SEMANTIC SIMILARITY OF ARABIC QUESTIONS USING MACHINE LEARNING SOLUTIONS

By

Shorouq Mahmoud Al-AlAwawdeh

Supervisor
Dr. Gheith Ali Abandah, Prof.

This Thesis was submitted in Partial Fulfillment of the Requirements for the master's degree of Science in Computer Engineering and Networks

School of Graduate Studies
The University of Jordan

August 23, 2021

Committee Decision

This Thesis (Semantic Similarity of Arabic Questions Using Machine Learning Solutions)

was successfully defended and approved on ----------

Examination Committee	**Signature**
Dr. Gheith A. Abandah, (Supervisor) Professor of Computer Science and Engineering	_____
Dr. Ramzi Saifan, (Member) Associate Professor of Computer Engineering	_____
Dr. Fahed Jubair, (Member) Assistant Professor of Computer Engineering	_____
Dr. Ali Al-Haj, (Member) Professor of Computer Engineering (Princess Sumaya University for Technology)	_____

Dedication

To my beloved Father's soul and my mother for their prayers, endless love, and support

To my faithful brothers; Moner, Mohnd, and Shadi

To my close friends who supported me all the time

To my supervisor, Prof. Gheith A. Abandah, who spared nothing towards my success

Acknowledgement

Initially, I would like to thank GOD for allowing me to complete my educational career to this moment.

I would like to thank my supervisor Prof. Gheith A. Abandah, for allowing me to do research work under his advice and for his suggestions, help, support, collaboration, friendship, and patience during my work on this thesis. I highly appreciate his efforts. He is an excellent mentor, and he taught me how to do research correctly. My gratitude is beyond words.

Finally, I would like to thank my family, Mahmoud Al-Awawdeh and Jamilah Al-Tamimi, for their constant love, support, and encouragement. Without them, none of this would have been possible. I cannot thank them enough, and they will always have my most profound respect and love.

List of Contents

Committee Decision ii

Dedication iii

Acknowledgement iv

List of Contents v

List of Figures vii

List of Tables ix

List of Abbreviations xi

Abstract xiii

Chapter I: Introduction 1

1.1 Semantic Questions Similarity 1

1.2 Semantic Similarity Types 2

1.3 Importance of Semantic Questions Similarity of Arabic 3

1.4 Research Objectives 5

1.5 Contributions 5

1.6 Thesis Outline 6

Chapter II: Literature Review 7

2.1 General Solutions 7

2.2 Arabic Solutions 10

2.2.1 Supervised Learning to Measure the Semantic Similarity 10

2.2.2 Paraphrase Identification and Semantic Text Similarity Analysis in Arabic News Tweets Using Lexical, Syntactic, and Semantic Features 11

2.2.3 Semantic Similarity of Arabic Sentences with Word Embeddings 13

2.2.4 Arabic Natural Language Processing and Machine Learning-Based Systems 13

2.2.5 Nsurl-2019 Semantic Question Similarity in Arabic Task 14

Chapter III: Technology Used 18

3.1 Bidirectional Encoder Representations from Transformers (BERT) Model 19

3.2 Additional Details for BERT 20

3.2.1 BERT Pre-Training Phase 20

3.2.2 Fine-tuning Phase 22

3.3 Architecture of BERT Bidirectional Self-Attention Transformer 23

3.3.1 Encoder and Decoder Stacks Encoder 23

3.3.2 Attention 26

3.3.3 Multi-Head Attention 26

3.3.4 Position-wise Feed-forward Networks 27

3.3.5 Embeddings and SoftMax 27

3.3.6 Positional Encoding 28

3.4 Ensemble Learning Mechanism 28
3.5 The Proposed Model 29
Chapter IV: Dataset Preparation 31
4.1 Data Analysis 31
4.1.1 Mawdoo3 Dataset 31
4.2 Preparing the Dataset 38
4.2.1 Preprocessing 38
4.2.2 Data Augmentation 41
Chapter V: Experiments and Results 45
5.1 Research Environment 45
5.2 Pre-training the AraBERT Model 46
5.3 Training Time 47
5.4 Results 49
5.4.1 Training Mawsoo3 Data Set 49
5.4.2 Training Mawdoo3 Dataset after Pre-training Model on the Back-Translation Dataset 51
5.4.3 Training on Mawdoo3 Dataset after Pre-training the Model on Translated Quora Question Pairs Dataset 53
5.4.4 Ensemble 55
5.5 Testing the Model 56
5.6 Discussion 57
Chapter VI: Conclusions and Future Work 60
References 62
ملخـــص 65

List of Figures

NUMBER	FIGURE CAPTION	PAGE
1	The model of measuring semantic similarity **(Wali *et al.*, 2015)**	11
2	The Overall methodology **(AL-Smadi *et al.*, 2017)**	12
3	The ANLP Phases **(Larabi *et al.*, 2019)**	14
4	The CNN Architecture Model **(Al-Theiabat and Al-Sadi, 2019)**	15
5	The Proposed Model of the Tha3roon Team **(Fadel *et al.*, 2019)**	16
6	Overall pre-training and fine-tuning procedures for BERT **(Devlin *et .al*, 2019)**	20
7	BERT input representation **(Devlin *et al.*, 2019)**	20
8	The BERT model used for classifying similarity sentences	23
9	The Transformer Model Architecture **(Vaswani *et al.*, 2017)**	25
10	. Multi-Head Attention **Structure (Vaswani *et al.*, 2017)**	27
11	Hard Voting Ensemble Method	29
12	The Model Archeticture	30
13	The full description of Mawdoo3 dataset	32
14	Plot of the count of the unique and repeated questions	33
15	Log-histogram of Question Appearance Counts	34
16	Sample of Quora Question Pairs **(Quora page, 2021)**	35
17	The Percentage of Questions that are Similar and Different	35
18	The data frame information for Quora before dropping the first 3 columns	36
19	The data frame information for Quora after dropping the first 3 columns	36
20	The command for translating Quora dataset from English to Arabic	37

21	The command to load translation from Pickle file	37
22	The command used for tokenization of Arabic questions	40
23	The output of the tokenization command for Arabic questions	40
24	The decoded output	40
25	The flowchart of enlarging Mawdoo3 dataset using rules: Transitive, Symmetric, and Reflexive	42
26	The back-translation process	43
27	The command to load back translation – Pickle file	43
28	The training loss for all pre-training models	43
29	Comparison of the total training time between AraBERT models in minutes	47
30	Comparison of the average total training time between augmented datasets for five **runs**	48
31	The training and validation loss of the Mawdoo3 dataset	49
32	The accuracy of all data augmentation types on five runs	50
33	The accuracy of Mawdoo3 dataset after pre-training the model on the back-translation **dataset**	51
34	The training loss of Mawdoo3 dataset after pre-training the model on the back-translation **dataset**	53
35	The training and validation loss for translated Quora question pairs dataset	53
36	The training and validation accuracy for translated Quora question pairs dataset	55
37	The final results of the accuarcy for all improvment stages	55
38	Comparison of error rates among all researchers who worked on this **task**	58

List of Tables

NUMBER	TABLE CAPTION	PAGE
1	Factoid Question Types	2
2	Similarity Examples	3
3	The ANLP Preprocessing Operations (Larabi *et al.*, 2019)	14
4	Summary of Best Result for the Participating Teams in NSURL	17
5	Examples for the Occurrence Numbers for Each Question	33
6	Comparison between Quora and Mawdoo3 Datasets	38
7	Sample of the translation process for Quora dataset	38
8	The punctuation marks	39
9	The total number of data with the **three augmentation types**	42
10	The back-translation examples for the Mawdoo3 dataset	44
11	Accuracy for pre-training AraBERT models	46
12	Accuracy of Mawdoo3 dataset	50
13	The accuracy of the pre-training model on the back-translation dataset	51
14	The accuracy of the Mawdoo3 dataset and all data augmentation types after pre-training the model on the back-translation **dataset**	52
15	The accuracy of the Mawdoo3 dataset and all data augmentation types after pre-training the model on the translated Quora **dataset**	54
16	AraBERT ensemble result with five experiments	56
17	Examples for making predictions with the model	57

18	Comparison with the results of trams participated in solving the task	59

List of Abbreviations

ANLP	Arabic Natural Language Processing
AraBERT	Bidirectional Encoder Representations from Transformer for Arabic
BERT	Bidirectional Encoder Representations from Transformer
BLEU	Bilingual Evaluation Understudy
CBOW	Continuous Bag-of-Word
CNN	Convolution Neural Network
ELMo	Embeddings from Language Models
GRU	Gated Recurrent Unit
IDF	Inverse Document Frequency
IR	Information Retrieval
LMF	Lexical Markup Framework
LSTM	Long Short-Term Memory
LSVM	Lagrangian Support Vector Machine
MaLSTM	Manhattan Long Short-Term Memory
MSE	Mean Square Error
NE	Named Entity
NER	Named Entity Recognition
NLP	Natural Language Processing
NSURL	Natural Language Processing Solutions for Under Resourced Languages
PI	Paraphrase Identification
POS	Part of Speech
PV-DBOW	Paragraph Vectors with Distributed Bag Of Words
PV-DM	Paragraph Vectors with Distributed Memory
QA	Question Answering

RNN	Recurrent Neural Network
SA	Sentiment Analysis
SICK	Sentences Involving Compositional Knowledge
STS	Semantic Text Similarity
SVR	Support Vector Regression
TreeLSTM	Tree Long Short-Term Memory

Semantic Similarity of Arabic Questions Using Machine Learning Solutions

By
Shorouq Mahmoud Al-Awawdeh

Supervisor
Dr. Gheith Ali Abandah, Prof

Abstract

The Arabic language is a native language for more than 400 million speakers worldwide. The Arabic language is rich because it can communicate phrase meanings, and one word may be described in a variety of ways of meaning, definition, and articulation.

Currently, more applications are supporting English compared to the Arabic language. So, the attention on the Arabic language should be increased because it is widely used on social media and search engines such as Facebook and Google engine. Many people use search engines to search about certain questions in a different context to find a solution.

Several methods have been proposed to solve the semantic similarity problem between two questions or sentences. The recent advances in deep learning and the availability of pre-trained English and Arabic models allowed the training of different kinds of datasets using fine tuning procedure. The most notable of these models is the pre-trained deep bidirectional transformers for language understanding (BERT). Such approaches raise the accuracies with a low error rate and are quickly obtaining new state-of-art results on several natural language processing (NLP) problems.

This thesis aims to find a model that can find the semantic question similarity between two Arabic questions with a low error rate. The dataset that we have is not enough to train a pre-training model efficiently. So, we adopt approaches to enlarge the training dataset such as data augmentation (transitive, symmetric, and reflexive).

The original dataset was used in a competition organized in the Workshop on Natural Language Processing Solutions for Under-Resourced Languages (NSURL) in 2019. So, in this project, we try to improve the accuracy and compare our results with other teams who have participated in this competition.

AraBERT v0.2 archives state-of-the-art results. We use the accuracy to evaluate the results after pre-training it on the augmented datasets. Firstly, we pre-trained the model on the Quora dataset, and the accuracy after training with all data augmentation types is 96.39%. Then, with the back-translation dataset, the accuracy is improved to 96.74%. Finally, the accuracy is further improved by an ensemble to 96.88%.

Chapter I: Introduction

This chapter introduces semantic questions similarity, semantic similarity types, the importance of semantic questions similarity of Arabic, our research objectives, and contributions, and thesis outline.

1.1 Semantic Questions Similarity

Many people use the internet websites like Google, Yahoo, and Bing to search for certain information or to ask about some fields such as education, industrial, religion, *etc*. As they allow you to reach the information on multiple electronic platforms. Users note that many questions/sentences have the same meaning, but they have different syntax.

Arabic Questions Answering systems are gaining high importance because the utilizing of Arabic content on the internet and the request for data increase continuously (Ezzeldin and Shaheen, 2012). Also, the methods related to information retrieval, natural language processing, and so on, are becoming increasingly important and are being improved to help users manage and process data (Wali *et al.*, 2015).

Semantic questions deal with many types of questions. Factoid questions are the first type that is concerned with questions that ask mainly about something like questions using the words as shown in Table 1. The second type is the questions that ask about the definition of a term or concept like questions using the words:" Why / لماذا" or "How/ كيف" (Ezzeldin and Shaheen, 2012). The third type is the questions that can be answered with a simple "yes" or "no" are logically called Yes/No questions. This type does not have a special syntax to express it (Ezzeldin *et al.*, 2012).

Table 1. Factoid Question Types

#	Question Word	Meaning	Function
1	When	متى	Date / Time
2	Where	أين	Place
3	How much/many	كم	Number / Quantity
4	Who	مَن هو/ هي	Person name
5	What	ما / ماذا	Organization / Event
6	Whose	لِمن	Ownership

1.2 Semantic Similarity Types

Finding word similarity is the main part of text similarity which is used as a major stage for sentences, questions, paragraphs, and document similarities (Wali *et al.*, 2015). Words similarity is divided into two ways: lexically, and semantically. A lexical way is when words have the same characters sequence. Also, it introduces a string-based algorithm that measures similarity (distance) using different measures like 1) Cosine similarity approach is used in lots of applications to find the similarity between two texts that are represented in the vector form *i.e.* each word in text has a certain dimension in the Euclidean space, 2) N-gram similarity technique determines the similarity based on the distance between each character in two strings, and 3) Jaccard coefficient similarity is determined based on the number of intersection elements divided by the number of union elements.

A semantic way is when words have the same thing, are opposite of each other, are used in the same context and one is a kind of another. It is divided into two types of measures based on a corpus which has a huge amount of written and spoken texts and based on knowledge (similarity and relatedness) that identifies the similarity degree

between words using data derived from a semantic network like WordNet (Gomaa and Fahmy, 2013).

The semantic similarity in Arabic sentences uses a word embedding-based system to determine semantic similarity (Kenter, *et al.*, 2016). The goal of this process is to allow predicting whether two Arabic questions are similar or not (Mawdoo3 page, 2019). Table 2 shows examples of different types of similar or dissimilar questions (NSURL, 2019).

Table 2. Similarity Examples

Semantically Similar	Question 1	Question 2
Yes	ما معنى الهجرة السرية؟	ما هي الهجرة السرية؟
Yes	ما هي العوامل التي تؤثر في انتماء الشخص للوطن؟	كيف ينمو شعور الانتماء للوطن؟
No	إلى ماذا يعود سبب تسمية الأردن؟	ماذا تسمى الأردن رسمياً؟
No	بم يتميز تطبيق الفايبر؟	كيف أعمل تسجيل خروج من الفايبر؟

1.3 Importance of Semantic Questions Similarity of Arabic

Arabic is one of the world's major languages. It is the fifth most widely spoken language in the world and is the second in terms of the number of speakers with over 250 million Arabic speakers (Wali *et al.*, 2015).

