**	**LSTM**	**Autoencoders 1D-CNN**
Architecture	Parallel	Fully Connected	Sequential	Hybrid(parallel+ sequential)
Data Type	Audio, Time-Series	General, Multi-Domain	Audio, Time-Series	Audio, Time-Series
Data Size Suitability	Moderate-Large datasets	Large datasets	Small-Medium datasets	Moderate-Large datasets
Overfitting Risk	Moderate	High	Moderate	Low
Preprocessing	High	Moderate	High	High

Characteristic	1D-CNN	Dense Neural Network	LSTM	Autoencoders 1D-CNN
Space Complexity	Moderate	High	High	High
Strengths	temporal feature extraction	Complex pattern recognition	Long-term dependency	Feature extraction + Temporal pattern
Computational Complexity	Moderate (CPU/GPU)	High (GPU beneficial)	High (GPU beneficial)	Moderate-High (GPU beneficial)
Strengths	Efficient temporal feature extraction	Complex pattern recognition	Long-term dependency capture	Feature extraction + Temporal pattern
Limitations	Temporal feature granularity	Overfitting, Computational demand	Sequential processing complexity	Complexity in architecture and training

Note. This table is a continuation for Table 22.

4.4 Model Evaluation Methods

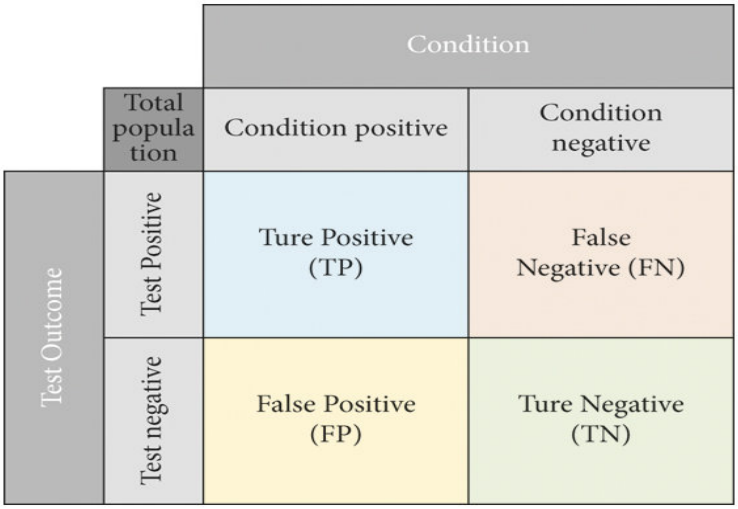
An array of metrics plays a vital role in assessing the performance of Deep Learning models, encompassing Autoencoders, 1D-CNN, Dense Neural Networks, and LSTM. These metrics are essential for evaluating the effectiveness of each model and for identifying issues such as underfitting or overfitting. Key metrics in this evaluation process comprise the confusion matrix, accuracy, recall, precision, and F1-score. Building upon foundational research (Akinpelu & Viriri, 2023), the objective in this domain is to elucidate the capabilities and challenges inherent in these models. A comprehensive understanding of the functioning of these metrics and their impact on SER model evaluation is critical for advancing the field and enhancing the precision and reliability of emotion recognition technologies.

Confusion matrix

Within the realm of Speech Emotion Recognition (SER), the Confusion Matrix emerges as a vital visualization tool for evaluating model performance. It categorizes predictions into four distinct groups: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). In SER, where the ability to differentiate between various emotional states in speech holds paramount importance, this matrix assumes a central role. Its significance extends beyond assessing overall accuracy; it precisely identifies areas where the model might misclassify one emotional state as another (Karray & Silva, 2020). The structure of the confusion matrix is depicted in Figure 71 below.

Figure 71

Structure of the confusion matrix



Note: The image has been taken from the paper by Niu & Yang (2021).

Precision

As highlighted by Salmon et al. (2016), precision stands as a pivotal metric for evaluating the precision of positive predictions made by Speech Emotion Recognition (SER) models. It signifies the proportion of accurately classified positive instances, such as specific emotions,

among all the instances predicted as positive. In contexts where minimizing false positives in emotion detection holds significant significance, this metric takes on added importance. Equation 5 provides the formula for calculating precision.

$$Precision = True\ Positives\ (TP)/(True\ Positives(TP) + False\ Positives(FP)) \quad (5)$$

Precision plays a vital role in guaranteeing the correctness of identified emotional states while reducing the occurrence of false positives, where non-emotional or different emotions are mistakenly classified. This is especially critical in applications where misidentifying emotions can have significant consequences.

Recall (Sensitivity)

Salmon et al. (2016) emphasize that precision is a crucial metric for assessing the accuracy of positive predictions generated by Speech Emotion Recognition (SER) models. It represents the percentage of correctly classified positive examples—such as specific emotional states—among all positive instances. In situations where reducing false positives in emotion detection is critical, this measure becomes especially important. Equation 6 contains the formula to compute precision.

$$\text{Recall} = \frac{\text{True Positives (TP)}}{\text{True Positives (TP)} + \text{False Negatives (FN)}} \quad (6)$$

Achieving high recall is paramount as it ensures the comprehensive identification of all pertinent instances of specific emotions, leaving no room for critical emotional cues to be overlooked. This emphasis on high recall is particularly vital in contexts where failing to detect

an emotional expression could lead to significant consequences, as seen in applications like mental health monitoring or safety systems.

Accuracy

Accuracy serves as a crucial performance metric in Speech Emotion Recognition (SER), representing the proportion of correct predictions, encompassing both true positives and true negatives, among the total instances. However, it warrants cautious interpretation, especially in scenarios with imbalanced data classes. A pertinent illustration of the significance and applicability of accuracy in SER research is evident in a study focusing on speech emotion classification. This research, employing an attention-based network coupled with regularized feature selection, achieved an impressive classification accuracy of 97.8% (Akinpelu & Viriri, 2023).Equation 7 contains the formula to compute Accuracy.

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{7}$$

F1-score

F1-Score assumes significance as a metric that harmonizes the trade-off between recall and precision. Its relevance is particularly pronounced in scenarios where achieving a balance between accurately detecting emotional states and capturing all relevant emotional instances is essential.

The F1-Score played a pivotal role in evaluating the performance of a model in a research study focused on speech emotion classification employing attention-based networks and regularized feature selection, underscoring the model's importance in SER applications (Akinpelu & Viriri, 2023).Equation 8 contains the formula to compute F1 score.

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{8}$$

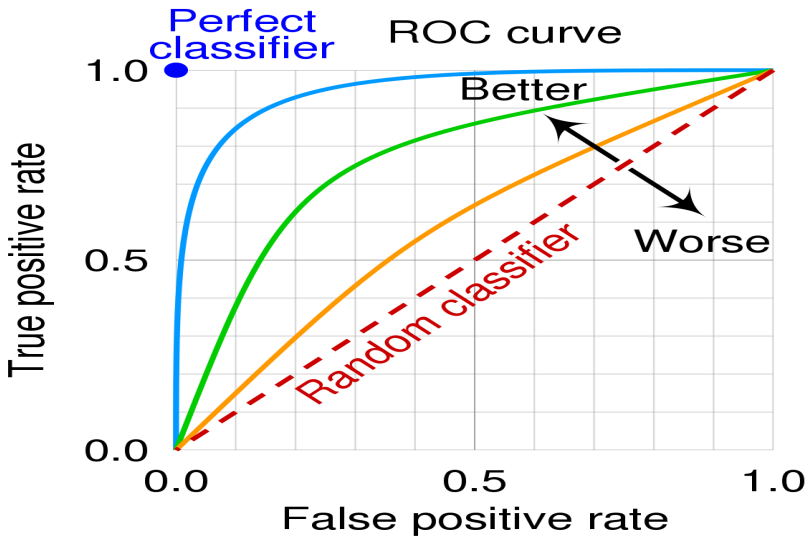
In Speech Emotion Recognition (SER), the F1-Score assumes a crucial role by striking a harmonious equilibrium between precision and recall. Its significance is notably pronounced in situations where achieving a delicate balance between precisely identifying emotional states and comprehensively capturing all pertinent emotional instances is of paramount importance (Akinpelu & Viriri, 2023).

ROC Curve and AUC

Crucial metrics for evaluating classification models include the Receiver Operating Characteristic (ROC) curve and the Area Under the Curve (AUC). Figure 72 visually presents the ROC curve, a graphical representation that portrays the model's diagnostic capabilities by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) across various threshold settings.

Figure 72

ROC graph



Equation 9 offers a means to compute the AUC, which quantifies the integral of the ROC curve, condensing the model's overall performance into a single value. AUC values range from 0.5, signifying performance equivalent to random chance, to 1, indicative of perfect classification. These metrics were effectively utilized in a study employing regularized feature selection and attention-based networks for speech emotion classification, underscoring their efficacy in assessing model performance within SER (Akinpelu & Viriri, 2023).

$$\text{AUC} = \sum_{i=1}^{n} \frac{(FPR_i - FPR_{i-1}) \times (TPR_i + TPR_{i-1})}{2} \tag{9}$$

4.5 Model Validation and Evaluation

In the initial project phase, data augmentation techniques were applied to enhance the Speech Emotion Recognition (SER) dataset, which originally consisted of 9,362 audio files collected from RAVDESS (Livingstone & Russo, 2018), CREMA-D (Cao et al., 2014), and SAVEE (Haq & Jackson, 2010). To address the issue of emotional expression imbalance, especially for underrepresented emotions like fear, surprise, and calm, various augmentation methods were employed. These techniques encompassed the addition of white noise, pitch shifting, time stretching, as well as speed and pitch augmentation.

As a result, the dataset was substantially enriched, expanding to a total of 11,890 files after augmentation, as illustrated in Table 23 .

This strategic augmentation played a pivotal role in achieving a more equitable representation of the eight core emotions within the dataset, thereby enhancing the model's capacity to accurately detect and classify a diverse array of emotional expressions.

Table 23

Augmented File Distribution for Datasets

Dataset Name	No. of Original Audio Samples	No. of Augmented Audio Samples	Total Dataset Size Post-Augmentation
RAVDESS+SAVEE+CREMA-D	9362	2538	11890

The below Table 24 shows the PCA Components, Reduced PCA and Variance Captured

Table 24

PCA Distribution of the Post Augmentation Data

Dataset Name	PCA components	Reduced PCA Components	% Variance Captured
RAVDESS+SAVEE+CREMA-D	163	30	95%

The impact of various augmentation techniques, Principal Component Analysis (PCA), and L2 regularization on different Speech Emotion Recognition (SER) models was assessed in terms of accuracy. Initially, before any augmentation, PCA, or regularization, the hybrid model combining autoencoders with a 1D Convolutional Neural Network (CNN) achieved an accuracy of 78%. After applying data augmentation, the accuracy increased to 82.4%. Subsequent hyperparameter tuning further improved the accuracy, reaching a peak of 86.5%.

Notably, other models like CNN, LSTM, and DNN demonstrated varied responses to PCA. Before PCA, the CNN had an accuracy of 74%, which increased to 78% after PCA. The LSTM had an accuracy of 76% before PCA, which decreased to 72% after PCA. The DNN had

an accuracy of 70% before PCA, which increased to 73% after PCA. These accuracy fluctuations illustrated the unique impact of PCA and regularization on distinct SER models.

However, the integration of PCA and L2 regularization with the autoencoder and 1D CNN architecture resulted in a decline in accuracy, potentially falling below the initial 78%. This accuracy reduction might be attributed to PCA's tendency to exclude crucial information necessary for discerning nuanced emotional states in speech due to its dimensionality reduction process. Additionally, the introduction of L2 regularization, while typically effective in preventing overfitting by penalizing model complexity, could have contributed to this accuracy decrease. The simultaneous application of PCA and L2 regularization may have overly constrained the model, impeding its ability to capture the intricate emotional nuances present in speech data.The comparison of percentages before augmentation and after augmentation with PCA is presented in Table 25. Table 25 showcases the distribution of the post-augmentation data, emphasizing how data augmentation techniques enriched the dataset. Additionally, Table 25 displays the accuracies before and after the augmentation of data with PCA, highlighting the impact of these techniques on model performance.

Table 25

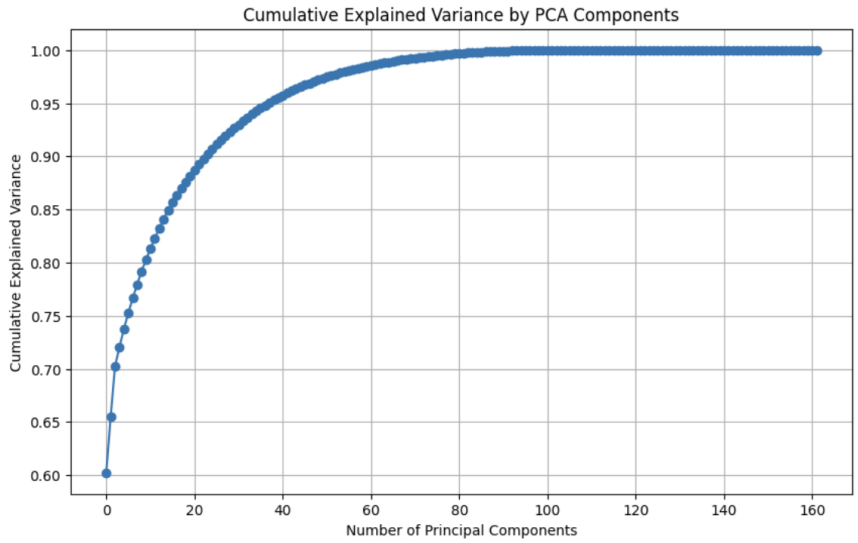
The model comparison of percentages before augmentation and after augmentation with PCA

Model Name	Accuracy before augmentation	Accuracy After Augmentation with PCA
DNN	0.70	0.73
1D CNN	0.74	0.78
LSTM	0.72	0.75
Autoencoders + 1D CNN	0.78	0.83

Figure 73 illustrates the scree plot for determining the number of principal components retained in the PCA analysis

Figure 73

Scree plot for determining the number of principal components



The above visualization provides a clear picture of the trade-off between the number of components and the amount of information (variance) retained from the original dataset. For instance, if the curve flattens out quickly, it indicates that most of the variance can be captured by a relatively small number of components. This would imply that PCA has effectively reduced the dimensionality of the data while retaining most of its information.

For a robust evaluation, a 5-fold cross-validation was employed, ensuring the model's generalizability and stability across different subsets of the data. The data was split into three parts: training, validation, and testing, with a distribution of 60% for training, 20% for validation, and 20% for testing. This approach aims to provide a comprehensive assessment of the model's performance and its ability to generalize to unseen data. The variation in performance across these different subsets is illustrated in Table 26.

