German International University
Informatics and Computer Science

Automating Grading of Short Arabic Speech Answers

Bachelor Thesis

Author: Ahmed Maged
Supervisors: Dr Nada Sharaf
Submission Date: 19 January, 2023

German International University
Informatics and Computer Science

Automating Grading of Short Arabic Speech Answers

Bachelor Thesis

Author: Ahmed Maged
Supervisors: Dr Nada Sharaf
Submission Date: 19 January, 2023

This is to certify that:

(i) the thesis comprises only my original work toward the Bachelor Degree

(ii) due acknowledgement has been made in the text to all other material used

Ahmed Maged
19 January, 2023

Acknowledgments

I would like to express my sincere gratitude to all those who have supported me during the preparation of this bachelor thesis. Firstly, I would like to thank my advisor, Dr. Nada Sharaf, for her guidance, encouragement, and invaluable feedback throughout the entire process. Her expertise and patience were instrumental in helping me to complete this work. I would also like to extend my appreciation to Omar Nael, who provided me with the resources I needed to complete this research. I am also grateful to my family and friends for their unwavering support and understanding during the time I spent working on this thesis. Finally, I would like to acknowledge the hard work and dedication of all the researchers and scientists in the field of Arabic NLP, whose work has greatly influenced my own. This work would not have been possible without the support of all of these individuals, and I am deeply grateful for their contributions.

Abstract

This research studies the use of Transformers, Deep Learning, and Machine Learning algorithms for automatic short arabic speech answers grading without any human intervention. Since as a human being, evaluating student work is a laborious and time-consuming process. Two different approaches were introduced in this project. The initial approach involves transforming the speech data into text form and subsequently utilizing it. The alternative approach is to directly use the speech data without converting it to text. Such conversion was executed through Google Speech-To-Text API. When working with text data that has been converted from audio, a variety of models were trained using embeddings generated from the Universal Sentence Encoder (USE). These models include Support Vector Machines (SVM), Logistic Regression, Recurrent Neural Networks (RNN), Long Short-term Memory (LSTM), ara-BERT, and ara-ELECTRA. These models were used for the classification task. Concerning the speech data, Mel-frequency cepstral coefficients (MFCCs) were utilized as the audio representation in the system. This is a common technique used in speech processing to extract features from audio signals. For the classification of the speech data, several techniques were implemented such as, Support Vector Machine (SVM), logistic regression, and Recurrent Neural Networks (RNN). The performance of the models was significantly better when they were trained with the converted text data, rather than the raw speech data. Through an in-depth examination, it was discovered that using BERT, Electra, Recurrent Neural Networks (RNN), and Support Vector Machine (SVM), resulted in state-of-the-art performance.

Contents

Acknowledgments **III**

1 Introduction **1**

2 Background **3**
2.1 Machine Learning . . . 3
2.1.1 Supervised Machine Leaning . . . 3
2.1.2 Unsupervised Machine Leaning . . . 4
2.1.3 Machine Learning Applications . . . 4
2.2 Deep Learning . . . 5
2.2.1 Functioning Principles of Deep Learning . . . 6
2.2.2 Deep Learning Algorithms . . . 6
2.2.3 Deep Learning Applications . . . 7
2.3 NLP - Natural Language Processing . . . 7
2.3.1 Data Pre-proceesing . . . 8
2.4 Embeddings . . . 9
2.4.1 Word Embeddings . . . 10
2.4.2 Sentence Embeddings . . . 10
2.5 Arabic Language Challenges . . . 10
2.5.1 Morphology . . . 10
2.5.2 Dialects Variety . . . 11
2.5.3 Lexical ambiguity . . . 11
2.6 Transformers . . . 11
2.6.1 BERT . . . 12
2.6.2 ELECTRA . . . 12
2.7 Metrics . . . 12
2.7.1 Word Error Rate . . . 12
2.7.2 Quadratic Weighted Kappa . . . 13

3 Related Work **15**
3.1 Automating Grading of Short Text Answers . . . 15
3.1.1 Text Datasets . . . 15
3.1.2 Machine Learning . . . 17
3.1.3 Deep Learning/Transformers . . . 19

3.2 Automating Grading of Short Speech Answers . . . 23
3.2.1 Speech Datasets . . . 23
3.2.2 Automating Grading of Short Speech Answers Pipeline . . . 24

4 Methodology **27**
4.1 Dataset . . . 27
4.2 Automating Grading of Short Text Answers . . . 28
4.2.1 Converted Text Dataset Preparation . . . 28
4.2.2 Converted Text Dataset Model Architecture . . . 31
4.3 Automating Grading of Short Speech Answers . . . 35
4.3.1 Speech Dataset Preparation . . . 35
4.3.2 Speech Dataset Model Architecture . . . 35

5 Results & Discussion **37**
5.1 Experiment Tools/Metrics . . . 37
5.1.1 Quadratic Weighted Kappa (QWK) . . . 37
5.1.2 Word Error Rate(WER) . . . 37
5.2 Automating Grading of Short Text Answers . . . 39
5.2.1 Speech-to-Text Conversion Results . . . 39
5.2.2 Converted Text Dataset Preparation Results . . . 40
5.2.3 Converted Text Dataset Models Results . . . 41
5.3 Automating Grading of Short Speech Answers . . . 42
5.3.1 Audio Representations Results . . . 42
5.3.2 Speech Dataset Models Results . . . 43
5.4 Discussion . . . 44
5.4.1 Automating Grading of Short Text Answers . . . 44
5.4.2 Automating Grading of Short Speech Answers . . . 45
5.4.3 Overview . . . 46

6 Conclusion **47**

7 Future Work **49**

Appendix **50**
List of Figures . . . 51
List of Tables . . . 52

References **58**

Chapter 1

Introduction

Automated grading systems are a useful tool for teachers, as they can save a significant amount of resources and time spent on grading exams. This allows teachers to focus on other important tasks that contribute to a better overall educational experience for their students. However, care must be taken to ensure that these systems are fair and accurate. Therefore, a lot of research has been conducted to ensure that automated grading systems can justify replacing traditional manual methods of grading.

Automated scoring has been around for sometime and has primarily been used for multiple-choice and true or false questions where there is a set of predefined options and one correct answer. However, there are other forms of automated scoring such as Automated Essay Scoring and Automated Short Answer Scoring which evaluates a student's language proficiency and understanding of a specific context respectively

Automated short answer scoring is used to assess a student's knowledge of a specific subject matter or domain, such as science, math, or biology. This type of scoring prioritizes evaluating the student's knowledge over their language proficiency. Therefore, issues such as spelling and grammar errors are often disregarded in these systems as they are not relevant to the evaluation of the domain-specific question.

Reference-based systems [30] is a common and effective method for implementing automated grading systems. These systems rely on comparing a student's answer with a pre-determined reference answer. Similarity measures [42], such as cosine similarity and Levenshtein distance, are used to determine how closely the student's answer matches the reference answer. These measures are used to determine the grade of the student's answer. This technique is efficient and easy to implement, which makes it popular among researchers and practitioners.

As machine learning technology has advanced, response-based systems have become increasingly popular for automated grading. These systems do not rely on a single reference answer, but instead, use a set of answers that have been scored by humans. This set of human-labeled answers is used to train a machine learning model in a supervised manner. This allows the model to learn from a diverse set of answers and understand

the nuances of different types of responses, rather than being limited to a single reference answer. This approach can lead to a more accurate and fair system than reference-based systems, but it also requires more labeled data to train the model. But this remained unexplored in the Arabic language. The only two response-based approaches for Arabic short answer scoring were presented by Abdelrahman ElNaka et al. [26] and Omar Nael et al. [41] where both researches introduced Automating Grading of Short Arabic Text Answers.

When it comes to our research we introduce Automating Grading of Short Arabic Speech Answers which was never explored before. The work in this research tries to fill in this gap by contributing both machine learning and deep learning response-based systems for Arabic short speech answer scoring.

In this research, two main approaches are presented.The initial approach involves transforming the speech data into text form and subsequently utilizing it. We then select the most appropriate embedding type, which is the Universal Sentence Encoder, for our numerical representation. Afterward, we implement both Support Vector Machine with fine-tuned parameters and Logistic Regression. Additionally, we also use Recurrent Neural Networks (RNNs) and Long Short-term Memory (LSTMs) models for deep learning. For the transformer-based models, we use ara-BERT [11] and ara-ELECTRA [12].

The second method employed in this study was to use the speech data as is, without any modification. This approach posed a greater challenge as it required more intricate pre-processing techniques and was more difficult to convert into numerical forms. To overcome this, Mel-Frequency Cepstral Coefficients (MFCCs) were utilized as the numerical representation of the speech data. Subsequently, fine-tuned Support Vector Machines (SVMs) and logistic regression models, and Recurrent Neural Network (RNN) deep learning models were also employed for our classification

Chapter 2

Background

2.1 Machine Learning

A sub-field of artificial intelligence[16] and computer science called Machine learning [64]. Focuses on using data and algorithms to simulate how humans learn, gradually improving the accuracy of the system. Machine learning algorithms build a model using sample data, also referred to as training data, in order to make predictions or decisions without being expressly programmed to do so. Machine learning has subclasses called Supervised learning and Unsupervised Learning

2.1.1 Supervised Machine Leaning

Supervised Learning[6] is defined by its use of labeled datasets in training algorithms to accurately classify data. The model modifies its input data weights until the model is properly fitted, supervised machine learning can be classified into two categories: Regression and Classification

A few examples of supervised machine learning algorithms include:

- **Support Vector Machine (SVM)**

  Support Vector Machines (SVMs)][48, 27] are used for classification and regression tasks, it is mostly used in classification problems. They are a powerful and flexible tool for building machine learning models and have been widely used in a variety of applications. SVMs are based on the idea of finding a hyperplane in a high-dimensional space that maximally separates different classes. In the case of a non-linear SVM, the hyperplane is a non-linear boundary

- **Logistic Regression**

Logistic regression is a linear model, which means that the decision boundary between the two classes is a linear hyperplane in the feature space. This means that the model makes predictions based on the combination of the input features using a linear function.

2.1.2 Unsupervised Machine Leaning

Unsupervised learning[7] is a kind of machine learning where users don't have to supervise the model, instead, it allows the model to operate independently in order to find patterns and information. It deals with unlabelled data, when dealing with Unsupervised learning, users can carry out more difficult processing tasks than when dealing with supervised learning, Unsupervised machine learning can be classified into two categories: Clustering and Association

A few examples of unsupervised machine learning algorithms include:

- K-means clustering
- K-NN (k nearest neighbors)

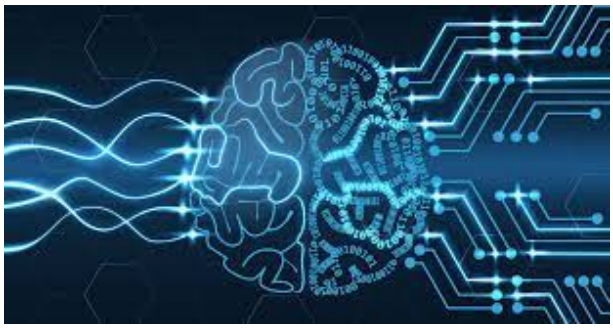


Figure 2.1: Machine Learning [8]

2.1.3 Machine Learning Applications

Machine learning nowadays is very important and is developing very quickly. Without realizing it, we use machine learning every day in applications like Google Maps, Google Assistant, Alexa.

Here are some real-world applications of Machine Learning:

1. **Image Recognition**[2]

   One of the popular uses of ML is image recognition. It is used to identify things like digital images, people, places, and objects. Automatic friend tagging recommendation is a common use of image recognition and face detection used by Facebook

2. **Speech Recognition**[2]

   Speech recognition, often known as "Speech to text" or "Computer speech recognition," is the process of turning spoken commands into text. Speech recognition applications currently use machine learning algorithms extensively. Used by Google Assistant, Siri, etc.