Arabic question answering systems are gaining great importance due to the increasing amounts of Arabic content on the Internet and the increasing demand for information.

Arabic question answering systems are considered one of the Arabic natural languages processing and information retrieval applications in which researchers

develop Arabic tools due to the increasing amount of Arabic content on the internet and the increasing demand for information.

Firstly, a question answering system must understand what the Arabic question is asking about and what the answer is. Secondly, predict if there is a semantic similarity between two Arabic questions correctly.

However, Arabic speakers can know semantic question/sentence similarity based on the context and their understanding of the question type and the proper answer. Mawdoo3 website supporting Arabic content needs semantic question similarity in Arabic to easily access the correct answer regardless of the format of the question. Moreover, machine-readable dictionaries and Arabic corpora need to improve and increase the amount of semantic Arabic words to find semantic similarity easily (Hammo *et al.*, 2002).

The Arabic language has complexities that can be considered more complex than English and Latin-based languages, which cannot reach high performance, such as:

1) The vocabulary of Arabic can have different meanings.

2) The Arabic language uses certain inflections (prefix, suffix).

3) Some letters have different forms that depended on their locations in the word.

Another reason for semantic question similarity is the question answering when using Chatbots *via* messaging applications in different fields such as customer support, education, news, health, and food; many users would ask a question with the same meaning/answer by using different questions format.

For all the reasons that we discussed above, we identified the importance of finding semantic questions like the Arabic language.

1.4 Research Objectives

The objectives and contributions of this project are summarized as follows:

1. Investigating the current solutions and implementing and testing alternative solutions for finding semantic question similarity for the Arabic language.
2. Building an accurate system to analyze the Arabic questions and identify the semantic similarity between two Arabic questions
3. Improving semantic similarity accuracy by using the ensemble learning model.
4. Selecting an efficient solution for finding semantic similarity of Arabic questions.

1.5 Contributions

We will use AraBERT v1 and v2 to select the best pre-training model to achieve a better result. First, pre-train the model on several datasets and note the change in the results using data augmentations. Then use the ensemble to improve the accuracy.

Finally, we think this thesis is an essential addition to the field of semantic questions similarity of Arabic language and provides a promising approach for computer support of the Arabic language.

1.6 Thesis Outline

The rest of this thesis is structured as follows. Chapter 2 represents a literature review of previous related approaches developed to solve the semantic Questions similarity problem for the Arabic language. Chapter 3 explains the technologies used in our work, including the AraBERT pre-training model utilized to solve semantic similarity problems and improve the performance using new technology. Chapter 4 describes our work's methodology, including our workload, data processing, and data augmentation. Chapter 5 describes the experiments that we did in this work, the results of our proposed system, and a comparison with state-of-the-art systems made. Finally, Chapter 6 shows conclusions and suggestions for future work.

Chapter II: Literature Review

There is much research about semantic questions/ sentences similarity for different languages such as English, Chines, Arabic, *etc.*, used several different techniques to get the best accuracy. The following sections survey the general solutions and Arabic applications used in this field.

2.1 General Solutions

Bromley *et al.* (1993) introduced the Siamese network, a non-linear metric learning architecture with similar information to solve various problems such as verifying signatures as image match. The Siamese network is designed as two twin networks, meaning that all weights and biases are linked, which means that both networks are symmetrical. Therefore, it learns representations with invariance and selectivity requirements through available information about similarity and dissimilarity between pairs of objects. In NLP applications, Siamese networks with convolutional layers have matched sentences (Hu *et al.*, 2014).

Mikolov *et al.* (2014) have demonstrated the effectiveness of word representations for NLP tasks such as word representation, named entity recognition, syntactic analysis, and machine translation. They proposed Paragraph Vector, an unsupervised framework that trains continuously distributed vector representations for variable length text. After training, they can use the D paragraph vectors as functions to feed directly into conventional machine learning techniques, as paragraph vectors in their experiments have two combination vectors: 1) PVDM and 2) PVDBOW to perform well on many tasks. Paragraph Vector performed better than other approaches in the IMDB dataset, with a lower error rate of 7.42%. Paragraph Vector performs well, offering a relative

improvement of 32% in terms of error rate. The paragraph vector method outperforms the bag of words and bigrams. This method helps to grasp the semantics of the input text.

Zhao *et al.* (2014) proposed a technique for learning vector space representations using latent semantic analysis and numerous functions. Another high throughput system, Bjerva *et al.* (2014), used the simultaneity of formal semantics and logical inference with features taken from the popular neural language model word2vec by Mikolov *et al.* (2013) its word vector used as input for a model.

Cho *et al.* (2014) proposed a new neural network architecture called Encoder-Decoder that can learn to map one sequence of variable length onto another sequence. This architecture serves two purposes: scoring a pair of sequences (based on likelihood) to improve overall translation performance for BLEU scores and generating a target sequence. The proposed architecture produces good results. However, one approach that has not been explored here is to replace all or part of the phrase by allowing the RNN Encoder-Decoder to suggest target phrases.

Kiros *et al.* (2015) proposed an unsupervised sentence coder framework called the Skip Thoughts model, which consists of three parts: encoder, decoder, and Objective, which extends word2vec's Skipgram approach from word to sentence level. This model uses an RNN encoder with GRU triggers and an RNN decoder with conditional GRU (which works just as well as LSTM) that tries to reconstruct the sentences immediately preceding and following. To tailor their approach to the sentence similarity problem, they first pass the sentence through the RNN encoder to extract a skip-thought vector for each sentence. Next, a linear classifier is trained on the SICK dataset using characteristics derived from the differences and products between the skip-thought

vectors for the sentence pairs occurring in each training example. Kiros *et al.* have considered eight problems: the semantic relationship using Pearson's r, Spearman's ρ, and the mean square error evaluation metric. In some examples, this approach does not recognize slight differences that change the meaning of a sentence.

Tai *et al.* (2015) suggested TreeLSTM. Each sentence is first converted into a parse tree with a separately trained parser, and the TreeLSTM calculates its hidden state in a given tree node from the corresponding word, and the hidden states of all Network disseminate the necessary information more efficiently than a sequentially restricted architecture. They are now produced by TreeLSTM instead of skipping thoughts. In the semantic similarity results between two sentences using dependency, TreeLSTM scores better for all measures where Pearson's r was 0.8676 and MSE was 0.2532.

Mueller and Thyagarajan (2016) proposed a new model known as the Manhattan LSTM model that includes LSTM networks which every kind of network has sentence pair. The LSTM learns a mapping from the distance of variable-length sequences of din-dimensional vectors ($d_{\text{in}} = 300$). This LSTM works as an encoder and a sentence is exceeded to it which updates its hidden states at every series index the use of sure equations. The output from every LSTM is entered as input to use the easy similarity function. Then the MaLSTM can predict semantic relatedness scoring for a given pair of sentences and that they train the Siamese network. They used *SemEval* to evaluate the predicted similarities in opposition to the given human-annotated similarities on 3 metrics: Pearson correlation, Spearman correlation, and MSE. The results of MaLSTM based on those metrics were 0.8822, 0.8345, and 0.2286, respectively.

2.2 Arabic Solutions

This subsection discusses some recent studies about finding semantic similarities of Arabic sentences and the methodology of each one.

2.2.1 Supervised Learning to Measure the Semantic Similarity

Wali *et al.* (2015) proposed a method to measure semantic similarities between Arabic sentences. This method consists of two phases: the learning phase and the test phase. The learning phase includes using a training corpus, a set of extracted features from the learning corpus, and an LMF standardized Arabic dictionary to train the learning algorithm. Also, there is pre-processing in this phase to have an annotated corpus and a training process to get a hyperplane equation *via* the training algorithm.

The testing phase implements the learning phase results in a vector format. The extraction vectors and the hyper equation generated in the learning phase are provided as input to the classification module. Each vector will have a score according to the coefficients of three features (lexical, semantic, and syntactic-semantic).

At the end of this phase, they obtain a similar decision between the sentence pair by collecting a set of 1380 sentences taken from dictionaries then choosing five classifiers that cover different algorithms, as shown in Figure 1. The evaluation results of classifiers were roughly between 96 - 98.7%. But in the longer sentence (>10 words), there are complex calculations, which reduce the system's performance.

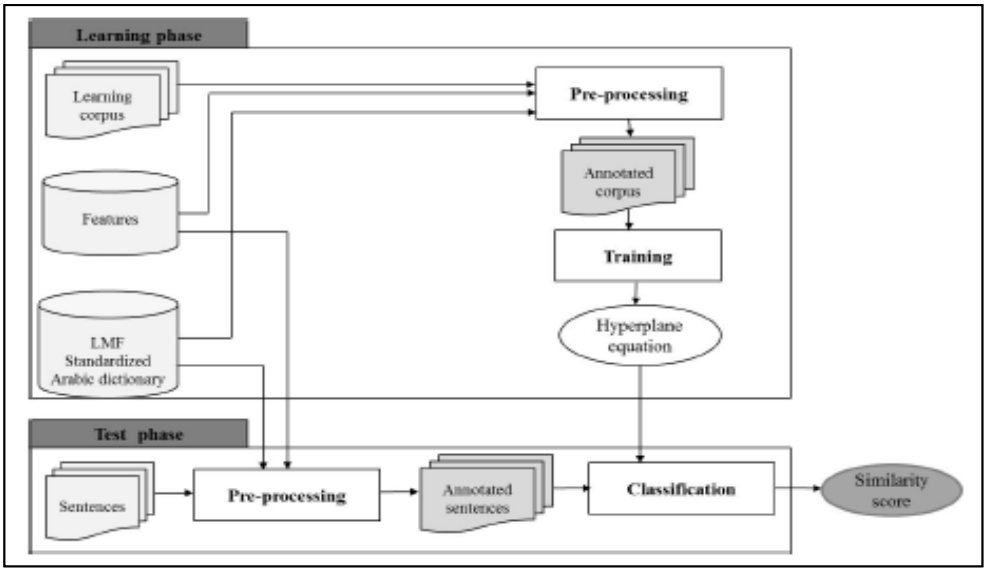


Figure 1. The model of measuring semantic similarity (Wali *et al.*, 2015)

2.2.2 Paraphrase Identification and Semantic Text Similarity Analysis in Arabic News Tweets Using Lexical, Syntactic, and Semantic Features

AL-Smadi *et al.* (2017) proposed a state-of-the-art approach for paraphrase identification and semantic text similarity analysis in Arabic news tweets. Paraphrase Identification (PI) is concerned with recognizing if two texts have the same meaning at several textual levels (document level, paragraph level, *etc*.). The PI will decide the binary value (True/False) if the two tweets have the same or different meanings. The STS is concerned with the degree of that similarity.

The STS will create a (0-5) scale to determine the degree of semantic similarity between two texts. Figure 2 shows the general process for preparing the dataset and finding the similarities. This approach improved both tasks by about 6% in PI ($F_1 = 0.872$) and about 9% in STS (P = 0.912) Therefore, adding the topic modeling and alignment functions to the SVR baseline developed with the lexical overlap features

resulted in better results in both tasks. It has proven its importance in the field of semantic similarity analysis and paraphrases recognition.

The word alignment feature involves the two-fold process of first calculating the similarity based on the minimum editing distance (*i.e.*, using the Levenshtein distance between words) and then calculating the cost of aligning those words based on their minimum editing distance.

The results indicated that using the first part itself was more meaningful than the second part (*i.e.*, alignment) for improving PI and STS analysis. Their results for PI were ($F_1 = 0.857$), ($F_1 = 0.842$) and for STS (Pearson = 0.905), (Pearson = 0.893).

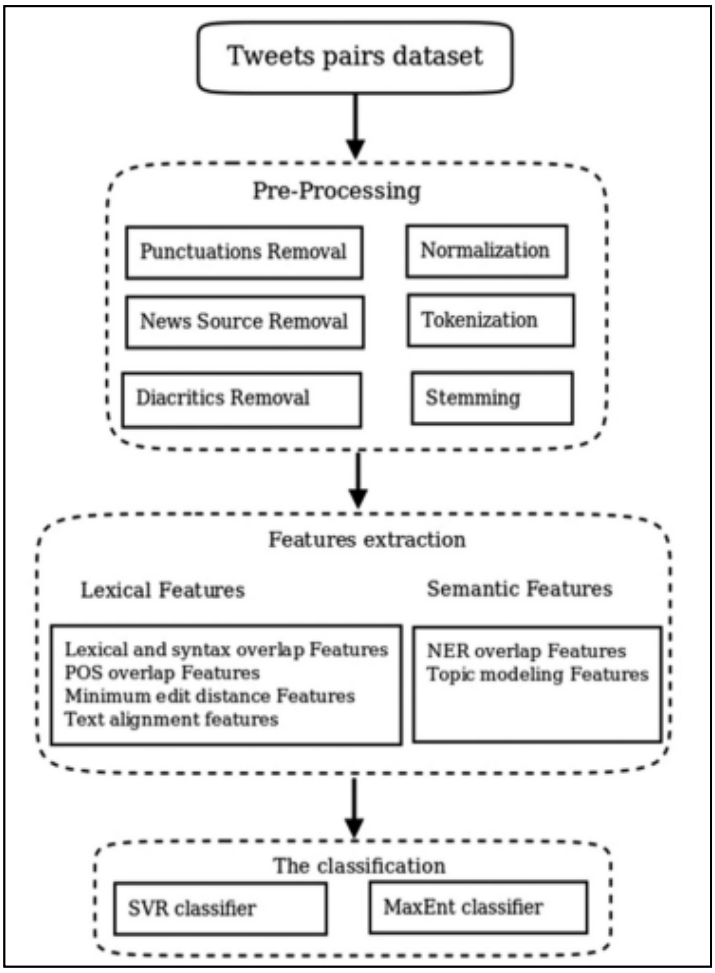


Figure 2. Overall methodology (AL-Smadi *et al.*, 2017)

2.2.3 Semantic Similarity of Arabic Sentences with Word Embeddings

Billah *et al.* (2017) proposed a model that computes the semantic similarity between two Arabic sentences using CBOW representation for Arabic. To train the Arabic CBOW requires choosing some parameters affecting the result vector-like vector size= 300 and window counting the number of a word for each pivot word = 5. There are three methods to measure the semantic similarity:

1. No weight that is a simple way done by summing S_1 and S_2vectors.
2. IDF weighting that serves as a measure of how much information the word provides.
3. POS weight that uses POS to estimate the POS of each word in a sentence *i.e*., for each type of tag there is a weight like a verb, noun, adjective, and preposition.

The correlation rate for each method reached 77.33%, 78%, and 79.69%, respectively. But the similarity score is not good enough when two sentences share the same word with a different meaning.

2.2.4 Arabic Natural Language Processing and Machine Learning-Based Systems

Larabi *et al.* (2019) summarized Arabic natural language processing (ANLP) that consists of developing techniques and tools to use and analyze the Arabic language in both written and spoken contexts. There are many phases of ANLP applications, as shown in Figure 3. Also, they discussed the preprocessing operations, as shown in Table 3.