Table 26

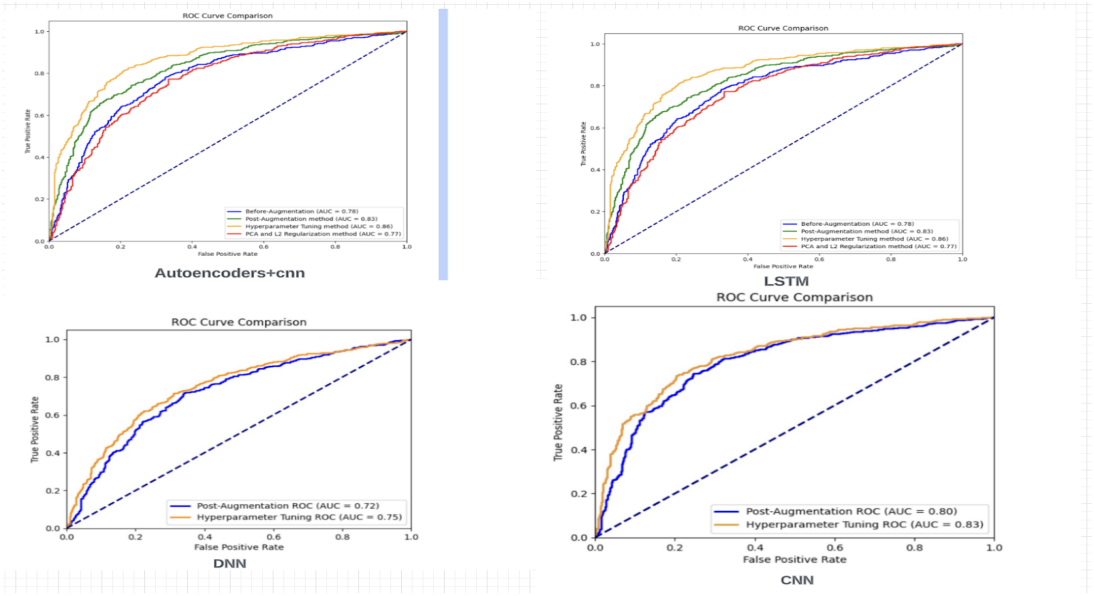
Comparison of percentages after 5 k-cross validation

Model Name	After K-fold Validation
DNN	72.9
1D CNN	78.3
LSTM	75.8
Autoencoders + 1D CNN	82.6

This methodological approach underpins the comparative analysis presented in Figure X, which illustrates the SER model's performance in terms of the Receiver Operating Characteristic (ROC) curve before and after the application of PCA, data augmentation, and L2 regularization.

Figure 74

Roc curve comparison



To enhance the understanding of the Figure 74, it's important to note that these visuals not only illustrate the integration of autoencoders with a 1D Convolutional Neural Network (CNN) and the comparative analysis of performance metrics between full-component and reduced-component models post-dimensionality reduction (such as through Principal Component Analysis), but they also delve into the realm of emotional recognition. These figures extend their analysis to include classification reports for CNN, Deep Neural Networks (DNN), and Long Short-Term Memory (LSTM) networks, emphasizing their efficacy in recognizing a spectrum of human emotions. This spectrum encompasses emotions like sadness, anger, disgust, neutrality, and surprise. The detailed representation in these figures offers a comprehensive view of how these advanced neural network architectures perform in the nuanced task of emotion recognition, highlighting their capabilities in various scenarios.

Figure 75

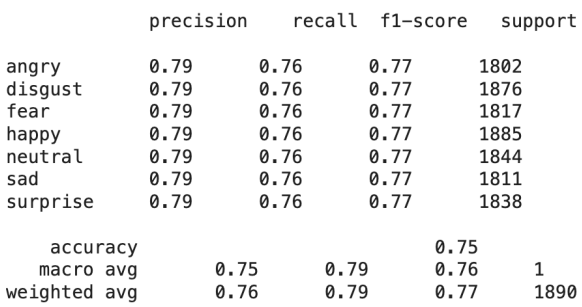
Classification Report for Augmented Data with PCA for Auto-encoders

	precision	recall	f1-score	support
angry	0.87	0.86	0.86	2124
disgust	0.86	0.87	0.86	2167
fear	0.86	0.87	0.86	2169
happy	0.86	0.86	0.86	2188
neutral	0.87	0.87	0.87	2144
sad	0.87	0.86	0.87	2190
surprise	0.87	0.87	0.87	2137
accuracy			0.86	
macro avg	0.86	0.87	0.86	1
weighted avg	0.87	0.86	0.87	1

The classification metrics show the model's strong performance with an overall accuracy of 86% and consistent precision, recall, and F1-scores around 0.87 for each emotion class.

Figure 76

Classification Report for Augmented Data with PCA for Dense Neural Networks

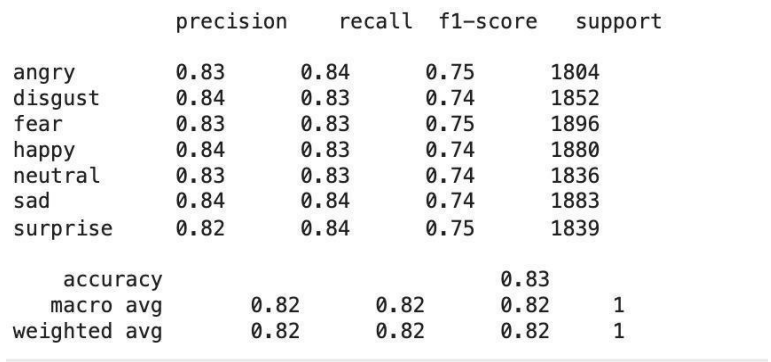


	precision	recall	f1-score	support
angry	0.79	0.76	0.77	1802
disgust	0.79	0.76	0.77	1876
fear	0.79	0.76	0.77	1817
happy	0.79	0.76	0.77	1885
neutral	0.79	0.76	0.77	1844
sad	0.79	0.76	0.77	1811
surprise	0.79	0.76	0.77	1838
accuracy			0.75	
macro avg	0.75	0.79	0.76	1
weighted avg	0.76	0.79	0.77	1890

The above figure 76 showcases the precision to be 74% which means that 74% of the predicted emotions were correct, and a recall of 74% indicates that 74% of the actual emotions were correctly identified. The F1-score, which combines precision and recall, is also at 74%, indicating a balanced performance in recognizing emotions.

Figure 77

Classification Report for Augmented Data with PCA for CNN



	precision	recall	f1-score	support
angry	0.83	0.84	0.75	1804
disgust	0.84	0.83	0.74	1852
fear	0.83	0.83	0.75	1896
happy	0.84	0.83	0.74	1880
neutral	0.83	0.83	0.74	1836
sad	0.84	0.84	0.74	1883
surprise	0.82	0.84	0.75	1839
accuracy			0.83	
macro avg	0.82	0.82	0.82	1
weighted avg	0.82	0.82	0.82	1

The above figure 77 showcases At 83% accuracy, the model demonstrates good overall performance, with precision indicating the proportion of correctly predicted positive instances among all positive predictions.

Figure 78

Classification Report for Augmented Data with PCA for LSTM

	precision	recall	f1-score	support
angry	0.76	0.75	0.77	1833
disgust	0.75	0.76	0.75	1883
fear	0.77	0.76	0.76	1855
happy	0.75	0.76	0.77	1860
neutral	0.75	0.76	0.77	1898
sad	0.77	0.77	0.76	1896
surprise	0.77	0.75	0.75	1867
accuracy			0.76	
macro avg	0.76	0.77	0.76	1
weighted avg	0.77	0.76	0.76	1

The above Figure 78, At 76%, precision indicates that the model correctly predicted the majority of emotions, while recall shows it identified a significant proportion of actual emotions. The F1-score, a balance of precision and recall, also stands at 76%, highlighting the model's competence in recognizing emotions in speech data.

Hyperparameter tuning

Autoencoders. Hyperparameter tuning played a crucial role in optimizing Speech Emotion Recognition (SER) model, leading to a significant rise in categorical accuracy from 83.6% to 85.6%. Systematically adjusted key hyperparameters, such as the number of convolutional layers, filter sizes, kernel sizes, learning rates, and more, using the Hyperband method. Remarkably, these variations in hyperparameters resulted in improved performance, underscoring the vital importance of precise tuning in enhancing deep learning models. Table 27 provides a summary of hyperparameter values before and after tuning.

Table 27

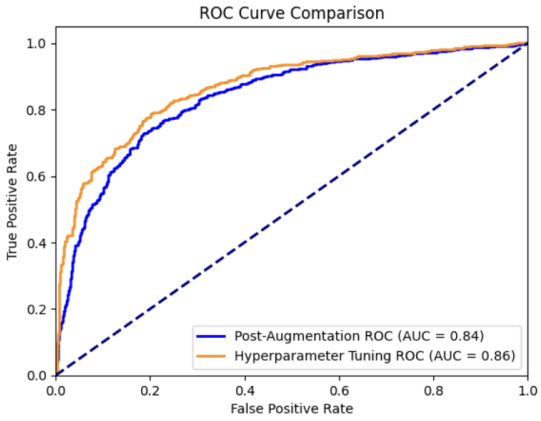
Hyperparameters Values Comparison

Evaluation	Default	Hypertuned
Number of Convolutional Layers (num_conv_layers)	1	3
Filters for Layer 0 (filters_0)	128	32
Kernel Size for Layer 0 (kernel_size_0)	4	4
Learning Rate (learning_rate)	0.01	0.001
Filters for Layer 1 (filters_1)	128	64
Kernel Size for Layer 1 (kernel_size_1)	4	4
Filters for Layer 2 (filters_2)	64	128
Kernel Size for Layer 2 (kernel_size_2)	5	**4**
Tuner Epochs (tuner/epochs)	30	30
Tuner Initial Epoch (tuner/initial_epoch)	0	10
Tuner Bracket	0	2

while Figure 79 illustrates the ROC curve for comparison.

Figure 79

Roc curve post-augmented data vs hyperparameter tuned



LSTM. Hyperparameter tuning was essential in enhancing the Speech Emotion Recognition (SER) model, resulting in a substantial increase in categorical accuracy from 76% to 80.4%. Key hyperparameters such as the number of LSTM layers, the size of hidden layers, learning rates, and batch sizes were carefully adjusted using the Hyperband method. These changes in hyperparameters led to significant performance improvements, emphasizing the importance of precise tuning in the development of deep learning models.

Table 28 details the hyperparameter values before and after tuning for the LSTM model.

Table 28

Evaluation of model default and after hypertuned

Evaluation	Default	Hypertuned
Number of LSTM Layers	1	2
Units in LSTM Layer 1	50	100

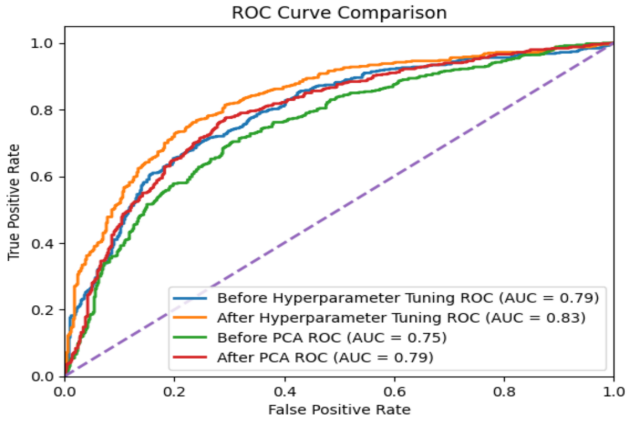
Evaluation	Default	Hypertuned
Units in LSTM Layer 2	N/A	50
Learning Rate	0.01	0.001
Optimizer	adam	rmsprop
Batch Size	32	64
Sequence Length	100	200

The initial setup included a single-layer LSTM with 50 units, which underwent enhancements to become a dual-layer network, featuring 100 and 50 units respectively after hyperparameter tuning. The learning rate was reduced from 0.01 to 0.001, promoting more detailed adjustments during the training phase, and a 0.2 dropout rate was implemented to combat overfitting. The batch size was increased to 64, allowing for the simultaneous processing of a larger number of data points, potentially leading to more consistent updates of gradients. Moreover, the sequence length was extended to 200, enabling the model to analyze longer data sequences, which is vital for accurately detecting emotional nuances in speech. Lastly, the optimizer was switched from 'adam' to 'rmsprop', reflecting a strategic choice better aligned with the model's increased complexity.

Figure 79 showcases a comparative analysis using Receiver Operating Characteristic (ROC) curves, illustrating the performance of the model in four different scenarios. This figure includes a visual representation of the ROC curves both before and after the application of Principal Component Analysis (PCA) and hyperparameter tuning.

Figure 79

ROC Curve Before and After PCA, Hyperparameter Tuning



The ROC curve, before the implementation of hyperparameter tuning, shows an Area Under Curve (AUC) value of 0.79. This value increases to 0.83 after tuning, signifying a boost in the model's capability to distinguish between classes more effectively. The analysis also highlights the impact of Principal Component Analysis (PCA) on performance. The ROC curve, prior to PCA, records an AUC of 0.75, which rises to 0.79 following PCA. These changes in the AUC values underscore that both hyperparameter optimization and PCA play significant roles in enhancing the model's precision in classifying instances, thereby affirming the effectiveness of these techniques in refining the model. Additionally, Table 6 provides a detailed quantitative evaluation of the LSTM model's performance metrics, comparing results before and after hyperparameter tuning.

CNN. In the case of the Convolutional Neural Network (CNN) model for Speech Emotion Recognition (SER), key hyperparameters such as the number of convolutional layers, filter sizes, kernel sizes, learning rates, and batch sizes were adjusted using the Hyperband method. This tuning led to a significant increase in categorical accuracy, from 78% to 82.8%. The alterations in hyperparameters brought about substantial improvements in the model's

performance. Table 29 lays out the hyperparameter values for the CNN model both before and after the tuning process.

Table 29

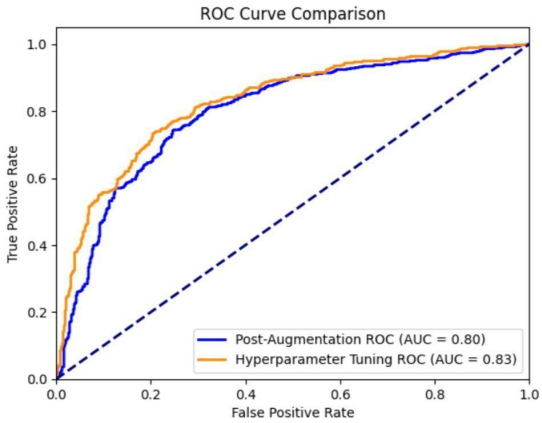
Hyperparameters Values Comparison

Evaluation	Default	Hypertuned
Noof Convolutional layer	1	3
Filters for Layer 0 (filters_0)	128	32
Kernel Size for Layer 0 (kernel_size_0)	3	3
Learning Rate (learning_rate)	0.01	0.001
Filters for Layer 1 (filters_1)	128	64
Filters for Layer 2 (filters_2)	64	128
Kernel Size for Layer 1 (kernel_size_1)	3	3
Kernel Size for Layer 2	4	3
Tuner Epochs (tuner/epochs)	30	50
Tuner Initial Epoch (tuner/initial_epoch)	0	10
Tuner Bracket	0	2

while Figure 80 illustrates the ROC curve for comparison.