2.2 Deep Learning

Deep learning, also called deep neural networks is a machine learning method as shown in 2.2 that teaches computers to perform tasks that comes naturally to humans. In deep learning, a computer model learns to carry out classification tasks directly from images, text, or even sound. High accuracy can be attained by deep learning models, sometimes even outperforming human ability.

Deep learning has two main limitations which are as follows

- Deep learning requires large amounts of labeled data
- Deep learning needs a lot of processing power.

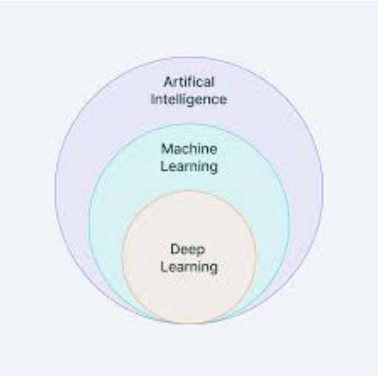


Figure 2.2: Deep Learning Formation [5]

2.2.1 Functioning Principles of Deep Learning

Deep learning [45] models are sometimes referred to as deep neural networks because the majority of deep learning techniques use neural network architectures, the word 'deep' describes the number of hidden layers in the neural network as shown in 2.3. While Traditional neural networks only have two or three hidden levels, deep networks can have more than 100 layers

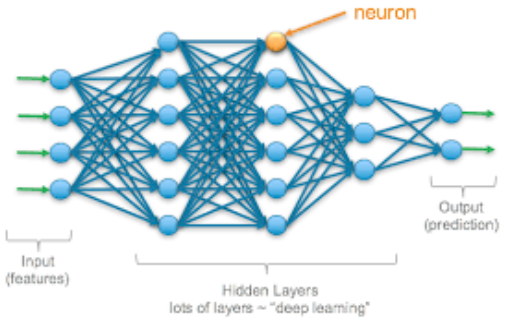


Figure 2.3: Deep Learning network architectures [51]

2.2.2 Deep Learning Algorithms

- **Recurrent Neural Networks (RNNs)**

  Apple's Siri and Google's voice search use RNNs. Unlike traditional neural networks, RNNs[45] utilize the network's sequential information, this property is crucial in numerous applications where the embedded structure in a data sequence transmits valuable information. RNN also is the first algorithm with an internal memory that remembers its input. This makes it ideal for solving machine learning issues involving sequential data.

- **Long Short Term Memory Networks (LSTMs)**

  Long Short-Term Memory (LSTM) [24] networks were developed to address the issue of long-term dependencies in RNNs caused by the vanishing gradient problem, LSTMs have feedback connections. This feature allows LSTMs to handle data sequences such as text and audio. Which makes them helpful for speech recognition, and music composition

2.2.3 Deep Learning Applications

Deep Learning has found its application in almost every business. A few examples of deep learning applications include:

- **Language Translations**

  Technology businesses are paying close attention to machine translation. This investment, along with recent deep learning advancements has led to considerable improvements in translation quality. According to Google, Google Translate's translation accuracy increased by 60% after switching to deep learning, more than 100 languages can now be translated by Google and Microsoft with near-human accuracy in a lot of them.

- **Speech recognition**

  Automatic speech recognition (ASR) is a function that allows programs to process human speech in a written format, It is often confused with voice recognition, voice recognition only attempts to recognize the voice of a single user, for speech recognition apps and APIs to comprehend and analyze human speech, they integrate the grammar, syntax, structure, and composition of speech and audio signals.

2.3 NLP - Natural Language Processing

The ability of computer software to comprehend human language is known as natural language processing[59]. Artificial intelligence[16] is used in natural language processing to take real-world data and process it in a way that computers can comprehend. Computers have reading programs and microphones, the same as people that have ears and eyes. Computers also have a program to process their various inputs same as people that have brains, these inputs are eventually translated into computer-readable code during processing. One of the main phases of NLP is Data Pre-processing.

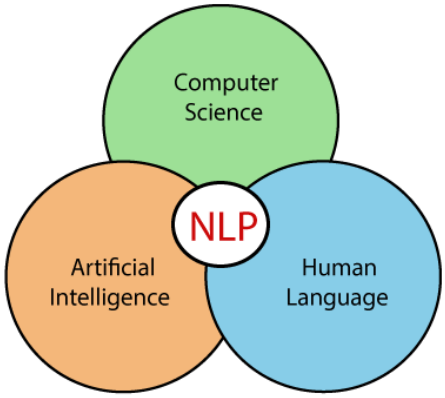


Figure 2.4: Natural Language Processing [14]

2.3.1 Data Pre-proceesing

Data Pre-processing is the cleaning and preparation of the dataset to improve the dataset usability and make it more comprehensible to the model.

This can be accomplished in a number of ways such as:

- Data Cleaning
- Feature Engineering
- Sampling Data
- Data Transformation
- Dimensionality Reduction

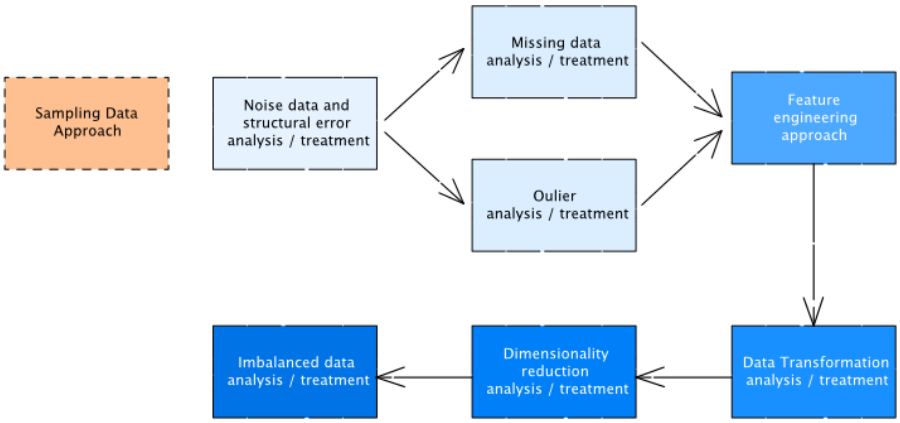


Figure 2.5: Pre-Processing Pipeline [1]

The pre-processing steps outlined in figure 2.5 are considered to be the basic pre-processing techniques. In contrast, Natural Language Processing (NLP) text pre-processing, as outlined in reference [59, 29], encompasses a range of techniques that are commonly used to reduce the dimensionality of text data. These techniques include methods such as:

- **Tokenization[29]:** It is a process of splitting each sentence into words

- **Lower casing[29]:** It is a process of converting a word to lowercase, words like Play and play have the same meaning but if Lower casing is not applied they will be represented as two distinct words in the vector space model

- **Stop words removal[47, 29]:** Stop words are commonly used words (a, an, the, etc.). These words don't actually mean anything important so they are removed such that only the special words that provide the most information about the text stay

- **Stemming[34]:** It is a process that involves removing the suffixes from words to obtain their stem

- **Lemmatization[29]:** It is a process of resolving a word to its lemma. It is harder to build than stemming but it's preferable because it does a morphological analysis of the words.

2.4 Embeddings

Embeddings [17] make it possible to represent longer pieces of text numerically as vectors, that computer algorithms, such as Machine Learning and Deep Learning models can handle directly. Conventional techniques like TF-IDF and one-hot encoding have a significant drawback. Which is that they are unable to capture the fine-grained semantic information found in human languages.

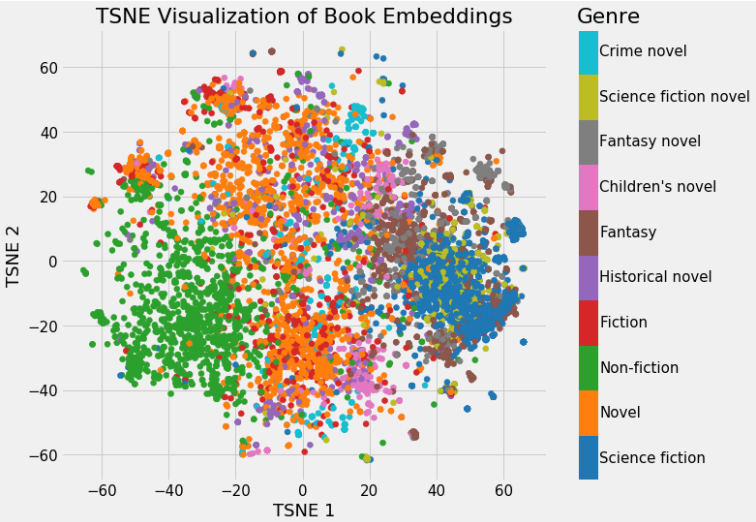


Figure 2.6: Embeddings Example [21]

2.4.1 Word Embeddings

Word embeddings generate a vector for each word depending on the context. Consider the sentences miss Mona is such a kind woman and he was going to miss the bus. With traditional word embeddings, the word vector for miss would be the same in both sentences while it has different meanings

Here are a few examples of Word Embeddings.

- **Google word2vec**
- **BERT word embedding**

2.4.2 Sentence Embeddings

Sentence embeddings can embed a whole sentence as one vector, it captures various semantic links between sentences, including similarity, contradiction, and entailment

Here are a few examples of Sentence Embeddings.

- **Doc2Vec:** It's an extension of Word2Vec
- **SentenceBERT**
- **Universal Sentence Encoder**

2.5 Arabic Language Challenges

There are not enough publicly available datasets and resources for the Arabic language. When it comes to speech Arabic datasets there are almost none available. Researchers solve this problem by translating the datasets to the Arabic language as described in research [41]. There is a heck of a lot of ambiguity in the Arabic language due to its extremely rigid rules and complex grammatical structure. Making it a challenging language in the context of NLP, some of these difficulties are as follows:

2.5.1 Morphology

When compared to English, Arabic words can take on a variety of forms, leading to multiple versions of a single word. This complicated morphology is one of the obstacles [20]. That's why Building Arabic stemming/lemmatizing tools like farasa [9] and tokenizers are more challenging.

2.5.2 Dialects Variety

The variation of dialects is a challenge in the Arabic language. Every country has its own version of Arabic. Although the general structure of Arabic is the same in most dialects, each dialect comes with additional rules and vocabulary [28], And for sure this may degrade the effectiveness of Arabic datasets.

2.5.3 Lexical ambiguity

Comparatively speaking to other languages, Arabic has a greater number of lexical ambiguity factors. Diacritization is one of the key factors that contribute to lexical ambiguity. Although the main purpose of diacritics is to identify the pronunciation of the word, it also determines the words' meaning in some cases [35], as shown in figure 2.1

Arabic	Meaning
عَلَّمَ	Taught
عِلْمٌ	Knowledge
عَلَمٌ	Flag
عَلِمَ	Knew
عُلِمَ	Is known
عَلَّمْ	Teach
عُلِّمَ	Is taught

Table 2.1: Lexical ambiguity in Arabic [41]

2.6 Transformers

A transformer [60] is a notable deep learning model that has been widely adopted in a variety of domains, including natural language processing (NLP), and speech processing. Due to the achievements that transformers have had in different fields. Transformers have become the mainstream NLP architecture, particularly for Transformer-based pre-trained models (PTMs). Google initially introduced Transformers in 2017. NLP tasks were handled by language models predominantly using recurrent neural networks (RNN) and convolutional neural networks (CNN) at the time of their inception.

