



**Faculty of Computers &  
Informatics**



**Benha University**

# **Automatic Video dubbing system from English to Arabic**

A senior project submitted in partial fulfillment of the requirements for the degree of  
Bachelor of Computers and Artificial Intelligence.

**Artificial Intelligence** Departement,

## *Project Team*

- 1- Mahmoud Mohamed Ibrahim Abd-Elhafez
- 2- Elsebaay Mohamed Sebaay
- 3- Esmail Mahmoud Esmail
- 4- Salah Sameh Salah Hussain Barakat
- 5- Ali Mosaad Abdelfattah Mostafa
- 6- Mohey El-Dien Ahmed Mohey El-Dein
- 7- Ahmed Mohamed Hafez Saad Hassan

## *Under Supervision of*

**Dr. Ibrahim Zaghloul**

Benha, June 2023

## DECLARATION

We hereby certify that this material, which we now submit for assessment on the program of study leading to the award of Bachelor of Computers and Artificial Intelligence in *Bachelor degree* is entirely our own work, that we have exercised reasonable care to ensure that the work is original, and does not to the best of our knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of our work.

**Signed:**

---

---

---

---

---

---

---

**Date:** June 21<sup>st</sup>, 2023

# ABSTRACT

*“If you speak to the man in the language he understands, that goes to his head. If you talk to him in his language, that goes to his heart.” - Nelson Mandela.*

This document presents a comprehensive study on video dubbing techniques and the development of a specialized video dubbing system. The objective is to replace the original voices in foreign language videos with the voices of performers speaking the language of the target audience (Nigel G. Ward, 2022), while ensuring synchronization between lip movements and the dubbed speech (Chenxu Hu1, 2021). The proposed system incorporates speech translation, voice synthesis, and video analysis techniques to achieve accurate dubbing.

The video dubbing system addresses the challenge of maintaining the original video's visual integrity during the automated dialogue replacement (ADR) process. By considering the speech duration of each token in translation (Yihan Wu, 2023 ), the system ensures that the length of the dubbed speech matches that of the source video, facilitating seamless integration without compromising timing and synchronization.

A cascaded approach (Marcello Federico, 2020) is employed, combining Automatic speech recognition, machine translation, voice synthesis, and video analysis techniques. Automatic speech recognition convert speech to written format while machine translation accurately translates the original dialogue into the desired target language, while voice synthesis generates natural and high-quality speech based on the translated text. The synthesized voices are carefully synchronized with the lip movements of the actors, ensuring a visually coherent dubbing result.

To achieve precise control over prosody and intonation, video analysis techniques analyze the lip movements and facial expressions of the actors. This information influences the rhythm and emphasis of the generated speech, closely matching the original performances and enhancing authenticity and believability.

Experimental evaluations, especially in English to Arabic dubbing, prove the effectiveness of the system, creating synchronized and culturally relevant dubbing for diverse audiences. Automated video dubbing enhances accessibility, engagement, and global distribution of multilingual content while preserving visual integrity for cross-cultural communication.

# Table of Contents

1	Introduction .....	8
1.1	The Importance of Automatic Video Dubbing .....	9
1.2	The challenges for Arabic STS evaluation .....	9
1.3	Problem statement.....	9
1.4	Motivation .....	10
1.5	Impact of the project .....	10
1.6	Organization Of the Project Report .....	10
2	Literature review .....	11
2.1	Basic Audio Terminology: .....	11
2.2	Automatic Speech Recognition (ASR) .....	13
٢,٢,١	wave encoding.....	14
٢,٢,٢	Text decoding .....	16
2.3	Machine Translation.....	17
2.3.1	Rule-based machine translation (RBMT) .....	18
2.3.2	Corpus Based Machine Translation .....	18
2.4	Text to Speech.....	20
2.4.1	Concatenative TTS .....	21
2.4.2	Mainstream 2-Stage .....	21
2.5	Disadvantages of cascaded system.....	22
3	Methodology .....	23
3.1	speech recognition.....	25
3.1.1	Wave2vec2.0 architecture .....	26
3.1.2	Pre-train wave2vec .....	27
3.1.3	CTC decoder .....	27
3.1.4	CTC Algorithm .....	28
3.1.1	Limitation of CTC and the naïve Argmax:.....	29
3.2	machine translation.....	30
3.2.1	google machine translation Architecture. ....	30
3.2.2	Residual Connections .....	31
3.3	text to speech .....	32
3.3.1	Google Text to speech (WaveNet).....	32
3.3.2	FastSpeech2 .....	35
3.4	Shakaala model.....	35
3.5	punctuation model .....	36
3.6	Datasets .....	37
3.6.1	LibriSpeech DATASET .....	37
3.6.2	MuST-C Dataset.....	38
3.6.3	MGB-3 Dataset.....	39
3.6.4	Dataset .....	40

4	Results of our AVD System.....	41
4.1	Evaluation Metrics .....	41
4.1.1	Perplexity.....	41
4.1.2	Word Error Rate (WER) .....	41
4.1.3	BLEU (bilingual evaluation understudy) .....	42
4.2	Results on evaluation datasets .....	47
4.2.1	Results Speech recognition .....	47
4.2.2	Results of Machine translation .....	47
5	System analysis and design.....	48
5.1	Assumptions .....	48
5.2	System requirements .....	48
5.3	System users and use cases .....	48
5.4	System architecture .....	49
5.5	Data Flow For Dubbing Video .....	50
5.6	Database design.....	51
5.6.1	Users table: .....	51
5.6.2	Uploaded videos table: .....	52
5.6.3	Users YouTube video table:.....	52
5.6.4	YouTube video table:.....	52
6	Android application .....	54
7	Conclusion and future work.....	60
7.1	Conclusion.....	60
7.2	Future work.....	60
8	References .....	61

# LIST OF FIGURES

Figure 2-1 Automatic Video Dubbing Process	11
Figure 2-2 Waveform	11
Figure 2-3 Power Spectrogram	12
Figure 2-4 Mel-spectrogram	13
Figure 2-5 Cascaded system for speech-to-speech translation	13
Figure 2-6 ASR component	13
Figure 2-7 Speech Recognition history	14
Figure 2-8 Wave2Vec2.0 model	15
Figure 2-9 Hubert model	15
Figure 2-10 Conformer model	16
Figure 2-11 Text Decoding	16
Figure 2-12 Summary of Machine translation	18
Figure 2-13 Encoder Decoder Architecture	19
Figure 2-14 RNN Encoder-Decoder	19
Figure 2-15 Transformer based Encoder-Decoder	20
Figure 2-16 Text to Speech summary	20
Figure 2-17 A basic high-level overview of mainstream 2-Stage TTS System	21
Figure 2-18 TTS Mainstream 2-stage	22
Figure 3-1 AVD System	24
Figure 3-2 speech translator block	25
Figure 3-3 Wave2vec pretrain architecture	25
Figure 3-4 wave2vec Feature Encoder	26
Figure 3-5 Wave2vec Context Network	26
Figure 3-6 Wave2vec Quantization Module	26
Figure 3-7 Wave2vec Finetune	27
Figure 3-8(Character probabilities predicted for each audio slice)	28
Figure 3-9 Naive Decoding	28
Figure 3-10 CTC Decoding	28
Figure 3-11(Many legitimate alignments produce the same output)	28
Figure 3-12 Decoding Problem	29
Figure 3-13 Beam Search	29
Figure 3-14 Google Machine Translation Architecture	31
Figure 3-15 LSTM with Residual	31
Figure 3-16 Residual Bidirectional LSTM	32
Figure 3-17 Causal Convolution Layer	32
Figure 3-18 Dilated Convolution Layer	33
Figure 3-19 Overview of residual block and complete architecture	34
Figure 3-20 Fast Speech 2	35
Figure 4-1 Clipped Precision Example	43
Figure 4-2 Precision uni-gram	44
Figure 4-3 Precision bi-gram	44
Figure 4-4 3 Precision tri-gram	44
Figure 4-5 Precision 4-gram	45
Figure 5-1 Use-case diagram	49
Figure 5-2 Overall architecture of our E-learning system	50
Figure 5-3 Data flow diagram for AVD	51
Figure 5-4 Entity-relationship diagram for database	53
Figure 6-1 Register (b) Login	54

<i>Figure 6-2 Entering wrong data / Entering email / sent email / resetting password</i>	55
<i>Figure 6-3 Home screen / adding videos</i>	56
<i>Figure 6-4 Entering video description / task completion message / uploaded video data</i>	56
<i>Figure 6-5 Application sidebar / User profile information / edit user profile</i>	58
<i>Figure 6-6 App Settings</i>	58

# *Chapter 1*

## **1 Introduction**

Video dubbing is a process that involves replacing the original soundtrack of a video with a new soundtrack in a different language while maintaining the same meaning. It requires significant linguistic and cultural adaptation to ensure synchronization between the new soundtrack and the video content, catering to the audience's cultural background. The film, television, and entertainment industry widely employ video dubbing to make visual content accessible to diverse language-speaking audiences. The primary objective of video dubbing is to provide a seamless viewing experience by ensuring that the audio and visual elements of the video are synchronized and accurately convey the intended meaning. Achieving this requires a high level of accuracy, attention to detail, and a deep understanding of the cultural nuances and linguistic conventions of both the source and target languages. The applications of video dubbing are diverse, spanning entertainment, education, and marketing. By enabling content producers to distribute their material to a global audience, video dubbing allows them to tap into a wider market. Furthermore, viewers can enjoy their favorite movies and TV shows in their native language, leading to a more immersive and enjoyable experience.

In the field of education, video dubbing can be used to create educational materials in multiple languages, which proves particularly advantageous for language learning. Students can enhance their language skills by watching videos in their target language, facilitating comprehension and practice. Video dubbing also plays a crucial role in marketing and advertising, as it enables businesses to create localized advertisements and promotional materials that resonate with their target audience. In an increasingly globalized world, effective communication with customers in different regions is essential, and video dubbing provides a means to achieve that.

In summary, video dubbing is an integral part of the entertainment, education, and marketing industries, allowing content producers to reach a wider audience and enabling viewers to enjoy content in their native language. To achieve high accuracy in video dubbing, meticulous attention to detail and a deep understanding of the cultural and linguistic nuances of both languages involved are necessary. Video dubbing is a valuable tool for creating engaging and accessible content that connects with audiences worldwide. In the context of increasing online learning, many students are drawn to self-learning through multilingual tutorials. However, sometimes these tutorials are provided in a foreign language, and the subtitles may not be correctly synchronized, leading to a frustrating viewing experience. To address this issue, dubbed videos offer an alternative solution.

Dubbing, as employed in filmmaking, involves adding new dialogue or sounds to the soundtrack of a motion picture that has already been shot. It is commonly used to translate foreign-language films for audiences. Although dubbed soundtracks rarely match the artistic quality of the original foreign language soundtracks, subtitles are often preferred by viewers as a means of understanding the



dialogue in foreign films. Dubbing is often necessary in the original-language version of a soundtrack for technical reasons. It helps overcome issues that arise from synchronized filming, where recorded dialogue may be unclear or inaudible in certain shots due to factors like long distances, external noises, or limitations in microphone placement. By utilizing dubbing, filmmakers can obtain high-quality dialogue regardless of the conditions during shooting.

Video dubbing aims to make video content invariant across cultures worldwide. Automatic video dubbing systems typically involve three sub-tasks: Automatic Speech Recognition (ASR), which transcribes the original speech into text in the source language (omitted when subtitles are available); Neural Machine Translation (NMT), which translates the source language text to the target language; and Text-to-Speech (TTS), which synthesizes the translated text into target speech.

### ***1.1 The Importance of Automatic Video Dubbing***

The average adult watches over five hours of TV and video content each day. We're surrounded by visual, interactive, and immersive content that we access on multiple devices, wherever we are in the world. Video dubbing aims to make the video content invariant across worldwide cultures. Multimedia is all about the language of your target audience. Therefore, any part of the user experience needs to be localized. This means everything a user might see or hear, including audio, video, images, sounds, subtitles, on-screen text, most of the educational material gets published in English as stated before so video dubbing solves this problem once and for all.

### ***1.2 The challenges for Arabic STS evaluation***

Automatic video dubbing faces several challenges in achieving effective and high-quality results. Lip sync accuracy, naturalness of dubbed voice, cultural adaptation, and localization are crucial aspects that require sophisticated algorithms and techniques. Multilingual and multicultural considerations pose complexities in developing accurate speech recognition and translation models. Voice actor selection and direction play a significant role in maintaining quality and consistency.

Technical limitations in speech transcription, translation, and voice synthesis need to be overcome. Cost and time efficiency are additional challenges that require careful management. Advancements in speech recognition, machine translation, voice synthesis, and audio-visual processing are essential to address these challenges and improve automatic video dubbing systems in the future.

### ***1.3 Problem statement***

We can use the source speech  $S$  with duration  $T_s$  if available and/or its transcription  $x = (x_1; \dots; x_{T_x})$ , we aim to generate the translated text sequence  $y = (y_1; \dots; y_{T_y})$  as well as the synthesized speech  $A$  with duration  $T_a$ , where  $T_a$  should be as close to  $T_s$  as possible, while maintaining the high translation quality of  $y$  as well as the rhythmic and fluency of  $A$ . And concentrate more on the alignment between the phoneme and lip motion in video.

Dubbing is a challenge because of the limitations of time and space. Time is the first limitation of dubbing. A dubber must be careful of the duration of the sentence to make it easy for the person to pronounce. A sentence like “Good morning my friend, how are you today?” Will be naturally translated into صباح الخير يا صديقي، كيف حالك اليوم؟ That is fine, of course, but it does not work in dubbing because a dubbing translator counts the syllables of a sentence. In this case the ten syllables in English that resulted in 15 ones in Arabic. Five more syllables are not acceptable. So, a translator must decide to delete some words without losing the meaning. صديقي can be understood from the scene so the word goes off and good morning can be replaced by another shorter sentence. So, the sentence becomes: مرحبا، كيف حالك اليوم؟. Which sometimes changes the linguistic structure of the dubbed sentences, so timing and linguistic spacing are two of some limitations when dubbing.

### ***1.4 Motivation***

The motivation behind video-to-video dubbing is to simplify and expedite the localization of video content, making it accessible to a wide range of viewers. By automating the lip-syncing process, this technology reduces the need for labor-intensive manual efforts and significant resources.

In the past, dubbing involved hiring voice actors to re-record dialogue in different languages, requiring meticulous alignment of lip movements with the new audio track. This manual process was time-consuming, expensive, and often resulted in suboptimal lip-sync accuracy. However, with the advancements in video-to-video dubbing technology, the process has become more streamlined and efficient.

### ***1.5 Impact of the project***

Video to video dubbing has a significant impact on education by enhancing accessibility, cultural relevance, and engagement. It allows English educational videos to be translated into Arabic, enabling students to learn, and improving comprehension. By preserving visual context and providing translated audio, video to video dubbing facilitates visual learning and improves information retention. Overall, video to video dubbing enhances the effectiveness of educational materials and provides more engaging learning experiences for our students.

### ***1.6 Organization Of the Project Report***

This report will discuss all the aspects of our automatic video to video dubbing system. Chapter 1 presents a brief introduction about the project, problem statement, scope, deliverables, and impact. In chapter 2 we provide a thorough discussion of state-of-the-art methods. Chapter 3 describes the methodology used for dubbing video while chapter 4 shows the results of our solution components. In chapter 5 we show the system analysis and design. Chapter 6 presents the android app and its back-end description. We talk about the integration of our project components in chapter 7. Finally, the conclusion and future work are shown in chapter 8.

# Chapter 2

## 2 Literature review

While dubbing video for some of language is done, a few works has been done for Arabic. In this chapter we describe the efforts that have been made to develop a model for AVD. First, we discussed Basic Audio, Text and Video Terminology, then we discussed three stages that applied to AVD. AVD stages are summarized in figure 2.1. Finally, we go on solutions for each stage.

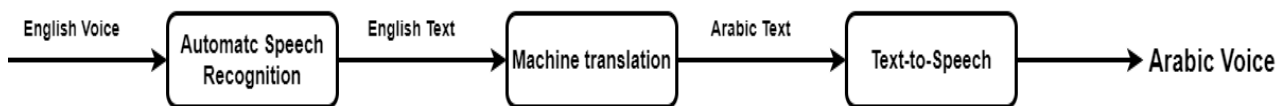


Figure 2-1 Automatic Video Dubbing Process

### 2.1 Basic Audio Terminology:

**Waveform:** A computer interprets an audio signal that changes amplitude over a fixed time frame. Each sample usually takes on 65,536 values (16-bits), and quality is measured in kHz.

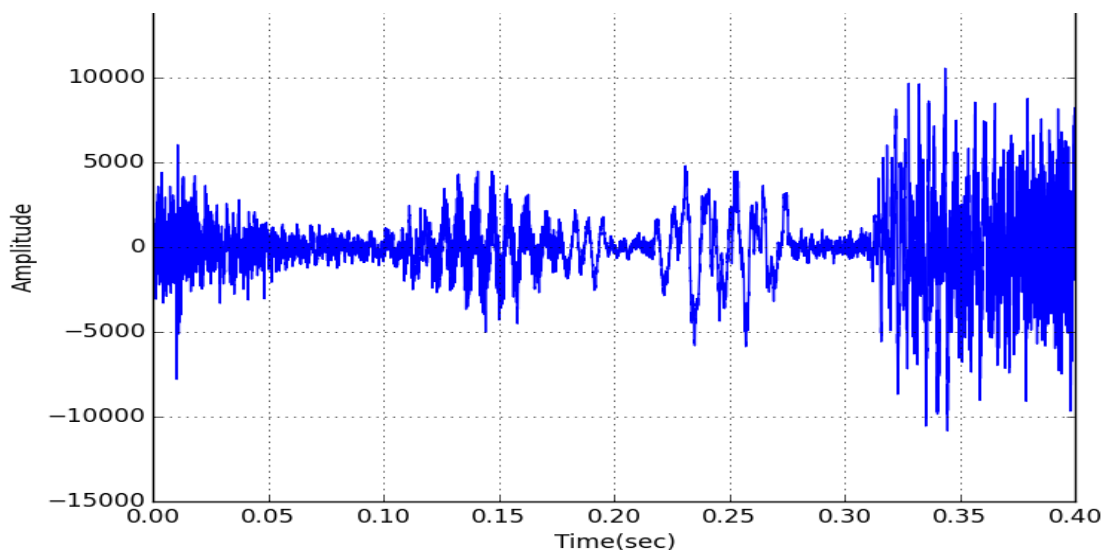
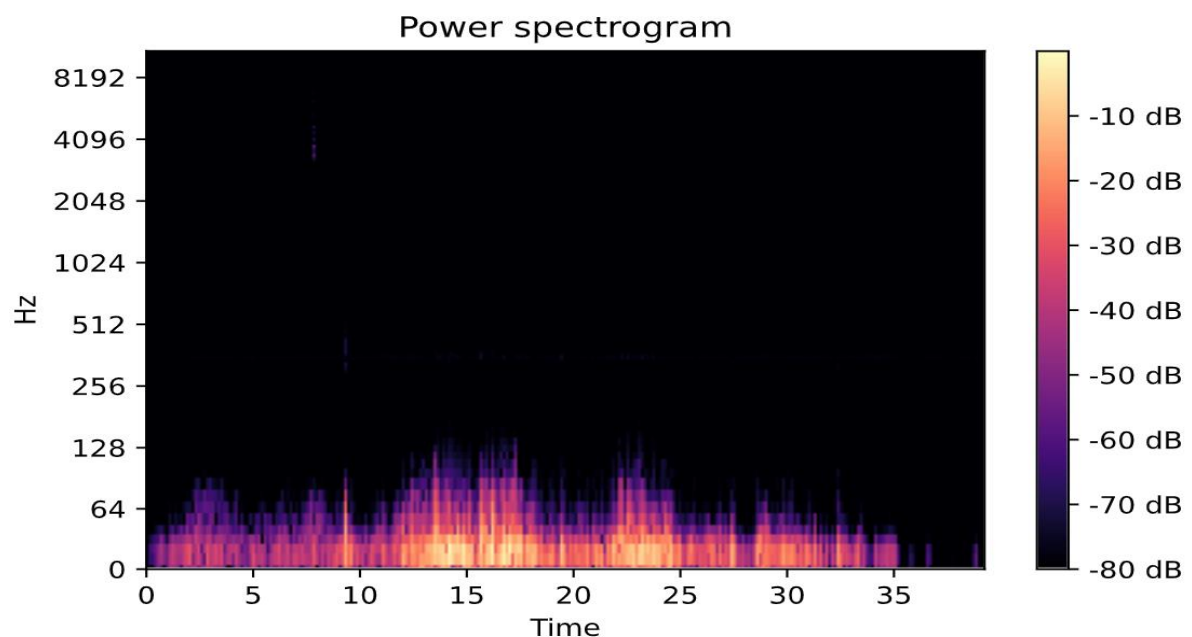


Figure 2-2 Waveform

**Phonemes:** The text-to-speech engine doesn't directly take characters as an input, but phonemes, specifically stressed ARPA (for English) as found in the CMU Dict. For example, green is G R IY1 N. Other languages may have different formats but the same usage. That is why we will convert our input text (during training) into phonemes that are distinct units of sounds that distinguish one word from another in a language. When your input is processed, it gets turned into phonemes by a neural network trained on known utterances, which also learns to generate spelling for novel words.