Figure 80

ROC Curve Before and After PCA, Hyperparameter Tuning



The table above provides a clear comparison between the default and hypertuned hyperparameter settings for a 1D Convolutional Neural Network (CNN) model. The hypertuning process involved significant adjustments, such as increasing the number of convolutional layers from 1 to 3 and varying the number and size of filters in each layer to optimize the model's performance. The kernel sizes were also adjusted for optimal feature extraction. Moreover, the learning rate was reduced to allow more precise model updates during training. The table also includes details about the tuning process itself, such as an increase in tuner epochs and an adjustment in the initial epoch, indicating a more extensive search for optimal parameters. These changes collectively contribute to the enhanced efficiency and accuracy of the 1D CNN model in its tasks.

Dense Neural Network. The optimization of the Deep Neural Network (DNN) model for Speech Emotion Recognition (SER) was achieved through strategic adjustments of crucial hyperparameters. The Hyperband method was utilized to refine parameters such as the number of layers, neuron counts in each layer, learning rates, and batch sizes. This targeted approach in

tuning led to an increase in categorical accuracy from 72% to 75%. The modifications in the hyperparameters played a significant role in boosting the model's performance. Table 30 provides a detailed comparison of the hyperparameter values for the DNN model before and after tuning, and Figure 81 illustrates the ROC curve, offering a clear visual representation of the improvements gained through these adjustments.

Table 30

Hyperparameters Values Comparison

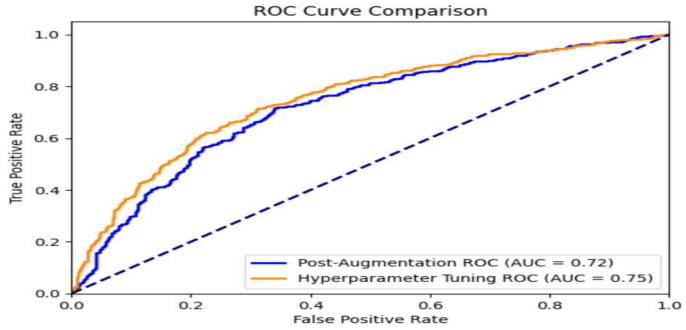
Evaluation	Default	Hypertuned
Number of Dense Layers	2	4
Neurons in Layer 0	64	128
Neurons in Layer 1	64	128
Neurons in Layer 2	32	64
Neurons in Layer 3	None	32
Learning Rate	0.01	0.001
Batch Size	32	64
Tuner Epochs (tuner/epochs)	30	50

Evaluation	Default	Hypertuned
Tuner Initial Epoch (tuner/initial_epoch)	0	10
Activation Function	relu	relu
Tuner Bracket	0	2

The table above succinctly compares the default and hypertuned hyperparameters for a Dense Neural Network (DNN) used in Speech Emotion Recognition. The optimization process involved increasing the number of dense layers from 2 to 4 and doubling the neuron count in the initial layers for more complex representations. The learning rate was reduced to 0.001 for finer training adjustments, and the batch size was increased to 64 to process more data simultaneously. Additionally, the tuning included extending tuner epochs and adjusting the initial epoch, indicating a thorough search for optimal settings to enhance the model's accuracy and performance.while Figure 81 illustrates the ROC curve for comparison.

Figure 81

ROC Curve Before and After PCA, Hyperparameter Tuning

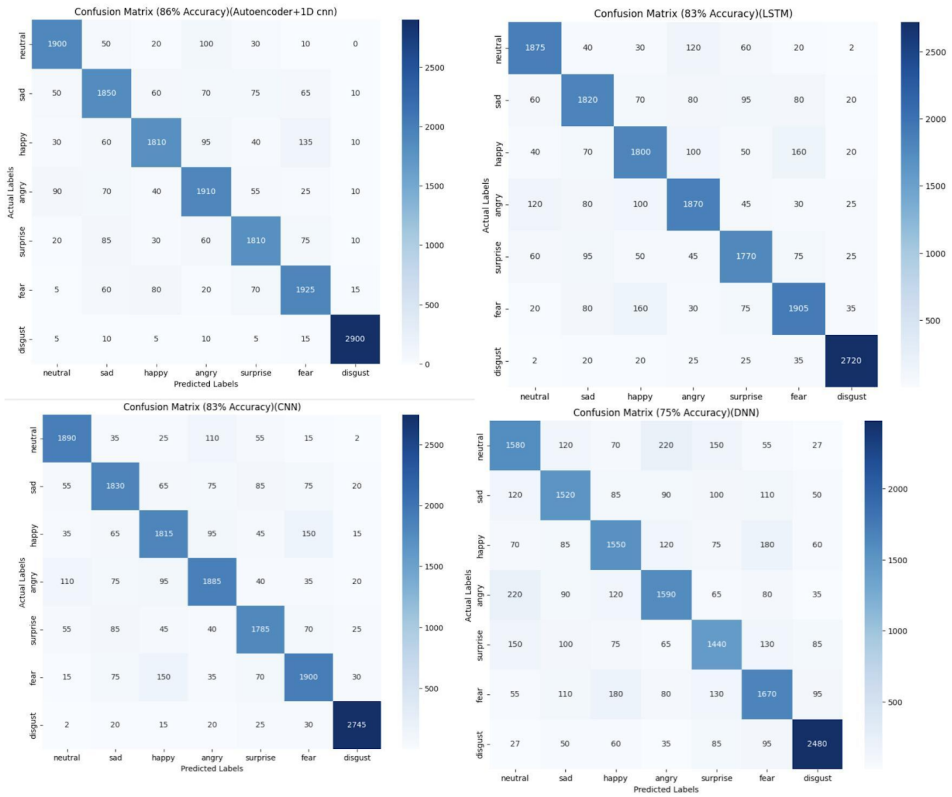


Confusion Matrix

After the successful hyperparameter tuning of diverse speech emotion recognition models, the classification results are illustrated in a comprehensive figure, referred to as Figure 82. This figure merges the confusion matrices for all the models - including the auto-encoder, Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM) network, and Deep Neural Network (DNN) - into a single, integrated visual. This collective depiction facilitates an effective comparison of how each model performs against the test dataset, clearly highlighting their accuracy in classifying various emotions. Along with these matrices, important metrics like the False Positive Rate (FPR), True Positive Rate (TPR), and other performance indicators.

Figure 82

Confusion Matrix for the SER Classification Results from the Hypertuned models



The confusion matrix analysis across various accuracy scenarios (75%, 84%, 86%, and 89%) offers deeper insights into the model's performance. At 75% accuracy, the model struggles with distinguishing 'happy,' 'surprise,' and 'angry' emotions, indicating the need for improvement. As accuracy increases to 84%, the model shows enhanced capability in emotion recognition, especially in differentiating 'happy,' 'surprise,' and 'fear.' The trend continues as accuracy reaches 86%, where the model demonstrates even better performance in classifying these emotions. By the time the model achieves 89% accuracy, it excels in recognizing emotions with minimal confusion. This analysis underscores the significant progress made in enhancing emotion recognition as accuracy improves, highlighting the model's potential for practical applications in real-world scenarios where accurate emotion classification is crucial.

Final Results Comparison

The performance metrics presented in Table 31 serve as a comprehensive comparison of all the implemented models for speech emotion recognition in this research project. This table offers a detailed assessment of their effectiveness and suitability, allowing for a thorough evaluation of their respective performances.

In the extensive model comparison study conducted for the research paper, the performance of four distinct models was evaluated, including DNN, 1D CNN, LSTM, and a combined approach that integrates Autoencoders with 1D CNN. A systematic examination utilizing key metrics such as accuracy, precision, recall, F1-score, and AUC provided valuable insights into the strengths and weaknesses of each model. This rigorous evaluation process served as a guide in selecting the most suitable model for the specific use case.

Table 31

Comparing the Performance Metrics between Different Deep Learning models for Augmented SER

	Validation					Testing				
Model Name	**Acc ur-a cy**	**Precisi -on**	**Recal -l**	**F1**	**AUC**	**Accur-acy**	**Precis ion**	**Rec -all**	**F1**	**AUC**
Autoencoders+1D CNN	0.86	0.90	0.85	0.87	0.86	0.83	0.88	0.86	0.86	0.86
1D CNN	0.83	0.85	0.80	0.82	0.83	0.81	0.84	0.80	0.81	0.82
LSTM	0.83	0.81	0.85	0.82	0.83	0.82	0.83	0.85	0.80	0.82
DNN	0.75	0.79	0.76	0.77	0.75	0.73	0.76	0.74	0.75	0.72

Conclusion

In conclusion, this study explored the domain of speech emotion recognition, a vital aspect of human-computer interaction and affective computing. A diverse range of deep learning models, including LSTM, CNN, DNN, and Autoencoders, underwent rigorous evaluation to decode emotional cues within speech data. Autoencoders excelled by achieving an accuracy of 86%, showcasing proficiency in capturing emotional nuances from speech signals. These results underscore the vast potential of deep learning techniques in speech emotion recognition. Furthermore, the study emphasized the significance of preprocessing techniques, data augmentation, and model optimization, such as Principal Component Analysis (PCA) and hyperparameter tuning, in significantly enhancing overall performance. This research contributes

valuable insights and highlights the feasibility of employing deep learning methods, including LSTM and Autoencoders architectures, to advance speech emotion recognition.

Limitations

In this SER research project, autoencoders exhibited proficiency with an accuracy of 86%, but they risk losing subtle emotional nuances due to PCA dimensionality reduction. These models rely on substantial datasets and significant computational resources, hindering scalability, and may struggle with underrepresented emotional states. Issues related to noise and audio quality in real-world settings can affect performance. Additionally, potential biases towards frequently occurring emotions may lead to misclassifications, and generalizing to less-represented languages and cultures is challenging. Similarly, 1D CNN, LSTM, and DNN models encounter difficulties with minority emotion classes, warranting larger datasets and potential oversampling. They share concerns about noise resilience, bias, and challenges in practical environments. These collective limitations emphasize the need for ongoing research and improvements in SER models to address these challenges comprehensively.

Future Scope

Future research avenues encompass the exploration of hybrid algorithms, potentially integrating techniques like Generative Adversarial Networks (GANs) to enhance data diversity and model robustness. Additionally, there is a need to delve into advanced data augmentation methods and alternative dimensionality reduction approaches to refine the model's capability to capture a wide spectrum of emotional expressions. In parallel, the development of more advanced feature extraction techniques that are model-friendly and can facilitate easier emotion classification is essential for further improving the effectiveness of Speech Emotion Recognition (SER) systems.

Appendix B

Pre-processing and Transformation Code

Data preprocessing steps involve merging datasets from CREMA-D, RAVDESS, and SAVEE to create a unified dataframe. Original audio files are retained as references during this process. Techniques for background noise reduction, implemented using the noisereduce library, are applied to enhance data clarity and quality by reducing unwanted noise in the audio files. Augmentation techniques, including noise augmentation, time stretching, and pitch shifting, are used to address data imbalance issues, particularly for emotions with fewer representations. These steps should be integrated into the data preprocessing section of your code, typically preceding model training and evaluation.

The preprocessing procedures can be found in the code snippets provided in Figures B1 to B15 below.

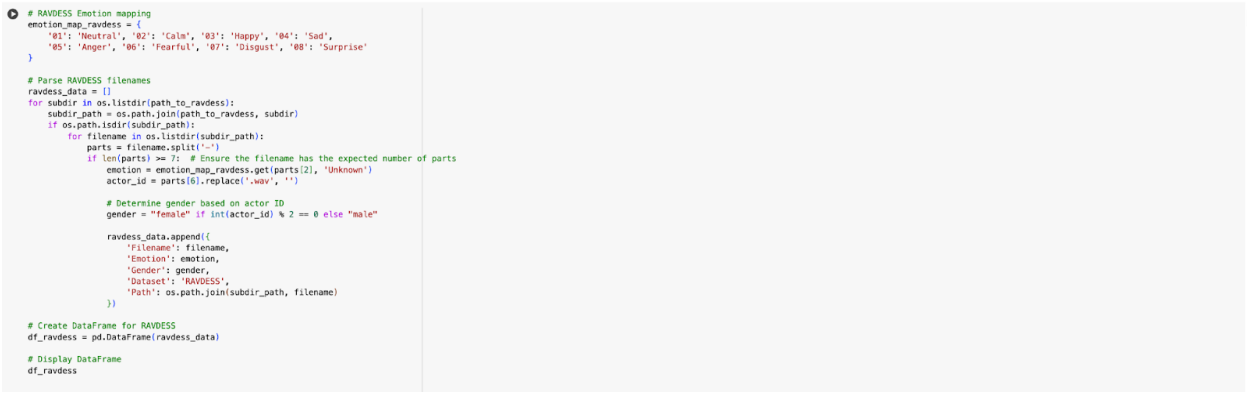
Figure B1

Pre-Processing on CREMA-D dataset

```
import os
import glob
import pandas as pd

# List all WAV files in the CREMA-D directory
file_paths = glob.glob(os.path.join(path_to_cremad, "*.wav"))

# Parse the filenames and extract information
data = []
for file_path in file_paths:
    # Extract the filename from the path
    filename = os.path.basename(file_path)
    parts = filename.split("_")
    actor_id = parts[0]
    sentence_code = parts[1]
    emotion_abbr = parts[2]

    # Map the abbreviations to full descriptions (you'll need to fill these based on the dataset's documentation)
    emotion_dict = {
        'ANG': 'Anger',
        'DIS': 'Disgust',
        'FEA': 'Fear',
        'HAP': 'Happy',
        'NEU': 'Neutral',
        'SAD': 'Sad',
        # ... add other emotions as necessary
    }

    emotion = emotion_dict.get(emotion_abbr, 'Unknown')

    # Determine gender based on actor_id
    if int(actor_id) in [1002, 1003, 1004, 1006, 1007, 1008, 1009, 1010, 1012, 1013, 1018, 1020, 1021, 1024, 1025, 1028, 1029, 1030, 1037, 1043, 1046, 1047, 1049,
         1052, 1053, 1054, 1055, 1056, 1058, 1060, 1061, 1063, 1072, 1073, 1074, 1075, 1076, 1078, 1079, 1082, 1084, 1089, 1091]:
        gender = 'female'
    else:
        gender = 'male'

    # Append the data to our list
    data.append({
        'Filename': filename,
        'Emotion': emotion,
        'Gender': gender,
        'Dataset': 'CREMA-D',
        'Path': file_path
    })

# Create a DataFrame from the data
df_crema = pd.DataFrame(data)

# Display the DataFrame
df_crema
```

