Development of a Performance Assessment System for Language Learning

Imen Kasrani
Wright State University

Follow this and additional works at: https://corescholar.libraries.wright.edu/etd_all

Part of the Computer Engineering Commons, and the Computer Sciences Commons

Repository Citation
https://corescholar.libraries.wright.edu/etd_all/1893
DEVELOPMENT OF A PERFORMANCE ASSESSMENT SYSTEM FOR LANGUAGE LEARNING

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science

By

IMEN KASRANI
Bach., Tunis University, 2008

2017
Wright State University

Yong Pei, Ph.D.
Thesis Director

Mateen M. Rizki, Ph.D.
Chair, Department of Computer Science

Committee on Final Examination

Yong Pei, Ph.D.

Mateen M. Rizki, Ph.D.

Paul Bender, Ph.D.

Anna F. Lyon, Ed.D.

Barry Milligan, Ph.D.
Interim Dean of the Graduate School
ABSTRACT


Recent advances in computer-assisted, language-speaking, learning/training technology have demonstrated its promising potential to improve the outcome of language learning in early education, special education, English as a Second Language (ESL), and foreign language. The growing number of readily available mobile app-based solutions help encourage interest in learning to speak a foreign language, but their effectiveness is limited due to their lack of objective assessment and performance feedback resembling expert judgment. For example, it has been recognized that, in early education, students learn best with one-on-one instructions. Unfortunately, teachers do not have the time, and it is challenging to extend the learning to the home without the assistance of an independent learning/training tool.

In this thesis research, our objective is to develop an effective and practical solution that will help people to learn and practice a new language independently at low cost. We have explored the use of real-time speech recognition, language translation, text synthesis,
artificial intelligence (AI), and language intelligibility assessment technologies to develop a learning/training system that provides automatic assessment and instantaneous feedback of language-speaking performance in order to achieve an independent-learning workflow. Furthermore, we have designed and implemented a successful prototype system that demonstrates the feasibility and effectiveness of such a computer-assisted independent learning/training solution. This prototype can be easily used on a computer, tablet, smartphone, and other portable devices, and provides a new learning experience that is augmented and enhanced by objective assessment and significant feedback in order to improve the language-speaking proficiency of its user. Additionally, it may be used for real-time translation to support conversation across different languages. Our experimental results demonstrate that the proposed system can sufficiently analyze the intelligibility of one’s speaking, accurately identify mispronounced words, and define a feedback that localizes and highlights errors for continuous practice toward perfection.
TABLE OF CONTENTS

1 INTRODUCTION ........................................................................................................... 1

1.1 Background of the Study .......................................................................................... 1

1.2 Survey of Existing Language Training Applications .............................................. 2

1.3 Why Automatic Performance Assessment for Language Training? ...................... 8

1.4 Contribution of this Thesis ....................................................................................... 9

1.5 Thesis Organization ................................................................................................. 10

2 BACKGROUND AND LITERATURE REVIEW .................................................... 11

2.1 Speech Recognition ................................................................................................. 11

2.1.1 Difference between Speech Recognition and Voice Recognition ................. 12

2.1.2 How Speech Recognition Works? ....................................................................... 13

2.1.3 Uses of Speech Recognition ................................................................................. 14

2.1.4 Limitations of Speech Recognition .................................................................... 14

2.1.5 Existing Speech Recognition Services ............................................................... 15

2.2 Speech Synthesis ...................................................................................................... 19

2.2.1 Uses of Text to Speech Technology ................................................................. 19

2.2.2 How Speech Synthesis Works? ......................................................................... 20

2.2.3 Text to Speech APIs ........................................................................................... 21

2.3 Language Translation .............................................................................................. 22
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.1</td>
<td>IBM Watson Language Translator</td>
<td>22</td>
</tr>
<tr>
<td>2.3.2</td>
<td>Microsoft Translator Text API</td>
<td>23</td>
</tr>
<tr>
<td>2.3.3</td>
<td>Google Cloud Translation API</td>
<td>23</td>
</tr>
<tr>
<td>2.4</td>
<td>Speech Assessment</td>
<td>24</td>
</tr>
<tr>
<td>3</td>
<td>PROTOTYPE DESIGN</td>
<td>26</td>
</tr>
<tr>
<td>3.1</td>
<td>Research Strategy</td>
<td>26</td>
</tr>
<tr>
<td>3.2</td>
<td>System Architecture</td>
<td>26</td>
</tr>
<tr>
<td>3.3</td>
<td>System Design</td>
<td>27</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Speech Recognition</td>
<td>28</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Language Translation</td>
<td>32</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Speech Synthesizer</td>
<td>33</td>
</tr>
<tr>
<td>3.3.4</td>
<td>Recognition</td>
<td>33</td>
</tr>
<tr>
<td>3.3.5</td>
<td>Speech Intelligibility Assessment</td>
<td>33</td>
</tr>
<tr>
<td>4</td>
<td>EXPERIMENTAL RESULTS</td>
<td>40</td>
</tr>
<tr>
<td>4.1</td>
<td>Using the Application for Learning a New Language</td>
<td>42</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Testing the Application with a Single Word</td>
<td>42</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Testing the Application with One Phrase</td>
<td>45</td>
</tr>
<tr>
<td>4.1.3</td>
<td>Testing the Application with One Sentence</td>
<td>48</td>
</tr>
<tr>
<td>4.2</td>
<td>Using the Application for Early Learning</td>
<td>51</td>
</tr>
<tr>
<td>4.3</td>
<td>Using the Application for Real-time Translation to Support Conversation Across Different Languages</td>
<td>55</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS (Continued)

5 CONCLUSIONS AND FUTURE WORK ................................................................. 57

5.1 Conclusion .................................................................................................. 57

5.2 Future Work .............................................................................................. 58

REFERENCES ................................................................................................. 59
LIST OF FIGURES