**Spectrograms:** Most work of previous research's and works on Speech tasks don't like to work on raw form of wave because it's so large and have high redundant data, so audio is converted into spectrograms, and Fourier transforms the source audio into the time-frequency domain. The transformation process chops up the duration of the sound signal into smaller signals before transformation then combines the output into a single view.



*Figure 2-3 Power Spectrogram*

**Mel-Spectrogram:** The human understanding of sound in terms of frequency can vary wildly by the impression of the frequency, and nature doesn't perceive sounds linearly. This is the reason why the Mel Scale was developed. Its crux is it considers the Decibel Scale when dealing with amplitudes (how loud) and logarithmic scale for frequencies (pitch). Put the voice features that are all stored in the Mel Spectrogram. We can see that the Mel-spectrogram provides a much clearer picture measured in decibels and optimized for input into our TTS. If you want a deeper mathematical understanding check out this great post [here](#).

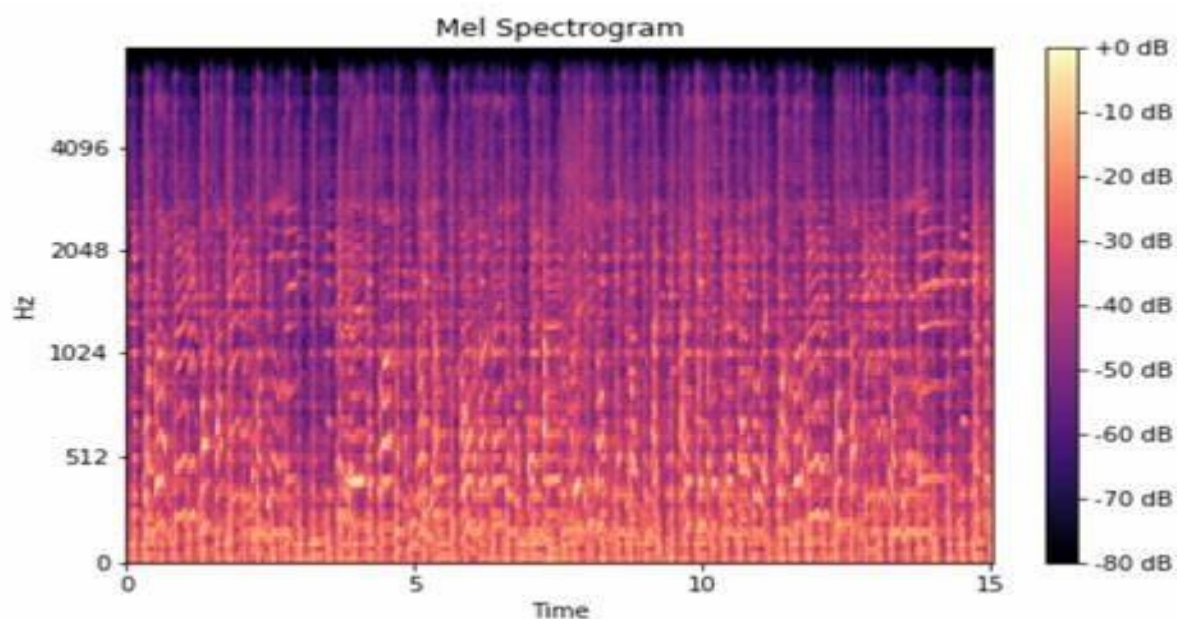


Figure 2-4 Mel-spectrogram

### AVD system:

A cascaded system shown Figure 2.5. A cascaded system refers to a sequential pipeline of different tasks or modules that work together to achieve the final goal. Each model in the cascade performs a specific task, and the output of one model serves as the input to the next model.

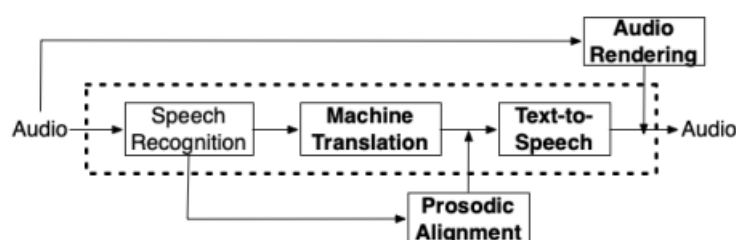


Figure 2-5 Cascaded system for speech-to-speech translation

In AVD, a cascaded system consists of three main modules: Automatic Speech Recognition (ASR), Neural Machine Translation (NMT), and Text-to-Speech (TTS).

## 2.2 Automatic Speech Recognition (ASR)

Automatic Speech Recognition (ASR) is a technology that transforms raw audio into a sequence of corresponding words. ASR strategy is summarized in Figure 2.6.

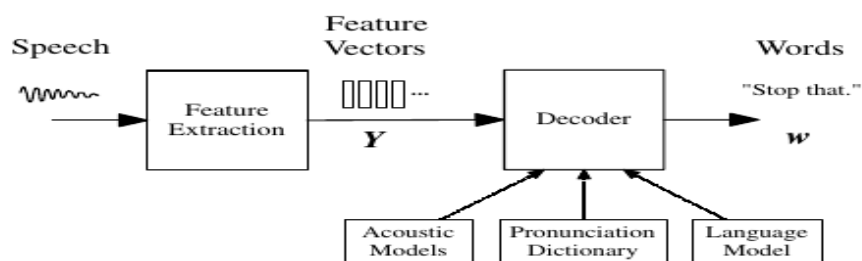


Figure 2-6 ASR component

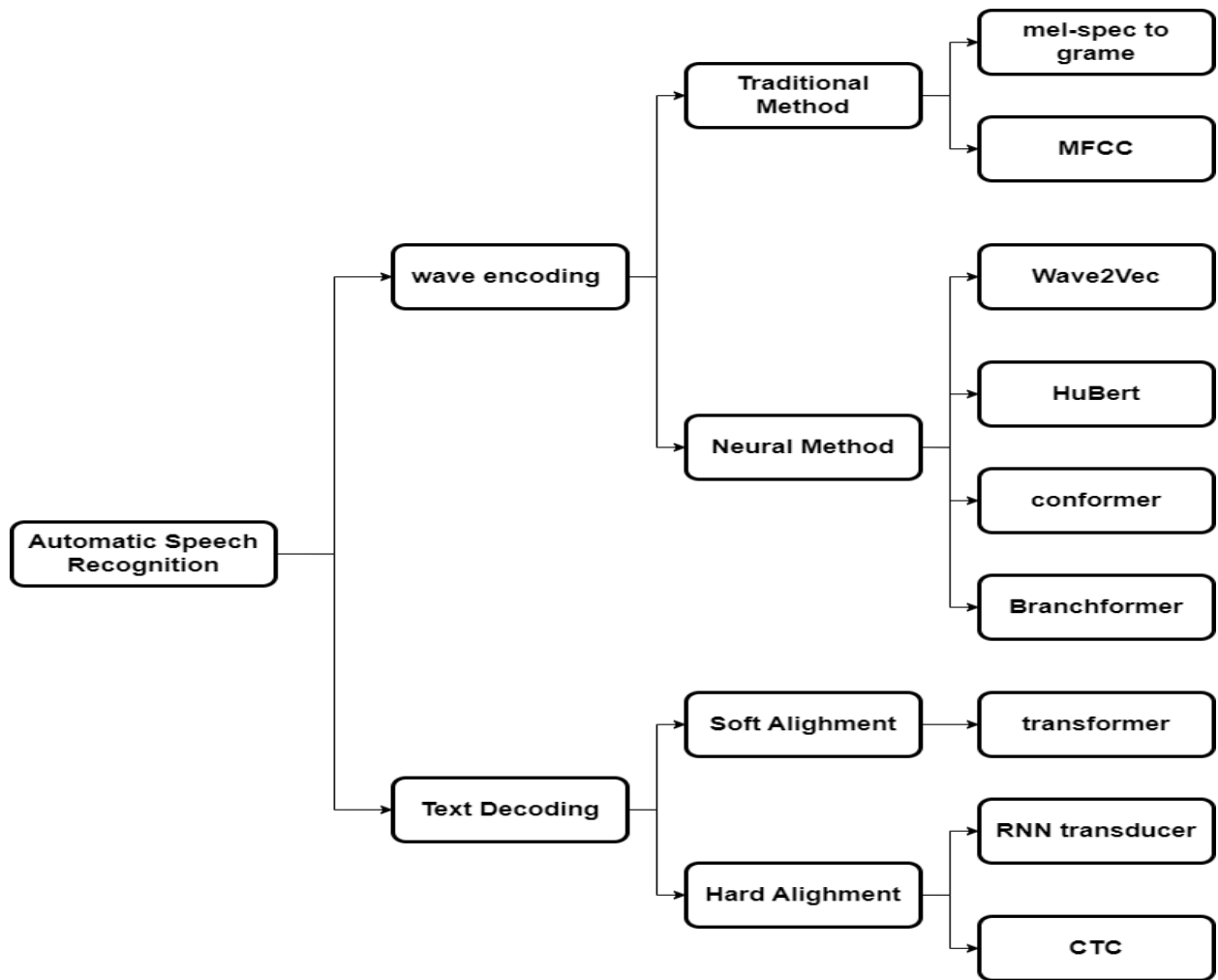


Figure 2-7 Speech Recognition history

### 2.2.1 wave encoding

refers to the process of converting speech audio signals into a suitable format for analysis and processing by a speech recognition system. This process involves transforming the continuous analog waveform of speech into a digital representation that can be fed into the speech recognition algorithm. There two methods: traditional methods & neural methods:

a) **traditional methods**: like Mel-spectrogram and MFCC. Mel-spectrogram is a representation of an audio signal that provides information about the frequency content of the signal over time.

b) **neural methods**: that use neural network models like Wave2vec (Alexei Baevski, 2020 ), HuBert (Wei-Ning Hsu, 2021), conformer (Anmol Gulati, 2020), BranchFormer (Yifan Peng, 2022), all these methods implemented with transformer encoder architecture with sum modification and trained with self-supervised learning.

c) **Wave2vec**: representation learning model that can extract meaningful representations from raw audio waveforms without the need for any transcription or linguistic annotations. Wave2Vec is designed to learn speech representations in a self-supervised manner. leverages the principles of self-supervised learning and contrastive predictive coding (CPC) to learn useful representations from audio data. First, use feature Extraction block with Convolutional Neural Networks and convert the raw audio to N vectors of size d, then inspired by masked language pre-training in Bert model we mask 15% of the vectors and force the model to predict them based on context. Wave2Vec learns to capture the underlying patterns and structure in the audio.

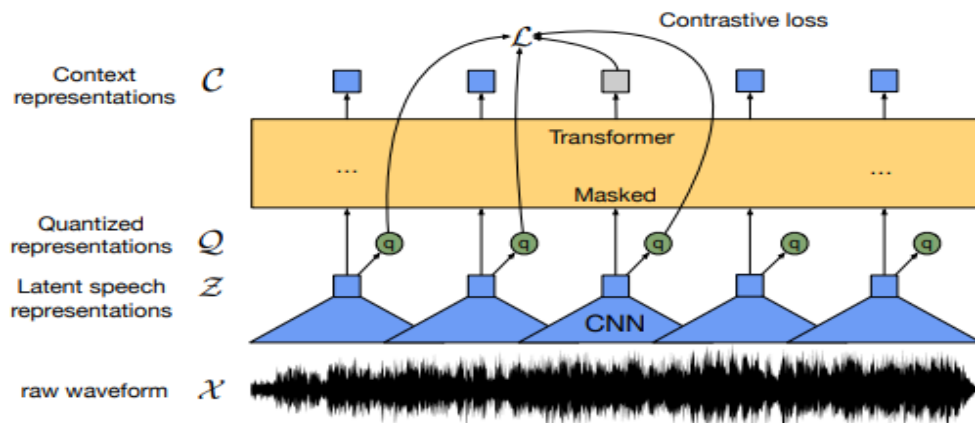


Figure 2-8 Wave2Vec2.0 model

i. **HuBert**: (Hidden unit BERT) is a self-supervised learning model designed for speech representation learning. It leverages the BERT (Bidirectional Encoder Representations from Transformers) architecture and incorporates offline clustering to generate noisy labels for pre-training. HuBert builds targets via a separate clustering process. Its re-uses embeddings from the BERT encoder to improve targets.

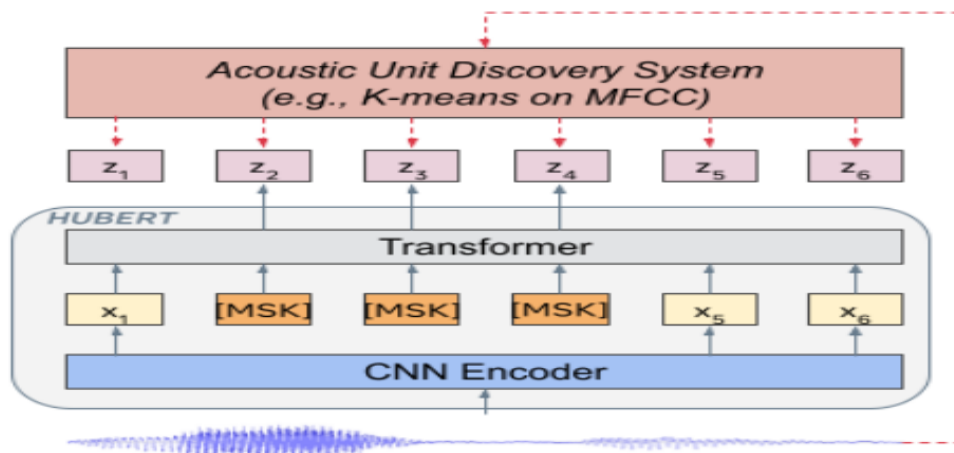


Figure 2-9 Hubert model

- ii. **Conformer**: The conformer is an architecture used for end-to-end speech recognition tasks. It combines convolutional neural networks (CNNs) and Transformer models to capture both local and global dependencies in the input audio signals.

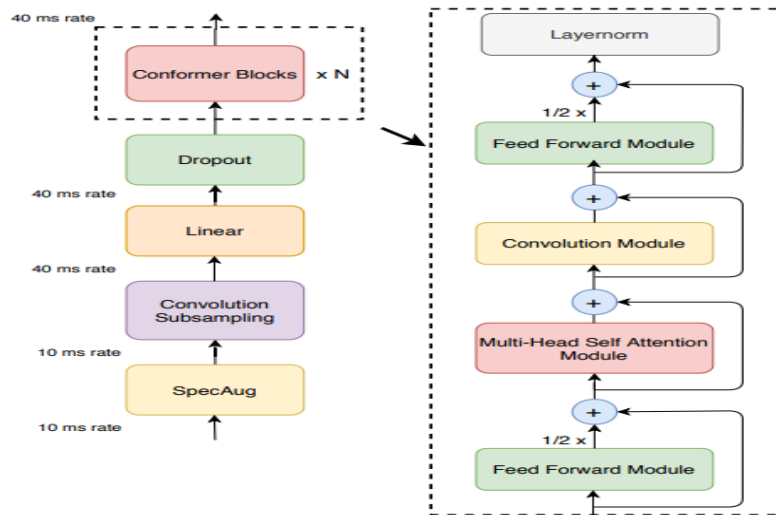


Figure 2-10 Conformer model

- iii. **Branchformer**: Conformer has proven to be effective in many speech processing tasks. It combines the benefits of extracting local dependencies using convolutions and global dependencies using self-attention. Inspired by this, the Authors propose a more flexible, interpretable, and customizable encoder alternative, Branchformer, with parallel branches for modeling various ranged dependencies in end-to-end speech processing. In each encoder layer, one branch employs self-attention or its variant to capture long-range dependencies, while the other branch utilizes an MLP module with convolutional gating to extract local relationships.

### 2.2.2 Text decoding

Is the process of generating human-readable text from taking learned representations or a sequence of Embedding Vectors and generating a sequence of tokens that form coherent and meaningful text. There is two abroach for text decoding:

- a) **Soft alignment**: The main purpose of the Soft Alignment model is to enable the model to focus on specific parts of the input sequence that are most relevant to the current step of processing. This is particularly useful in tasks where the input sequence length can vary, and the model needs to selectively attend to different parts of the sequence for generating accurate and meaningful output like transformers.

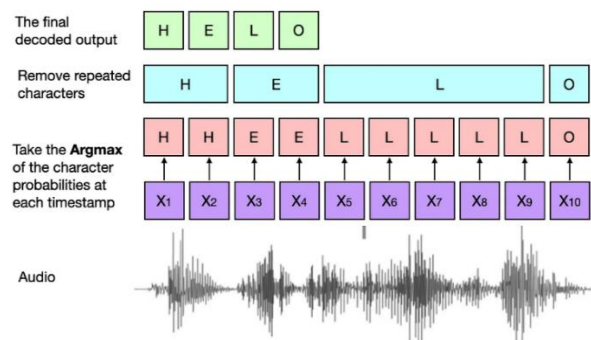


Figure 2-11 Text Decoding



b) **Hard alignment:** Unlike soft Alignment we force the model to monotonically alignment between text and wave Embedding. Approaches used in Hard alignments like:

- i. **RNN transducer:** The RNN Transducer model is based on recurrent neural networks (RNNs) and operates in an autoregressive manner, meaning it predicts output tokens one at a time while considering the entire input sequence. It does not rely on an explicit alignment between the input and output sequences, making it suitable for tasks where the alignment is not readily available, such as speech recognition.
- ii. **Connectionist Temporal Classification:** The CTC model takes an acoustic input (usually in the form of audio spectrograms) and produces a sequence of phonemes or characters as the output. The model operates in a "blank" and "label" framework, where the blank symbol represents the absence of any label at a particular time step, and the label symbol represents the output classes (phonemes or characters). The CTC model outputs a probability distribution over all possible label sequences given the input.

