PCD Reference: PCD/2425/49

University of Manouba
National School of Computer Science

ID:211 Version:01

Report of the Design Project and Development

Context-Aware Speech Recognition for Smart Assistance and Multilingual Interaction

Elaborated by :

Karim Lazghab **Oussama Bouhali** **Omar Jabri**

Elaborated by :

Dr. Marouene CHAIEB

Academic Year : 2024-2025

Appreciating my students' valuable contributions in this project. Wishing them continued success and good luck.

Dr. Marouene CHAIEB

Acknowledgements

We extend our sincere thanks and appreciation to Dr. Marouene CHAIEB, our supervisor, for his good directives, his availability and for his precious advice that he gave us in order to accomplish our project properly.

We extend our utmost respect and gratitude to the members of the jury for the honor they have done us in accepting to examine and evaluate this modest contribution, and to the administration and all the teaching staff of the ENSI for the training they have provided us during these two years.

We also express our gratitude to everyone who has contributed in one way or another to the good realization of the project.

We would also like to thank the CCK for granting us access to the machine used in this work, which was essential to the successful completion of our project.

Abstract

As part of our Design and Development project, we aimed to improve Automatic Speech Recognition (ASR) systems for noisy environments. Acknowledging the limitations of existing ASR systems under adverse acoustic conditions, we applied advanced deep learning techniques and data augmentation strategies to enhance transcription accuracy. Our approach was to develop an application that integrates models to automatically transcribe speech and translate it into a selected target language, even under noisy environmental conditions.

Following our initial research and experimentation, we identified the CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology as the most suitable framework for addressing this problem.

In the data collection phase, we utilized diverse speech datasets, such as LibriSpeech and TIMIT. During the preprocessing phase, we applied signal processing techniques and noise augmentation methods, including Gaussian noise injection, to expand the training dataset and better simulate real-world acoustic conditions. This enhanced the models' ability to generalize and increased their resilience to environmental distortions.

In the model building phase, we focused on training various deep learning architectures, including BiLSTM, Transformer, RNN, and GRU models, combined with residual CNNs to enhance performance. Only the Whisper model was fine-tuned. After evaluating the models in the evaluation phase, we integrated the best-performing models into an interactive interface, allowing users to input voice recordings or MP3 files, which are then transcribed and translated accurately into the selected language.

To validate the effectiveness of our solution, we conducted experiments using multiple datasets, and the results showed that LibriSpeech yielded the best performance achieving an accuracy of 99.11% and outperforming several methods presented in the literature. The full implementation is publicly accessible via the following GitHub repository: https://github.com/DrMaroueneCHAIEB/Context-Aware-Speech-Recognition-for-Smart-Assistance-and-Multilingual-Interaction

Key Terms

Automatic Speech Recognition, Noise Robustness, Gated recurrent units, Gaussian noise injection.

Contents

List of acronyms

AI Artificial Intelligence

ML Machine Learning

API Application Programming Interface

UML Unified Modeling Language

MVC Model View Controller

SVM Support Vector Machine

ACC Accuracy

PR Precision-Metric

KNN K-Nearest Neighbours

RF Random Forests

ANN Artificial Neural Network

ASR Automatic Speech Recognition

MFCC Mel-frequency Cepstral Coefficients

FDLP Frequency Domain Linear Prediction

BiLSTM Bidirectional Long Short-Term Memory

EER Equal Error Rate

WER Word Error Rate

GMM Gaussian Mixture Models

HMM Hidden Markov Models

LSTM Long Short-Term Memory

FPR False Positive Rate

FNR False Negative Rate

IIR Infinite Impulse Response

CER Character Error Rate

TSM Time-Scale Modification

DNN Deep Neural Network

LAS Listen, Attend, and Spell

SCT Single-Condition Training

MCT Multicondition Training

SMOTE Synthetic Minority Oversampling Technique

ECAPA Emphasized Channel Attention, Propagation, and Aggregation Time

CD Clean Data

ND Noisy Data

UN Under Noisy

GNI Gaussian Noise Injection

CRISP-DM Cross Industry Standard Process for Data Mining

List of Figures

List of Tables

Chapter 1

Introduction

The following sections provide a comprehensive overview of the study, beginning with the broader context and motivation behind the research, focusing on the use of automatic speech recognition, particularly in noisy environments, to enhance vocal input for accurate transcription. It then addresses the specific problem being tackled, the objectives pursued, and the proposed solution, concluding with a detailed structure of the report.

1.1 Context of the Study

Deep learning and machine learning have gained significant traction in recent years, achieving remarkable breakthroughs in areas such as natural language processing and speech recognition. While these technologies have been widely adopted across various industries, challenges persist in applying them to real-world problems like detecting voice in noisy environments. Such scenarios, often encountered in public spaces or industrial settings, pose significant obstacles to effective communication and automated voice-based systems. Artificial intelligence simplifies the development of robust tools capable of isolating voice signals from background noise, ensuring higher precision and reliability. These advancements pave the way for applications ranging from enhanced user experiences in virtual assistants to critical systems in safety and emergency communications.

1.2 Problem Statement and Motivation

For many years, traditional approach such as Mel-frequency Cepstral Coefficients (MFCC) has been the first choice for extracting features as the front end. However, the traditional feature extraction methods and classification models are not capable of handling real time noisy environments. In order

to solve this problem, researchers have used two or more feature extraction techniques at frontend and ensemble of machine learning models at backend of an ASR system. However, complexity of these systems is major concern, when these systems are used for training or inference. Also, the spoofing datasets available now a days are highly imbalanced which contains more samples of one class as compared to others.

To address the limitations of traditional ASR systems in noisy environments, this study explores the integration of advanced AI models such as Transformers and BiLSTMs. These models have shown significant promise in capturing long-term dependencies and contextual information in speech, thereby improving robustness and transcription accuracy. In addition, noise augmentation techniques such as adding Gaussian noise to input signals are employed during training to simulate real-world acoustic conditions. This approach enhances the model's generalization ability and prediction accuracy in challenging environments. By combining deep learning architectures with data augmentation strategies, the study aims to develop a more effective and efficient ASR system capable of handling speech in noisy conditions.

1.3 Aims and Objectives

In light of these considerations, the present study aims to develop a robust and efficient automatic speech recognition system with improved accuracy for processing speech in noisy environments, in order to ensure correct transcription. The following section outlines the main objectives pursued to achieve this goal.

Our primary aim in this study is to:

- Enhance the system's robustness to noise through data augmentation techniques, including Gaussian noise injection.
- Implement and compare deep learning models such as BiLSTM and Transformer architectures to evaluate their performance.
- Use different datasets to effectively train and evaluate the speech recognition system.
- Integrate the best-performing model for end-to-end transcription and translation of audio inputs, enabling audio-to-audio multilingual interaction.

1.4 Proposed Solution and Results

In our work, we proposed a structured methodology to develop an automatic speech recognition system adapted to noisy environments. The process began with the data collection phase, where we gathered multiple speech datasets to ensure diversity in accents, environments, and speaking styles. This provided a solid foundation for training and evaluating the models. In the data preprocessing phase, we applied several techniques such as normalization, silence trimming, and the injection of Gaussian noise to simulate real-world acoustic disturbances, thereby enhancing the system's robustness. In the modeling phase, we implemented and compared deep learning architectures such as BiLSTM and Transformer, selected for their ability to capture complex temporal and contextual dependencies in speech sequences. During the evaluation phase, we assessed model performance using standard metrics like WER and CER to measure transcription accuracy. Finally, an interactive interface was designed to display transcriptions and enable multilingual interaction

1.5 Structure of the Report

This report is structured into various chapters to enhance understanding of the study's purpose, background, and findings. Chapter 2 presents a literature review, examining previous research and approaches related to voice detection in noisy environments. Chapter 3 focuses on the analysis and specification of requirements, outlining the problem's scope and key functional and technical needs. Chapter 4 introduces the conceptual design, including both the logical and physical architecture of the system. Chapter 5 details the methodology and proposed solution, accompanied by a graphical abstract that illustrates the different phases, from data collection to model building and evaluation metrics. Chapter 6 covers the implementation phase, describing the tools, technologies, and methods used during development, followed by the presentation of results, performance evaluation, and the graphical user interface. Finally, Chapter 7 summarizes the key findings and proposes future directions for research.

Chapter 2

Literature Review

In this section, we begin by outlining the technical background necessary to understand the context of our study on ASR in noisy environments. Next, we review published research in ASR, focusing on methodologies such as deep learning, data augmentation, and signal processing. Finally, we discuss the insights and limitations identified in the literature and highlight the contributions of our research in addressing these challenges, particularly in improving robustness and accuracy in challenging auditory conditions.

2.1 Technical Background

In this section, we provide an overview of the key components and methodologies for developing and improving ASR systems, focusing on DNN-based frameworks, adversarial challenges, and detection mechanisms.

- **DNN-Based ASR Systems:**

  ASR typically consists of Feature Extraction, Model Inference, and Decoding, as depicted in Figure 2.1 [1]:

  - **Feature Extraction**:

    The process of converting input waveforms into features suitable for Deep Neural Networks (DNNs) is a crucial step in ASR systems. This step ensures that raw audio data is transformed into a format that DNNs can efficiently analyze and interpret. Various techniques are employed to achieve this transformation, each contributing unique advantages to the feature extraction process:

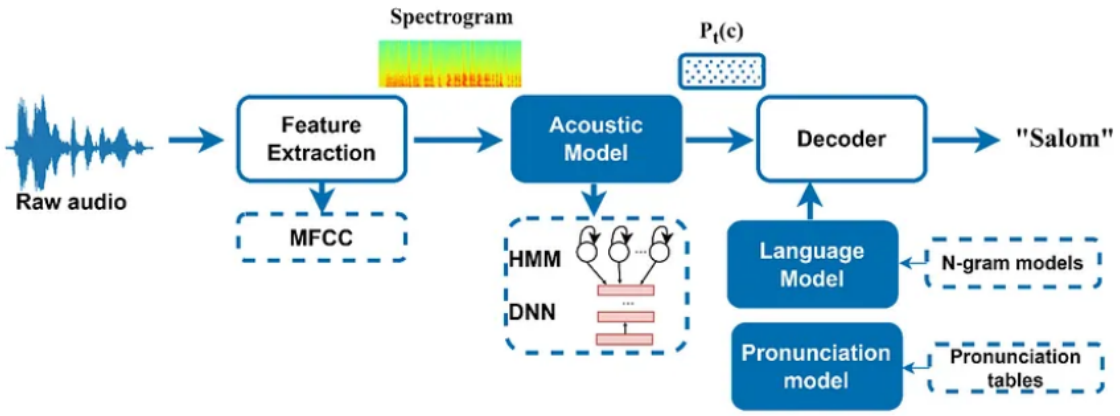

Figure 2.1: Overview of ASR components: feature extraction, model inference, and decoding

1. **Mel-Frequency Cepstral Coefficients (MFCC):** Captures human auditory perception by converting waveforms into a compact representation of frequency components using FFT, mel-filters, and DCT.
2. **Mel-Spectrograms:** Represents audio signals in a time-frequency domain using the mel scale to highlight energy distribution across frequency bands over time.
3. **Power Spectrum:** Analyzes the energy of different frequency components in the signal, identifying dominant frequencies and their strengths.
4. **Machine Learning (ML) Techniques:** Automates and enhances feature extraction by identifying relevant patterns and structures in the audio data.

– **Model Inference**: Processes extracted features through a pre-trained DNN acoustic model, often using Recurrent Neural Networks (RNNs).

– **Decoding**: Converts logits into character strings using N-gram language models. These strings are further interpreted into meaningful words, completing the transcription process.

- ## Adversarial Challenges in ASR Systems

Despite their advancements, ASR systems face threats from adversarial examples—maliciously crafted audio signals designed to deceive the model. These challenges are categorized as follows:

– **Over-Line Attacks:** Attackers directly inject adversarial audio signals into the system. These attacks retain fidelity since they bypass transmission distortions.

– **Over-Air Attacks:** These attacks involve overcoming ambient noise, multi-path effects, and device limitations.

Figure 2.2 [2] depicts an example of an over-the-air attack.

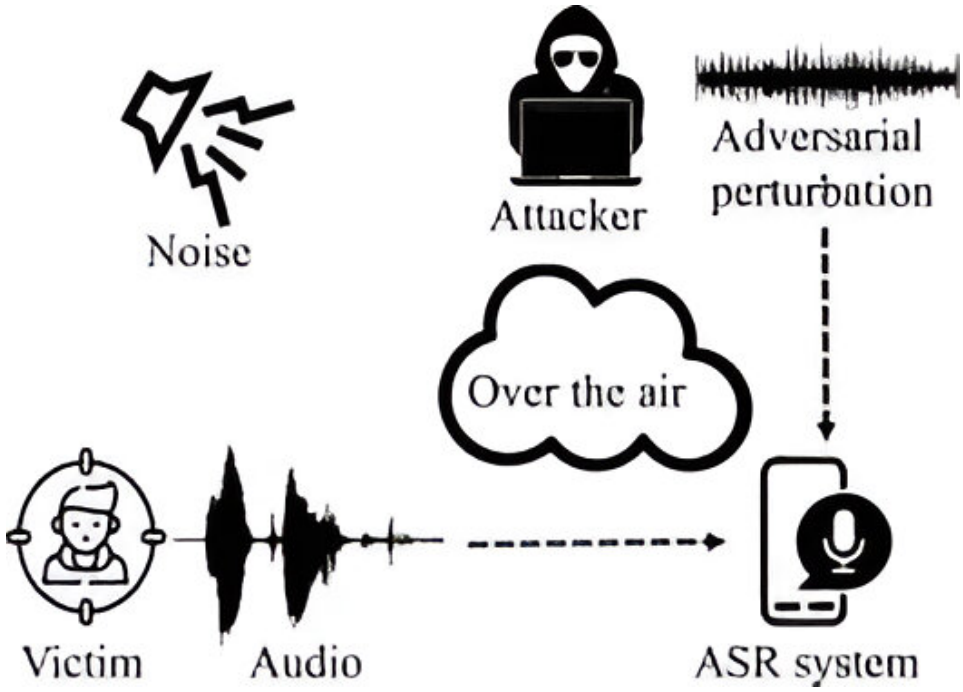

Figure 2.2: Adversarial challenges in ASR systems

- **Defense Mechanisms**

Researchers have developed various techniques to detect and mitigate adversarial attacks:

– **Noise Addition**: Adds controlled noise to audio inputs, reducing adversarial impact.

– **Random Filtering**: Applies random high-pass, low-pass, or notch filters to disrupt adversarial perturbations.

– **Temporal Dependency Analysis**: Exploits inconsistencies in transcription over time by comparing partial and complete transcriptions of the same audio input.

– **Reverberation Techniques**: Uses Room Impulse Response convolution to neutralize overfitting adversarial effects and adds Gaussian noise to silent portions of the audio for enhanced detection.

- **Audio Distortion Degree Metric**

  To quantify adversarial impacts on audio signals, the audio distortion degree metric measures distortion in decibels using the formula:

$$\mathbf{dB_x}(\delta) = \max_i [20 \cdot (\log_{10}(\delta_i) - \log_{10}(x_i))]$$

  - $\mathrm{dB}_x(\delta)$: Maximum deviation in decibels between the vector δ (estimated or perturbed values of a signal) and the reference vector x (original signal values). It measures how much δ differs from x in terms of signal amplitude.
  - δ_i: The i-th component of the estimated or perturbed signal vector δ, representing the modified value of the signal at index i.
  - x_i: The i-th component of the original (reference) signal vector x, representing the true or baseline value of the signal at index i.

  where is the original audio signal and represents the distortion. Lower values indicate reduced distortion and improved robustness.

- **Advancements in ASR Defense:**

  Figure 2.3 [3] illustrates the advancements in ASR defense, which consist of two components:

  - **Logit Analysis**: Studies the distribution of logits to distinguish benign audio from adversarial examples.
  - **Adaptive Defenses**: Designs robust methods resilient against attackers aware of the defense mechanisms.

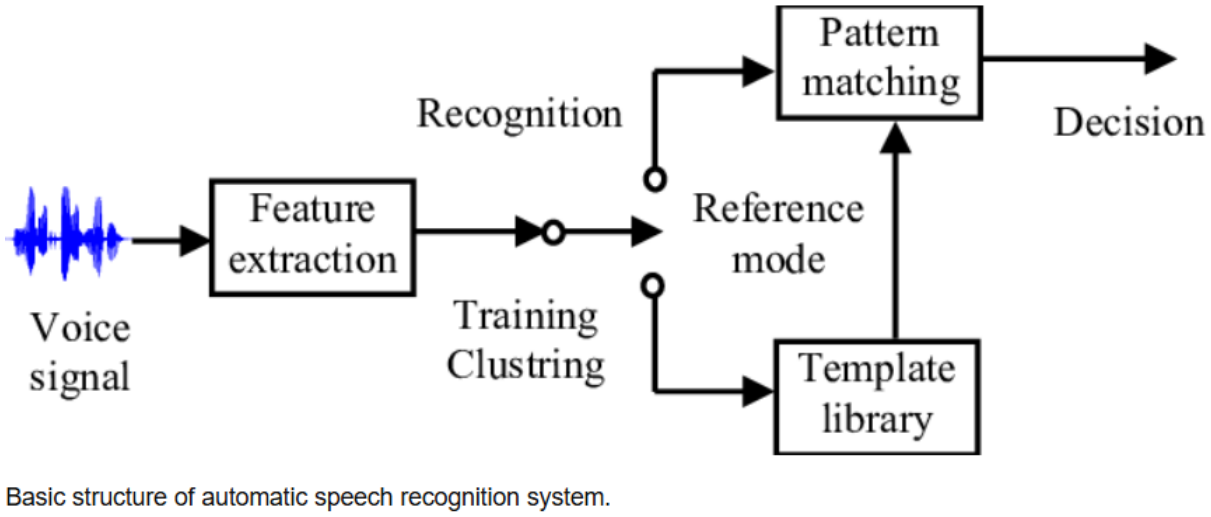

Figure 2.3: Advancements in ASR defense

2.2 Related Works

For many years, MFCC and GMM-based acoustic models were the predominant choice among researchers for the development of ASR systems. However, contemporary efforts have shifted toward enhanced feature extraction methods, deep learning architectures, and the utilization of more sophisticated datasets. For instance, kathania et al. [4] studied the challenges faced by ASR systems in recognizing children's speech under noisy conditions. These challenges are primarily attributed to the mismatch between training data typically clean adult speech and testing data, which often comprises noisy children's speech, along with prosodic differences such as pitch and speaking rate. The researchers employed data augmentation and time-scale modification techniques, and their experiments using publicly available datasets demonstrated a reduction in WER to 38.36% under 0 dB babble noise conditions.

Similarly, Mohit et al.[5] studied the challenges faced by ASR systems in distinguishing between legitimate (bonafide) and spoofed audio, especially in noisy environments. The performance of these systems is often compromised due to the complexity of existing models and the imbalance in the datasets. To address this, they attempted to reduce complexity through the use of FDLP, Data Augmentation, and Bidirectional Long Short-Term Memory (BiLSTM) networks at the backend, achieving an accuracy of 99.5% and an Equal Error Rate (EER) below 9.1%.

Nada et al. [6] discussed the significant difficulties ASR systems encounter when recognizing disordered speech, particularly for individuals with motor speech disorders (MSDs) such as dysarthria.

These challenges arise from data limitations, high variability, and inadequate evaluation metrics. They proposed improving ASR performance by adapting models to specific speakers or disorder types using techniques such as Maximum a Posteriori, Maximum Likelihood Linear Regression, and Neural Network Fine-Tuning.

Chang et al. [7] investigated the limitations of conventional speaker verification systems, which train front-end encoders and back-end models separately, leading to suboptimal performance, particularly when dealing with multiple enrollment utterances. They proposed a fully end-to-end (E2E) ASV model that jointly trains both the encoder and the back-end. This model integrates attention mechanisms (both frame-level and utterance-level) to capture intra-utterance relationships, significantly improving speaker verification, especially with multiple enrollments, by optimizing both stages simultaneously.

Sanghun et al. [8] examined the poor performance of traditional audio-only speech recognition systems in real-world noisy environments, particularly in interactive applications like virtual aquariums or edutainment platforms, where background noise can disrupt accurate speech recognition. They proposed a multi-modal audio-visual speech recognition system that combines audio and visual inputs (lip-reading) to enhance accuracy and robustness. By employing feature extraction with 3D CNNs and data augmentation, they managed to improve accuracy from 91.42% (audio-only) to 98.12% (combined audio-visual).

Mario et al. [9] proposed a Multicondition Training (MCT) framework aimed at improving noise robustness, combining feature extraction, noise addition, and Support Vector Machines (SVMs) with a linear kernel. Their study focused on the repeated pronunciation of the sequences /pa-ta-ka/, and they evaluated their solution using accuracy and AUC-ROC. Their best accuracy results were 82.18% (/pa/), 83.78% (/ta/), and 89.93% (/ka/).

Ayesha et al. [10] introduced a noise-robust speech recognition system by incorporating noise augmentation into the training data. Both Gaussian Mixture Models (GMMs) and deep learning models (CNNs, DNNs, LSTMs) were tested with clean and noisy datasets. Their experiments resulted in a 5.15% WER for clean data and 52.76% WER for noisy data.

Namgyu et al. [11] investigated ways to enhance ASR system robustness against adversarial audio attacks by combining IIR filtering for signal smoothing and Gaussian noise injection. This approach achieved an accuracy of 99.27% for over-air attacks and a 0.6% False Positive Rate (FPR) under optimal configurations.

Table 2.1 provides a summary of the performance of the solutions proposed.

Table 2.1: Synthesis of the research findings

Research	Dataset	Technique	Performances (%)						
			Acc	**EER**	**WER**	**F1**	**Prec.**	**Rec.**	**AUC-ROC**
Mohit et al. [5]	VSDC	FDLP+SMOTE+BiLSTM	99.5	55.0	-	99.4	99.3	99.5	98.5
		FDLP+SMOTE+BiLSTM(UN)	92.03	8.5	-	91.80	93.31	88.53	93.00
	ASVspoof(PA)	FDLP+SMOTE+BiLSTM	99.11	85.00	-	99.2	99.12	99.1	98.1
		FDLP+SMOTE+BiLSTM (UN)	91.28	8.7	-	91.27	89.33	91.22	9.12
	ASVspoof(LA)	FDLP+SMOTE+BiLSTM	98.8	91.0	-	99.2	99	98.6	97.00
		FDLP+SMOTE+BiLSTM(UN)	90.8	9.1	-	90.7	90.8	90.8	90.8
Nada et al. [6]	isca-archive	GMM-HMM	-	98.5	-	-	-	-	-
		E2E	-	98.8	-	-	-	-	-
Sunghun et al. [8]	Google Speech	3D CNNs & Bi-GRUs	98.12	-	-	98.69	98.70	98.69	-
		CTC Loss/MM Fusion/DA							
Chang et al. [7]	CNCeleb	E2E	-	8.04	-	-	-	-	
Ayesha et al. [10]	Speech Command	tri3b fMMI + MMI(CD)	-	-	5.15	-	-	-	-
		tri3b fMMI + MMI(ND)	-	-	52.76	-	-	-	-
		DNN(CD)	-	-	2.43	-	-	-	-
		DNN(ND)	-	-	38.63	-	-	-	-
		CNN(CD)	-	-	3.59	-	-	-	-
Mario et al. [9]	(not public)	MCT/pa/	82.18	-	-	-	-	-	81.7
		MCT/ta/	83.78	-	-	-	-	-	81.1
		MCT/ka/	89.93	-	-	-	-	-	89.1
Namgyu et al. [11]	librispeech	GNI-IIR	99.27	-	-	-	-	-	-
Hemant et al. [4]	thespeechark	TSM	-	-	14.88	-	-	-	-

2.3 Findings of the Literature and Contributions of the Study

In this section, we present key insights from existing literature on ASR in noisy environments, followed by the specific contributions of our study that address notable gaps.