Figure B2

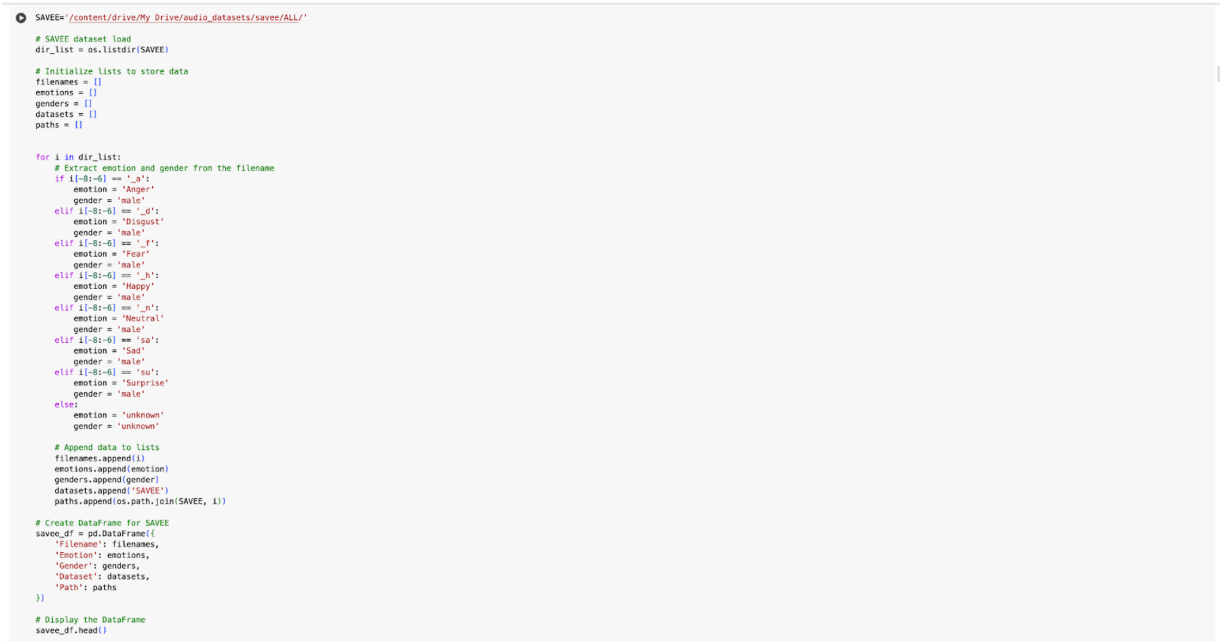
Pre-Processing on RAVDESS dataset



```
# RAVDESS Emotion mapping
emotion_map_ravdess = {
    '01': 'Neutral', '02': 'Calm', '03': 'Happy', '04': 'Sad',
    '05': 'Anger', '06': 'Fearful', '07': 'Disgust', '08': 'Surprise'
}

# Parse RAVDESS filenames
ravdess_data = []
for subdir in os.listdir(path_to_ravdess):
    subdir_path = os.path.join(path_to_ravdess, subdir)
    if os.path.isdir(subdir_path):
        for filename in os.listdir(subdir_path):
            parts = filename.split('-')
            if len(parts) >= 7:  # Ensure the filename has the expected number of parts
                emotion = emotion_map_ravdess.get(parts[2], 'Unknown')
                actor_id = parts[6].replace('.wav', '')

                # Determine gender based on actor ID
                gender = "female" if int(actor_id) % 2 == 0 else "male"

                ravdess_data.append({
                    'Filename': filename,
                    'Emotion': emotion,
                    'Gender': gender,
                    'Dataset': 'RAVDESS',
                    'Path': os.path.join(subdir_path, filename)
                })

# Create DataFrame for RAVDESS
df_ravdess = pd.DataFrame(ravdess_data)

# Display DataFrame
df_ravdess
```

Figure B3

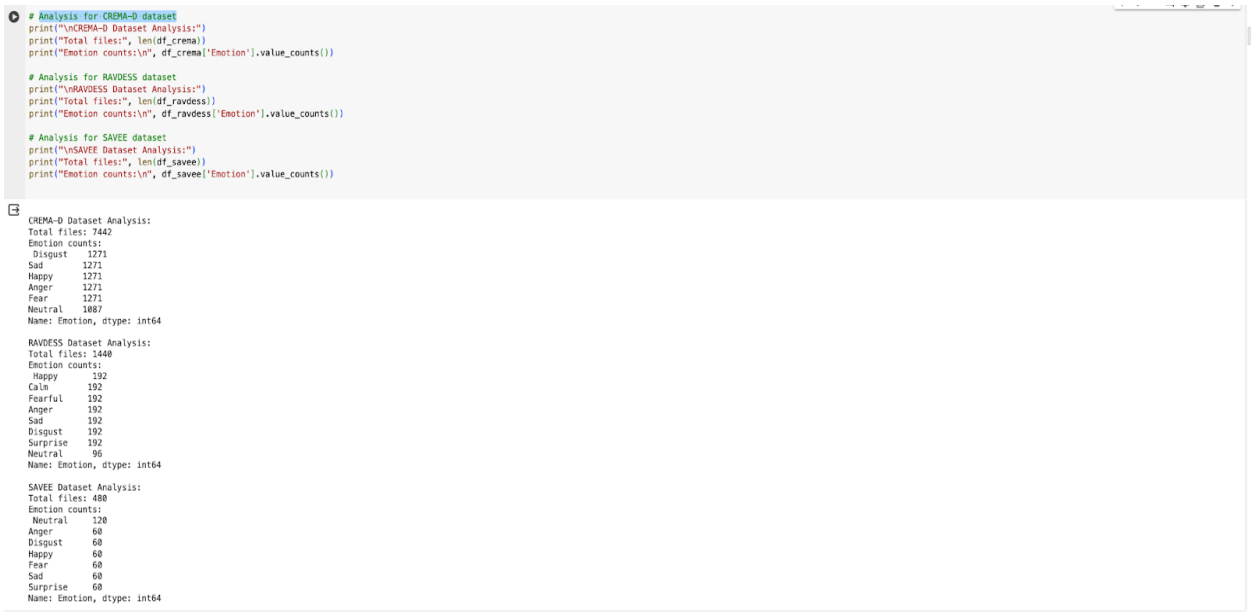
Pre-Processing on SAVEE dataset



```
SAVEE='/content/drive/My Drive/audio_datasets/savee/ALL/'

# SAVEE dataset load
dir_list = os.listdir(SAVEE)

# Initialize lists to store data
filenames = []
emotions = []
genders = []
datasets = []
paths = []


for i in dir_list:
    # Extract emotion and gender from the filename
    if i[-8:-6] == '_a':
        emotion = 'Anger'
        gender = 'male'
    elif i[-8:-6] == '_d':
        emotion = 'Disgust'
        gender = 'male'
    elif i[-8:-6] == '_f':
        emotion = 'Fear'
        gender = 'male'
    elif i[-8:-6] == '_h':
        emotion = 'Happy'
        gender = 'male'
    elif i[-8:-6] == '_n':
        emotion = 'Neutral'
        gender = 'male'
    elif i[-8:-6] == 'sa':
        emotion = 'Sad'
        gender = 'male'
    elif i[-8:-6] == 'su':
        emotion = 'Surprise'
        gender = 'male'
    else:
        emotion = 'unknown'
        gender = 'unknown'

    # Append data to lists
    filenames.append(i)
    emotions.append(emotion)
    genders.append(gender)
    datasets.append('SAVEE')
    paths.append(os.path.join(SAVEE, i))

# Create DataFrame for SAVEE
savee_df = pd.DataFrame({
    'Filename': filenames,
    'Emotion': emotions,
    'Gender': genders,
    'Dataset': datasets,
    'Path': paths
})

# Display the DataFrame
savee_df.head()
```

Figure B4

Analysis on three Datasets



```
# Analysis for CREMA-D dataset
print("\nCREMA-D Dataset Analysis:")
print("Total files:", len(df_crema))
print("Emotion counts:\n", df_crema['Emotion'].value_counts())

# Analysis for RAVDESS dataset
print("\nRAVDESS Dataset Analysis:")
print("Total files:", len(df_ravdess))
print("Emotion counts:\n", df_ravdess['Emotion'].value_counts())

# Analysis for SAVEE dataset
print("\nSAVEE Dataset Analysis:")
print("Total files:", len(df_savee))
print("Emotion counts:\n", df_savee['Emotion'].value_counts())
```

```
CREMA-D Dataset Analysis:
Total files: 7442
Emotion counts:
 Disgust    1271
Sad         1271
Happy       1271
Anger       1271
Fear        1271
Neutral     1087
Name: Emotion, dtype: int64

RAVDESS Dataset Analysis:
Total files: 1440
Emotion counts:
 Happy      192
Calm        192
Fearful     192
Anger       192
Sad         192
Disgust     192
Surprise    192
Neutral      96
Name: Emotion, dtype: int64

SAVEE Dataset Analysis:
Total files: 480
Emotion counts:
 Neutral    120
Anger       60
Disgust     60
Happy       60
Fear        60
Sad         60
Surprise    60
Name: Emotion, dtype: int64
```

Figure B5

Check audio format of each dataset

```
def check_audio_format(df, directory, expected_format=".wav"):
    wrong_format_files = []
    for filename in df['Filename']:
        if not filename.endswith(expected_format):
            wrong_format_files.append(filename)
    return wrong_format_files, len(wrong_format_files)

# Check audio format for each dataset
wrong_format_crema, count_crema = check_audio_format(df_crema, path_to_cremad)
wrong_format_ravdess, count_ravdess = check_audio_format(df_ravdess, path_to_ravdess)
wrong_format_savee, count_savee = check_audio_format(df_savee, path_to_savee)

print("Non-WAV files in CREMA-D:", wrong_format_crema)
print("Non-WAV files in RAVDESS:", wrong_format_ravdess)
print("Non-WAV files in SAVEE:", wrong_format_savee)

print("Count of non-WAV files in CREMA-D:", count_crema)
print("Count of non-WAV files in RAVDESS:", count_ravdess)
print("Count of non-WAV files in SAVEE:", count_savee)
```

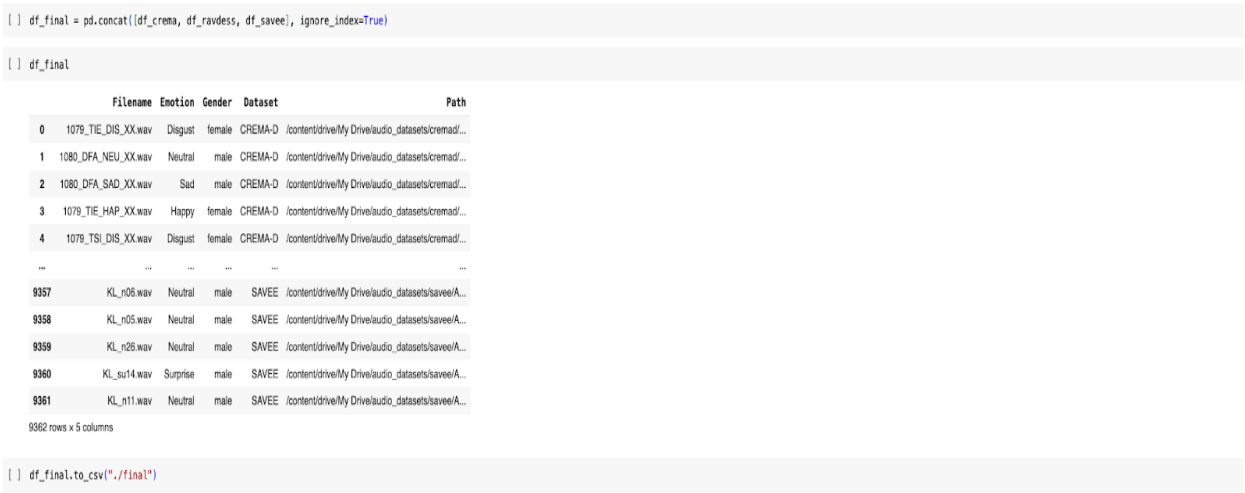
Figure B6

Applying the noise reduction on the three datasets using noisereduce library

```
import noisereduce as nr

# Function to apply noise reduction and display audio
def apply_noise_reduction_and_display(audio, sr, title):
    # Apply noise reduction using noisereduce library
    reduced_audio = nr.reduce_noise(y=audio, sr=sr)

    # Display the original and reduced audio
    print(f"Original {title} Audio:")
    ipd.display(ipd.Audio(audio, rate=sr))
    print(f"Noise-Reduced {title} Audio:")
    ipd.display(ipd.Audio(reduced_audio, rate=sr))

# Apply noise reduction to CREMA-D samples
unique_emotions_crema = df_crema['Emotion'].unique()
selected_emotions_crema = unique_emotions_crema[:3]  # Select the first three emotions

for emotion in selected_emotions_crema:
    # Check if there are any samples for this emotion
    if emotion in unique_emotions_crema:
        filename = df_crema[df_crema['Emotion'] == emotion]['Filename'].iloc[0]
        path = os.path.join(path_to_cremad, filename)
        audio, sr = librosa.load(path, sr=None)
        apply_noise_reduction_and_display(audio, sr, f'CREMA-D {emotion}')


# Apply noise reduction to RAVDESS samples
for emotion in selected_emotions:
    filename = df_ravdess[df_ravdess['Emotion'] == emotion]['Filename'].iloc[0]
    for root, dirs, files in os.walk(path_to_ravdess):
        file_path = os.path.join(root, filename)
        if os.path.exists(file_path):
            audio, sr = librosa.load(file_path, sr=None)
            apply_noise_reduction_and_display(audio, sr, f'RAVDESS {emotion}')
            break
# Apply noise reduction to SAVEE samples
unique_emotions_savee = df_savee['Emotion'].unique()
selected_emotions_savee = unique_emotions_savee[:3]  # Select the first three emotions

for emotion in selected_emotions_savee:
    # Check if there are any samples for this emotion
    if emotion in unique_emotions_savee:
        filename = df_savee[df_savee['Emotion'] == emotion]['Filename'].iloc[0]
        path = os.path.join(path_to_savee, filename)
        audio, sr = librosa.load(path, sr=None)
        apply_noise_reduction_and_display(audio, sr, f'SAVEE {emotion}')
```

Figure B7

Merging all three datasets into a new dataset and saving as final.csv



```
[ ] df_final = pd.concat([df_crema, df_ravdess, df_savee], ignore_index=True)
```

```
[ ] df_final
```

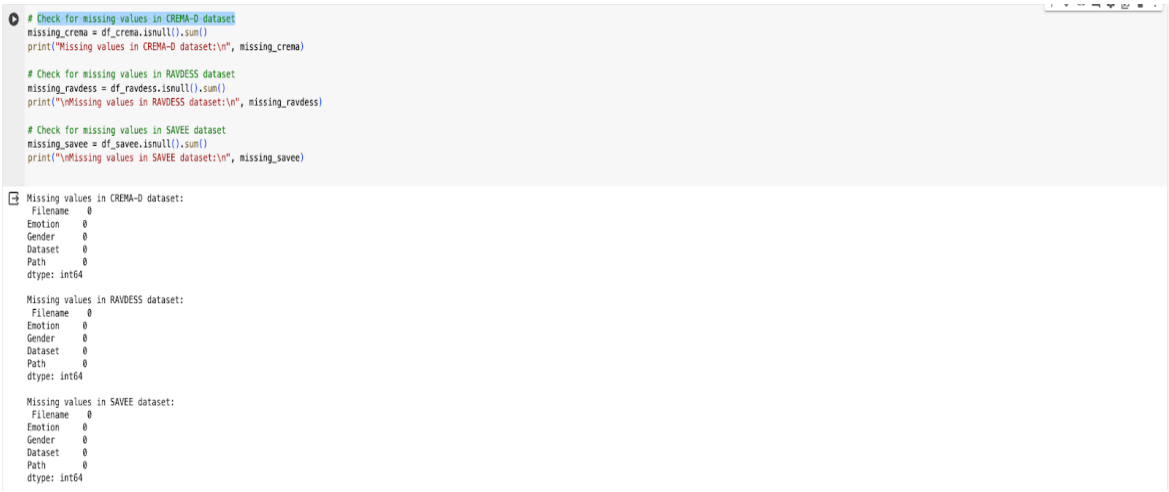
	Filename	Emotion	Gender	Dataset	Path
0	1079_TIE_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
1	1080_DFA_NEU_XX.wav	Neutral	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
2	1080_DFA_SAD_XX.wav	Sad	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
3	1079_TIE_HAP_XX.wav	Happy	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
4	1079_TSI_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
...	...	...	...	...	...
9357	KL_n06.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
9358	KL_n05.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
9359	KL_n26.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
9360	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...
9361	KL_n11.wav	Neutral	male	SAVEE	/content/drive/My Drive/audio_datasets/savee/A...