## 2.3 *Machine Translation*

Translation on a basic level is a substitution of words in one language for words in another, but that alone rarely produces a good translation because recognition of whole phrases and their closest counterparts in the target language is needed. Not all words in one language have equivalent words in another language, and many words have more than one meaning. Machine translation is the process of using artificial intelligence to automatically translate text from one language to another without human involvement. Modern machine translation goes beyond simple word-to-word translation to communicate the full meaning of the original language text in the target language. It analyzes all text elements and recognizes how the words influence one another. Machine translation techniques summarized in Figure 2.12.

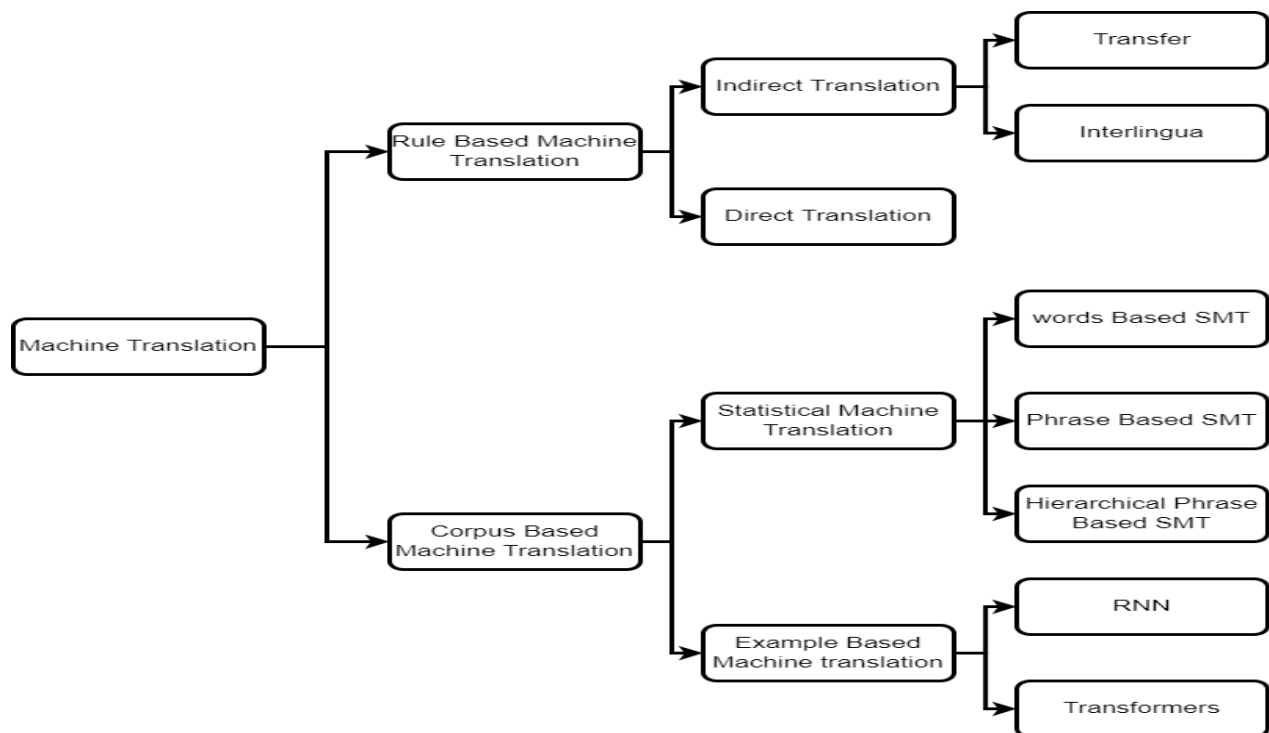


Figure 2-12 Summary of Machine translation

### 2.3.1 Rule-based machine translation (RBMT)

Is a machine translation approach based on hardcoded linguistic rules. RBMT systems are based on linguistic information about source and target languages retrieved from dictionaries and grammar covering the main semantic, morphological, and syntactic regularities of each language respectively. Monolingual and bilingual dictionaries are used to map input words to output words, and rules describe the grammatical structure of the input and output languages. It is divided into indirect translation and direct translation.

a) **Indirect translation:** a translation from a translated version, or multiple translated versions, of the ultimate source text. For instance, if a text in Arabic is translated into Portuguese via English the result is what we call an indirect translation.

b) **Direct translation** aims to directly translate a source sentence to a target sentence without relying on an intermediate language. This approach contrasts with the traditional phrase-based or statistical machine translation (SMT) methods that often involve multiple stages and intermediate representations.

### 2.3.2 Corpus Based Machine Translation

Corpus-based machine translation (CBMT) is an approach to machine translation that relies on large collections of texts, known as corpora, to build translation models. It can use statistical methods or based on examples:

a) **Statistical methods machine translation (SMT)**: relies on statistical models to generate translations based on patterns and probabilities. SMT systems were widely used in the field of machine translation before the emergence of neural machine translation (NMT) models.

b) **Neural machine translation (NMT)**: an approach to machine translation that utilizes neural networks, particularly deep learning models, to translate text from one language to another by using Encoder-Decoder architecture Figure 2-13. NMT has gained significant attention and popularity in recent years due to its ability to generate fluent and contextually accurate translations. example models:

- i. **RNN-based**: both the Encoder and Decoder are RNN units or it's variant. The Encoder takes the source lang and produces context vector for it and then the decoder try to generate the target lang based on the source context vector and the previous generated words.

To address the vanishing gradient problem, more advanced variants of RNNs have been developed, such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU).

These variants incorporate gating mechanisms that allow the network to

control the flow of information and selectively update the hidden state. LSTMs and GRUs have been successful in capturing long-range dependencies and have become the standard choice for many sequence modeling tasks.



Figure 2-13 Encoder Decoder Archtitecture

- ii. **Transformer-based**: Attention mechanism in both Encoder and Decoder (Ashish Vaswani, 2017) which allows the model to capture relationships between different positions in the input sequence. Unlike recurrent neural networks (RNNs) that process sequences sequentially, the Transformer model can parallelize computation across the entire sequence, making it more efficient. The transformers architecture shown in Figure 2-15:

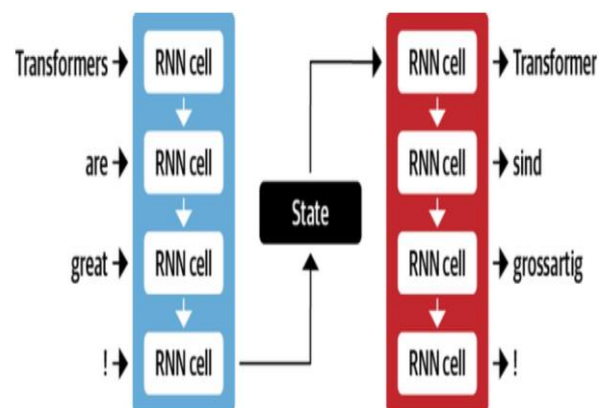


Figure 2-14 RNN Encoder-Decoder

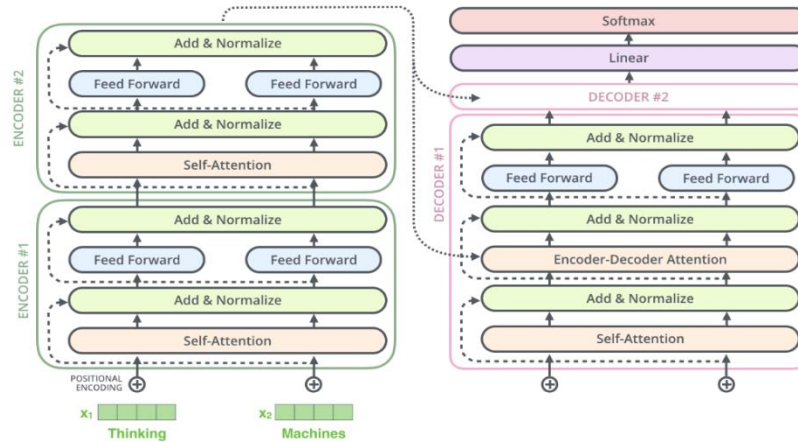


Figure 2-15 Transformer based Encoder-Decoder

## 2.4 Text to Speech

Text to Speech (TTS), which aims to synthesize intelligible, natural, and high-quality speech from the input text, has seen tremendous progress in recent years. Specifically, the prevalent methods have shifted from concatenative synthesis parametric synthesis to end-to-end neural network-based synthesis, where the quality of the synthesized speech is improved by a large margin and is close to that of the human counterpart. The general paradigm of end-to-end neural network-based methods usually first generate the acoustic feature (e.g., Mel-spectrogram) from text using encoder-decoder architecture autoregressively or non-autoregressively, then reconstruct the waveform signal using vocoder. When it comes to the recent multi-speaker TTS system, the speaker embedding is often extracted using speaker verification system, and is fed to the decoder of the TTS system in order to encourage the model to obtain a timbre inclination for the speaker of interest. Text to Speech (TTS) techniques summarized in Figure 2-16.

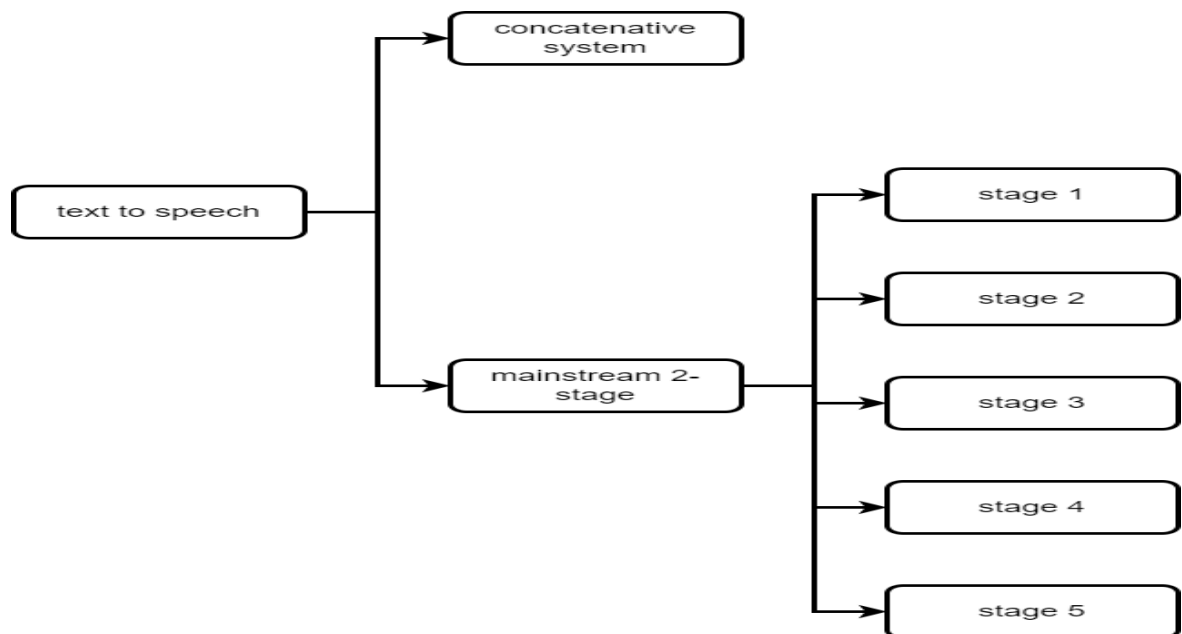


Figure 2-16 Text to Speech summary

### 2.4.1 Concatenative TTS

Concatenative speech synthesis is a Traditional old school technique that uses a stored speech database where speech is mapped to specific words. Concatenative speech generates synthetic speech by concatenating small units of pre-recorded speech, typically known as "units" or "phonemes," to form complete utterances.

The architecture of a concatenative speech synthesis:

**Text Analysis:** The input text is processed to identify linguistic information such as phonetic transcriptions, prosody, and other linguistic features required for speech generation.

**Unit Selection:** A database of pre-recorded speech units is created, which includes many phonemes or smaller units. During unit selection, the system determines which units to concatenate based on the desired linguistic output.

**Unit Concatenation:** The selected units are concatenated in a specific order to generate the desired speech output. This concatenation process ensures that the transitions between units are smooth and natural, providing a high-quality synthetic speech output.

**Prosody Modification:** Prosody refers to the patterns of stress, intonation, and rhythm in speech. Concatenative speech synthesis systems often incorporate techniques to modify the prosody of the synthesized speech to match the desired speaking style, such as adjusting pitch, duration, and stress patterns.

**Signal Processing:** The concatenated units are usually processed using various signal processing techniques to enhance the naturalness and quality of the synthesized speech. This may involve applying filters, pitch modification, or other techniques to shape the spectral and temporal characteristics of the speech signal.

**Voice Synthesis:** It is the final step. It involves generating the synthetic speech waveform from the concatenated units and applying any necessary post-processing, such as adding background, noise reduction, voice equalization, or other audio enhancements.

### 2.4.2 Mainstream 2-Stage

TTS has evolved from concatenative synthesis to parametric synthesis to neural network based. A hybrid parametric TTS approach that relies on a Deep Neural Network consisting of an acoustic model and neural vocoder to approximate the parameters and relationship between input text and the waveform that make up speech.

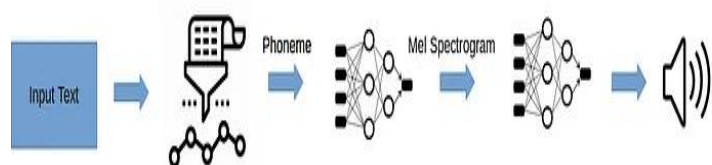


Figure 2-17 A basic high-level overview of mainstream 2-Stage TTS System

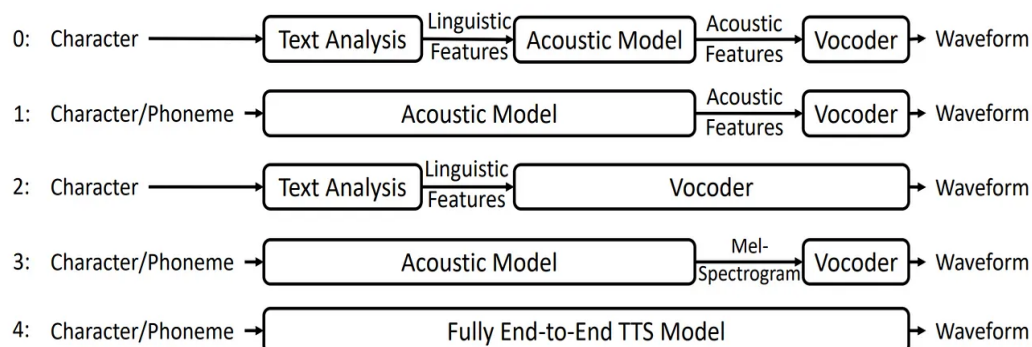
A basic high-level overview of mainstream 2-Stage TTS System at Figure 2-17:

- i. **Text Preprocessing and Normalization:** Simply the precursor step for the input text. It will be converted into the target language linguistic features in the form of a vector input into the acoustic model. This is done through normalization and converting to phonemes via grapheme-to-phoneme conversion.
- ii. **Acoustic Model:** Algorithms are optimized to convert the preprocessed/normalized text into Mel-spectrograms as output. The algorithms you need to convert the linguistic features vector to acoustic features to a Mel-Spectrogram. The Spectrogram ensures that we have now accounted for all relevant audio features.
- iii. **Neural Vocoder:** The input for the final step is the Mel-Spectrograms that are translated into a waveform via the Neural Vocoder. While there are many different types of neural vocoders, the modern ones today have a GAN foundation. we broke down the Mainstream 2-Stage method, as can be reviewed in Figure 2-18.

The current mainstream 2-Stage (Acoustic Model + Vocoder) neural network-based models have significantly improved the quality of synthesized speech. Prominent methods will first generate Mel-spectrogram from text and then synthesize speech from Mel-spectrogram using a neural vocoder such as WaveNet. There is also currently an evolution towards a next-generation fully End-to-End TTS model that we'll discuss.

## 2.5 Disadvantages of cascaded system

Figure 2-18 TTS Mainstream 2-stage



Some of disadvantages of cascaded system in speech-to-speech translation:

- 1) **Error Propagation:** Errors made in one stage can propagate in the next stages. If the ASR component makes mistakes in recognizing the speech, those errors impact the machine translation and TTS model, and if the ASR recognize the speech but machine translation model

make mistake there propagate in TTS model, also the ASR and machine translation be good but TTS not so the result won't be accurate. So, this error propagation can lead to compounded inaccuracies and reduced overall translation quality.

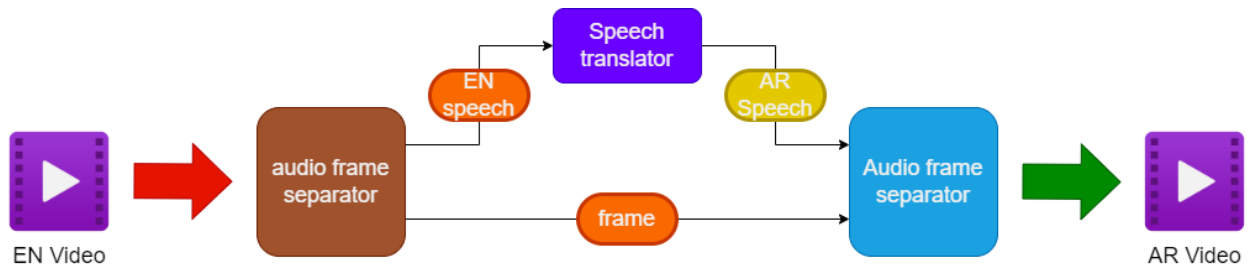
- 2) **Lack of End-to-End Optimization:** In a cascaded system, the ASR and MT components are **trained** independently, optimizing each component separately. This approach may not fully capture the interactions and dependencies between the two stages, potentially leading to suboptimal performance. The lack of end-to-end optimization can limit the ability of the system to learn and improve the joint modeling of speech and translation.
- 3) **Complexity and Maintenance:** Cascaded systems tend to be more complex and require the integration of multiple components, each with its own set of parameters, training procedures, and **resource** requirements. This complexity can make the system harder to maintain, develop, and update. It also requires expertise in ASR, MT, and TTS domains, making it challenging to optimize and improve the overall system.
- 4) **Longer Latency:** The cascaded approach involves sequential processing, where speech recognition is performed first, then machine translation, then TTS. This sequential nature can introduce additional latency in the translation process, as it requires waiting for the completion of ASR before initiating the translation stage then convert translated text to speech. For real-time or interactive applications, this latency can be a disadvantage, especially when quick and immediate translation is required.
- 5) **Resource Requirements:** Cascaded systems typically require separate training data for ASR, MT, and TTS, which can pose challenges in terms of data collection and availability. Acquiring and maintaining large-scale parallel corpora for training both components can be time-consuming and expensive. This is particularly true for less-resourced languages like Arabic.