Figure 1.1: Babbel ........................................................................................................... 3
Figure 1.2: Duolingo ......................................................................................................... 3
Figure 1.3: Busuu ............................................................................................................... 3
Figure 1.4: Memrise ......................................................................................................... 4
Figure 1.5: MindSnacks .................................................................................................. 5
Figure 1.6: HelloTalk ..................................................................................................... 6
Figure 1.7: HiNative ........................................................................................................ 6
Figure 1.8: TripLingo ..................................................................................................... 6
Figure 1.9: Leaf ............................................................................................................... 7
Figure 2.1: Speech to Text Translation .......................................................................... 11
Figure 2.2: Speech Recognition System ........................................................................ 13
Figure 2.3: Siri Speech Recognition System .................................................................. 15
Figure 2.4: Amazon Alexa System ................................................................................ 16
Figure 2.5: Android Speech Recognition API ............................................................... 17
Figure 2.6: IBM Watson Speech Recognition API ....................................................... 18
Figure 2.7: Google Cloud Speech API .......................................................................... 18
Figure 2.8: Text to Speech Conversion ......................................................................... 19
Figure 2.9: Speech Synthesis System ........................................................................... 20
Figure 2.10: IBM Watson Text to Speech API ............................................................. 21
Figure 2.11: MaryTTS Speech Synthesis .................................................................... 22
LIST OF FIGURES (Continued)

Figure 2.12: IBM Watson Language Translator ................................................................. 23
Figure 2.13: Microsoft Translator ......................................................................................... 23
Figure 2.14: Google Cloud Translation .................................................................................. 24
Figure 3.1: Overview of the System ...................................................................................... 28
Figure 3.2: Client-Server Model .......................................................................................... 30
Figure 4.1: Sample of Low-Grade Performance for a Single Word Test ....................... 43
Figure 4.2: Sample of Intermediate-Grade Performance for a Single Word Test .......... 44
Figure 4.3: Sample of High-Grade Performance for a Single Word Test ....................... 45
Figure 4.4: Sample of Low-Grade Performance for a Single Phrase Test ..................... 46
Figure 4.5: Another Sample of Low-Grade Performance for a Single Phrase Test ....... 47
Figure 4.6: Sample of High-Grade Performance for a Single Phrase Test ..................... 48
Figure 4.7: Sample of Low-Grade Performance for Sentence Test ............................... 49
Figure 4.8: Sample of Intermediate-Grade Performance for Sentence Test ............... 50
Figure 4.9: Sample of High-Grade Performance for Sentence Test ............................... 51
Figure 4.10: Sample of Low-Grade Performance Result ............................................... 53
Figure 4.11: Sample of Intermediate-Grade Performance Result ................................. 54
Figure 4.12: Sample of High-Grade Performance Result ............................................... 55
LIST OF TABLES

Table 3.1: Compare Two Strings Using the Levenshtein Algorithm ................................... 35
Table 3.2: Compare 2 Sentences Using the Levenshtein Algorithm ...................................... 37
Table 4.1: Decision Intervals of the Outcome Notes ................................................................. 41
ACKNOWLEDGEMENTS

I cannot express enough thanks to my thesis advisor, Dr. Yong Pei, for his continuous encouragement and motivation, and I appreciate his assistance in the research and writing of this report.

My sincere thanks also go to Dr. Mateen M. Rizki, Dr. Paul Bender, and Dr. Anna F. Lyon, who were involved in the validation committee for this research project. I would also like to thank the Department of Computer Science and Engineering at Wright State for giving me the opportunity to conduct this research.

Special thanks to my colleagues, Ashutosh Shivakumar and Miteshkumar Mahendrabhai Vasoya, for their useful debate and exchanges of knowledge that enriched this experience.

Finally, I would like to acknowledge my profound gratitude to my parents and my husband for the support and continuous encouragement throughout my years of study. Most importantly, I would love to thank my wonderful kids, Cyrine and Omar, for their love.
1 INTRODUCTION

1.1 Background of the Study

Language learning is an essential part of child development in early education and special education, as well as in English as a Second Language (ESL) and foreign language. For example, many people travel to other nations for such purposes as education, business, and tourism, but through the use of social media, even more people worldwide are interacting with each other to share their cultures, ideas, and visions in an open globally connected society. That said, there is still often a language barrier that limits effective communication. As a result, more so now than ever, there is a need to learn new languages.

Learning a new language often means enrolling in language classes that require enormous monetary investment and a long-term commitment. Often, they demand continuous, if not endless, practice, and, many people do not have the time. Others simply prefer to learn a language without the stressful environment created by exams and grades. Thankfully, the latest advances in technologies such as mobile smart devices, speech recognition, natural language processing, and Internet of Things has made the learning of a foreign language not only simple, but also fun. Learning new language is fast and easy with the proliferation of computers and mobile devices, and in the past 5 years, we have observed hundreds of language learning apps for both iOS and Android that seek to meet the needs of new language learners. These apps cover almost all languages at little to no cost, and provide the private, virtual, and all-inclusive environment necessary for learning.
and perfecting a new language through reading, writing and speaking. In the past, in order to learn a new word, one had to search in a dictionary through thousands of words to make a single translation. Today, we can translate any word to any language we want in a matter of seconds. High quality apps continuously arrive to help users learn new languages effectively and efficiently [1].

1.2 Survey of Existing Language Training Applications

There are 5 types of language learning applications [2] [3]:

1. Language courses:

Babbel, Duolingo, Busuu, as shown in Figures 1.1 to 1.3, are the most popular applications of language courses. These applications use translation and dictation to emulate traditional language classes. Learners read text and listen to videos, then interpret and answer questions. These apps are also used to help memorize vocabulary. For speaking training, they use the pronunciation of a native speaker for every word and phrase. This is an unorganized way of learning information because these apps start with complex words and tricky phrases, providing only a way for improving vocabulary rather than effective methods for enhancing conversational skills.
2. FlashCards and SRS:

Memrise, as shown in Figure 1.4, Tinycards, and AnkiApp are examples of FlashCards and Spaced Repetition System applications. They provide a way of practicing vocabulary using memorization of words and phrases, structured as a competitive game in which users are rewarded with points for every correct answer. It is worth noting that Memrise also has a unique feature that associates a new word with similar words from the user’s native language to help make a link between words for better memorization.
3. Educational games:

MindSnacks, as shown in Figure 1.5, is an educational game that helps users learn grammar and vocabulary and practice listening. In addition, this application teaches words and phrases by limiting the time in which to guess the correct answer. It is more applicable to children than to adults because it uses cartoon image.
4. Q&A, chat and social:

The most popular chat and social applications used in learning new languages are HelloTalk, HiNative, and TripLingo, illustrated in Figures 1.6 to 1.8. They use real-time conversation with unknown native speakers and a text-to-voice option to help pronounce received messages. TripLingo is different from HelloTalk and HiNative in that provides the learner with information related to the place that he/she wants to travel. HiNative is a chat application that uses question and answer features so that the learner can ask the native about their language and culture. Hence, it is a place for one to introduce themselves more than it is a place to correctly practice a new language.
5. Contextual reference:

Leaf, as shown in Figure 1.9, is one of the contextual reference applications that explains the necessary words that the learner needs to know when encountering new situations. It is an application used only for learning English.
In summary, current language training applications are limited in the following aspects [4]:

- **Improve writing more than speaking:** Most of the language training applications do not provide an efficient listening or speaking experience. Users can learn some new vocabulary and constructions, but unfortunately cannot carry on a deep conversation with a native speaker of the foreign language. Learning a new language is not only about learning new words and formulating new phrases with appropriate syntax; it is also about being understandable when pronouncing words.

