Cairo University, Faculty of Computers and Artificial Intelligence

FACULTY OF COMPUTERS AND ARTIFICAL INTELLIGENCE, CAIRO UNIVERSITY

AI498: Graduation Project
Summer 2025
Second Semester

Arabic Lip-Reader

Project Supervisor:

Dr. Eman Ahmed

Done By:

Hazem Khaled Sabry Kamel (20210111)
11410120210111@stud.cu.edu.eg

Abdelrahman Ahmed Adel Abdelaziz (20210194)
11410120210194@stud.cu.edu.eg

Abdelrahman Wael Mohamed Ashour (20210231)
11410120210231@stud.cu.edu.eg

Essa Hassan Ahmed Ramzy (20210281)
11410120210281@stud.cu.edu.eg

Eslam Mohamed Abdelazim Ali (20211013)
11410120211013@stud.cu.edu.eg

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
VSR	Visual Speech Recognition
CTC	Connectionist Temporal Classification
Conformer	Convolution-Augmented Transformer
DC-TCN	Densely Connected Temporal Convolutional Network
MS-TCN	Multi-Scale Temporal Convolutional Network
MB-TCN	Multi-Branch Temporal Convolutional Network
ResNet	Residual Network
CNN	Convolutional Neural Network
ROI	Region Of Interest
CER/WER	Character/Word Error Rate
LRW-AR	Lip Reading in the Wild - Arabic
LRS	Lip Reading Sentences
SOS	Start-of-Sequence
EOS	End-of-Sequence
GPU	Graphics Processing Unit

Chapter 1: Introduction

Visual Speech Recognition (VSR), commonly known as lip-reading, is a rapidly advancing field at the intersection of computer vision and artificial intelligence. It involves the task of interpreting spoken language from visual information, primarily the movement of a speaker's lips, without any reliance on audio signals. While early VSR systems focused on recognizing isolated words, recent breakthroughs in deep learning have made it possible to tackle the more complex challenge of transcribing continuous, sentence-level speech. This project is dedicated to advancing this technology specifically for the Arabic language.

1.1 Problem Definition

The ability to automatically transcribe speech from video has seen remarkable progress, with models like those developed by the MPC001 team for the auto_avsr project achieving high accuracy on large-scale English datasets (LRS2, LRS3, VoxCeleb2). However, this success has not been uniformly distributed across all languages. The Arabic language, with its unique phonetic and visemic characteristics, remains significantly underserved.

There is a distinct lack of high-performance, publicly available systems designed for continuous, sentence-level Arabic lip-reading. This gap is caused by two primary factors:

- Data Scarcity: A shortage of large, well-annotated video datasets of Arabic speakers at the sentence level.
- Linguistic Complexity: The visual ambiguity of many Arabic phonemes and the effects of coarticulation present unique challenges that models trained on English may not inherently handle.

This project addresses the problem of developing an effective end-to-end system capable of deciphering full sentences in Arabic from silent video, bridging a critical gap in assistive and communication technology.

1.2 Objective

The primary objective of this project is to design, implement, and rigorously evaluate a deep learning model capable of transcribing continuous, sentence-level spoken Arabic from silent video footage of a speaker's lips.

To achieve this, the following sub-objectives are defined:

- To curate and meticulously refine a specialized dataset suitable for training a sentence-level Arabic VSR model.
- To adapt a state-of-the-art visual speech recognition architecture, inspired by leading models in the field, to the specific requirements of the Arabic language.
- To leverage transfer learning by fine-tuning a model pre-trained on large-scale English datasets, enabling it to learn Arabic-specific visemes more effectively.
- To systematically evaluate the model's performance using standard industry metrics, such as Word Error Rate (WER) and Character Error Rate (CER).
- To develop a proof-of-concept user interface to demonstrate the practical application of the lip-reading system.

1.3 Motivation

The development of an accurate Arabic lip-reading system is driven by its potential to create significant societal and technological impact. The primary motivations include:

- Enhancing Accessibility: The system can serve as a vital communication aid for the deaf and hard-of-hearing community, as well as for individuals with speech impediments or those who have lost their voice due to medical conditions such as laryngectomy.
- Enabling Communication in Noisy Environments: In settings where audio is corrupted or overwhelmed by noise—such as in factories, on public transit, or during military operations—a VSR system can enable reliable communication and dictation.
- Improving Security and Privacy: It allows for silent dictation in public spaces, ensuring the privacy of sensitive information. It can also be used in forensic analysis to decipher speech from video recordings where the audio is inaudible or was never recorded.
- Advancing Arabic AI: This project contributes directly to the field of Arabic artificial intelligence, addressing the need for more sophisticated tools and models for this major world language.

1.4 Challenges and Limitations

While the potential is great, the task of automatic lip-reading is inherently challenging and this project operates under specific limitations.

Challenges:

- Visual Ambiguity (Homophemes): The core challenge of VSR is that many distinct sounds (phonemes) look visually identical when spoken (visemes). For example, in Arabic, the sounds for 'ba' (بَ) and 'ma' (مَ) can appear very similar, forcing the model to rely heavily on context to disambiguate them.
- Coarticulation: In continuous speech, the appearance of a mouth movement is influenced by the movements that precede and follow it. This makes segmenting and identifying phonemes much harder than in isolated-word recognition.
- Speaker Variability: Differences in lip shape, facial hair, speaking style, accents, and head movements across different speakers create a wide variance that the model must learn to generalize over.

Limitations:

- Dataset Scope: The performance of our model is ultimately dependent on the size and diversity of our fine-tuning dataset. While we have taken care to curate and expand our data, it cannot encompass all dialects, speakers, and vocabularies of the Arabic language.
- Controlled Environment: The model is trained on videos with relatively constrained conditions (e.g., front-facing speakers, adequate lighting). Its performance may degrade significantly with non-frontal head poses, poor lighting, or occlusions of the mouth area.
- Computational Constraints: Training large-scale VSR models is computationally intensive. The scope of our experimentation, including hyperparameter tuning and the number of architectures tested, is constrained by the available computational resources.

Chapter 2: Related Work

Paper 1:

Title: "Cross-Attention Fusion of Visual and Geometric Features for Large Vocabulary Arabic Lipreading" [1]

Problem:

- Existing lipreading methods struggle to combine visual & landmark features due to simple combination methods like concatenation that may not yield optimal feature vectors.
- A significant limitation in Arabic lipreading research is the lack of large-scale datasets.

Method:

- Proposes a cross-attention fusion network combining visual features (CNN) with geometric features (GNN) for Arabic lipreading.
- Introduces a new large-scale Arabic lipreading dataset, LRW-AR.
- LRW-AR Dataset Creation Pipeline:
  - Video Collection: Scraped Arabic-language TV content from YouTube.
  - Shot Boundary Detection: Segmented videos into shorter clips based on scene changes.
  - Face Detection and Tracking: Identified and tracked faces within each clip to isolate single-speaker segments.
  - Data Cleaning: Manually inspected and removed irrelevant videos.
  - Annotation: Used automatic speech recognition (Vosk) to transcribe audio, followed by manual correction.
  - Video Splitting: Generated clips focusing on frequently occurring Arabic words.
  - Data Cropping: Standardized and cropped videos to focus on the speaker's face.