## *Chapter 3*

### **3 Methodology**

In the proposed methodology, the objective is to obtain an Arabic video from an English video. This process involves several steps. Initially, the English video is passed through a block that separates the audio and video frames. The separated audio is then fed into a speech translator, which converts the English audio into translated Arabic speech. Simultaneously, the video frames are preserved for further processing. The translated Arabic speech and the preserved video frames are merged together

in a merging block. This merging process combines the translated speech with the corresponding video frames, resulting in the creation of an Arabic video. By following this methodology, the desired outcome of obtaining an Arabic video from an English video can be achieved effectively. This process is shown in figure 3-1.



*Figure 3-1 AVD System*

While the speech translator block that described previous figure in the pipeline utilizes the existing cascaded system described in Chapter 2, we can further improve the pipeline by incorporating two additional models. The first model is specifically designed to add punctuation to the generated English subtitles, as punctuation plays a crucial role in enhancing the quality of translated Arabic text. The second model is dedicated to adding Tashkeel (diacritical marks) to the Arabic text, which significantly improves the quality of the resulting Arabic speech. By integrating these two models into the existing pipeline, we aim to achieve a more consistent and comprehensive approach to generating Arabic videos from English videos.

The modified pipeline introduces the following two models:

1. **Punctuation Model:** This model analyzes the Arabic text and incorporates appropriate punctuation to improve the translation quality in English subtitles.
2. **Tashkeel Model:** This model processes the Arabic text after translation and adds Tashkeel (diacritical marks) to ensure accurate pronunciation and clarity in the generated Arabic speech.

The revised pipeline retains the original three stages but includes the Punctuation Model before the machine translation step and the Tashkeel Model after the machine translation stage. By integrating these additional models, we aim to improve the overall accuracy, fluency, and comprehension of the generated translations and speech by addressing punctuation and diacritical marks in both Arabic and English texts.



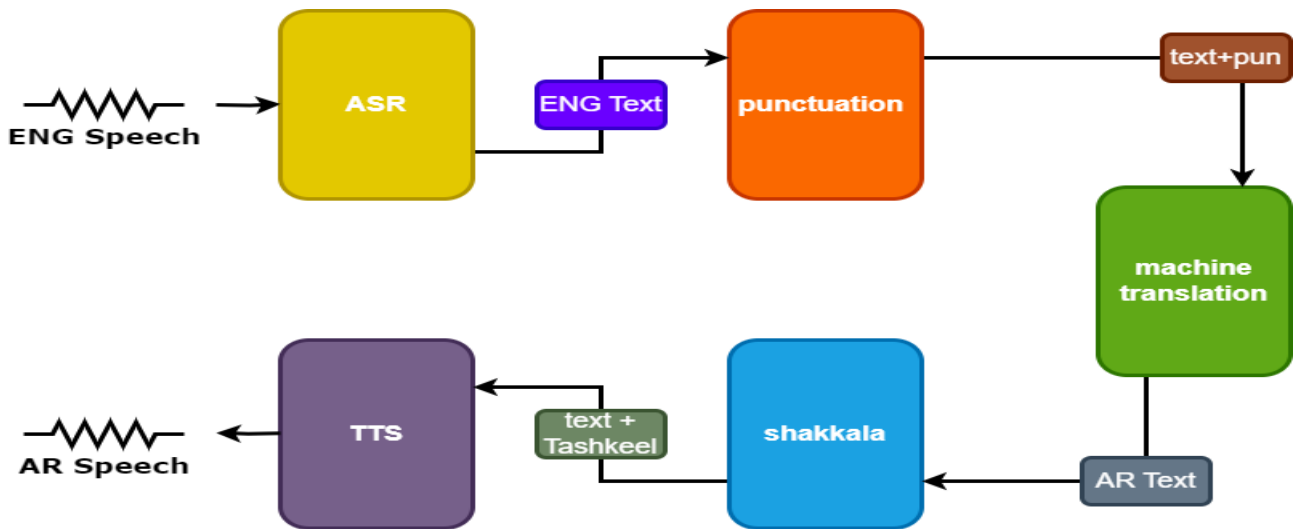


Figure 3-2 speech translator block

### 3.1 speech recognition

Automatic Speech Recognition (ASR) technology converts audio to text by training a system on a dataset of speech samples and transcriptions. We try two solution to solve this problem:

- 1) Google ASR (Automatic Speech Recognition)
- 2) Wave2Vec2.0 with finetune

1) **Google ASR** (Automatic Speech Recognition) is a closed-source model developed by Google to address the ASR problem and is used in their production applications. There is no available information regarding the model's training process or the specific architecture used. We can access it through the Google API. Its advantages include outstanding performance, continuous improvement through updates, and seamless integration with other Google services. However, potential privacy concerns related to data usage and the requirement of an internet connection for real-time processing can be considered as disadvantages.

2) **Wave2Vec2.0** with fine-tuning utilizes Transformer-based neural networks for speech processing, employing a training objective similar to BERT's masked language modeling. By pre-training on unlabeled speech data and subsequently fine-tuning on a smaller labeled dataset, Wave2Vec2.0 enables efficient self-supervised training. Remarkably, it outperforms previous systems by achieving superior results using just one hour of labeled data, compared to the requirement of 100 times more labeled data previously. The

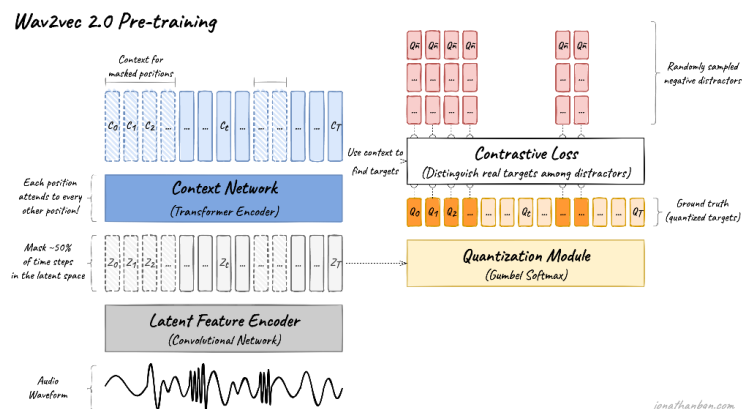


Figure 3-3 Wave2vec pretrain architecture

model begins with self-supervised pre-training, aiming to minimize two objectives. Firstly, a masked language model objective similar to the one used in BERT. Secondly, a contrastive loss is employed between the context vector generated from the transformer encoder and its corresponding quantized vector before the encoder. During the fine-tuning process, a linear layer is utilized as a language model with the number of hidden units equal to the vocabulary size. The objective for fine-tuning is the CTC (Connectionist Temporal Classification) loss.

Overall arch ...figure

### 3.1.1 Wave2vec2.0 architecture

- 1) **feature encoder:** The feature encoder in wav2vec 2.0 reduces audio dimensionality by converting the waveform into feature vectors through a 7-layer convolutional neural network with 512 channels per layer. Before inputting to the network, the waveform is normalized. The feature encoder's convolutional layers have decreasing kernel width and strides. It has a receptive field of 400 samples or 25MS at a 16 kHz sample rate.

Wave2vec 2.0 Latent Feature Encoder

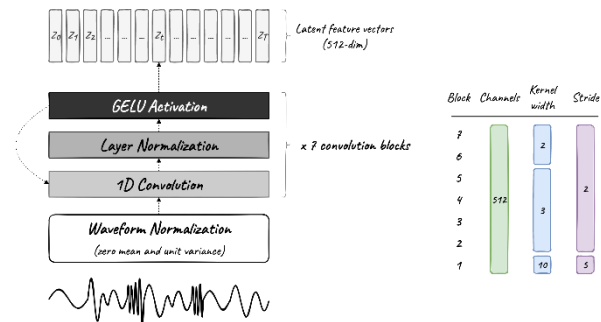


Figure 3-4 wave2vec Feature Encoder

- 2) **The context network:** The context network in wav2vec 2.0 employs a Transformer encoder architecture to process the latent feature vectors. It consists of multiple Transformer blocks, with the number varying based on the model size (12 for BASE and 24 for LARGE). A feature projection layer increases the dimension from 512 to 768 for BASE or 1,024 for LARGE. The Transformer blocks capture contextual information and dependencies among the latent features.

Wave2vec 2.0 Context Network (Transformer Encoder)

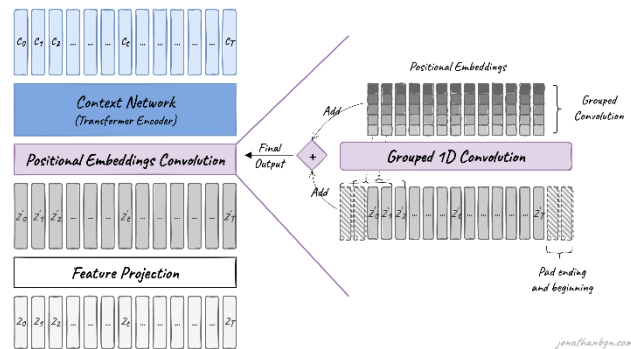


Figure 3-5 Wave2vec Context Network

- 3) **Quantization Module:** Using Transformers for speech processing faces a challenge due to the continuous nature of speech. Unlike written language, which can be discretized into finite vocabulary units like words or sub-words, speech lacks such natural sub-units. While phones can be used as discrete

Wave2vec 2.0 Quantization Module

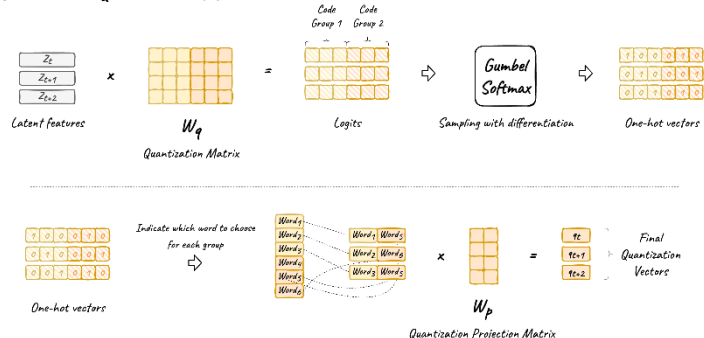


Figure 3-6 Wave2vec Quantization Module

units, labeling the entire dataset beforehand by humans would be necessary, preventing pre-training on unlabeled data. Wav2vec 2.0 learns discrete speech units by sampling from the Gumbel-SoftMax distribution. Codewords from codebooks are concatenated to form units. With 2 groups and 320 words in each, a maximum of 102,400 speech units is possible. The latent features in wav2vec 2.0 are multiplied by a quantization matrix to produce logits, representing scores for possible codewords in each codebook. The Gumbel-SoftMax trick enables differentiable sampling of codewords based on converted probabilities, introducing controlled randomness with a temperature parameter for training and codeword utilization.

### 3.1.2 Pre-train wave2vec

In the pre-training process of wav2vec 2.0, unlabeled speech data is trained using a contrastive task. This involves randomly applying a mask to approximately 50% of the projected latent feature vectors in the latent space. The masked positions are then replaced by the corresponding trained vector Z'M. These modified vectors are subsequently passed through the Transformer network, enabling the model to learn contextual representations from the unlabeled data. In wav2vec 2.0, the context vectors are projected and matched in dimension with the quantized speech units. For each masked position, 100 negative distractors are sampled. The model computes cosine similarity between the projected context vector and the true positive target, as well as negative distractors. The contrastive loss promotes similarity with the true positive target and penalizes similarity with negative distractors.

### 3.1.3 CTC decoder

We finetune with CTC decoder.

1. The initial component of Wav2Vec2 comprises CNN layers that extract acoustically meaningful but contextually independent features from raw speech. These layers are frozen during fine-tuning since they are already adequately trained in pre-training. Linear layer output matches vocabulary size for character classification.

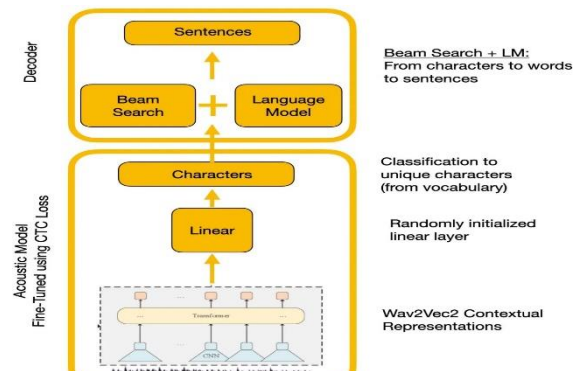
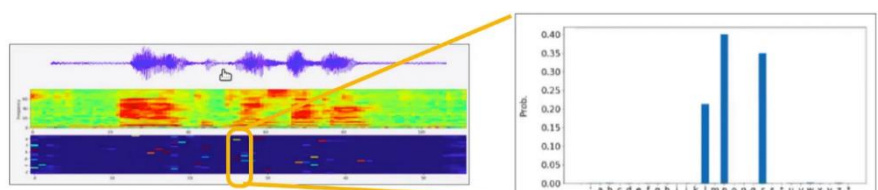


Figure 3-7 Wave2vec Finetune

2. The Challenge of Decoding: Now that we have a fine-tuned STT model, how do we get the transcribed text for a speech sample?

Given an audio sample, the audio is sliced into evenly spaced chunks of time. These chunks are passed through the fine-tuned acoustic model, which gives a per-timestamp probabilities matrix. The acoustic model predicts character probabilities for each time slice.

The image below shows an audio signal (top row), transformed into a spectrogram (middle row) and the acoustic model's per-timestamp output as a heat-map (bottom row). Each heat-map time slot can be



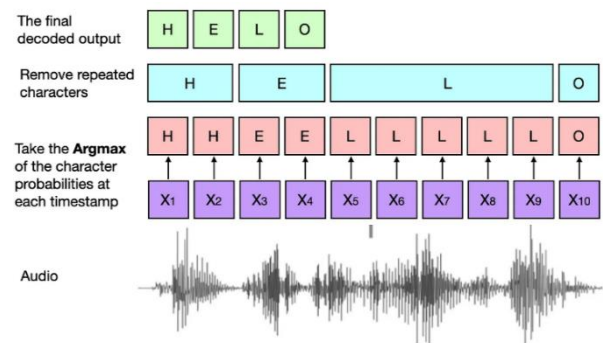
viewed as a histogram of the character probabilities.

*Figure 3-8(Character probabilities predicted for each audio slice)*

3. **Naive Decoding:** Our first, naive decoding solution: For each audio slice, choose the most probable character using the Argmax function. What are the problems with the naive decoder?

i. STT model output has more timestamps than transcribed characters, resulting in excess predicted characters.

ii. Collapsing duplicated consecutive characters can pose challenges for words like "HELLO." Additional techniques are needed to handle such cases and ensure accurate representation of words with repeated characters.



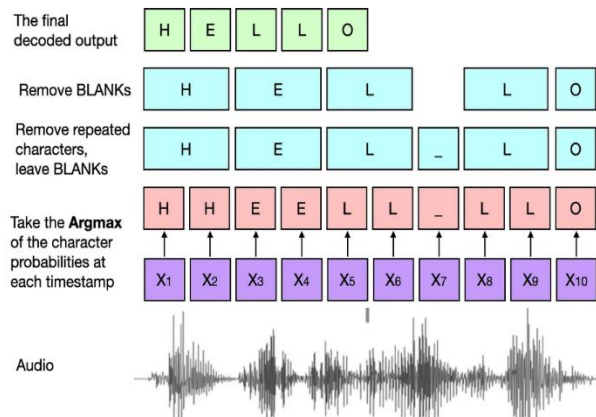
*Figure 3-9 Naive Decoding*

### 3.1.4 CTC Algorithm

CTC stands for Connectionist Temporal Classification, and it is an algorithm, encoding method, and a loss function.

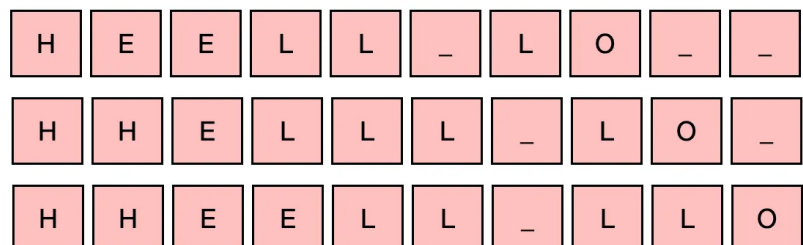
How can we use CTC encoding to solve the problems described above?

CTC adds a BLANK symbol to the possible predicted character set. This BLANK can be used in the predicted output when We do not want to transcribe a character (when there is silence in the audio) and between consecutive characters that should not be collapsed. Note that the same final output could be decoded from multiple.



*Figure 3-10 CTC Decoding*

CTC decoded predictions (see figure below). We will come back to this in the next section.



*Figure 3-11(Many legitimate alignments produce the same output)*

### 3.1.1 Limitation of CTC and the naïve Argmax:

The naïve Argmax + CTC decoding algorithm described above is not an optimal solution for two reasons:

- i. **problem 1:** When a decoder has the capability to choose multiple character options per time step instead of solely relying on the argmax character, it can produce various potential output sentences.

The argmax output, which is the most likely according to a standard decoding approach, may not align with the most likely collapsed output string. For instance, if the decoder generates "AB\_", "B\_B," and "BB" as the top three outputs with associated probabilities, combining the probabilities of the two "BB" outputs reveals a higher likelihood than that of the argmax output "AB\_." This demonstrates that considering alternative options through a less greedy decoding strategy is crucial to capture the most probable collapsed output.

	Output	Output Probability
Most probable output	[ A B _ ]	0.3
Second most probable output	[ B _ B ]	0.25
Third most probable output	[ _ B B ]	0.1

Figure 3-12 Decoding Problem

- ii. **Problem 2:** Since the decoded output comes from an acoustic model, the output sentence may end up having:
  - Misspelled words
  - Something that is not a word (“phor” instead of “for”)
  - A word that sounds that same but has a completely different meaning (bear vs. bare or knot vs. not)

**The solution:** Score every possible path in the character probabilities matrix then Combine the scores of equivalent paths finally Choose the path with the highest combined score as the final output.

As a fast approximate solution, we use Beam-Search. Beam-search is a (breadth-first) heuristic search algorithm that explores a graph by expanding the most promising nodes in a limited set.

1. Start by taking the N best characters from the first time slice and keep their probabilities. These are the N initial beams.
2. In the next time step, try to add a second character to each of the beams. They are scored by multiplying their character probabilities.

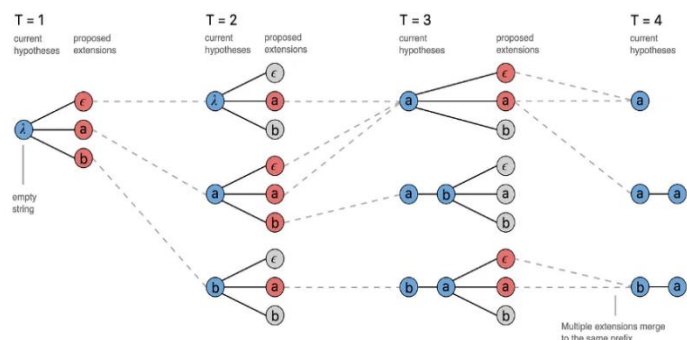


Figure 3-13 Beam Search



3. If multiple paths arrive at the same output, collapse them into one and add their scores.
4. Prune the number of beams kept down to N (according to their scores).
5. Repeat these steps for each of the time steps.