- **Lack of performance assessment and feedback:** Current applications rarely evaluate speaking skills and language pronunciation quality. To make the learning of foreign language more efficient, applications need to deliver meaningful feedback that evaluates the quality of the user’s speech. A successful application for learning new language needs to be able to make a real evaluation of mispronounced speech, and recognize an incorrect accent.
• **It is all about gaming:** Applications that depend more on gaming than the actual fundamentals of a language can be problematic long-term, as passing levels and scoring becomes more important than learning and practicing the language.

### 1.3 Why Automatic Performance Assessment for Language Training?

Whether you want to learn a foreign language for traveling or for improving your communication skills, the principal goal is to become a fluent speaker. People can learn vocabulary and grammar, and then can read words and even sentences after practicing a new language, but often the challenge they are facing is speaking fluently in the language. For learners, either beginners or someone who needs a refresher, feedback on their level of speaking is necessary in order to correct and perfect their speaking. Learners need accurate feedback on their pronunciation training because they are often unable to recognize the precise problem in their pronunciation. In traditional language classes, this is often achieved through practice sessions, such as roleplays, with assessment and feedback from the instructor.

Clearly, instantaneous assessment and feedback for improvement is key to effective independent language-speaking learning/training because they provide accurate evidence of the merits, progresses, and limitations of a learner’s skills. However, the lack of timely accurate, and consistent assessment capabilities that have acceptable operating complexity and affordable cost significantly limit the effectiveness of using mobile apps as a viable option to learn foreign language.

Thus, in this thesis research, we will address this issue by focusing on the design and development of a performance assessment system that can offer the opportunity for language learners to have a more complete picture of what they learn in pronunciation skills.
and what they need to enhance. The feedback needs to be modeled after human judgment and should be able to be easily interpreted by the learner. An automatic scoring system is appropriate, as it gives the learner instant information on overall result quality [5]. Moreover, providing instant feedback gives the learner an idea about their level of progress and gives them a chance to improve their skills over successive attempts.

1.4 Contribution of this Thesis

In this thesis research, we have been developing a low-cost, ultra-portable solution for accurate and automatic assessment of speaking of a second language. It incorporates the latest advances in deep learning-based speech recognition, natural language processing, and mobile and cloud computing technologies to enable automatic and instantaneous speech assessment and analysis. It can be used conveniently at home, in the classroom, via distance learning, and potentially over social media platforms for language learners to practice and improve their skills. Furthermore, it provides recommendations that directly address each individual’s particular needs to better motivate them to practice by themselves, resulting in faster learning outcomes.

The main objective of this thesis is to be able to support learning to speak a new language by providing automatic evaluation and feedback of pronunciation skills. To meet this objective, this thesis proposes the following key capabilities and features:

1. **Recognize speech correctly**: The most important condition necessary for a valid application to produce excellent results is that it must have accurate speech recognition. The speech needs to be clear, and the noise encountered when capturing the voice needs to be managed so that good levels of recognition accuracy can be guaranteed.
2. **Translate the speech**: The application must be able to translate the recognized text to the selected second language.

3. **Synthesize the speech**: The application must be able to convert the translated text into the speech of the desired language.

4. **Performance assessment**: The application must be able to evaluate the user’s speech performance in a way that is comparable to human observations. This phase should give instantaneous feedback to the user, present the overall score as a graded bar, and add specific feedback on incorrectly pronounced words.

1.5 **Thesis Organization**

The rest of this thesis is organized as follows: in Chapter 2, we provide an effective and critical evaluation of previous research and existing techniques and methods related to our work. In Chapter 3, we explain our research methodologies, design, implementation, and the data collection process. In Chapter 4, we present the experimental results and performance evaluations and analysis for all methods proposed. Finally, in Chapter 5, we provide our conclusions and offer future recommendations to continue this work.
2 BACKGROUND AND LITERATURE REVIEW

In this chapter, we will review the related technologies that help enable accurate and automatic assessment of speaking of a language, including:

- Speech recognition
- Speech synthesis
- Language translation
- Speech intelligibility assessment

2.1 Speech Recognition

![Figure 2.1: Speech to Text Translation](image)
Speech recognition is a way of transforming a spoken language captured from a microphone or telephone to text, as illustrated in Figure 2.1, where the computer is able to identify the words of the spoken language and convert them to text. Today, speech recognition occupies our life. For example, many companies use an instructed automated voice recording system that allows users to press a button and have a question answered using speech recognition technology and vocal answering. Speech recognition software is being used more every day, and can now be installed not only on computers, but also on mobile devices, for example, we can give a verbal command for phone calls on smartphones.

Speech recognition performance is measured based on two things: accuracy and speed [6]. The accuracy is the error rate of converting spoken language to text, and the speed measured for the real-time factor.

2.1.1 Difference between Speech Recognition and Voice Recognition

Speech recognition recognizes the words or phrases of the speaker; thus, it is language dependent. It analyzes and understands the spoken speech. Conversely, voice recognition recognizes the voice of the speaker itself [7]. Moreover, it is used to detect the person who is speaking, independent of the language and ignoring the meaning of speech in order to identify the physical person speaking. Hence, it analyzes the voice of the speaker. Generally, it is used for security, similar to fingerprinting.
2.1.2 How Speech Recognition Works?

![Speech Recognition System Diagram]

Figure 2.2: Speech Recognition System

A speech recognizer tries to transform a digitized acoustic signal to language text. As illustrated in Figure 2.2, it breaks the signal into phonemes, identifies the phonemes, and then transforms the phonemes to words through the following steps:

**Step 1**: Audio input: Voice is captured using microphones.

**Step 2**: Feature extraction or digital signal processing: The analog wave is converted to digital data using an analog-to-digital converter (ADC) and the digitized sound is filtered to remove excess noise. Then, the sound waves are turned into numbers. It uses sampling that recording the height of the wave at equally spaced points at frequent intervals with a sampling rate of 16 kHz (16,000 samples per second) [6] [8].
Step 3: Phonetics breakdowns: The signal is broken into phonemes, using Fourier to transform and break complex sound waves into simple sound waves.