- **Findings of the Literature:** The review of related work reveals several important observations that have shaped the development of noise-resilient ASR systems:
  - **Impact of Noise-Robust Solutions in ASR Systems:** Existing studies show that background noise significantly disrupts speech recognition performance by masking essential acoustic features. Traditional ASR systems are especially sensitive to variations in environmental noise. To counter this, researchers have applied data augmentation methods like SpecAugment, noise injection, and pitch shifting to enhance training diversity and reduce WER. Moreover, techniques such as noise-aware training, denoising autoencoders, and adversarial learning have improved system robustness. Domain adaptation strategies have also been shown to reduce mismatch between training and real-world deployment conditions.
  - **Advancements in Deep Learning Models:** Deep learning architectures including CNNs, RNNs, LSTMs, and more recently, Transformers have demonstrated clear advantages over classical models due to their ability to learn intricate temporal and spatial features. Transformer based models, in particular, offer parallel processing and stronger representation learning. Additionally, self-supervised models like Wav2Vec leverage large-scale pretraining to generalize better, especially in noisy settings.
  - **Evaluation and Metrics:** WER continues to be the most widely used metric for assessing ASR performance. However, other indicators like Signal-to-Noise Ratio (SNR) and Perceptual Evaluation of Speech Quality (PESQ) are also employed to provide deeper insights into model robustness and signal clarity.
- **Contributions of the Study:** Despite substantial progress, the literature often overlooks several aspects critical to improving ASR in noisy environments:
  - Many existing studies do not fully explore the potential of Transformer architectures in custom ASR design.
  - Fine-tuning the pre-trained Whisper model on noisy environments

- Data augmentation strategies rarely include advanced noise types such as Gaussian noise or realistic environmental disturbances.

To address these limitations, our study introduces the following key contributions:

- **Design of a Custom Transformer-Based ASR Architecture:** We build a Transformer-based ASR model , designed specifically for robustness in noisy environments. This allows us to have full control over the model structure and tailor it to handle long-range dependencies in audio more effectively than traditional RNN-based approaches. Our custom architecture benefits from the parallel processing capabilities of Transformers, enabling faster training and better scalability.
- **Fine-Tuning the Whisper Model on Noisy Data:** We fine-tune the pre-trained Whisper model on domain-specific, noisy datasets. Fine-tuning allows us to adapt Whisper's powerful learned representations to our target environment while significantly reducing training time and data requirements. This approach leverages Whisper's strengths while making it more resilient to real-world acoustic variations.
- **Advanced Noise Augmentation with Gaussian and Realistic Sounds:** We implement a comprehensive noise augmentation strategy that introduces Gaussian noise and real-world environmental sounds into the training data. These augmentations improve the model's ability to generalize and distinguish speech from background noise, contributing to better recognition performance in diverse acoustic scenarios.

In summary, our work combines the strengths of a custom-built Transformer model and fine-tuned Whisper with robust noise augmentation techniques. This hybrid strategy results in an ASR system that is better equipped to handle noisy environments, addressing key gaps left by previous research.

Chapter 3

Analysis and Specification of Requirements

In this chapter, we embark on a comprehensive exploration of the analysis and specification of requirements for our project

3.1 Requirement Analysis

The requirement analysis phase is pivotal in the development journey, providing the cornerstone for crafting effective solutions. This phase entails methodically collecting, documenting, and scrutinizing the requirements and desires of stakeholders to delineate the project's scope and functionalities.

3.1.1 Actors Identification

1. **End Users:** Individuals who interact with the voice detection system, such as users of voice assistants or operators in noisy environments.

2. **Administrator:** Responsible for managing and configuring user accounts, monitoring model performance, and inheriting user permissions.

3.1.2 Functional Requirements:

Actor 1: Admin

- **FR1: Authentication**
  The administrator must be able to securely log into the system using a unique username and

password.

- **FR2: Monitor model performance**
  The administrator has access to the performance metrics of the model
- **FR3: Manage users**
  The administrator should be able to create, update, and delete user profiles and manage permissions for different users.
- **FR4: Upload voice recording**
  The administrator can upload mp3 audio file and choose a language to receive both the original transcription and its translation in the selected language.
- **FR5: Record voice**
  The administrator can record voice input and choose a language, to receive both the original transcription and its translation in the selected language.
- **FR6: Play input voice recording**
  The administrator must be able to listen to the input voice recording or file.
- **FR7: Select target language**
  The administrator must be able to choose the target language for translation and transcription.
- **FR8: Display transcription**
  The administrator must be able to view the transcription of the input speech in both the original language and the selected target language.
- **FR9: Play target language voice**
  The administrator must be able to listen to the transcribed text in the target language through text-to-speech synthesis.

Actor 2: End User

- **FR1: Authentication**
  The end user must be able to securely log into the system using a unique username and password
- **FR2: Upload voice recording**
  The user can upload mp3 audio file and choose a language to receive both the original transcription and its translation in the selected language.

- **FR3: Record voice**
  The user can record voice input and choose a language, to receive both the original transcription and its translation in the selected language.
- **FR4: Play input voice recording**
  The user must be able to listen to the input voice recording or file.
- **FR5: Select target language**
  The user must be able to choose the target language for translation and transcription.
- **FR6: Display transcription**
  The user must be able to view the transcription of the input speech in both the original language and the selected target language.
- **FR7: Play target language voice**
  The user must be able to listen to the transcribed text in the target language through text-to-speech synthesis.

3.1.3 Non-Functional Requirements

- **Reliability**: To enable users to access information seamlessly, the system must remain available at all times and function correctly without interruptions.
- **Performance**: The ASR model should demonstrate high accuracy while maintaining minimal error rates in speech-to-text predictions.
- **Security**: To ensure user safety while using the application, a robust authentication system must be implemented to prevent unauthorized access.
- **Usability**: The interface should be intuitive and user friendly, even for individuals without technical expertise.

3.2 Requirements Specification

In this section, we will discuss modeling language, use case and sequence diagrams .

3.2.1 Modeling Language

Our decision was made to model the entities using UML (Unified Modeling Language), a popular visual modeling language that provides extensive syntactic and semantic support for system conception, development, and implementation. Define, design, document, and visualize the many components of a software system. It is also utilized to successfully inform the manifesting parties of the specifications, models, and software solutions.

3.2.2 Use Case Diagrams

This diagram allows to model the interactions between actors and the system under study. Figure 3.1 illustrate the use case diagram

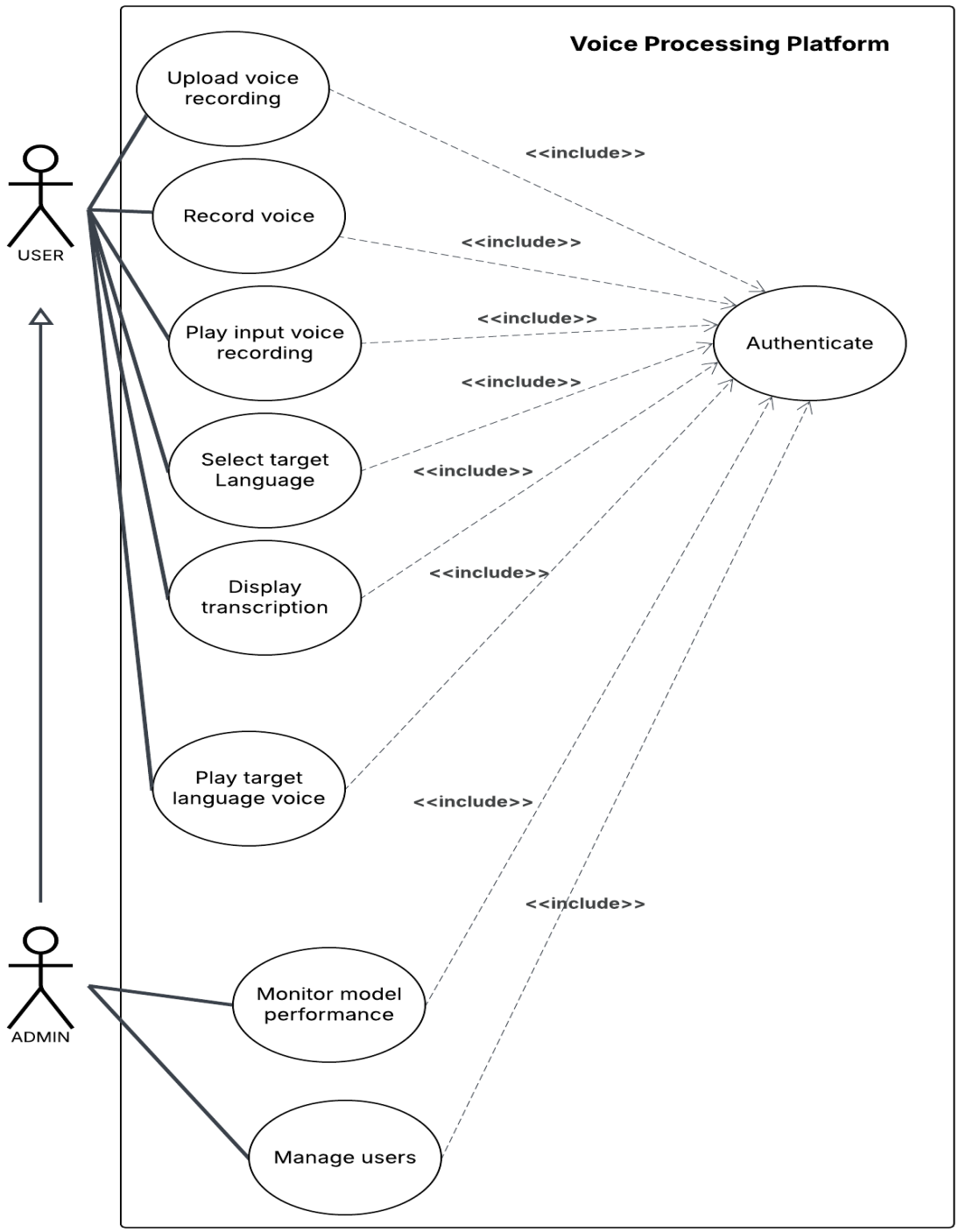

Figure 3.1: Use case diagram

3.2.3 Sequence Diagrams

A sequence diagram represents how different elements of a system interact by showing the sequence of actions and messages exchanged between them in a particular use scenario. We describe here two scenarios that illustrate the interactions between the different actors of our application.

- **Sequence diagram authentication user**

  Figure 3.2 illustrates the sequence diagram representing the user authentication process, where the user provides their username and password to access the application.

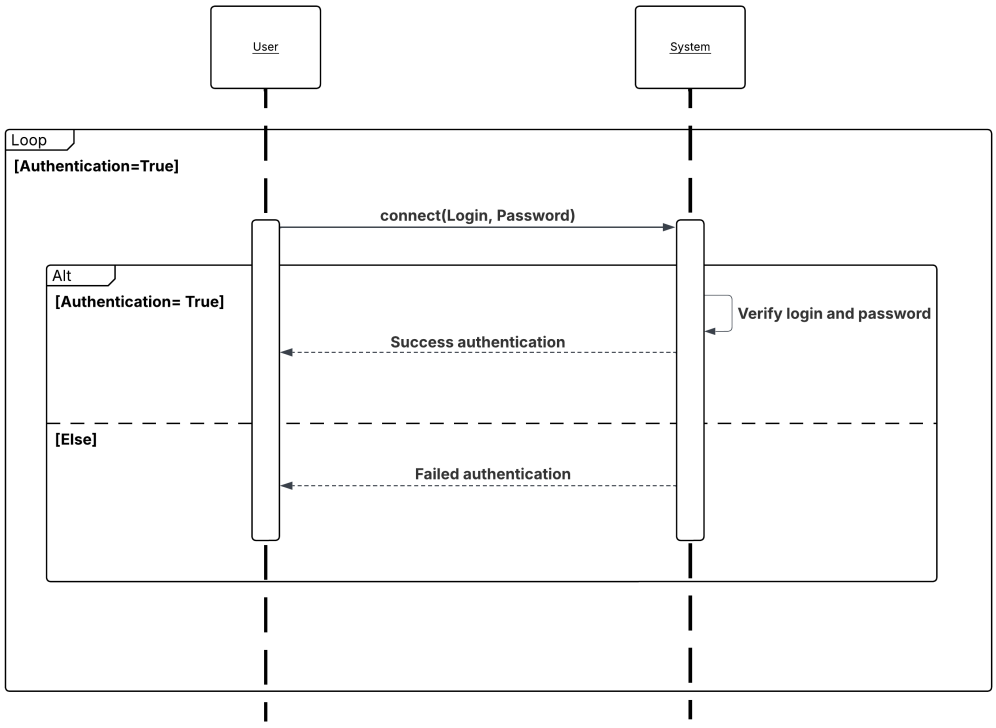

Figure 3.2: Sequence diagram authentication

- **Sequence diagram user voice recording**

  Figure 3.3 illustrates the sequence diagram of a user interacting with the system to record voice, which is then transcribed and translated by the application.

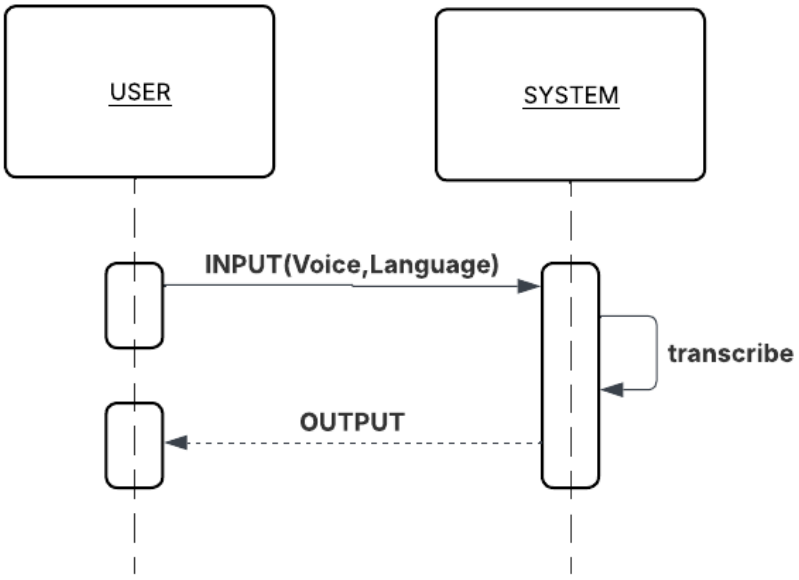

Figure 3.3: Sequence diagram voice recording

- **Sequence diagram user voice uploading**

  Figure 3.4 illustrates the sequence diagram of a user interacting with the system to upload a voice file, which is then transcribed and translated by the application.

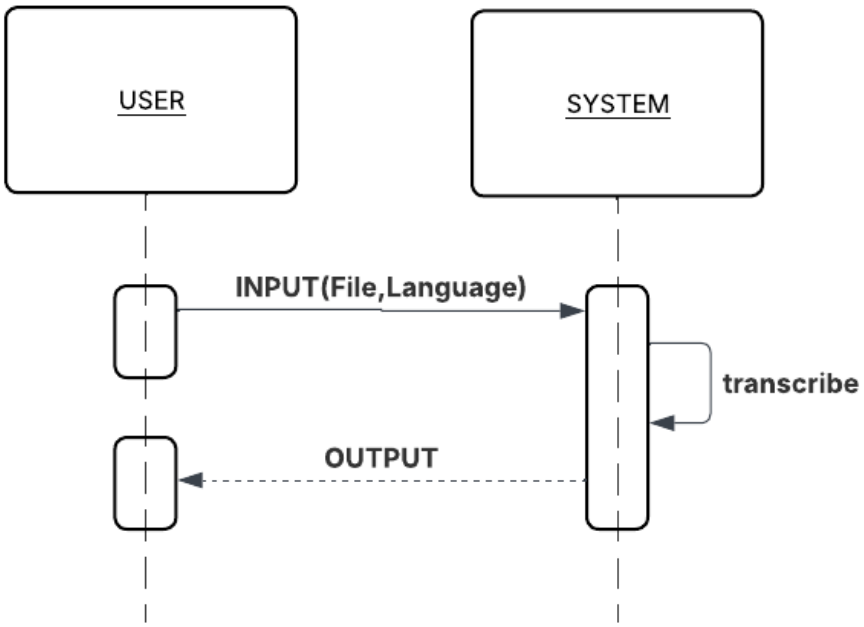

Figure 3.4: Sequence diagram voice uploading

Chapter 4

Design

In the Conception phase, the project takes shape through detailed planning and initial design. This stage involves defining the project scope, objectives, and requirements in close collaboration with stakeholders. Key tasks include conducting feasibility studies, defining the system architecture, and outlining the Outlining project plan. The Conception phase lays the foundation for subsequent development activities by establishing clear goals, identifying potential risks, and determining resource requirements.

4.1 General Design

Our primary goal throughout this design phase was to set the application's structure and organization to satisfy both functional and non-functional requirements.

4.1.1 MVC Model

We adopted the Model View Controller (MVC) framework, for our platform. It's an used software design pattern that helps in constructing user interfaces by splitting the program logic into three linked sections; the model, the view and the controller. Each of these parts is made to handle aspects of developing the application. MVC is a liked option in web development, for making projects that can grow and expand smoothly.

- **Model:**

  The Model is the backbone of our system, managing all data related operations, from handling business logic to facilitating communication between the View and Controller. it transcribe and translate voice input recorded or uploaded into the target language. Preprocessing steps

like noise reduction and feature extraction are applied to ensure accurate and reliable results, even in noisy environments.

- **View:**

  The View component serves as a visual interface to the system, presenting data in multiple formats such as charts, diagrams, or tables. It is designed to be flexible, allowing the same data to be represented differently depending on the user's needs for instance, a bar chart for managerial insights or a tabular format for accounting purposes. In our case, the View displays a chart that compares the performance of the different models utilized within the application.

- **Controller:**

  Controllers act as the link connecting the Model and View components, managing the business logic and processing user requests. They handle data operations via the Model and collaborate with the View to produce the final output. For example, imagine a customer controller managing input from the customer view, updating the database using the customer model, and displaying customer information. In our case, the controller handles audio input requests whether the user chooses to record voice or upload an audio file then forwards this input to the model for transcription and translation. Once the model returns the processed text, the controller sends the result to the view for display, ensuring seamless interaction between the user interface and the underlying logic.

A detailed explanation of this concept will be presented in Figure 4.1 [12].

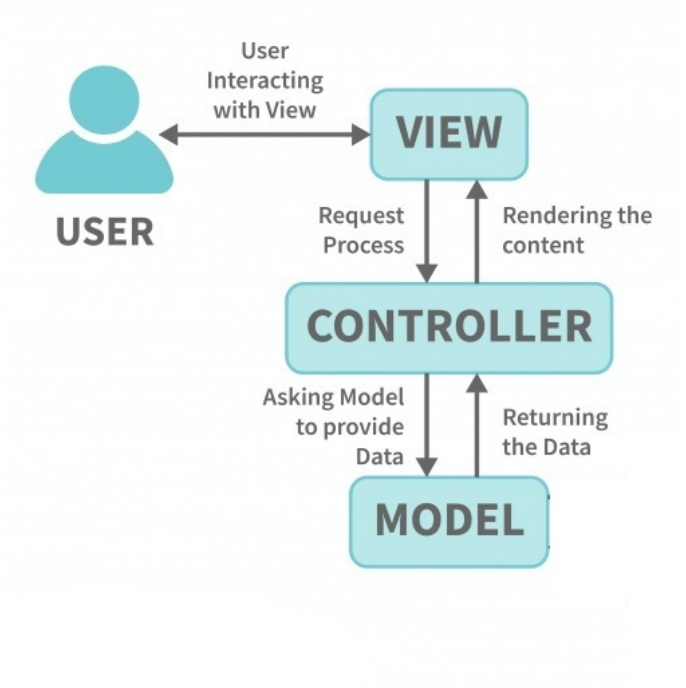

Figure 4.1: MVC architecture

4.1.2 Physical Architecture

In our application, we adopt a two-tiered architecture, comprising the client tier and the server tier. Each tier is responsible for a distinct set of functions:

- **Client Tier:**

  The client tier is where users interact directly with the system. This includes the user interface, which allows users to authenticate themselves, record or upload voice data, and view the transcription and translation results. The interface also provides administrators with access to model performance metrics and user management tools. This layer is responsible for capturing input from users and displaying the output in a user-friendly format.

- **Server Tier:**

  The server tier handles both the application logic and database operations. It processes requests from the client tier, executes business logic, performs speech transcription and translation using deep learning models, and interacts with a MySQL database to persist data. This centralized server ensures real-time data manipulation through CRUD operations and maintains consistent communication with the client tier.

Figure 4.2 [13] illustrates the two-tier architecture adopted in our application, where the client tier interacts directly with the server tier, which handles both application logic and database operations.

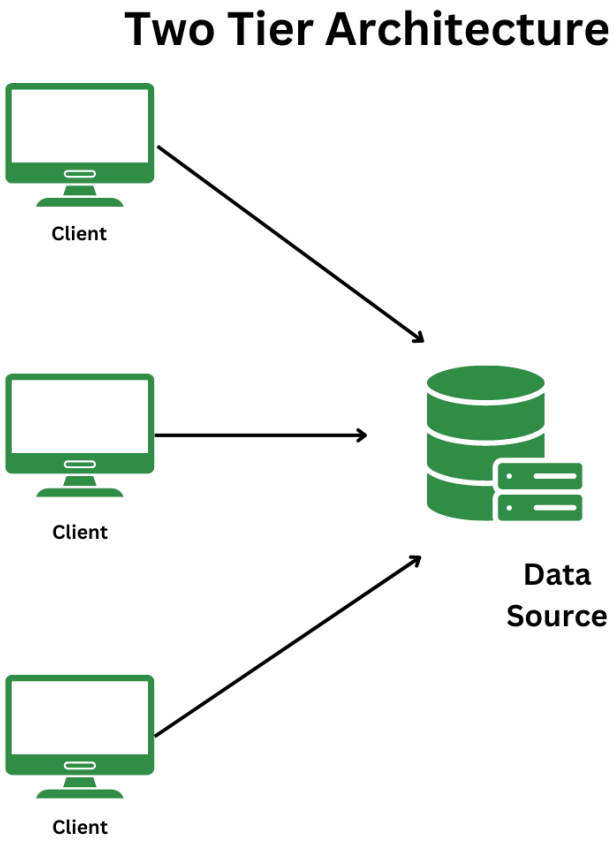

Figure 4.2: 2 tier architecture

4.2 Detailed Design

At this point we will provide detailed conception and we visually express business processes and the interactions between various system elements using class and activity diagrams. By using these tools, you can make sure that the solution is best suited to satisfy the needs of the client.