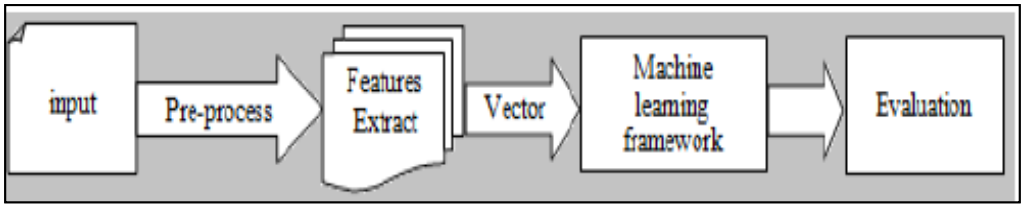


Figure 3. The ANLP Phases (Larabi *et al.*, 2019)

Table 3. The ANLP Preprocessing Operations (Larabi *et al.*, 2019)

Operations	Examples (before applying the operation)	Examples (after applying the operation)
Data Cleaning	ذَهبَ أحمدُ إلى المدرسةِ Ahmed went to school	ذهب أحمد المدرسة
Normalization	ذهب أحمد إلى المدرسة	ذهب أحمد إلى المدرسة
Tokenization	ذهب أحمد إلى المدرسة	(ذهب، أحمد، إلى، المدرسة)
Stemming	(المكتبة، الكاتب، الكتاب) The library, the writer, the book	كتب Wrote

They presented the most frequently used classifiers in ANLP applications that provide excellent results like 1) Support Vector Machine (SVM) for text classification and NLP-problem in different languages, 2) Naïve Bayes classification for text categories that assigns documents to associated categories, and 3) Decision Trees for many NLP-related classifications and Arabic Named Entity Recognition applications.

2.2.5 Nsurl-2019 Semantic Question Similarity in Arabic Task

Al-Theiabat and Al-Sadi (2019) used four different deep learning approaches to find the semantic similarity such as RNN, CNN, Multi-head Network Model, and BERT models. In CNN, for each pair of the question, there are 3 layers followed by activation and max pooling, and the output of each question is a feature representation that is used to find the similarity label by Cosine similarity as shown in Figure 4. In RNN, the input sequence of the question pair is encoded by the dictionary, and they used bi-directional

GRU to get the similarity label as output. In Multi-head Network Model, the benefit of this model is to learn on different locations of encoded-words, so it consists of a stacked encoder-decoder with 8 heads. In BERT, it is one of the pre-training language models that depend on the multi-head self-attention (transformer) technique to provide state-of-art accuracy on lots of tasks. The model that has the best F1 score was BERT with 96.05%.

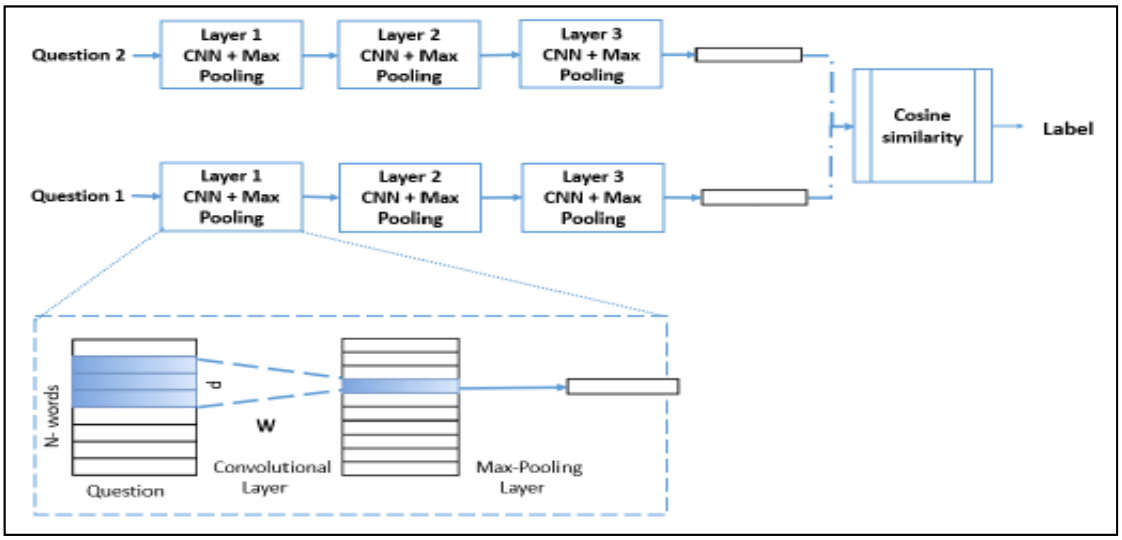


Figure 4. The CNN Architecture Model (Al-Theiabat and Al-Sadi, 2019)

Fadel *et al.* (2019) proposed a model divided into several layers: input layer, sequence representation extractor, merging layer, and deep neural network that consists of 4 fully connected layers, as shown in Figure 5. Before that, there are significant steps that make the dataset ready to use, pre-processing. So, using this step, they augmented the dataset from 11,997 to 45,514 examples using four different methods (positive transitive, negative transitive, symmetric, and reflexive). In the input layer, to extract word embedding, they used the Arabic ELMo pre-trained model and fed it into a sequence representation extractor that consists of two Ordered Neurons LSTM to apply sequence weighted attention. After that, they merged them by using the pair-wise

squared distance function for each question pair. The F1-score of this model is 96.49% in the public leaderboard and 94.84% in the private leaderboard.

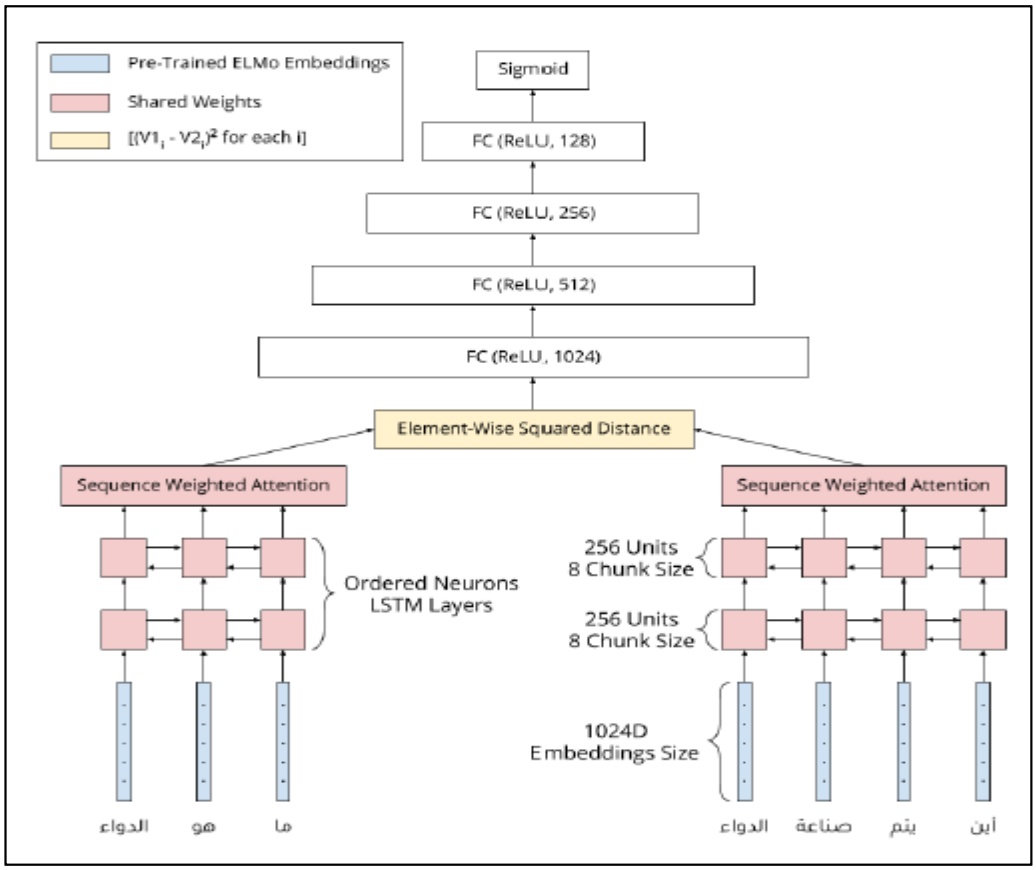


Figure 5. The Proposed Model of the Tha3roon Team (Fadel *et al.*, 2019)

Sharifi *et al.* (2019) proposed a controlled method. The first step is preprocessing, which prepares the data set, such as removing diacritics, tattwil, correcting punctuation, *etc*., to input the model. The second step is to extract features. Then, support vector machines classify the similarity of sentences. Finally, they used the *SelectKBest* algorithm to build the model, which allows them to obtain the number of features selected as input. *SelectKBest* has multiple options to choose the best result. And based on the evaluation of each calculation function, they selected 38 practical functions. The result of SelectKBest F (38) is 82.58%. Table 4 summarizes the best accuracy score for some of the participated teams at NSURL-2019 shared Task 8.

Table 4. Summary of Best Result for the Participating Teams in NSURL

Team Name	Classifier/ Model	F1-score
The Inception (Al-Theiabat and Al-Sadi, 2019)	BERT	0.9592
Tha3aroon (Fadel *et al.*, 2019)	ELMo	0.9485
Onekaggler	Bidirectional GRU	0.9481
Speech Translation	LSVM	0.8269
AtyNegar (Sharifi *et al.*, 2019)	SVM	0.8258

Chapter III: Technology Used

Recently, many pre-trained language models improve natural language processing tasks, like question answering, text classification, and language inference. Pre-trained models deal with a massive amount of unlabeled data to understand the language representation before adjusting the model to downstream tasks to reduce the time needed for training the model from scratch. Specifically, to utilize the pre-trained model on downstream tasks, an excellent fine-tuning strategy should be adopted to improve performance (Antoun *et al.*, 2021).

The most popular type of pre-trained language model is BERT, an abbreviation for Bidirectional Encoder Representations from Transformers. In addition, a wide range of its various models considers BERT by inserting different modifications to boost this model, such as adjusting model structure, enhancing, and optimizing fine-tuning strategies with additional knowledge (Antoun *et al.*, 2021).

BERT is designed to adapt both right and left context in all layers to pre-train deep bidirectional representations from the unlabeled text. Furthermore, just by adding one additional output layer, the pre-trained BERT model may be fine-tuned to design modern models applicable for a wide range of tasks (Devlin *et .al*, 2019).

Specifically, we will use the AraBERT model that describes pretraining the BERT transformer model for the Arabic language. Remarkably, the AraBERT model applies the BERT model on a large-scale Arabic corpus.

In comparison to English, Arabic is a morphologically rich language with few resources and a less investigated syntax. Due to these limitations, ANLP tasks such as

Sentiment Analysis, NER, and question answering represent a challenge for researchers (Antoun *et al.*, 2021).

The larger corpus is used for pre-training, the more efficiency gain. For that reason, we add three different datasets, namely Mawdoo3, back-translation, translated Quora question pairs, to the AraBERT origin dataset. Moreover, to boost our result, additional effective mechanism: ensemble was added to our work.

In general, in this chapter, we will discuss the anatomy of the BERT model that the basic model used in the AraBERT model and our work. Consequently, we will spotlight the attention mechanism and ensemble shortly and show the overall proposed system for this task.

3.1 Bidirectional Encoder Representations from Transformers (BERT) Model

In detail, BERT consists of two stages: pre-training and fine-tuning. First, in the pre-training stage, the distinctive pre-training tasks are used to train the model. Then, sequentially, during the fine-tuning, all pre-trained parameters that initialized in the first stage are fined tuned depending on labeled data from downstream tasks (Devlin et .al, 2019), as shown in Figure 6.

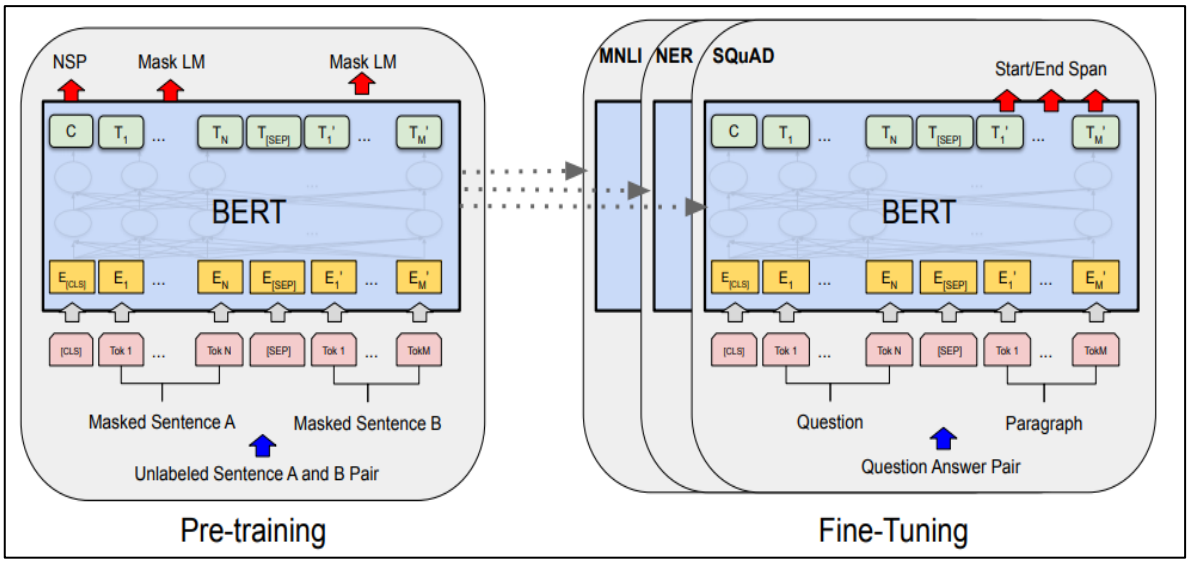


Figure 6. Overall pre-training and fine-tuning procedures for BERT (Devlin *et .al*, 2019)

3.2 Additional Details for BERT

This section briefly explains the major phases of the BERT model. for each task, there are some changes in fine-tuning. Based on tasks, we can choose sequence-level tasks such as Sentence pair classification tasks or token-level tasks such as Question answering tasks.