Step 4: Acoustic model: Identify phonemes: Matching the segments to known phonemes.

Step 5: Language model: Transforms the phonemes into words, running the contextual phoneme plot and comparing them to a large library of known words using the statistical modeling system to determine the most likely outcomes. It may combine the phonemes as needed.

2.1.3 Uses of Speech Recognition

For a short list of samples:

✓ Dictation: for people, who have trouble writing.

✓ Commercial/industrial application: military, medical, education, or first responders.

✓ People with disabilities: people who cannot use their hands or have impaired vision or hearing.

✓ Automated phone system: most companies now use an automated voice answering system and some use a button to answer questions.

✓ Gaming.

2.1.4 Limitations of Speech Recognition

- Require high signal-to-noise-ratio (i.e., noise may impair the accuracy): user needs to stay in a quiet place and place the microphone close his/her mouth.

- Overlapping speech: if too many people talk at the same time.

- Intensive use of computer power.
• Homonyms: two words that sound and are spelled the same but have different meanings.

• Inaccuracy due to different accents.

2.1.5 Existing Speech Recognition Services

There are several mature speech recognition services being offered through cloud services, e.g., iOS Siri, Amazon Alexa, Android speech to text, IBM Watson, and Google Cloud Speech.

Siri Voice API:

Apple’s Siri, as illustrated in Figure 2.4, is integrated into many Apple iOS and Mac devices. It uses machine learning technology. Siri supports two features: first, as a dictation system, it transcribes the voice to text. Second, as an AI-assistant, it can take a voice-command to enable convenient interaction with the device. For example, we can ask Siri about the weather, for reminders or scheduling, or to call somebody, but it may not always be accurate due to speech recognition limitation [9].

Figure 2.3: Siri Speech Recognition System
Amazon Alexa:

Alexa Voice Service API (AVS) [10], as shown in Figure 2.4, is a cloud speech recognition service from Amazon. It can respond to general commands like playing music, providing a weather forecast, or adjusting lighting. It supports some region like UK, Canada, Germany, Austria, India, and Japan, can be used in any device connected to internet, and has speaker and microphone. It also claims to recognize speech in high-noise backgrounds [8].

Figure 2.4: Amazon Alexa System

Android Speech to Text API:

Android speech to text API, illustrated in Figure 2.5, is a free technology [11]. It can convert voice to text without needing an internet connection for recognition, recognizing speech in both online and offline mode. Offline mode requires the installation of a language package, and online mode sends the input voice to the server, converts the voice speech to text, and then sends it back to the application. Android speech to text API has the capability to recognize speech offline, whereas Google Cloud Speech API and
Watson Speech API needs internet connection to work. The disadvantage of using this API is that it can function only on Android devices.

![Android Speech Recognition API](image)

**Figure 2.5: Android Speech Recognition API**

**IBM Watson Speech to Text:**

IBM Watson Speech to Text, illustrated in Figure 2.6, recognizes speech to text in real time with good accuracy [12]. In order to have a high accuracy, it uses AI to combine language information with audio composition, accepting many types of audio formats with file sizes less than 100 MB. In addition, it returns partial recognition results when it becomes available. Compared to others API’s, it supports a significantly lower number of languages; to date, it supports only eight different languages and the service needs to be paid for [13].
**Google Cloud Speech API:**

Google Cloud Speech API, as illustrated in Figure 2.7 translates text to speech using effective neural network modeling and machine-learning algorithm. It recognizes over 110 languages [14]. It can also provide interim results while the user is either speaking in real time speaking or using a recorded audio file less than 60 seconds in length. Furthermore, it can give word hints depending on the context provided as well as filter incorrect text results for certain languages. The other features of this API are that it supports many audio encodings and works on any device and with any platform that supports REST or gRPC. The API works reasonably well and with high accuracy, even in a noisy environment.

**Figure 2.7: Google Cloud Speech API**
2.2 Speech Synthesis

Figure 2.8: Text to Speech Conversion

Speech synthesis (shown in Figure 2.8), also called text-to-speech (TTS), translates natural language text into speech [15]. It has the ability to convert text to voice, the computer, smartphone, tablet, or another device reading the produced audio stream aloud. Speech synthesis is the opposite of speech recognition, as it is a speech to text converter rather than a text to speech converter.

The quality of a speech synthesizer is measured in two perspectives: naturalness and intelligibility. Naturalness is the ability to resemble the human voice, and intelligibility is the clearness of the output voice.

2.2.1 Uses of Text to Speech Technology

Text to speech has multiple benefits for many users:

- Usability for people with learning and physical disabilities
- Entertainment productions such as games and animations
- Learning a new language
- Improving speech skills
2.2.2 How Speech Synthesis Works?

![Speech Synthesis System Diagram]

**Figure 2.9: Speech Synthesis System**

A text to speech mechanism analyzes written text, converts the analyzed text to a phonemic representation, then transforms the phonemic to waveforms that are output as sound [16] [17], as illustrated in Figure 2.9.

**Stage 1:** text to word: It is the phase of text analysis. Numbers, abbreviations, acronyms, dates, and time need to be converted to text. Also, it figures out the tense of the verbs, the type of phrase, and the beginnings and the endings of sentences. We call this phase pre-processing or normalization.

**Stage 2:** words to phonemes: It is the phase of linguistic analysis. After analyzing the words, we will pass to the phase of processing where we convert the words to phonemes.

**Stage 3:** phonemes to sound: After we have converted the text to phonemes representation, we need to convert the phonemes to sound. The synthesizer takes the symbolic linguistic generated from the front-end and generates the sound. To convert phonemes to speech, there are three different methods. The first one is to record all the phonemes with a human voice. In the second method, the machine produces the phonemes itself by creating straightforward sound frequencies. The third method imitates the mechanism of the human voice.
2.2.3 Text to Speech APIs

There are several technologies that make translation from text to speech, such as IBM Watson Text to Speech, FreeTTS, and MaryTTS.