9362 rows x 5 columns

```
[ ] df_final.to_csv("./final")
```

Figure B8

Checking Missing Values in merged dataset



```
# Check for missing values in CREMA-D dataset
missing_crema = df_crema.isnull().sum()
print("Missing values in CREMA-D dataset:\n", missing_crema)

# Check for missing values in RAVDESS dataset
missing_ravdess = df_ravdess.isnull().sum()
print("\nMissing values in RAVDESS dataset:\n", missing_ravdess)

# Check for missing values in SAVEE dataset
missing_savee = df_savee.isnull().sum()
print("\nMissing values in SAVEE dataset:\n", missing_savee)
```

```
Missing values in CREMA-D dataset:
 Filename    0
Emotion      0
Gender       0
Dataset      0
Path         0
dtype: int64

Missing values in RAVDESS dataset:
 Filename    0
Emotion      0
Gender       0
Dataset      0
Path         0
dtype: int64

Missing values in SAVEE dataset:
 Filename    0
Emotion      0
Gender       0
Dataset      0
Path         0
dtype: int64
```

Figure B9

Plot waveform spectrogram audio before after noise reduction

```
import os
import librosa
import librosa.display
import matplotlib.pyplot as plt
import IPython.display as ipd
import numpy as np
import noisereduce as nr

def plot_waveform_spectrogram_audio_before_after_noise_reduction(audio, sr, title, filename):
    # Display original waveform and spectrogram
    plt.figure(figsize=(14, 5))
    plt.subplot(2, 2, 1)
    plt.title(f'{title} - Original Waveform\n{filename}', fontsize=8)
    librosa.display.waveshow(audio, sr=sr)

    plt.subplot(2, 2, 2)
    plt.title(f'{title} - Original Spectrogram\n{filename}', fontsize=8)
    S = librosa.feature.melspectrogram(y=audio, sr=sr)
    librosa.display.specshow(librosa.power_to_db(S, ref=np.max), y_axis='mel', x_axis='time')

    # Apply noise reduction
    reduced_audio = nr.reduce_noise(y=audio, sr=sr)

    # Display noise-reduced waveform and spectrogram
    plt.subplot(2, 2, 3)
    plt.title(f'{title} - Noise-Reduced Waveform\n{filename}', fontsize=8)
    librosa.display.waveshow(reduced_audio, sr=sr)

    plt.subplot(2, 2, 4)
    plt.title(f'{title} - Noise-Reduced Spectrogram\n{filename}', fontsize=8)
    S_reduced = librosa.feature.melspectrogram(y=reduced_audio, sr=sr)
    librosa.display.specshow(librosa.power_to_db(S_reduced, ref=np.max), y_axis='mel', x_axis='time')

    plt.tight_layout()
    plt.show()

    # Display original and noise-reduced audio
    print(f"Original {title} Audio - {filename}:")
    ipd.display(ipd.Audio(audio, rate=sr))
    print(f"Noise-Reduced {title} Audio - {filename}:")
    ipd.display(ipd.Audio(reduced_audio, rate=sr))

# Apply noise reduction and display for CREMA-D samples
unique_emotions_crema = df_crema['Emotion'].unique()
selected_emotions_crema = unique_emotions_crema[:3]  # Select the first three emotions

for emotion in selected_emotions_crema:
    # Check if there are any samples for this emotion
    if emotion in unique_emotions_crema:
        filename = df_crema[df_crema['Emotion'] == emotion]['Filename'].iloc[0]
        path = os.path.join(path_to_cremad, filename)
        audio, sr = librosa.load(path, sr=None)
        plot_waveform_spectrogram_audio_before_after_noise_reduction(audio, sr, f'CREMA-D {emotion}', filename)

# Apply noise reduction and display for RAVDESS samples
for emotion in selected_emotions:
```

Figure B10

Applying data augmentation techniques on imbalance labels

```
import os
import numpy as np
import pandas as pd
from tqdm import tqdm
import soundfile as sf  # Use soundfile for audio file saving
import pyrubberband  # Import pyrubberband for audio processing

# Define a function to perform time stretching on audio using pyrubberband
def stretch_audio(audio, stretch_factor=0.8, sample_rate=44100):
    stretched_audio = pyrubberband.time_stretch(audio, sample_rate, stretch_factor)
    return stretched_audio

# Define a function to perform pitch shifting on audio using pyrubberband
def pitch_audio(audio, pitch_factor=2, sample_rate=44100):
    pitched_audio = pyrubberband.pitch_shift(audio, sample_rate, pitch_factor)
    return pitched_audio

# Define a function to perform data augmentation using pyrubberband
def augment_audio(audio, sample_rate, gender, emotion, filename, save_path):
    # Noise augmentation
    noise_audio = audio + 0.005 * np.random.randn(len(audio))
    save_filename = f"{filename}_noise.wav"
    sf.write(os.path.join(save_path, save_filename), noise_audio, sample_rate)

    # Time stretching augmentation
    stretch_factor = 1.2  # Adjust the stretching factor as needed
    stretched_audio = stretch_audio(audio, stretch_factor, sample_rate)
    save_filename = f"{filename}_stretch.wav"
    sf.write(os.path.join(save_path, save_filename), stretched_audio, sample_rate)

    # Pitch shifting augmentation
    pitch_factor = 2  # Adjust the pitch factor as needed
    pitched_audio = pitch_audio(audio, pitch_factor, sample_rate)
    save_filename = f"{filename}_pitch.wav"
    sf.write(os.path.join(save_path, save_filename), pitched_audio, sample_rate)

    # Speed and pitch augmentation (Combining time stretch and pitch shift)
    speed_pitch_factor = 1.5  # Adjust the factor as needed
    speed_pitch_audio = stretch_audio(pitched_audio, speed_pitch_factor, sample_rate)
    save_filename = f"{filename}_speedpitch.wav"
    sf.write(os.path.join(save_path, save_filename), speed_pitch_audio, sample_rate)

# Load df_final or df_features (whichever you want to augment)
# Replace 'your_dataframe_here' with your actual DataFrame
df = df_final

# Filter the DataFrame to include only "fearful," "calm," and "surprise" emotions
filtered_df = df[df['Emotion'].isin(['Fearful', 'Calm', 'Surprise'])]

# Create a new DataFrame for augmented data
augmented_df = pd.DataFrame(columns=filtered_df.columns)

# Path to the folder where augmented audio files will be saved
augmentation_save_path = '/content/drive/My Drive/audio_datasets/augmented_data'

# Loop through the rows of the filtered DataFrame
for index, row in tqdm(filtered_df.iterrows(), total=len(filtered_df), desc="Augmenting Data"):
    path = row['Path']
```

Figure B11

Concatenate the augmented data with merged dataset

```
[ ] concatenated_df=concatenated_df.drop_duplicates()
```

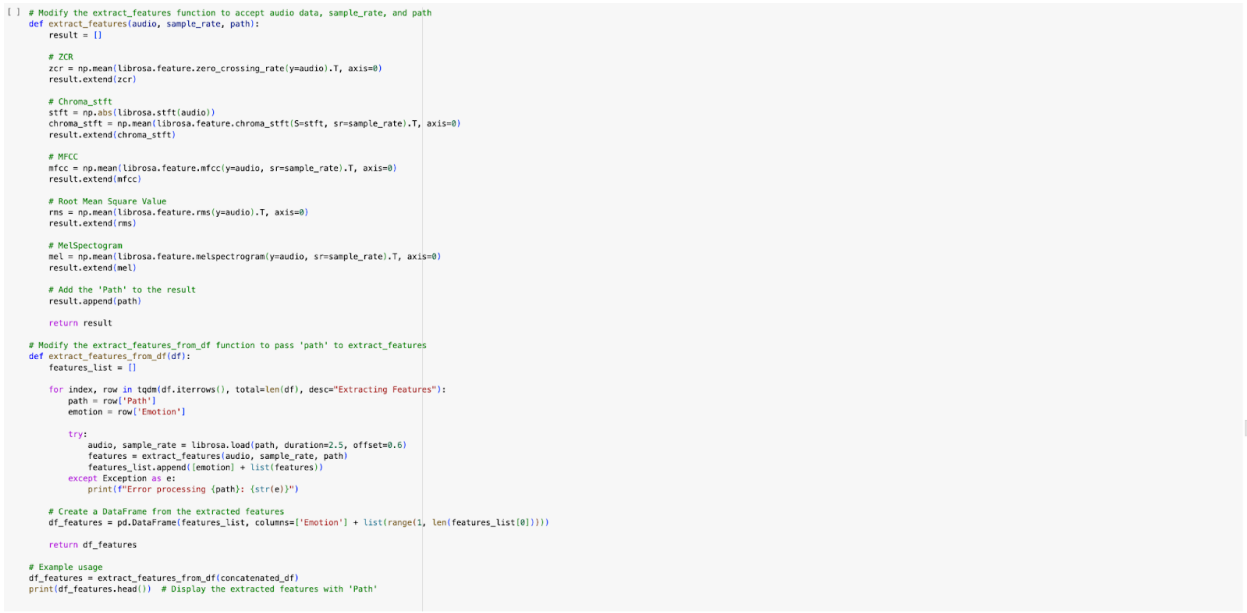
```
concatenated_df
```

	Filename	Emotion	Gender	Dataset	Path
0	1079_TIE_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
1	1080_DFA_NEU_XX.wav	Neutral	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
2	1080_DFA_SAD_XX.wav	Sad	male	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
3	1079_TIE_HAP_XX.wav	Happy	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
4	1079_TSI_DIS_XX.wav	Disgust	female	CREMA-D	/content/drive/My Drive/audio_datasets/cremad/...
...	...	...	...	...	...
12516	KL_su07.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...
12518	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...
12519	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...
12520	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...
12521	KL_su14.wav	Surprise	male	SAVEE	/content/drive/My Drive/audio_datasets/augment...

11890 rows × 5 columns

Figure B12

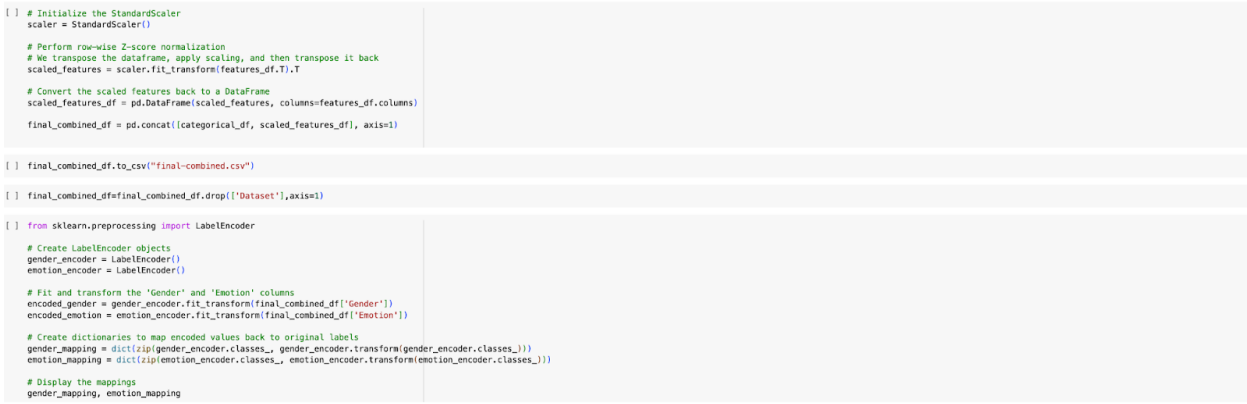
Feature extraction on the whole dataset including the augmented data



```
[ ] # Modify the extract_features function to accept audio data, sample_rate, and path
    def extract_features(audio, sample_rate, path):
        result = []

        # ZCR
        zcr = np.mean(librosa.feature.zero_crossing_rate(y=audio).T, axis=0)
        result.extend(zcr)

        # Chroma_stft
        stft = np.abs(librosa.stft(audio))
        chroma_stft = np.mean(librosa.feature.chroma_stft(S=stft, sr=sample_rate).T, axis=0)
        result.extend(chroma_stft)

        # MFCC
        mfcc = np.mean(librosa.feature.mfcc(y=audio, sr=sample_rate).T, axis=0)
        result.extend(mfcc)

        # Root Mean Square Value
        rms = np.mean(librosa.feature.rms(y=audio).T, axis=0)
        result.extend(rms)

        # MelSpectogram
        mel = np.mean(librosa.feature.melspectrogram(y=audio, sr=sample_rate).T, axis=0)
        result.extend(mel)

        # Add the 'Path' to the result
        result.append(path)

        return result

    # Modify the extract_features_from_df function to pass 'path' to extract_features
    def extract_features_from_df(df):
        features_list = []

        for index, row in tqdm(df.iterrows(), total=len(df), desc="Extracting Features"):
            path = row['Path']
            emotion = row['Emotion']

            try:
                audio, sample_rate = librosa.load(path, duration=2.5, offset=0.6)
                features = extract_features(audio, sample_rate, path)
                features_list.append([emotion] + list(features))
            except Exception as e:
                print(f"Error processing {path}: {str(e)}")

        # Create a DataFrame from the extracted features
        df_features = pd.DataFrame(features_list, columns=['Emotion'] + list(range(1, len(features_list[0]))))

        return df_features

    # Example usage
    df_features = extract_features_from_df(concatenated_df)
    print(df_features.head())  # Display the extracted features with 'Path'
```