Model Architecture:

- Visual Feature Extraction: 3D convolutional block + ResNet-18.
- Geometric Feature Extraction: Graph Neural Network (GNN) on facial landmarks.
- Feature Fusion: Multi-head attention (FusionNet) to combine visual and geometric features.
- Sequence Decoding: Multi-scale dilated Temporal Convolutional Network (MS-TCN).

Results:

- LRW-AR Dataset: Contains 20,000 videos, 100-word classes, 36 speakers.
- Achieved 85.85% accuracy on LRW-AR with FutionNet and 83.45% without geometric features (visual-only).
- Ablation Studies: Demonstrated the effectiveness of the cross-attention mechanism and the importance of facial landmarks.
- The optimal accuracy is attained when utilizing 33 landmarks (the lower half landmarks of the face from the 68 dlib landmarks).

Conclusions:

- The introduction of the LRW-AR dataset provides a valuable resource for Arabic lipreading research, addressing a significant gap in the field.
- Cross-attention is effective for fusing visual and geometric features, outperforming simpler methods.

Paper 2:

Title: "Arabic Lip Reading with Limited Data Using Deep Learning" [2]

Problem:

- The limited availability of large-annotated datasets for Arabic lipreading is a common challenge in low-resource language applications.
- Standard deep learning models often require significant amounts of training data, posing a challenge for Arabic lipreading.
- Need for Targeted Solutions, the goal is to develop a lipreading system that is robust and accurate even with limited data.

Method:

- New Dataset: A dataset of 20 common Arabic words spoken by 40 native speakers was created.
- Separate Training Approach: A CNN-based viseme classifier and a GRU-based temporal model are trained *separately* to address data limitations by exploiting redundant viseme frames.
- Hybrid CNN-GRU Model: A CNN-GRU architecture combines spatial feature extraction using CNN and temporal modeling using the GRU for end-to-end word recognition.
- Transfer Learning: The pre-trained viseme classifier is used to directly extract visual features from a new dataset, reducing the training load.

Results:

- High Accuracy: Achieved 83.02% accuracy on the newly created dataset.
- Effective Transfer Learning: The pre-trained viseme classifier improved accuracy on a separate Arabic dataset (approximately 3% increase in recognition accuracy).
- Viable Biometric Potential: Viseme shape itself shows sufficient information to uniquely identify individuals.
- CNN-GRU model reduced the number of parameters and size of the model while increasing accuracy.

Conclusions:

- Feasibility with Limited Data: The proposed approach is viable and effective in scenarios with limited Arabic lipreading data.
- Viseme Redundancy is Key: Using the redundant features of Arabic viseme frames is an asset to reducing the needed dataset size for Arabic automatic lip-reading applications.
- Biometric Applications: Viseme shape has potential for biometric identification applications.

Paper 3:

Title: "TRAINING STRATEGIES FOR IMPROVED LIP-READING" [3]

Problem:

- Isolated word lip-reading lacks comprehensive study combining recent advancements in temporal models and training strategies to maximize performance on LRW.

Method:

- Systematically investigates the impact of data augmentations (Time Masking, Mixup), temporal models (BGRU, MS-TCN, DC-TCN), self-distillation, and word boundary indicators through ablation studies. Encoder pre-training is also explored.

Results:

- Achieves SOTA accuracy on LRW (93.4%, increasing to 94.1% with pre-training on additional datasets).
- Time Masking is the most effective augmentation.
- DC-TCN outperforms MS-TCN and BGRU.
- Self-distillation and word boundary indicators provide gains.
- Improved performance stems from increased accuracy on difficult words.

Conclusions:

- Combining DC-TCN, Time Masking, Mixup, word boundaries, and self-distillation yields significant lip-reading performance gains, particularly on challenging words.

Paper 4:

Title: "AUTO-AVSR: AUDIO-VISUAL SPEECH RECOGNITION WITH AUTOMATIC LABELS" [4]

Problem:

- Accurate labeling of audio-visual datasets is time-consuming and expensive.
- Existing methods to leverage unlabeled data often involve complex two-step processes like self-supervised learning or knowledge distillation.

Method:

- Use publicly available pre-trained ASR models (e.g., Whisper, wav2vec 2.0) to automatically transcribe large unlabeled audio-visual datasets (AVSpeech, VoxCeleb2).
- Combine these automatically labeled datasets with existing labeled datasets (LRS2, LRS3) to create a large-scale training set.
- Train audio-only (ASR), video-only (VSR), and audio-visual (AV-ASR) models on this augmented dataset using a Conformer-based architecture.
- The entire pipeline uses publicly accessible models and datasets to ensure reproducibility.

Results:

- Use publicly available pre-trained ASR models (e.g., Whisper, wav2vec 2.0) to automatically transcribe large unlabeled audio-visual datasets (AVSpeech, VoxCeleb2).
- Combine these automatically labeled datasets with existing labeled datasets (LRS2, LRS3) to create a large-scale training set.
- Train audio-only (ASR), video-only (VSR), and audio-visual (AV-ASR) models on this augmented dataset using a Conformer-based architecture.
- The entire pipeline uses publicly accessible models and datasets to ensure reproducibility.

Conclusions:

- Use publicly available pre-trained ASR models (e.g., Whisper, wav2vec 2.0) to automatically transcribe large unlabeled audio-visual datasets (AVSpeech, VoxCeleb2).
- Combine these automatically labeled datasets with existing labeled datasets (LRS2, LRS3) to create a large-scale training set.
- Train audio-only (ASR), video-only (VSR), and audio-visual (AV-ASR) models on this augmented dataset using a Conformer-based architecture.
- The entire pipeline uses publicly accessible models and datasets to ensure reproducibility.

Paper 5:

Title: "Visual Lip-Reading for Quranic Arabic Alphabets and Words Using Deep Learning" [5]

Problem:

- Lack of research studies on visual speech recognition for the Arabic language in general, and its absence in the Quranic research.
- The need to build a data-driven system with more Arabic alphabet.
- To help better recognize the alphabets, it would be necessary to have systems that could correctly read the precise recitation of Quranic knowledge.

Method:

- Introduces a new Arabic lip-reading dataset (AQAND) containing 10490 videos captured from multiple viewpoints of letter-level (single letters/ Quranic disjoined letters) and word-level data based on Al-Qaida Al-Noorania book.
- Employs ResNet-18 after converting 2D convolution to 3D.
- Employs transfer learning to pre-train the CNN.

Results:

- Achieved accuracies of 83.3%, 80.5%, and 77.5% on words, disjoined letters, and single letters, respectively.
- Identified challenges such as the lip movements during the isolated letters and distinguishing between similar phonemes with different articulation points.

Conclusions:

- This approach of using the new Arabic lip-reading dataset (AQAND) and deep learning methods, specifically a CNN with ResNet-18 and transfer learning, performs well.
- The proposed study can be a new and effective approach for more accurate Quranic recitation that is more correct and authentic, therefore expanding the way we can improve learning systems.