**IBM Watson Text to Speech API:**

The IBM Watson Text to Speech service illustrated in Figure 2.10 provides an API to synthesize written text into natural-sounding speech in different languages [18], particularly English, French, German, Italian, Japanese, Spanish, and Brazilian Portuguese. In addition, it provides 13 voices: a male and/or female voice for each language. Another feature of this API is that the synthesized audio returns in mp3 format. It also guarantees a small delay of the result. One limitation is that it is only free for 30 days.

![Figure 2.10: IBM Watson Text to Speech API](image)

**FreeTTS:**

Free TTS is an open source software [19]. It is a system that synthesizes a text to speech, written totally in the Java programming language and is an implementation of Sun's Java Speech API. It is based on Flite. In 2002, Benchmarks indicated that FreeTTS
ran two to three times faster than Flite. The limitation of this system is that it can only convert a text to the English language.

**MaryTTS:**

MaryTTS (Modular Architecture for Research on Speech Synthesis), illustrated in Figure 2.11, is another open source application, written entirely in Java programming language [20]. It was a shared project between DFKI’s Language Technology Lab and the Institute of Phonetics at Saarland University. It supports many languages, including German, British and American English, French, Italian, Luxembourgish, Russian, Swedish, Telugu, and Turkish. It can also run on multiple platforms.

![MaryTTS](image)

**Figure 2.11: MaryTTS Speech Synthesis**

### 2.3 Language Translation

Language translation service translate text input from one language to another language. We will discuss some of the most popular cloud-based translation services like IBM Watson language translator, Microsoft Translator, and Google Cloud Translation.

#### 2.3.1 IBM Watson Language Translator

The IBM Watson Language Translator service, illustrated in Figure 2.12, can translate a text from one language to another language [21]. It is different from other translation services only by the way in which it protects your data; however, it supports only 10 languages includes English, Arabic, French, German, Italian, Japanese, Portuguese, Korean, and Spanish.
2.3.2 Microsoft Translator Text API

Microsoft Translator Text API, shown in Figure 2.13, is a cloud-based machine translation [22]. It supports 60 languages and can detect them automatically. It can be used in any application, supports many languages, and can be integrated into any platform and used with any operating system. However, it has limited free usage.

2.3.3 Google Cloud Translation API

The Google Cloud Translation API, as illustrated in Figure 2.14, can translate text between 2 different languages [23] using the state-of-the-art Neural Machine Translation. It uses language detection when the source language is not given, and supports more than
100 languages, even making translations between groupings of thousands of languages pairs.

Figure 2.14: Google Cloud Translation

2.4 Speech Assessment

Speech language assessment is a complex process that requires an accurate comparison between a recognized spoken text and the translated or given text. For instance, we need to compare the two texts to find the number of incorrect words that the learner spoke. Then, based on the result of the comparison, the learner will be given a feedback of his/her weakness and strengths in speaking the language.

To compare and identify the similarity/dissimilarity between two texts, we need to measure the distance between them. There exist different algorithms that count the number of operations needed to transform one string to another string. In the following, we explain the most popular algorithms that are used for this purpose [24].

**Levenshtein Distance Algorithm:** Compare the characters of two strings. Return the sum of the costs of the minimum number of substitutions, deletions, and insertions required to transform one string to the other. The length of the input strings does not have to be equal. With complexity of $O(nm)$, one of its disadvantages is that although this algorithm can be applied to two long strings, the real-time performance of the algorithm will depend on the length of the strings.
**Hamming Distance:** Return only the number of substitutions observed, and only applied for strings of equal length.

**Longest Common Substring Distance:** Remove the minimum numbers of characters from both strings to have identical substrings.

**Jaro-Winkler Distance:** Measure the edit distance between two strings. It allows only the transposition between two adjacent characters to transform one word into the other.
3 PROTOTYPE DESIGN

In this chapter, we will provide the research methodologies of this thesis. We will describe the strategies, the methods, and the approach used in this research. We will describe in more details the selection of particular technologies in our prototype design.

3.1 Research Strategy

In this research, we intend to augment and enhance the existing computer-assisted language training idea, as evident by many existing language training applications, to enable independent learning workflow. Particularly, the new research takes the existing research and tries to build a critical new capability by providing automatic performance assessment and feedback for language learning/training.

3.2 System Architecture

To support a low-cost mobile device-centered solution, we decided to adopt a mobile-cloud hybrid solution that takes advantage of the processing and storage capabilities, the capacities of cloud computing service, and the portability and the availability of mobile devices.

Cloud-based applications can access databases, servers, and storage over the Internet. Cloud computing allows the fast process of data over the Internet in real-time from wherever without concerns of storage or computing power [25]. It is a virtual on-demand delivery of service. It doesn’t require an investment in local hardware and
Moreover, it doesn’t require the user to install, configure, and manage the hardware and the software. It provides a low-cost and a low-latency access to the required resources at any time and from anywhere. The most popular providers of cloud computing services are: Google Cloud, Amazon Web Services, Microsoft Azure, IBM Bluemix, and Aliyun. The principal benefits of cloud computing are:

**Flexible capacity:** Computing needs can be increased or decreased depending on the user traffic. With cloud computing, we don’t have to guess the ever-changing capacity that we will need as new customers subscribe to our services.

**Cheap cost of service:** Cloud users pay per use model, and the service is metered. The users pay only for what they use. It cuts the cost, particularly the initial required investment.

**Disaster recovery:** The cloud maintains multiple copies of data, which is less expensive than manually storing redundant data in multiple data centers.

On the other hand, there are disadvantages of cloud computing:

**Security:** Essential information can be deleted or accessed by hackers. However, cloud service providers may use encryption and other security measures to protect against such risk. Often, cloud service providers are more likely able to keep the security measures up-to-date compared to individuals.

### 3.3 System Design

In this section, we will discuss the main design choices made for our prototype system, as illustrated in Figure 3.1.
3.3.1 Speech Recognition

In this research, we used the Google Cloud Speech service with streaming speech recognition for real-time audio feeding from a microphone.
3.3.1.1 Google Cloud Speech API

Google Cloud Speech API supports 3 types of speech recognition.

**Synchronous speech recognition:** It recognizes a local or remote audio file of less than one minute in length. Cloud Speech API returns the text as soon as the recognition is done. Need to send the audio file content directly to Cloud Speech API or recognize an audio file that already exists in Google Cloud Storage. The advantage of this method is that it is faster and simpler than the other two services.

**Asynchronous speech recognition:** It recognizes an audio file of approximately 180 minutes of length and stored in Google Cloud Storage.