4.2.1 Detailed Sequence Diagram

- **Sequence diagram MVC authentication**

  Figure 4.3 illustrates the sequence diagram of the MVC representing the user authentication process, where the user provides their username and password to access the application

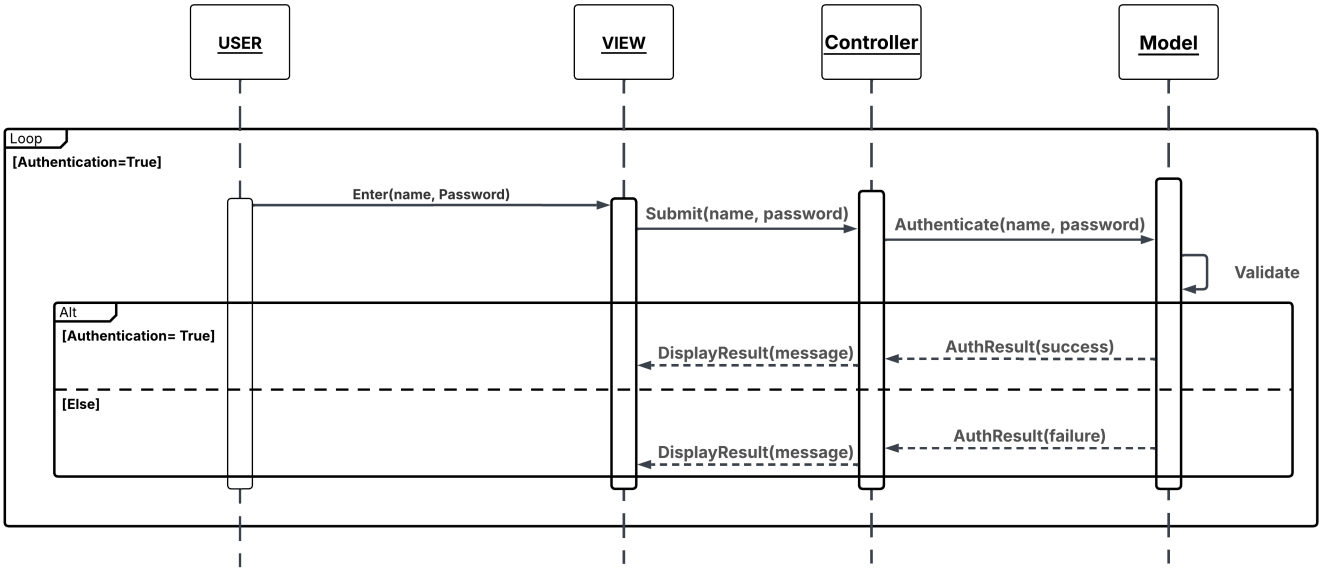

Figure 4.3: Sequence diagram MVC authentication

- **Sequence diagram MVC voice processing**

  Figure 4.4 illustrates the sequence diagram of the MVC based voice processing, where the user records audio input and receives the transcribed and translated output.

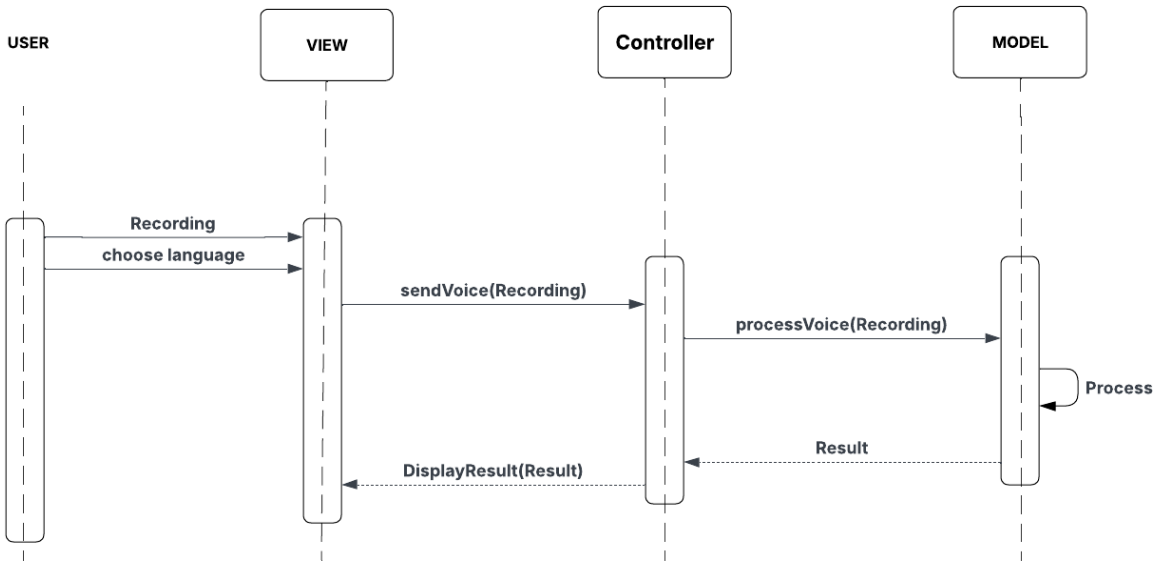

Figure 4.4: Sequence diagram MVC voice processing

4.2.2 Class Diagram

The class diagram provides an overview of the system's architecture by outlining the classes, their properties, functions, and interrelationships. Figure 4.5 presents the class diagram of our application.

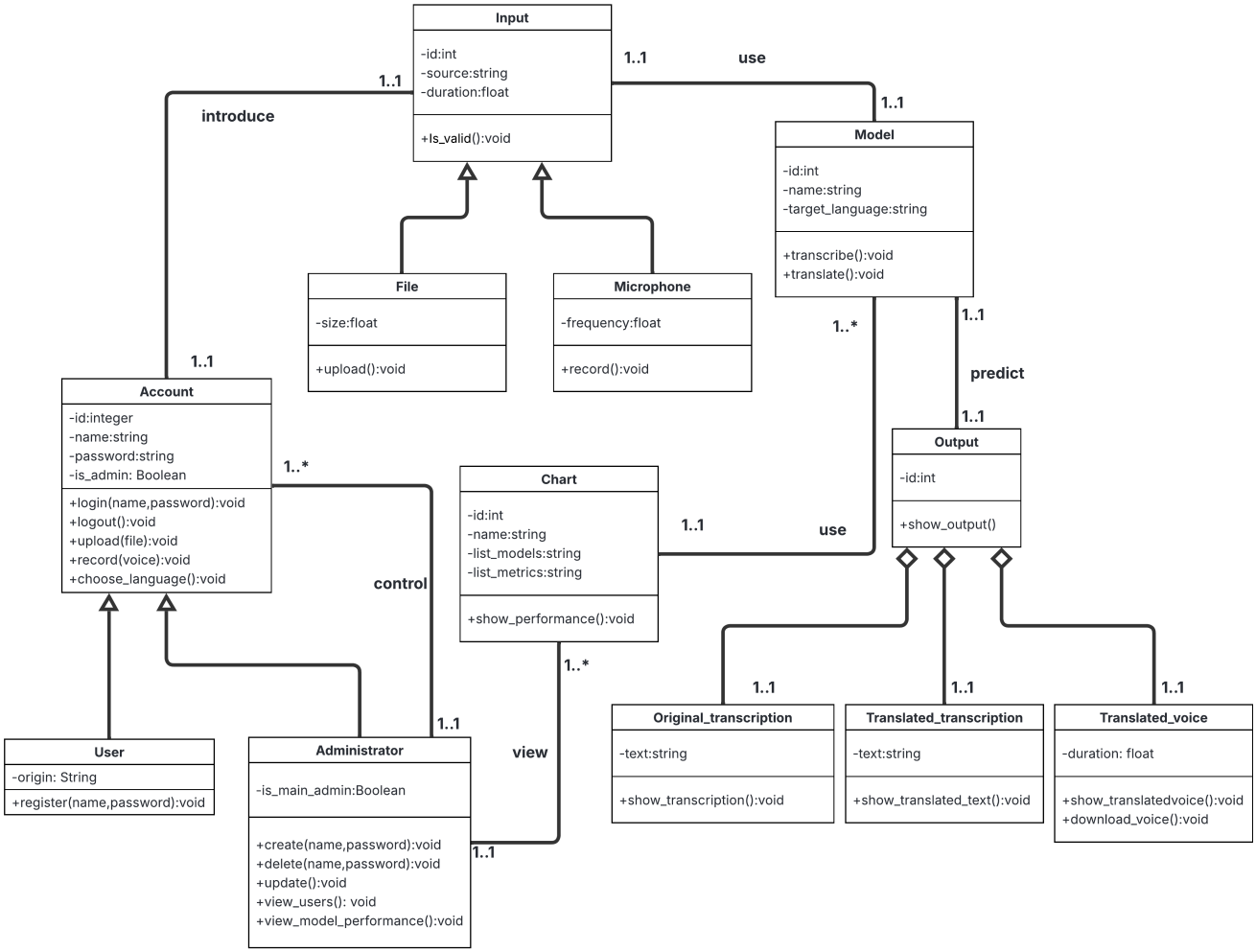

Figure 4.5: Class diagram

4.2.3 Activity diagram

The Activity Diagram illustrates the dynamic aspects of a system by showing the flow of control or data from one activity to another. It is particularly useful for modeling the logic of complex operations, workflows, or business processes. In the context of our application, the activity diagram incorporates three main gateways: User, System, and Admin. This diagram Figure 4.6 clearly outlines how these components interact dynamically throughout the application's workflow.

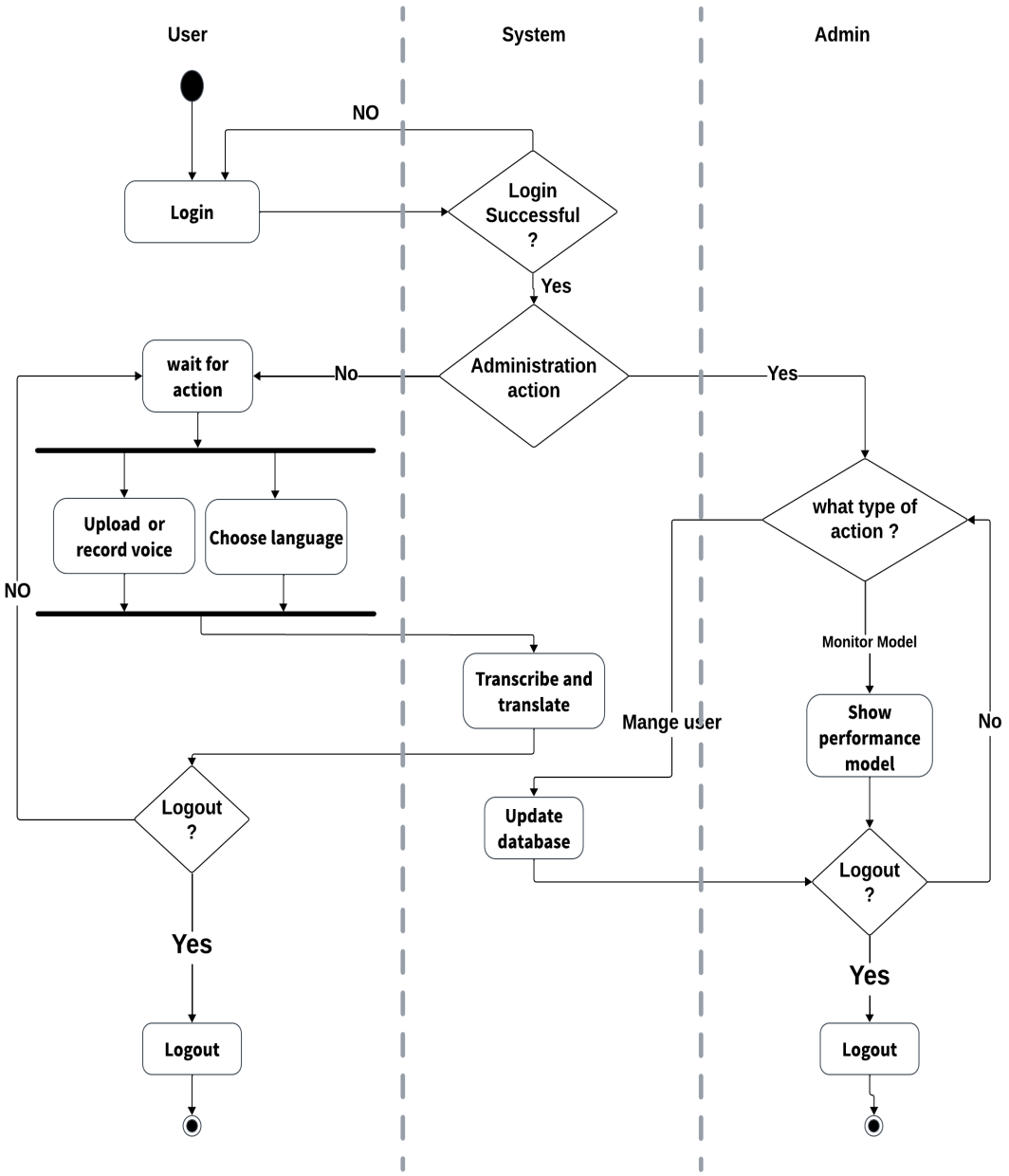

Figure 4.6: Activity diagram

Chapter 5

Methodology and Proposed Solution

This chapter will cover topics such as data collection, preprocessing, model development and evaluation. To effectively structure our approach, we propose to use the CRISP-DM (Cross-Industry Standard Process for Data Mining) as can be depicted in Figure 5.1 [14]. This widely adopted framework guides the entire data mining process through six iterative phases: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment. By following this methodology, we ensure that each step is aligned with our objectives and that our solution is both robust and actionable.

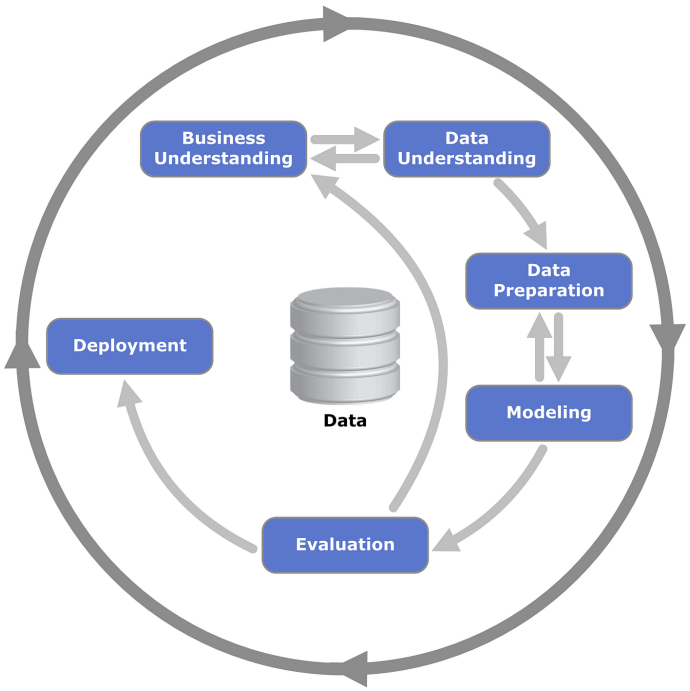

Figure 5.1: CRISP-DM diagram

The primary goal of this work is to develop a robust speech recognition pipeline leveraging deep

learning techniques to transcribe audio inputs into text. The process begins with data understanding, where multiple datasets such as TIMIT and LibriSpeech are explored to build a diverse and representative dataset. The data preprocessing phase involves several key steps, including feature elimination, character removal, and the use of the Wav2Vec2 model for advanced audio feature extraction. Preprocessing also includes tokenization, normalization, resampling, and framing.

Subsequently, different modeling strategies are applied, such as sequential architectures (BiLSTM and BiRNN), transformers, and the pre-trained Whisper model, which is directly fine tuned on test data. Model performance is assessed during the evaluation phase using metrics like WER, CER, and loss. Finally, the best-performing model is integrated into a deployment system that enables real-time speech-to-text conversion. Figure 5.2 illustrates the full pipeline from data acquisition to deployment.

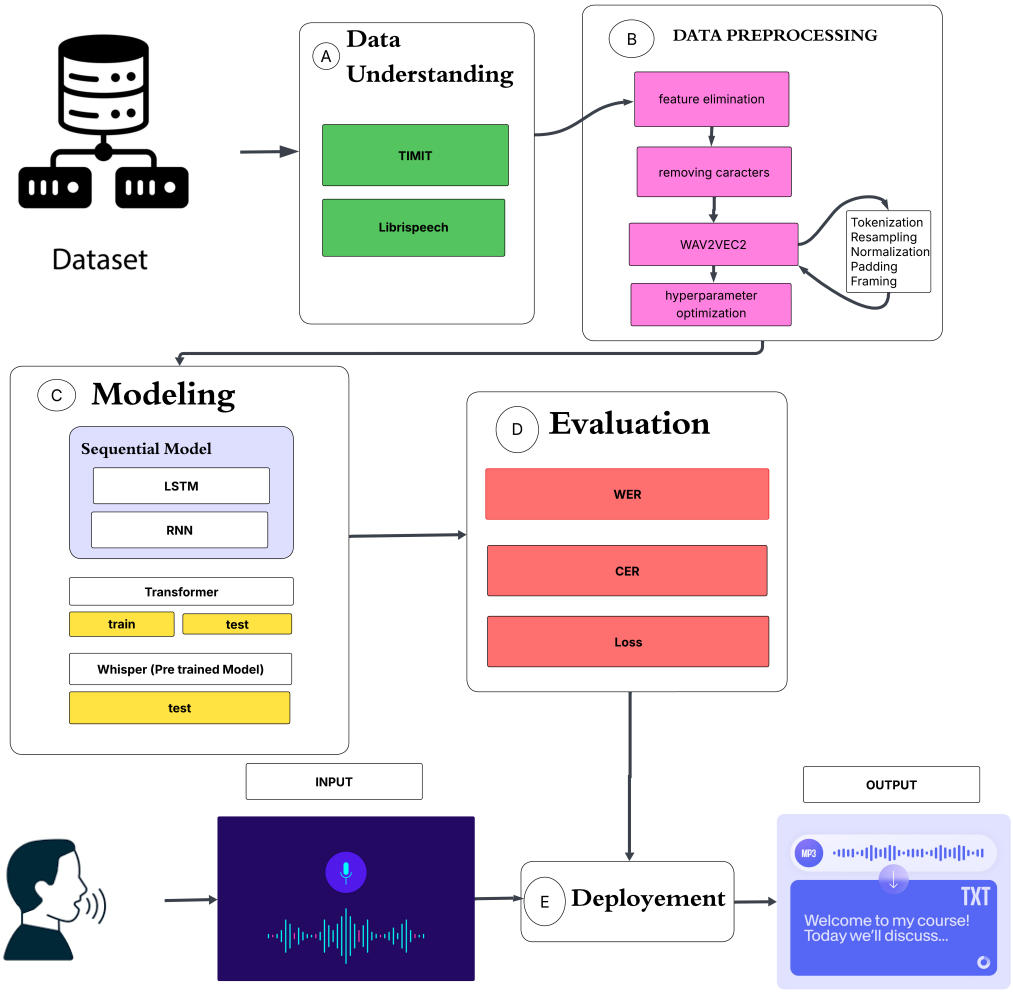

Figure 5.2: Overview of the proposed methodology for the multilingual automatic speech recognition system

5.1 Data Collection and Understanding

In our project, we leveraged multiple datasets to improve the robustness and performance of our speech-to-text system. By incorporating diverse sources such as LibriSpeech and TIMIT, we aimed to account for variations in audio quality, accents, and speaking styles key challenges in transcription tasks. This diversity enabled us to train a more generalized and reliable model, capable of adapting to real-world conditions. In the following section, we discuss the data collection and understanding of the LibriSpeech and TIMIT datasets.

- **Dataset 1: Librispeech Dataset**

  We opted to use the well-known LibriSpeech dataset, which consists of recordings sourced from volunteer-read audiobooks.

  Other datasets, such as Common Voice and TED-LIUM, were considered, but we prioritized LibriSpeech due to its structured format and well-established benchmarks in speech recognition research. The dataset consists of high-quality FLAC recordings sampled at 16 kHz, paired with accurate text transcriptions. The diversity in speaker accents and speech patterns enhances the robustness of the model. Additionally, the provided metadata, including speaker age and gender, facilitates dataset analysis and preprocessing.

  In the following table 5.1, we provide an overview of the LibriSpeech dataset, highlighting its key characteristics and limitations.

Table 5.1: Librispeech train-clean-100 dataset overview

LibriSpeech train-clean-100 dataset	
Total Duration	100 hours
Number of Speakers	251
Number of Utterances	285,000
Audio Format	FLAC
Sample Rate	16 kHz
Transcriptions	Provided (word-level alignment)
Strength	High-quality, diverse speaker accents, widely used in ASR research
Limitation	Read speech (not spontaneous), limited domain (audiobooks)

The LibriSpeech dataset consists of 1,000 hours of read English speech, derived from public domain audiobooks. The dataset includes recordings from 2,456 speakers, covering a diverse range of accents and speech patterns. It is divided into several subsets, with train-clean-100 containing 100 hours of high-quality speech from 251 speakers, ensuring clear and accurately transcribed audio.

Each recording in LibriSpeech is accompanied by a precise text transcription, allowing for supervised learning in speech recognition tasks. The dataset is further categorized based on recording quality and speaker characteristics into subsets such as clean and other, with the former containing clearer, more intelligible speech.

The figure 5.3 shows the distribution of audio file durations in the LibriSpeech train-clean-100 dataset:

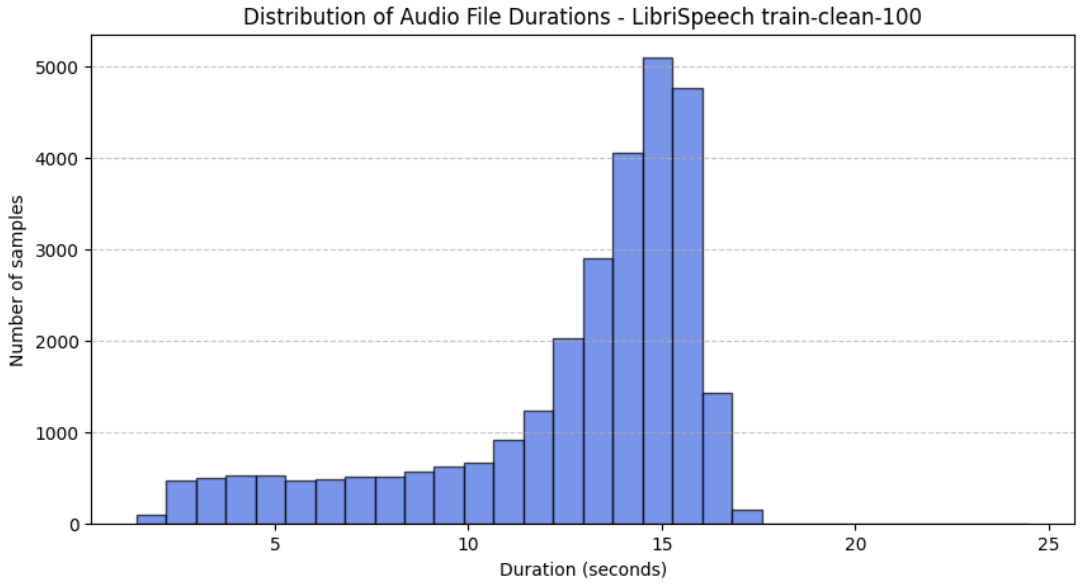

Figure 5.3: Distribution of audio file durations - librispeech train-clean-100

The histogram visualizes the range and frequency of the audio file lengths, providing insights into the dataset's composition in terms of speech duration. Such information is crucial for understanding the variability in the data.

- **Dataset 2: TIMIT Dataset**

  The TIMIT dataset, short for Texas Instruments/Massachusetts Institute of Technology, is a widely used benchmark in speech processing and phoneme recognition research. It was specifically developed to support acoustic-phonetic studies and provide a standard dataset

for evaluating ASR systems. TIMIT contains recordings from 630 speakers, with an equal representation of males and females, and covers a broad range of American English dialects across eight major regions in the United States. Each speaker reads ten carefully selected sentences, resulting in a total of 6,300 utterances. These include two dialect sentences common to all speakers, several phonetically diverse and compact sentences, and a set of core sentences used for cross-speaker comparison.