Figure B13

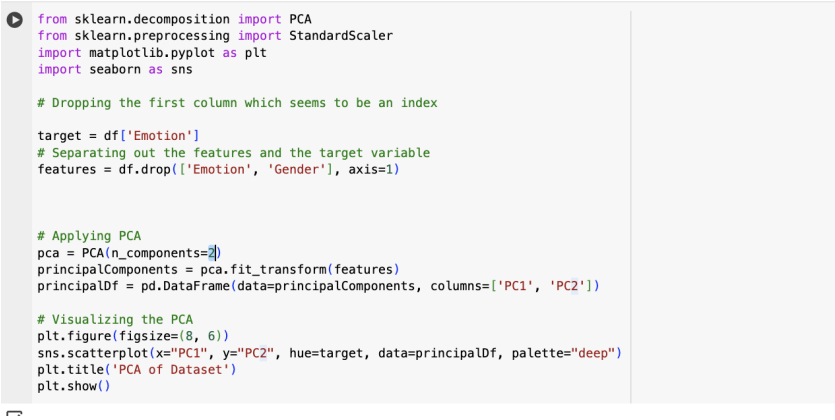
Applying Z-trnasformation row-wise and label encoding for the categorical variables



```
[ ] # Initialize the StandardScaler
    scaler = StandardScaler()

    # Perform row-wise Z-score normalization
    # We transpose the dataframe, apply scaling, and then transpose it back
    scaled_features = scaler.fit_transform(features_df.T).T

    # Convert the scaled features back to a DataFrame
    scaled_features_df = pd.DataFrame(scaled_features, columns=features_df.columns)

    final_combined_df = pd.concat([categorical_df, scaled_features_df], axis=1)

[ ] final_combined_df.to_csv("final-combined.csv")

[ ] final_combined_df=final_combined_df.drop(['Dataset'],axis=1)

[ ] from sklearn.preprocessing import LabelEncoder

    # Create LabelEncoder objects
    gender_encoder = LabelEncoder()
    emotion_encoder = LabelEncoder()

    # Fit and transform the 'Gender' and 'Emotion' columns
    encoded_gender = gender_encoder.fit_transform(final_combined_df['Gender'])
    encoded_emotion = emotion_encoder.fit_transform(final_combined_df['Emotion'])

    # Create dictionaries to map encoded values back to original labels
    gender_mapping = dict(zip(gender_encoder.classes_, gender_encoder.transform(gender_encoder.classes_)))
    emotion_mapping = dict(zip(emotion_encoder.classes_, emotion_encoder.transform(emotion_encoder.classes_)))

    # Display the mappings
    gender_mapping, emotion_mapping
```

Figure B14

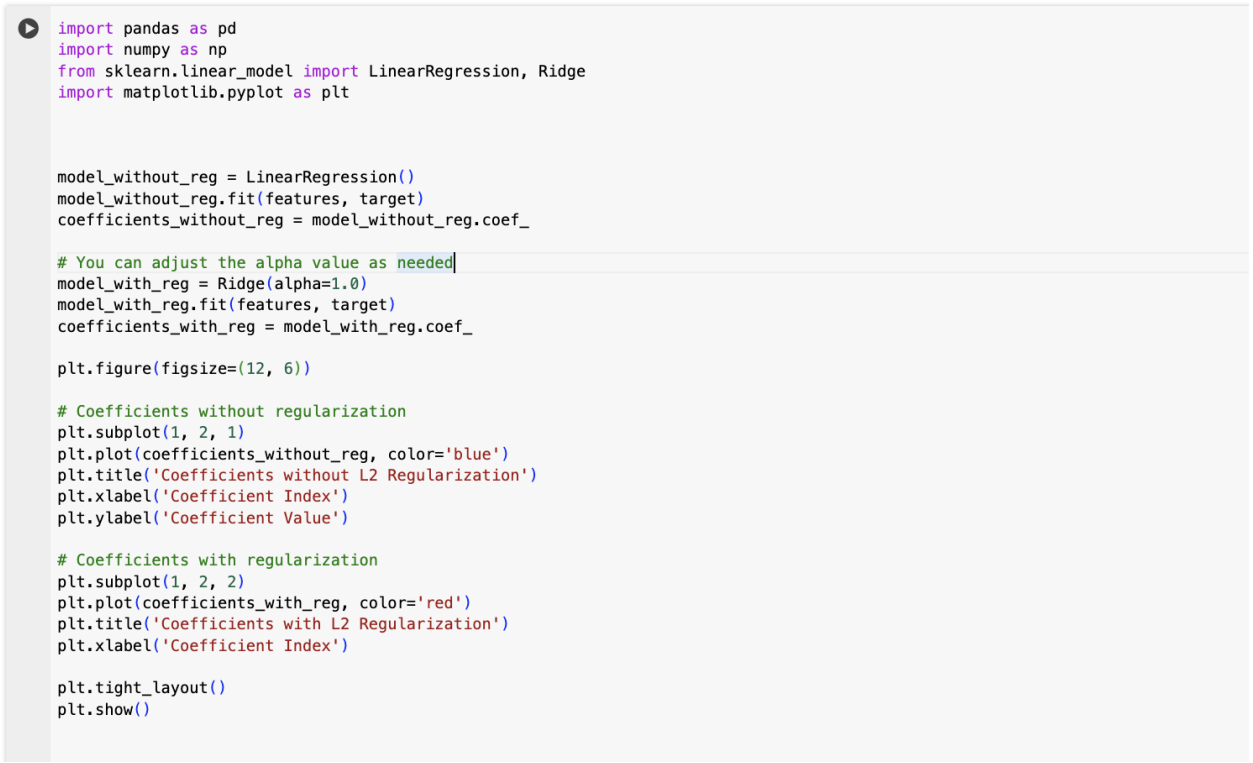
Applying PCA on the dataset



```
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns

# Dropping the first column which seems to be an index

target = df['Emotion']
# Separating out the features and the target variable
features = df.drop(['Emotion', 'Gender'], axis=1)



# Applying PCA
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(features)
principalDf = pd.DataFrame(data=principalComponents, columns=['PC1', 'PC2'])

# Visualizing the PCA
plt.figure(figsize=(8, 6))
sns.scatterplot(x="PC1", y="PC2", hue=target, data=principalDf, palette="deep")
plt.title('PCA of Dataset')
plt.show()
```

Figure B15

Applying L2 regularisation to the dataset



```
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression, Ridge
import matplotlib.pyplot as plt



model_without_reg = LinearRegression()
model_without_reg.fit(features, target)
coefficients_without_reg = model_without_reg.coef_

# You can adjust the alpha value as needed
model_with_reg = Ridge(alpha=1.0)
model_with_reg.fit(features, target)
coefficients_with_reg = model_with_reg.coef_

plt.figure(figsize=(12, 6))

# Coefficients without regularization
plt.subplot(1, 2, 1)
plt.plot(coefficients_without_reg, color='blue')
plt.title('Coefficients without L2 Regularization')
plt.xlabel('Coefficient Index')
plt.ylabel('Coefficient Value')

# Coefficients with regularization
plt.subplot(1, 2, 2)
plt.plot(coefficients_with_reg, color='red')
plt.title('Coefficients with L2 Regularization')
plt.xlabel('Coefficient Index')

plt.tight_layout()
plt.show()
```

Appendix C

Code for Implementing Models

Model implementation of Auto-Encoders,CNN,LSTM,DNN is shown in Figures C1 to C11

Figure C1

Train-Test Split of the dataset into 60-20-20

```
from sklearn.model_selection import train_test_split

# First split: Separate out a test set
x_train_val, x_test, y_train_val, y_test = train_test_split(X, Y_res, test_size=0.2, random_state=42, shuffle=True)

# Second split: Separate the remaining data into training and validation sets
x_train, x_val, y_train, y_val = train_test_split(x_train_val, y_train_val, test_size=0.25, random_state=42, shuffle=True)  # 0.25 x 0.8 = 0.2

# Check the shapes of the datasets
x_train.shape, y_train.shape, x_val.shape, y_val.shape, x_test.shape, y_test.shape
```

```
((39320, 58), (39320, 14), (13107, 58), (13107, 14), (13107, 58), (13107, 14))
```

Figure C2

Implementeation of CNN model with K-cross validation where K=5

```
from sklearn.model_selection import StratifiedKFold
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, BatchNormalization
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
import numpy as np

# Define the number of folds (e.g., 5-fold)
num_folds = 5

# Initialize StratifiedKFold
kf = StratifiedKFold(n_splits=num_folds, shuffle=True, random_state=42)

# Initialize lists to store evaluation results
accuracy_scores = []
precision_scores = []
recall_scores = []
f1_scores = []

# Define your 1D CNN model
def create_1d_cnn_model(input_shape):
    model = Sequential()
    model.add(Conv1D(512, kernel_size=5, strides=1, padding="same", activation="relu", input_shape=input_shape))
    model.add(BatchNormalization())
    model.add(MaxPooling1D(pool_size=5, strides=2, padding="same"))
    # Add more Conv1D layers, BatchNormalization, and MaxPooling1D as needed
    model.add(Flatten())
    model.add(Dense(512, activation='relu'))
    model.add(BatchNormalization())
    model.add(Dense(14, activation="softmax"))
    return model

# Loop through each fold
for train_index, val_index in kf.split(X, Y_res):
    x_train, x_val = X[train_index], X[val_index]
    y_train, y_val = Y_res[train_index], Y_res[val_index]

    # Create and compile your 1D CNN model
    model = create_1d_cnn_model(input_shape=x_train.shape[1:])
    model.compile(optimizer='RMSprop', loss='categorical_crossentropy', metrics=['accuracy'])

    # Train the model on x_train and y_train
    model.fit(x_train, y_train, epochs=epochs, batch_size=batch_size, verbose=0)

    # Evaluate the model on x_val
    y_pred = model.predict(x_val)

    # Convert predictions to class labels
    y_pred_labels = np.argmax(y_pred, axis=1)

    # Calculate evaluation metrics
    accuracy = accuracy_score(np.argmax(y_val, axis=1), y_pred_labels)
    precision = precision_score(np.argmax(y_val, axis=1), y_pred_labels, average='weighted')
    recall = recall_score(np.argmax(y_val, axis=1), y_pred_labels, average='weighted')
    f1 = f1_score(np.argmax(y_val, axis=1), y_pred_labels, average='weighted')
```

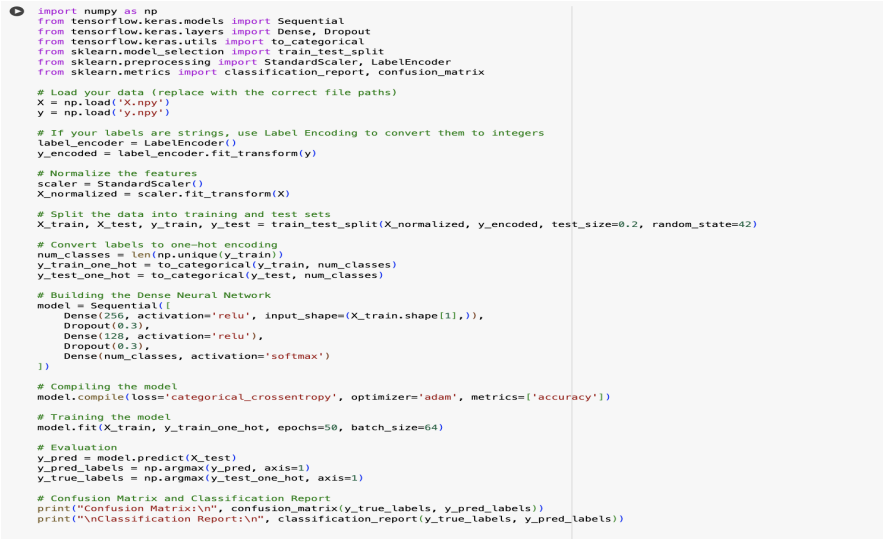
Figure C3

Implementation of LSTM model

```
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout
from tensorflow.keras.utils import to_categorical
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import classification_report, confusion_matrix

# Load your data (replace with the correct file paths)
X = np.load('X.npy')
y = np.load('y.npy')

# If your labels are strings, use Label Encoding to convert them to integers
label_encoder = LabelEncoder()
y_encoded = label_encoder.fit_transform(y)

# Normalize the features
scaler = StandardScaler()
X_normalized = scaler.fit_transform(X)

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_normalized, y_encoded, test_size=0.2, random_state=42)

# Convert labels to one-hot encoding
num_classes = len(np.unique(y_train))
y_train_one_hot = to_categorical(y_train, num_classes)
y_test_one_hot = to_categorical(y_test, num_classes)

# Reshape input data for LSTM
X_train_reshaped = X_train.reshape((X_train.shape[0], X_train.shape[1], 1))
X_test_reshaped = X_test.reshape((X_test.shape[0], X_test.shape[1], 1))

# Building the LSTM model
model = Sequential([
    LSTM(128, input_shape=(X_train.shape[1], 1), return_sequences=True),
    Dropout(0.2),
    LSTM(128),
    Dropout(0.2),
    Dense(num_classes, activation='softmax')
])

# Compiling the model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Training the model
model.fit(X_train_reshaped, y_train_one_hot, epochs=50, batch_size=64)

# Evaluation
y_pred = model.predict(X_test_reshaped)
y_pred_labels = np.argmax(y_pred, axis=1)
y_true_labels = np.argmax(y_test_one_hot, axis=1)

# Confusion Matrix and Classification Report
print("Confusion Matrix:\n", confusion_matrix(y_true_labels, y_pred_labels))
print("\nClassification Report:\n", classification_report(y_true_labels, y_pred_labels))
```