3.2.1 BERT Pre-Training Phase

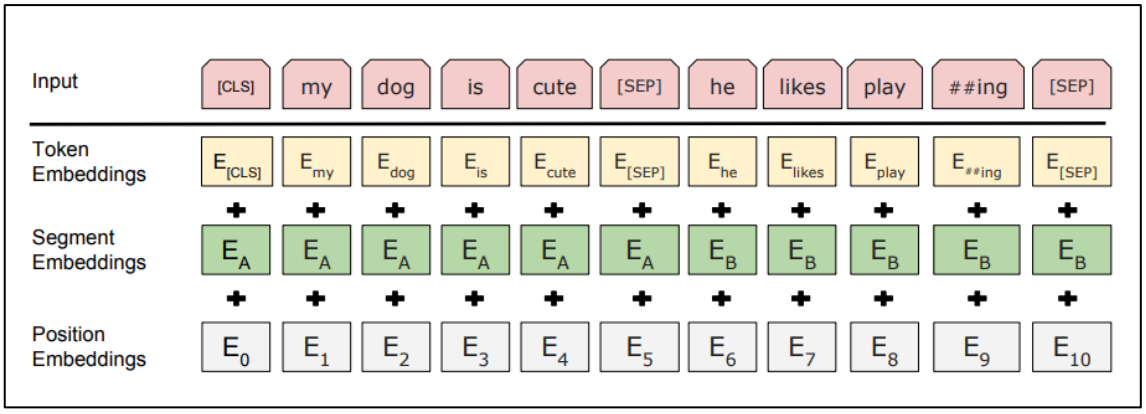
The input representation of BERT allows this model to cover a wide range of downstream tasks by representing both a single sentence and a pair of sentences as one input token sequence for BERT. Mainly, this is done by taking two sentences randomly from the corpus. The sentence that refers to a contiguous text with a varied length may be shorter or longer than single sentences (Devlin *et al*., 2019).

The BERT offers pre-trained models with settings like the number of layers (*i.e*., transformer layers), the parameters, the hidden size, and the number of self-attention heads. All of these settings provide multiple BERT models called either BERT-Large or

BERT-base. In addition, there are pre-training models for the English language and other 104 languages, including Arabic, called the multilingual model.

The BERT model uses Wordpiece embeddings (Wu *et al.*, 2016) with a 30,000-token vocabulary. Furthermore, the first sentence receives the *A* embedding for classification tasks, and the second receives the *B* embedding. Each sentence always starts with a special classification ([CLS]) which be helpful to represent the aggregate sequence for classification tasks. However, a special token ([SEP]) is used to separate Sentence pairs packed into a single sequence, or a learned embedding is added to every token to determine if it belongs to sentence A or sentence B. In Figure 7, the BERT input representation is implemented, revealing that the input representation is built by summing the corresponding token, segment, and position embeddings for a given token (Devlin *et al*., 2019).

Figure 7. BERT input representation (Devlin *et al.*, 2019)



3.2.2 Fine-tuning Phase

The hyperparameter of a model for both pre-training and fine-tuning stages are the same despite some differences between them like batch size, learning rate, and several training epochs to determine the optimal hyperparameter values for a specific task. The BERT model uses the stochastic gradient descent (SGD) method to mitigate the cross-entropy loss in fine-tuning. Practically, the performance of BERT fine-tuning is susceptible to being affected by the variation of data training orders and noise.

The enhancement of fine-tuning takes the attention of some researchers, such as (Xu *et al.*, 2020). In this paper, different fine-tuning strategies and hyperparameters were examined to improve BERT for text classification. However, to achieve this improvement on the performance of fine-tuning, they need to leverage external data or knowledge. Therefore, the ensemble method is used with the BERT model and achieves better performance by combining several fine-tuned-based models. Nevertheless, this method has drawbacks like training cost and model size (Antoun *et al.*, 2021) (Devlin et .al, 2019).

We are interested in the sentence pairs classification task to apply the Arabic tasks through fine-tuning. Figure 8 illustrates the required model.

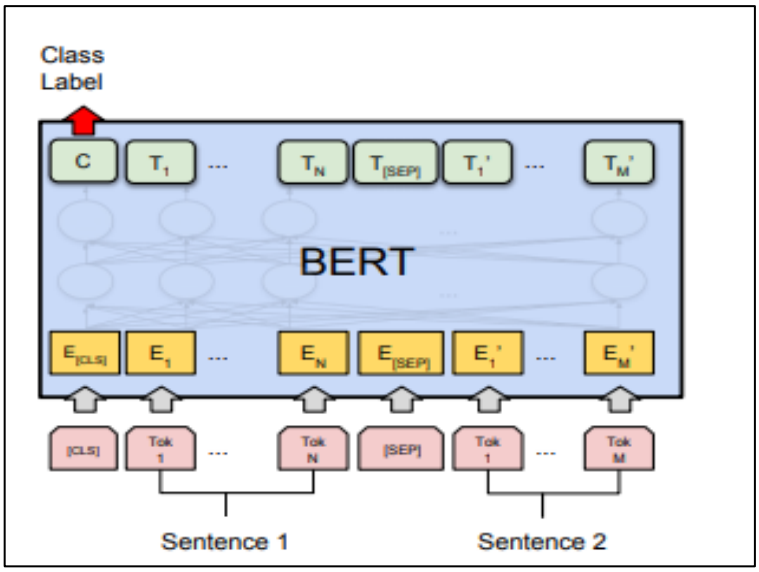


Figure 8. The BERT model used for classifying similarity sentences

3.3 Architecture of BERT Bidirectional Self-Attention Transformer

The BERT model has inherited the implementation of the transformer (Vaswani *et al.*, 2017). A bidirectional self-attention transformer consists of a multi-headed self-attention instead of using recurrent layers most commonly used in the encoder-decoder design. Moreover, in the encoder-decoder architectures, the encoder performs a mapping process between an input sequence of symbol representation $(x_1, \cdots, x_n)$ and a sequence of continuous representations $z = (z_1, \cdots, z_n)$ to produce an output sequence of a symbol $(y_1, \cdots, y_m)$ one element at a time. As fully connected layers for both the encoder and decoder, the result from the previous step is fed to the next step.

This Transformer follows the former architecture using stacked self-attention and pointwise shown in Figure 9.

3.3.1 Encoder and Decoder Stacks Encoder

The encoder consists of N = 6 symmetrical layers piled on top of each other. Each layer is divided into two sub-layers. The first is a multi-head self-attention system, while the second is a position-wise fully linked feed-forward network. The output of

dimension $d_{\text{model}} = 512$ is the result of all sub-layers in the model along with the embedding layers. Further, the residual connection around each sub-layers is utilized and followed by layer normalization (Vaswani *et al.*, 2017).

As an encoder, the decoder consists of N = 6 symmetrical layers piled on top of each other. However, the decoder adds a third sub-layer to the exists two sub-layers, this additional sub-layer implements multi-head attention over the output of the encoder stack. The objective of inserting the self-attention sub-layer is to prevent the positions from reaching subsequent positions. This masking can guarantee the prediction of position *i* because of relying only on the known outputs at positions less than *i* (Vaswani *et al.*, 2017).

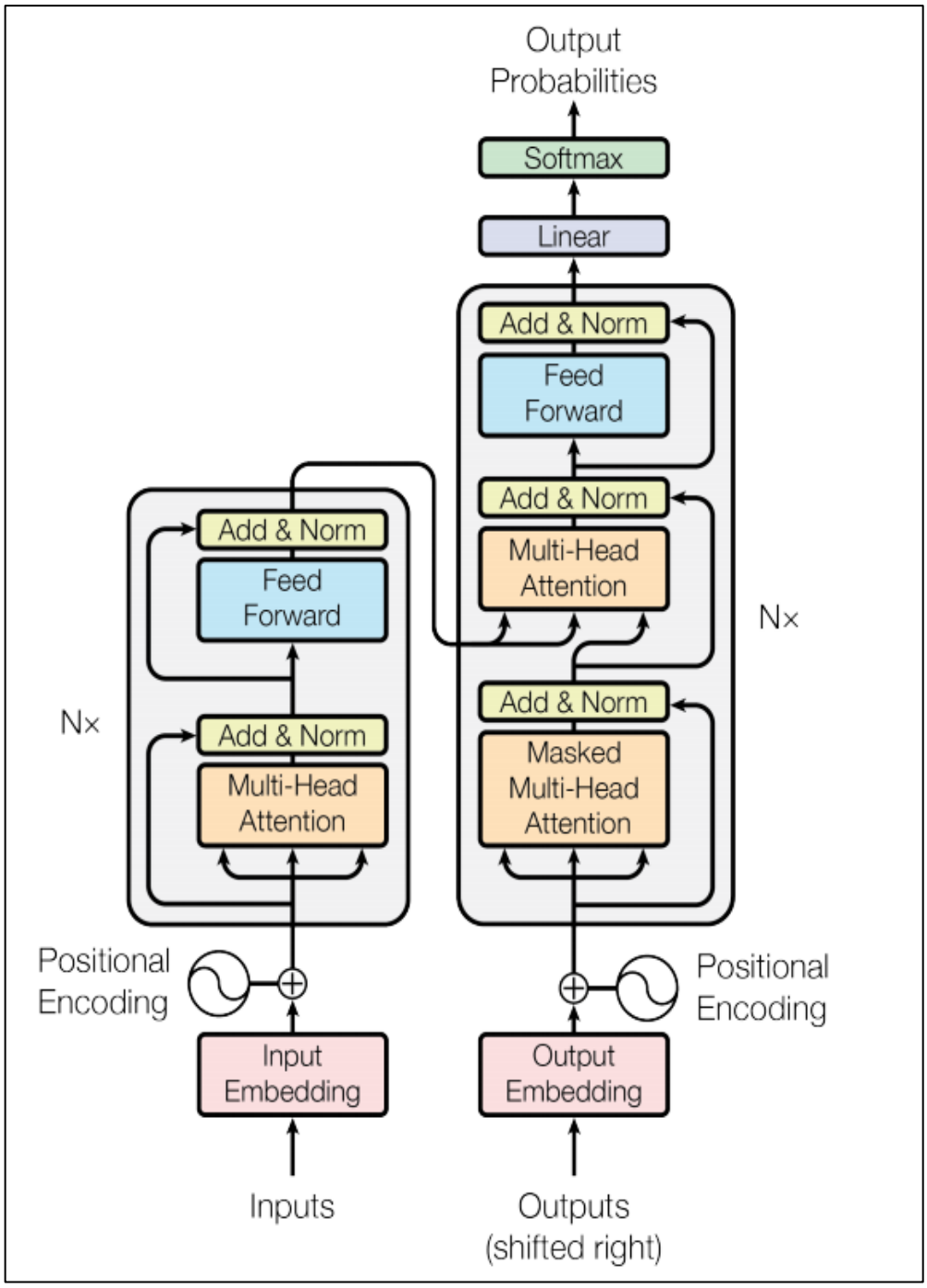


Figure 9. The Transformer Model Architecture (Vaswani *et al.*, 2017)

3.3.2 Attention

The attention function deals with three vectors, the query, keys, values, to compute the output vector. This process depends on mapping the query and the corresponding key to compute the weight assigned to each value. After that, the weighted sum of the values forms the output eventually (Vaswani *et al.*, 2017). To calculate the output matrix, Devlin *et al.* (2019) illustrated how in Equation (1):

$$Z = Attention\ (Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (1)$$

where:

- Q: Query matrix
- K: Key matrix
- V: Value matrix
- D_K: the dimension of key

3.3.3 Multi-Head Attention

The use of multi-head attention is beneficial to parallelize the operation of attention function on a linear implementation of queries, keys, and values h times with different learned linear projections to d_k, d_q, and d_v dimensions, respectively. Instead of using a single attention function with model-dimensional keys, values, and queries. The multi-head attention allows the model to access information from different representation subspaces at different positions concurrently, as shown in Figure 10 (Vaswani *et al.*, 2017).

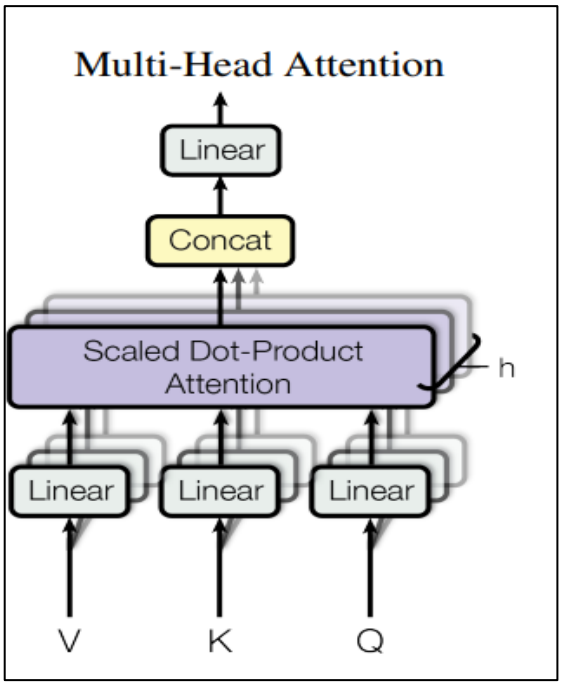


Figure 10. Multi-Head Attention Structure (Vaswani *et al.*, 2017)

3.3.4 Position-wise Feed-forward Networks

The fully connected feed-forward network is a part of each layer in encoder and decoder layers, along with attention sub-layers. This network is a composite of two linear transformations separated by a *ReLU* activation applied to each position individually identically (Vaswani *et al.*, 2017).

3.3.5 Embeddings and SoftMax

The former transduction models consider the use of learned embeddings to get the vectors of dimension d_{model} *via* transforming the input tokens and output tokens. On the other hand, the usual learned linear transformation and *softmax* function are used to convert the decoder output to predicted next-token probabilities. In this model, embedding layers and the pre-*softmax* linear transformation share the same weight matrix (Vaswani *et al.*, 2017).

3.3.6 Positional Encoding

Positional encoding is to determine the position of tokens in sequence and maintain the order of the sequence some additional information about the position should be adopted to the model. For that reason, the positional encodings are summed to the input embeddings.

3.4 Ensemble Learning Mechanism

One of the machine learning models that combine the predictions from more than two models is Ensemble. The effect of Ensemble on multiple models produces accurate results. The main reason to use this method is to achieve better predictive performance. Moreover, the reason why it performs slightly better than a single model is that different models make different “mistakes” (misclassify an observation), so we could have two models that misclassify a specific instance and three models that get it right, so by voting we get a correct prediction for that instance. In contrast, if we had used one of the two missed models individually, we would have gotten an incorrect prediction. Because we already have good accuracy for the individual models, the probability of the majority of the votes being incorrect is low enough.

There are several well-known ensemble methods such as voting, staching, and boosting. In this project, we used the voting for classification because the predictions are the results of the majority vote of participating models. For the voting ensemble method, there are two approaches, hard voting, and soft voting. The hard voting includes taking the summation of the predictions for each class label and predicting the most popular vote from the class labels. But the soft voting includes taking the summation of the predicted probabilities (score) for each class label and predicting the largest probability from the class labels (Machine Learning Mastery page, 2021).