2.6.1 BERT

Bidirectional Encoder Representations from Transformers (BERT) [22] is an open-source framework for machine learning and natural language processing (NLP). BERT is aimed to assist computers to understand ambiguous words in the text by establishing context from the surrounding text. The BERT framework, a new language representation model from Google AI, employs pre-training and fine-tuning to produce state-of-the-art models for a variety of tasks. Such as question-answering systems, sentiment analysis, and language inference. BERT is an attention mechanism that discovers contextual relationships between words in a text, in its default form. BERT is pre-trained on two distinct but related NLP tasks using this bidirectional capability: Masked Language Modeling (MLM) and Next Sentence Prediction (NSP)

Since BERT is pre-trained on a large corpus of English text. Research [11] introduced a new pre-trained BERT model on Arabic language, which is named araBERT. AraBERT was tested on three Arabic tasks that are different in nature: Sentiment Analysis, Named Entity Recognition, and Question Answering. The experiment results show that ARABERT achieves state-of-the-art performances on most datasets.

2.6.2 ELECTRA

ELECTRA [19] is a transformer with a unique pre-training method that trains both the generator and discriminator transformer models. The generator, replaces tokens in the sequence, trained as a masked language model, while the discriminator (the ELECTRA contribution) attempts to determine which tokens are replaced by the generator. This pre-training activity is referred to as replaced token detection and serves as a substitute for input masking.

Since ELECTRA is pre-trained on a large corpus of English text, Research [12] introduced a new pre-trained ELECTRA model pre-trained on Arabic language, which is ara-ELECTRA. The model is pre-trained using the replaced token detection objective on large Arabic text corpora. They evaluated the model on multiple Arabic NLP tasks, and it results showed that ara-ELECTRA outperforms state-of-the-art Arabic language representation models

2.7 Metrics

2.7.1 Word Error Rate

Word Error Rate (WER)[4] is a common metric for evaluating the performance of speech recognition systems. It represents the number of errors (substitutions, deletions, and insertions) in transcribed text, compared to a reference transcript. WER is calculated as

WER = (S + D + I) / N, where S is the number of substitutions, D is the number of deletions, I is the number of insertions, and N is the total number of words in the reference transcript. Thus, WER measures the total number of errors in the transcribed text as a fraction of the total number of words in the reference transcript. A lower WER indicates better performance, with a WER of 0 indicating perfect transcription accuracy. WER is a widely used metric in speech recognition research and development, and it is also commonly used to evaluate the performance of commercial speech recognition systems, and here is an example of how WER works between a sentence and its reference as shown in figure 2.7

Figure 2.7: WER metric of two sentences

2.7.2 Quadratic Weighted Kappa

QWK (Quadratic Weighted Kappa)[3] is a metric used to evaluate the agreement between two raters or annotators. It is often used in the context of measuring the inter-rater reliability or inter-annotator agreement of a classification task. QWK takes into account the possibility of agreement occurring by chance and adjusts for this by calculating the expected agreement given the distribution of ratings across all categories. It is a measure of the extent to which the ratings of two raters are correlated, with a value of 0 indicating no agreement beyond chance, a value from 0-0.2 indicating Poor agreement, a value from 0.2-0.4 indicating Moderate agreement, a value from 0.4-0.6 indicating Good agreement,

a value from 0.6-0.8 indicating Very good agreement, a value from 0.8-1 indicating Perfect agreement, and with a value of 1 indicating Complete agreement. QWK is considered to be a more robust and reliable measure of agreement than simple percent agreement, as it takes into account the level of disagreement between the raters as well as the specific categories being rated.

Chapter 3

Related Work

3.1 Automating Grading of Short Text Answers

3.1.1 Text Datasets

3.1.1.1 Arabic

The work presented by Abdelrahman ElNaka et al.[26] introduced to us the AraScore-Dataset for the first time, achieving state of the art performance. The following steps were taken to create the dataset:

- The dataset was collected from a school in English.
- The dataset was normalized to a scale from 0-3.
- Augmenting the data was done using back-translation. This translates the text to any language and then back to its original language.

Another research published by O. Nael et al.[41] introduced a new dataset which is ASAP short answer scoring dataset. OCR tools were used to convert the scanned answers to text. Google translate API was used to translate the dataset from English to Arabic, leading to Some loss of data. The translation step also led to the disregarding of diacritics, which is usually not included by students in their answers, causing a problem in automated NLP systems because in the Arabic language a word could have several meanings based on its diacritics as mentioned in section 2.5.3. The dataset contains 10 questions with 1600 answers for each question with a minimum grade of 0 to a maximum grade of 3.

3.1.1.2 Other Languages

- **English**

  Work published by Nitin Madnani et al.[38] introduced to us an English corpus that contains over 130 questions spanning 4 subjects: Arts, Math, English language, and Social Studies containing 230,000 responses. Only 10 publicly content-based questions out of the 130 are available from the Hewlett Foundation's Automated Student Assessment Prize competition. The remaining questions are not accessible to the public and are now being used in various classroom settings.

  Brian Riordan et al. [50] introduced to us 3 English datasets which are ASAP-SAS [55], Power grading [62], and SRA. ASAP-SAS contains 10 prompts with 2200 responses each covering science, biology, and ELS. Responses are scored on a scale of 0 to 2 or 0 to 3 depending on the prompt. Power grading contains 10 prompts with about 700 short responses each. SRA consists of two subsets[25]. Beetle and SciEntsBankk, the SRA dataset contains a large number of prompts while SAS datasets contain few prompts [55].

  The research presented by Jörg Sawatzki et al. [53] introduced The English short answer grading dataset of the University of North Texas [40]. The dataset contains 87 questions, and on average 28.1 evaluated answers per question about the topic Data Structures. The questions are short, they can usually be answered in only one sentence and no knowledge transfer is required.

- **German**

  Jörg Sawatzki et al. [53] introduced a German dataset extracted from an online exam in the learning management system Moodle3. It contains 233 questions, and on average 15.4 evaluated answers per question. The maximum achievable score depends on the question, a maximum of 6, 8, or 10 points can be achieved. The dataset includes annotations with the criteria for grading performance. While many model answers contain only short hints.

- **Hindi and Marathi**

  The dataset used in the research published by Dolly Agarwal et al. [10] is ScAA which is in Hindi and Marathi languages comprising 8 science topics with 4 questions per topic, as shown in table 3.1. The dataset is collected via an Android app causing a problem where any child could take the assessment whenever they like without adult supervision through the phone interface. This led to the presence of noise within the dataset.

	Hindi	Marathi
Total Questions	32	32
total answers	10988	1955
total unique answers	7205	1435
total correct answers	3843	488
Average unique correct answers per question	41	7
Average answers length (in words)	15	15

Table 3.1: ScAA dataset [10]

3.1.2 Machine Learning

3.1.2.1 Arabic

Previous research published by Abdelrahman ElNakaa et al.[26] implemented SVC, Random Forest Classifier, Support Vector Machine Regressor, Random Forest Regressor, Multi-layer Perceptron Regressor, and Multi-layer Perceptron Classifier. These models were then evaluated on three different datasets which are AraScore-Dataset, AR-ASAG, and Hewlett Foundation SAS dataset. Their research focuses on response-based systems due to the research gap in this area. In the experiment, Cross-validation was used instead of single-train-test split. Since single-train-test split may yield a bias while cross-validation eliminates such biases while performing better with small datasets. The results show that SVM performed better when using the Hewlett Foundation SAS dataset. While Random Forest Learners performed better when using AR-ASAG and AraScore Datasets, as shown in tables 3.2 3.3 3.4 3.5.

Dataset	SVM	Random-Forest	Multi-Layer Perceptron Regressor
Hewlett Foundation SAS Dataset	0.401	0.741	0.213
AR-ASAG	1.218	0.957	0.359
AraScore-Dataset	0.582	0.255	0.091

Table 3.2: Regressor MSE Results using Word-Embeddings [26]

Dataset	SVM	Random-Forest	Multi-Layer Perceptron Regressor
Hewlett Foundation SAS Dataset	0.372	0.733	0.132
AR-ASAG	1.103	0.873	0.265
AraScore-Dataset	0.437	0.193	0.073

Table 3.3: Regressor MSE Results using Sentence-Embeddings [26]

Dataset	SVM	Random-Forest	Multi-Layer Perceptron Regressor
Hewlett Foundation SAS Dataset	49.323%	32.621%	60.263%
AR-ASAG	26.216%	34.577%	55.372%
AraScore-Dataset	69.288%	81.443%	91.970%

Table 3.4: Classier MSE Results using Word-Embeddings [26]

Dataset	SVM	Random-Forest	Multi-Layer Perceptron Regressor
Hewlett Foundation SAS Dataset	54.347%	34.225%	66.374%
AR-ASAG	32.899%	35.303%	60.270%
AraScore-Dataset	72.350%	83.566%	94.701%

Table 3.5: Clasaifier MSE Results using Sentence-Embeddings [26]

3.1.2.2 English

Work presented by Nitin Madnani et al. [38] tested 8 models which are as follows:

- Random forest Classifier
- Logistic Regression
- SVC(linear)
- SVC(RBF)
- Random Forest Regressor
- SVR(linear)
- SVR(RBF)
- Rescaled SVR

Mean squared error (MSE) was the default metric in their research. Cross-validation [54] due to the same reasons mentioned in section 3.1.2.1. It was also observed that the average response length of a question had a larger impact on the number of extracted features than the number of responses available for each question. Their research also showed us important notes that spelling or grammatical mistakes aren't important in scientific subjects like Computer Science, Math, etc as long as the specific information

is included in the answer. The corpus used contains 130 questions spanning 4 subjects, and more than 230,000 human-scored responses. Putting into consideration that the response's quality is not in its length, but in its content. The results had shown that the best performance was achieved by probabilistic support vector classier with linear and RBF kernels, and tuned hyperparameters as shown in table 3.6. The results showed that non-tuned hyper-parameters led to lower performance, especially on SVMs. Non-tuning hyper-parameters is also worse for non-linear learners than the linear learner, and worse for regressors than classifiers. The results also show that there are no significant differences in performance due to linearity, learner type, or learner family. linearity is either linear or non-linear, type is either a classifier or regressor, and family is either a logistic regression, random forest, or support vector machine.

Learner	not tuned	tuned
Logistic Regression	.401	.391
Random Forest Classifier	.385	.368
Random Forest Regressor	.356	.356
SVC (linear)	.336	.326
SVR(linear)	.514	.388
SVC(RBF)	.434	.321
SVR(RBF)	.723	.342
Rescaled SVR	.546	.343

Table 3.6: Learners Average MSE [38]

3.1.3 Deep Learning/Transformers

3.1.3.1 Arabic

In other researches, their focus was on implementing a deep learning model on Arabic datasets. The work published by Omar Nael et al.[41] uses a deep learning response-based system due to the shortage of this approach. Different preprocessing steps have been conducted, firstly cleaning the text from numbers and symbols and some artifacts from the translation process, and secondly lemmatizing using farasa [9], When running the experiment on BERT and ELECTRA light preprocessing was performed, the same cleaning process was used but lemmatization was not performed. When it comes to the methodology, Sentence embeddings were generated using Universal Sentence Encoder. RNN basic models were used such as standard RNN, LSTM, and Bi-LSTM, all of these models consist of 5 layers. BERT and ELECTRA models were used too. 20% of the training data were split for validation. There were two methods presented. The first is to train the model on each question individually, and the other method was training the

model on the whole dataset at once including question ID in the features, All models were tested on both these methods, and the results had shown that BERT and ELECTRA perform state of the art. This is probably due to both models are deeply bidirectional. Standard RNN, LSTM, and Bi-LSTM have similar scores as shown in table 3.7. It was also shown that training the models on the whole dataset showed better results than training on each question individually.