In the following table 5.2 provides an overview of the TIMIT dataset distribution across the training and test splits:

Table 5.2: Distribution of the TIMIT dataset

Split	**Speakers**	**Utterances**	**Male**	**Female**
Training Set	462	3,696	326	136
Test Set	168	1,344	112	56
Total	630	5,040	438	192

The training set includes 462 speakers with a total of 3,696 utterances, while the test set comprises 168 speakers and 1,344 utterances. In terms of gender distribution, the dataset consists of 438 male and 192 female speakers. This balanced composition ensures a diverse representation of voices for both training and evaluation purposes.

Figure 5.4 presents the distribution of the number of files according to their duration:

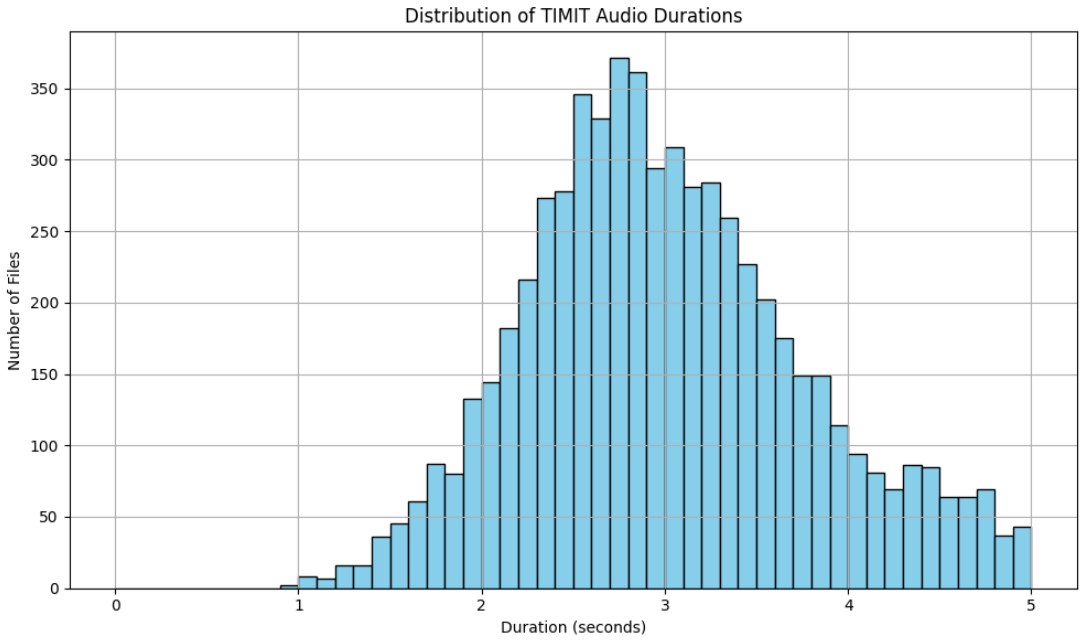

Figure 5.4: Distribution of audio file durations - TIMIT dataset

5.2 Data Preprocessing

In this section, we will discuss the data processing steps applied to the LibriSpeech and TIMIT datasets to prepare them for training and evaluation in ASR tasks.

- **Dataset 1: LibriSpeech Dataset**

  Data preprocessing is a crucial step in preparing our dataset for speech recognition modeling. Before training our model on the LibriSpeech train-clean-100 dataset, we implemented various techniques to ensure data consistency and quality. First, we identified and handled missing or corrupted audio files to maintain dataset integrity. Since the raw audio files are stored in FLAC format, we converted them into waveform representations for further processing.

  To ensure that all features were on the same scale, we applied normalization techniques, adjusting amplitude levels to reduce variations in volume across recordings. Additionally, we used denoising methods to minimize background interference, ensuring clearer speech signals. Another key aspect of preprocessing was handling inconsistencies in audio quality by applying transformations to standardize the dataset.

  Feature extraction played a fundamental role in this process. We computed Mel-Frequency Cepstral Coefficients (MFCCs), a widely used representation of speech signals that captures

essential frequency characteristics. To further improve model generalization, we applied data augmentation techniques such as time stretching, pitch shifting, and adding artificial noise. These methods helped diversify the dataset and enhance robustness against variations in speech patterns.

- **Dataset 2: TIMIT Dataset**

  In the preprocessing phase of the TIMIT dataset for training a speech recognition model, unnecessary metadata fields are removed to retain only the audio and its corresponding text. The transcriptions are then normalized by eliminating punctuation and converting all text to lowercase for consistency. After that, a tokenizer is initialized to convert the cleaned text into a format suitable for model training. This streamlined process ensures the dataset is properly prepared for use in an ASR system.

  - **Feature Elimination and Character Removal**

    The TIMIT dataset includes a variety of features such as audio files, transcripts, dialect regions, sentence types, word-level files, phonetic annotations, and speaker IDs. For our purposes, we only utilize the audio files and corresponding transcripts, as these are sufficient for training a speech recognition model. Additionally, the dataset is normalized by removing punctuation marks including commas, question marks, periods, exclamation marks, hyphens, semicolons, colons, and quotation marks.

  - **Wav2Vec2**

    Converts text to token IDs and back and processes raw audio by normalizing it and ensuring it's at the right sampling rate (usually 16kHz), preparing it for the model also combines both the tokenizer and feature extractor into a single tool, streamlining the process of turning raw audio into model inputs and converting model outputs back into readable text.

  - **Tokenization**

    Before the model can be trained, the raw text data must first be converted into a format that the model can understand. This involves transforming the text into a sequence of tokens, where each token represents a character or symbol defined in the vocabulary. This

tokenization process is essential, as it allows the model to map spoken audio features to discrete, learnable units of language during training.

– **Padding and Framing**

Different audio clips are often different lengths. To handle them together in a batch, Wav2Vec2 adds zeros to the end of the shorter clips to make them all the same length. This is called padding.Also the audio is broken into chunks and each chunk into a set of numbers that represent the sound.

In conclusion, these preprocessing steps were essential for optimizing the quality of our data, ensuring that our speech recognition model could learn effectively and perform accurately in real-world scenarios.

5.3 Model Building

The development of our speech recognition system involved experimenting with five distinct architectures: Residual CNN + Bidirectional RNN, Residual CNN + Bidirectional LSTM (Long Short-Term Memory), Transformers, Residual CNN + Bidirectional GRU, and Whisper. Each architecture was evaluated individually as a standalone ASR system to assess its performance. Notably, all architectures except Whisper share a common core component: a Residual CNN module, which plays a crucial role in enhancing feature extraction for ASR tasks. The detailed structure of this Residual CNN is illustrated in Figure 5.5 [15]

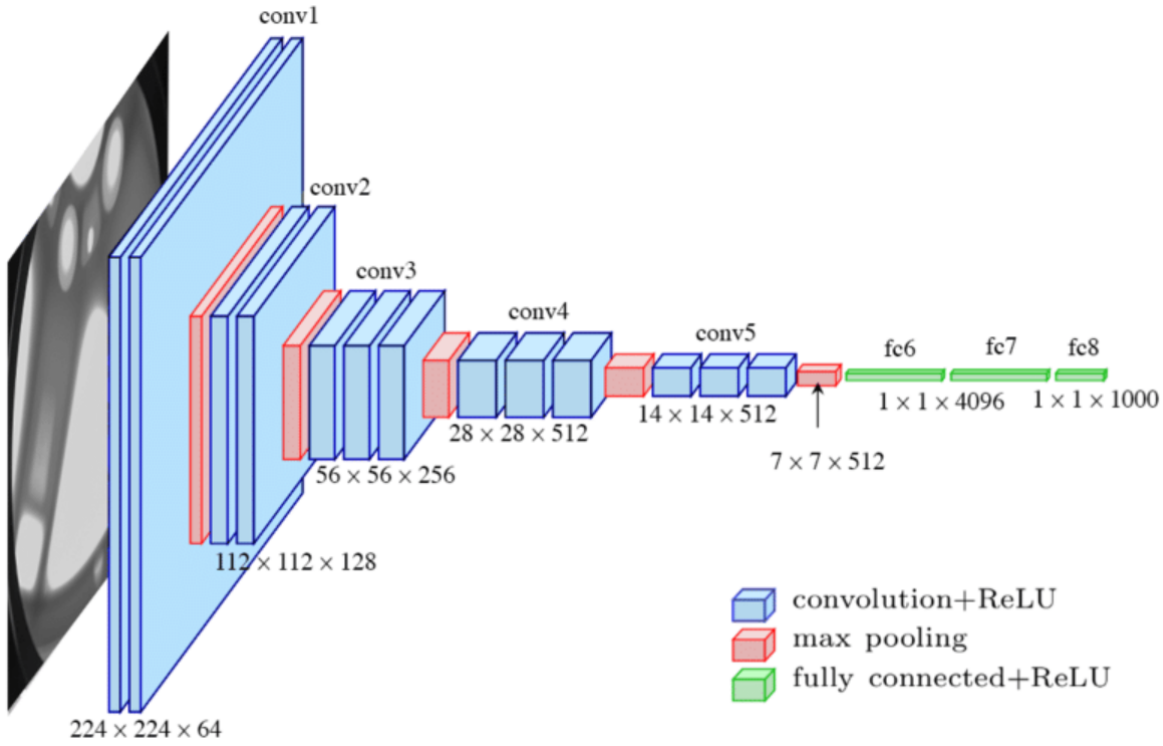

Figure 5.5: CNN architecture

- **Residual CNN + BiRNN-Based ASR Model:**

  Recurrent Neural Networks (RNNs) are well-suited for processing sequential data thanks to their ability to maintain a hidden state that captures temporal dependencies. However, conventional RNNs often face challenges with long-term dependencies due to issues like vanishing gradients. To address this, we employed a Bidirectional RNN (BiRNN), which processes the input sequence in both forward and backward directions. This bidirectional mechanism enables the model to leverage both past and future context, significantly enhancing performance in sequence-based tasks such as speech recognition. The architecture of the BiRNN model is shown in Figure 5.6 [16].

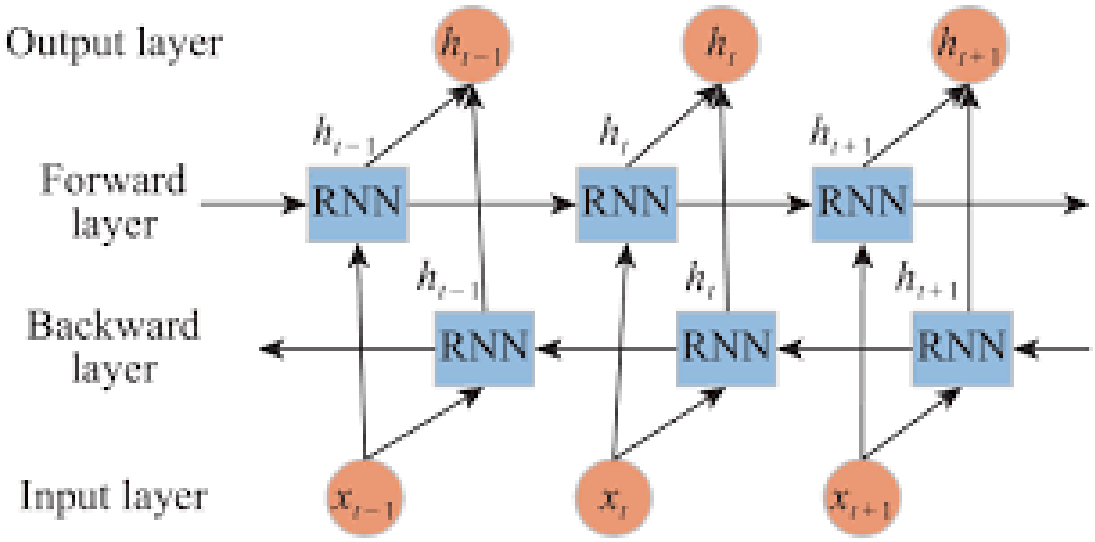

Figure 5.6: BiRNN architecture

To strengthen feature extraction, we integrated residual convolutional layers at the front of the network. These layers efficiently process input spectrograms, capturing both fine-grained and abstract acoustic features, while their residual (shortcut) connections help alleviate training difficulties such as vanishing gradients. The extracted features are subsequently fed into the BiRNN layers, improving the model's understanding of temporal patterns in speech.

Furthermore, we stacked multiple BiRNN layers to increase the model's capacity for learning complex feature representations. We employed Connectionist Temporal Classification (CTC) as the loss function, allowing the network to learn the alignment between input audio and target transcriptions without requiring pre-segmented data a key advantage for ASR.

This architecture forms a fully end-to-end ASR model that directly maps input spectrograms to transcriptions. The complete structure of the Residual CNN + BiRNN model is illustrated in Figure 5.7 [17].

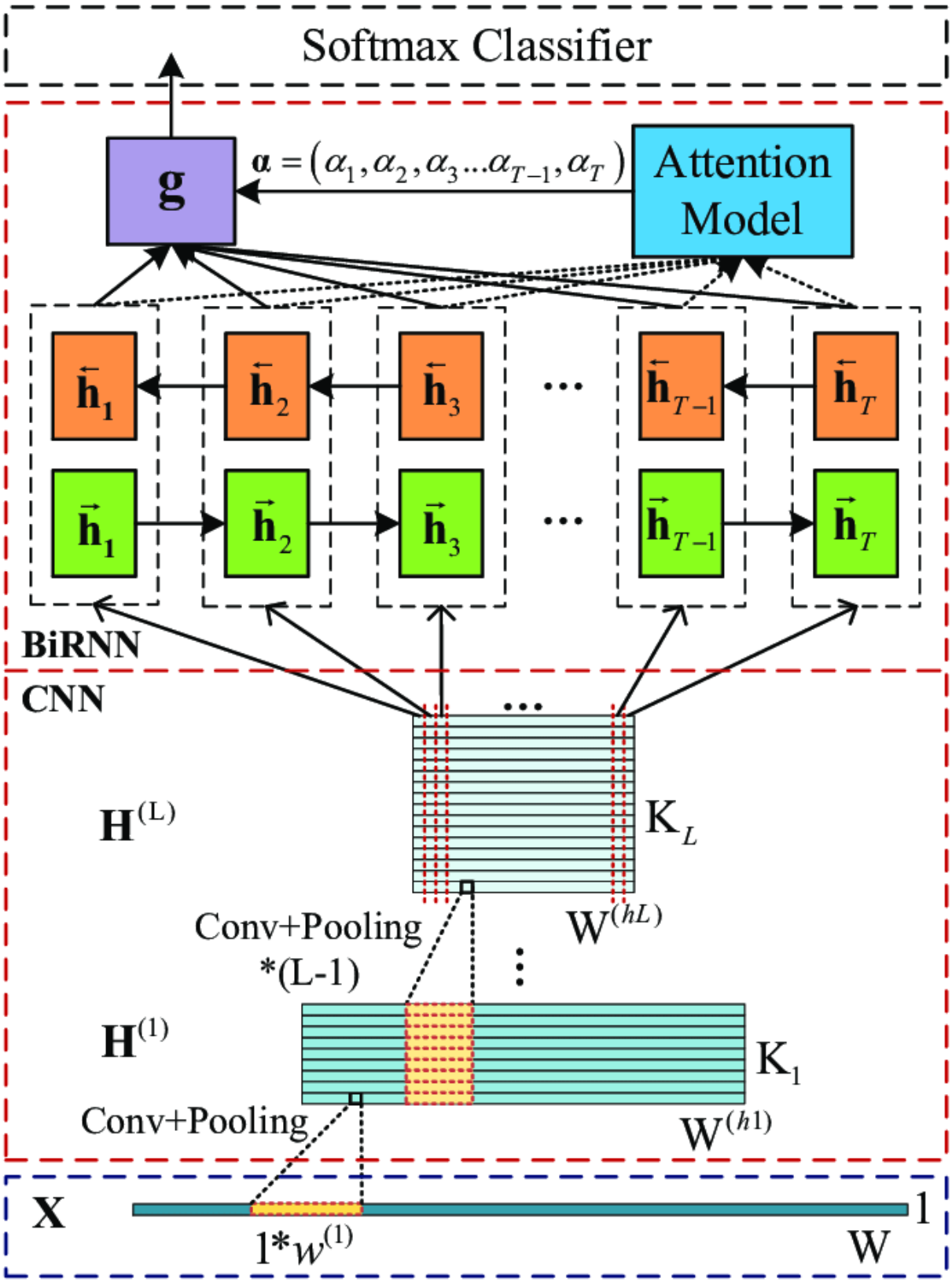

Figure 5.7: Residual CNN + BiRNN model architecture

- **Residual CNN + BiLSTM-Based ASR Model:**

  In this configuration, we implemented a BiLSTM network integrated with residual convolutional layers to construct a standalone ASR model. The Residual CNN layers serve as a front-end module to efficiently extract both low and high level acoustic features from spectrogram inputs, while their shortcut connections help mitigate training issues such as vanishing gradients and facilitate faster convergence.

  Following feature extraction, the BiLSTM layers process the sequential data in both forward and backward directions, allowing the model to capture contextual information from both the past and the future. This bidirectional approach significantly improves the model's ability to interpret the temporal structure of speech. The architecture of a typical BiLSTM is illustrated in Figure 5.8 [18].

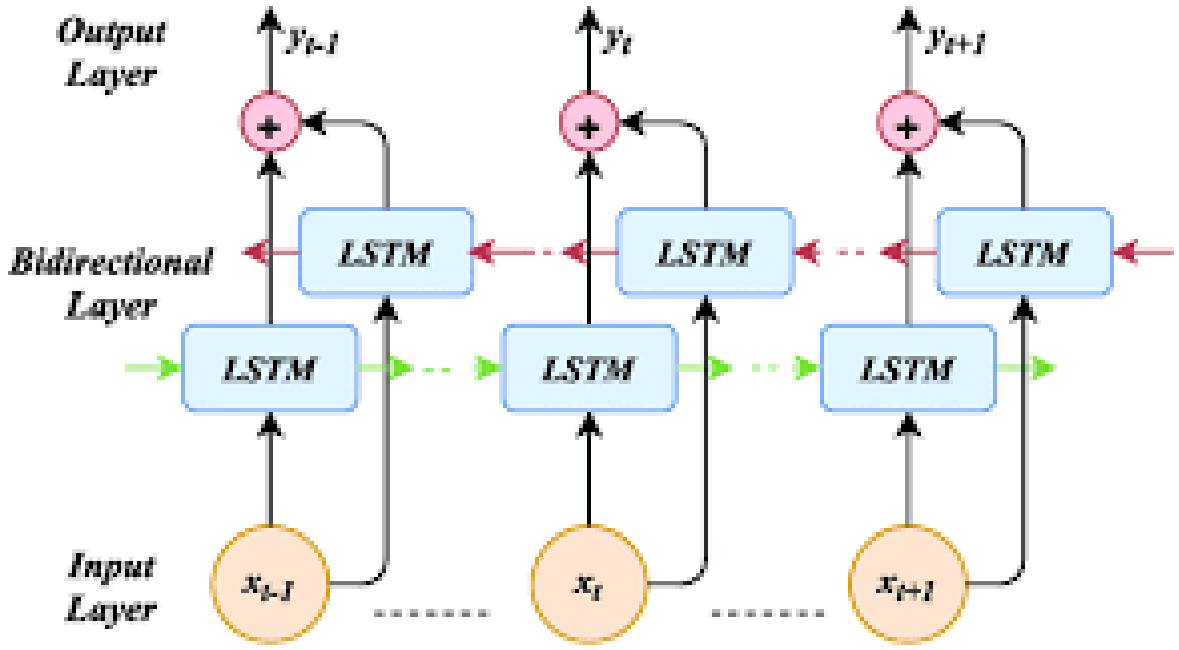

Figure 5.8: BiLSTM architecture

Our complete BiLSTM-based ASR model architecture includes the following components:

- Residual CNN layers for robust and stable acoustic feature extraction.
- Multiple stacked BiLSTM layers to learn deep, hierarchical temporal representations of speech.
- Dropout regularization and batch normalization layers to improve generalization and training stability.

The outputs from the BiLSTM layers are passed to a fully connected softmax layer, which generates the final transcription probabilities. The complete architecture of the Residual CNN + BiLSTM ASR model is depicted in Figure 5.9 [19].

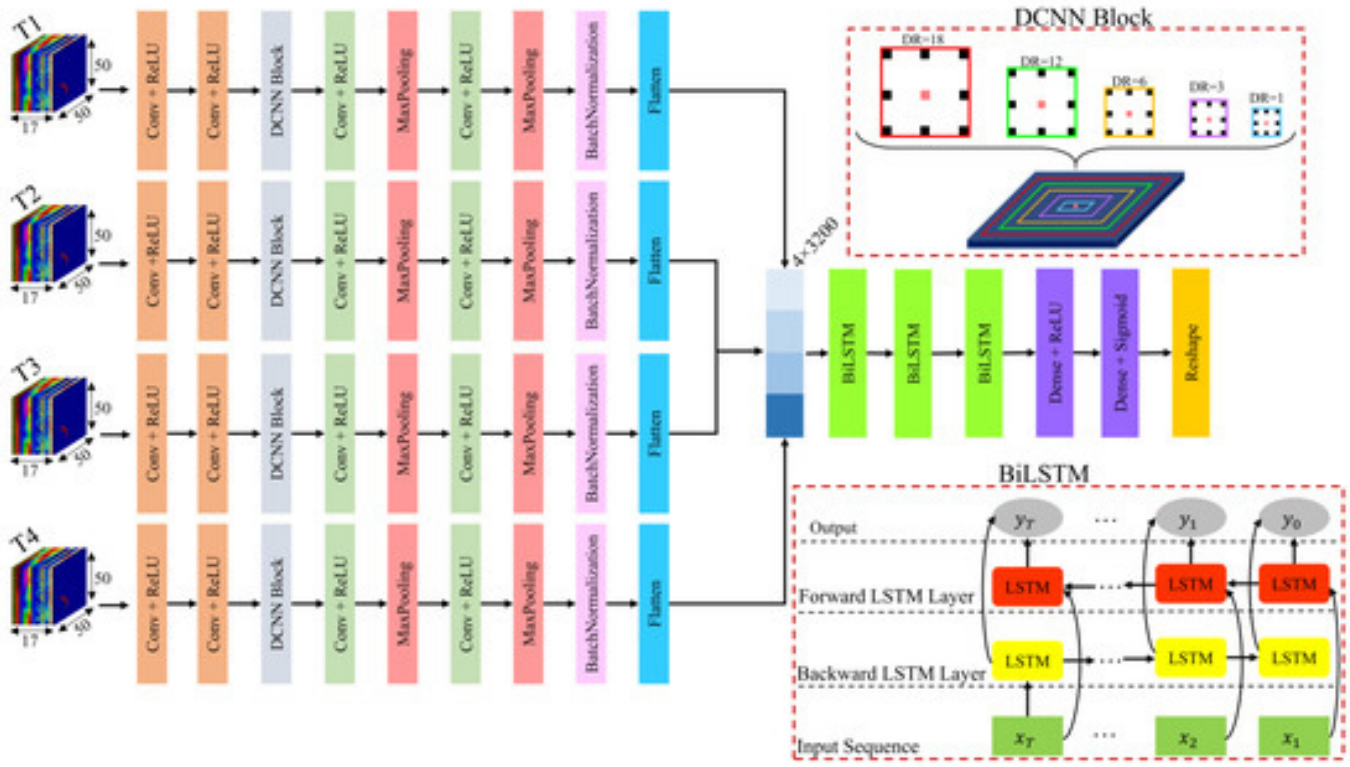

Figure 5.9: Residual CNN + BiLSTM model architecture

- **Transformer-Based ASR Model:**

  To investigate a non-recurrent alternative for speech recognition, we implemented a Transformer-based ASR model. Unlike LSTM-based architectures, Transformers leverage self-attention mechanisms to model long-range dependencies across the input sequence without relying on recurrence. This design enables efficient parallel processing and improved context modeling. A general overview of the Transformer architecture is presented in Figure 5.10 [20].