### 3.2 *machine translation*

GNMT is Google's Neural Machine Translation system that overcomes limitations of traditional methods. It uses a deep LSTM network with attention connections and wordpiece units to handle rare words. GNMT achieves better translation accuracy, reduces errors by 60% compared to phrase-based systems, and maintains competitive results on benchmarks.

GNMT, Google's Neural Machine Translation system, utilizes an architecture with an encoder network, a decoder network, and an attention module. The encoder has bi-directional and uni-directional layers with residual connections. The model is divided among multiple GPUs for faster training. It consists of 8 encoder LSTM layers (1 bi-directional and 7 uni-directional) and 8 decoder layers. During training, the bi-directional encoder layers compute in parallel, followed by the uni-directional layers on separate GPUs. The bottom decoder layer output is used for attention context, sent to all remaining decoder layers. The SoftMax layer is also partitioned across GPUs, depending on the output vocabulary size.

context  $a_i$  for the current time step is computed according to the following formulas:

$$s_t = \text{AttentionFunction}(y_{i-1}, x_t) \quad \forall t, \quad 1 \leq t \leq M$$

$$p_t = \exp(s_t) / \sum_{t=1}^M \exp(s_t) \quad \forall t, \quad 1 \leq t \leq M$$

$$a_i = \sum_{t=1}^M p_t \cdot x_t$$

where Attention Function in our implementation is a feed forward network with one hidden layer.

#### 3.2.1 *google machine translation Architecture.*

The model uses a sequence-to-sequence learning framework with attention. It consists of an encoder network, a decoder network, and an attention network. The encoder converts the source sentence into vectors, while the decoder generates symbols until the end-of-sentence symbol is reached, with the help of the attention module.

Let  $(X, Y)$  be a source and target sentence pair. Let  $X = x_1, x_2, x_3, \dots, x_M$  be the sequence of  $M$  symbols in the source sentence and let  $Y = y_1, y_2, y_3, \dots, y_N$  be the sequence of  $N$  symbols in the target sentence. The encoder is simply a function of the following form:

$$x_1, x_2, \dots, x_M = \text{EncoderRNN}(x_1, x_2, x_3, \dots, x_M)$$

The conditional probability of the sequence  $Y$  given  $X$  can be decomposed using the chain rule and fixed-size vectors.

$$P(Y|X) = P(Y|x_1, x_2, x_3, \dots, x_M)$$

$$= \prod_{i=1}^N P(y_i|y_0, y_1, y_2, \dots, y_{i-1}; x_1, x_2, x_3, \dots, x_M)$$

where  $y_0$  is a special “beginning of sentence” symbol that is prepended to every target sentence. During inference we calculate the probability of the next symbol given the source sentence encoding and the decoded target sequence so far:

$$P(y_i|y_0, y_1, y_2, y_3, \dots, y_{i-1}; x_1, x_2, x_3, \dots, x_M)$$

The decoder consists of an RNN network and a SoftMax layer, generating hidden states for symbol prediction. Deep encoder and decoder RNNs are crucial for accuracy, similar to deep LSTMs. The attention module and past decoder output are utilized.

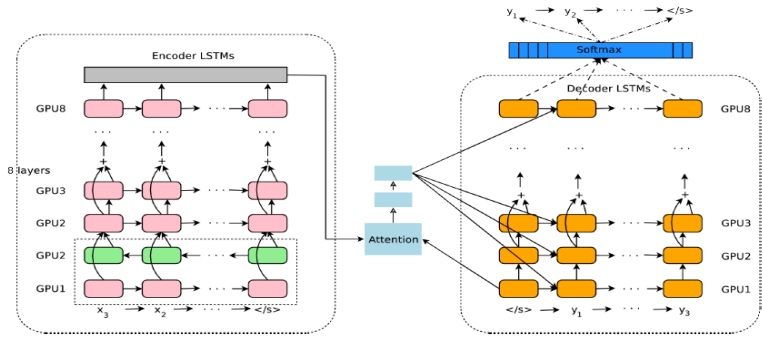


Figure 3-14 Google Machine Translation Architecture

### 3.2.2 Residual Connections

Deep stacked LSTMs improve accuracy, but training becomes challenging beyond 8 layers due to gradient issues. Simple stacked LSTM layers work well up to 4 layers, with diminishing performance beyond 6 layers and poor results beyond 8 layers.

The difference between normal stacked LSTM and our stacked LSTM with residual connections.

On the left: simple stacked LSTM layers. On the right: our implementation of stacked LSTM layers with residual connections. With residual connections, input to the bottom LSTM layer ( $x_{0i}$  's to LSTM1) is element-wise added to the output from the bottom layer ( $x_{1i}$  's). This sum is then fed to the top LSTM layer (LSTM2) as the new input.

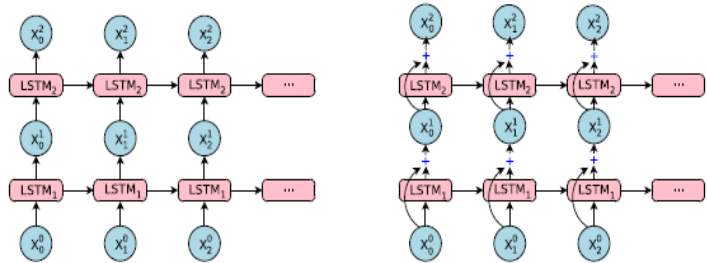


Figure 3-15 LSTM with Residual

Residual connections greatly improve the gradient flow in the backward pass, which allows us to train very deep encoder and decoder networks. In most of our experiments, we use 8 LSTM layers for the encoder and decoder, though residual connections can allow us to train substantially deeper networks.

#### Bi-directional Encoder for First Layer

For translation systems, the information required to translate certain words on the output side can appear anywhere on the source side. Often the source side information is approximately left-to-right, similar to the target side, but depending on the language pair the information for a particular output word can be distributed and even be split up in certain regions of the input side.

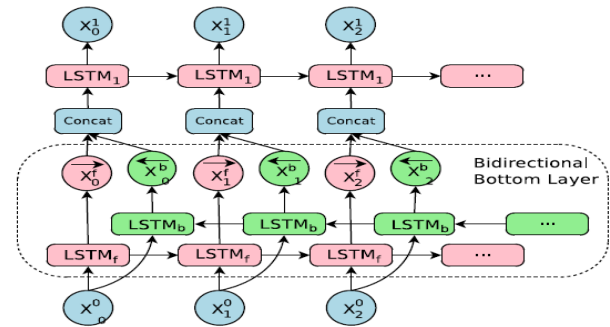


Figure 3-16 Residual Bidirectional LSTM

## 3.3 text to speech

### 3.3.1 Google Text to speech (WaveNet)

WaveNet is a deep neural network designed to generate high-quality audio waveforms. It demonstrates the ability to efficiently train on extensive audio datasets and surpasses other text-to-speech systems in terms of naturalness. WaveNet can mimic different speakers and even produce realistic music. Additionally, it exhibits promising capabilities in phoneme recognition as a discriminative model.

#### 3.3.1.1 The Causal Convolution Layer

is a crucial component of WaveNet. In the context of signal and system theory, a causal system relies on past and current inputs rather than future inputs. WaveNet maintains causality by generating the current

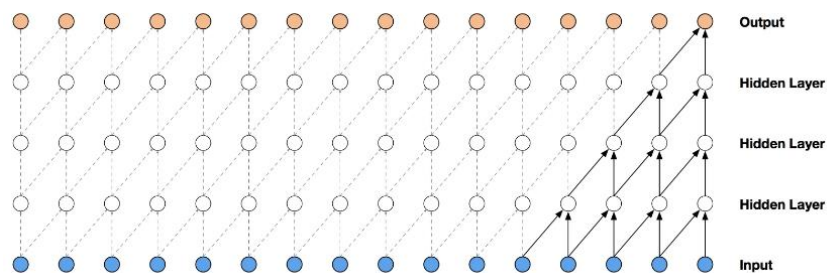


Figure 3-17 Causal Convolution Layer

acoustic intensity solely based on previous data. This approach preserves the autoregressive property and ensures the preservation of sample ordering within the network's architecture. For training of 1 output sample, 5 input samples are used. Receptive field of this network is 5.



$$p(x) = \prod_{n=1}^N p(x_N | x_1, x_2, \dots, x_{N-1}),$$

samples are denoted as  $x = (x_1, x_2, \dots, x_N)$ ,  $p(\cdot)$  is the probability.

The following equation is used for generation of new samples by predicting probability of next samples, given the probabilities of previous and current samples.

### 3.3.1.2 Dilated Convolution Layer

Dilated convolution, also known as convolution with holes or a-trous convolution, expands the receptive field efficiently by using a larger filter with zeros. Stacked dilated convolutions allow networks to have large receptive fields while maintaining input resolution.

Training one sample with dilated

convolution requires 16 inputs compared to 5 in causal convolution.

Each block in the model has a receptive field of size 1024, serving as an efficient and discriminative alternative to a  $1 \times 1024$  convolution. However, the model encounters difficulties when generating samples in silent input regions.

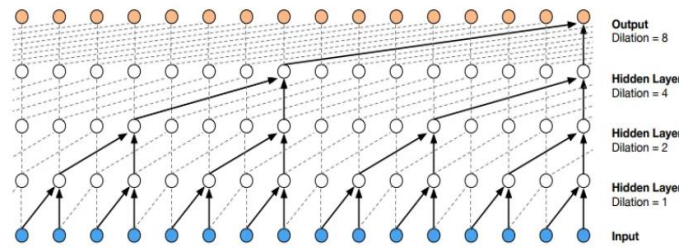


Figure 3-18 Dilated Convolution Layer

### 3.3.1.3 Gated Activation Units

Gated activation units' model complex operations with an equation:

$$z = \tanh(W_{f,k} * x) \cdot \sigma(W_{g,k} * x),$$

where  $*$  is a convolution operator,  $\cdot$  is an element wise multiplication operator,  $\sigma(\cdot)$  is the sigmoid activation function,  $k$  is the layer index and  $W_f$ , and  $W_g$ , are weight matrix of filters and gate respectively.

### 3.3.1.4 Residual block and Skip Connections

Residual blocks and skip channels, inspired by Pixel CNN, are utilized in the network to accelerate convergence and enable training of deeper models.

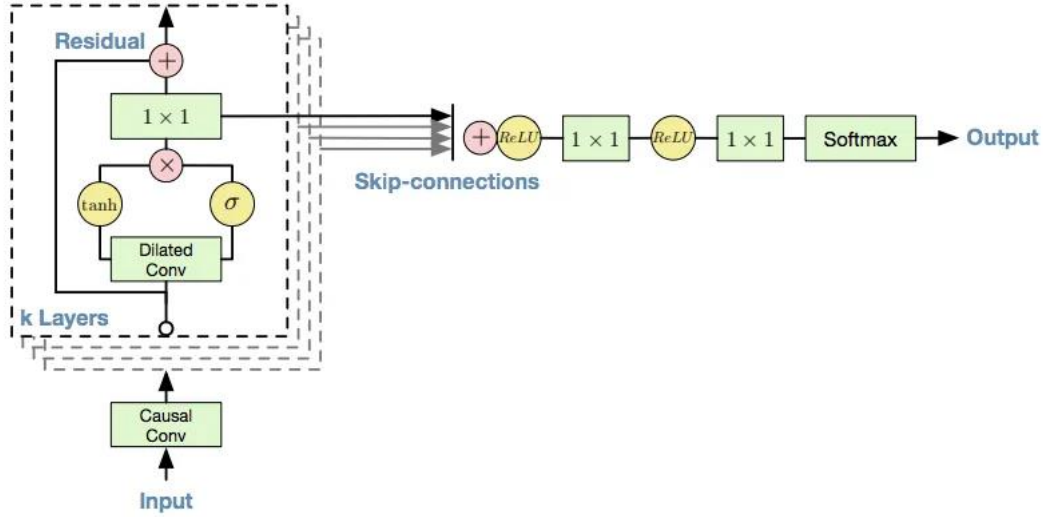


Figure 3-19 Overview of residual block and complete architecture

### 3.3.1.5 Global and Local Conditioning

When WaveNet model is conditioned on auxiliary input features (linguistic feature or acoustic feature), denoted by  $h$  (latent representation of features), it is represented as  $p(x|h)$ .

$$p(x) = \prod_{t=1}^T p(x_t | x_1, x_2, \dots, x_{t-1}, h)$$

Conditional Probability on auxiliary input features

WaveNet can be conditioned on input variables to control its audio generation. There are two types of conditioning: global and local. Global conditioning captures the speaker's identity and influences the output distribution across all timesteps using this equations:

$$z = \tanh(W_{f,k} * x + V_{f,k}^T h) \cdot \sigma(W_{g,k} * x + V_{g,k}^T h),$$

Where  $V_{*,k}$  is a learnable linear projection, and the vector  $V_{f,k}$   $h$  is broadcast over the time dimension.

Local conditioning in WaveNet captures the context and speaking style by incorporating the local features of speech. This is achieved through upsampling using transposed convolution or repeated sampling, as represented by this equation :

$$z = \tanh(W_{f,k} * x + V_{f,k} * y) \cdot \sigma(W_{g,k} * x + V_{g,k} * y),$$

Local conditioning for upsampling using transposed convolution

### 3.3.2 FastSpeech2

FastSpeech 2 is a non-autoregressive text-to-speech (TTS) model that synthesizes speech quickly while maintaining quality. It addresses the limitations of its predecessor by training directly with ground-truth targets, incorporating more variation information (duration, pitch, energy), and introducing FastSpeech 2s for parallel waveform generation. Experimental results show improved speed and voice quality compared to FastSpeech and autoregressive models.

The overall architecture for FastSpeech 2 and 2s. LR in subfigure (b) denotes the length. regulator proposed in FastSpeech. LN in subfigure (c) denotes layer normalization. many mapping problems, with simpler training pipeline and higher voice quality. At last, we extend FastSpeech 2 to FastSpeech 2s for fully end-to-end text-to-waveform synthesis.

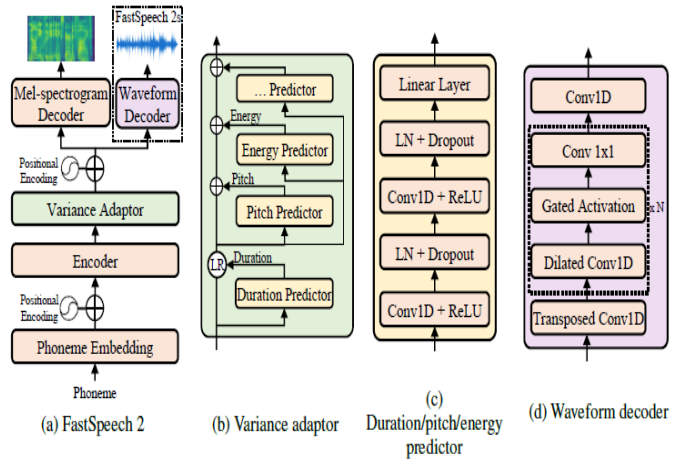


Figure 3-20 Fast Speech 2

#### Model overview.

FastSpeech 2 is an improved non-autoregressive text-to-speech model that eliminates the teacher-student distillation pipeline. It utilizes a variance adaptor, including duration, pitch, and energy predictors, to enhance the model's capability to handle the one-to-many mapping problem in TTS. Ground-truth mel-spectrograms are used as training targets instead of distilled ones, resulting in improved voice quality. The duration predictor uses more accurate phoneme durations obtained through forced alignment. Additionally, pitch and energy predictors provide additional variance information. FastSpeech 2s further simplifies the training pipeline by directly generating waveforms from text, without the need for cascaded mel-spectrogram generation and waveform generation stages. This approach achieves faster training and inference speeds while maintaining high voice quality.

### 3.4 Shakaala model

The model addresses the challenging task of adding diacritics to Arabic text and offers various applications, including speech synthesis and language learning.

#### Problem Overview:

The diacritization of Arabic text is a complex problem due to the unique characteristics of the Arabic language. This project aims to develop an automated diacritization system to assist fluent and non-fluent Arabic speakers in understanding and producing properly diacritized content.

#### Existing Systems Review:

This project involves a critical review of currently available systems, measures, and resources for Arabic text diacritization. Traditional rule-based approaches and closed-source tools are compared with the proposed neural Shakkala system. The aim is to identify the strengths and weaknesses of each approach and highlight the superiority of deep learning techniques.

#### ***Model Architecture:***

The proposed model leverages deep neural networks (DNNs) for Arabic text diacritization. DNNs have shown remarkable performance in various natural language processing (NLP) tasks and are expected to excel in this diacritization problem as well. The specific architecture and design choices of the model are detailed in this section.

#### ***Significance and Future Directions:***

The utilization of deep learning techniques for Arabic text diacritization showcases the immense potential for enhancing the accuracy and efficiency of diacritization systems. This graduation project contributes to the advancement of Arabic NLP research by addressing the diacritization problem and providing a benchmark dataset. Future work may focus on further improving the model's performance, exploring additional architectures, and expanding the dataset for comprehensive evaluations.

### ***3.5 punctuation model***

The Punctuation Model is an integral component of our graduation project, designed to enhance the accuracy and effectiveness of punctuation prediction within natural language processing tasks. This section aims to provide a comprehensive explanation of the Punctuation Model, its purpose, underlying principles, and implementation details. The Punctuation Model serves as an intelligent system that predicts and inserts appropriate punctuation marks within a given text. Its primary objective is to improve readability and understanding by enforcing grammatical structure and conveying the intended meaning more effectively. The Punctuation Model addresses the challenge of punctuation prediction, a crucial aspect of natural language processing. Accurate placement of punctuation marks is essential for proper interpretation and comprehension of written text. By automating this process, the model aims to reduce human effort, enhance the efficiency of content creation, and improve overall writing quality.

- a. **Punctuation Prediction:** The model utilizes machine learning techniques to predict suitable punctuation marks, such as periods, commas, question marks, exclamation marks, and more, based on the context of the input text.
- b. **Contextual Understanding:** The model considers the surrounding words, grammar, and syntactic structures to make accurate predictions and ensure coherence within the text.
- c. **Language Independence:** The Punctuation Model is designed to work across multiple languages, accommodating a wide range of text inputs and catering to diverse user requirements.

### ***Limitations:***

- a. **Ambiguity:** Punctuation prediction can be challenging in cases where the context is ambiguous or lacks clear grammatical cues. The model may occasionally make incorrect predictions in such scenarios.
- b. **Domain-specific Adaptation:** The model's performance may vary across different domains and writing styles. Fine-tuning or retraining may be necessary to achieve optimal results in specific domains.

## **3.6 Datasets**

We used the following datasets:

- LibriSpeech<sup>1</sup>: used for Pretraining Wave2Vec2.0, it is a collection of approximately 1,000 hours of audiobooks that are a part of the LibriVox project.
- MuST-C<sup>2</sup>: is a multilingual speech translation corpus whose size and quality will facilitate the training of end-to-end systems for SLT from English into several target languages.
- MGB-3<sup>3</sup>: Egyptian Arabic Speech recognition in the wild. Every sentence was annotated by four annotators. More than 15 hours have been collected from YouTube.