**Streaming speech recognition:** It streams an audio voice to Cloud Speech API and receives the result at the same time the audio is processing. It is available only through gRPC. It streams a real-time audio to Google Cloud Speech. Two types of streaming speech recognition are available: either via a local audio file or via audio stream fed from a microphone. The audio content length should not exceed one minute for every request.

In this application, we used the Java programming language. We tried to develop an application that recognizes a real-time streamed audio feed from a microphone: We used Java Sound API and Speech Cloud API to reach our goal.
3.3.1.2 Client-Server Model

![Client-Server Model Diagram]

**Figure 3.2: Client-Server Model**

A client-server model is a distributed communication architecture. The server provides a service and the client request for service, as depicted in Figure 3.2. A single server can serve multiple clients at the same time. The client requests a connection from the server that can be accepted or rejected. The communication between the client and the server is established via the Internet or other networks. Many protocols use this model, such as HTTP, SMTP, and DNS. Many computer applications use this model, for example, email, and the Web.

3.3.1.3 Google Remote Procedure Call (gRPC)

gRPC is an open source remote procedure call developed by Google [26]. It uses the client server model and can run in any environment. gRPC defines a service method and specifies the methods that we can call remotely. It supports many languages, such as Java, C++, C#, Objective-C, Node.js, Go, Python, PHP, and Ruby. We can implement a method from a server written in Java that receives a call from a client written in PHP. It
uses protocol buffers that allow serialization. The server side implements the methods and handles client calls, and the client-side calls procedures that are located in a remote server machine in the same way it would a local object, but across the network. Furthermore, the client uses the stub, which then sends the request to the remote server to execute a specific procedure with packing parameters. gRPC functions as synchronous or asynchronous: Synchronous RPC blocks calls until it gets a response from the server. If the network is asynchronous, it is critical that the RPC calls must also be asynchronous because we cannot, in many cases, block calls while waiting for a response. The advantage of using this protocol, which is based on HTTP/2 transport, is that it allows the building of real-time applications.

gRPC has four types of service models:

**Unary RPCs:** The client sends a single request and receives a single response from the server.

**Server streaming RPCs:** The server sends a stream of response after a single request from the client instead of a single response.

**Client streaming RPCs:** The client sends a stream of requests instead of a single request and receives a single response from the server.

**Bidirectional streaming RPCs:** The call is initiated by the client; it calls the method and sends the metadata. The server can wait until the client sends all the messages and then respond, or the server and the client make one-to-one relation as a deduction, and the client sends a message to which the server responds, and so on.
3.3.1.4 Improvement to Google Cloud Speech API

Google Cloud Speech API provides the code for a real-time audio stream from microphone detection for only four programming languages: C#, NODE.JS, PYTHON, and GO. Because we chose Java as the programming language, we tried to develop a client-side version in Java. We have succeeded in developing a Java-version of the real-time audio streaming that captures audio from a microphone using a target data line provided by the Java Sound API. Furthermore, we used a buffer to read from the microphones that have a limited size. Also, we used multithreading to reach our goal.

3.3.1.5 Multithreading

Multithreading is a process of executing multiple threads within the same program in parallel. Threads are lightweight processes, and several threads can exist in a process to perform many operations simultaneously. Threads are independent and therefore don’t affect each other, but they share resources such as the same memory area. They read and write from the same memory at the same time [27].

To convert from speech to text using the Google Cloud Speech Recognition service, we used multithreaded programming. We created two threads: the first thread sends in real time the audio captured from the microphone to the recognizer. The second thread receives the text result from the recognizer nearly simultaneously (depending on load and network).

3.3.2 Language Translation

We used cloud-based translation service to translate text from one source language to the target language.
3.3.2.1 Google Cloud Translation API

Google Cloud Translation API translates text from one source language to the target language. We used to translate the spoken language in the same way as real-life translation. We used Google Cloud Translation for testing. The translation step didn’t detect the language, so we need to specify the source and the target languages, and then we only need to specify the text that we want to translate.

3.3.3 Speech Synthesizer

We used Mary TTS to translate the text to speech because it is an open source platform written in Java synthesize many natural languages. It is a client server system. Mary TTS translates text to speech in 4 steps:

1. Init mary: Creates the client.
2. Synthesis.
3. Writes output in wav file.
4. Reads from the file using SourceDataLine from Java Sound API.

3.3.4 Recognition

We used speech recognition here also for multiple languages: French, English, or Italian.

3.3.5 Speech Intelligibility Assessment

We compare two sequences word by word: The first sequence is the output of the translation to the target language, which serves as the control; the second sequence is from the recognition of speech when the trainee speaks in the target language. To provide feedback, we need to find both wrong words in the second and missing words in the first, and the words that should be used to replace them to give the correct sentence.
3.3.5.1 Levenshtein Distance Algorithm

The Levenshtein algorithm is a minimum edit distance algorithm. It measures the similarity between two strings. Also, it calculates the minimum numbers of change, including deletion, insertion, and substitutions, required to transform one string to another string. The time complexity of the algorithm is $O(n \times m)$, where $n$ and $m$ are the lengths of the first string and the second string, respectively. Moreover, the space complexity is $O(n \times m)$ because it memorizes in matrix.

The Levenshtein distance algorithm has been used in many fields:

- Spelling correction
- Speech recognition
- Information extraction
- DNA analysis
- Machine translation
- Plagiarism detection

We can use this algorithm for iterative or recursive implementation:

1. The recursive implementation: This implementation is inefficient because it calculates the Levenshtein distance of the strings many times.

2. The iterative implementation: We can create a full matrix for both strings, or we can create two matrix rows for each string. In Table 3.1, we provide an example that compares two strings “EXECUTION” and “INTENTION.”
Table 3.1: Compare Two Strings Using the Levenshtein Algorithm

<table>
<thead>
<tr>
<th></th>
<th>E</th>
<th>X</th>
<th>E</th>
<th>C</th>
<th>U</th>
<th>T</th>
<th>I</th>
<th>O</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>I</td>
<td>1</td>
<td></td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>N</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>T</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>E</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>N</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>T</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>8</td>
</tr>
<tr>
<td>I</td>
<td>7</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>O</td>
<td>8</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>N</td>
<td>9</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>7</td>
<td>6</td>
<td>5</td>
</tr>
</tbody>
</table>

We used Backtrace to compare the two strings and found the type of errors (substitution, deletion, and insertion) to have a similar word. This example has 5 errors.