Question	Reference-based	SVM	RNN	LSTM	Bi-LSTM	BERT	ELECTRA
1	0.59	0.55	0.71	0.71	0.74	0.77	0.81
2	0.52	0.56	0.47	0.51	0.51	0.69	68
3	0.44	0.51	0.65	0.63	0.64	0.65	0.66
4	0.43	0.45	0.62	0.65	0.65	0.62	0.63
5	0.38	0.49	0.51	0.52	0.68	0.80	0.81
6	0.40	0.46	0.80	0.80	0.81	0.84	0.86
7	0.49	0.46	0.42	0.42	0.44	0.70	0.65
8	0.50	0.48	0.53	0.55	0.53	0.52	0.53
9	0.45	0.52	0.72	0.71	0.71	0.68	0.70
10	0.47	0.52	0.72	0.71	0.71	0.68	0.70
Mean QWK	0.46	0.50	0.65	0.63	0.65	0.70	0.71
Whole dataset	0.46	0.55	0.71	0.70	0.72	0.77	0.78

Table 3.7: Models results [41]

3.1.3.2 English

Research published by Brian Riordan et al. [50], converted the word tokens of each response to embeddings, then features are extracted from the embeddings to the convolutional network layer. Then this output forms the input to the LSTM layer. These hidden states of the LSTM are aggregated in either the mean over time layer or attention layer. An attention mechanism is used which takes the dot product of each LSTM hidden state and a vector that is trained with the network. A single vector from the aggregate layer is used as the input for a fully connected layer. This layer generates a scalar (regression) or class label (classification) as shown in figure 3.1

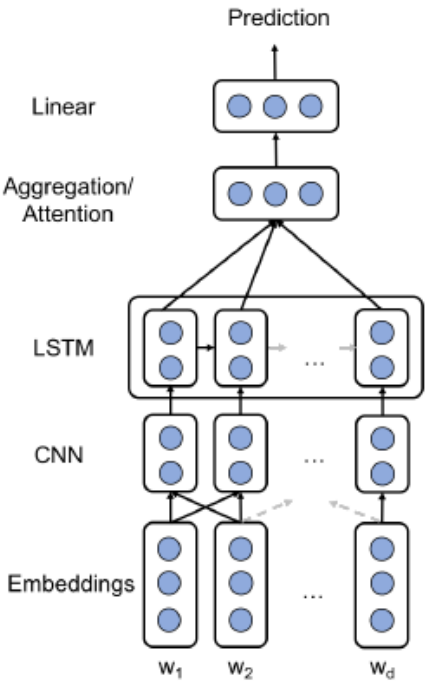


Figure 3.1: Neural Network Architecture [50]

Regarding the setup, the researchers adopted the work done by research [58]. The texts are tokenized with standard NLTK tokenizer and lowercased. Each response is padded with a dummy token to a uniform length. Mean squared error loss for regression models was used and cross-entropy loss for classification models was used. As for the training network, RMSProp is used with p=0.9 and learning rate=0.001. Results have shown that the mean-over-time layer achieves the best results on the SRA-SEB dataset. It was hypothesized that the mean-over-time layer is helpful when the input consists of longer responses. On all datasets, pre-trained embeddings with tuning performed better. Where Regression outperformed classification.

3.1.3.3 German

The work presented by Jörg Sawatzki et al.[53] focuses on Automatic Short Answer Grading (ASAG). Two datasets were used, a German dataset was taken from the Business Administration of a German bachelor's program, and an English dataset was taken from the University of North Texas [40]. Pearson correlation [15], mean absolute error, and root mean square error were the evaluation metrics used. The architecture is based on the Ans2vec approach [31]. Firstly the model answer and the student answer are converted into two embedding vectors MA and SA. Then the dot product and the absolute difference of the two embedding vectors are calculated and concatenated, The result of

the concatenation is the input vector for a linear model in order to predict the score. The Models used are as follows,

- **Ans2vec-MUSE-Logit**
- **ans2vec-Skip-Logit-Baseline[31]:** uses combine-skip vectors for the embeddings and logistic regression as a classifier. Since no skip embeddings are available for German, This model is only evaluated for English.
- **Ans2vec-Skip-SVM**: The SVM [27] is used for classification and due to the lack of German combine-skip embeddings, they also evaluated this model on the English dataset only.
- **Ans2vec-MUSE-SVM:** The MUSE [61] is used for the embeddings. SVM [27] is used for classification.
- **BERT Fine-Tuned architecture:** It is based on BERT [23]. It's supplemented with a linear regression layer that provides a prediction of the score given an answer. Then the model takes the model answer and the student answer without prior embedding as input, and then separates the model answer and the student answer with a separator token, and performs tokenization into word pieces. this approach is used in regression rather than classification since our German and English datasets scores are integer values. The BERT models used are as follows. BERT-EN[23], BERT-DE-Deepset[23], BERT-Multilingual [23].

Regarding the English dataset the results had shown that BERT models provide the best results, BERT-EN is the best of all models but BERT-Multilingual provides only slightly worse numbers as shown in table 3.8.

Model	Pearson	RMSE	MAE
[39]	0.63	0.91	-
Ans2vec-Skip-Baseline	0.33(+0.0%)	1.27(+0.0%)	0.73 (+0.0%)
Ans2vec-Skip-SVM	0.49 (+48.6%)	1.09 (-14.0%)	0.60 (-16.8%
Ans2vec-MUSE-Logit	0.38 (+13.6%)	1.24 (-2.2%)	0.69 (-4.5%)
Ans2vec-MUSE-SVM	0.56 (+67.5%)	1.02 (-19.1%)	0.56 (-23.7%)
BERT-EN	0.79 (+138.5%)	0.69 (-45.3%)	0.41 (-43.3%)
BERT-Multilingual	0.79 (+137.1%)	0.70 (-44,6%)	0.43 (-44,6%)

Table 3.8: Basic experiments with the English dataset [53]

Concerning the German dataset, results had shown that Ans2vec-MUSE-Logit performs slightly better on the German data set, while Ans2vec-MUSE-SVM performs

slightly better on the English data set. The results of BERT[23] on the german dataset are only slightly worse than the English data. The best model is BERT-DE-DBMDZ as shown in table 3.9.

In this research, they made a small experiment which is to remove the annotations and re-evaluate BERT-DE-DBMDZ. Only annotations for the maximum achievable score of each question are added to the model answers. The results showed that removing the annotations reduces the classification accuracies.

Model	Pearson	RMSE	MAE
Ans2vec-MUSE-Logit	0.39	2.52	1.87
Ans2vec-MUSE-SVM	0.44 (+11,2%)	2.68 (+6,6%)	1.90 (+1,9%)
BERT-DE-Deepset	0.74 (+89.2%)	1.76 (-30.3%)	1.31 (-29.7%)
BERT-DE-DBMDZ	0.75 (+90.2%)	1.75 (-30.6%)	1.30 (-30.6%)
BERT-DE-DBMDZ	0.71 (+81.4%)	1.83 (-27.4%)	1.38 (-26.3%)

Table 3.9: Basic experiments with the German data set [53]

3.2 Automating Grading of Short Speech Answers

3.2.1 Speech Datasets

Three datasets were represented in the research published by Nitin Kamalesh Palanisamy et al. [43], which are as follows: ESC-50, UrbanSound8K, and GTZAN. The Environment Sound Classification(ESC-50)[44] consists of 2000 clips belonging to 50 classes. Each lasting 5 seconds. The clips consist of a variety of environmental sounds, ranging from birdsong to car horns. The UrbanSound8k dataset[52] consists of 8732 clips belonging to 10 classes of different urban sounds. Each clip lasts less than or equals 4 seconds. GTZAN dataset consists of 1000 30s-long music clips, belonging to 10 distinct genre classes

The YouTube-100M dataset was represented in the research presented by Shawn Hershey et al. [32]. It consists of 100 million YouTube videos, 70M samples for training, 10M for evaluation, and 20M video validation. Videos average 4.6 minutes each. Each video is labeled with 1 or more identifiers from a set of 30,871 labels. The labels for each video are assigned automatically based on context, image content, and metadata (title, description, comments, etc.). The labels range from very general (e.g. "Song") to very particular (e.g. "Cormorant"). A few examples are shown in Figure 3.10.

Label prior	Example Labels
0.1...0.2	Song, Music, Game, Sports, Performance
0.01...0.1	Singing, Car, Chordophone, Speech
0^{-5}	Custom Motorcycle, Retaining Wall
10^{-6}	Cormorant, Lecturer

Table 3.10: Examples of labels from the 30K set. [32]

3.2.2 Automating Grading of Short Speech Answers Pipeline

The pipeline of short speech answers grading consists of two main steps. Firstly, the audio clips should be converted into a feature representation that is machine-readable. One of the most popular libraries for these feature representations is Liborsa. Secondly, is to implement the most appropriate model for the classification. Deep learning is more common when it comes to audio/image processing.

Work presented by Kamalesh Palanisamy et al.[43] started by choosing the most suitable representations on the ESC-50 dataset, which were mentioned earlier in section 3.2.1. Three feature representations were teseted which are as follows: Log-Spectrograms, Log-Melspectrograms MFCCs, and Gammatone-Spectrogram. A simple CNN architecture similar to SoundNet was used as a baseline for this experiment, and Log-Melspectrograms showed the best representation. Three standard models trained on ImageNet were used, which are as follows.

- **Inception[57]:** An Inception is a combination of all the layers, 1x1 Convolutional layer, 3x3 Convolutional layer, and 5x5 Convolutional layers.
- **ResNet[33]:** It consists of several blocks stacked on top of each other. Each residual block has two 3x3 convolutional layers. Each convolutional layer is followed by a batch normalization layer and a ReLU activation function.
- **DenseNet[33]:** It connects each layer to every other layer in a feed-forward fashion.

The results had shown that simple pre-trained ImageNet models can achieve state-of-the-art performance on both ESC-50 and UrbanSound8K datasets as shown in 3.11.

Model	GTZAN	ESC-50	UrbanSound8K
Choi, Keunwoo, et al. [18]	89.80%	-	69.10%
Multi-Stream Network [37]	-	84.90%	-
Attention-Based CRNN [63]	-	86.10%	-
ES-ResNet [13]	-	91.50%	85.42%
GTZAN [56]	94.50%	-	-
DenseNet (Random)	88.50%	72.50%	76.32%
DenseNet (Pretrained)	91.39%	91.16%	85.14%
DenseNet (Pretrained Ensemble)	90.50%	92.89%	87.42%

Table 3.11: Overall Results Of Models On Three Different Datasets [43]

Work presented by Shawn Hershey et al. [32] used log-Mel spectrogram for the audio representation on the Youtube-100 dataset, which was mentioned in section 3.2.1. The architecture baseline was a fully connected deep neural network. For the baseline experiments, only the 10% most frequent labels of the original 30K were trained and evaluated. Five models were tested which are as follows:

- **Fully Connected Model:** It was used and also acted as a baseline network. With RELU activation function, N layers, and M units for each layer. They swept over N = 2,3,4,5,6 and M = 500, 1000, 2000, 3000, 4000. The best-performing model had N = 3, and M = 1000 units.
- **AlexNet [36]:** The original well-known architecture AlexNet was used. It was designed for a 224 X 224 X 3 input with an initial 11 X 11 convolutional layer with a stride of 4.
- **VGG:** It was used with a few changes to the final layer (3087 units with a sigmoid), as well as the use of batch normalization instead of LRN.
- **Inception V3:** It was used with a modification in the network by removing the first four layers including the max pool, also the average pool size was changed to 10 X 6 to reflect the change in activations.
- **ResNet-50;** The last architecture used was ResNet-50 which was also modified by removing the stride of 2 from the 7 X 7 convolution, the average pool size changed to 6 X 4. The results had shown that ResNet achieves the best performance as shown in table 3.12

Architectures	Steps	Time	AUC	d-prime	mAP
Fully Connected	5M	35h	0.851	1.471	0.058
AlexNet	5M	82h	0.894	1.764	0.115
VGG	5M	184h	0.911	1.909	0.161
Inception V3	137M	184h	0.918	1.969	0.181
ResNet-50	5M	119h	0.916	1.952	0.182
ResNet-50	17M	356h	0.926	2.041	0.212

Table 3.12: Architectures comparison [53]

Chapter 4

Methodology

This research proposes two methods, the initial approach involves transforming the speech data into text form and subsequently utilizing it. The alternative approach is to directly use the speech data without converting it to text. These approaches will be discussed in more detail in this section.