### **3.6.1 LibriSpeech DATASET**

The LibriSpeech corpus is a collection of approximately 1,000 hours of audiobooks that are a part of the LibriVox project. Most of the audiobooks come from Project Gutenberg. The training data is split into 3 partitions of 100hr, 360hr, and 500hr sets while the dev and test data are split into the 'clean' and 'other' categories, respectively, depending upon how well or challenging Automatic Speech Recognition systems would perform against. Each of the dev and test sets is around 5hr in audio length. This corpus also provides the n-gram language models and the corresponding texts excerpted from the Project Gutenberg books, which contain 803M tokens and 977K unique words.

---

<sup>1</sup> <https://paperswithcode.com/dataset/librispeech>

<sup>2</sup> [MuST-C En-AR | Kaggle](#)

<sup>3</sup> <https://arabicspeech.org/mgb3-asr-2/>

Table 1 Librispeech dataset analysis

	NUMBER OF TOKENS	NUMBER OF UNIQUE WORDS	NUMBER OF HOURS
<b>TRAIN</b>	794M	977K	960
<b>DEV</b>	4M	-	5
<b>TEST</b>	5M	-	5

### 3.6.2 MuST-C Dataset

MuST-C is a multilingual speech translation corpus whose size and quality will facilitate the training of end-to-end systems for SLT from English into several target languages. For each target language, MuST-C comprises several hundred hours of audio recordings from English TED Talks, which are automatically aligned at the sentence level with their manual transcriptions and translations.

It consists of four directories, one for each set on which the corpus has been partitioned:

- **train** the set utilized to train the baseline system.
- **dev** the set utilized as validation set during training.
- **tst-COMMON** the set utilized to evaluate the baseline system.
- **tst-HE** an additional test set (not used for baseline training and evaluation)

Each directory is organized in two sub-directories:

- **txt** it contains three textual files: (1) the English sentences, (2) the corresponding aligned target language sentences and (3) the English audio alignment info in yaml format, one line for each sentence.
- **wav** contains audio files, one for each TED talk.

Table 2 MuST-C dataset analysis

	<i><b>TRAIN</b></i>	<i><b>DEV</b></i>	<i><b>TST-COMMON</b></i>	<i><b>TST-HE</b></i>
<i><b>TALKS</b></i>	2412	11	27	12
<i><b>SENTENCES</b></i>	212085	1073	2019	578
<i><b>WORDS EN</b></i>	4520522	24274	41955	13080
<i><b>WORDS AR</b></i>	4000457	21387	36443	10912
<i><b>TIME</b></i>	463h15m44s	2h28m34s	4h04m39s	1h26m51s

### 3.6.3 MGB-3 Dataset

The MGB-3 Arabic data comprises 16 hours of multi-genre data collected from different YouTube channels. The 16 hours have been manually transcribed. The chosen Arabic dialect for this year is Egyptian. Given that dialectal Arabic has no orthographic rules, each program has been transcribed by four different transcribers using these transcription guidelines. The MGB-3

data is split into three groups: adaptation, development, and evaluation data.

Given that dialectal Arabic does not have a clearly defined orthography, different people tend to write the same word in slightly different forms. Therefore, instead of developing strict guidelines to ensure a standardized orthography, variations in spelling are allowed. Thus, multiple transcriptions were produced, allowing transcribers to write the transcripts as they deemed correct. Every file has been segmented and transcribed by four different Egyptian annotators.

Egyptian broadcast data collected from YouTube.

This year, we collected about 80 programs from different YouTube channels. The first 12 minutes from each program have been transcribed and released. This sums up to roughly 16 hours in total divided as follow:

- **Adaptation:** 12 minutes \* 24 programs
- **Development:** 12 minutes \* 24 programs
- **Evaluation:** 12 minutes \* 31 programs

Table 3 MGB-3 dataset analysis

	TIME	NUMBER OF PROGRAMS
<b>TRAIN</b>	4.8 h	24
<b>DEV</b>	4.8 h	24
<b>TEST</b>	6.2 h	24

### 3.6.4 Dataset

To benchmark the performance of the model, a cleaned dataset extracted from the Tashkeela Corpus is utilized. The dataset consists of 55K lines comprising approximately 2.3 million words. It serves as a valuable resource for evaluating different approaches and tools in Arabic diacritization.



# Chapter 4

## 4 Results of our AVD System

### 4.1 Evaluation Metrics

We choose the following evaluation metrics:

- **Perplexity**
- **Word Error Rate (WER)**
- **BLEU (bilingual evaluation understudy)**

#### 4.1.1 Perplexity

In information theory, perplexity is a measurement of how well a probability distribution or probability model predicts a sample. It may be used to compare probability models. A low perplexity indicates the probability distribution is good at predicting the sample.

The perplexity PP of a discrete probability distribution  $p$  is defined in Equation 1 as

$$PP(p) := 2^{H(p)} = 2^{-\sum_x p(x) \log_2 p(x)} = \prod_x p(x)^{-p(x)}$$

Equation 1

where  $H(p)$  is the entropy (in bits) of the distribution and  $x$  ranges over the events. (The base need not be 2: The perplexity is independent of the base, provided that the entropy and the exponentiation use the same base.) This measure is also known in some domains as the (order-1 true) diversity.

A model of an unknown probability distribution  $p$ , may be proposed based on a training sample that was drawn from  $p$ . Given a proposed probability model  $q$ , one may evaluate  $q$  by asking how well it predicts a separate test sample  $x_1, x_2,$

...,  $x_N$  also drawn from  $p$ . The perplexity of the model  $q$  is defined as Equation 2.

The exponent  $-\frac{1}{N} \sum_{i=1}^N \log_b q(x_i)$  may also be interpreted as a **cross-entropy**:

$$H(\tilde{p}, q) = - \sum_x \tilde{p}(x) \log_b q(x)$$

#### 4.1.2 Word Error Rate (WER)

Word error rate (WER) is a common metric of the performance of a speech recognition or machine translation system.

The **WER** is derived from the Levenshtein distance, working at the word level instead of the phoneme level. The WER is a valuable tool for comparing different systems as well as for

evaluating improvements within one system. This kind of measurement, however, provides no details on the nature of translation errors and further work is therefore required to identify the main source(s) of error and to focus any research effort.

This problem is solved by first aligning the recognized word sequence with the reference (spoken) word sequence using dynamic string alignment. Examination of this issue is seen through a theory called the power law that states the correlation between perplexity and word error rate. **Word**

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

*Equation 3*

**error rate can then be computed as:**

Where:

- **S** is the number of substitutions
- **D** is the number of deletions
- **I** is the number of insertions
- **C** is the number of correct words
- **N** is the number of words in the reference (**N=S+D+C**)

#### **4.1.3 BLEU (bilingual evaluation understudy)**

BLEU (bilingual evaluation understudy) is an algorithm for evaluating the quality of text which has been machine-translated from one natural language to another. Quality is the correspondence between a machine's output and that of a human: **"the closer a machine translation is to a professional human translation, the better it is"**. This is the central idea behind BLEU. Invented at IBM in 2001, BLEU was one of the first metrics to claim a high correlation with human judgements of quality and remains one of the most popular automated and inexpensive metrics.

Scores are calculated for individual translated segments—generally sentences— by comparing them with a set of good quality reference translations. Those scores are then averaged over the whole corpus to reach an estimate of the translation's overall quality. Intelligibility or grammatical correctness are not considered.

**BLEU's** output is always a number between **0 and 1**. This value indicates how similar the candidate text is to the reference texts, with values closer to 1 representing more similar texts. Few human translations will attain a score of 1, since this would indicate that the candidate is identical to one of the reference translations. For this reason, it is not necessary to attain a score of 1. Because there are more opportunities to match, adding additional reference translations will increase the BLEU score. Before we go on BLUE algorithm, we should define some used measure and terminology:

##### **4.1.3.1 Precision**

This metric measures the number of words in the Predicted Sentence that also occur in the Target Sentence. Let's say, that we have:

- **Target Sentence:** He eats an apple.
- **Predicted Sentence:** He ate an apple.

We would normally compute the Precision using the formula:

**Precision = Number of correct predicted words / Number of total predicted words**

Precision = 3 / 4

#### 4.1.3.2 Clipped Precision

Let's go through an example to understand how it works. Let's say, that we have the following sentences:

- **Target Sentence 1:** He eats a sweet apple.
- **Target Sentence 2:** He is eating a tasty apple.
- **Predicted Sentence:** He He He eats tasty fruit.

We now do two things differently:

- We compare each word from the predicted sentence with all of the target sentences. If the word matches any target sentence, it is correct.
- We limit the count for each correct word to the maximum number of times that that word occurs in the Target Sentence. This helps to avoid the Repetition problem. This will become clearer below.

Word	Matching Sentence	Matched Predicted Count	Clipped Count
He	Both	3	1
eats	Target 1	1	1
tasty	Target 2	1	1
fruit	None	0	0
<b>Total</b>		<b>5</b>	<b>3</b>

*Figure 4-1 Clipped Precision Example*

For instance, the word "He" occurs only once in each Target Sentence. Therefore, even though "He" occurs thrice in the Predicted Sentence, we 'clip' the count to one, as that is the maximum count in any Target Sentence.

**Clipped Precision = Clipped number of correct predicted words / Number of total predicted words**

Clipped Precision = 3 / 6

## Let's go on how is Bleu Score calculated?

Let's say we have an NLP model that produces a predicted sentence as below. For simplicity, we will take just one Target Sentence, but as in the example above, the procedure for multiple Target Sentences is very similar.

**Target Sentence:** The guard arrived late because it was raining.

**Predicted Sentence:** The guard arrived late because of the rain.

The first step is to compute Precision scores for 1-grams through 4-grams.

We use the Clipped Precision method that we just discussed.

### - Step 1 Precision 1-gram

Precision 1-gram = Number of correct predicted 1-grams / Number of total predicted 1-grams

**Target Sentence:**    The guard arrived late because it was raining  
                                 ↓       ↓       ↓       ↓       ↓  
**Predicted Sentence:** The guard arrived late because of the rain

*Figure 4-2 Precision uni-gram*

So, Precision 1-gram ( $p_1$ ) = 5 / 8

### - Step 2 Precision 2-gram

Precision 2-gram = Number of correct predicted 2-grams / Number of total predicted 2-grams

**Target Sentence:**    The guard arrived late because it was raining  
                                 \_\_\_\_\_  
**Predicted Sentence:** The guard arrived late because of the rain

*Figure 4-3 Precision bi-gram*

So, Precision 2-gram ( $p_2$ ) = 4 / 7

### - Step 3 Precision 3-gram

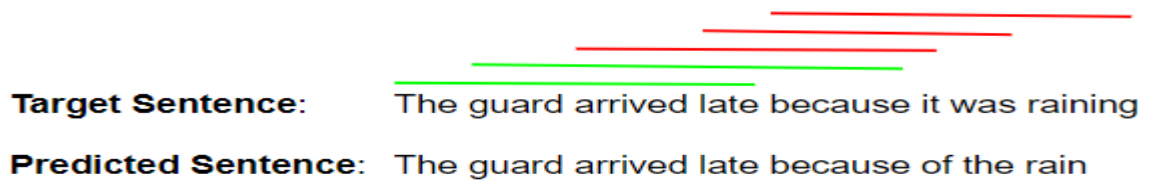
Similarly, Precision 3-gram ( $p_3$ ) = 3 / 6

**Target Sentence:**    The guard arrived late because it was raining  
                                 \_\_\_\_\_  
**Predicted Sentence:** The guard arrived late because of the rain

*Figure 4-4 3 Precision tri-gram*

## - Step 4 Precision 4-gram

And Precision 4-gram ( $p_4$ ) = 2 / 5



**Target Sentence:** The guard arrived late because it was raining

**Predicted Sentence:** The guard arrived late because of the rain

Figure 4-5 Precision 4-gram

## Geometric Average Precision Scores

Next, we combine these Precision Scores using the formula below. This can be computed for different values of  $N$  and using different weight values. Typically, we use  $N = 4$  and

uniform weights  $w_n = N / 4$

$$\begin{aligned}
 \text{Geometric Average Precision } (N) &= \exp\left(\sum_{n=1}^N w_n \log p_n\right) \\
 &= \prod_{n=1}^N p_n^{w_n} \\
 &= (p_1)^{\frac{1}{4}} \cdot (p_2)^{\frac{1}{4}} \cdot (p_3)^{\frac{1}{4}} \cdot (p_4)^{\frac{1}{4}}
 \end{aligned}$$

Equation 4

### 4.1.3.3 Brevity Penalty

The third step is to compute a ‘Brevity Penalty’.

If you notice how Precision is calculated, we could have output a predicted sentence consisting of a single word like “The” or “late”. For this, the 1-gram Precision would have been  $1/1 = 1$ , indicating a perfect score. This is obviously misleading because it encourages the model to output fewer words and get a high score.

To offset this, the Brevity Penalty penalizes sentences that are too short.

$$Brevity\ Penalty = \begin{cases} 1, & \text{if } c > r \\ e^{(1-r/c)}, & \text{if } c \leq r \end{cases}$$

Equation 5

- $c$  is predicted length = number of words in the predicted sentence and
- $r$  is targeting length = number of words in the target sentence

This ensures that the Brevity Penalty cannot be larger than 1, even if the predicted sentence is much longer than the target. And, if you predict very few words, this value will be small.

In this example,  $c = 8$  and  $r = 8$ , which means Brevity Penalty = 1.

#### 4.1.3.4 Bleu Score

Finally, to calculate the Bleu Score, we multiply the Brevity Penalty with the Geometric Average of the Precision Scores.

$$Bleu(N) = Brevity\ Penalty \cdot Geometric\ Average\ Precision\ Scores(N)$$

Equation 6 Blue Score

Bleu Score can be computed for different values of  $N$ . Typically, we use  $N = 4$ .

- BLEU-1 uses the unigram Precision score.
- BLEU-2 uses the geometric average of unigram and bigram precision.
- BLEU-3 uses the geometric average of unigram, bigram, and trigram precision.
- and so on.

Strengths of Bleu Score the reason that Bleu Score is so popular is that it has several strengths:

- It is quick to calculate and easy to understand.
- It corresponds with the way a human would evaluate the same text.
- Importantly, it is language-independent making it straightforward to apply to your NLP models.
- It can be used when you have more than one ground truth sentence.
- It is used very widely, which makes it easier to compare your results with other work.

Weaknesses of Bleu Score Despite its popularity, Bleu Score has been criticized for its weaknesses:

- It does not consider the meaning of words. It is perfectly acceptable to a human to use a different word with the same meaning e.g., Use “watchman” instead of “guard”. But Bleu Score considers that an incorrect word. It looks only for exact word matches. Sometimes a

variant of the same word can be used e.g., “rain” and “raining”, but Bleu Score counts that as an error.

- It ignores the importance of words. With Bleu Score an incorrect word like “to” or “an” that is less relevant to the sentence is penalized just as heavily as a word that contributes significantly to the meaning of the sentence.
- It does not consider the order of words e.g. The sentence “The guard arrived late because of the rain” and “The rain arrived late because of the guard” would get the same (unigram) Bleu Score even though the latter is quite different.

## 4.2 Results on evaluation datasets

In this section we illustrate our results on the evaluation datasets

### 4.2.1 Results Speech recognition

We tested Wave2Vec2.0 and Google speech Recognition on **MuCT-C** Benchmark test set to find out how well each model performs individually. Table 4 shows the results of each model.

Model	WER
Wave2Vec2.0	<b>3.3</b>
GASR	<b>5.24</b>

*Table 4 ASR WER metric on MuST-C*

From table 4, We can see that Wave2Vec2.0 model has achieved the best results.

### 4.2.2 Results of Machine translation

We tested Google machine translation on MuCT-C Benchmark test set in order to find out how well each model performs individually. Table 5.

Model	BLUE
GNMT	<b>0.29</b>

*Table 5 MT BLUE Metric*

# Chapter 5

## 5 System analysis and design

Mudablaj consists of 2 major modules: An android app for users, and the model which described in chapter 3. In this chapter we describe our system analysis and design phases. The aim of this chapter is to allow the reader to understand the overall architecture of the system and how each component in the system is designed.

### 5.1 Assumptions

The following assumptions are made to simplify the overall architecture of the system:

- We use videos as the main input of our system.
- Videos mainly will be in: English.
- Each video is identified by a unique name which is shared between the user's library and the system. YouTube videos is out of scope of this project.
- For outputs it will be either dubbed video or translated transcript of the video.

### 5.2 System requirements

The requirements of our system are very straightforward due to the assumptions stated in section Here is a list of our basic requirements.

- Users must be able to create user accounts.
- Users must be able to create dubbed videos.
- Users must be able to upload videos.
- Users must be able to add videos to their library.
- Users must be able to access and download other users shared videos.
- Users must receive immediate feedback about the video uploading results once the video is uploaded.
- The dubbing process should be fast (less than 5 minutes per video).
- The dubbing process should be fault tolerant: If the system crashes or is disconnected from the database, it should be able to continue dubbing after recovery without affecting the rest of the system.
- The system should be secure against malicious attacks.

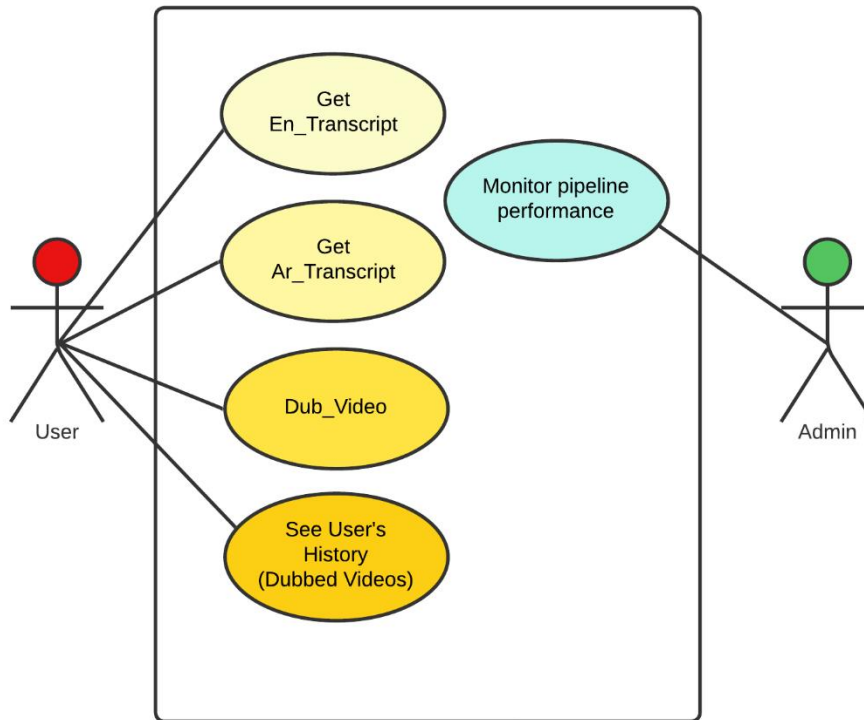
### 5.3 System users and use cases

There are two types of users of our system:

1. Students
2. Normal users

Figure 5.1 shows the use-case diagram of our system.





*Figure 5-1 Use-case diagram*

Here is a brief description about each use case:

- Get En\_Transcript: The user receives the translated English transcript.
- Get Ar\_Transcript: The user receives the translated Arabic transcript.
- Dub\_Video: the user can upload videos to get dubbed.
- See User's History (Dubbed Videos): the user can see his output history as either dubbed video or the translated subtitles of the video.
- Monitor pipeline performance:
  - The admin can access the system to monitor the results of the system.
  - The admin can access the system to ensure that no malicious actions or bugs are in the system.
  - The admin can access the accounts to login and logout of the system.