Chapter 3: Proposed Model

This chapter provides a detailed specification of the system developed for Arabic lip-reading. We begin by describing the cornerstone of our project: the custom-curated and enhanced dataset. We then present the high-level system design, followed by an in-depth look at the various deep learning models we implemented and compared. Finally, we list the key technologies and frameworks that enabled our research and development.

3.1 Dataset

The performance of any deep learning model is fundamentally tied to the quality and scale of its training data. Recognizing the absence of a public, sentence-level dataset tailored for the phonetic nuances of Arabic, we constructed a custom corpus through a multi-stage process of curation, automated expansion, and linguistic refinement. Our methodology was designed to create a high-quality evaluation benchmark while simultaneously building a large-scale training set.

Our starting point was the LRW-AR dataset [6], which provides a rich source of raw material of Modern Standard Arabic videos. The final curated and expanded dataset developed for this project has been made publicly available [7].

3.1.1. Curated Evaluation Set (2,000 Samples)

To establish a "gold standard" for reliable model evaluation, our process began with manual curation. We filtered and selected the 2,000 best video samples from the LRW-AR dataset based on clarity, and speaker visibility. These samples underwent an intensive labeling and refinement pipeline:

1. Initial Diacritization: The transcriptions were first processed with the Farasa Diacritizer to generate a baseline of vocalized text.
2. Intensive Manual Correction: This is the most critical step. Each of the 2,000 labels was manually reviewed and corrected by a native speaker to ensure the diacritics perfectly matched the actual pronunciation in the video.
3. Phonetic and Linguistic Enhancements: During the manual review, we applied a series of linguistic rules to create a precise visual-phonetic mapping:
    - Removal of Non-Articulatory Letters: Letters that are written but not pronounced (e.g., the 'lam' in "sun letters" like الشمس) were removed from the labels.
    - Refined Pronunciation Rules: Specific pronunciation rules were enforced, such as correctly transcribing the definite article (ال) based on the following letter and representing the final sound of taa marbuta (ة).

This cleaned and phonetically-aligned 2,000-sample corpus serves as our primary evaluation and validation set, providing an accurate and reliable benchmark for model performance.

3.1.2. Automated Training Set Expansion (10,000 Samples)

To train a robust model capable of generalization, a larger volume of data was essential. We expanded our dataset by processing an additional 10,000 video samples from the LRW-AR corpus.

The audio waveforms from these videos were extracted and transcribed using Google's Gemini 2.5 Pro, a state-of-the-art multimodal LLM model with advanced audio understanding capabilities. This automated approach allowed us to efficiently generate high-quality text labels for a large volume of data. These newly labeled samples underwent the same automated refinement process (the programmatic rule application) ensuring consistency.

3.1.3. Final Dataset Composition and Training Strategy

Our final dataset consists of two key components: the 10,000 automatically labeled samples and the 2,000 manually verified ones. For training, we combined these two sets. To balance the trade-off between data quantity (from the Gemini-labeled set) and data quality (from the manually-verified set), we employed a weighted sampling strategy. This approach gives greater importance to the high-quality, phonetically precise examples from our 2,000-sample corpus during the training batches, ensuring the model is more heavily guided by accurate labels while still benefiting from the vocabulary and speaker diversity of the larger, automatically-labeled set. The held-out portion of the 2,000-sample set remained strictly for evaluation.

3.2 System Design

The overall system is designed as an end-to-end pipeline that takes a raw video file as input and produces a transcribed Arabic sentence as output. The process can be broken down into three main stages, as illustrated in Figure 3.1.

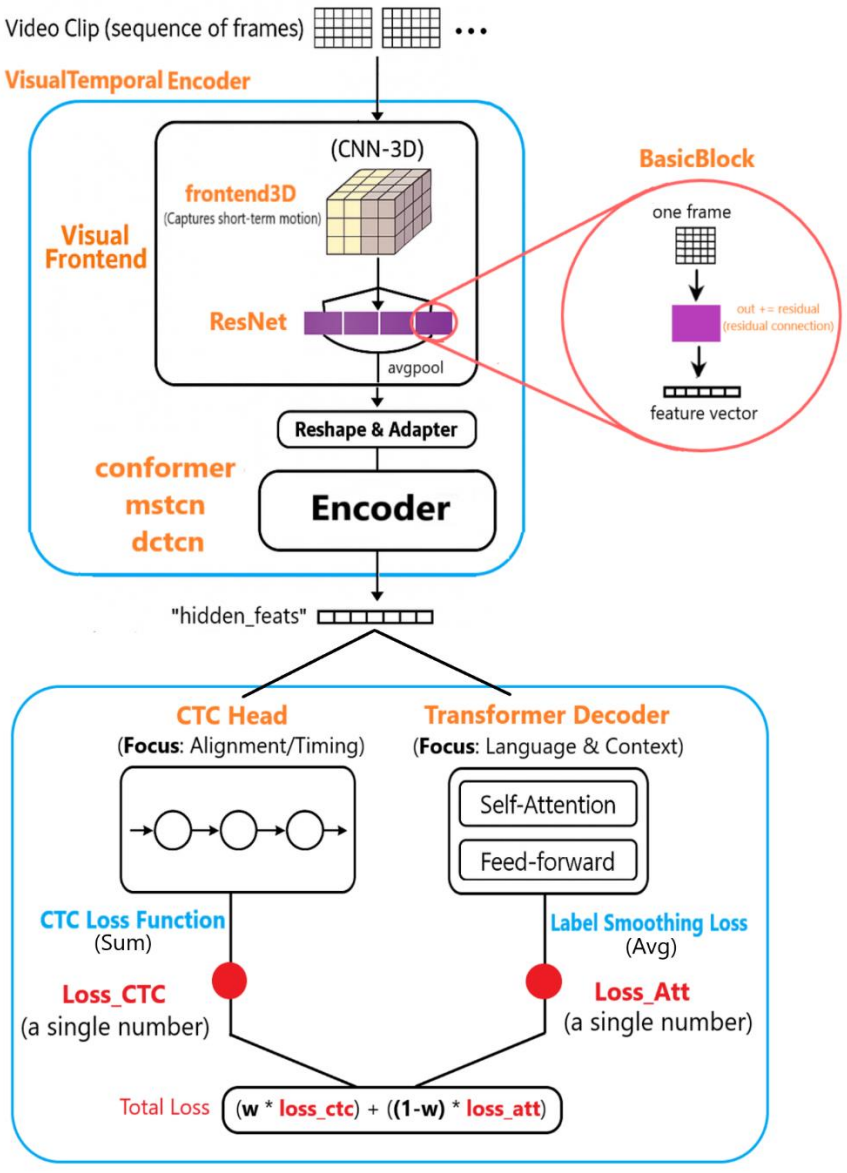

Figure 3.1: High-Level System Architecture

- Preprocessing: The input video is first processed to isolate and prepare the visual data. This involves running a face detector to locate the speaker's face in every frame. From the facial landmarks, the mouth region (Region of Interest, or ROI) is precisely cropped. This sequence of mouth ROIs is then normalized, converted to grayscale, and assembled into a tensor ready for the model.
- Core VSR Model (Encoder-Decoder): The prepared data is fed into the main Visual Speech Recognition model.
    - The Visual Frontend extracts spatio-temporal features from the mouth movements.
    - The Temporal Encoder processes the sequence of features to understand the context and relationships over time.
    - The Hybrid Decoder uses this contextual information to generate the most probable sequence of Arabic phonemes.
- Post-processing & Inference: The raw output from the model is a sequence of phoneme probabilities. A beam search decoding algorithm is applied to this output to find the most likely complete sentence, which is then presented to the user as the final transcription.