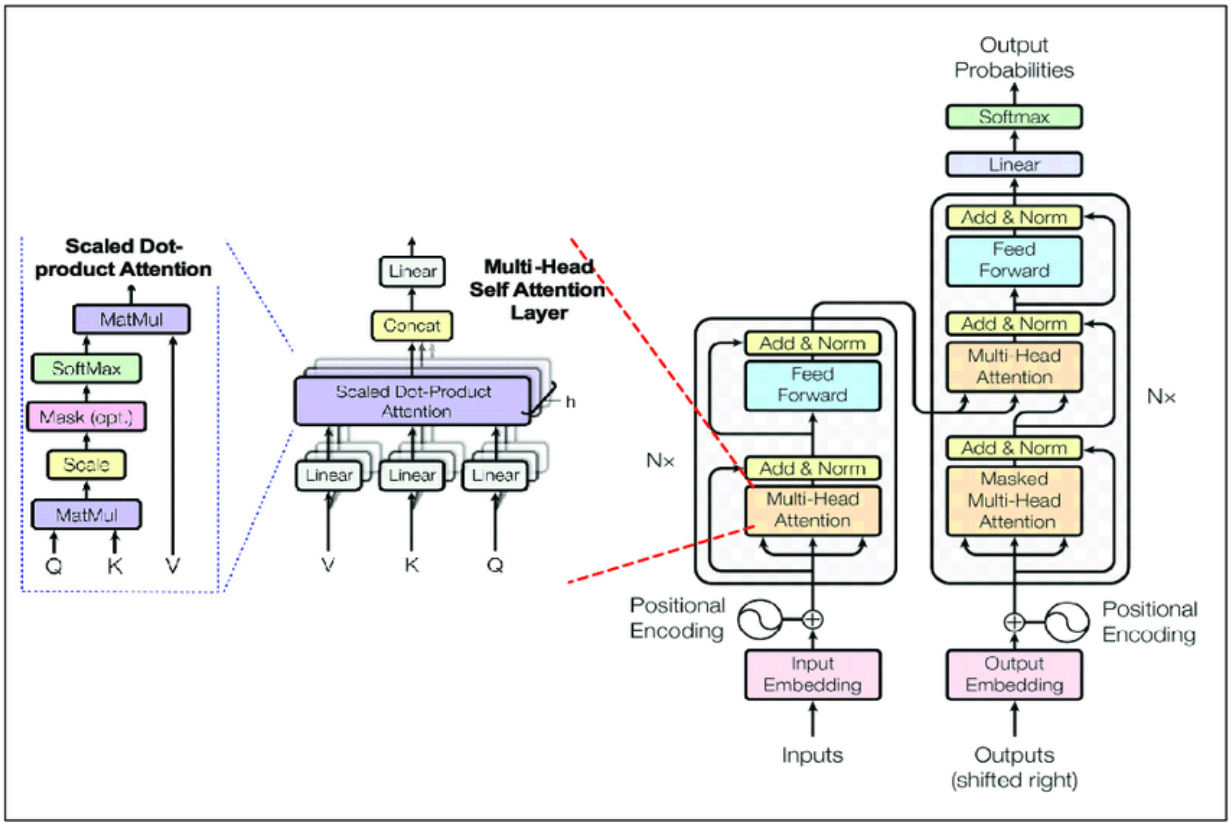

Figure 5.10: Transformer architecture

Our Transformer-based ASR model integrates the following key components:

- Multi-head self-attention layers that allow the model to focus on various parts of the speech input simultaneously, enhancing contextual understanding.
- Positional encoding to retain the sequential nature of speech, compensating for the lack of recurrence.
- Feedforward layers with residual connections, which promote stable and efficient training by facilitating gradient flow.

To further improve performance, we also experimented with pretrained Transformer-based models such as Wav2Vec 2.0. These models, trained on large-scale speech corpora, were fine-tuned on our specific dataset, enabling better generalization and improved accuracy.

The complete architecture of our ASR model combining residual CNN layers with Transformer blocks is shown in Figure 5.11 [21].

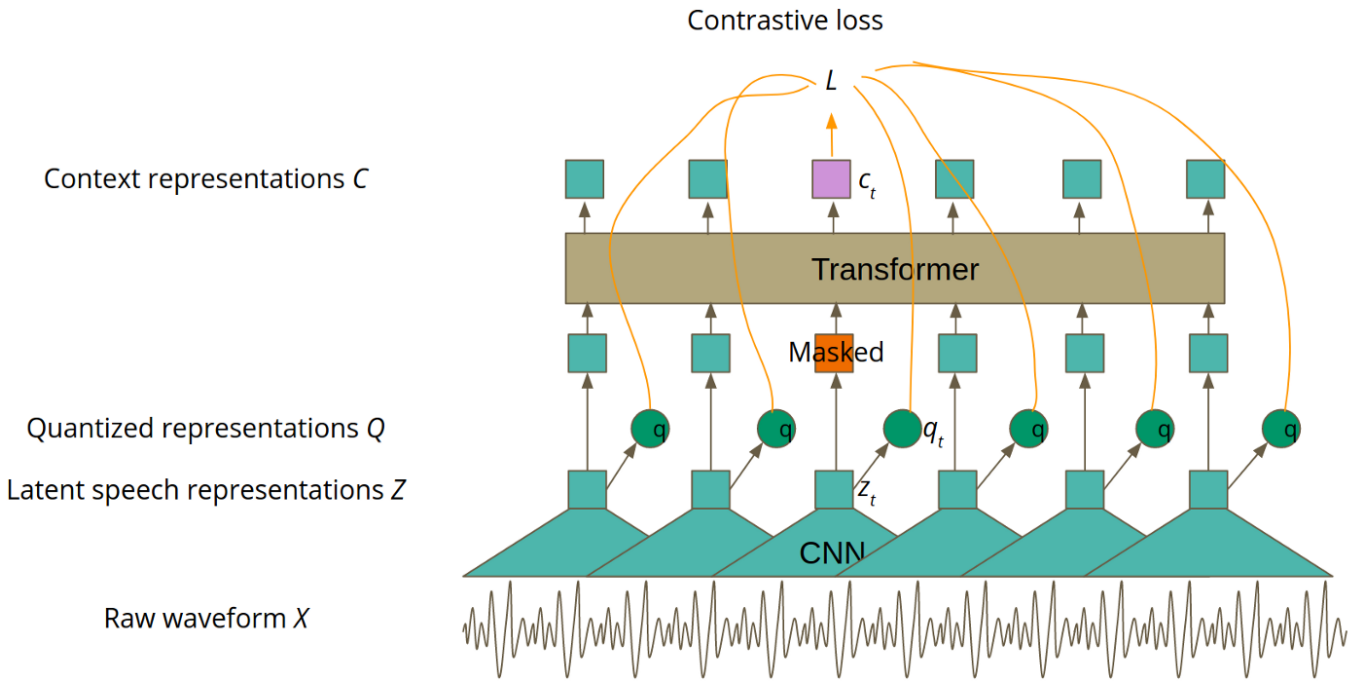

Figure 5.11: Residual CNN + Transformer model architecture

- **Residual CNN + BiGRU-Based ASR Model:**

  The BiGRU-based ASR model combines residual convolutional layers with Bidirectional Gated Recurrent Units (BiGRUs) to transcribe spoken language into text effectively. This hybrid architecture leverages the strengths of both convolutional and recurrent components. The general architecture of the BiGRU is shown in Figure 5.12 [22].

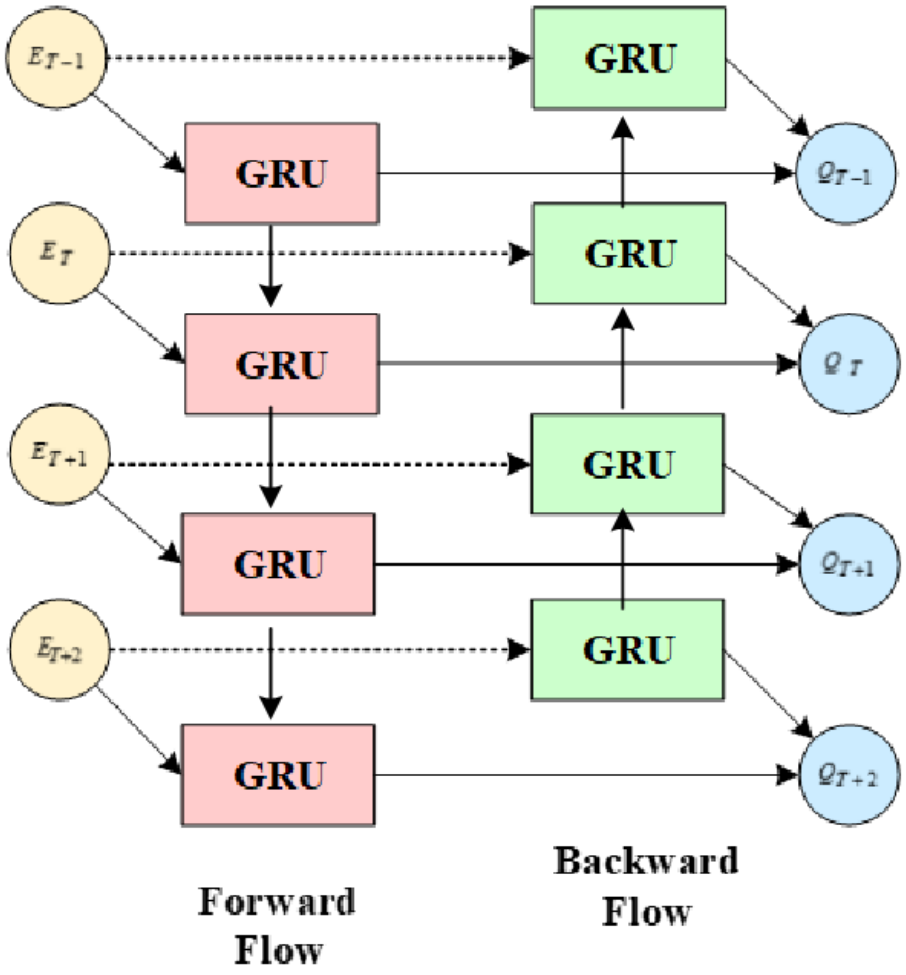

Figure 5.12: BiGRU architecture

At the front end of the model, residual CNN layers are employed to extract meaningful acoustic features from input spectrograms. These layers are capable of capturing both low-level and high level representations while residual connections facilitate improved gradient flow and stable training in deep architectures.

The extracted features are then processed by BiGRU layers, which extend standard GRUs by incorporating bidirectional context. This allows the model to analyze the speech signal from both forward and backward temporal directions, significantly enhancing its ability to capture contextual dependencies in the audio sequence.

The combination of residual CNNs for spatial feature extraction and BiGRUs for temporal modeling results in a powerful end-to-end ASR architecture. This model is both computationally efficient and effective in recognizing complex speech patterns.

The complete structure of the Residual CNN + BiGRU model is presented in Figure 5.13 [23].

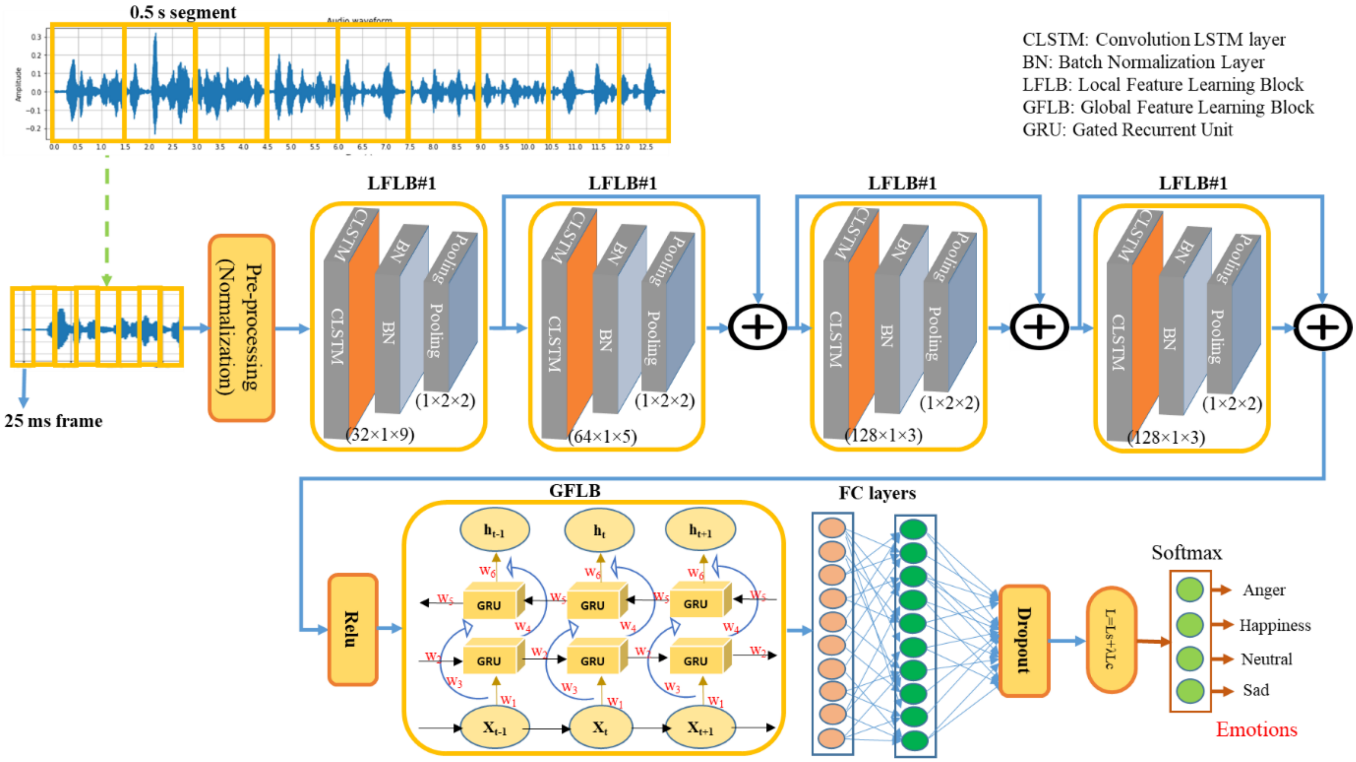

Figure 5.13: Residual CNN + BiGRU model architecture

- **Whisper Model:**

  Whisper is a state of the art ASR system developed by OpenAI, available in multiple model sizes including tiny, small, medium, and large. Each variant presents a trade off between transcription accuracy and computational efficiency. Smaller models, such as tiny and small, are lightweight and optimized for real time transcription or deployment in resource constrained environments. In contrast, the medium model offers a balanced performance profile, while the large model delivers the highest transcription accuracy at the cost of increased computational demands.

  In our experiments, we integrated and fine-tuned the Whisper model to evaluate its ASR capabilities within our framework. Among the available variants, the large model demonstrated superior performance, effectively handling complex speech patterns, diverse accents, and noisy audio conditions. Its robust generalization makes it well suited for high precision transcription tasks, particularly in professional or multilingual contexts where transcription quality is paramount.

The architecture of the Whisper model we fine-tuned, based on a Transformer backbone, is illustrated in Figure 5.14 [24].

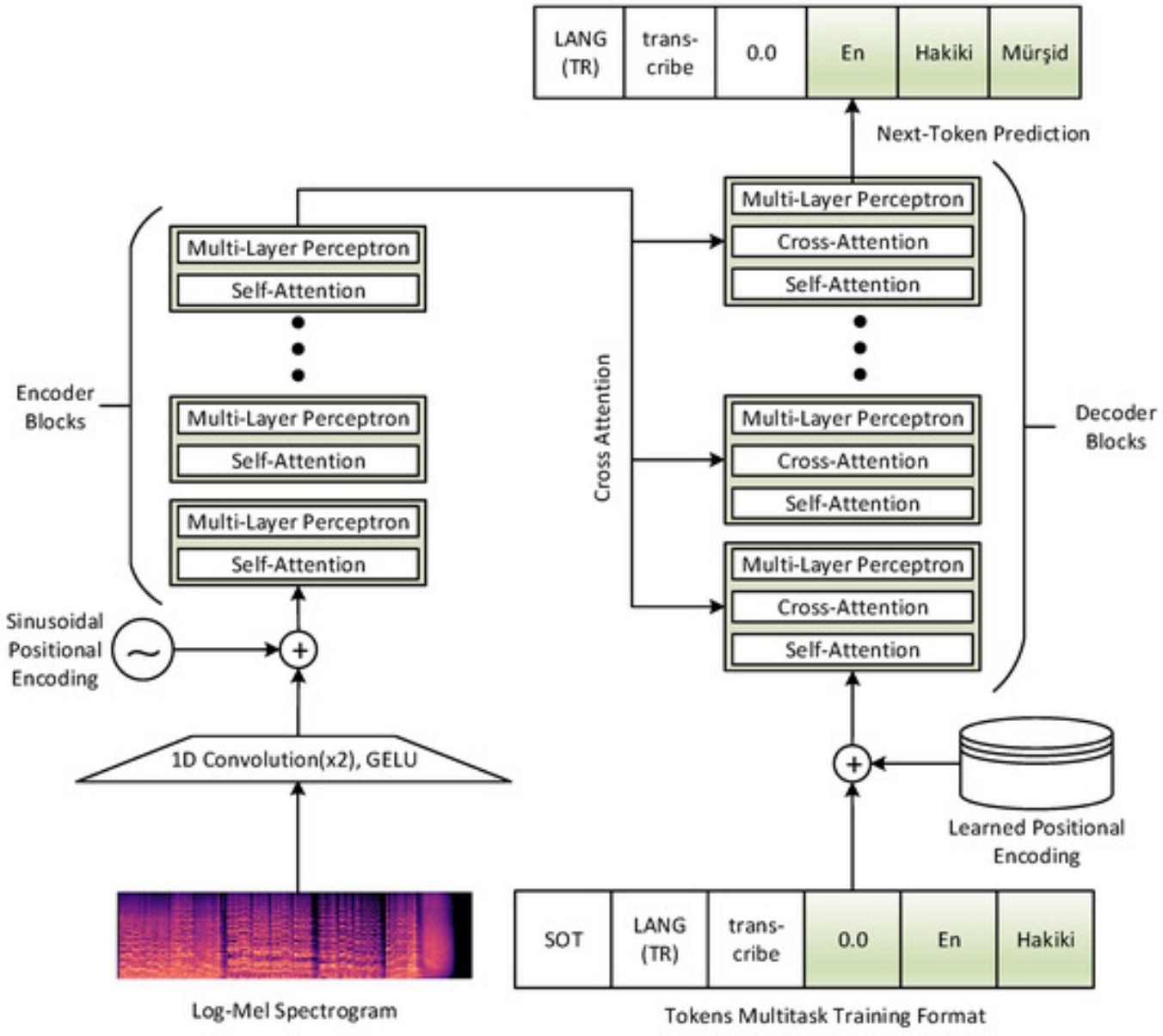

Figure 5.14: Whisper architecture

Each of these models was evaluated separately, allowing us to compare their effectiveness in ASR tasks and analyze their strengths and limitations.

5.4 Evaluation Metrics

In ASR, evaluating model performance is essential for ensuring accurate transcriptions. Our evaluation relies on several key metrics, including Validation Loss, WER, and CER. While WER and CER quantify transcription errors to assess the overall accuracy of the model, Validation Loss, typically computed using the Connectionist Temporal Classification (CTC) loss function, provides insight into the model's ability to generalize to unseen data. The CTC loss is fundamental in training sequence-to-sequence models without requiring explicit alignment between input speech and output text

- **Validation Loss**

  Validation loss is the loss function computed on a separate validation dataset that is not used during training. It measures the model's ability to generalize to unseen data. In the context of ASR, this loss is often based on the CTC function, which is specifically designed for sequence-to-sequence tasks where the alignment between the input audio and the target transcription is unknown. A low validation loss indicates that the model is performing well on new data, while a sudden increase may suggest overfitting.

- **CTC Loss**

  Since speech and text sequences vary in length, traditional supervised learning techniques cannot directly map each frame of an audio signal to a specific character. The CTC loss function addresses this issue by enabling models to learn probabilistic alignments between input and output sequences. Given an audio input X and a target transcription Y, CTC computes the probability $P(Y|X)$ by summing over all possible alignments, allowing the model to generate transcriptions without predefined time alignments. The CTC loss is defined as:

  $$\mathcal{L}_{CTC} = -\log P(Y|X)$$

  CTC introduces a special blank token that helps merge repeated characters and aligns outputs with variable length speech signals.

- **WER**

  WER evaluates transcription accuracy at the word level and is calculated as:

  $$WER = \frac{S + D + I}{N}$$

where S represents the number of word substitutions, D is the number of deletions, I is the number of insertions, and N is the total number of words in the reference transcript. A lower WER indicates a more accurate model.

- **CER**

  CER measures accuracy at the character level, making it useful for languages with complex morphology or when dealing with short words. It is defined as:

$$CER = \frac{S + D + I}{N}$$

  where S, D, and I refer to character-level substitutions, deletions, and insertions, and N is the total number of characters in the reference text. A lower CER suggests higher transcription accuracy.

- **Accuracy**

  Accuracy is a metric used to evaluate how correctly a model transcribes spoken language. It quantifies the proportion of words that are correctly predicted by the system, taking into account the number of substitutions, deletions, and insertions compared to the reference transcription.

  The formula for accuracy is given by:

$$Accuracy = \frac{N - S - D - I}{N}$$

  where:

  - N is the total number of words in the reference transcription,
  - S is the number of substitutions,
  - D is the number of deletions,
  - I is the number of insertions.

  Accuracy is related to the WER as follows:

$$Accuracy = 1 - WER$$

Beyond numerical evaluation, we analyze WER and CER trends across different datasets to refine model training. The use of CTC loss ensures that our ASR models effectively align speech to text, making them robust in handling real world speech variations. By leveraging these metrics, we optimize our models to improve speech recognition performance and minimize transcription errors.

Chapter 6

Implementation and Results

The realization phase is an essential step. It enable the solutions put forth during the design process to be implemented. This section will cover the specifics of our solution's implementation, from setting up the development environment to displaying the features that have been put into practice.

6.1 Environment and Working Tools

The hardware environment forms the backbone of the system, providing the necessary computational resources to support AI model training, data processing, and application execution. The setup includes a high-performance server along with multiple work units that contribute to the project's overall efficiency.

6.1.1 Hardware Environment

In this section, we examine the hardware environment by detailing the specifications of the work units involved. Table 6.1 presents an overview of the hardware configurations used throughout the project.

Table 6.1: Hardware specifications per work unit

Work Unit	**Processor**	**RAM**	**Graphics Card**
Unit A (from CCK)	Intel Xeon Gold 6348	256GB	NVIDIA A40-24Q
Unit B	Intel Core i5 13th generation	32GB DDR5	NVIDIA RTX 3050
Unit C	Intel Core i5 12th generation	24GB DDR4	NVIDIA RTX 2050
Unit D	Intel Core i5 10th generation	8GB DDR4	NVIDIA MX330

6.1.2 Software Environment

In this segment, we'll explore the software environment, highlighting the key tools and platforms utilized for development and collaboration across projects.