4.1 Dataset

Selecting an appropriate dataset proved to be a difficult task, as obtaining a credible Arabic speech dataset is rare. As a result, locating an Arabic short speech answers dataset was not possible, which necessitated the creation of the dataset from the ground up. The dataset was generated by converting a suitable Arabic text dataset into an audio format.

The Automated-student assessment prize - short answer scoring text dataset by the Hewlett Foundation (ASAS-SAS)[46] was translated by Omar Nael et al. [41] from English to Arabic, this translated version is the one chosen for our conversion. The dataset consists of 10 questions with an average of 1704 answers per question as shown in Table 4.1, and answers are scored on a scale from 0-2 or 0-3 depending on the prompt.

Converting a huge dataset from text to speech manually is almost impossible, and converting a small portion of it could lead to several challenges and problems, that can arise when working with small datasets, especially when dealing with deep learning models since they need a huge amount of data to be well training. A Text-To-Speech API was our only solution.

Google Text-To-Speech API and gTTs API were tested by converting their speech outputs to texts followed by comparing these converted sentences with the original ones. Both converted sentences were evaluated by WER metric which was explained in detail in section 2.7.1. Since Google Text-To-Speech API outperformed gTTs, it was the API used in this research.

Question	Answers Count
1	1672
2	1278
3	1808
4	1657
5	1795
6	1797
7	1799
8	1799
9	1798
10	1640

Table 4.1: Question Count

4.2 Automating Grading of Short Text Answers

4.2.1 Converted Text Dataset Preparation

The text dataset used in this research went through a preparation process, which was divided into four distinct steps. These steps are illustrated in Figure 4.1. Each of these steps will be described in further detail in the sections that follow. The text dataset preparation process is critical to ensure that the data is clean, consistent, and ready.

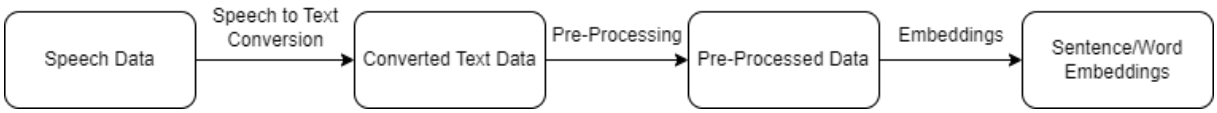


Figure 4.1: Text Data Preparation Pipeline

4.2.1.1 Speech-To-Text Conversion

The most popular Speech-To-Text APIs are Google Speech-To-Text API, and Watson Speech-To-Text API by IBM, both showed great results. However, Google Speech-To-Text API was chosen for this task.

These APIs are not totally accurate, a lot of words will be wrongly converted, and that occurs especially when words' pronunciations are similar to each other, or when the spoken words are not clear. Moreover, regarding numbers, some are converted into words and some are converted into digits, which confuses the classification model. All these

issues surely affect our classification accuracy, however, these problems will be treated in the pre-processing stage. Another untreatable obstacle is that Google Speech-To-Text API does not support code-switching, it converts to a single language. Since our dataset is only in Arabic, this issue was not fatal, it mainly affected question 5 since it has some English terms. Figure 4.2 illustrates the contrast between question 5 original answer, which contains some English terms, and the converted text answer, which is only in Arabic, using WER metric. Which was explained in detail in section 2.7.2

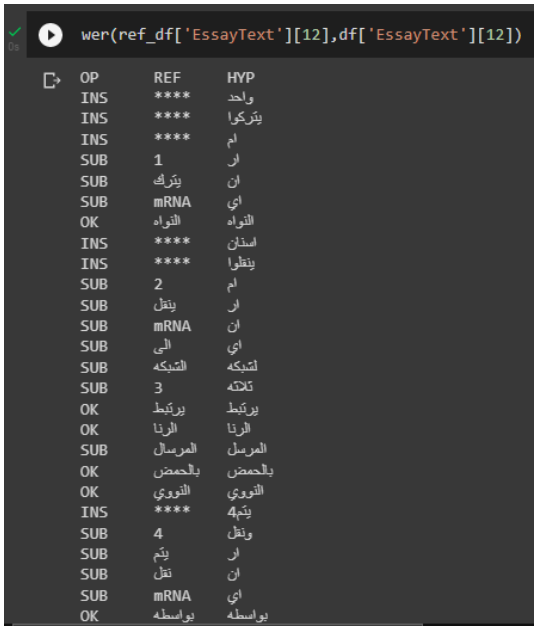


Figure 4.2: Question 5 English Terms Issue

One of the advantages of Google Speech-To-Text API is that it resulted in a very clean dataset after the conversion. Additionally, the dataset sentences were correctly spelled because Google Speech-To-Text API contains a dictionary of Arabic words and their pronunciations. Even if a spoken word is wrongly pronounced or even does not exist at all, Google API will search for the most similar word to it in its dictionary, which is both a pro and a con. A pro since, the dataset will not contain spelling or punctuation mistakes, leading to fewer data pre-processing, on the other hand, a con because the words which are wrongly converted will have a huge effect on the resulting sentence.

4.2.1.2 Converted Text Dataset Pre-Processing

For sure not all Pre-Processing techniques increase classification accuracy, it depends on the dataset, and when it comes to our dataset here are the pre-processing techniques that benefit our dataset.

- **Converting numbers to digits**

  As mentioned before, due to Speech-To-Text conversion, some numbers are converted into words, and some are converted into digits. This was solved by implementing the num2words method that converts all number words to digits.

- **Separating the letter (و)**

  First of all, in the Arabic language the letter و means 'and', one of the Speech-To-Text conversion problems is that the letter و sometimes is stuck to the word after, and sometimes it's not. In the Arabic language, the letter و does not always mean 'and', while some words have the letter as part of its spelling, for example, the word وردة which means a flower. That's why just separating the letter و from the beginning of any word will result in some wrongly punctuated words. A method was created that only separates the letter و which means 'and' from words.

While some NLP pre-processing techniques when applied led to a decrease in our classification accuracy, which are as follows.

- **Removing Stop Words**

  Although reducing the dataset size should cause an increase in performance, but also some datasets like ours are sensitive to stop words removal. Its main issue is turning negative sentences into positive ones. For example, "I told you that she was not happy" after the stop words removal will result in "told happy", the overall meaning of the resulting sentence is positive which is not the case in the original sentence.

- **Lemmatization/Stemming**

  We applied Farsa lemmatizer and ISRi stemmer during pre-processing, however, it resulted in a decline in classification accuracy. Therefore, Lemmatizers nor stemmers were used.

All these pre-processings were applied to our dataset before being fed to our Machine Learning and Deep Learning models. Regarding the dataset being fed to transformers,

only tokenization was applied. Finally, after finishing the pre-processing, the dataset is divided into 80% training data and 20% testing data using the `train_test_split` method, and 20% of the training data were allocated for validation.

4.2.1.3 Embeddings

Embeddings are an essential step in the process of transforming text data into a format that can be understood by models. The research presented two types of embeddings: word embeddings and sentence embeddings. These embeddings are used to capture the meaning of the text data and are used as input for the models.

- **Sentence Embeddings**

  Sentence embeddings were used as a representation of input text for our Deep Learning and Machine Learning models. Three different sentence embeddings were tested, including fastText, Universal Sentence Encoder(USE), and Sentence-BERT [49]. Of these, USE outperformed them.

- **Word Embeddings**

  Word embeddings were used for our transformers models. The sentences were tokenized using Electra tokenizer when being fed to Electra model, while Bert tokenizer was used when feeding the sentence to Bert model. With a token length of 450 for both tokenizers, where each sentence was encoded into an array of IDs and an attention mask array of ones and zeros. Each element in the array of IDs represents a word.

4.2.2 Converted Text Dataset Model Architecture

4.2.2.1 Machine learning

Both Support Vector Machines (SVMs) and Logistic Regression are machine learning algorithms that were used in this research. SVMs are a popular choice for classification tasks and are known for their ability to handle high-dimensional data and separate complex data with non-linear boundaries. In this specific case, the kernel parameter, which determines the type of boundary that separates the data, was set to "linear". This means that the algorithm is using a linear hyperplane to separate the data, rather than a non-linear boundary. The "c" value is a parameter that controls the balance between the margin and the classification error. It varied from 2 to 15 depending on the question, which means that the algorithm is trying different values to find the optimal balance for each question.

On the other hand, Logistic Regression is used to predict the probability of a certain class or event occurring, making it a powerful tool for predicting outcomes based on a set of input variables.

4.2.2.2 Deep learning

Recurrent Neural Networks (RNNs), Long Short-Term Memories (LSTMs) were the models implemented, due to their superior effectiveness with word sequences, as explained in detail in section 2.2.2

Firstly, the RNN model that was used in this research consisted of four layers in total, each layer serving a specific purpose in the model's architecture. The first layer was an RNN layer, which is a type of recurrent neural network that is particularly well-suited for sequential data such as speech or text. This layer had 128 units and was designed to process the input data. The activation function used in this layer was the rectified linear unit (relu) function, which is commonly used in deep learning models for its ability to introduce non-linearity into the model.

Following the first RNN layer, a dropout layer was added to the model. The dropout rate was set at 0.2, which is a common value used in deep learning models. The purpose of the dropout layer is to prevent overfitting, which is when a model becomes too complex and starts to memorize the training data rather than generalizing it to new data. By randomly dropping out some units during training, the model is forced to learn more robust features.

The third layer in the model was also an RNN layer, but with a smaller number of units, specifically 64. This layer was designed to extract more abstract features from the input data. This layer also had the relu function as an activation function.

Finally, the fourth and last layer was a dense layer. This layer was responsible for making the final prediction, and it had either 4 or 3 output units, depending on the number of possible classes in the dataset. The activation function used in this layer was the softmax function, which is commonly used in multi-class classification problems as it allows for the prediction of multiple classes. This function converts the raw output of the dense layer into class probabilities.

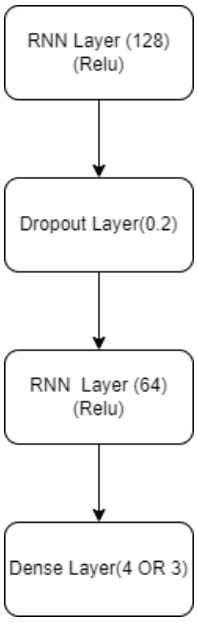


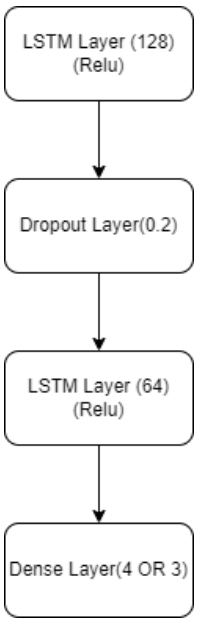
Figure 4.3: RNN Neural Network Diagram

The LSTM model that was used in this research was composed of four layers, each with a specific role in the overall architecture of the model. The first layer was an LSTM layer, which is a type of recurrent neural network that is particularly effective in handling sequential data such as speech or text. This layer had 128 units, which are used to process the input data. The activation function used in this layer was the rectified linear unit (relu) function, which is widely used in deep learning models due to its ability to introduce non-linearity.

After the LSTM layer, a dropout layer was added to the model, with a rate of 0.2. The dropout layer helps to prevent overfitting, which occurs when a model is too complex and starts memorizing the training data instead of generalizing to new data. By randomly dropping out some units during training, the model is forced to learn more robust features.