The first analysis of the solution is:

E X – ECMUTION

I N T E N – TION

1 deletion T

1 insertion U

3 substitutions: E → I, X → N, C → N

The second analysis of the solution is:

- EX E CU TION
3.3.5.2 How we Used this Algorithm?

We compared two sentences word by word using this algorithm by creating two arrays of strings. It returned the percentage of similar words, and we used the iterative implementation. Additionally, we created a matrix to hold the computed values. We used this function to compute the values in the matrix.

$$\min\{D(I - 1, J) + 1, D(I, J - 1) + 1, D(I - 1, J - 1) + \text{match}\}$$

$$\text{match} = 0 \text{ if } s0(I) = s1(J) \text{ or } \text{match} = 1 \text{ if } s0(I) \neq s1(J)$$

Time complexity: O (n*m)

Space complexity: O(n*m)

Backtrace complexity: O(n + m)

The Levenshtein algorithm is very useful because it compares two strings with respect to the order and returns the minimum cost to have similar sentences. However, one of the disadvantages of this approach is that if we want to use this algorithm for a long array of strings, the process will be very slow because of high complexity.

Algorithm steps:

- Create two arrays of strings.
- Create the matrix of the values.
- Found the number of mismatch from $D[n, m]$, where $n, m$ are the length of the two arrays of string.
• Assign a score of the percentage of similarity on the two strings.

• Trace back the matrix to find the words that we need to delete, insert, and replace.

In the Table 3.2, we illustrate the comparison between 2 sentences using the same approach of Levenshtein algorithm.

**Table 3.2: Compare 2 Sentences Using the Levenshtein Algorithm**

<table>
<thead>
<tr>
<th></th>
<th>the</th>
<th>weather</th>
<th>is</th>
<th>nice</th>
<th>today</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>weather</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>is</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>it</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>nice</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>day</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

The comparison between “the weather is nice today” and “the weather is it nice day” give 2 errors: 1 replacement between “today” and “day” and 1 deletion of the word “it.”

**3.3.5.3 Another Algorithm Used to Compare Two Sentences**

We also tried another algorithm to compare two sentences word by word. It returned the percentage of similar words:

1. Count the frequency of strings of the first sequence in the second sequence and store it in a hash map.
2. Count the frequency of strings of the second sequence in the first sequence and store it in the hash map.

3. If the frequency of any string in sequence one has a frequency of 0, then this string is missing.

4. If the frequency of any string in sequence two has a frequency of 0, then this string is wrong.

5. If the difference in frequency between the same string in both sequences is greater than 0 (i.e., the word is unnecessarily repeated or omitted), then it is wrong.

6. Calculate the number of missed and wrong words and calculate the percentage of similarity.

This algorithm returns the percentage of similar strings, but does not consider the order. If the percentage of similar words is 100%, then we check the order; otherwise, we don’t have to.

3.3.5.4 The Feedback Algorithm

The calculation of the percentage of similarity used this formula:

\[ 100 - \left( \frac{100}{\text{Max}(\text{lengthSentence1}, \text{lengthSentence2})} \times \left( \text{Number of Incorrect words} \right) \right) \]

Where:

\[ \text{Number of Incorrect words} = \text{missed words} + \text{wrong words} + \text{replaced words} \]

After the words that are missed, wrong, and that need to be substituted were identified by back tracing, we highlight the words in the text area by calculating the position of the word in the array and the index of the word in the text area. To deal with the punctuation
marks in the sentences, we stored the words in a hash map identified by the position in the array as the key and the index in the text area as the value.
4 EXPERIMENTAL RESULTS

In the previous chapters, we focused on the methods used to achieve our goal. In this chapter, we focus on the accuracy of the performance assessment of language learning and will illustrate the results of our experimental studies. Particularly, we have developed experiments for each of the three potential uses of this system:

1. Learning a new language: English as a Second Language (ESL) or learning a foreign language.
2. Early education/special education.
3. Real-time translation to support conversation across different languages.

The analysis of learner speech gives effective instant feedback to make the learner aware of implications for fluency that he/she did not observe otherwise. This feedback includes two parts:

1. The percentage of similarity between the spoken text and the original given or translated text.
2. Highlighting words that the learner needs to work on, such as wrong (replaced), missing (deleted), and inserted text.

Furthermore, feedback plays a crucial role in learning. It helps the learner clearly know the needed adjustment. Indeed, it helps the learner to know whether he/she achieved the goal or not. The evaluation system of language learning also helps the trainer to develop training courses that concentrate better on identified weakness and provide a highly personalized learning experience [5].
The important steps included to evaluate language learning progress include:

- Compare the spoken text to the original given or translated text by finding the words that need to be replaced, deleted, and inserted.
- Calculate the percentage of similarity between the two texts.
- Highlight the words in green that need to be replaced from the original given or translated text and spoken text.
- Highlight the words in red that need to be deleted in the spoken text.
- Highlight the words in yellow that need to be inserted in the original given or translated text.

Table 4.1 explains how the notes are determined based on the score of similarity in our prototype.

**Table 4.1: Decision Intervals of the Outcome Notes**

<table>
<thead>
<tr>
<th>Interval of similarity</th>
<th>Note</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0%-33%</td>
<td>Try Again</td>
<td>red</td>
</tr>
<tr>
<td>33%-66%</td>
<td>Not Bad</td>
<td>yellow</td>
</tr>
<tr>
<td>66%-100%</td>
<td>Good Job</td>
<td>green</td>
</tr>
</tbody>
</table>

As a result, the outcomes of our objective performance assessment are:

1. Low grade (noted as Try Again in feedback);
2. Intermediate grade (noted as Not Bad in feedback); or
3. High grade (noted as Good Job in feedback).
4.1 Using the Application for Learning a New Language

When starting the application, the user needs to select the source language and the target language. Currently, we provide in our application support for three languages: American English, French, and Italian. The user will click the microphone button to start talking in the source language (e.g., the user’s native language). As evident from Figures 4.1 to 4.13, the speech recognition of the user is successfully converted to text and is displayed in the designated text area on the screen. Next, the text displayed in the first text area will be automatically translated to the target language, and then translated to audio for the learner. The user can choose to listen again by clicking on the speaker button. When the user is ready to try, he/she needs to click the second microphone button and repeat the words inside the translated text area. The speech recognition will be again successfully converted to text, but this time in the target language, and it will be displayed in the specified text area. Finally, the system will compare the translated text to the last recognized text. The score of similarity will be displayed, and the missed words, wrong words, and the words that need to be substituted will be highlighted.