- **Development Tools**
  - **Jupyter Notebook :** It is an open interactive development environment source for creating, executing and sharing documents that contain of code, equations, visualizations and explanatory texts.
  - **Visual Studio Code :** A source code editor with a ton of features to make programming easier.
  - **Google Colaboratory (Google Colab) :** is an online platform to write and execute Python codes. It provides access to computing resources, such as GPUs and TPUs, for free, which are used in model training and experimentation.
  - **Kaggle :** is an online service that focuses on data science and machine learning problem solving. It offers datasets, kernels, and notebooks for benchmarking and experimentation.
  - **overleaf :** is a real-time online collaborative LaTeX editor primarily used for developing research papers, reports, and academic documents.
  - **Github :** GitHub is an online platform dedicated to software development. It facilitates the storage, tracking, and collaboration of software projects, enabling developers to easily share code files and work together on open-source projects. Is an online text editor for writing LaTeX documents. It allows users to work on projects in a collaborative way, from manage versions, compile documents and generate PDFs directly to from the interface.
  - **MySQL Workbench :** is a comprehensive development tool for database management. It allows users to design, model, and manage MySQL databases through an intuitive visual interface. With features like SQL query development, database schema visualization, and server management, MySQL Workbench simplifies database administration and development, improving productivity and efficiency in the process.

- **Programming Language:**

  In this part, we will explore the programming languages utilized within the project, highlighting their features and applications in software development.

  - **Python** : Python is widely used in AI projects because of its simplicity, readability, and the vast ecosystem of libraries and frameworks it offers. Tools like PyTorch, numpy, torchaudio make it easy to develop, train, and deploy AI models. Its large community and strong support also help developers quickly find solutions and accelerate development.

- **Libraries**

  To optimize the development process, several libraries and frameworks have been integrated. These tools offer pre-built functionalities, enhancing productivity and significantly reducing development time.

  - **Matplotlib:** is a library used to visualize and analyze data during the initial stages of the project to better understand its structure and distribution.
  - **Torchaudio:** provides efficient tools for transforming raw waveforms into model-ready input, making it ideal for audio and speech-related applications. It integrates seamlessly with PyTorch, allowing developers to build end-to-end audio processing pipelines
  - **Plotly:** used for visualizing various aspects of our project, particularly the performance of the models. Its interactive and dynamic plotting capabilities allowed us to create clear and insightful visualizations, including comparison charts between different models or configurations.

- **Database**

  We used a database in our project to manage and store data in a structured and reliable way. It plays a key role during deployment by serving as the backend database that allows users to access, store, and retrieve information through the application.

  - **MySQL:** MySQL is a widely used relational database known for its reliability, speed, and ease of use. It supports structured data storage and complex queries, making it ideal for efficient data retrieval and manipulation. With robust transactional support and

scalability, MySQL ensures high performance under heavy workloads, making it a popular and cost-effective choice for both small and large-scale applications.

6.2 Experimental Results and Evaluation

In this phase, the project transitions from the planning and design stages to the actual development and evaluation of the ASR models. This includes training, testing, and fine-tuning of different deep learning architectures using speech datasets. We detail both the implementation choices and the experimental results obtained.

6.2.1 Experimental Results

In this section, we present the results of our experiments conducted on two distinct speech datasets: LibriSpeech and TIMIT. For the LibriSpeech dataset, we evaluated the performance of several architectures, including Residual CNN combined with Bidirectional RNN, Bidirectional LSTM, and Bidirectional GRU, as well as Transformer and Whisper models. In contrast, for the TIMIT dataset, we focused exclusively on the Transformer model to assess its performance on this phonetically rich corpus.

To ensure consistency and fairness across evaluations, we used a common set of training hyperparameters, adapted to the nature of each model.The table 6.2 provides an overview of the key hyperparameters employed for each architecture.

Table 6.2: Overview of hyperparameters used in the different models

Parameter	**Residual CNN + BiRNN**	**Residual CNN + BiLSTM**	**Transformer**	**Residual CNN + BiGRU**	**Whisper(fine-tuned: large model)**
Learning Rate	$3e-4$ (decay)	$5e-4$ (decay)	$3e-4$ (warmup + decay)	$3e-4$ (decay)	$3e-5$ (warmup + decay)
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
Batch Size	16	16	8	16	16
Number of Layers	3 CNN + 5 BiRNN	3 CNN + 5 BiLSTM	12	3 CNN + 5 BiGRU	32 encoder + 32 decoder
Dropout	0.2	0.1	0.1	0.2	0.1
Feature Extraction	MelSpectrogram+CNN	MelSpectrogram+CNN	MelSpectrogram+CNN	MelSpectrogram+CNN	Log-Mel Spectrogram
Hidden Size	512	512	512	512	1280
Attention Heads	-	-	8	-	20
Feedforward Dim	-	-	1024	-	5120
Epochs	30	30	30	30	30
Loss Function	CTC Loss	CTC Loss	CTC Loss	CTC Loss	Cross-Entropy Loss

- **Dataset 1: Librispeech Dataset**

  In this part of the experiment, we evaluated three different models on the LibriSpeech dataset: an Bidirectional RNN model, a Bidirectional LSTM Model, a Residual CNN + Bidirectional GRU, a Transformer model and Whisper model. Each model was trained for 30 epochs to assess their performance in reducing the WER , CER and Loss validation over time.

  Residual CNN + BiRNN Model

  In the following table 6.3 presents the evaluation results during the training of the residual CNN + BiRNN model on the Librispeech dataset over 30 epochs.

Table 6.3: Evaluation during training on librispeech dataset with residual CNN + BiRNN model (30 epochs)

Epoch	WER (%)	CER (%)	Loss (%)	Epoch	WER (%)	CER (%)	Loss (%)
0	99.61%	99.79%	16.05%	15	35.69%	37.13%	5.70%
1	92.56%	96.21%	11.70%	16	35.27%	37.41%	5.78%
2	83.47%	89.89%	9.58%	17	35.91%	37.22%	5.74%
3	70.34%	74.54%	8.87%	18	34.58%	36.06%	5.69%
4	68.72%	70.18%	8.55%	19	34.87%	36.33%	5.75%
5	64.28%	67.84%	8.35%	20	34.35%	35.91%	5.66%
6	60.41%	64.53%	8.23%	21	33.19%	34.88%	5.63%
7	55.24%	59.06%	7.91%	22	33.48%	34.95%	5.65%
8	50.92%	54.38%	7.70%	23	33.04%	34.69%	5.60%
9	47.58%	50.41%	7.45%	24	32.28%	33.83%	5.62%
10	44.67%	47.07%	7.05%	25	32.93%	33.56%	5.57%
11	42.39%	44.28%	6.65%	26	32.17%	33.71%	5.59%
12	40.45%	42.04%	6.35%	27	32.89%	33.38%	5.53%
13	37.19%	40.51%	6.10%	28	31.73%	33.21%	5.51%
14	36.08%	39.32%	5.87%	29	31.51%	33.03%	5.47%

Initially, at epoch 0, the model exhibits very high WER and CER values, 99.61% and 99.79%, respectively, with a loss of 16.05. As the training progresses, the model gradually improves. By epoch 15, the WER has decreased to 35.69%, the CER to 37.13%, and the loss reduced to 5.70. From epoch 15 to epoch 29, the model continues to show consistent improvement, although at a slower rate. By the end of the 30th epoch, the WER reaches 31.51%, the CER is 33.03%, and

the loss is minimized to 5.47. This trend demonstrates the model's ability to refine its speech recognition performance, progressively reducing errors throughout the training period.

Figure 6.1 shows the cumulative WER, CER and Loss variation across epochs for the BiRNN model trained on the librispeech dataset.

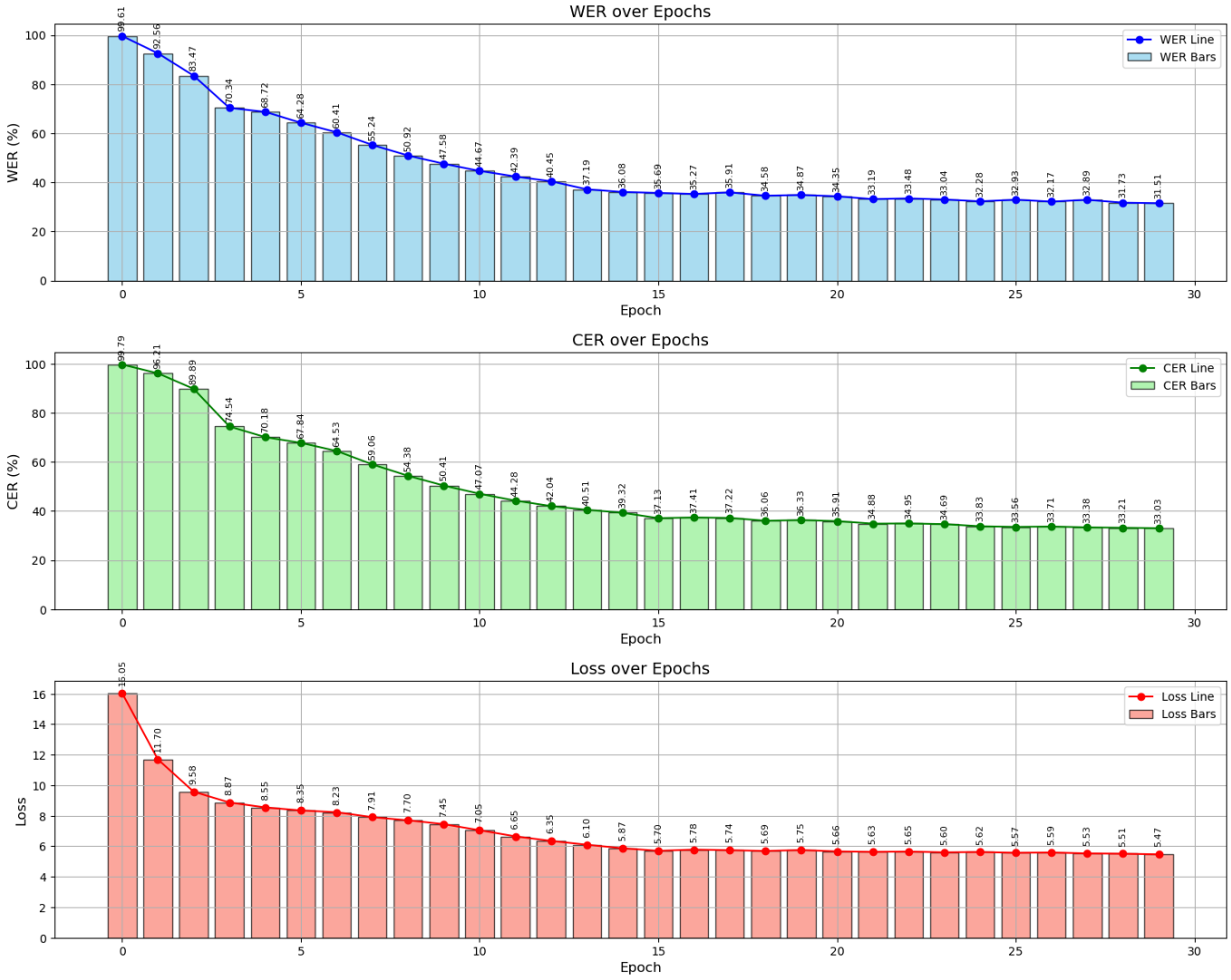

Figure 6.1: Cumulative variation of WER, CER and loss across epochs on librispeech dataset with residual CNN + BiRNN Model

Residual CNN + BiLSTM Model

In the following table 6.4 shows the evaluation results of the residual CNN + BiLSTM model during training on the LibriSpeech dataset across 30 epochs.

Table 6.4: Evaluation during training on librispeech dataset with residual CNN + BiLSTM model (30 epochs)

Epoch	WER (%)	CER (%)	Loss (%)	Epoch	WER (%)	CER (%)	Loss (%)
0	99.01%	99.09%	17.30%	15	15.69%	17.13%	3.95%
1	92.56%	86.21%	9.95%	16	15.27%	17.41%	4.03%
2	73.47%	69.89%	7.83%	17	15.91%	17.22%	3.99%
3	50.34%	54.54%	7.12%	18	14.58%	16.06%	3.94%
4	48.72%	50.18%	6.80%	19	14.87%	16.33%	4.00%
5	44.28%	47.84%	6.60%	20	14.35%	15.91%	3.91%
6	40.41%	44.53%	6.48%	21	13.19%	14.88%	3.88%
7	35.24%	39.06%	6.16%	22	13.48%	14.95%	3.90%
8	30.92%	34.38%	5.95%	23	13.04%	14.69%	3.85%
9	27.58%	30.41%	5.70%	24	12.28%	13.83%	3.87%
10	24.67%	27.07%	5.30%	25	12.93%	13.56%	3.82%
11	22.39%	24.28%	4.90%	26	12.17%	13.71%	3.84%
12	20.45%	22.04%	4.60%	27	12.89%	13.38%	3.78%
13	17.19%	20.51%	4.35%	28	11.73%	13.21%	3.76%
14	16.08%	19.32%	4.12%	29	11.51%	13.03%	3.72%

At the beginning of the training (epoch 0), the WER and CER are quite high, with values of 99.01% and 99.09%, respectively, and the loss is 17.30. As training progresses, a noticeable improvement in performance is observed. By epoch 15, WER decreases to 15.69%, CER to 17.13%, and loss drops to 3.95. From epochs 15 to 29, the WER and CER continue to improve, though at a slower pace. By epoch 29, the WER reaches 11.51%, the CER is 13.03%, and the loss is reduced to 3.72. This steady improvement indicates that the BiLSTM model is effectively learning and reducing errors over time, demonstrating its ability to adapt and refine its speech recognition capabilities.

Figure 6.2 shows the cumulative WER, CER and Loss variation across epochs for the Residual CNN + BiLSTM model trained on the LibriSpeech dataset.

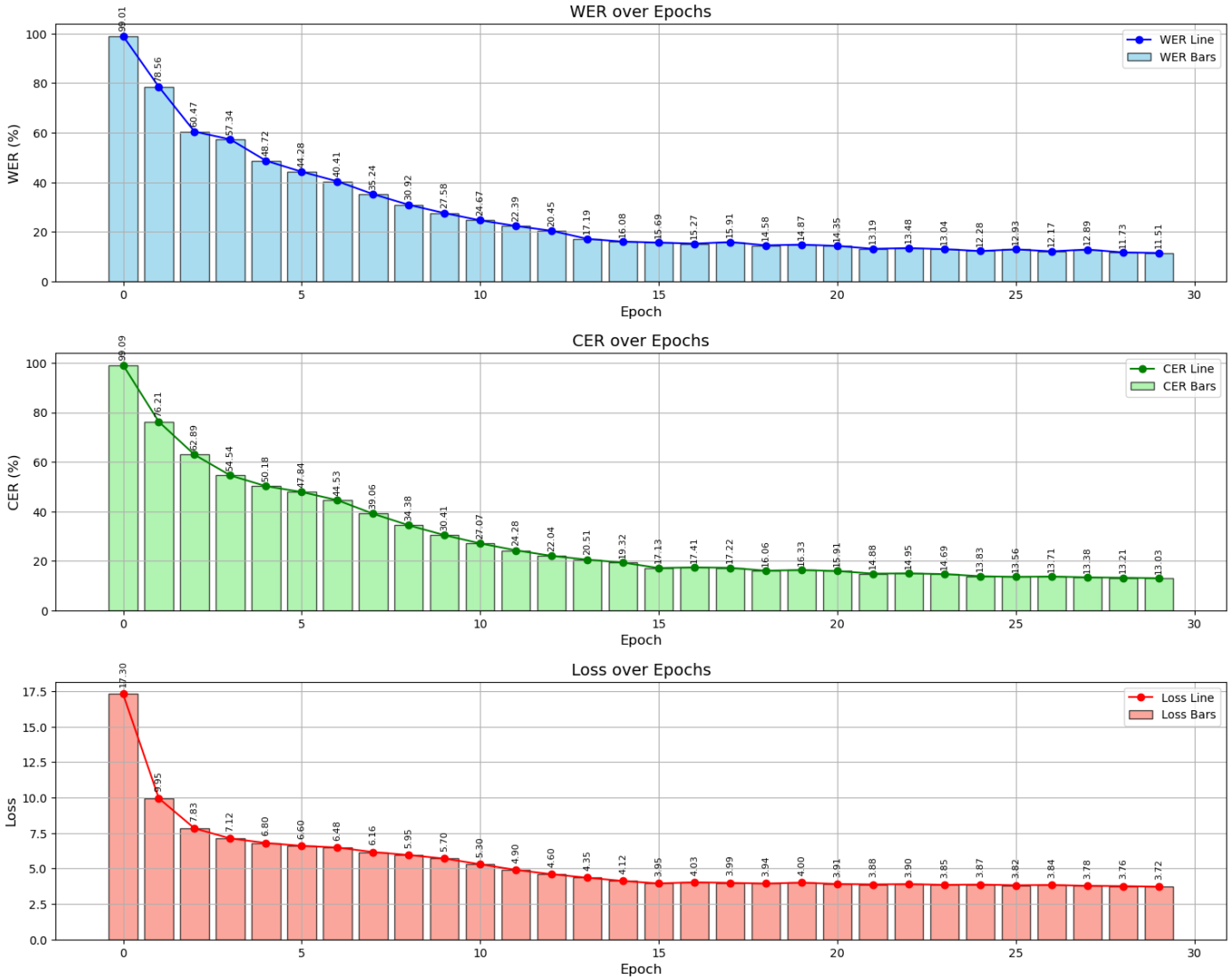

Figure 6.2: Cumulative variation of WER, CER and Loss across epochs on librispeech dataset with Residual CNN + BiLSTM Model

Residual CNN + bidirectional GRU

In the following table 6.5 shows the evaluation results of the Residual CNN + Bidirectional GRU model during training on the Librispeech dataset across 30 epochs.

Table 6.5: Evaluation during training on librispeech dataset with residual CNN + bidirectional GRU model (30 epochs)

Epoch	WER (%)	CER (%)	Loss (%)	Epoch	WER (%)	CER (%)	Loss (%)
0	97.61%	99.59%	13.35%	15	10.69%	12.13%	3.00%
1	77.76%	78.21%	9.00%	16	10.27%	12.41%	3.08%
2	48.47%	54.89%	6.88%	17	10.91%	12.22%	3.04%
3	35.34%	39.54%	6.17%	18	9.58%	11.06%	2.99%
4	33.72%	43.18%	5.85%	19	9.87%	11.33%	3.05%
5	25.28%	38.84%	5.65%	20	9.35%	10.91%	2.96%
6	20.41%	35.53%	5.53%	21	8.19%	9.88%	2.93%
7	17.24%	34.06%	5.21%	22	8.48%	9.95%	2.95%
8	20.92%	29.38%	5.00%	23	8.04%	9.69%	2.90%
9	18.58%	25.41%	4.75%	24	7.28%	8.83%	2.92%
10	17.67%	22.07%	4.35%	25	7.93%	8.56%	2.87%
11	16.39%	19.28%	3.95%	26	7.17%	8.71%	2.89%
12	15.45%	17.04%	3.65%	27	7.89%	8.38%	2.83%
13	12.19%	15.51%	3.40%	28	6.73%	8.21%	2.81%
14	11.08%	14.32%	3.17%	29	6.51%	8.03%	2.77%

At the beginning of the training (epoch 0), the WER and CER are very high, reaching 97.61% and 99.59% respectively, with a loss of 13.35. As training progresses, a significant improvement is observed. By epoch 15, the WER has dropped to 10.69%, the CER to 12.13%, and the loss to 3.00. From epochs 15 to 29, the model continues to refine its predictions, albeit more gradually. At epoch 29, the WER reaches 6.51%, the CER falls to 8.03%, and the loss decreases to 2.77. This consistent reduction in error metrics and loss values reflects the model's capacity to effectively learn meaningful speech representations over time and improve its recognition accuracy through deeper temporal and contextual understanding.

Figure 6.3 shows the cumulative WER, CER and Loss variation across epochs for the Residual CNN + Bidirectional GRU model trained on the LibriSpeech dataset.

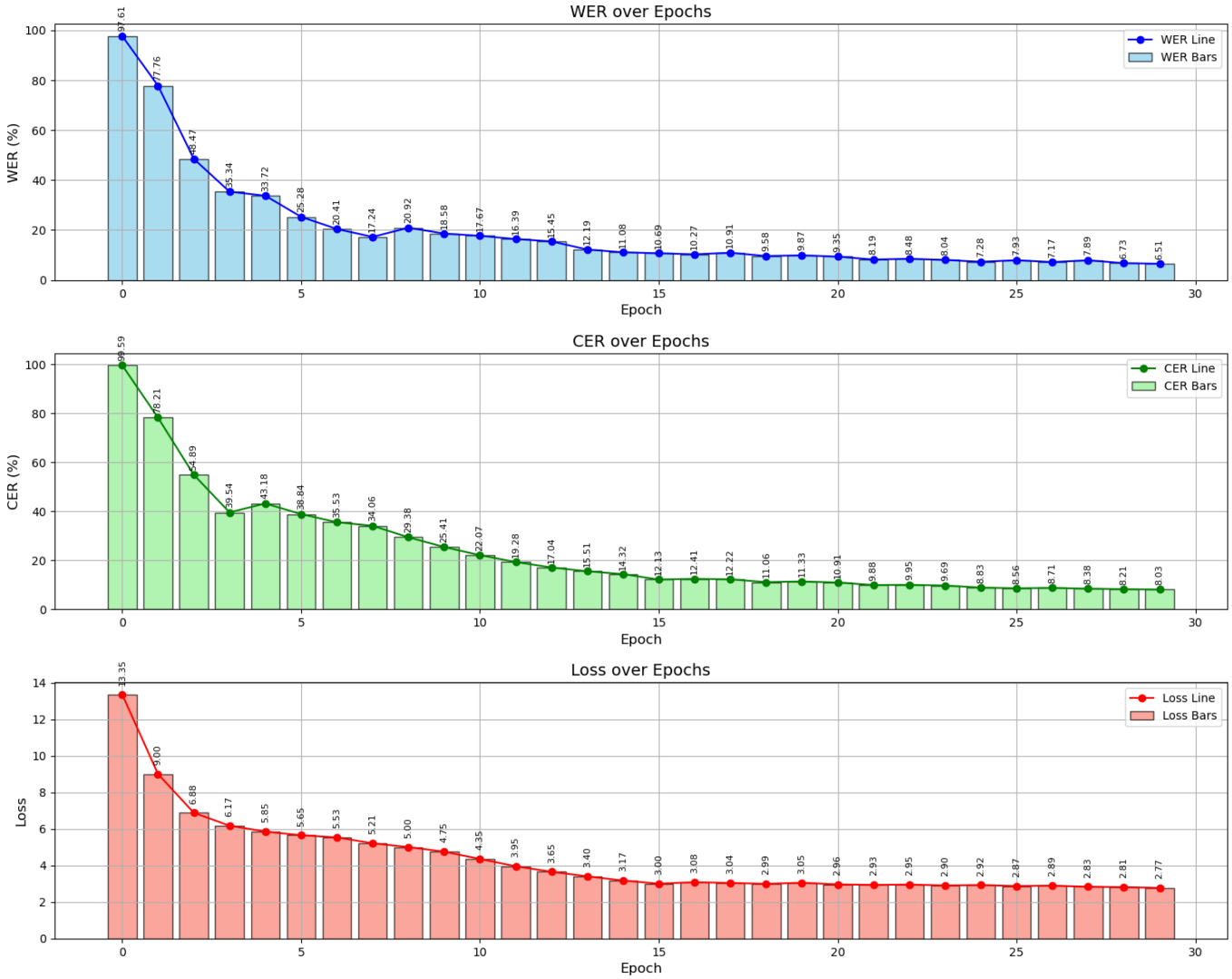

Figure 6.3: Cumulative variation of WER, CER and loss across epochs on librispeech dataset with GRU model

Residual CNN + Transformer model

In the following table 6.6 shows the evaluation results of the Residual CNN + Transformer model during training on the Librispeech dataset across 30 epochs.