We just trained five models with different random seeds and made predictions by hard voting between them (hard voting means voting by label predictions *e.g.,* we have [1, 0, 1, 0, 1] as predictions from our five models, so the final prediction is 1 as that's the most popular vote). Figure 11 shows the process of ensemble learning mechanism that we will do in our task.

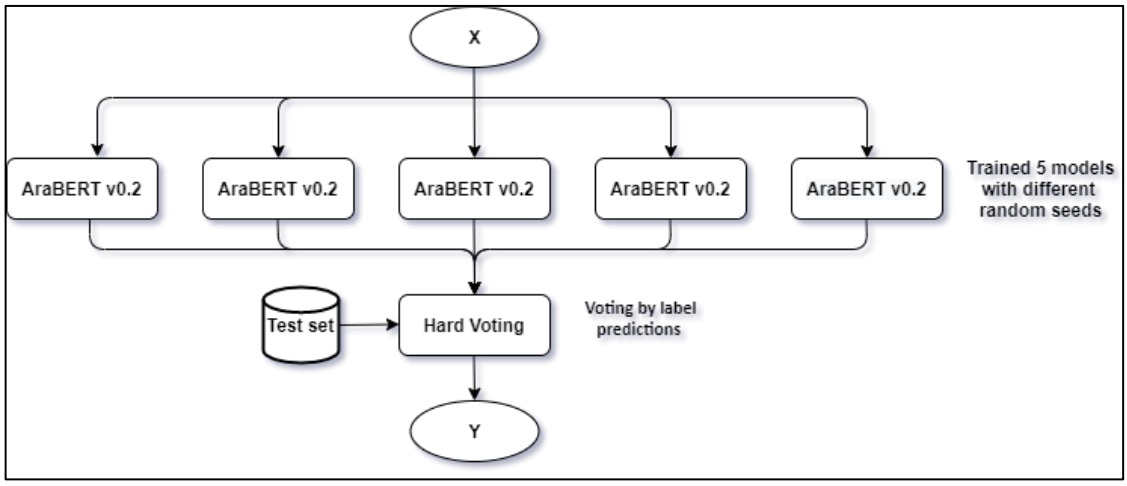


Figure 11. Hard Voting Ensemble Method

3.5 The Proposed Model

In this section, we will show the overall structure that we explained in the previous section. The BERT model is the base model that can finetune with a further layer to create a state-of-the-art model for several tasks. After pre-training BERT on the Arabic language, the AraBERT is a derived model for the Arabic NLP task. Next, we will pre-train the AraBERT model on more Arabic datasets divided into Mawdoo3, back-translation Mawdoo3, and Translated Quora question pairs datasets and then train the model on the main dataset to note the improved accuracy. Finally, ensemble learning methods are essential to achieve the best performance. Figure 12 shows the model divided into three phases: pre-training, training, and testing.

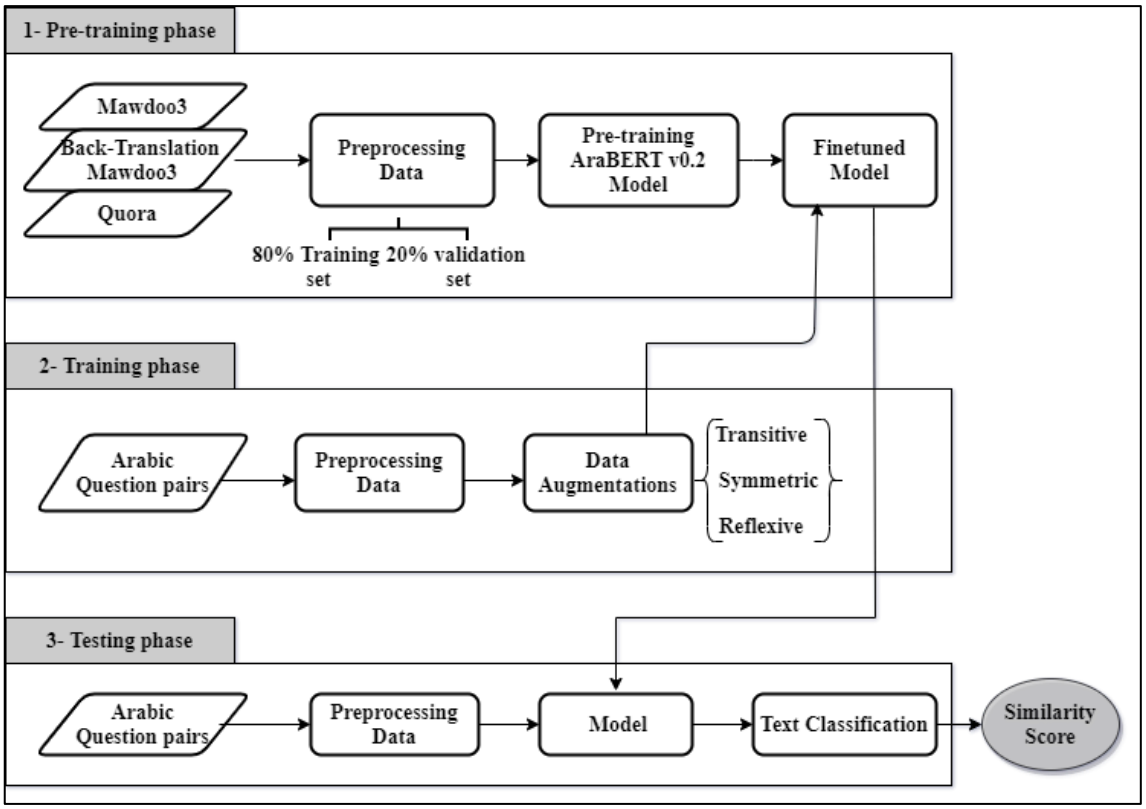


Figure 12. The Model Archeticture

Chapter IV: Dataset Preparation

This chapter aims to study, analyze, and preprocess the Mawdoo3 dataset suitable for our proposed system. The ANLP Preprocessing Operations consist of several steps before feeding into the model, such as clean the data from punctuation marks and special characters, tokenize data, stem data, normalize data (Larabi *et al.*, 2019). The following is a review of the most topics discussed in this chapter.

- Analyze the Mawdoo3 dataset.
- Analyze the original and translated the Quora Question pairs dataset
- Prepare the dataset for the model.

4.1 Data Analysis

Pre-training models must need many datasets from different resources based on a challenge or a task. For example, the dataset of the pre-training BERT for Arabic language understanding covers major Arabic news from various media in different Arab regions. From this critical point, we suggest several ways to generate more Arabic questions using translation and augmentation processes.

4.1.1 Mawdoo3 Dataset

Our experimental data is provided by Mawdoo3. Mawdoo3 dataset contains many different types of Arabic questions with its label. To know the information about the dataset that we are used as a training dataset, Figure 13 shows a full description of the Mawdoo3 dataset. It contains 11,997 labeled pairs, the number of data columns, the data type for Question 1 and Question 2 is the object, the label is an integer, and a sample of the first five questions from training datasets.

The dataset divides into two parts as 8,282 questions for training data and 3,715 questions for testing data. Label '1' means the question pairs are similar with 55.01% out of the training dataset, where label '0' means the question pairs are different with 44.99% out of the training dataset.

Figure 13. The full description of Mawdoo3 dataset

The count of questions in the training dataset that are either repeated or unique is 23,994 questions, whereas 10,861 questions are unique, and 7,960 questions are repeated. Figure 14 represents a plot of the count of the unique and repeated questions.

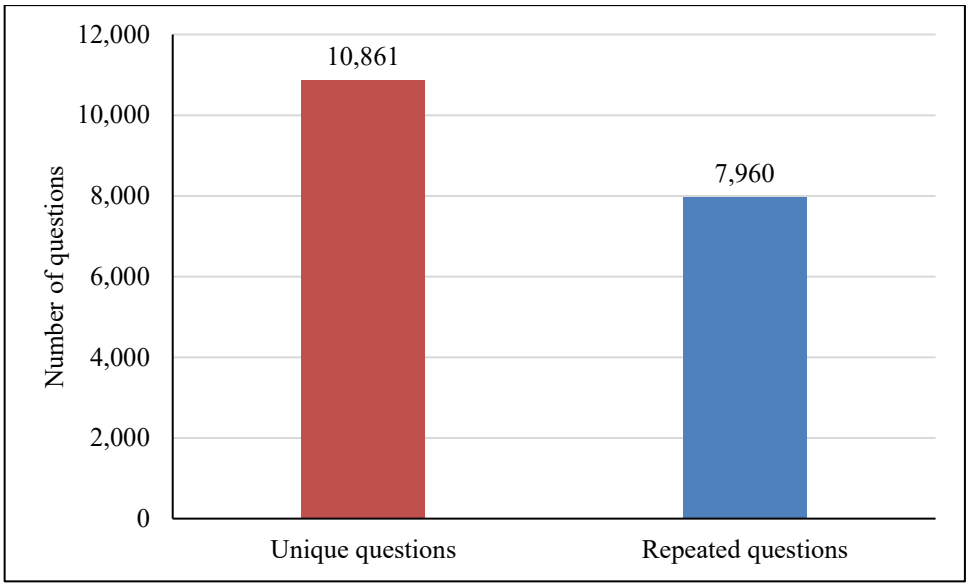


Figure 14. Plot of the count of the unique and repeated questions

Many questions are repeated in the dataset. To know the number of occurrences of the question for each question, Figure 15 shows that as a Log-histogram plot, and Table 5 shows examples for the occurrence number of questions depending on the following figure. For example, 2759 questions have been repeated two times in the dataset.

Table 5. Examples for the Occurrence Numbers for Each Question

Number of occurrences	Number of questions
2	**2759**
5	**205**
10	**9**
20	**1**

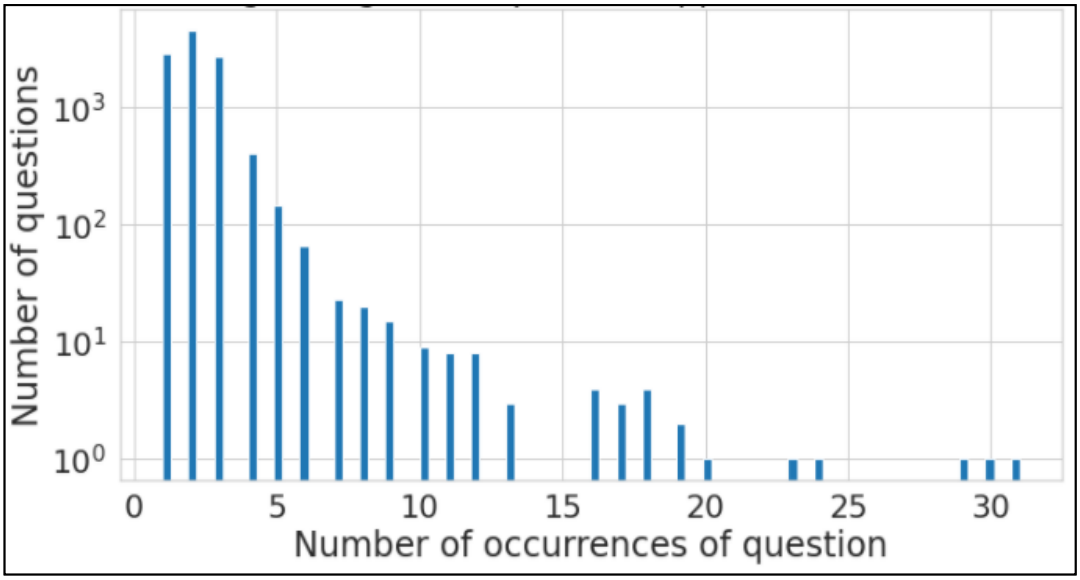


Figure 15. Log-histogram of Question Appearance Counts

4.1.2 The Original and Translated Quora Question Pairs Dataset

Quora is an online Knowledge-sharing platform that gives researchers a chance to apply its dataset to solve different tasks related to Quora. Quora support various areas such as machine learning, natural language processing, network science, *etc*. the first dataset released to finding similarity between English Questions. Every single record has two questions (Quora page, 2021).

The dataset consists of more than 400,000 records of pairs of probable duplicate questions. Each line contains IDs for each question, two full texts of questions, and a binary value that indicates if the pair of questions are similar or not, as shown in Figure 16 (Quora page, 2021).

id	qid1	qid2	question1	question2	is_duplicate
447	895	896	What are natural numbers?	What is a least natural number?	0
1518	3037	3038	Which pizzas are the most popularly ordered pizzas on Domino's menu?	How many calories does a Dominos pizza have?	0
3272	6542	6543	How do you start a bakery?	How can one start a bakery business?	1
3362	6722	6723	Should I learn python or Java first?	If I had to choose between learning Java and Python, what should I choose to learn first?	1

Figure 16. Sample of Quora Question Pairs (Quora page, 2021)

After analyzing the dataset of Quora, based on is_duplicate columns, the parentage of non-duplicate questions is 63%, and 37% for duplicate questions, as shown in figure 17. This property may be helpful to assess the translation process changing this or not.

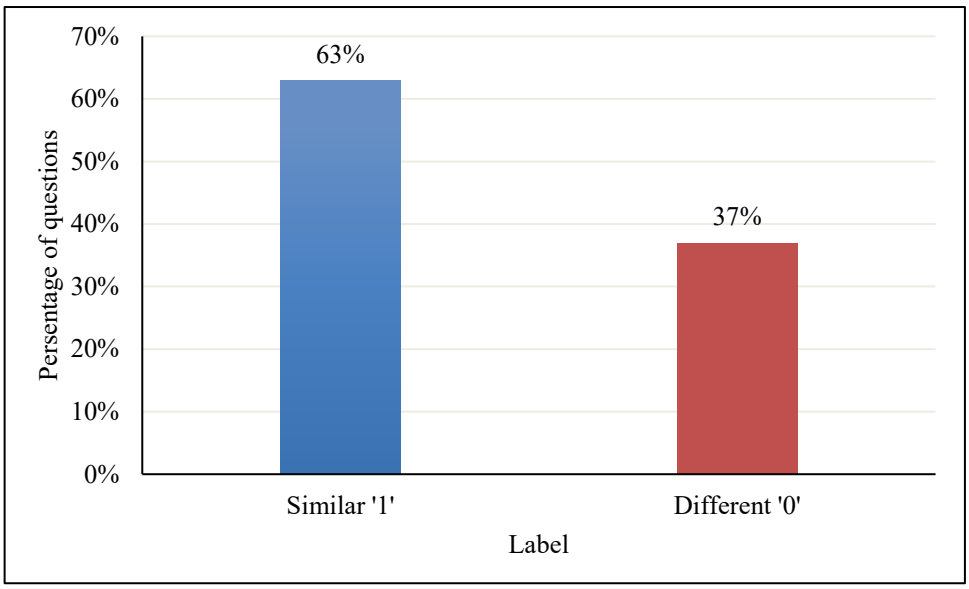


Figure 17. The Percentage of Questions that are Similar and Different

From the above discussion, we conclude that there are similarities between our dataset and the Quora question pairs dataset in principle. So, we need to modify the Quora question dataset to be suitable for the Mawdoo3 dataset. Table 6 shows the comparison between them to plan the steps we need to get the same datasets.