The following results present a more complete sample and explanation of our experiments.

4.1.1 Testing the Application with a Single Word

A user who wants to learn a foreign language will start by pronouncing a single word.
Figure 4.1: Sample of Low-Grade Performance for a Single Word Test

In Figure 4.1, we tested a single word. The source language is English, and the target language is French. Because the user read the word “couverture” wrong, the word “couverture” and the word “converse” need to be replaced to have a correct result. Thus, they are both highlighted in green. The pronunciation is incorrect, so the score of similarity is 0%.
In Figure 4.2, the source language is French, and the target language is Italian. The user pronounced the word “scarpe” correct, but added another word that was wrong. Consequently, the percentage of similarity is 50%.
In Figure 4.3, we tested the application with only one word. The word “armoire” is translated correctly to the word “cabinet.” The speaker pronounced the word “cabinet” correctly. Thus, the score of similarity is 100%.

4.1.2 Testing the Application with One Phrase

The user can make progress in learning from a single word to a whole phrase.
Figure 4.4: Sample of Low-Grade Performance for a Single Phrase Test

In Figure 4.4, English is the source language, and Italian is the target language. The user reads the phrase “correre attraverso i boschi” which is wrong. Thus, the words that need to be replaced are highlighted in green, and the words that need to be deleted are highlighted in red. The pronunciation is not correct, so the score of similarity is 0%.
Figure 4.5: Another Sample of Low-Grade Performance for a Single Phrase Test

Figure 4.5 illustrates the result of another test with one phrase. English is the source language, and French is the target language. The feedback gives a percentage of similarity of 33% because only the word “bloc” is correct. The user needs to replace the word “retour” with “autour” and the word “de” with the word “du,” which are highlighted accordingly.
Figure 4.6: Sample of High-Grade Performance for a Single Phrase Test

Figure 4.6 illustrates the test of one phrase that received a high-grade result. The pronunciation of the translated text is correct.

4.1.3 Testing the Application with One Sentence

As illustrated in Figures 4.7 to 4.13, the application shows successful assessment results for a short or long sentence.
Figure 4.7: Sample of Low-Grade Performance for Sentence Test

Figure 4.7 tested a sentence that is translated from English to French. The score gives a percentage of similarity of 0%.
Figure 4.8 gives a sample of an intermediate grade performance of one sentence translated from English to French. The percentage of similarity is 66%. The user still needs to replace the word “elle” with “il,” which is highlighted with green, and add the word “si,” which is highlighted in yellow.
Figure 4.9: Sample of High-Grade Performance for Sentence Test

Figure 4.9 illustrates the test of one sentence that earned a high grade. The user translated the text from English to French correctly; therefore, they scored 100%.

4.2 Using the Application for Early Learning

The application can be also used for the learning of a first language for children. A child can write or copy a text and try to repeat to see if he/she is able to read the correct pronunciation, eliminating the need for an instructor. Due to the time limit of this thesis
research, we did not make a new customized GUI for this use. Instead, we will use the same app to demonstrate the use, as shown in Fig. 4.10. A user selects the target language, and then writes or pastes a text into the translated text area. By clicking on the speaker button, the app will synthesize the written text into voice, so the child can listen to the word, phrase, or sentence. Once he/she feels ready to try it out, they can simply click on the microphone button and read the text in the translated text area. Then, the system will capture and assess the speech and give feedback to the user, as shown in the following figures.
Figure 4.10: Sample of Low-Grade Performance Result

Figure 4.10 shows text being typed into the textbox in French and the corresponding assessment results. The pronunciation of the text is completely wrong. Our system provides a correct comparison between the written and spoken text shows: the user needs to substitute the green text with the yellow text. Thus, the user got a score of 0% similarity.
Figure 4.11: Sample of Intermediate-Grade Performance Result

Figure 4.11 represents a test for a user that wants to learn English. This user made progress in pronunciation, with only 30% unmatched text. The user needs to pronounce “dismissed” correctly.
Figure 4.12: Sample of High-Grade Performance Result

Figure 4.12 depicts a completely successful pronunciation of an English learner. The learner received a 100% score of similarity.

4.3 Using the Application for Real-time Translation to Support Conversation Across Different Languages

We can use this application for translation of real-time speech between two languages. Again, due to the time limit of this thesis research, we did not make a new
customized GUI for this use. Figure 4.13 shows an example of successful real-time translation of speech from English to French. It may only use the first two components without the need for the assessment.

Figure 4.13: Sample of Successful Real-Time Translation of Speech from English to French
5 CONCLUSIONS AND FUTURE WORK

5.1 Conclusion

Learning a new language is a challenge for children as well as for adults. Although many applications are available to help people learn new languages, the development of an effective and practical solution that helps users to learn and practice a new language independently at a low cost will significantly improve learning efficiency and outcomes.

In this thesis research, we have successfully developed a language-speaking learning tool that harnesses the latest advances in real-time speech recognition, language translation, text synthesis, AI, and language intelligibility assessment technologies to produce an automatic assessment and instantaneous feedback of language-speaking performance to help achieve an independent-learning workflow. Our experimental results demonstrate that the proposed system can sufficiently and accurately analyze the intelligibility of one’s speaking, correctly identify the mispronounced words, and define a feedback mechanism that localizes and highlights errors for continuous practice toward perfection.

In addition, we have demonstrated three potential uses for our solution and the corresponding workflows:
1. Learning a new language: English as a Second Language (ESL) or learning a foreign language
2. Early education/special education
3. Real-time translation to support conversation across different languages

5.2 Future Work

The research proposed in this thesis has answered many questions and made way for even more questions. Some promising lines of research arise from this work that can be further pursued.

First, the application supports three languages: American English, French, and Italian. However, we can easily add other languages and many accents. For instance, Google Cloud Speech Recognition alone can support over 110 languages. The merging of multiple cloud speech services may further expand on these capabilities as well as speech recognition performances.

The second line of research, as discussed earlier in the thesis, is that our solution can be used by kids for first language learning, for people that have special learning needs, for people that need to learn a foreign language, and for people that need translation. Developing customized GUIs for each application and for mobile devices will be helpful for these particular uses.

Finally, developing a speech recognition service that supports correct punctuation will enhance language understanding and learning, as every language has different rules and different letters. Furthermore, the feedback and the assessment of stress-timed language like English and syllable-timed languages like French or Spanish may be different and need to be further studied.
REFERENCES