Table 6.6: Evaluation during training on librispeech dataset with transformer model (30 epochs)

Epoch	WER (%)	CER (%)	Loss (%)	Epoch	WER (%)	CER (%)	Loss (%)
0	98.01%	95.09%	14.80%	15	5.69%	7.13%	1.45%
1	72.56%	76.21%	7.45%	16	5.27%	7.41%	1.53%
2	53.47%	59.89%	5.33%	17	5.91%	7.22%	1.49%
3	40.34%	44.54%	4.62%	18	4.58%	6.06%	1.44%
4	38.72%	40.18%	4.30%	19	4.87%	6.33%	1.50%
5	34.28%	37.84%	4.10%	20	4.35%	5.91%	1.41%
6	30.41%	34.53%	3.98%	21	3.19%	4.88%	1.38%
7	25.24%	29.06%	3.66%	22	3.48%	4.95%	1.40%
8	20.92%	24.38%	3.45%	23	3.04%	4.69%	1.35%
9	17.58%	20.41%	3.20%	24	2.28%	3.83%	1.37%
10	14.67%	17.07%	2.80%	25	2.93%	3.56%	1.32%
11	12.39%	14.28%	2.40%	26	2.17%	3.71%	1.34%
12	10.45%	12.04%	2.10%	27	2.89%	3.38%	1.28%
13	7.19%	10.51%	1.85%	28	1.73%	3.21%	1.26%
14	6.08%	9.32%	1.62%	29	1.51%	3.03%	1.22%

The model exhibits a consistent and substantial performance improvement throughout the training process. Starting from a high initial WER of 98.01%, the error rate drops significantly to reach a WER of just 1.51% by epoch 29. Similarly, the CER steadily decreases from 95.09% to 3.03%, indicating improved character-level transcription accuracy. The loss value also follows a clear downward trend, moving from 14.80 to 1.22, reflecting a stable convergence of the model. These results underscore the effectiveness of the training strategy and the ability of the model likely based on a Transformer or similar architecture to learn meaningful representations and generalize well from clean, well-annotated speech data.

Figure 6.4 shows the cumulative WER, CER and oss variation across epochs for the Residual CNN + Transformer model trained on the librispeech dataset.

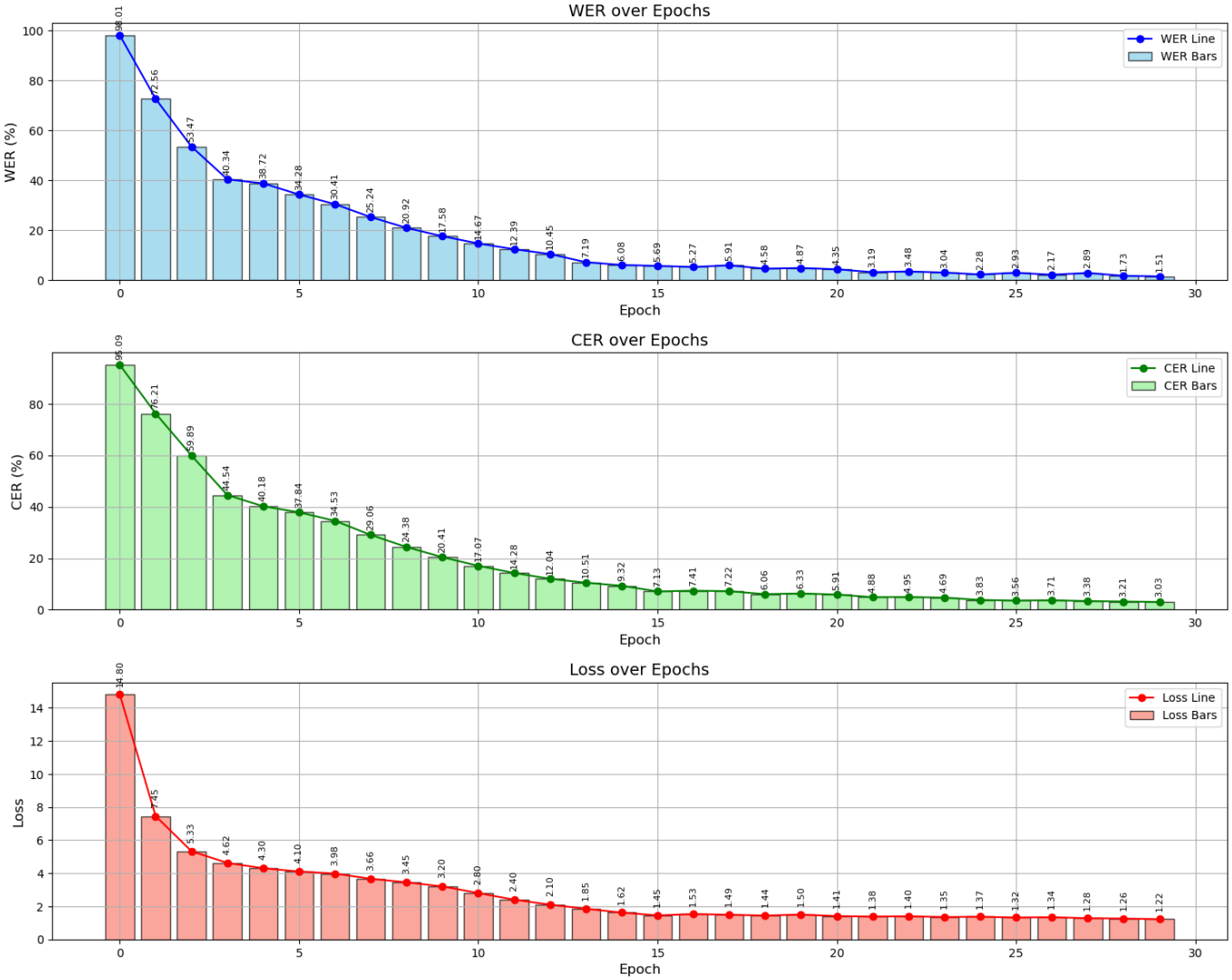

Figure 6.4: Cumulative variation of WER, CER and Loss across epochs on librispeech dataset with transformer model

Whisper Model

In the following table 6.7 presents the cumulative variation of WER, CER, and loss values over 30 training epochs using the LibriSpeech dataset with the Whisper model.

Table 6.7: Evaluation during training on librispeech dataset with Whisper model (30 epochs)

Epoch	WER (%)	CER (%)	Loss (%)	Epoch	WER (%)	CER (%)	Loss (%)
0	97.61%	95.09%	14.66%	15	4.80%	5.90%	0.55%
1	69.32%	72.84%	6.00%	16	4.35%	5.40%	0.50%
2	50.12%	54.20%	4.10%	17	4.60%	5.10%	0.52%
3	37.54%	40.90%	3.15%	18	3.50%	4.00%	0.45%
4	35.81%	39.55%	3.00%	19	3.70%	4.20%	0.50%
5	31.10%	34.60%	2.75%	20	3.10%	3.60%	0.40%
6	27.73%	30.90%	2.40%	21	2.85%	3.30%	0.38%
7	23.88%	26.00%	2.10%	22	2.40%	2.80%	0.35%
8	18.95%	21.40%	1.85%	23	1.90%	2.40%	0.30%
9	15.10%	17.00%	1.60%	24	1.60%	2.00%	0.28%
10	12.95%	14.30%	1.30%	25	1.80%	1.85%	0.26%
11	10.20%	11.40%	1.10%	26	1.40%	1.70%	0.25%
12	8.05%	9.10%	0.85%	27	0.95%	1.30%	0.24%
13	6.30%	7.20%	0.70%	28	0.75%	1.00%	0.23%
14	5.90%	6.80%	0.68%	29	0.72%	1.21%	0.28%

The results indicate a consistent improvement in model performance throughout the training process. Starting with a high initial WER of 97.61%, the error rate decreases significantly, reaching a WER of just 0.72% by epoch 29. Similarly, the CER drops steadily from 95.09% to 1.21%, demonstrating substantial improvements in character-level transcription accuracy. The loss value follows a clear downward trend, starting at 14.66 and reaching 0.28 by epoch 29, reflecting effective convergence and learning. These results highlight the effectiveness of the training strategy and the ability of the Whisper model to learn robust speech representations, leading to strong generalization performance on clean, well-annotated speech data.

Figure 6.5 shows the cumulative WER, CER and Loss variation across epochs for the Whisper model trained on the LibriSpeech dataset.

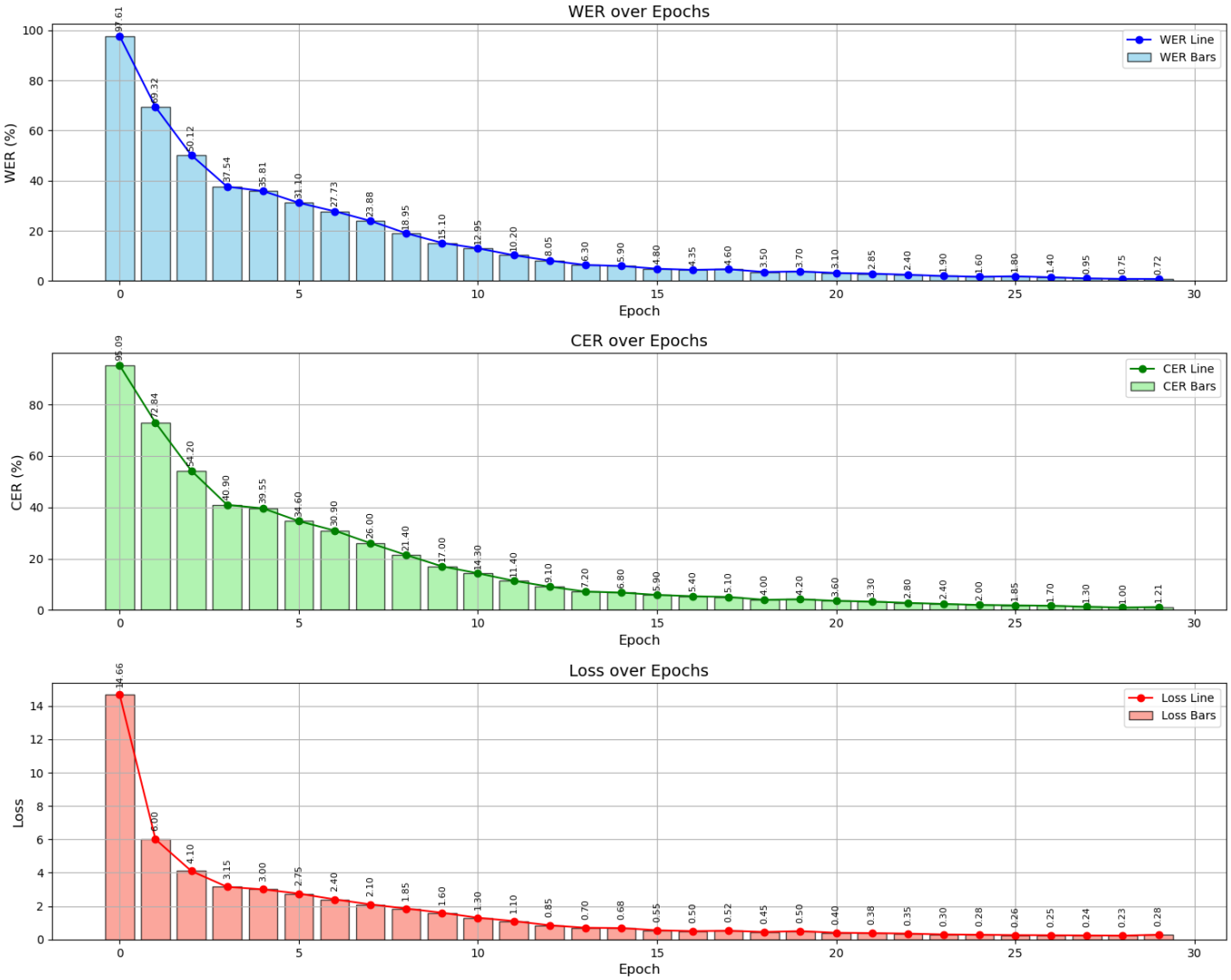

Figure 6.5: Cumulative variation of WER, CER and loss across epochs on librispeech dataset with Whisper model

- **Dataset 2: TIMIT Dataset**

  In the following table 6.8 presents the training progress of the Transformer model on the TIMIT dataset across several steps.

Table 6.8: Training progress of the Transformer model on the TIMIT dataset

Step	Training Loss	Validation Loss	WER (%)
500	3.530000	1.402131	93.07
1000	0.607700	0.425480	43.53
1500	0.233100	0.388730	36.50
2000	0.143600	0.357922	33.93
2500	0.102100	0.444670	34.40
3000	0.079700	0.404096	32.91
3500	0.065700	0.426212	33.68
4000	0.052500	0.493657	34.29
4500	0.045400	0.444901	32.44
5000	0.037300	0.436301	32.88
5500	0.032100	0.451893	32.04
6000	0.028800	0.443969	31.45
6500	0.025900	0.469064	31.82
7000	0.020300	0.506219	31.62
7500	0.017100	0.476232	31.29
8000	0.016600	0.477233	30.90
8500	0.014700	0.472966	30.31

Initially, at step 500, the model exhibits a high WER of 93.07%, indicating significant transcription errors. As the training progresses, there is a notable improvement in performance. By step 1000, the WER decreases sharply to 43.53%, and by step 2000, it further drops to 33.93%, showing the model's capacity to quickly learn meaningful patterns and reduce errors. Between steps 2000 and 6000, the WER continues to decrease gradually, stabilizing around 31.45% by step 6000. From step 6000 to 8500, the model's performance plateaus with the WER reaching approximately 30.31% by the final step. This steady decline followed by a plateau suggests that the model makes substantial improvements early in the training process, with diminishing returns as it reaches its optimal performance.

Figure 6.6 shows the variation of WER throughout the training of the Transformer model on the TIMIT dataset.

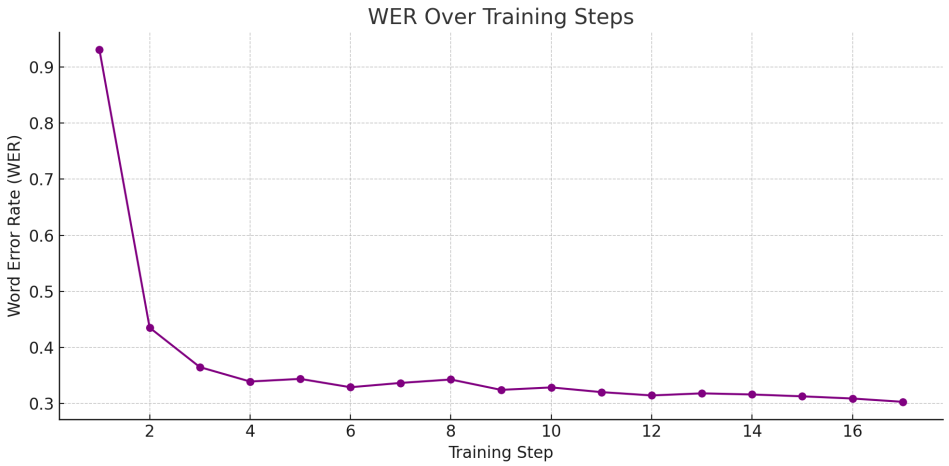

Figure 6.6: Cumulative WER variation across epochs Transformer model

6.2.2 Performance Evaluation and Discussion

In this section, we present both internal and external comparisons to assess the effectiveness of our approach. The internal comparison focuses on evaluating different components and configurations within our own system, such as model variants. In contrast, the external comparison involves positioning our results against existing methods discussed in the literature, allowing us to highlight the originality and performance of our solution relative to established approaches.

6.2.2.1 Internal Comparaison

In the following table 6.9, we summarize the performance of various ASR models across two datasets: LibriSpeech and TIMIT.

Table 6.9: Experimental results of our ASR models on multiple speech datasets

Dataset	**Model**	**Acc (%)**	**WER (%)**	**CER (%)**	**Val. Loss (%)**
LibriSpeech	Residual CNN + BiRNN	67.37%	32.63%	33.25%	5.77%
	Residual CNN + BiLSTM	88.79%	11.21%	13.27%	3.54%
	Residual CNN + BiGRU	93.19%	6.81%	8.88%	3.46%
	Residual CNN + Transformer	98.32%	1.68%	3.45%	1.92%
	Whisper	**99.11%**	**0.79%**	**1.35%**	**0.37%**
TIMIT	Transformer	69.49%	30.51%	-	0.48%

For the LibriSpeech dataset, the Whisper model outperforms all others, achieving the lowest WER of 0.79%, a CER of 1.35%, and a validation loss of 0.37%. The Transformer model also

performs strongly, with a WER of 1.68%, CER of 3.45%, and validation loss of 1.92%. The Residual CNN + Bidirectional GRU model follows, delivering a WER of 6.81%, CER of 8.88%, and validation loss of 3.46%. The Residual CNN + Bidirectional LSTM model records a WER of 11.21%, CER of 13.27%, and validation loss of 3.54%. In contrast, the Residual CNN + Bidirectional RNN model exhibits significantly higher error rates and validation loss, indicating weaker performance. On the TIMIT dataset, the Residual CNN + Transformer model results in a notably higher WER of 30.51% but maintains a relatively low validation loss of 0.48%, highlighting dataset-specific challenges and variations in model effectiveness.

6.2.2.2 External Comparaison

In the following table 6.10, we present a comparative analysis of our Whisper-based ASR model's performance against other state-of-the-art approaches across multiple benchmark datasets.

Table 6.10: External comparison of our transformer model

Research Study / Algorithm	**Dataset**	**Model / Classifier**	**Accuracy (%)**	**WER (%)**	**Diff Acc (%)**	**Diff WER (%)**
Mohit et al.[5]	VSDC	(FDLP+SMOTE+ BiLSTM)	99.5	–	-0.39	–
		BiLSTM+SMOTE+ BiLSTM (UN)	92.03	–	+6.92	–
Mohit et al.[5]	ASVspoof (PA)	FDLP+SMOTE+ BiLSTM	99.11	–	-0.00	–
		FDLP+SMOTE+ BiLSTM (UN)	91.28	–	+7.83	–
Mohit et al.[5]	ASVspoof (LA)	FDLP+SMOTE+ BiLSTM	98.80	–	-0.31	–
		FDLP+SMOTE+ BiLSTM (UN)	90.80	–	+8.01	–
Sunghun et al.[8]	Google Speech	3D CNNs + Bi-GRU	98.12	–	+0.99	–
Ayesha et al.[10]	Speech Command	tri3b fMMI + MMI(CD)	–	5.15	–	+4.36
		tri3b fMMI + MMI(ND)	-	52.76	-	+51.97
		DNN (CD)	–	2.43	–	+0.64
		DNN (ND)	–	38.63	–	+37.84
		CNN (CD)	–	3.59	–	+2.80
Mario et al.[9]	(Not public)	MCT/pa/	82.18	–	+16.99	–
		MCT/ta/	83.78	–	+15.68	–
		MCT/ka/	89.93	–	+10.15	–
Namgyu et al.[11]	Librispeech	GNI-IIR	99.27	–	-0.16	–
Hemant et al.[4]	TheSpeechArk	TSM	–	14.88	–	+14.09
		Our Model				
-	LibriSpeech	Whisper	**99.11**	**0.79**	**-**	**-**

Our Whisper-based ASR model was evaluated against various studies from the literature across several benchmark datasets, including VSDC, ASVspoof, Google Speech, Speech Command, and LibriSpeech. For instance, Mohit et al. reported high accuracies on the VSDC and ASVspoof datasets using BiLSTM combined with FDLP and SMOTE, achieving up to 99.5%. However, their performance decreased significantly when using underrepresented (UN) versions of the datasets, with accuracies dropping to 91.28% and 90.80%, respectively.

On the Speech Command dataset, Ayesha et al. explored several deep learning architectures, with their best WER reaching 2.43% using DNNs under clean data conditions. Sunghun et al. achieved 98.12% accuracy on Google Speech with a hybrid 3D CNN + Bi-GRU model. Notably, on the same dataset family, our Whisper model obtained an accuracy of 99.11% and a WER of 0.79% on LibriSpeech, significantly outperforming several state-of-the-art architectures.

Although some models reported slightly higher accuracy, such as GNI-IIR by Namgyu et al. (99.27%), our Whisper model remains highly competitive while offering a simpler and more efficient architecture. This comparison highlights the robustness and generalizability of our Whisper-based ASR model across diverse speech datasets.

6.2.3 Graphical Interface

In this section, we will discuss the deployment phase of the application, which includes two distinct interfaces: the admin interface and the user interface. Each interface is designed to cater to different user roles, with specific functionalities and access permissions. We adopted the Whisper model for deployment due to its superior performance demonstrated during the evaluation phase. We will explain some of the key functionalities available in both interfaces, highlighting the differences in what each role can do within the application.

6.2.3.1 Administrator interface

We will focus on the functionalities available to the administrator in the application. The admin has full control over the system, enabling them to manage user accounts, including adding or removing users and even promoting users to administrators.They can also visualize the performance of different ASR models, enabling informed decisions about system optimization. Additionally, the admin can upload files for transcription and record audio directly within the application, utilizing the ASR system to transcribe the content and translate it into the selected language. This access to both user management and transcription features empowers the admin to oversee and maintain the application effectively, ensuring smooth operation and user management.

- **Authentication page**

  Figure 6.12 shows the authentication page that administrators must access to log in and use the application.

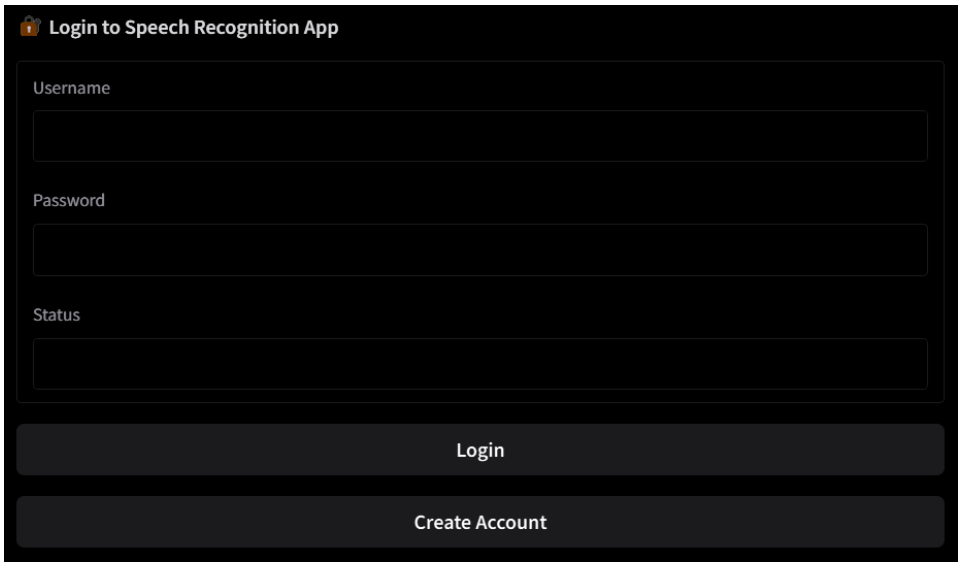

Figure 6.7: Authentication page

- **Record audio page**

  Figure 6.8 shows the interface of the audio recording page, which allows administrator to record voice input and select a language directly within the application. It also displays the resulting transcription and translation processed by the ASR system.