3.3 Used Models

Our core contribution involves a comparative analysis of different temporal encoder backbones within a consistent end-to-end framework. All models share the same visual frontend and hybrid decoder structure but differ in their temporal encoding module.

3.3.1. Visual Frontend: 3D-CNN + ResNet-18

This component extracts features from the sequence of mouth images. It consists of:

- A 3D Convolutional Layer (Conv3D) to capture short-term motion between frames.
- A ResNet-18 Backbone to learn robust and deep spatial features for each frame's visual content.

3.3.2. Temporal Encoders: A Comparative Study

We experimented with three distinct architectures for this critical component:

1. Conformer: This model represents the current state-of-the-art by combining self-attention and convolution. It excels at capturing both global, long-range dependencies and local, fine-grained patterns in the speech sequence. (Figure 3.2)
2. MS-TCN (Multi-Scale Temporal Convolutional Network): This purely convolutional network uses multiple parallel branches with different 1D convolution kernel sizes. This allows it to simultaneously analyze the temporal sequence at various scales, from very short to long-term dependencies. (Figure 3.3)
3. DCTCN (Densely Connected Temporal Convolutional Network): This deep TCN uses dilated convolutions to achieve a large receptive field efficiently. Its key feature is dense connectivity, where each layer receives the feature maps from all preceding layers, promoting feature reuse and robust gradient flow. (Figure 3.4)

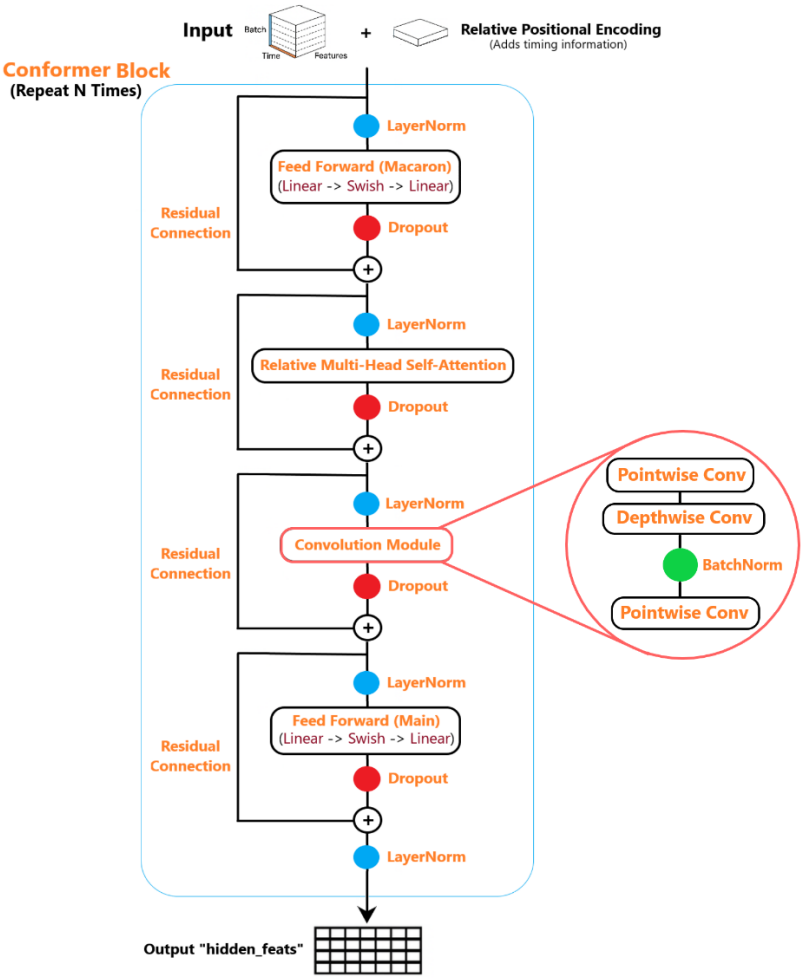

Figure 3.2: Conformer Architecture

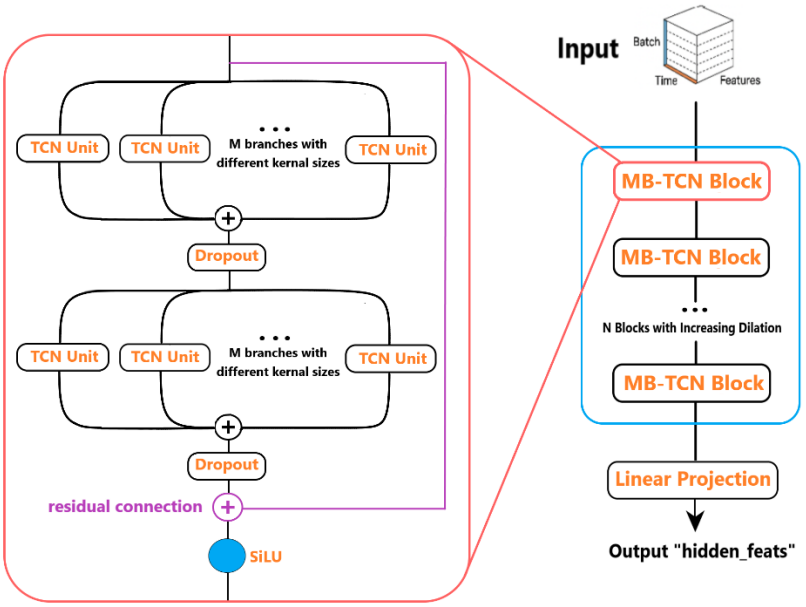

Figure 3.3: MSTCN Architecture

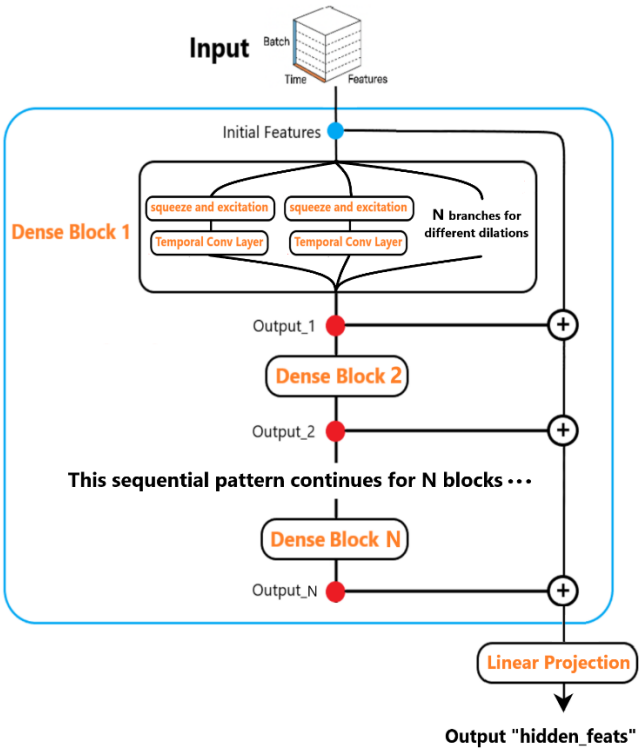

Figure 3.4: DCTCN Architecture

3.3.3. Hybrid CTC/Attention Decoder

To generate the final transcription, we use a hybrid training and decoding strategy:

- Attention-based Decoder: A 6-layer Transformer decoder generates the output sequence by attending to the most relevant encoded features.
- CTC Branch: A parallel Connectionist Temporal Classification loss is applied to the encoder's output. This acts as a regularizer, forcing a monotonic alignment that stabilizes training and improves performance. The final prediction combines the outputs of both mechanisms.