## 5.4 System architecture

Figure 5-2 shows the overall architecture of the system. The system follows a distributed architecture due to its modular design. The server is distributed across two machines: Application front-end machine and firebase machine.

- Firebase machine: contains most of the back end of the system that serves the android app. It contains the following components:
  - Firebase storage: This is the centralized database of our system. We use a cloud-hosted database for our database.
  - Logical cloud function: This is the back end of the web application for both students and instructors. It's written in PHP language.

- Logical fire store: Serves the android app by offering an HTTP API for android clients. It's written in PHP language too.

The logical servers are tightly coupled to the database so that the interaction between them and the database becomes as fast as possible. Especially for videos submission which is very crucial process and must be both fast and fault tolerant.

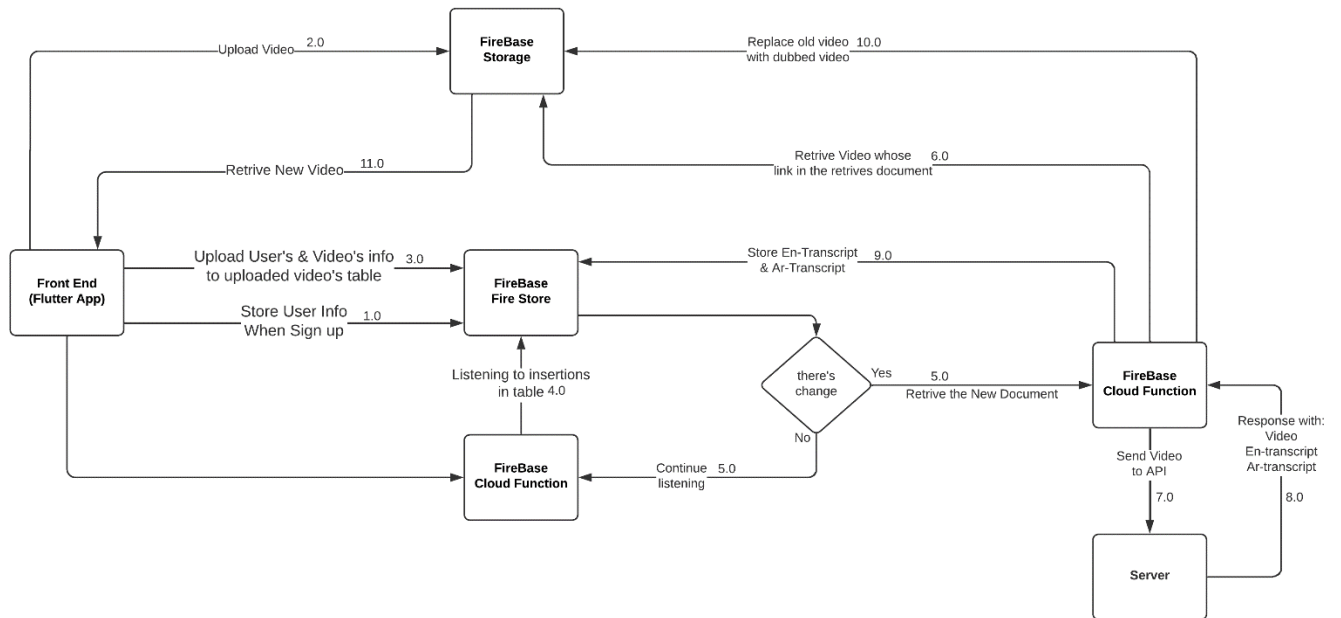


Figure 5-2 Overall architecture of our E-learning system

**Front end flutter app:** It contains the module that runs as a background task in an infinite loop. It looks up the database for any uploaded videos (which are stored by the logical servers). If it finds any, it immediately sends them to the firebase storage and stores the result back in the firebase fire store. This module is loosely coupled from the application server so that the dubbing process becomes independent from the uploading of videos. This is very important because we don't want the dubbing process to slow down the dubbing process.

Another reason for separating the auto grader is that the auto dubbing process is computationally expensive and therefore there is a heavy workload on the auto dubbing machine. A single machine may not be able to handle the workload of both dubbing and serving the applications.

For clients, users will be android clients and the admins will be android clients. We may develop another website for clients in future work.

## 5.5 Data Flow For Dubbing Video

Video dubbing is the backbone of our system. In this section, we focus on the data flow for Dubbing the video and getting English and Arabic transcripts. Figure 5.3 shows this data flow. Here is the description of each process:

1. Sign Up: The user creates an account, and his data is stored in the database Through this account, he can upload and dub videos, and he can keep the dubbed videos on his account.

2. Uploading video: When a user uploads the video, the video is saved to firebase storage then cloud function upload it to the API.

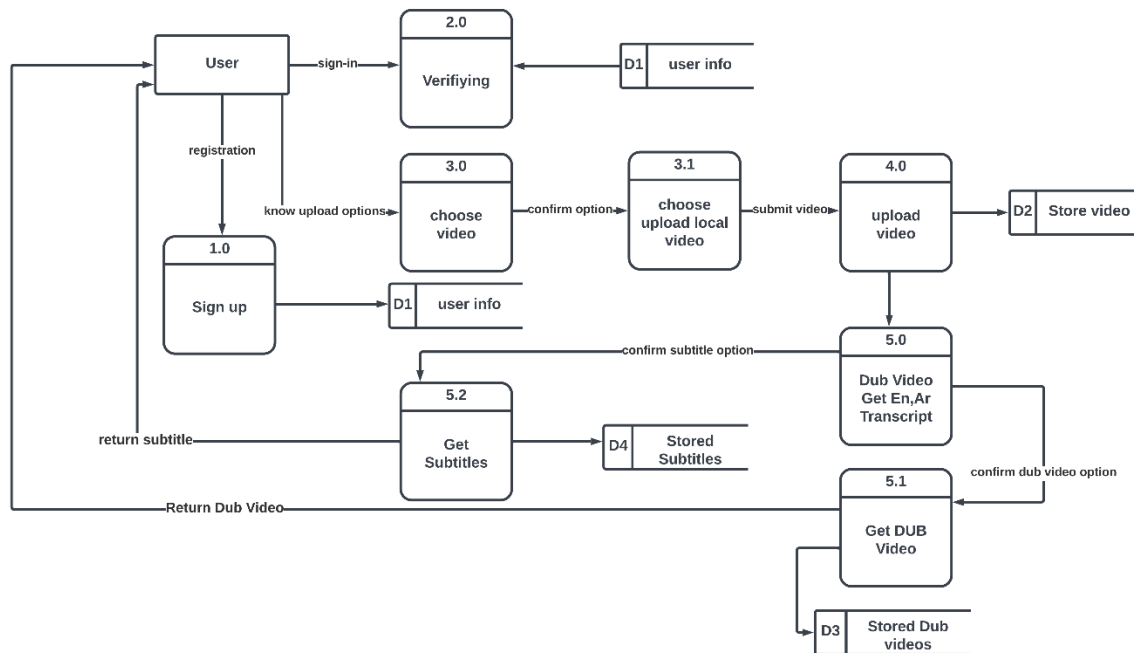


Figure 5-3 Data flow diagram for AVD

3. Dub video: After uploading the video to API, the API response with dubbed video and English transcript, and Arabic transcript to the cloud function.
4. Storing the response: when the cloud function receives response from API, it replaces the old video with the new one, and save English transcript and Arabic transcript to database.
5. Return the results: after storing results in the database flutter application retrieve them and show them to the user.

## 5.6 Database design

Figure 5.4 shows the entity-relationship diagram (ERD) of our database. Here is a description of each table:

### 5.6.1 Users table:

Stores users' information.

Written into when new user registers.

Read from on retrieving user data such as when displaying student profile.

Columns:

- ID column: data type is string and constraints are: primary key.
- Username column: data type is string.
- Email column: data type is string.
- Image column (profile image path): data type is string.

Has a relationship with uploaded videos, users 's YouTube videos tables.

### ***5.6.2 Uploaded videos table:***

Stores uploaded videos information.

Written into when a new video is uploaded.

Columns:

- ID column: data type is string and constraints are: primary key.
- username column: data type is string (foreign key).
- Video path column: data type is string.
- Arabic transcript column: data type is string.
- English transcript column: data type is string.

### ***5.6.3 Users YouTube video table:***

Stores relationship between users and YouTube videos.

Written into when adding new YouTube video by the user.

Used when retrieving all YouTube video that is uploaded by a specific user.

Columns:

- mail column: data type is string (foreign key)
- video link column: data type is string (foreign key).

### ***5.6.4 YouTube video table:***

Stores YouTube video information.

Written into when adding new YouTube video.

Read from when it is needed.

Columns:

- Link: data type is string.
- Arabic transcript column: data type is string.
- English transcript column: data type is string.
- ID column: data type is string and constraints: Primary key.
- Path column: data type is string.

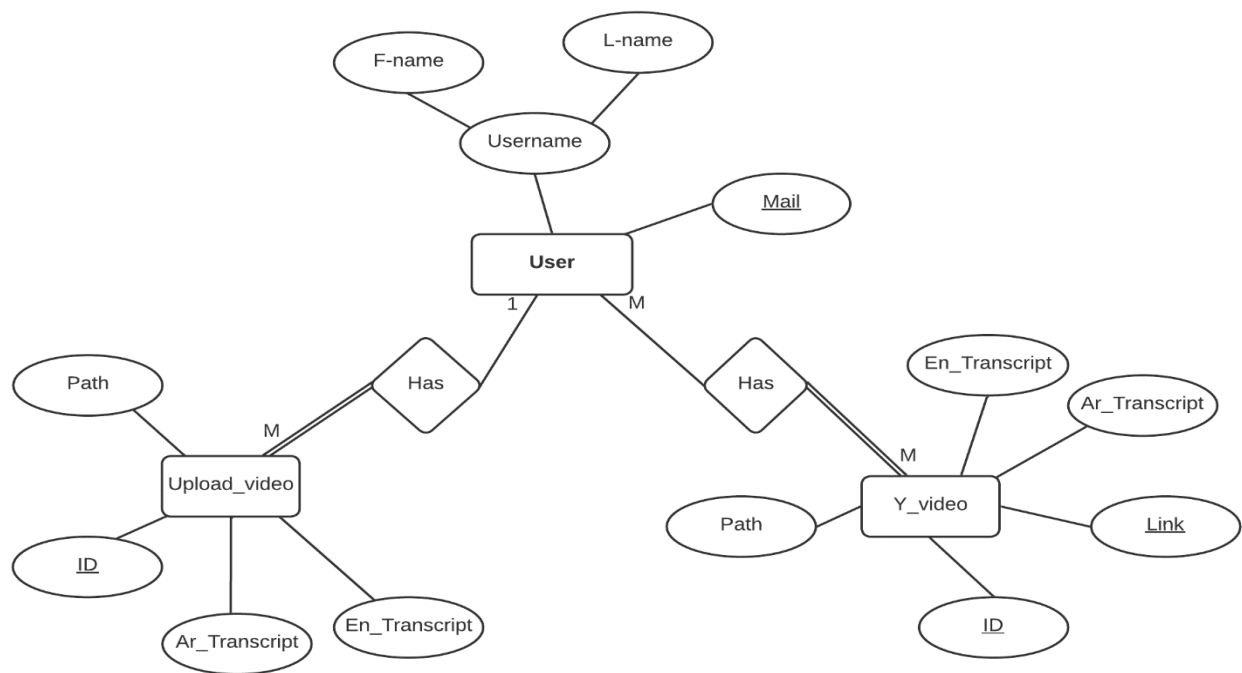


Figure 5-4 Entity-relationship diagram for database

# Chapter 6

## 6 Android application

We've developed an android application in order to provide a lightweight method for students to interact with our system. The main features of the application for users evolve around creating accounts, uploading videos, creating dubbed videos, seeing other users videos, and extracting translated transcript of videos. Here is a brief description about the scenario of using the web application.

### Step 1: Creating Account / Login

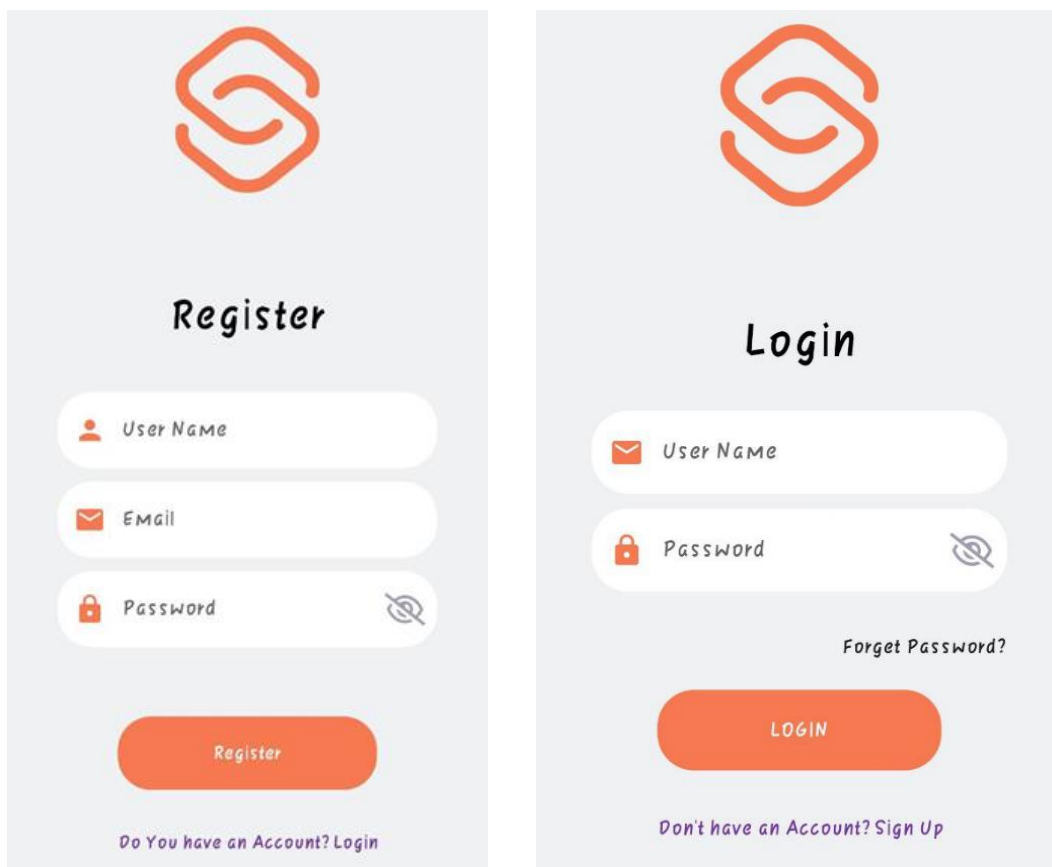
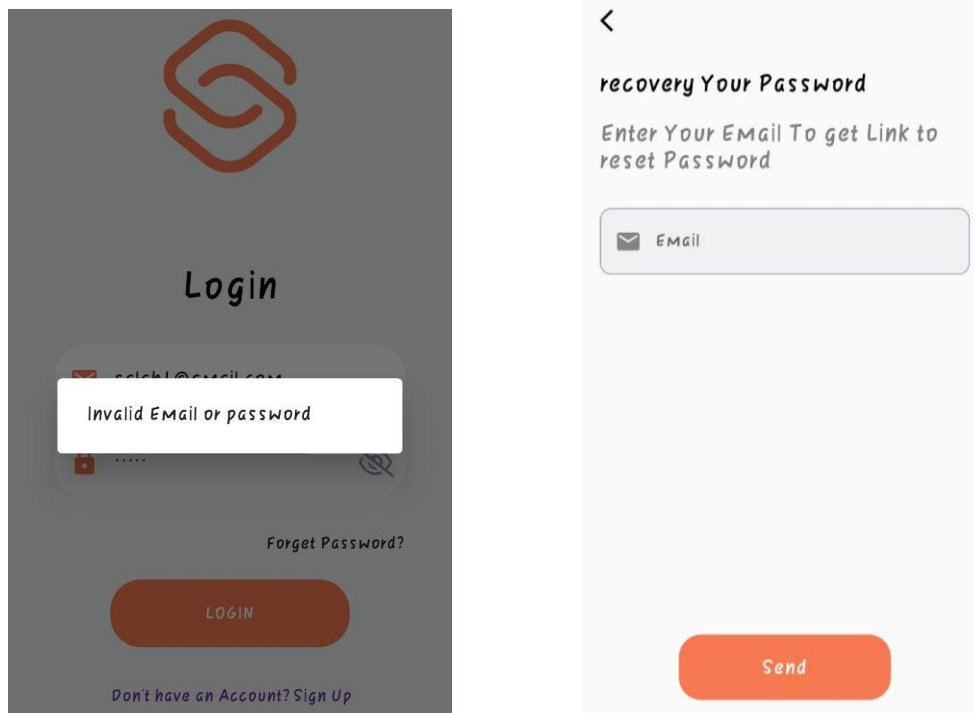


Figure 6-1 Register (b) Login

Each user must have an account in order to use our system. To create an account, the user simply must enter his name, email, and password. Once an account is created, the user will enter the home screen of the app where an (initially empty) home screen of dubbed videos is shown. If the user already has an account, he simply must log in using his email and password in the login screen. Figure 6-1 shows the create-account screen and login screen.

### Entering wrong email or password / recovering password

In case of entering a wrong email or password the app displays an error message to inform the user about his actions. If the user wants to reset his password, the user uses the “forget password?” Button. Once clicked, the user will enter the forget password screen to enter his email address which the app will send an email to, with the password reset link. Figure 7.2 shows the Entering wrong email or password and recovering password.



After the user will enter the forget password screen to enter his email address which the app will send an email to, with the password reset link. Then the user will be able to enter a new password for his account.