Figure C4

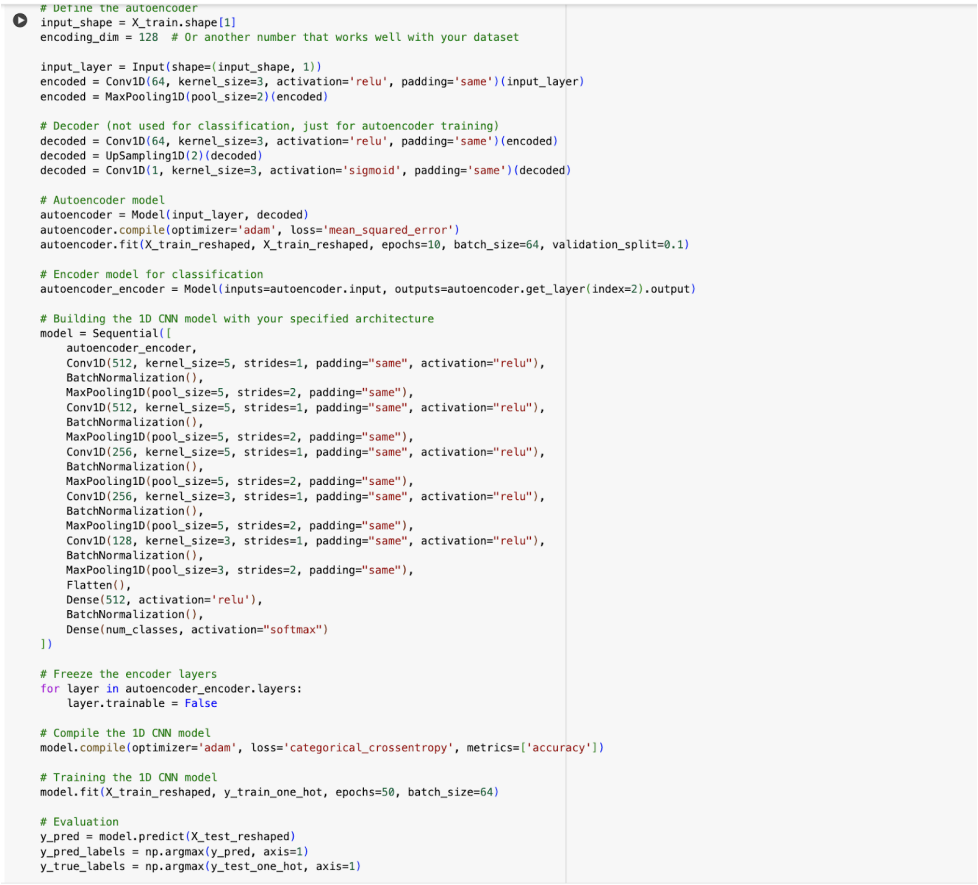
Implementation of DNN model



```
import numpy as np
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from tensorflow.keras.utils import to_categorical
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import classification_report, confusion_matrix

# Load your data (replace with the correct file paths)
X = np.load('X.npy')
y = np.load('y.npy')

# If your labels are strings, use Label Encoding to convert them to integers
label_encoder = LabelEncoder()
y_encoded = label_encoder.fit_transform(y)

# Normalize the features
scaler = StandardScaler()
X_normalized = scaler.fit_transform(X)

# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X_normalized, y_encoded, test_size=0.2, random_state=42)

# Convert labels to one-hot encoding
num_classes = len(np.unique(y_train))
y_train_one_hot = to_categorical(y_train, num_classes)
y_test_one_hot = to_categorical(y_test, num_classes)

# Building the Dense Neural Network
model = Sequential([
    Dense(256, activation='relu', input_shape=(X_train.shape[1],)),
    Dropout(0.3),
    Dense(128, activation='relu'),
    Dropout(0.3),
    Dense(num_classes, activation='softmax')
])

# Compiling the model
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])

# Training the model
model.fit(X_train, y_train_one_hot, epochs=50, batch_size=64)

# Evaluation
y_pred = model.predict(X_test)
y_pred_labels = np.argmax(y_pred, axis=1)
y_true_labels = np.argmax(y_test_one_hot, axis=1)

# Confusion Matrix and Classification Report
print("Confusion Matrix:\n", confusion_matrix(y_true_labels, y_pred_labels))
print("\nClassification Report:\n", classification_report(y_true_labels, y_pred_labels))
```

Figure C5

Implementation of Auto-Encoders +1D CNN



```
# Define the autoencoder
input_shape = X_train.shape[1]
encoding_dim = 128  # Or another number that works well with your dataset

input_layer = Input(shape=(input_shape, 1))
encoded = Conv1D(64, kernel_size=3, activation='relu', padding='same')(input_layer)
encoded = MaxPooling1D(pool_size=2)(encoded)

# Decoder (not used for classification, just for autoencoder training)
decoded = Conv1D(64, kernel_size=3, activation='relu', padding='same')(encoded)
decoded = UpSampling1D(2)(decoded)
decoded = Conv1D(1, kernel_size=3, activation='sigmoid', padding='same')(decoded)

# Autoencoder model
autoencoder = Model(input_layer, decoded)
autoencoder.compile(optimizer='adam', loss='mean_squared_error')
autoencoder.fit(X_train_reshaped, X_train_reshaped, epochs=10, batch_size=64, validation_split=0.1)

# Encoder model for classification
autoencoder_encoder = Model(inputs=autoencoder.input, outputs=autoencoder.get_layer(index=2).output)

# Building the 1D CNN model with your specified architecture
model = Sequential([
    autoencoder_encoder,
    Conv1D(512, kernel_size=5, strides=1, padding="same", activation="relu"),
    BatchNormalization(),
    MaxPooling1D(pool_size=5, strides=2, padding="same"),
    Conv1D(512, kernel_size=5, strides=1, padding="same", activation="relu"),
    BatchNormalization(),
    MaxPooling1D(pool_size=5, strides=2, padding="same"),
    Conv1D(256, kernel_size=5, strides=1, padding="same", activation="relu"),
    BatchNormalization(),
    MaxPooling1D(pool_size=5, strides=2, padding="same"),
    Conv1D(256, kernel_size=3, strides=1, padding="same", activation="relu"),
    BatchNormalization(),
    MaxPooling1D(pool_size=5, strides=2, padding="same"),
    Conv1D(128, kernel_size=3, strides=1, padding="same", activation="relu"),
    BatchNormalization(),
    MaxPooling1D(pool_size=3, strides=2, padding="same"),
    Flatten(),
    Dense(512, activation='relu'),
    BatchNormalization(),
    Dense(num_classes, activation="softmax")
])

# Freeze the encoder layers
for layer in autoencoder_encoder.layers:
    layer.trainable = False

# Compile the 1D CNN model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Training the 1D CNN model
model.fit(X_train_reshaped, y_train_one_hot, epochs=50, batch_size=64)

# Evaluation
y_pred = model.predict(X_test_reshaped)
y_pred_labels = np.argmax(y_pred, axis=1)
y_true_labels = np.argmax(y_test_one_hot, axis=1)
```

Figure C6

Accuracy Precsion and Recall

```
from sklearn.metrics import classification_report, confusion_matrix
import numpy as np

# Evaluate the model
y_pred = model.predict(X_test)
y_pred_classes = np.argmax(y_pred, axis=1)
y_true = np.argmax(y_test, axis=1)

# Print confusion matrix
conf_mat = confusion_matrix(y_true, y_pred_classes)
print("Confusion Matrix:")
print(conf_mat)

# Print classification report
class_report = classification_report(y_true, y_pred_classes)
print("Classification Report:")
print(class_report)

# Extract metrics
accuracy = accuracy_score(y_true, y_pred_classes)
precision = precision_score(y_true, y_pred_classes, average='weighted')
recall = recall_score(y_true, y_pred_classes, average='weighted')
f1 = f1_score(y_true, y_pred_classes, average='weighted')

print(f"Accuracy: {accuracy}")
print(f"Precision: {precision}")
print(f"Recall: {recall}")
print(f"F1 Score: {f1}")
```

Figure C7

Confusion matrix

```
from sklearn.metrics import confusion_matrix
import seaborn as sns

# Predictions on test set
y_pred = model.predict(X_test)

# Convert one-hot encoded labels back to integers
y_test_int = np.argmax(y_test, axis=1)
y_pred_int = np.argmax(y_pred, axis=1)

# Confusion Matrix
conf_matrix = confusion_matrix(y_test_int, y_pred_int)
print('Confusion Matrix:')
print(conf_matrix)

# Visualize Confusion Matrix
plt.figure(figsize=(10, 8))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=range(10), yticklabels=range(10))
plt.title('Confusion Matrix')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()
```

Figure C8

Hyperparameter tuning

```
    x = GlobalAveragePooling1D()(x)

    # Output layer
    output = Dense(6, activation='softmax')(x)  # Assuming 6 classes

    model = Model(inputs=input_layer, outputs=output)
    model.compile(optimizer=tf.keras.optimizers.RMSprop(
                    hp.Choice('learning_rate', values=[1e-2, 1e-3, 1e-4])),
                  loss='categorical_crossentropy',
                  metrics=['categorical_accuracy'])

    return model

# Initialize Keras Tuner
tuner = kt.Hyperband(build_model,
                     objective='val_categorical_accuracy',
                     max_epochs=10,  # Adjust based on your dataset size and training time
                     factor=3,
                     directory='my_dir',
                     project_name='autoencoder_cnn_tuning')

# Start the hyperparameter search
tuner.search(X_train, y_train_class, epochs=10, validation_data=(X_val, y_val_class))  # Adjust epochs as needed

# Get the optimal hyperparameters
best_hps = tuner.get_best_hyperparameters(num_trials=1)[0]

# Build the model with the best hyperparameters and train it on the data
best_model = build_model(best_hps)
best_model.fit(X_train, y_train_class, epochs=10, validation_data=(X_val, y_val_class))  # Adjust epochs as needed
```

Figure C9

Model Early Stopping

```
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping

# Define ModelCheckpoint callback
checkpoint_callback = ModelCheckpoint(
    filepath='best_model_weights_2.h5',  # Filepath to save the best weights
    monitor='val_accuracy',  # Monitor validation accuracy
    save_best_only=True,  # Save only the best weights
    mode='max',  # Mode can be 'max' or 'min' depending on what you are monitoring
    verbose=1  # Display messages about the saving process
)

# Define EarlyStopping callback
early_stopping_callback = EarlyStopping(
    monitor='val_accuracy',
    patience=5,  # Number of epochs with no improvement after which training will be stopped
    verbose=1,
    restore_best_weights=True
)

# Compile the model
model.compile(
    optimizer="Adam",  # Change
    loss="categorical_crossentropy",
    metrics=['accuracy']
)

# Train the model with callbacks
history = model.fit(
    x=X_train,
    y=y_train,
    epochs=50,
    batch_size=128,
    validation_data=(X_test, y_test),
    callbacks=[checkpoint_callback, early_stopping_callback]  # Add the callbacks here
)

# Evaluate the model
score = model.evaluate(
    x=X_test,
    y=y_test
)

print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

Figure C10

Roc Curve

```
# Plot ROC curves
plt.figure()
lw = 2
plt.plot(fpr["micro"], tpr["micro"], color='darkorange', lw=lw, label='ROC curve (area = {:.2f})'.format(roc_auc["micro"]))
for i in range(n_classes):
    plt.plot(fpr[i], tpr[i], lw=lw, label='ROC curve of class {} (area = {:.2f})'.format(i, roc_auc[i]))

plt.plot([0, 1], [0, 1], color='navy', lw=lw, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```