3.4 Used Technologies

The implementation and experimentation of this project were made possible by a number of key technologies, libraries, and frameworks:

Deep Learning Framework:

- PyTorch: The primary framework used for building, training, and evaluating all deep learning models.
- ESPnet: A comprehensive framework for end-to-end speech processing, utilized for its Conformer based model, CTC/Attention loss functions and beam search decoding modules.

Core Python Libraries:

- NumPy: For efficient numerical operations and data manipulation.
- OpenCV: Utilized primarily for reading video files frame by frame and for general image manipulation tasks during preprocessing.

- RetinaFace (ResNet50 Backbone): A deep learning-based face detector used to accurately locate the speaker's face in each video frame. Its robust performance is essential for the subsequent mouth cropping step.
- Torchvision: Used for its efficient video and image I/O operations, specifically for writing the preprocessed mouth region tensors to disk before training.

Data Processing:

- FFmpeg: A command-line tool used for extracting audio and manipulating video files.
- Google Gemini API: Leveraged for the automated transcription of audio to create our sentence-level dataset.
- Farasa: An open-source Arabic text processing toolkit used for the initial diacritization of our text labels.

Models and Checkpoints:

- auto_avsr (mpc001) [8]: We utilized the public checkpoint (vsr_trlrs2lrs3vox2avsp_base.pth) from this project, pre-trained on LRS2, LRS3, and VoxCeleb2, as the foundation for our transfer learning approach.

Development and Training Environment:

- Jupyter Notebooks: For interactive development, data analysis, and experimentation.
- Kaggle: Utilized for accessing the necessary GPU resources required for training the deep learning models in a timely manner.

Chapter 4: Experimental Work and Results

This chapter details the experimental methodology, training procedures, and comparative results of the models developed for Arabic lip-reading. Our experiments were designed to systematically evaluate the performance of different temporal encoder architectures and to assess the impact of diacritization on transcription accuracy.

4.1 Experimental Setup

All experiments were conducted on the Kaggle platform, utilizing its GPU resources. The core framework for model development was PyTorch, with key components from the ESPnet toolkit being integrated for the Conformer architecture and its associated loss functions.

4.1.1 Hyperparameter Configuration

To ensure a fair comparison, a consistent set of hyperparameters was used across all model architectures during training. Key parameters are detailed in Table 4.1.

Table 4.1: General Hyperparameter Configuration

Parameter	Value / Details
Optimizer	AdamW
Learning Rate	1.03e-4
Optimizer Betas	(0.9, 0.98)
Weight Decay	0.02 (Applied to all parameters except LayerNorm and bias terms)
Scheduler	WarmupCosineScheduler (5 epochs warmup, 60 total epochs)
Batch Size	64
Gradient Accumulation	4 steps
Data Augmentation	Random Horizontal Flip, Random Crop (88x88), Color Jitter, Random Affine Transform, Temporal Masking & Jittering
Transfer Learning	Initialized from vsr_trlrs2lrs3vox2avsp_base.pth. Visual frontend was frozen.

4.1.2 Model Architectures5

All models shared a common visual frontend (3D-CNN + ResNet-18) and a hybrid CTC/Attention decoder. The core of our experimentation lay in comparing the temporal encoder architectures detailed in Table 4.2.

Table 4.2: Temporal Encoder Architectural Configurations

Parameter	Conformer	MS-TCN	DCTCN
Primary Module	Self-Attention & Convolution	Multi-Scale Convolution	Densely Connected Convolution
Attention/Hidden Dim	768	768	768
Num Blocks / Block Config	12 Blocks	—	[4, 4, 4, 4]
Feed-Forward Units	3072	—	—
Attention Heads	12	—	—
CNN Kernel Size	31	[3, 5, 7, 9]	[3, 5, 7, 9]
Dilation Sizes	—	—	[1, 2, 4, 8]
Dropout Rate	0.1	0.2	0.2

4.2 Training Methodology

We employed a curriculum learning strategy for the hybrid CTC/Attention architecture. This staged approach, detailed in Table 4.3, helps stabilize training by first establishing a strong encoder alignment before refining the decoder's contextual understanding.

Table 4.3: Curriculum Learning Schedule

Stage	Epochs	CTC Weight	Objective
1	1-20	0.7	CTC-Focused: Encourages monotonic alignment between visual features and output tokens, stabilizing the encoder.
2	21-40	0.5	Balanced: Jointly optimizes CTC and attention, allowing the decoder to learn richer contextual relationships.
3	41-60	Annealed 0.4 -> 0.2	Attention-Focused: Fine-tunes the decoder to generate more coherent and contextually accurate sequences.

4.3 Evaluation Metric

The primary metric for evaluating model performance is the Token Error Rate, referred to by the conventional name Character Error Rate (CER). It is crucial to clarify that the fundamental unit of error is a *token*, whose definition varies based on the experimental setup:

- Diacritized ("Dia") Setup: A token is a single Arabic character combined with its diacritics (e.g., تَ, بُ, مْ). Vocabulary size: 227 tokens.
- Pure (No-Diacritics) Setup: A token is a single, base Arabic character (e.g., ت, ب, م). Vocabulary size: 38 tokens.

The CER is calculated as the Levenshtein distance—the sum of substitutions (S), insertions (I), and deletions (D)—normalized by the total number of tokens (N) in the ground-truth. For the "Pure" model evaluation, the ground-truth labels were stripped of all diacritics to ensure a fair comparison.

4.4 Comparative Results

The final validation CER, evaluated on our hold-out set of 250 manually-verified video samples, is summarized in Table 4.4.

Table 4.4: Model Performance Comparison (Validation CER)

Model Architecture	Diacritics Setting	Best Validation CER
Conformer	No ("Pure")	17.84%
Conformer	Yes ("Dia")	23.49%
MS-TCN	No ("Pure")	24.00%
MS-TCN	Yes ("Dia")	29.68%
DCTCN	No ("Pure")	20.55%
DCTCN	Yes ("Dia")	28.12%

4.5 Analysis of Results

The experimental results lead to two primary conclusions:

1. Architectural Superiority of Conformer: The Conformer architecture significantly outperforms both TCN-based models. Its superior ability to model both local and global dependencies is evident from the large performance gap.
2. The Negative Impact of Diacritics: The "Pure" Conformer model achieved a CER of 17.84%, a relative improvement of over 6% compared to its diacritized counterpart (23.49%). This strongly suggests that including diacritics is detrimental to performance. The visual information for diacritics is often negligible, and their inclusion drastically increases the vocabulary size (from 38 to 227 tokens). This forces the model to learn a much sparser and more ambiguous mapping, hindering its ability to accurately predict the fundamental base characters.