The third layer of the model was also an LSTM layer, but with a smaller number of units, specifically 64. This layer was designed to extract more abstract features from the input data and also uses relu as an activation function.

The final layer of the model was a dense layer, responsible for making the final prediction and it had either 4 or 3 output units, depending on the number of possible classes in the dataset. The activation function used in this layer was the softmax function, which is commonly used in multi-class classification problems as it allows for the prediction of multiple classes. This function converts the raw output of the dense layer into class probabilities.



In this research, various hyperparameters were used for both RNN and LSTM models to optimize the performance of the model. The first one was the learning rate, which was

Figure 4.4: LSTM Neural Network Diagram

set to 0.01. This value represents the step size used by the optimizer to update the model's weights during training. Since, the most commonly used value is 0.01 it was chosen for deep learning models and works well in many cases. Another cruel hyperparameter is the batch size, which determines the number of samples that are processed in each iteration of the training process. A batch size of 4 was used in this research. This value is relatively small, which can help to prevent overfitting, however, can slow down the training process.

The optimizer used in this research was Adam, which is a popular choice for Deep Learning models. It is known to work well with a wide range of datasets and is particularly well-suited for problems with small datasets and complex architectures.

Finally, the loss function used was sparse categorical cross-entropy. This loss function is commonly used in multi-class classification problems and it is designed to penalize the model when it makes a mistake in its predictions. It compares the predicted class probabilities with the true class labels and calculates the difference between them. The goal of the optimizer is to minimize this loss function during training, so that the model can make predictions that are as accurate as possible.

4.2.2.3 Transformers

BERT and ELECTRA were implemented since they have demonstrated state-of-the-art performance in lots of researches, and since their enormous effect on small datasets. Ara-BERT [11] is a version of the BERT language model that has been specifically trained on Arabic text, while ara-ELECTRA [12] is a variant of the ELECTRA model that has also been trained on Arabic data. These models have shown strong results in various natural language processing tasks, particularly when working with Arabic language data, as explained in detail in section 2.6. Both ara-BERT and ara-ELECTRA were provided by the hugging face library

BERT (Bidirectional Encoder Representations from Transformers) model was used, which is a transformer-based neural network architecture that is commonly used for natural language processing tasks. The BERT model consisted of 3 layers: an embedding layer as the first layer, followed by the BERT transformer layer, and finally a classification layer. The embedding layer is used to convert the input text data into a numerical representation that can be understood by the model. The BERT transformer layer is responsible for extracting the context-based representations of the input text data, which are then passed to the classification layer for the final output.

ELECTRA (Efficiently Learning an Encoder that Classifies Tokens Accurately) model was used, which is a transformer-based neural network architecture similar to BERT that is also commonly used for natural language processing tasks. The ELECTRA model also consisted of 3 layers: an embedding layer as the first layer, followed by the ELECTRA transformer layer, and finally a classification layer. The embedding layer is used to

convert the input text data into a numerical representation that can be understood by the model. The ELECTRA transformer layer is responsible for extracting the context-based representations of the input text data, which are then passed to the classification layer for the final output.

AdamW optimizer was used for training both transformers, which is a variant of the Adam optimizer that is specifically designed for training large neural networks. The learning rate, which determines the step size at which the optimizer makes updates to the model's parameters, was set to 0.004. The number of epochs, which is the number of times the model is trained on the entire dataset, was set to 4 or 5 depending on the question. The batch size, which is the number of samples used in one iteration of the training process, was set to 16 or 32 or 64 also depending on the question.

4.3 Automating Grading of Short Speech Answers

4.3.1 Speech Dataset Preparation

In order to effectively analyze audio clips using machine or deep learning techniques, it is necessary to convert the audio data into a format that can be understood and processed by a computer. Various methods for representing audio data, such as Root Mean Square (RMS), Mel-Frequency Cepstral Coefficients (MFCCs), and Mel-Spectrogram representations were tested to determine which was most effective. These representations were fine-tuned to produce both 1-dimensional arrays and 2-dimensional representation arrays for each audio clip. Ultimately, the 1-dimensional MFCCs representation was found to be the most effective and was used in both machine learning models and deep learning models, as it yielded the best results. Additionally, all of the data pre-processing was performed using the Librosa library.

4.3.2 Speech Dataset Model Architecture

4.3.2.1 Machine learning

Support Vector Machines (SVMs) are a type of supervised learning algorithm that can be used for classification tasks. The SVM algorithm uses a concept called the kernel trick to transform the input data into a higher dimensional space where it becomes possible to find a linear boundary that separates the different classes. The kernel parameter in an SVM determines the type of kernel function used in the transformation. In this context, the kernel parameter is set to "linear", which means that the algorithm is using a linear kernel function. The c-value is a hyperparameter of the SVM algorithm that controls the trade-off between maximizing the margin and minimizing the miss-classification error. In this context, the c-value is varied from 2 to 20 depending on the question.

Logistic Regression was used to fit a classification model. It is a linear method, but it can be extended to handle multiple classes. Logistic Regression is a powerful, widely used, and well-understood method.

4.3.2.2 Deep learning

Recurrent Neural Networks (RNNs) are a type of neural network that are well-suited for processing sequential data, such as audio signals. They are able to capture patterns in the data that may not be evident in individual time steps or frames, which makes them useful for tasks such as audio classification.

In this context, the same RNN model architecture as the text dataset RNN model architecture was used, as mentioned in section 4.2.2.2. The only difference being in the number of epochs, which were changed to 10.

Chapter 5

Results & Discussion

This section of the research is dedicated to providing an in-depth analysis of the results obtained from the two approaches used. The results for each approach will be evaluated and discussed in detail. The purpose of this section is to give a comprehensive understanding of the performance of the methods and the outcomes they produced. The results will be presented, and any relevant observations will be highlighted to give a clear picture of the effectiveness of the methods used.

5.1 Experiment Tools/Metrics

5.1.1 Quadratic Weighted Kappa (QWK)

The QWK was used since it is a chance-adjusted index of agreement. It can be used to quantify the amount of agreement between an algorithm's predictions and trusted labels of the same objects as was explained in detail in section 2.7.2.

$$\kappa = 1 - \frac{\sum_{i=1}^{c} w_i(p_i - pe_i)^2}{\sum_{i=1}^{c} w_i(1 - pe_i)^2} \tag{5.1}$$

5.1.2 Word Error Rate(WER)

When it comes to the similarity between the converted speech-to-text dataset compared to the original dataset, Word Error Rate (WER) metric really shines, since it puts into consideration not only the substituted words but deleted and inserted words too, as explained in detail in section 2.7.1

$$WER = \frac{S+D+I}{N} \tag{5.2}$$

5.1.2.1 Word Error Rate Limitations & Solutions

Finding the WER between the converted sentences and the original ones was a challenging task. As some letters in the Arabic language are somehow similar and if exchanged, would result in a spelling error despite having the same meaning. For example letters أ and ا, letters ة and ه. As a result, WER metric considers certain words as non-similar only due to spelling mistake which is not an important factor for our model. For example the words واحده and واحدة, انسان and إنسان as shown in figure 5.1.

Another limitation is the letter و which means 'and', is sometimes joined with the word following it. This was explained in detail in section 4.2.1.2. For example, the words و كان and وكان were considered nonsimilar as shown in figure 5.2.

```
wer(ref_df['EssayText'][3],df['EssayText'][3])

OP      REF     HYP
OK      يجب     يجب
OK      على     على
OK      الطالب  الطالب
SUB     أن      ان
OK      يذكر    يذكر
SUB     الصخرة  الصخره
SUB     الأفضل  الافضل
OK      وما     وما
OK      هي      هي
DEL     الصخرة  ****
SUB     الأسوأ  الصخره
SUB     في      لسويفل
SUB     الإجراء. جراي
#cor 6
#sub 6
#del 1
#ins 0
{'WER': 0.538,
 'numCor': 6,
 'numSub': 6,
 'numIns': 0,
 'numDel': 1,
 'numCount': 13}
```

Figure 5.1: Example 1 of WER metric problems

Figure 5.2: Example 2 of WER metric problems

In order to accurately measure the correctness of our conversion, the following was done:

- Standardizing certain letters, for example the letter ۅ̈ was converted to ۆ
- Separating the letter و that means 'and
- Removing punctuation
- Removing extra space

All these pre-processing methods are done on both the converted and the original dataset, which is used as our reference to know our conversion accuracy. Without these pre-processing being done we may conclude that our conversion is somehow not similar. While being similar with just minor differences which doesn't affect the meaning nor the classification accuracy, only affecting the WER metric since it's very sensitive.

5.2 Automating Grading of Short Text Answers

5.2.1 Speech-to-Text Conversion Results

Table 5.1 illustrates the similarity between the converted dataset and the original dataset by using the WER (Word Error Rate) metric. As can be seen from the table, question 5 has a high WER score, which is likely due to the fact that it includes a significant number of English terms, as discussed in section 4.2.1.1. Additionally, the study also evaluated the WER metric without applying any pre-processing, which was discussed in Section 5.1.2.1. This evaluation was carried out on two questions, question 1 and question 10, which resulted in WER scores of 0.575 and 0.599 respectively. These scores indicate the level of similarity between the converted dataset and the original dataset. Lower scores reflect higher similarity, while higher scores reflect lower similarity

Question	WER
1	0.338
2	0.401
3	0.357
4	0.368
5	0.541
6	0.326
7	0.350
8	0.315
9	0.357
10	0.382
Mean	0.374

Table 5.1: Word Error Rate (WER) results

5.2.2 Converted Text Dataset Preparation Results

In the research, three different pre-processing techniques were used to process the NLP data, which are Stop Words Removal, Lemmatization using Farasa, and Stemming using ISRI. To evaluate the performance of these techniques, questions 1 and 10 were used as a test sample, as outlined in tables 5.2 and 5.3. The results of the tests showed that applying any of the three NLP pre-processing techniques led to a decline in classification accuracy. This means that the performance of the classifier got worse when these pre-processing techniques were applied

Model	Stop Words Removal	Farasa Lemmatization	ISRI Stemming	Without applying these techniques
LSTM	0.68	0.71	0.29	0.71
RNN	0.69	0.68	0.307	0.729
SVM	0.69	0.67	0.31	0.715
Logistic Regression	0.63	0.66	0.34	0.66

Table 5.2: Question 1 QWK Results For Each Pre-Processing Techniques

Model	Stop Words Removal	Farasa Lemmatization	ISRI Stemming	Without applying these techniques
LSTM	0.65	0.624	0.63	0.646
RNN	0.65	0.646	0.63	0.729
SVM	0.65	0.626	0.645 5	0.66
Logistic Regression	0.64	0.645	0.65	0.65

Table 5.3: Question 10 QWK Results For Each Pre-Processing Techniques

Three types of sentence embedding were tested, including FastText, Universal Sentence Encoder(USE), and Sentence-BERT. Questions 1 and 10 were used for this testing. As shown in tables 5.4 and 5.4, Universal Sentence Encoder(USE) outperformed the other embedding techniques used in this experiment.

Model	Fast Text	Sentence-BERT	Universal Sentence Encoder(USE)
LSTM	0.56	0.31	0.71
RNN	0.67	0.26	0.729
SVM	0.45	0.32	0.715
Logistic Regression	0.48	0.34	0.66

Table 5.4: Question 1 QWK Results For Each Embedding Type

Model	Fast Text	Sentence-BERT	Universal Sentence Encoder(USE)
LSTM	0.55	0.526	0.646
RNN	0.54	0.556	0.729
SVM	0.44	0.55	0.66
Logistic Regression	0.39	0.54	0.65

Table 5.5: Question 10 QWK Results For Each Embedding Type

5.2.3 Converted Text Dataset Models Results

After conducting experiments and selecting the most appropriate Embeddings, Preprocessing techniques, and model architecture, as described in detail in section 4.2. Table 5.6 presents the results of QWK (Quadratic Weighted Kappa) for Machine Learning, Deep Learning, and Transformer models.