Table 6. Comparison between Quora and Mawdoo3 Datasets

	Quora Question pair	**Mawdoo3**
Language of dataset	English	Arabic
IDs for each question	Yes	No
Number of columns	6	3
Number of rows	404,290	11,997
Size	50.88MB	1.36MB

Based on Table 6, The following steps explain the modifications on the Quora dataset:

1. Drop first 3 columns (id, qid1, qid2) as shown in Figure 18.

```
Dataframe info.:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 404351 entries, 0 to 404350
Data columns (total 6 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   id            404351 non-null  int64
 1   qid1          404351 non-null  int64
 2   qid2          404351 non-null  int64
 3   question1     404350 non-null  object
 4   question2     404349 non-null  object
 5   is_duplicate  404351 non-null  int64
dtypes: int64(4), object(2)
memory usage: 18.5+ MB
None
```

Figure 18. The data frame information for Quora before dropping the first 3 columns

```
Dataframe info.:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 404290 entries, 0 to 404289
Data columns (total 3 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   question1     404289 non-null  object
 1   question2     404288 non-null  object
 2   is_duplicate  404290 non-null  int64
dtypes: int64(1), object(2)
memory usage: 9.3+ MB
None
```

Figure 19. The data frame information for Quora after dropping the first 3 columns

2. Save the modified dataset in a new data frame to use in the next step.

3. The Quora question pairs are over 400,000 question pairs. Translating all of that would take a long time and training the models would take even longer. We are using the google translate API to translate, and it has a limit to how many requests you can do at a specific time window, as shown in Figure 20.

```
with open('../input/quoradic/quora_chunk1.pkl', 'rb') as f:
    quora_chunk1 = pickle.load(f)
```

Figure 20. The command for translating Quora dataset from English to Arabic

4. We can get the translations in python as shown in Figure 21.

```
#load our translations from the file
import pickle

with open('../input/arabert/translations.pkl', 'rb') as f:
      translations = pickle.load(f)
```

Figure 21. The command to load translation from Pickle file

Table 7 shows the number of translated Quora question pairs examples. We have used about 10,000 pairs of questions as a chunk; using the whole dataset would take something like more than 10 hours to translate all of them, even though it's just an estimate (could be more, could be less). Because the google API limit and training our model with all 400,000 question pairs also would take a while for sure.

Table 7. Sample of the translation process for Quora dataset

Question 1	Question 2	label
what are natural numbers? ما هي الأعداد الطبيعية؟	what is the least natural number? ما هو أقل عدد طبيعي؟	0
which pizzas are the most popularly ordered pizzas on Domino's menu? ما هي البيتزا الأكثر طلبًا على قائمة دومينوز؟	how many calories does Domino's pizza have? كم عدد السعرات الحرارية في بيتزا دومينوز؟	0
how do you start a bakery? كيف تبدأ مخبز؟	how can one start a bakery business? كيف يمكن للمرء أن يبدأ عمل مخبز؟	1
should I learn python or java first? يجب أن تعلم بايثون أو جافا أولا؟	if I had to choose between learning java and python, what should I choose to learn first? إذا كان لي أن تختار بين جافا التعلم بايثون، ما يجب أن أختار لتعلم أولا؟	1

4.2 Preparing the Dataset

Before the Arabic dataset feed into the model, the major ANLP phase is the data preprocessing to be ready to use it. So, we depict the main phases of our model including data preprocessing, and data augmentation.

4.2.1 Preprocessing

It is the first step to process the dataset before feeding it into the model. Our data preprocessing function takes the dataset as a *.csv* file, shuffles it, and splits it into 80% for the training set and 20% for the validation set. Marie-Sainte *et .al* (2018) discussed the preprocessing operations, and we consider one of these operations. It is a cleaning data by separating the punctuations marks as shown in Table 8 from the letter to preserve the context of the questions.

- The example that describes before and after separating between letter and a question mark:

Before preprocessing	After preprocessing
الى ماذا يهدف الاستثمار؟	الى ماذا يهدف الاستثمار ؟

Table 8. The punctuation marks

.	,	'	@	"	!	$	%	^	&	*	(
)	-	/	+	|	{	}	[	]	_	=	<
>	~	؛	\	:	،	′	;				

Also, In Transformer, the main tool for preprocessing data is called tokenizer. The main function is to split a given question into words (*i.e.,* or subwords, punctuation marks, *etc.*) named tokens. For using a pretrained model, we called *AutoTokenizer* class imported from the transformers' library to use the associated pretrained tokenizer. Using the from_pretrained method, we can download the vocab used during pretraining the AraBERT base v0.2 model on our datasets—for example, the vocabulary size when pretraining AraBERT on mawdoo3 dataset was 64000.

We applied the tokenizer for both lists of training and validation datasets for Question 1 and Question 2. This step returns a list of integers which means that each token has an input id.

When sending a batch of questions at a time to the tokenizer, we should add more specific requirements to the tokenizer command (HuggingFace page, 2020) , such as:

1. Padding: to pad each question to the maximum length there is in a batch

2. Truncation: to truncate each question to the maximum length the model accepts (*e.g.,* max length = 512).

3. Returns tensors: to return tensors.

Figure 22 shows the command used to tokenize a list of questions with an example, and Figure 23 shows the output of this command.

```
batch_sentences = ['ما هي الطرق الصحيحة للاعتناء بالحامل ؟',
                   'كيف اهتم بطفلي ؟']
encoded_input = tokenizer(batch_sentences,padding=True, truncation=True, max_length=512,return_tensors='tf')
encoded_input
```

Figure 22. The command used for tokenization of Arabic questions

```
{'input_ids': <tf.Tensor: shape=(2, 11), dtype=int32, numpy=
array([[    2,   394,   634,  3165, 11417, 12430,   625,  2330,   711,
         105,     3],
       [    2,  1732,  7772,  1018, 49941,   105,     3,     0,     0,
            0,     0]], dtype=int32)>, 'token_type_ids': <tf.Tensor: shape=(2, 11), dtype=int32, numpy=
array([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
       [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]], dtype=int32)>, 'attention_mask': <tf.Tensor: shape=(2, 11), dtype=int32, numpy=
array([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
       [1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0]], dtype=int32)>}
```

Figure 23. The output of the tokenization command for Arabic questions

From Figure 23, we noticed that the tokenizer command returns a dictionary with string keys and tensors values (*i.e.,* input id, token type id, and attention mask). The attention mask list refers to tokens that the model should pay attention to; otherwise, it represents padding. Figure 24 shows an actual Arabic question after decoding the encoded input with special tokens like [CLS], [SEP], and [PAD].

```
for ids in encoded_input["input_ids"]:
    print(tokenizer.decode(ids))
```

```
[CLS] ما هي الطرق الصحيحة للاعتناء بالحامل ؟ [SEP]
[CLS] كيف اهتم بطفلي ؟ [SEP] [PAD] [PAD] [PAD] [PAD]
```

Figure 24. The decoded output

4.2.2 Data Augmentation

As we mentioned previously, the training dataset contains 11,997 question pairs. To make the model generalize better, we need to enlarge the dataset before training using several rules, as shown in Figure 25.

Assume that we have 3 questions (Q1, Q2, Q3)

1- Transitive

There are two types of transitions. The first one is *Positive* Transitive: if Q1 is similar to Q2 and Q2 is similar to Q3, then Q1 is similar to Q3. The second type is *Negative* Transitive: if Q1 is similar to Q2 and Q2 is *NOT* similar to Q3, then Q1 is *NOT* similar to Q3. We noticed that the Transitive rule enlarges the Arabic dataset from 11,997 to 17,487 question pairs.

2- Symmetric

In the code, we created the new data frame to swap between question 1 and question2 quickly. If Q1 is similar to Q2, then Q2 is similar to Q1, and if Q1 is NOT similar to Q2, then Q2 is NOT similar to Q1. We noticed that this rule enlarges the dataset to 34,974 (doubling the number of examples of the previous rule).

3- Reflexive

We created a new data frame for reflexive to copy all the unique questions with their label in the code. As we assume initially, if Q1 is similar to itself and unique, then duplicate it. We noticed that this rule enlarges the number of examples to 45,514 as a total size.

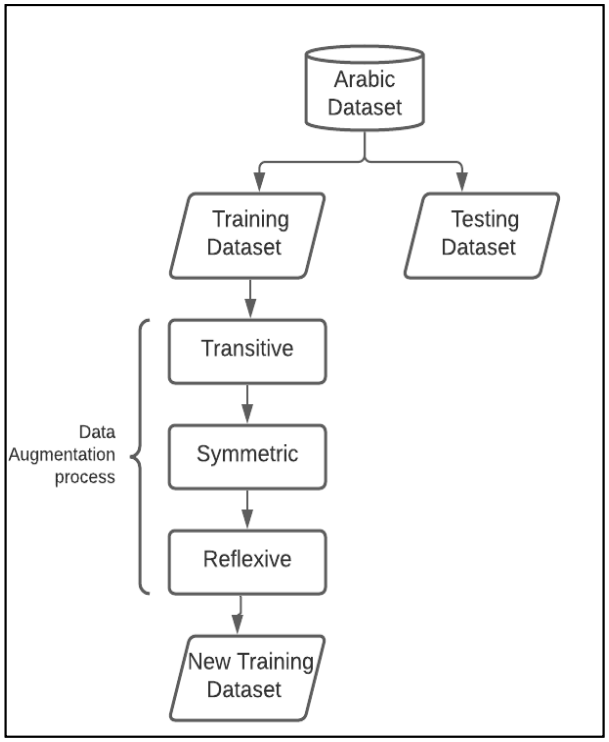


Figure 25. The flowchart of enlarging Mawdoo3 dataset using rules: Transitive, Symmetric, and Reflexive

Table 9 summarizes the total number of question pairs and the increase for each type of data augmentations: original, transitive, symmetric, and reflexive. We calculated the amount of increment by subtracting the original dataset from each data augmentation type.

Table 9. The total number of data with the three augmentation types

Data Augmentation	Training Example Number	Amount of increment
Original	11,997	0
Original and transitive	17,487	5,490
Original, Transitive, and Symmetric	34,974	22,977
Original, Transitive, Symmetric. and Reflexive	45,514	33,513

4- Data Translation

There are several NLP data augmentation methods. Also, In this project, we used the Back translation process to generate a new data set with different words while keeping the context of text data. The main function of Back translation is to translate the text data (sentences or questions) to another language such as English, French, Spanish, *etc.*, and then repeat the translation process to get the original language (Neptune.ai blog page, 2021). Figure 26 shows the back translation process for the Mawdoo3 dataset from Arabic to English and back using Google translation. We noticed that the questions still have the same meaning through this process, but they have different syntax, as shown in Table 10.

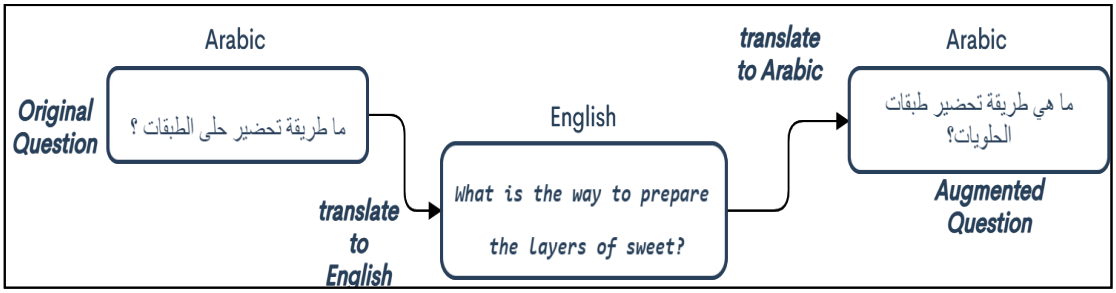


Figure 26. The back-translation process

The output of translated dataset from Arabic to English *via* google translation takes a lot of time, so we can load translated dataset by the translation.pkl file, as shown in Figure 27.

```
with open('back_translations.pkl', 'rb') as f:
    back_translations = pickle.load(f)
```

Figure 27. The command to load back translation – Pickle file

Table 10. The back-translation examples for the Mawdoo3 dataset

Original Question	English	Augmented Question
ما هي الطرق الصحيحة للاعتناء بالحامل ؟	What are the correct ways to take care of a pregnant woman?	ما هي الطرق الصحيحة للعناية بالمرأة الحامل؟
ما هي وسائل الاتصالات الحديثة ؟	What are the modern means of communication?	ما هي وسائل الاتصال الحديثة؟
ما طريقة تحضير محشي الكوسا ؟	How to prepare stuffed zucchini?	طريقة تحضير الكوسا المحشي؟
ما طريقة تحضير حلى الطبقات ؟	What is the way to prepare the layers of sweet?	ما هي طريقة تحضير طبقات الحلويات؟
من الآيات القرآنية عن الراعي والرعية ؟	Of the Quranic verses about the shepherd and the subjects?	من الآيات القرآنية عن الراعي والموضوعات؟

Chapter V: Experiments and Results

AraBERT pre-training model is evaluated on various Arabic language understanding downstream tasks such as sentiment analysis, named entity recognition, and question answering, and their result performed well (Antoun *et al.*, 2021).

This chapter aims to briefly explain the working environment that we used to implement our project. Why and how we select the AraBERTv0.2 model. The execution time spent in implementing it on the Mawdoo3 and Translated Quora question pairs datasets. A review of the results we obtained when we implemented it on the two datasets. And finally, made a comparison between these results with other results and interpreted them.

5.1 Research Environment

Our experiments on the AraBERT pre-training model were implemented on a DELL workstation with Intel(R) processor Intel Core-i5 3317U CPU @ 1.70 GHz. It has an 8GB DDR3 RAM, 128GB SSD, Windows 10 Pro, and 64-bit operating system. We have used the Kaggle environment and Google Colab to train more experiments, specifically when increasing the number of epochs and the size of the Arabic dataset, so it needs to accelerate the runtime of execution of this process.

The program was written using Python 3.7 language on PyCharm Community IDE. PyCharm was developed by the Czech company JetBrains (Darryl K, 2010). We chose the PyCharm environment because it can code analysis and support data science with Anaconda (PyCharm Page, 2021).

5.2 Pre-training the AraBERT Model

Before we execute any experiment, we should pre-train the model using another huge data set. In this section, three pre-training attempts separately use several Question pairs datasets such as the Mawdoo3 dataset, back translation dataset, and translated Quora Question pairs dataset to learn the model before training.