4.6 Qualitative Evaluation: Sample Predictions

Table 4.5 provides a qualitative comparison of predictions from our models. The "Ground Truth" column shows the diacritic/diacritic-stripped target used for evaluating the model.

Table 4.5: Sample Predictions from Different Models

Text Style	Ground Truth	Conformer	MS-TCN	DCTCN
Diacritics	تَنْظِيمُ ٱدَّوْلَةِ ٱلْإِسْلَامِيُّ	تَنْظِيمُ ٱدَّوْلَةِ ٱلْإِسْلَامِيُّ	تَنْظِيمُ ٱدَّوْلَةِ ٱلْإِسْلَامِيَّةِ	تَنْظِيمُ ٱدَّوْلَةِ ٱلْإِسْلَامِيِّ
Pure	تنظيم ٱدولة ٱلإسلامي	تنظيم ٱدولة ٱلإسلامي	تنظيم ٱدولة ٱلإسلامية	تنظيم ٱدولة ٱلإسلامية
Diacritics	صَادِرِ ٱلْجَزِيرَهْ بِإِرْتِفَاعْ	صَادِرُ لِلْجَزِيرَةِ بِٱرْتِفَاعِ	مُرَاسِلُ ٱلْجَزِيرَةِ بِٱرْتِفَاعِ	صَادِرُ ٱلْجَزِيرَةِ ٱرْتِفَاعِ
Pure	صادر ٱلجزيره بإرتفاع	صادر ٱلجزيرة بٱتفاع	مصادر ٱلجزيرة بسفاع	صادر للجزيرة بٱتفاع
Diacritics	رَئِيسُ ٱلْوُزَرَاءْ ٱلْبُنَانِيُّ	رَئِيسُ ٱلْوُزَرَاءْ ٱلْبُنَانِيُّ	رَئِيسُ ٱلْوُزَرَاءِ ٱلْبُنَانِيِّ	رَئِيسُ ٱلْوُزَرَاءِ ٱلْبُنَانِيُّ
Pure	رئيس ٱلوزراء ٱلبناني	رئيس ٱلوزراء ٱلبناني	رئيس ٱلوزراء ٱليمني	رئيس ٱلوزراء ٱلبناني
Diacritics	مَبْعُوثُ ٱلْأُمَمِ ٱلْمُتَّحِدْ	مَبْعُوثُ ٱلْأُمَمِ ٱلْمُتَّحِدَةِ	مَبْعُوثِ ٱلْأُمَمِ ٱلْمُتَّحِدَةِ	مَبْعُوثُ ٱلْأُمَمِ ٱلْمُتَّحِدَةِ
Pure	مبعوث ٱلأمم ٱلمتحد	مبعوث ٱلأمم ٱلمتحدة	مبعوث ٱلأمم ٱلمتحدة	مبعوث ٱلأمم ٱلمتحدة
Diacritics	وِزَارَةِ ٱدِّفَاعِ ٱلْأَمِيرِيكِي	وِزَارَةِ ٱدِّفَاعِ ٱلْأَمْرِيكِيِّ	زَارَةِ ٱدِّفَاعْ ٱلْأَمْرِيكِيِّ	وِزَارَةِ ٱدِّفَاعِ ٱلْأَمْرِيكِيِّ
Pure	وزارة ٱدفاع ٱلأميريكي	وزارة ٱدفاع ٱلأمريكي	وزارة ٱدفاع ٱلأمريكي	وزارة ٱدفاع ٱلأمريكي
Diacritics	مَعَ ٱلْمُعَارَضَهْ ٱسُّورِيَّهْ	مَعَ ٱلْمُعَارَضَهْ ٱسُّورِيَّهْ	مَعَ ٱلْمُعَارَضَةِ ٱسُّورِيَّهْ	مَعَ ٱلْمُعَارَضَةِ ٱسُّورِيَّةِ
Pure	مع ٱلمعارضه ٱسوريه	مع ٱلمعارضه ٱسوريه	من ٱلمعارضة ٱسورية	مع ٱلمعارضة ٱسورية
Diacritics	قُوَّاتِ ٱتَّحَالُفْ فِي ٱلْيَمَنْ	قُوَّاتِ ٱتَّحَالُفِ ٱلْيَمَنْ	قُوَّاتِ ٱتَّحَالُفِ ٱلْيَمَنِيِّ	قُوَّاتِ ٱتَّحَالُفِ ٱلْيَمَنِ
Pure	قوات ٱتحالف في ٱليمن	قوات ٱتحالف ٱليمن	قوات ٱتحالف ٱليمن	قوات ٱتحالف ٱليمني
Diacritics	مُحَمَّدْ بِنْ سَلْمَانْ	مُحَمَّدْ بِنْ سَلْمَانْ	مُحَمَّدْ بِنْ سَلْمَانْ	مُحَمَّدْ بِنْ سَلْمَانْ
Pure	محمد بن سلمان	محمد بن سلمان	محمد بن سلمان	محمد بن سلمان

The selected samples qualitatively reinforce the quantitative findings, clearly demonstrating the performance hierarchy among the models. For straightforward utterances, such as the common name "مُحَمَّدْ بِنْ سَلْمَانْ", all models achieve perfect accuracy, showing their competence on videos with ideal visual conditions.

The performance differences become evident in more challenging phrases. The Conformer > DCTCN > MSTCN hierarchy is perfectly illustrated in one example where Conformer provides an error-free transcription, DCTCN introduces a minor character-level error, and MSTCN makes a critical semantic mistake by substituting "مع" for "من". This pattern of progressive degradation is consistent across various test cases.

Furthermore, the table highlights a recurring pattern where the Conformer and DCTCN models perform at a similarly high level, while the MSTCN model consistently lags with more severe errors. A clear example of this is the semantic confusion between "ٱليمني" and "ٱلبناني", a mistake only made by the MSTCN model. These examples confirm that the Conformer architecture is the most robust, followed closely by DCTCN. In contrast, the MSTCN model is more susceptible to significant prediction errors as utterance complexity increases.

Chapter 5: User Interface

While quantitative metrics like Character Error Rate (CER) are essential for evaluating model performance, the ultimate goal of this project is to create a usable system that bridges communication gaps. To demonstrate the practical viability of our research, we developed a complete, end-to-end mobile application.

This chapter details the architecture, implementation, and functionality of our proof-of-concept system. The application serves as an interactive platform for users to experience the lip-reading models in real-time, providing tangible evidence of the project's success and a foundation for future development.

5.1 System Architecture

To ensure a responsive user experience while handling computationally intensive deep learning models, we adopted a client-server architecture. This design separates the user-facing interface from the backend processing, allowing each component to be optimized for its specific role.