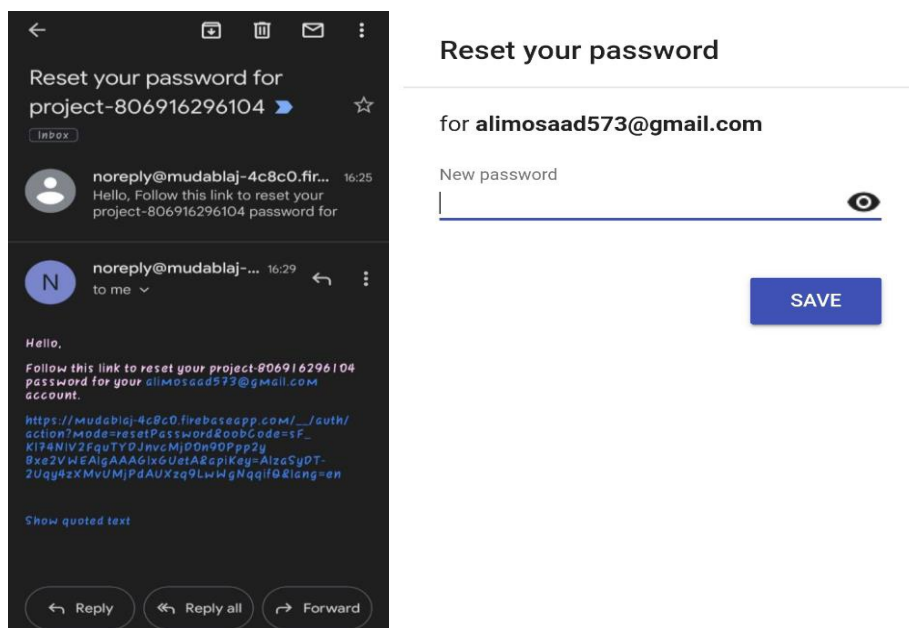


Figure 6-2 Entering wrong data / Entering email / sent email / resetting password

## Step 2: Entering home screen

In the home screen (figure 7.3), the already-shared videos are displayed. If users haven't shared any videos yet, the feed is empty. The user can add a video by pressing the plus button in the middle of the screen and entering the video picker screen to choose which video to upload from his local phone's storage. Once the user chooses the video, the new video will appear in the videos list to add a description for the video content. Figure 7.3 illustrates the home screen

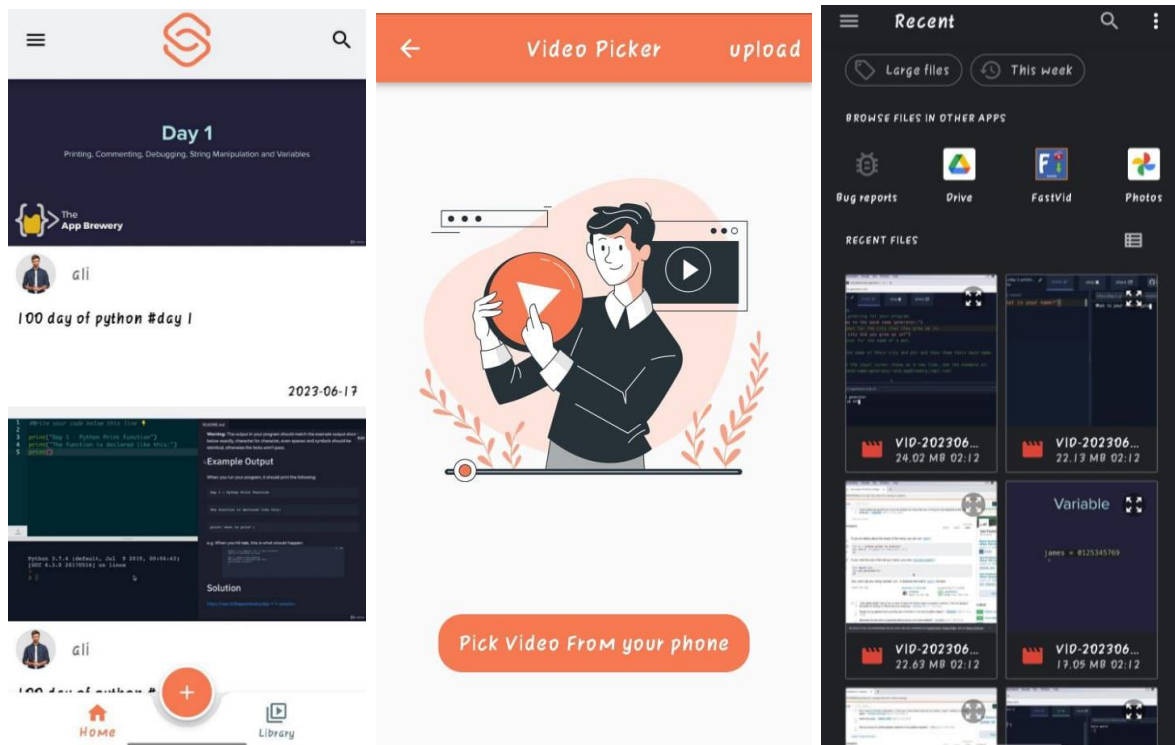


Figure 6-3 Home screen / adding videos

In the home screen (figure 7.4), the user adds video content descriptions to ensure that the content is understood without the need for playing them. After adding the video description, a message will appear confirming that the task was added successfully. Then a screen displaying the uploaded video will appear showing the video thumbnail and the entered description and the date in which the video was added. Figure 7.4 illustrates the process

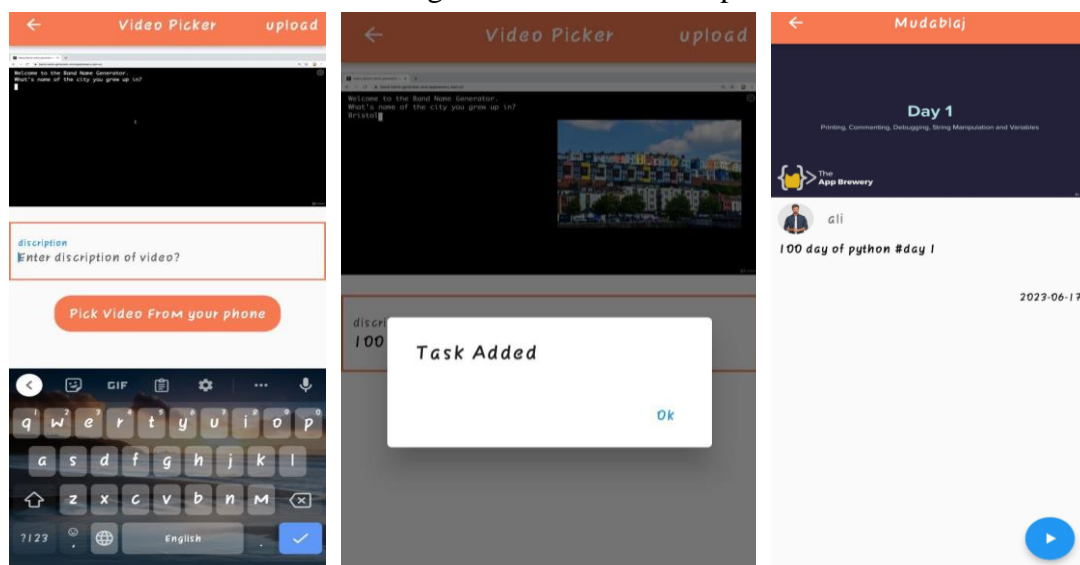


Figure 6-4 Entering video description / task completion message / uploaded video data



### Step 3: Exploring user's library

We refer to the private uploaded videos as user's library. After choosing the video the user will enter the video info screen. In this screen, the user can find the uploaded videos and video description and uploading date at the bottom left of the screen there's a play/pause button for the chosen video. The main content in the course room is to enable the user to view his private videos and play it anytime at ease. Figure 7.5 illustrates the user's library.

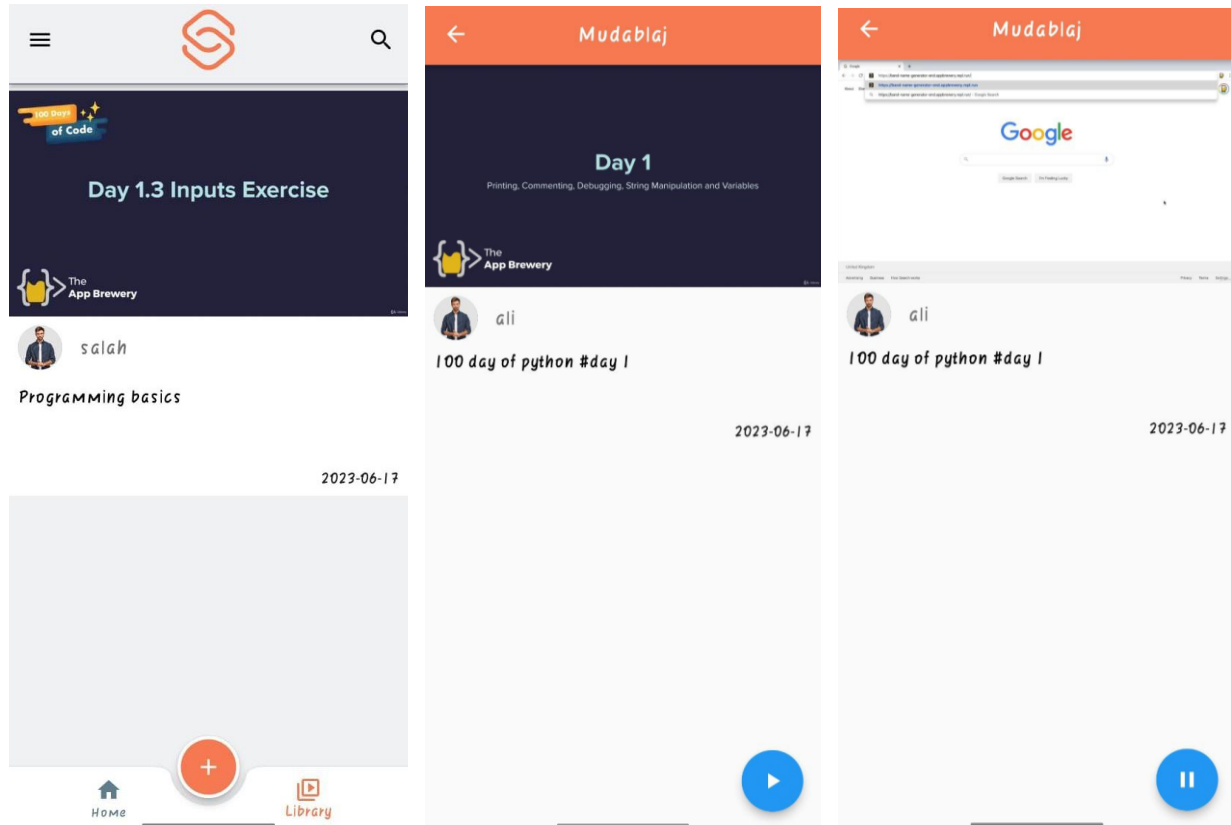


Figure 7.5 User's private library / viewing video info / playing video

### Step 4: exploring the application

In the application side bar menu, there's four options for the user: returning to the home page, entering the user's profile, enter the app settings and logout from the account. Above all the prementioned there's users account photo along with the username of the current active user. When the user choose my profile icon, the user will be redirected into the into the user's info screen showing all it's all his private information starting from the top down, there's user picture, user's e-mail. The user also can add photo. Or edit his account. Also change password. Or reach out for help and support or log out from his account. When user choose to edit his information. He will be redirected to A to the edit profile screen. He can either change his username or e-mail. Then press save to confirm his actions. Figure 7.6 illustrates the app's sidebar.

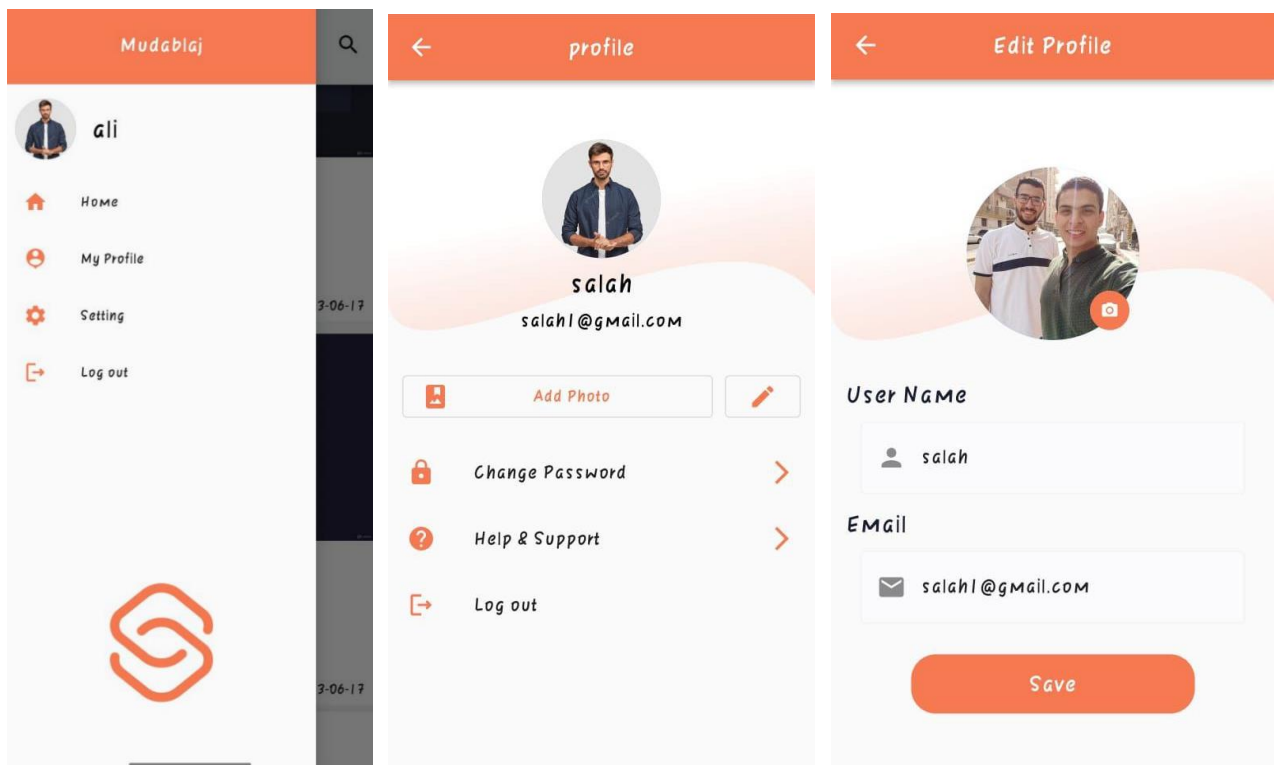


Figure 6-5 Application sidebar / User profile information / edit user profile

When a user clicks on Settings. He will be the redirected into the settings screen. Where he can either change application language to be either Arabic or English and change of theme mode either dark mode or light mode. Figure 7.7 illustrates the app's settings.

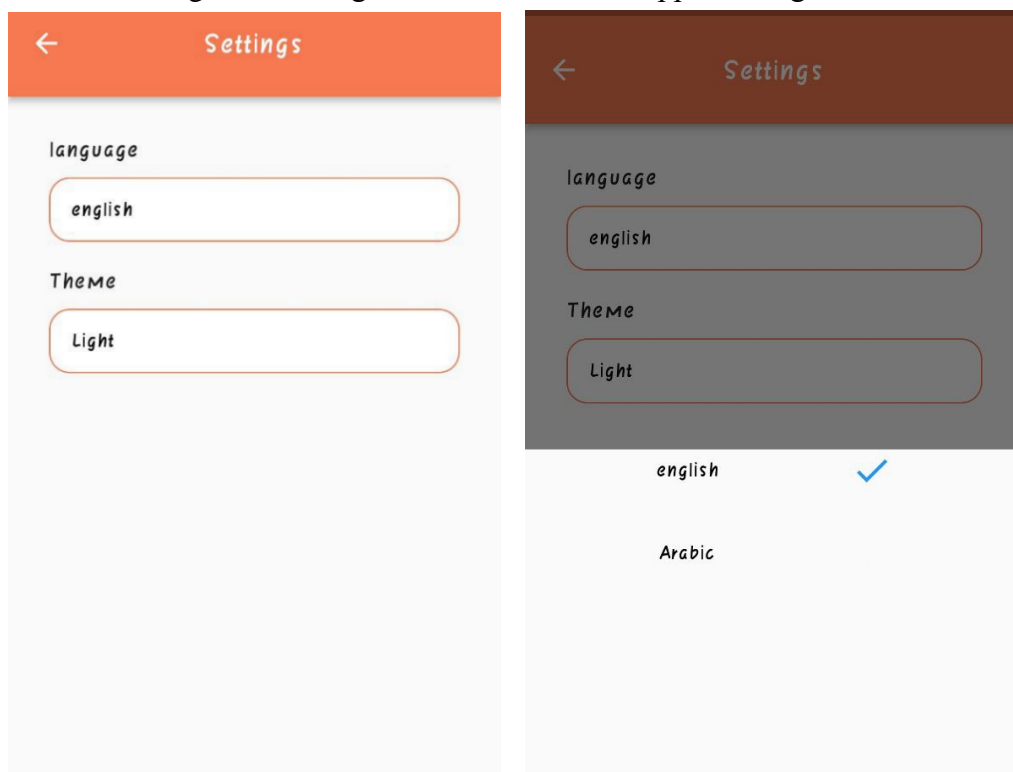


Figure 6-6 App Settings



# Chapter 7

## 7 Conclusion and future work

### 7.1 Conclusion

In summary, our project focuses on the development of a video dubbing system using artificial intelligence techniques to translate English videos into Arabic. We have achieved success in creating a system that delivers high-quality dubbing, bridging the language gap and increasing accessibility to multimedia content. However, there are areas for future improvement. These include expanding language support, refining dubbing accuracy, enhancing voice recognition capabilities, improving audio quality, and incorporating additional features. By addressing these aspects, we aim to advance the capabilities of our system and provide a more immersive and authentic dubbing experience. We are grateful for the support received throughout the project and are excited about the potential impact of our work in the field of AI-based video dubbing.

### 7.2 Future work

**Expansion of Supported Languages:** A prospective direction would be to augment the dubbing system's capabilities to encompass a broader spectrum of languages. This entails incorporating supplementary languages into the linguistic database, thereby facilitating extensive multilingual dubbing functionalities.

**Enhancement of Dubbing Precision:** A prospective objective lies in refining the accuracy of the dubbing process to attain meticulous alignment between the original dialogue and the dubbed text. Such enhancements may entail advancements in grammatical intricacies, pronunciation nuances, and sentence structure intricacies.

**Voice Recognition Advancements:** Future endeavors may concentrate on refining the system's voice recognition capabilities, specifically in discerning and distinguishing diverse voices. This would foster heightened accuracy in the dubbing process and mitigate discrepancies arising from inaccurate voice recognition.

**Augmentation of Audio Quality:** Future enhancements can be directed toward augmenting the audio quality generated during the dubbing process. This may involve developing sophisticated techniques for audio processing, noise reduction, or optimizing frequency balance to deliver an enhanced auditory experience.

**Incorporation of Additional Features:** Future improvements may encompass the integration of supplementary features and functionalities within the dubbing system. For instance, this could involve expanding control over intonation patterns and rhythmical elements in speech, thereby enhancing the naturalness and expressiveness of the output.

## 8 References

- 1- Alexei Baevski, H. Z.-r. (2020 ). wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. *NeurIPS*. META.
- 2- Anmol Gulati, J. Q.-C. (2020). Conformer: Convolution-augmented Transformer for Speech Recognition. *NeurIPS*.
- 3- Ashish Vaswani, N. S. (2017). Attention Is All You Need. *NeurIPS*.
- 4- Chenxu Hu<sup>1</sup>, Q. T. (2021). Neural Dubber: Dubbing for Videos According to Scripts. *NeurIPS*.
- 5- Marcello Federico, R. E.-C. (2020). From Speech-to-Speech Translation to Automatic Dubbing. *Proceedings of the 17th International Conference on Spoken Language Translation* (pp. 257–264). Association for Computational Linguistics.
- 6- Nigel G. Ward, J. E. (2022). *Dialogs Re-enacted Across Languages*. UTEP-CS-22-108.
- 7- Rong Ye, M. W. (2022). Cross-modal Contrastive Learning for Speech Translation. *NAACL*.
- 8- Wei-Ning Hsu, B. B.-H. (2021). HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units. *NeurIPS* (p. 10). META.
- 9- Yifan Peng, S. D. (2022). Branchformer: Parallel MLP-Attention Architectures to Capture Local and Global Context for Speech Recognition and Understanding. *ICML*.
- 10- Yihan Wu, J. G. (2023 ). VideoDubber: Machine Translation with Speech-Aware Length Control for Video Dubbing. *AAAI* .