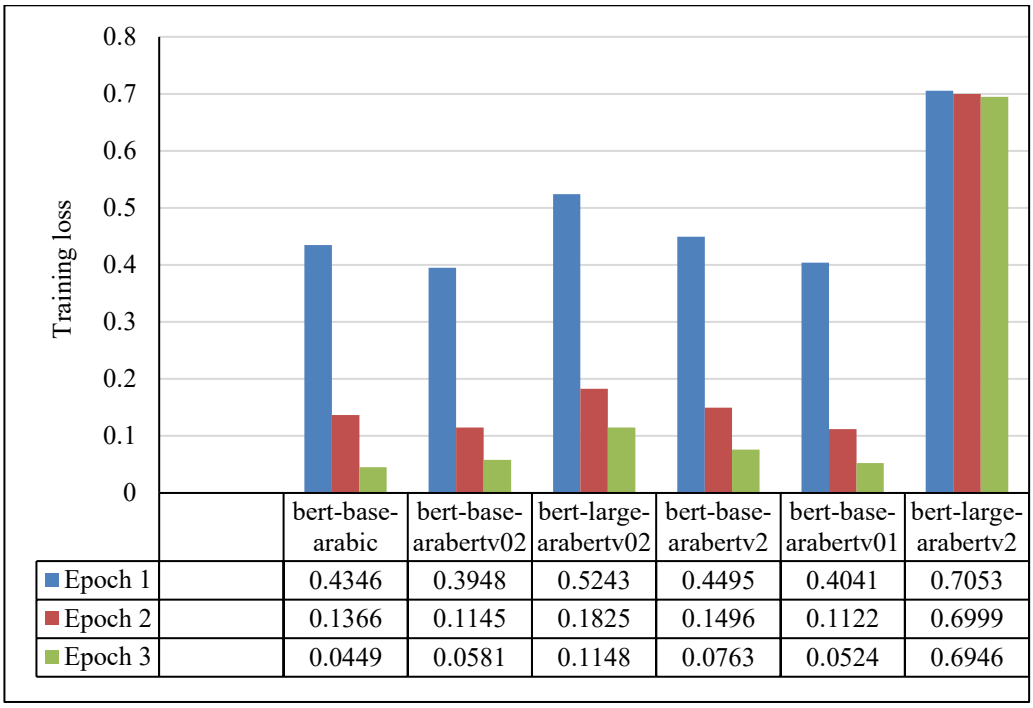
We have utilized the training_model and calculate_metrics functions created to train different models and look at the metrics. We can load the pre-training model we want to train from the hugging face hub model website as a checkpoint.

The first attempt is pretraining the AraBERT model on the Mawdoo3 dataset; it contains 11,997 question pairs. Table 11 shows different versions of the AraBERT model with its accuracy. We noticed that the AraBERT v0.2 has a higher value than other versions of the model.

Table 11. Accuracy for pre-training AraBERT models

Model	Size (MB/Params)	Dataset (Size)	Accuracy
AraBERTv1-base	543MB /136M	23GB	0.9397
AraBERTv0.2-base	543MB/ 136M	77GB	**0.9626**
AraBERTv0.2-large	1.38G 371M	77GB	0.5424
AraBERTv2-large	1.38G 371M	77GB	0.9359
AraBERTv2-base	543MB/ 136M	77GB	0.5375
AraBERTv0.1-base	543MB/ 136M	23GB	0.9469

Figure 28 shows the comparison between all versions of the pre-training model and based on training loss. We noticed that the relationship between loss and accuracy is inversely proportional. For example, the AraBERT v0.2 has lower loss values 0.4495, 0.1496, and 0.0763 at Epochs 1, 2, and 3, respectively



		bert-base-arabic	bert-base-arabertv02	bert-large-arabertv02	bert-base-arabertv2	bert-base-arabertv01	bert-large-arabertv2
Epoch 1		0.4346	0.3948	0.5243	0.4495	0.4041	0.7053
Epoch 2		0.1366	0.1145	0.1825	0.1496	0.1122	0.6999
Epoch 3		0.0449	0.0581	0.1148	0.0763	0.0524	0.6946

Figure 28. The training loss for all pre-training models

Based on the accuracy results, we have selected the bert-base-arabertv02 that achieves the highest result to train it on augmented datasets (*i.e.*, original, transitive, symmetric, reflexive).

5.3 Training Time

This project is interested in the training times for original and augmented datasets because they take a very long time in transfer learning. It is determined by the environment in which the project is carried out. The relationship between the environment and the amount of the dataset is directly proportional. So, the more dataset

processed, the more training time. Therefore, it is essential to pertaining models on a huge dataset to learn as much as possible.

We implemented our project using the GPU of the Kaggle website to boost the training process.

As shown in Figures 29 and 30, we compute the training time for all versions of pretraining AraBERT models on the Mawdoo3 dataset and training time for each data augmentation type, respectively. We noticed that the pretraining training time for the AraBERT base v0.2 model was better in both time and accuracy. Also, we noticed that the training time over five runs for each data augmentation step was always higher when increasing the number of question pairs.

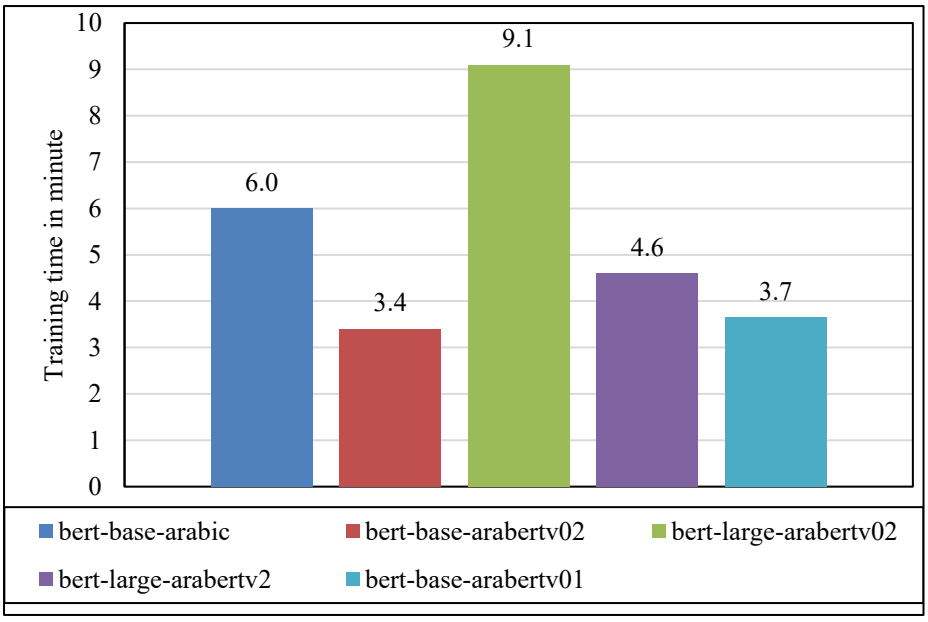


Figure 29. Comparison of the total training time between AraBERT models in minutes

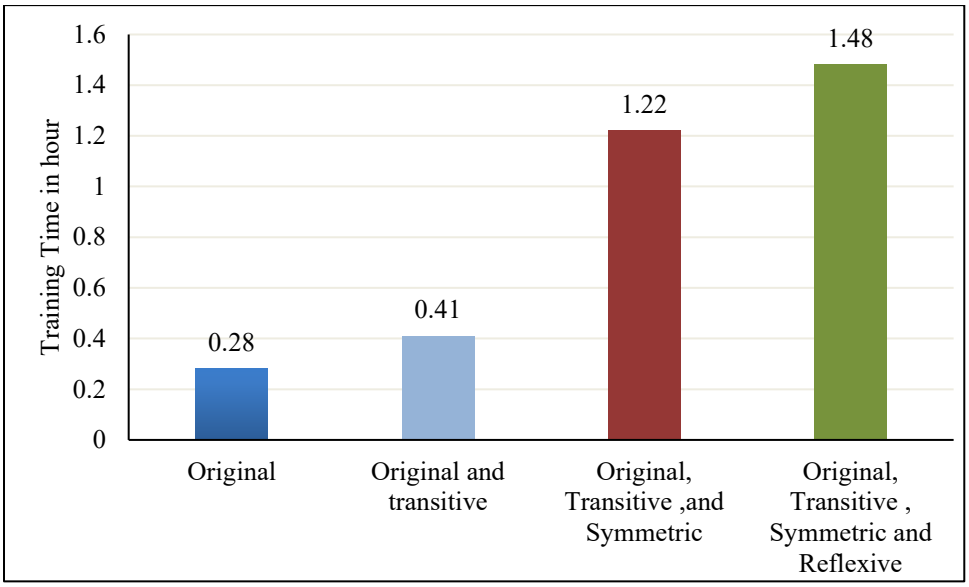


Figure 30. Comparison of the average total training time between augmented datasets for five runs

5.4 Results

In this work, all experiments have been done based on the parameters that give the best results with the following hyperparameters:

- Optimizer: Adam
- Learning Rate: 0 - $5x10^{-5}$ (learning rate schedules to be able to hit the minimum more precisely)
- Loss Function: Sparse Categorical Cross entropy
- Batch Size: 32
- Number of Epochs: 3

5.4.1 Training Mawsoo3 Data Set

In this section, we train the original dataset on AraBERT base v0.2, as shown in table 10. Firstly, we noticed that the augmentation process on the original dataset outperforms the original dataset itself. Moreover, the second note is about data augmentation itself, so when increasing the size of the dataset using clever techniques, we will get better accuracy, but this increases the training time.

Table 12. Accuracy of Mawdoo3 dataset

	Original	Data augmentation		
		Transitive	Symmetric	Reflexive
Accuracy	0.9558	0.9580	0.9607	0.9658

Figure 31 shows how training loss and validation loss decrease when the number of epochs is increased. In this experiment, we applied early stopping *via* a callback library to select the number of epochs. This way is to stop training once the model does not make any improvement in the result. The monitor that we used to specify the model performance is validation loss. So, we noticed that the training process stops after six epochs pass without any improvement in the accuracy.

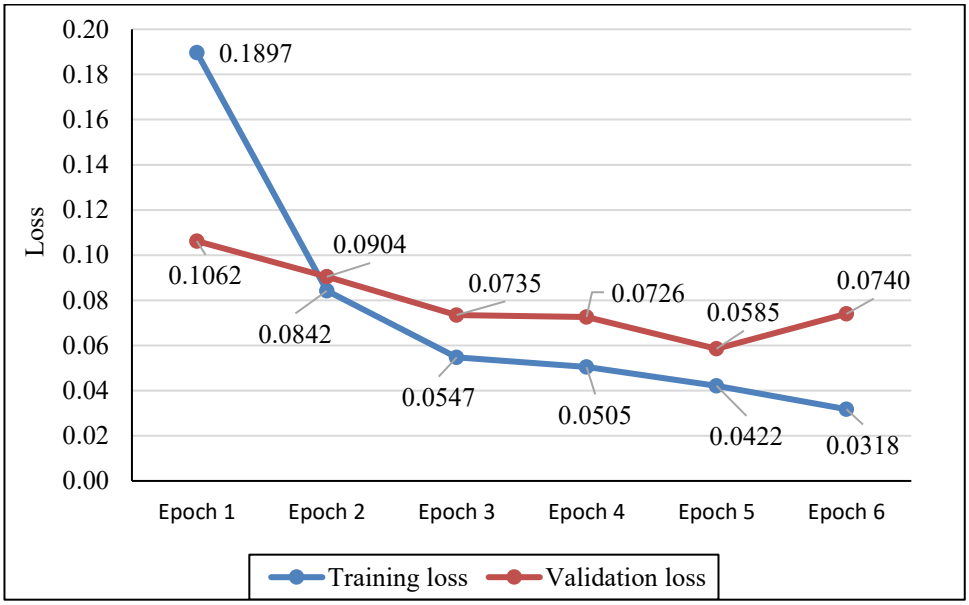


Figure 31. The training and validation loss of the Mawdoo3 dataset

We tried to train more than one experiment to gain the neural networks more chances to learn and improve the model performance. Figure 32 shows the accuracy of five runs for the original dataset with all data augmentation types. We noticed that the reflexive achieved higher accuracy than others at all the run trails.

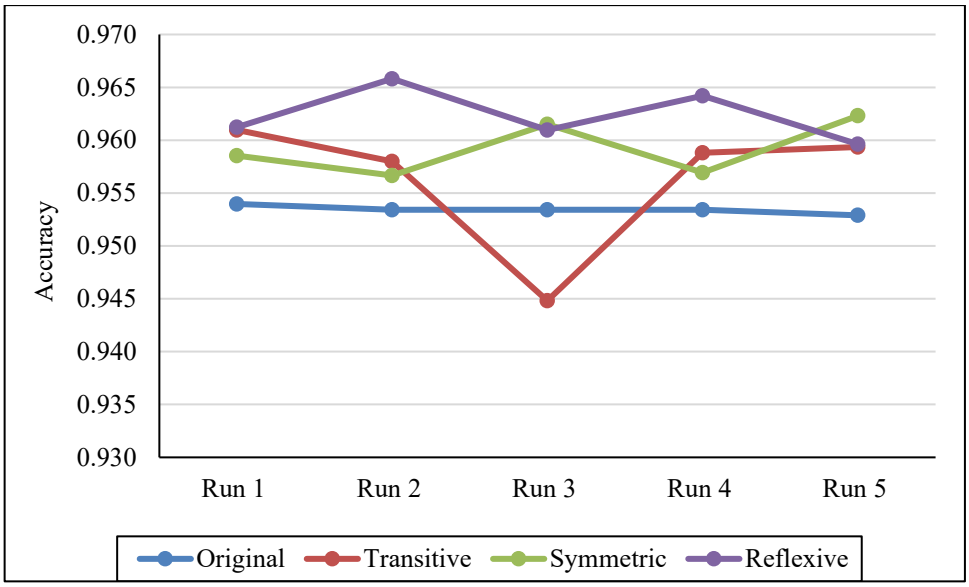


Figure 32. The accuracy of all data augmentation types on five runs

5.4.2 Training Mawdoo3 Dataset after Pre-training Model on the Back-Translation Dataset

In this section, we trained the original dataset on the AraBERT base v0.2, where the model is pre-trained on the back-translation dataset. Table 13 shows the accuracy for the first way (from Arabic to English) and the second way (from English to Arabic) on different runs.

Table 13. The accuracy of the pre-training model on the back-translation dataset

Epochs	Translation 1	Translation 2
Run 1	0.9407	0.9502
Run 2	0.9434	0.9418
Run 3	0.9375	0.9526
Run 4	0.9405	0.9388
Run 4	0.9351	0.9461
Avg Accuracy	0.9394	0.9459

Translation 2 describes the results for training on the questions that translated to English and back to Arabic. We noticed that the accuracy would be changed; may this be useful to train nurals more and more. So, we determined the average accuracy for all runs to decide which one is better than the other. The average accuracy of Translation 2 was 94.59% which means that they show a better result than the once translated ones (Translation 1).