- Frontend (Client): A cross-platform mobile application built using Flutter. This allows for a single codebase to serve both Android and iOS users, providing a native look and feel. The frontend is responsible for all user interactions, including video capture, option selection, and displaying the final transcription.
- Backend (Server): A high-performance web server built with FastAPI. FastAPI was chosen for its speed, automatic generation of interactive API documentation (via Swagger UI), and native support for asynchronous operations, which is crucial for handling long-running inference tasks without blocking.

The workflow is as follows: The Flutter app sends a video and user-selected parameters to the FastAPI backend. The backend performs the heavy lifting of video preprocessing and model inference and then returns the final text transcription to the app to be displayed to the user.

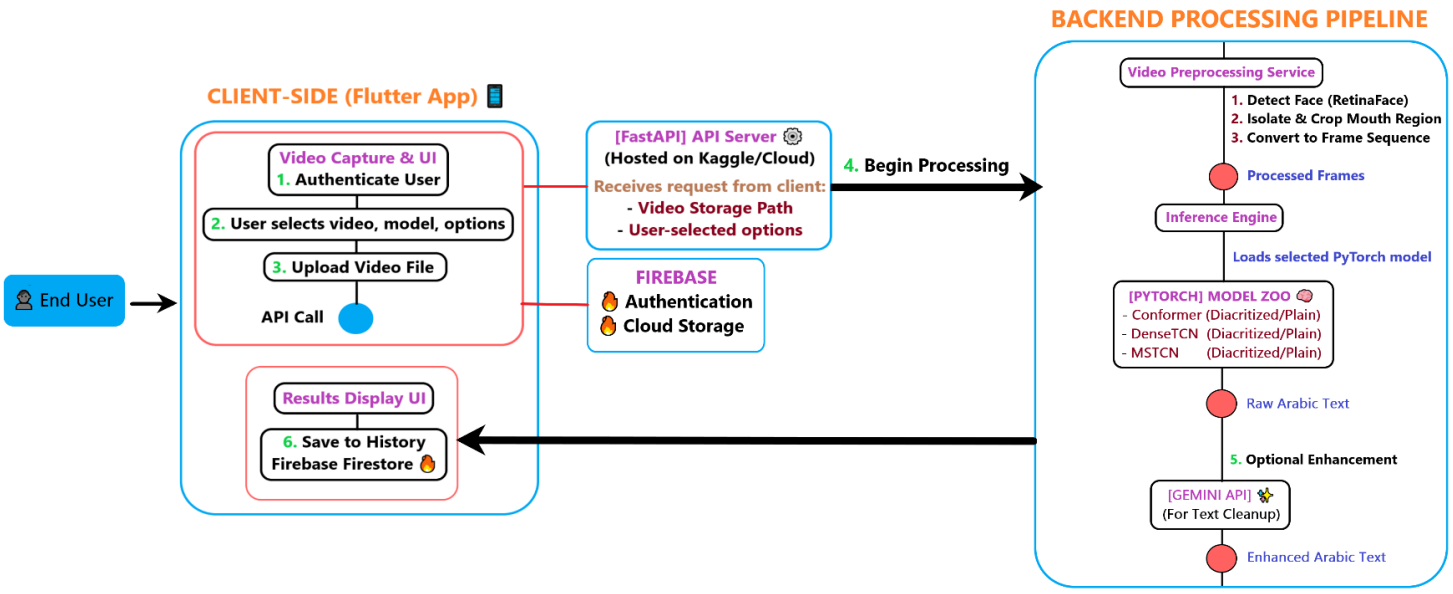

Figure 5.1: High-level application architecture

5.2 Backend Implementation (FastAPI)

The backend is the core processing engine of our system. It exposes a single, robust API endpoint (/transcribe/) that handles all inference requests.

Key Components:

- API Framework: FastAPI manages incoming HTTP requests, data validation (using Pydantic models), and response generation.
- Model Loading: Based on the user's request, the backend dynamically loads the appropriate pre-trained PyTorch model weights for the selected architecture (Conformer, MS-TCN, or DCTCN) and diacritization setting.
- Inference Pipeline: Upon receiving a video, the backend executes the full preprocessing pipeline (face detection, mouth cropping, normalization), runs the video tensor through the selected model to generate a sequence of character probabilities, and decodes this sequence into a raw text string.

5.3 Frontend Implementation (Flutter)

The frontend was designed with simplicity and usability in mind, providing an intuitive interface for users to interact with the models.

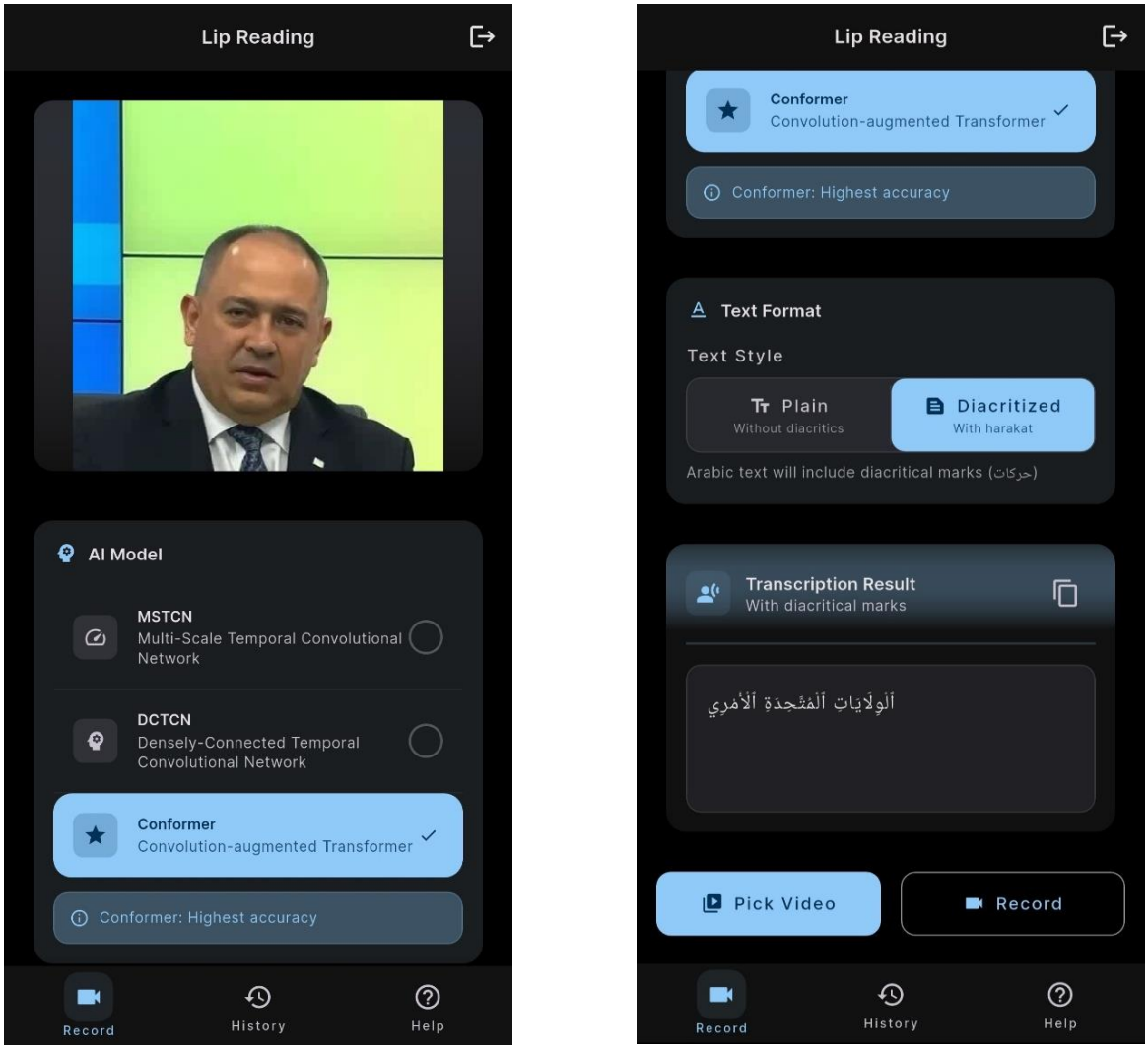

Figure 5.2: Screenshots of the Flutter application interface

Core Features of the User Interface:

1. Video Input: Users are presented with two options for providing a video:
    - Upload from Gallery: Select an existing video from the device's storage.
    - Record with Camera: Open the device's camera to record a new video directly within the app.

2. Model Selection: A dropdown menu allows the user to choose which of the three trained models to use for transcription:
    - Conformer
    - MS-TCN
    - DCTCN

3. Diacritization Control: A simple toggle switch allows the user to select whether the transcription should be performed with or without Arabic diacritics. This directly corresponds to the "Dia" or "Pure" training approaches.
4. Transcription Display: After processing, the final transcribed text is displayed clearly on the screen.

5.4 Advanced Functionality: Gemini-Powered Enhancement

To further improve the quality and readability of the output, we integrated an optional post-processing step using Google's Gemini Flash model.

The raw output from our VSR models is a literal, character-by-character transcription. It often lacks proper spelling, context-aware grammar, and the natural flow of human language. When the user enables the "Enhance with Gemini" option:

1. The raw transcription from our model is first generated.
2. This raw text is then sent as a prompt to the Gemini API with an instruction to correct any grammatical errors, add missing characters.
3. The refined, more coherent text from Gemini is then returned to the user.

This feature transforms the system from a simple transcription tool into a more sophisticated communication aid.

5.5 Deployment and Accessibility

For the purpose of this project, the FastAPI backend is deployed on a Kaggle Notebook to leverage its free GPU resources for inference. To make the backend accessible to the Flutter application over the internet, we use localtunnel.

This setup creates a secure, temporary public URL for our locally-running FastAPI server. The Flutter application is configured to send its API requests to this URL. While this is not a production-ready deployment solution, it serves as a highly effective method for demonstrating the fully functional, end-to-end system in a real-world setting.

Chapter 6: Conclusion

This project addressed the significant gap in assistive technology for continuous, sentence-level Arabic lip-reading, a task made challenging by the language's unique visemic properties and a scarcity of suitable datasets. Our primary objective was to design, implement, and evaluate an end-to-end pipeline capable of transcribing spoken Arabic from silent video.

We constructed a novel, large-scale dataset for this task by leveraging Google Gemini 2.5 Pro for automated transcription and combining it with a hand-corrected subset for robust training and evaluation. Our comparative analysis of Conformer, MS-TCN, and DCTCN architectures showed that the Conformer model was superior, achieving a best Character Error Rate (CER) of 20.14%. A key insight was the impact of diacritization on performance. Contrary to our initial hypothesis, the model trained without diacritics performed better, suggesting that the expanded vocabulary and visual ambiguity of diacritical marks may introduce more noise than signal.

To demonstrate real-world applicability, we developed a proof-of-concept mobile application using Flutter and FastAPI. This interactive system allows users to select between the models, choose a transcription style (with or without diacritics), and optionally enhance the output with Google's Gemini API for improved coherence. This application serves as a tangible validation of our research and a solid foundation for a practical communication tool.

Limitations

We acknowledge this project's limitations. Our dataset, while novel, lacks the full diversity of Arabic dialects and was created from videos with controlled conditions (frontal poses, good lighting), meaning performance may degrade "in-the-wild." Furthermore, while the CER is competitive, errors related to visually ambiguous phonemes (homophemes) and short, unstressed words persist, highlighting the inherent constraints of a vision-only system.

Future Work

The most impactful future work is to improve model generalization by training on larger, more diverse datasets. A crucial next step is to incorporate emerging datasets like LRS-Lang (a multilingual 1300 hours TEDx talks dataset coming soon), which, with an expected 60 hours of varied Arabic sentences, would dramatically improve the model by exposing it to a wider range of vocabulary and dialects. From a modeling perspective, integrating a powerful Arabic language model into the beam search decoding process could significantly improve contextual understanding and resolve ambiguities. For the application, migrating the backend to a scalable cloud service is the logical next step for enabling stable public access and user feedback collection.

In closing, this project has successfully delivered a complete and effective system for Arabic lip-reading. By establishing a new performance benchmark, providing a publicly accessible framework, and identifying key challenges and opportunities, we have laid a strong foundation for future innovation in the field of Arabic assistive technology. The complete source code, pre-trained models, and application framework are publicly available in our project repository to encourage further development [9].

References

[1] Daou, S., Ben-Hamadou, A., Rekik, A., & Kallel, A. (2025). Cross-Attention Fusion of Visual and Geometric Features for Large-Vocabulary Arabic Lipreading. Technologies, 13(1), 26.

[2] Jabr, Z., Etemadi, S., & Mozayani, N. (2024). Arabic Lip Reading with Limited Data Using Deep Learning. IEEE Access.

[3] Ma, P., Wang, Y., Petridis, S., Shen, J., & Pantic, M. (2022, May). Training strategies for improved lip-reading. In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 8472-8476). IEEE.

[4] Ma, P., Haliassos, A., Fernandez-Lopez, A., Chen, H., Petridis, S., & Pantic, M. (2023, June). Auto-avsr: Audio-visual speech recognition with automatic labels. In ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 1-5). IEEE.

[5] Aljohani, N. F., & Jaha, E. S. (2023). Visual Lip-Reading for Quranic Arabic Alphabets and Words Using Deep Learning. Computer Systems Science & Engineering, 46(3).

[6] Daou, S. (2024). LRW-AR [Dataset]. Open Science Framework. https://doi.org/10.17605/OSF.IO/RZ49X

[7] Kamel, H. K. S., Ramzy, E. H. A., Abdelaziz, A. A. A., Ashour, A. W. M., & Ali, E. M. A. (2025). LRC-AR Dataset (based on Daou, 2024) [Dataset]. Kaggle. https://kaggle.com/datasets/hazemkhaled21/lrc-ar

[8] Ma, P. (2023). Auto-AVSR: Audio-Visual Speech Recognition with Automatic Labels (GitHub repository). GitHub. https://github.com/mpc001/auto_avsr

[9] Kamel, H. K. S., Ramzy, E. H. A., Abdelaziz, A. A. A., Ashour, A. W. M., & Ali, E. M. A. (2025). Arabic Lip Reading (GitHub repository). GitHub. https://github.com/Essa-Ramzy/Arabic-Lip-Reading