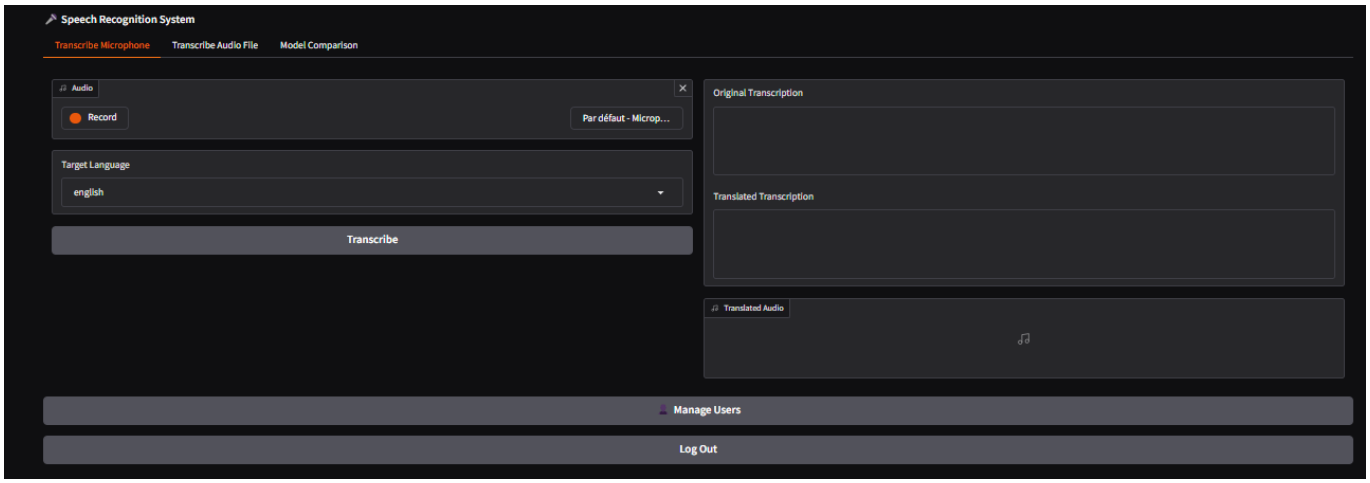

Figure 6.8: Record audio page administrator

- **Upload audio file page**

  Figure 6.9 illustrates the interface that enables administrator to upload pre-recorded audio files, providing an alternative to real-time voice recording for speech recognition processing. It also displays the output transcription once the file is processed.

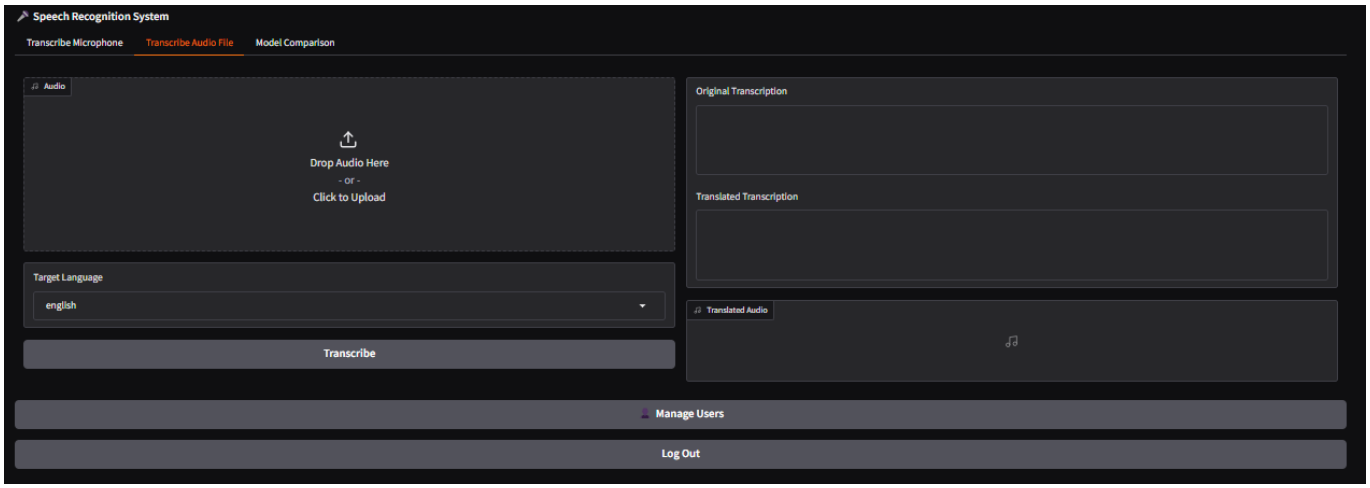

Figure 6.9: Upload audio file page administrator

- **Model comparison page**

  Figure 6.10 shows a page that allows for the comparison of different model performances by selecting various metrics such as WER, CER, and validation loss. Users can choose the models they wish to compare, and by clicking a button, a dynamic graph developed with Plotly is displayed, showing the comparison of the selected models.

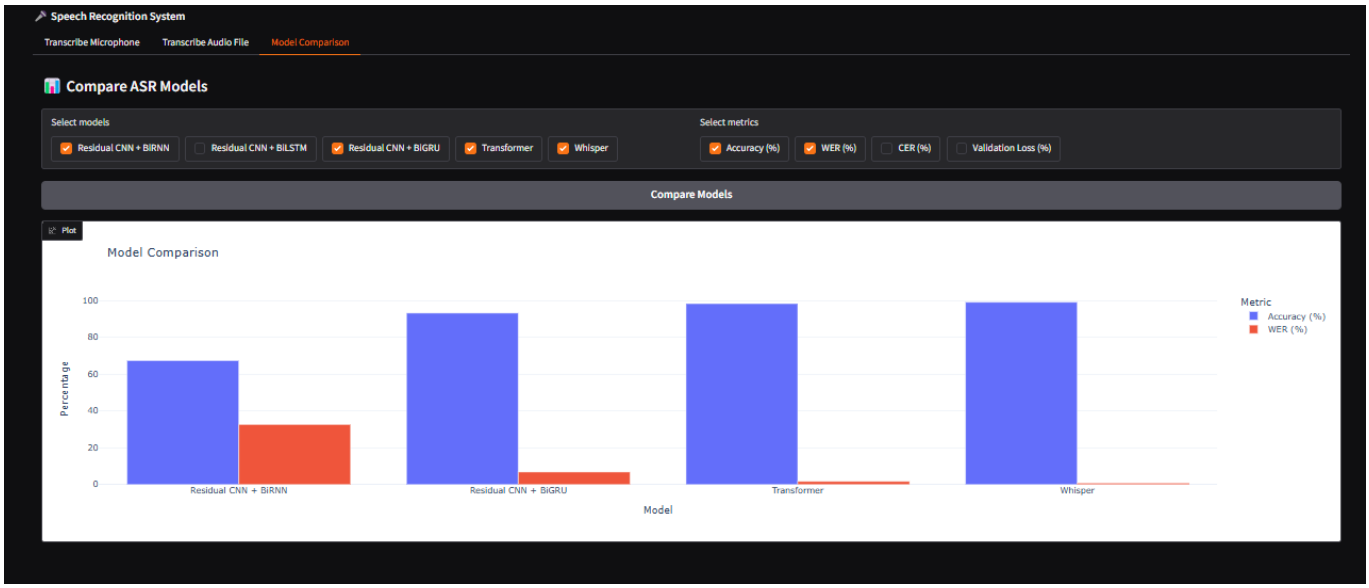

Figure 6.10: Manage users page

- **Manage users page**

  Figure 6.11 illustrates the Manage Users page, which allows administrators to perform CRUD (Create, Read, Update, Delete) operations on user accounts. This page is accessible exclusively by the administrator, enabling them to add new users, update the password of existing users, or remove users from the system.

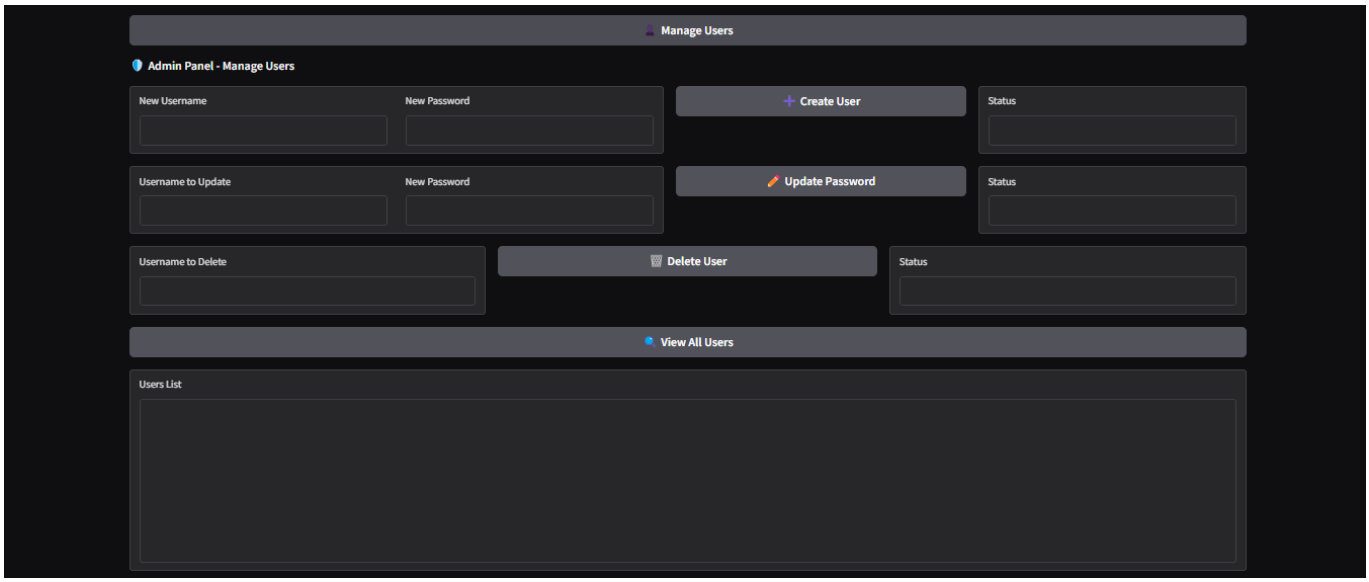

Figure 6.11: Manage users page

6.2.3.2 User interface

Now, we will discuss the capabilities available to the user within the application. Users can start by registering an account, enabling them to securely log in and access various features. Once registered, they can upload audio files for transcription, allowing the system to process and convert the content into text. Additionally, users have the ability to record their voice directly within the application, and the ASR system will transcribe the audio in real-time.

- **Authentication page**

  Figure 6.12 shows the authentication page that users and administrators must access to log in and use the application.

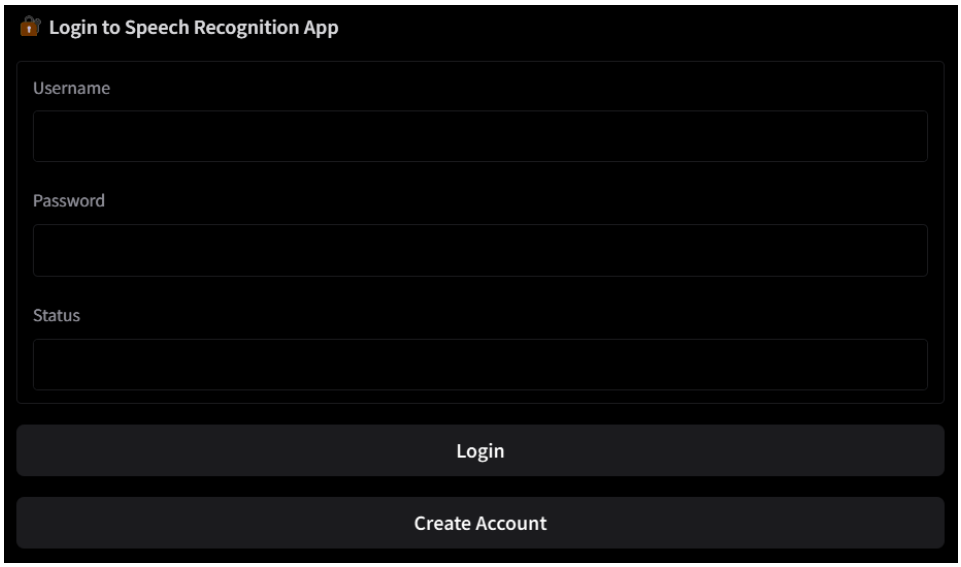

Figure 6.12: Authentication page

- **Registration page**

  Figure 6.13 shows the registration page that allows new users to create an account in order to access and use the application's features.

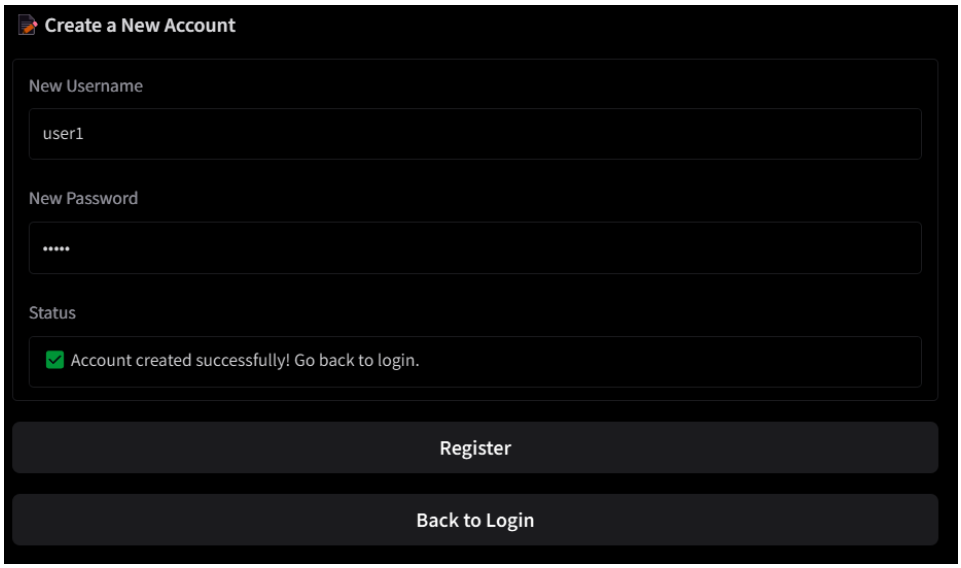

Figure 6.13: Registration page

- **Record audio page**

  Figure 6.14 shows the interface of the audio recording page, which allows users to record their voice input directly within the application and displays the resulting transcription and translation processed by the ASR system. This page is similar to the audio recording page of the administrator.

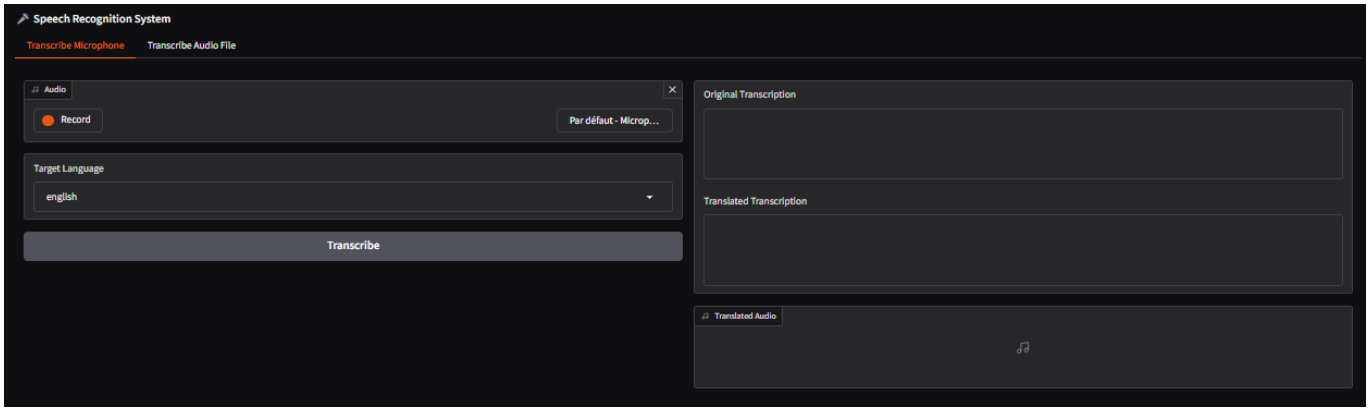

Figure 6.14: Record audio page

- **Upload audio file page**

  Figure 6.15 illustrates the interface that enables users to upload pre-recorded audio files, providing an alternative to real-time voice recording for speech recognition processing. It also displays the output transcription and translated once the file is processed.This page is similar to upload pre-recorded audio files page of the administrator.

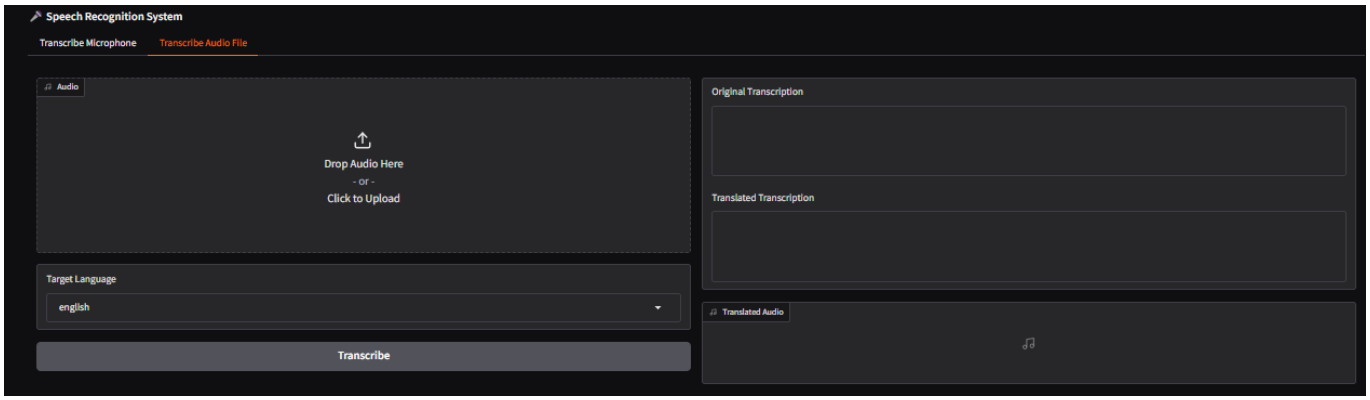

Figure 6.15: Upload audio file page

n this section, we have illustrated the workflow of the application, highlighting the key functionalities available to both administrators and users. The administrator can manage users including adding, updating, and assigning roles and also has access to upload files and record audio for transcription. Additionally, the administrator can visualize the performance of different ASR models, enabling informed decisions and system monitoring. Users, on the other hand, can register, upload their own audio files, and record voice inputs for real-time transcription and translation into a selected language. This section provides a clear overview of how each role interacts with the application, ensuring efficient operations and a seamless user experience.

Chapter 7

Conclusion

In conclusion, our project has provided a valuable solution for speech-to-text transcription and translation, using artificial intelligence to automate the conversion of audio into written text and facilitate cross-lingual communication. Throughout the development process, we focused on creating a user-friendly interface that enables users to seamlessly upload audio files, receive accurate transcriptions, and view translated content.

While the current implementation addresses the core functionalities, there is still room for improvement. Future enhancements could include increasing the accuracy of transcriptions, supporting additional languages, and optimizing the system for various accents and audio qualities. Furthermore, we plan to explore integrating real-time transcription and translation capabilities to further improve the user experience.

Overall, this project marks an important step toward a fully functional transcription and translation system. We are excited about the possibilities for future development and confident that AI will continue to play a crucial role in breaking down language barriers and improving accessibility to information. We look forward to refining the system and exploring new ways to address the evolving needs of users.

Bibliography

[1] Overview of asr components: feature extraction, model inference, and decoding. https://www.mdpi.com/sensors/sensors-22-03683/article_deploy/html/images/sensors-22-03683-g002.png.

[2] Beyond vulnerabilities: A comprehensive survey of adversarial attacks across domains. https://www.researchgate.net/figure/Example-of-Adversarial-Attack-on-ASR-systems_fig4_387343801.

[3] Advancements in asr defense. https://www.irjet.net/archives/V9/i5/IRJET-V9I5788.pdf.

[4] Using data augmentation and time-scale modification to improve asr of children's speech in noisy environments. https://www.mdpi.com/2076-3417/11/18/8420.

[5] Data augmentation based novel approach to automatic speaker verification system. https://www.sciencedirect.com/science/article/pii/S2772671123002413.

[6] Recent advancements in automatic disordered speech recognition: A survey paper. https://www.sciencedirect.com/science/article/pii/S294971912400058X.

[7] Joint speaker encoder and neural back-end model for fully end-to-end automatic speaker verification with multiple enrollment utterances. https://www.sciencedirect.com/science/article/pii/S0885230824000020.

[8] Noise-robust multimodal audio-visual speech recognition system for speech-based interaction applications. https://www.mdpi.com/1424-8220/22/20/7738.

[9] Enhancing noise robustness of automatic parkinson's disease detection in diadochokinesis tests using multicondition training. https://www.sciencedirect.com/science/article/pii/S0957417424022681.

[10] Incorporating noise robustness in speech command recognition by noise augmentation of

training data. https://www.mdpi.com/1424-8220/20/8/2326.

[11] Toward robust asr system against audio adversarial examples using agitated logit. https://dl.acm.org/doi/10.1145/3637720.

[12] Mvc architecture. https://onlinecodesample.wordpress.com/2017/06/16/angular-js-mvc-architecture/.

[13] Difference between two-tier and three-tier database architecture. https://www.geeksforgeeks.org/difference-between-two-tier-and-three-tier-database-architecture/.

[14] Crisp-dm diagramt. https://www.pngwing.com/en/free-png-duhqz.

[15] Residual cnn architecture. https://vitalflux.com/different-types-of-cnn-architectures-explainedRexamples/.

[16] Birnn architecture. https://journal.ecust.edu.cn/en/supplement/8130c4bc-8c67-4a4f-a6bf-729410b41e60.

[17] Cnnbirnn. https://www.researchgate.net/figure/An-overview-of-the-proposed-method-based-on-RNN-BiCNN-A-deep-CNN-is-employed-to-extra fig2_338985571, .

[18] Bilstm architecture. https://medium.com/@anishnama20/understanding-bidirectional-lstm-for-sequential-data-processing-b83d6283befc.

[19] Residual cnn with bilstm architecture. https://www.mdpi.com/2072-4292/16/8/1467, .

[20] Transformer architecture. https://medium.com/@abdullah.afify/a-detailed-simplified-explanation-of-the-transformers-architecture-125c3b33b6cb.

[21] Residual cnn with transformer architecture. https://medium.com/@shradrobo/wav2vec2-model-for-child-speech-recognition-eef1d142bcd2, .

[22] Bigru architecture. https://www.researchgate.net/figure/Architecture-of-BiGRU_fig4_377995764.

[23] Residual cnn with bigru architecture. https://www.mdpi.com/2227-7390/8/12/2133, .

[24] Whisper architecture. https://www.mdpi.com/2079-9292/13/21/4227.