Question	SVM	Logistic Regression	LSTM	RNN	BERT	ElECTRA
1	0.71	0.66	0.71	0.73	0.73	0.71
2	0.53	0.41	0.49	0.52	0.50	0.57
3	0.61	0.52	0.58	0.57	0.63	0.65
4	0.64	0.58	0.62	0.60	0.52	0.67
5	0.55	0.39	0.60	0.61	0.59	0.58
6	0.79	0.48	0.80	0.80	0.85	0.77
7	0.43	0.44	0.45	0.47	0.59	0.57
8	0.57	0.545	0.54	0.56	0.62	0.65
9	0.72	0.71	0.69	0.75	0.75	0.76
10	0.66	0.65	0.64	0.73	0.70	0.60
Mean	0.62	0.54	0.61	0.63	0.65	0.65

Table 5.6: QWK Text Dataset Results Among All Implemented Models

5.3 Automating Grading of Short Speech Answers

5.3.1 Audio Representations Results

Three methods of audio representation were compared and evaluated in the study, which are RMS, Mel-Spectrogram, and MFCCs. The effectiveness of these representations was assessed using questions 1 and 10 as outlined in tables 5.7 and 5.8. The results of the evaluation showed that the MFCCs representation had the best performance, surpassing both RMS and Mel-Spectrogram representation in terms of performance

Model	RMS	Mel-Spectrogram	MFCCs
RNN	0.26	0	0.3
SVM	0.21	0.47	0.45
Logistic Regression	0.24	0.43	0.43

Table 5.7: Question 1 QWK results for each Audio Embedding Type

Model	RMS	Mel-Spectrogram	MFCCs
RNN	0.4	0	0.3
SVM	0.4	0.58	0.59
Logistic Regression	0.412	0.578	0.56

Table 5.8: Question 10 QWK results for each Audio Embedding Type

5.3.2 Speech Dataset Models Results

After conducting experiments and selecting the most appropriate Audio representations and model architecture, as described in detail in section 4.3. Tables 5.9 and 5.10 present the results of QWK (Quadratic Weighted Kappa) and accuracy for Machine Learning and Deep Learning models.

Question	SVM	Logistic Regression	RNN
1	0.45	0.46	0.3
2	0.24	0.33	0
3	0.52	0.42	0
4	0.48	0.43	0.23
5	0.55	0.26	0.26
6	0.523	0.28	0
7	0.25	0.30	0.21
8	0.439	0.30	0.32
9	0.68	0.62	0.59
10	0.59	0.53	0.47
Mean	0.472	0.39	0.238

Table 5.9: QWK Speech Dataset Results Among All Implemented Models

Question	SVM	logistic Regression	RNN
1	43%	45%	33%
2	42%	41%	35%
3	66%	61%	54%
4	67%	66%	61%
5	80%	76%	76%
6	84%	83%	82%
7	57%	60%	54%
8	58%	53%	51.6%
9	69%	64%	59%
10	70%	66%	53%
Mean	63.6%	61.5%	56.16

Table 5.10: Accuracy Speech Dataset Results Among All Implemented Models

5.4 Discussion

5.4.1 Automating Grading of Short Text Answers

The average Word Error Rate (WER) of our Speech-To-Text conversion was 0.374, as illustrated in Table 5.1, which is considered acceptable. Taking into account both the challenges with Speech-to-Text conversion discussed in section 4.2.1.1 and the limitations of the WER metric outlined in section 5.1.2.1.

Removing stop words, applying Farasa lemmatization, or using ISRI stemming did not lead to any improvement in classification performance. Lemmatization is the process of reducing words to their base form while stemming is the process of reducing words to their root form. These techniques are often used to reduce the data's dimensionality and group similar or related words together. However, in this case, it seems that these techniques resulted in the loss of important information that the classifier needed to accurately predict the class. Therefore, these techniques were not implemented.

In terms of embeddings, The Universal Sentence Encoder (USE) uses a transformer-based architecture, which allows it to effectively encode the meaning of variable-length sentences. Additionally, the USE is trained on a diverse range of data, which helps it to generalize well to new sentences. On the other hand, FastText is a simple and efficient model that generates embeddings by averaging the embeddings of the words in a sentence. It is not as powerful as the USE when it comes to handling more complex sentence structures. SentenceBERT is a fine-tuned version of BERT, a transformer-based model, which is trained on a large corpus of sentence-level data. SentenceBERT is pre-trained on a smaller corpus than USE and it is not fine-tuned on a diverse range of tasks like USE which is why it may not perform as well as USE.

Overall, the USE's transformer-based architecture, pre-training on a diverse range of data, and fine-tuning on specific tasks, make it more potent than FastText and Sentence-BERT for many natural language processing tasks. As highlighted in section 5.2.2

In the end, it was found that SVM (Support Vector Machine) performed the best among the machine learning models that were tried. Although, our data is linearly separable and the number of features is low. It is possible that in some cases, SVMs may outperform Logistic Regression even when the data is linearly separable and the number of features is not too high. This is because SVMs are less sensitive to the presence of outliers than Logistic Regression. Because of the nature of the optimization problem of SVM, which tries to maximize the margin between the two classes, the model is less affected by individual data points that are far from the decision boundary, which can be the case with outliers. Another reason could be that SVMs can achieve higher accuracy when the number of samples is relatively small. Logistic Regression requires a large number of samples to achieve good performance, while SVMs can achieve good performance even with a smaller number of samples, which is actually our case.

There was no significant difference in performance between the deep learning models implemented. However, RNN (Recurrent Neural Network) performed the best among

them, since RNNs are well-suited for processing sequential data and capturing the temporal dependencies in the data. Both transformer models used performed the best among all the classification models. One of the main reasons for their high performance is that they are pre-trained on a large corpus of text data, which allows them to learn a very rich set of representations for the words and phrases in the text. This pre-training enables BERT and ELECTRA to understand the context in which words are used, which is crucial for many natural language understanding tasks. Another reason for their performance is that they use the transformer architecture, which is based on a self-attention mechanism. This allows the model to selectively attend to different parts of the input, and to weigh the importance of different words, which helps to improve the model's ability to understand the meaning of the text.

SVM, both Deep Learning models, and both Transformers models demonstrated a QWK (Quadratic Weighted Kappa) of very good agreement, as illustrated in section 5.2.3.

5.4.2 Automating Grading of Short Speech Answers

MFCC (Mel-Frequency Cepstral Coefficients) was chosen due to its superior performance compared to other audio representations, as highlighted in section 5.3.1. One of the main reasons MFCCs perform well is that they are designed to capture the characteristics of human speech perception, which is sensitive to the Mel-frequency scale. Mel-frequency is a non-linear scale that more closely matches human perception of sound frequencies. While RMS (Root Mean Square) is a simple energy-based feature that represents the overall energy of the audio signal. It doesn't capture the spectral characteristics of the signal and it's not very robust against additive noise and other distortions. On the other hand, Mel-spectrogram is a feature representation that is based on the Mel-frequency scale, it captures the spectral characteristics of the audio signal, but it doesn't perform as well as MFCCs in many tasks because it doesn't incorporate the human perception characteristics of the Mel-frequency scale.

Eventually, SVM (Support Vector Machine) demonstrated the best performance. Both of the machine learning models outperformed RNN (Recurrent Neural Network) deep learning model, as demonstrated in section 5.3.2. One reason why SVMs may perform better than deep learning models in audio classification is that SVMs are less sensitive to the presence of noise and other distortions in the audio data, which is common in real-world audio signals. SVMs are based on the idea of finding a hyperplane that maximally separates different classes in a high-dimensional feature space, and they are particularly useful when the data is not linearly separable. This makes SVMs a more robust model for audio classification tasks, where the data may be noisy or have other distortions. Another reason why SVMs may perform better than deep learning models in audio classification tasks is that deep learning models typically require a large amount of labeled data to achieve good performance, while SVMs can achieve good performance even with a smaller amount of labeled data

Both of the machine learning models implemented displayed a QWK (Quadratic Weighted Kappa) of good agreement, while the RNN model showed moderate agreement.

5.4.3 Overview

The results indicate that classification models that are designed to handle text data perform significantly better when applied to audio data. This is likely due to a variety of factors. One potential reason is that text data is typically more structured and organized than audio data, making it easier for a model to extract relevant information. Additionally, text data can be easily pre-processed and transformed into numerical representations that can be used as input for models. Audio data, on the other hand, requires more complex pre-processing and feature extraction techniques before it can be used as input for a model.

Chapter 6

Conclusion

The goal of this thesis was to create a model that can automatically assess students' short answer questions without the need for human input. To accomplish this, we employed two methods. The first approach involved converting the speech data into text form and then utilizing it. The second approach was to use the speech data directly without converting it to text form. Both methods were used to achieve the aim of the thesis which is to implement a model to automatically evaluate students' short answer questions without any human intervention.

Regarding the approach that involves utilizing the converted text dataset. Various machine learning and deep learning models were applied to it. These models included Support Vector Machines (SVM), Logistic Regression, Recurrent Neural Networks (RNN), and Long Short-Term Memory (LSTM), as well as different types of transformers such as BERT and ELECTRA. Sentence embeddings were generated using the Universal Sentence Encoder and fed to the machine learning and deep learning models, while BERT and ELECTRA tokenizers were used to feed the transformer models. The results of this approach showed that SVM, RNN, BERT, and ELECTRA all performed exceptionally well, with a Quadratic Weighted Kappa (QWK) of 0.62, 0.63, 0.65, and 0.65 respectively, which is considered state-of-the-art performance in this field.

Regarding the approach that utilizes the speech dataset as it is, without any conversion. To make the speech data machine-readable, Mel-frequency cepstral coefficients (MFCCs) were used to extract audio representations. These representations were then fed to different machine learning models such as Support Vector Machines (SVM) and Logistic Regression, as well as a deep learning model, Recurrent Neural Networks (RNN). The results of this approach showed that the SVM model performed exceptionally well, achieving a Quadratic Weighted Kappa (QWK) of 0.472, which is considered state-of-the-art performance. However, the RNN deep learning model did not perform well, showing poor QWK agreement.

The conclusion was reached that the first approach, which involved utilizing the converted text dataset, performed significantly better than the second approach, which utilized the speech dataset as it is. This is likely because the models used are better suited

to work with text data rather than audio data. The results obtained from the first approach were considered state-of-the-art performance, and it was determined that it is a more effective method for this task.

Chapter 7

Future Work

Future work can be build on gathering a comprehensive and high-quality Arabic speech answers dataset by acquiring this data from educational institutions such as schools or universities. The aim of this initiative is to effectively collect real-world data that will be of great significance to the Arabic Natural Language Processing (NLP) research community. This dataset will provide a valuable resource for researchers working on speech recognition, language modeling, and other related areas of Arabic NLP.

It's also worth exploring other neural network models in addition to transformers when working with audio data. For example, convolutional neural networks (CNNs) have been known to produce strong results when working with audio data, and it could be beneficial to try them. Additionally, transformer models, which are known for their ability to handle sequential data, have been successfully applied in audio-related tasks and had provided promising results.