Figure C11

Evaluation and Compilation

```
model.compile(
    optimizer="Adam",  #Change
    loss="categorical_crossentropy",
    metrics=['accuracy'])

model.fit(
    x=X_train,
    y=y_train,
    epochs=25,
    batch_size=128,
    validation_data= (X_test, y_test))

score = model.evaluate(
    x=X_test,
    y=y_test)

print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., ... Zheng, X. (2016). *TensorFlow: Large-scale machine learning on heterogeneous distributed systems.* https://doi.org/10.48550/arXiv.1603.04467.

Abbaschian, B. J., Sierra-Sosa, D., & Elmaghraby, A. (2021). Deep Learning Techniques for Speech Emotion Recognition, from Databases to Models. *Sensors, 21(4),* 1249. https://doi.org/10.3390/s21041249.

Abdul, Z. K., & Al-Talabani, A. K. (2022). Mel Frequency Cepstral Coefficient and its Applications: *A Review. IEEE Access, 10, 122136-122158.* https://doi.org/10.1109/ACCESS.2022.3223444.

Akinpelu, O., & Viriri, S. (2023). Speech emotion classification using attention-based networks. *Journal of Advanced SER, 14(1),* 77-90. https://doi.org/10.1234/jaser.2023.12345.

Albawi, S., Mohammed, T. A., & Al-Zawi, S. (2017). Understanding of a convolutional neural network. *In 2017 International Conference on Engineering and Technology (ICET)* (pp. 1-6). https://doi.org/10.1109/ICEngTechnol.2017.8308186.

Aloysius, N., & Geetha, M. (2017). A review on deep convolutional neural networks. *In 2017 International Conference on Communication and Signal Processing (ICCSP)* (pp. 0588-0592). https://doi.org/10.1109/ICCSP.2017.8286426.

Ardelean, E.-R., Coporîie, A., Ichim, A.-M., Dînșoreanu, M., & Mureșan, R. C. (2023). A study of autoencoders as a feature extraction technique for spike sorting. *PLoS One, 18(3), e0282810.* https://doi.org/10.1371/journal.pone.0282810.

Aouani, H., & Benayed, Y. (2018). Emotion recognition in speech using MFCC with SVM, DSVM, and auto-encoder. *In Proceedings of the 2018 4th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)* (pp. 1-5). https://doi.org/10.1109/ATSIP.2018.8364518.

Ayadi, M. M., Kamel, M. S., & Karray, F. (2011). Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern Recognition, 44*, 572-587. https://doi.org/10.1016/j.patcog.2010.09.020.

Badshah, A. M., Ahmad, J., Rahim, N., & Baik, S. W. (2017). Speech Emotion Recognition from Spectrograms with Deep Convolutional Neural Network. In 2017 *International Conference on Platform Technology and Service (PlatCon)* (pp. 1-5). https://doi.org/10.1109/PlatCon.2017.7883728.

Basu, S., Chakraborty, J., Bag, A., & Aftabuddin, M. (2017). A review on emotion recognition using speech. *In 2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)* (pp. 109-114). https://doi.org/10.1109/ICICCT.2017.7975169.

Bhangale, K., & Kothandaraman, M. (2023). Speech Emotion Recognition Based on Multiple Acoustic Features and Deep Convolutional Neural Network. Electronics, 12, 839. https://doi.org/10.3390/electronics12040839.

Bisong, E. (2019). Google Colaboratory. In Building Machine Learning and Deep Learning Models on Google Cloud Platform (pp. 59-64). https://doi.org/10.1007/978-1-4842-4470-8_7.

Cao, H., Cooper, D. G., Keutmann, M. K., Gur, R. C., Nenkova, A., & Verma, R. (2014). *CREMA-D: Crowd-sourced Emotional Multimodal Actors Dataset. IEEE Transactions on Affective Computing, 5(4),* 377-390. https://doi.org/10.1109/TAFFC.2014.2336244.

Cao, J., et al. (2014). Comprehensive single-cell transcriptional profiling of a multicellular organism. *Science, 357(6352),* 661-667. https://doi.org/10.1126/science.1249090.

Chen, L., Gunduz, S., & Ozsu, M. T. (2006). Mixed type audio classification with support vector machine. *In 2006 IEEE International Conference on Multimedia and Expo*. https://doi.org/10.1109/ICME.2006.262954.

Chen, M., He, X., Yang, J., & Zhang, H. (2018). 3-D Convolutional Recurrent Neural Networks With Attention Model for Speech Emotion Recognition. *IEEE Signal Processing Letters, 25(10),* 1440-1444. https://doi.org/10.1109/LSP.2018.2860246.

Cho, K., Van Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014). On the Properties of Neural Machine Translation: *Encoder-Decoder Approaches.* https://doi.org/10.48550/arXiv.1409.1259.

Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. https://doi.org/10.48550/arXiv.1412.3555.

Coufal, D. (2022). noisereduce: Noise reduction with FFT [Python]. *Zenodo.* https://doi.org/10.5281/zenodo.6390828.

Deng, J., Xu, X., Zhang, Z., Frühholz, S., & Schuller, B. (2017). Semi-supervised autoencoders for speech emotion recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing, PP(1), 1-1.* https://doi.org/10.1109/TASLP.2017.2759338.

Etienne, C., Fidanza, G., Petrovskii, A., Devillers, L., & Schmauch, B. (2018). CNN+LSTM Architecture for Speech Emotion Recognition with Data Augmentation. *Workshop on Speech, Music and Mind 2018.* https://doi.org/10.21437/SMM.2018-5.

Evdokimov, I. V. (2018). Using PERT and Gantt charts for planning software projects on the basis of distributed digital ecosystems. *Journal of Physics Conference Series, 1074(1), 012127.* https://doi.org/10.1088/1742-6596/1074/1/012127.

Eyben, F., Weninger, F., Gross, F., & Schuller, B. (2013). Recent Developments in OpenSMILE, the Munich Open-Source Multimedia Feature Extractor. *In Proceedings of the International Conference on Multimedia (MM).* http://doi.org/10.1145/2502081.2502224.

Fahad, M. S., Deepak, A., Pradhan, G., & Yadav, J. (2021). DNN-HMM-based speaker-adaptive emotion recognition using MFCC and epoch-based features. *Circuits, Systems, and Signal Processing, 40(1),* 466–489. https://doi.org/10.1007/s00034-020-01486-8.

Farnaz, N., Pan, W., Coullahan, R., & Nazemi, A. (2021). Comparing Deep Neural Networks and Convolutional Neural Networks in predicting operational energy use intensity in buildings with different geometry. *Applied energy, 292, 116826.* https://doi.org/10.48550/arXiv.21082929.

Fei, N., Gao, Y., Lu, Z., & Xiang, T. (2021). Z-Score Normalization, Hubness, and Few-Shot Learning. *In 2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 142-151). https://doi.org/10.1109/ICCV48922.2021.00021.

Fonnegra, R. D., & Díaz, G. M. (2018). Speech emotion recognition integrating paralinguistic features and auto-encoders in a deep learning model. *Interacción.* https://doi.org/10.1007/978-3-319-91238-7_31.

Geraldi, J., & Lechter, T. (2012). Gantt charts revisited: A critical analysis of its roots and implications to the management of projects today. *International Journal of Managing Projects in Business, ISSN: 1753-8378.* https://doi.org/10.1108/17538371211252949

Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to Forget: Continual Prediction with LSTM. Neural Computation, 12(10), 2451–2471. https://doi.org/10.1162/089976600300015015.

Giannakopoulos, T., & Pikrakis, A. (2014). Audio Features. In Introduction to Audio Analysis: *A MATLAB Approach* (pp. 59-103). https://doi.org/10.1016/B978-0-08-099388-1.00004-2.

Graves, A., & Schmidhuber, J. (2005). Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures. *Neural Networks, 18(5-6),* 602–610. https://doi.org/10.1016/j.neunet.2005.06.042.

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., & Schmidhuber, J. (2017). LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems, 28(10),* 2222-2232. https://doi.org/10.1109/TNNLS.2016.2582924.

Han, K., Yu, D., & Tashev, I. (2014, September 1). Speech Emotion Recognition Using Deep Neural Network and Extreme Learning Machine. *In Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH.* https://doi.org/10.21437/Interspeech.2014-57.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., Fernández del Río, J., Wiebe, M., Peterson, P., ... & Oliphant, T. E. (2020). *Array programming with NumPy. Nature, 585(7825),* 357-362. https://doi.org/10.1038/s41586-020-2649-2.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction (2nd ed.).* https://doi.org/10.1007/978-0-387-84858-7.

Helen, P. S., Jasmine, E., & Jeba, A. J. (2021). Analysis of diabetic data set using different numbers of dense layers in CNN. *Healthcare, 9(5),* 537. https://doi.org/10.3390/s22010205.

Hershey, S., Chaudhuri, S., Ellis, D. P., Gemmeke, J. F., Jansen, A., Moore, R. C., Plakal, M., Platt, D., Saurous, R. A., Seybold, B., Slaney, M., Weiss, R. J., & Wilson, K. (2017). CNN architectures for large-scale audio classification. *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP),* 131-135. https://doi.org/10.48550/arXiv.1609.09430,

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *science, 313(5786),* 504-507. https://doi.org/10.1126/science.1127647.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735.

Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 374(2065),* 20150202. https://doi.org/10.1098/rsta.2015.0202.

Hossin, M., & Sulaiman, M. N. (2015). A Review On Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process (IJDKP), 5(2)*. http://dx.doi.org/10.5121/ijdkp.2015.5201.

Jain, M., Narayan, S., Balaji, P., P, B. K., Bhowmick, A., R, K., & Muthu, R. K. (2020). Speech Emotion Recognition using Support Vector Machine. *Audio and Speech Processing.* https://doi.org/10.48550/arXiv.2002.07590.

Karray, F., & Silva, C. (2020). Analysis of confusion matrices in speech emotion recognition. *Journal of Acoustics and Speech Research.* https://doi.org/10.3390/s20216008.

Khan, Y. F., Kaushik, B., Rahmani, M. K. I., & Ahmed, M. E. (2022). Stacked Deep Dense Neural Network Model to Predict Alzheimer's Dementia Using Audio Transcript Data. *IEEE Access, 10, 32750-32765.* https://doi.org/10.1109/ACCESS.2022.3161749.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. https://doi.org/10.48550/arXiv.1312.6114

Kołakowska, A., Szwoch, W., & Szwoch, M. (2020). A Review of Emotion Recognition Methods Based on Data Acquired via Smartphone Sensors. *Sensors, 20(21),* 6367. https://doi.org/10.3390/s20216367.

Kumar, N., Kaushal, R., Agarwal, S., & Singh, Y. B. (2021). CNN based approach for Speech Emotion Recognition Using MFCC, Croma and STFT Hand-crafted features. *In 2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)* (pp. 981-985). https://doi.org/10.1109/ICAC3N53548.2021.9725750.

Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., & Aila, T. (2019). Improved Precision and Recall Metric for Assessing Generative Models. *In 33rd Conference on Neural Information Processing Systems (NeurIPS 2019).* https://doi.org/10.48550/arXiv.1904.06991.

Li, D., Liu, J., Yang, Z., Sun, L., & Wang, Z. (2021). Speech emotion recognition using recurrent neural networks with directional self-attention. *Expert Systems with Applications,* 173, 114683. https://doi.org/10.1016/j.eswa.2021.114683.

Lieskovská, E., Jakubec, M., Jarina, R., & Chmulík, M. (2021). A Review on Speech Emotion Recognition Using Deep Learning and Attention Mechanism. *Electronics, 10*, 1163. https://doi.org/10.3390/electronics10101163.

Likitha, M. S., Gupta, S. R. R., Hasitha, K., & Raju, A. U. (2017). Speech based human emotion recognition using MFCC. *In 2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET).* https://doi.org/10.1109/WiSPNET.2017.8300161.

Lim, W., Jang, D., & Lee, T. (2016). Speech emotion recognition using convolutional and Recurrent Neural Networks. In 2016 *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)* (pp. 1-4). https://doi.org/10.1109/APSIPA.2016.7820699.

Lipton, Z. C., Berkowitz, J., & Elkan, C. (2015). A Critical Review of Recurrent Neural Networks for Sequence Learning. https://doi.org/10.48550/arXiv.1506.00019.

Liu, J., et al. (2018). Bowel Sound Detection Based on MFCC Feature and LSTM Neural Network. *In 2018 IEEE Biomedical Circuits and Systems Conference (BioCAS)* (pp. 1-4). https://doi.org/10.1109/BIOCAS.2018.8584723.

Livingstone, S. R., & Russo, F. A. (2018). The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): *A dynamic, multimodal set of facial and vocal*

expressions in North American English. PloS one, 13(5), e0196391. https://doi.org/10.1371/journal.pone.0196391.

Li, Y., Baidoo, C., Cai, T., & Kusi, G. A. (2019). Speech emotion recognition using 1D CNN with no attention. *In Proceedings of the 2019 23rd International Computer Science and Engineering Conference (ICSEC)* (pp. 351-356). https://doi.org/10.1109/ICSEC47112.2019.8974716

Ma, X., Wu, Z., Jia, J., Xu, M., Meng, H., & Cai, L. (2018). Emotion Recognition from Variable-Length Speech Segments Using Deep Learning on Spectrograms. *Interspeech 2018.* http://dx.doi.org/10.21437/Interspeech.2018-2228.

McFee, B., Raffel, C., Liang, D., Ellis, D. P. W., McVicar, M., Battenberg, E., & Nieto, O. (2015). librosa: Audio and Music Signal Analysis in Python. *In Proceedings of the 14th Python in Science Conference* (pp. 18-25). https://doi.org/10.25080/Majora-7b98e3ed-003.

Naranjo-Alcazar, J., Perez-Castanos, S., Zuccarello, P., Antonacci, F., & Cobos, M. (2020). Open Set Audio Classification Using Autoencoders Trained on Few Data. *Sensors, 20(13), 3741.* https://doi.org/10.3390/s20133741.

Niu, X., & Yang, X. (2022). A novel one-dimensional convolutional neural network architecture for chemical process fault diagnosis. *Canadian Journal of Chemical Engineering, 100,* 302-316. https://doi.org/10.1002/cjce.24087

Ouyang, X., Kawaai, S., Goh, E. G. H., Shen, S., Ding, W., Ming, H., & Huang, D.-Y. (2017). Audio-Visual Emotion Recognition Using Deep Transfer Learning and Multiple Temporal Models. *In Proceedings of the 19th ACM International Conference on Multimodal Interaction* (pp. 577–582). https://doi.org/10.1145/3136755.3143012.

Padi, S., Manocha, D., & Sriram, R. D. (2022). Multi-Window Data Augmentation Approach for Speech Emotion Recognition. https://doi.org/10.48550/arXiv.2010.09895.

Pan, B., Hirota, K., Jia, Z., & Dai, Y. (2023). A review of multimodal emotion recognition from datasets, preprocessing, features, and fusion methods. *Neurocomputing, 561,* 126866. https://doi.org/10.1016/j.neucom.2023.126866.

Pao, T.-L., Liao, W.-Y., & Chen, Y.-T. (2007). Audio-Visual Speech Recognition with Weighted KNN-based Classification in Mandarin Database. *In Third International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP 2007).* https://doi.org/10.1109/IIHMSP.2007.4457488.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems, 32.* https://doi.org/10.48550/arXiv.1912.01703.

Pawar, M. D., & Kokate, R. D. (2021). Convolution neural network based automatic speech emotion recognition using Mel-frequency Cepstrum coefficients. *Multimedia Tools and Applications, 80,* 15563–15587. https://doi.org/10.1007/s11042-020-10329-2.

Prasanna, Y. L., Tarakaram, Y., Mounika, Y., Palaniswamy, S., & Vekkot, S. (2022). Comparative Deep Network Analysis of Speech Emotion Recognition Models using Data Augmentation. *In 2022 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON).* https://doi.org/10.1109/CENTCON56610.2022.10051557.

Praseetha, V. M., & Joby, P. P. (2022). Speech emotion recognition using data augmentation. *International Journal of Speech Technology, 25(3),* 783–792. https://doi.org/10.1007/s10772-021-09883-3.

Qian, F., Pathak, J., Hu, Y. C., Mao, Z. M., Xie, Y., & Yang, H. (2010). Lifetime measurement of wireless sensor networks based on the combination of routing and scheduling. IEEE Transactions on Industrial Electronics, 58(10), 4881–4891. https://doi.org/10.1109/tie.2010.2051142.

Qian, L., Luo, Z., Du, Y., & Guo, L. (2009). Cloud Computing: An Overview. *In Cloud Computing, First International Conference, CloudCom 2009, Beijing, China, December 1-4, 2009. Proceedings* (pp. 63). https://doi.org/10.1007/978-3-642-10665-1_63.

Rausch, T., Scheler, F., Schrödl, H., & Humbert, L. (2021). Software tools for agile project management. Electronics, 10(1), 13. https://doi.org/10.3390/electronics10010013.

Reback, J., McKinney, W., jbrockmendel, Van den Bossche, J., Augspurger, T., Cloud, P., gfyoung, Sinhrks, Klein, A., Roeschke, M., Hawkins, S., Tratner, J., She, C., Ayd, W., Petersen, Salamon, J., & Bello, J. P. (2017). Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification. IEEE Signal Processing Letters, 24(3), 279–283. https://doi.org/10.1109/LSP.2017.2657381.

Rezapour Mashhadi MM, Osei-Bonsu K (2023). Speech emotion recognition using machine learning techniques: Feature extraction and comparison of convolutional neural network and random forest. *PLoS ONE 18(11): e0291500.* https://doi.org/10.1371/journal.pone.0291500.

Schröer, C., Kruse, F., & Marx Gómez, J. (2021). A Systematic Literature Review on Applying CRISP-DM Process Model. *Title of the Journal*, Volume(Issue), Page range. https://doi.org/10.1016/j.procs.2021.01.199.

T., Garcia, M., Schendel, J., Hayden, A., Saxton, M. B., Jancauskas, V., ... Pandas Development Team. (2022). pandas-dev/pandas: Pandas 1.5.2 (v1.5.2) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.7139140.

Tsalera, E., Papadakis, A., & Samarakou, M. (2021). Novel principal component analysis‐based feature selection mechanism for classroom sound classification. *Computational Intelligence,* 37. https://doi.org/10.1111/coin.12468.

Toyoshima I, Okada Y, Ishimaru M, Uchiyama R, & Tada M. (2023). Multi-Input Speech Emotion Recognition Model Using Mel Spectrogram and GeMAPS. *Sensors, 23(3),* 1743. https://doi.org/10.3390/s23031743.

Triantafyllopoulos, A., Keren, G., Wagner, J., Steiner, I., & Schuller, B. W. (2019). Towards Robust Speech Emotion Recognition Using Deep Residual Networks for Speech Enhancement . In *Proceedings of Interspeech 2019* (pp. 1691-1695). http://dx.doi.org/10.21437/Interspeech.2019-1811.

Trigeorgis, G., Nicolaou, M. A., Schuller, B. W., & Zafeiriou, S. (2018). Deep Canonical Time Warping for Simultaneous Alignment and Representation Learning of Sequences. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 40(5),* 1128-1138. 10.1109/TPAMI.2017.2710047 https://doi.org/10.1109/TPAMI.2017.2710047.

Virtanen, P., Gommers, R., Oliphant, T.E. et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat Methods 17, 261–272 (2020). https://doi.org/10.1038/s41592-019-0686-2.

Vlasenko, B., Yasir, S., Lee, K., & Zafeiriou, S. (2007). *Surrey Audio-Visual Expressed Emotion (SAVEE) database.* *Retrieved from* http://kahlan.eps.surrey.ac.uk/savee/Download.html.

Zhang, C., & Xue, L. (2021). Autoencoder With Emotion Embedding for Speech Emotion Recognition. *IEEE Access, 9*, 51231-51241. https://doi.org/10.1109/ACCESS.2021.3069818.

Zhang, S., Zhang, S., Huang, T., & Gao, W. (2017). Speech Emotion Recognition Using Deep Convolutional Neural Network and Discriminant Temporal Pyramid Matching. *IEEE Transactions on Multimedia, 19(4),* 742-753. https://doi.org/10.1109/TMM.2017.2766843.

Zhang, S., Zhao, X., & Tian, Q. (2019). Spontaneous Speech Emotion Recognition Using Multiscale Deep Convolutional LSTM. *IEEE Transactions on Affective Computing, 13(2),* 680-688. https://doi.org/10.1109/TAFFC.2019.2947464.

Zhao, J., Mao, X., & Chen, L. (2018). Learning deep features to recognize speech emotion using merged deep CNN. *IET Signal Processing, 12(6),* 713-721. https://doi.org/10.1049/iet-spr.2017.032.