Appendix

List of Figures

2.1 Machine Learning [8] 4
2.2 Deep Learning Formation [5] 5
2.3 Deep Learning network architectures [51] 6
2.4 Natural Language Processing [14] 8
2.5 Pre-Processing Pipeline [1] 8
2.6 Embeddings Example [21] 9
2.7 WER metric of two sentences 13

3.1 Neural Network Architecture [50] 21

4.1 Text Data Preparation Pipeline 28
4.2 Question 5 English Terms Issue 29
4.3 RNN Neural Network Diagram 33
4.4 LSTM Neural Network Diagram 34

5.1 Example 1 of WER metric problems 38
5.2 Example 2 of WER metric problems 39

List of Tables

2.1 Lexical ambiguity in Arabic [41] 11

3.1 ScAA dataset [10] 17
3.2 Regressor MSE Results using Word-Embeddings [26] 17
3.3 Regressor MSE Results using Sentence-Embeddings [26] 17
3.4 Classier MSE Results using Word-Embeddings [26] 18
3.5 Clasaifier MSE Results using Sentence-Embeddings [26] 18
3.6 Learners Average MSE [38] 19
3.7 Models results [41] 20
3.8 Basic experiments with the English dataset [53] 22
3.9 Basic experiments with the German data set [53] 23
3.10 Examples of labels from the 30K set. [32] 24
3.11 Overall Results Of Models On Three Different Datasets [43] 25
3.12 Architectures comparison [53] 26

4.1 Question Count 28

5.1 Word Error Rate (WER) results 40
5.2 Question 1 QWK Results For Each Pre-Processing Techniques 40
5.3 Question 10 QWK Results For Each Pre-Processing Techniques 41
5.4 Question 1 QWK Results For Each Embedding Type 41
5.5 Question 10 QWK Results For Each Embedding Type 41
5.6 QWK Text Dataset Results Among All Implemented Models 42
5.7 Question 1 QWK results for each Audio Embedding Type 42
5.8 Question 10 QWK results for each Audio Embedding Type 42
5.9 QWK Speech Dataset Results Among All Implemented Models 43
5.10 Accuracy Speech Dataset Results Among All Implemented Models 43

Bibliography

[1] 6 Techniques of Data Preprocessing | Scalable Path®. https://www.scalablepath.com/data-science/data-preprocessing-phase. [Online; accessed 2023-01-09].

[2] Applications of Machine Learning - Javatpoint. https://www.javatpoint.com/applications-of-machine-learning. [Online; accessed 2023-01-05].

[3] Simple Explanation of Quadratic Weighted Kappa. https://www.kaggle.com/code/prashant111/simple-explanation-of-quadratic-weighted-kappa/notebook. [Online; accessed 2023-01-05].

[4] Understanding Word Error Rate (WER) in Automatic Speech Recognition (ASR) | Wingman. https://www.trywingman.com/blog-posts/what-is-word-error-rate-in-automatic-speech-recognition. [Online; accessed 2023-01-05].

[5] What is Machine Learning? The Ultimate Beginner's Guide. https://www.v7labs.com/blog/machine-learning-guide. [Online; accessed 2023-01-09].

[6] What is Supervised Learning? | IBM. https://www.ibm.com/topics/supervised-learning. [Online; accessed 2023-01-05].

[7] What is Unsupervised Learning? | IBM. https://www.ibm.com/topics/unsupervised-learning. [Online; accessed 2023-01-05].

[8] admin . 8 Top Technology Trends for 2019 and the Jobs They'll Create - NabiApp. https://nabiapp.com/8-top-technology-trends-for-2019-and-the-jobs-theyll-create/, apr 4 2019. [Online; accessed 2023-01-18].

[9] Ahmed Abdelali, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. Farasa: A fast and furious segmenter for arabic. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: Demonstrations*, pages 11–16, 2016.

[10] Dolly Agarwal, Somya Gupta, and Nishant Baghel. Scaa: A dataset for automated short answer grading of children's free-text answers in hindi and marathi. In *Proceedings of the 17th International Conference on Natural Language Processing (ICON)*, pages 430–436, 2020.

[11] Wissam Antoun, Fady Baly, and Hazem Hajj. Arabert: Transformer-based model for arabic language understanding. *arXiv preprint arXiv:2003.00104*, 2020.

[12] Wissam Antoun, Fady Baly, and Hazem Hajj. Araelectra: pre-training text discriminators for arabic language understanding. *arXiv preprint arXiv:2012.15516*, 2020.

[13] Yusuf Aytar, Carl Vondrick, and Antonio Torralba. Soundnet: Learning sound representations from unlabeled video. *Advances in neural information processing systems*, 29, 2016.

[14] Mert Barbaros. Natural Language Processing : Not An Easy Task. https://medium.com/barbaros-blog/natural-language-processing-not-an-easy-task-190e6afb3a86, apr 25 2021. [Online; accessed 2023-01-09].

[15] Jacob Benesty, Jingdong Chen, Yiteng Huang, and Israel Cohen. Pearson correlation coefficient. In *Noise reduction in speech processing*, pages 1–4. Springer, 2009.

[16] Margaret A Boden. *Artificial intelligence*. Elsevier, 1996.

[17] Jose Camacho-Collados and Mohammad Taher Pilehvar. From word to sense embeddings: A survey on vector representations of meaning. *Journal of Artificial Intelligence Research*, 63:743–788, 2018.

[18] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. Transfer learning for music classification and regression tasks. *arXiv preprint arXiv:1703.09179*, 2017.

[19] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*, 2020.

[20] Kareem Darwish, Nizar Habash, Mourad Abbas, Hend Al-Khalifa, Huseein T Al-Natsheh, Houda Bouamor, Karim Bouzoubaa, Violetta Cavalli-Sforza, Samhaa R El-Beltagy, Wassim El-Hajj, et al. A panoramic survey of natural language processing in the arab world. *Communications of the ACM*, 64(4):72–81, 2021.

[21] arvindpdmn devbot5S. Word Embedding. https://devopedia.org/word-embedding, sep 29 2019. [Online; accessed 2023-01-09].

[22] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[24] Rian Dolphin. Lstm Networks | A Detailed Explanation. https://towardsdatascience.com/lstm-networks-a-detailed-explanation-8fae6aefc7f9, dec 12 2021. [Online; accessed 2023-01-11].

[25] Myroslava O Dzikovska, Rodney Nielsen, and Chris Brew. Towards effective tutorial feedback for explanation questions: A dataset and baselines. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 200–210, 2012.

[26] Abdelrahman ElNaka, Omar Nael, Hadeel Afifi, and Nada Sharaf. Arascore: Investigating response-based arabic short answer scoring. *Procedia Computer Science*, 189:282–291, 2021.

[27] Theodoros Evgeniou and Massimiliano Pontil. Support vector machines: Theory and applications. In *Advanced Course on Artificial Intelligence*, pages 249–257. Springer, 1999.

[28] Ali Farghaly and Khaled Shaalan. Arabic natural language processing: Challenges and solutions. *ACM Transactions on Asian Language Information Processing (TALIP)*, 8(4):1–22, 2009.

[29] Virendra Gawande, Ms Huda Al Badi, CAS-Ibri UTAS, UTAS Oman, Oman CAS-Ibri, Ms Khaloud Al Makharoumi, and Ms Rashida Cain. Study design and implementation of nlp techniques for automated grading of answers: A conceptual model. 2021.

[30] Wael Hassan Gomaa and Aly Aly Fahmy. Ans2vec: A scoring system for short answers. In *International Conference on Advanced Machine Learning Technologies and Applications*, pages 586–595. Springer, 2019.

[31] Wael Hassan Gomaa and Aly Aly Fahmy. Ans2vec: A scoring system for short answers. In *International Conference on Advanced Machine Learning Technologies and Applications*, pages 586–595. Springer, 2019.

[32] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. Cnn architectures for large-scale audio classification. In *2017 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 131–135. IEEE, 2017.

[33] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[34] Anjali Ganesh Jivani et al. A comparative study of stemming algorithms. *Int. J. Comp. Tech. Appl*, 2(6):1930–1938, 2011.

[35] Mariam Khader, Arafat Awajan, and Akram Alkouz. Textual entailment for arabic language based on lexical and semantic matching. *International Journal of Computing & Information Sciences*, 12(1):67–74, 2016.

[36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[37] Xinyu Li, Venkata Chebiyyam, and Katrin Kirchhoff. Multi-stream network with temporal attention for environmental sound classification. *arXiv preprint arXiv:1901.08608*, 2019.

[38] Nitin Madnani, Anastassia Loukina, and Aoife Cahill. A large scale quantitative exploration of modeling strategies for content scoring. In *Proceedings of the 12th workshop on innovative use of nlp for building educational applications*, pages 457–467, 2017.

[39] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[40] Michael Mohler, Razvan Bunescu, and Rada Mihalcea. Learning to grade short answer questions using semantic similarity measures and dependency graph alignments. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 752–762, 2011.

[41] Omar Nael, Youssef ELmanyalawy, and Nada Sharaf. Arascore: a deep learning-based system for arabic short answer scoring. *Array*, 13:100109, 2022.

[42] Akeem Olowolayemo, Santhy David Nawi, and Teddy Mantoro. Short answer scoring in english grammar using text similarity measurement. In *2018 International Conference on Computing, Engineering, and Design (ICCED)*, pages 131–136. IEEE, 2018.

[43] Kamalesh Palanisamy, Dipika Singhania, and Angela Yao. Rethinking cnn models for audio classification. *arXiv preprint arXiv:2007.11154*, 2020.

[44] Karol J Piczak. Esc: Dataset for environmental sound classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 1015–1018, 2015.

[45] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018.

[46] Automated Student Assessment Prize. The hewlett foundation: Short answer scoring, 2020.

[47] Jaideepsinh K Raulji and Jatinderkumar R Saini. Stop-word removal algorithm and its implementation for sanskrit language. *International Journal of Computer Applications*, 150(2):15–17, 2016.

[48] Sunil Ray. Svm | Support Vector Machine Algorithm in Machine Learning. https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/, sep 12 2017. [Online; accessed 2023-01-05].

[49] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

[50] Brian Riordan, Andrea Horbach, Aoife Cahill, Torsten Zesch, and Chungmin Lee. Investigating neural architectures for short answer scoring. In *Proceedings of the 12th workshop on innovative use of NLP for building educational applications*, pages 159–168, 2017.

[51] Stacey Ronaghan. Deep Learning: Overview of Neurons and Activation Functions. https://srnghn.medium.com/deep-learning-overview-of-neurons-and-activation-functions-1d98286cf1e4, jul 26 2018. [Online; accessed 2023-01-09].

[52] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello. A dataset and taxonomy for urban sound research. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 1041–1044, 2014.

[53] Jörg Sawatzki, Tim Schlippe, and Marian Benner-Wickner. Deep learning techniques for automatic short answer grading: Predicting scores for english and german answers. In *Artificial Intelligence in Education: Emerging Technologies, Models and Applications*, pages 65–75. Springer, 2022.

[54] Cullen Schaffer. Selecting a classification method by cross-validation. *Machine learning*, 13(1):135–143, 1993.

[55] Mark D Shermis. Contrasting state-of-the-art in the machine scoring of short-form constructed responses. *Educational Assessment*, 20(1):46–65, 2015.

[56] Bob L Sturm. The gtzan dataset: Its contents, its faults, their effects on evaluation, and its future use. *arXiv preprint arXiv:1306.1461*, 2013.

[57] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[58] Kaveh Taghipour and Hwee Tou Ng. A neural approach to automated essay scoring. In *Proceedings of the 2016 conference on empirical methods in natural language processing*, pages 1882–1891, 2016.

[59] Jalaj Thanaki. *Python natural language processing*. Packt Publishing Ltd, 2017.

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[61] Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, et al. Multilingual universal sentence encoder for semantic retrieval. *arXiv preprint arXiv:1907.04307*, 2019.

[62] Torsten Zesch, Michael Heilman, and Aoife Cahill. Reducing annotation efforts in supervised short answer scoring. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 124–132, 2015.

[63] Zhichao Zhang, Shugong Xu, Shunqing Zhang, Tianhao Qiao, and Shan Cao. Attention based convolutional recurrent neural network for environmental sound classification. *Neurocomputing*, 453:896–903, 2021.

[64] Zhi-Hua Zhou. *Machine learning*. Springer Nature, 2021.