After pre-training the model on the back-translation dataset, we trained the same model on the original dataset and all data augmentation types. Table 14 shows the accuracy of datasets. We notice that once the size of the dataset increased, then the accuracy will improve.

Table 14. The accuracy of the Mawdoo3 dataset and all data augmentation types after pre-training the model on the back-translation dataset

	Original	**Data augmentation**		
		Transitive	**Symmetric**	**Reflexive**
Accuracy	0.9526	0.9598	0.9631	0.9674

Figure 33 shows the training accuracy and validation accuracy for the data augmentation achieved high accuracy. We calculated the absolute accuracy for each data augmentation using the binary_accuracy method. For example, the accuracy for Reflexive is 96.74%.

We noticed that the training process stops after eight epochs pass without improvement based on the early stopping method. Figure 34 shows the training loss and validation loss where the monitor of early stopping for all experiments is val_loss.

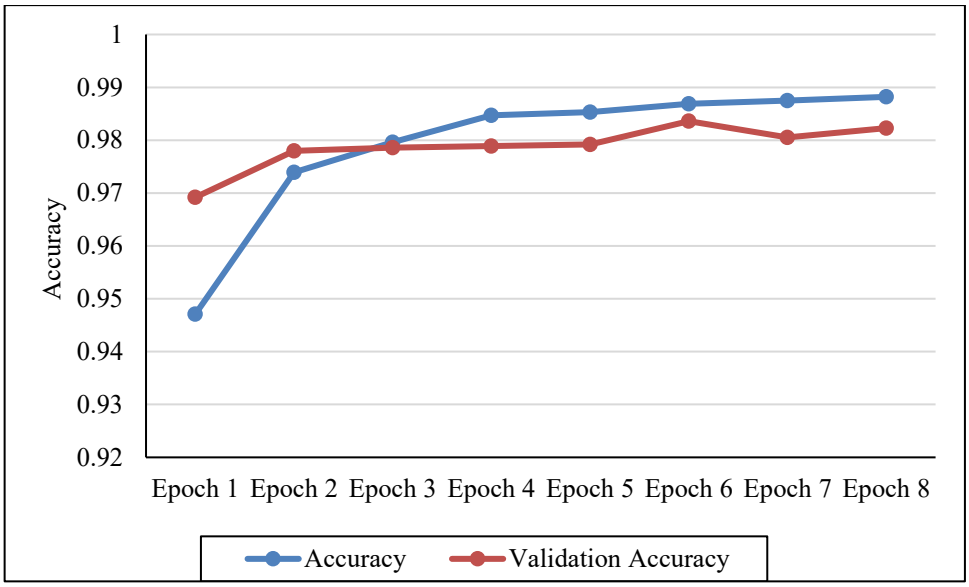


Figure 33. The accuracy of Mawdoo3 dataset after pre-training the model on the back-translation dataset

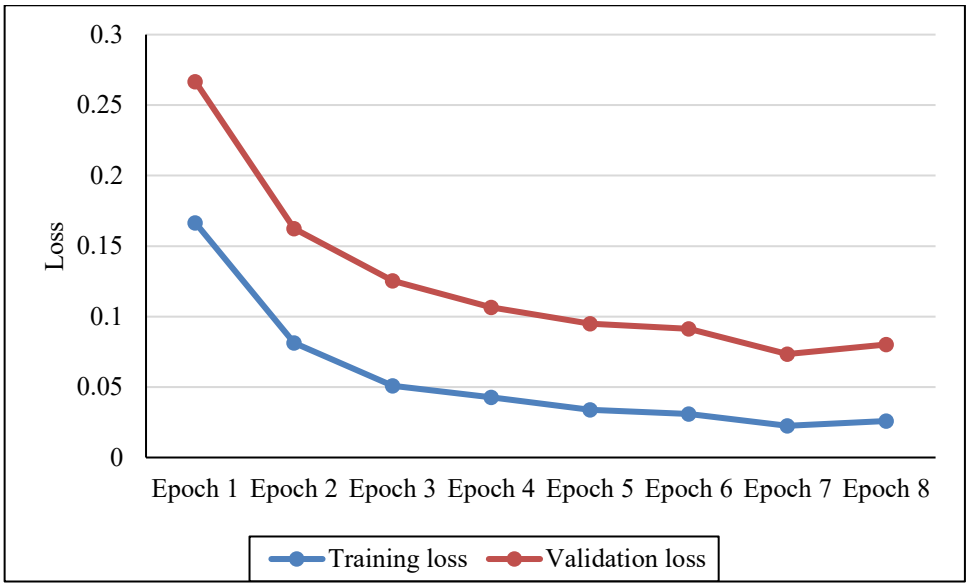


Figure 34. The training loss of Mawdoo3 dataset after pre-training the model on the back-translation dataset

5.4.3 Training on Mawdoo3 Dataset after Pre-training the Model on Translated Quora Question Pairs Dataset

Generally, in the translation process, we know that the meanings of words may be changed each time we want to translate either a question or a sentence.

The time is directly proportional to the text size. So, for example, when we translated the Quora dataset into chunks that consisted of 10,000 pairs of a question, the time consumed was roughly three hours. Then we used these chunks to pre-train the AraBERT model.

Table 14 shows model accuracy for the Mawdoo3 dataset after pre-trained the model on the translated Quora dataset. We noticed that the result did not improve the accuracy compared with the previous pre-trained dataset.

Table 15. The accuracy of the Mawdoo3 dataset and all data augmentation types after pre-training the model on the translated Quora dataset

	Original	**Data augmentation**		
		Transitive	**Symmetric**	**Reflexive**
Accuracy	0.9451	0.9510	0.9591	0.9639

Figure 35 shows the training accuracy and validation accuracy for the data augmentation achieved high accuracy (*i.e.,* Reflexive).

Figure 36 shows the training loss and validation loss for the highest accuracy (Reflexive). Based on the early stopping monitored by validation loss, at epoch 5 we noticed that the validation loss will be increased, and after that, there is no improvement.

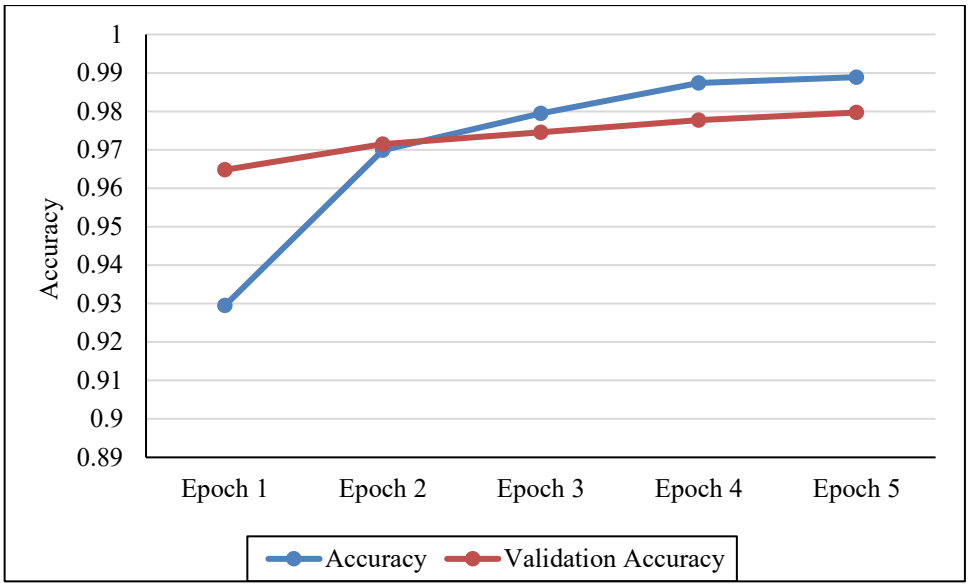


Figure 35.The training and validation accuracy for translated Quora question pairs dataset

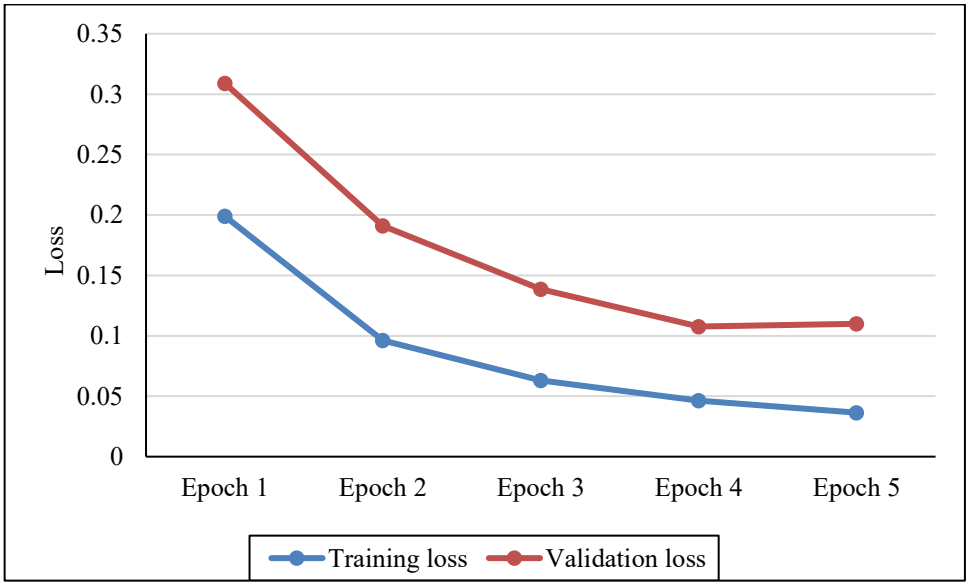


Figure 36.The training and validation loss for translated Quora question pairs dataset

5.4.4 Ensemble

The Ara-BERT model improves the results when enlarging the dataset. Therefore, we conducted other experiments with different random seeds to ensemble the AraBERT model. Hard voting is used as an ensemble method in which the predictions for each AraBERT experiment are involved in voting to get the final prediction.

Table 16 shows the output of the ensemble model from Ara-BERT v0.2 with different numbers of experiments each time with a different random seed. Thus, each experiment has different results at each time.

Because of using a voting ensemble, the result was slightly better than the previous experiments that we did. Moreover, we noticed that the improvement rate (original to reflexive) is relatively 0.54%.

Table 16. AraBERT ensemble result with five experiments

	Original	**Data augmentation**		
		Transitive	**Symmetric**	**Reflexive**
Experiment 1	0.9515	0.9512	0.9542	0.9582
Experiment 2	0.9502	0.9625	0.9549	0.9574
Experiment 3	0.9614	0.9548	0.9626	0.9652
Experiment 4	0.9588	0.9630	0.9648	0.9649
Experiment 5	0.9564	0.9595	0.9583	0.9661
Vote Accuracy	0.9619	0.9632	0.9656	0.9688

Generally, the Reflexive achieved the best result at different experiments. Thus, the approved result for our task is 96.88%.

5.5 Testing the Model

All previous sections illustrated variant techniques that improved the model's performance and decreased the error rate. A recent step is testing several Arabic question pairs to check the effectiveness of the model.

Table 17 shows the similarity and non-similarity scores for the number of examples. For example, the first question pair has 99.66% as a non-similarity score,

where the error rate for this question is 0.0034. Also, the similarity score of third question pairs was 98.95% which means that the error rate was 1.05%.

Table 17. Examples for making predictions with the model

No.	Question 1	Question 2	Label	Error rate	Score
1	ما هي عاصمة سنغافورا؟	متى أُحتلت سنغافورة من القوات اليابانية؟	NOT SIMILAR	0.338%	0.9966
2	بم يتميز تطبيق الفايبر؟	كيف أعمل تسجيل خروج من الفايبر؟	NOT SIMILAR	0.369%	0.9963
3	أين تقع قبرص في قارة أوروبا؟	ما هو موقع قبرص في قارة أوروبا؟	SIMILAR	1.05%	0.9895
4	كيف احضر كيكة الشوكولاتة والموز؟	من طرق تحضير كيكة الشوكولاتة والموز؟	SIMILAR	1.029%	0.9897
5	كم تبلغ نسبة انتشار المسيحية في جزيرة فيجي؟	كم تبلغ نسبة انتشار الإسلام في جزيرة فيجي؟	NOT SIMILAR	0.46%	0.9965

5.6 Discussion

We found that pre-training models like AraBERT v0.2 with more question pairs dataset yielded better results in all types of data augmentations, especially at Reflexive because the question pairs of the Mawdoo3 dataset increased to 45,514 records.

Figure 37 shows the growth in the accuracy for all experiments such as pre-trained the model on the mawdoo3 dataset, Back-translation dataset, and translated Quora dataset and then train the model with data augmentation types.

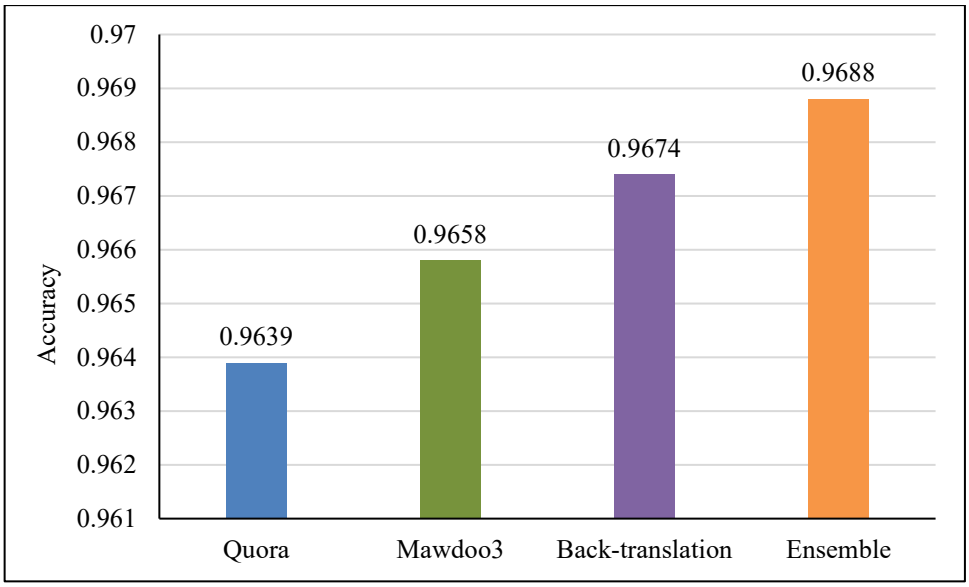


Figure 37. The final results of the accuarcy for all improvment stages

Some researchers worked on this task using different approaches and a unified dataset called Mawdoo3, but they achieved varying results. Table 18 summarized all previous solutions and compared them with our results. Also, Figure 38 shows the error rate among them. We minimize the error in the rate of 0.96 compared with the Inception team that proposed a BERT model as a solution for this task.

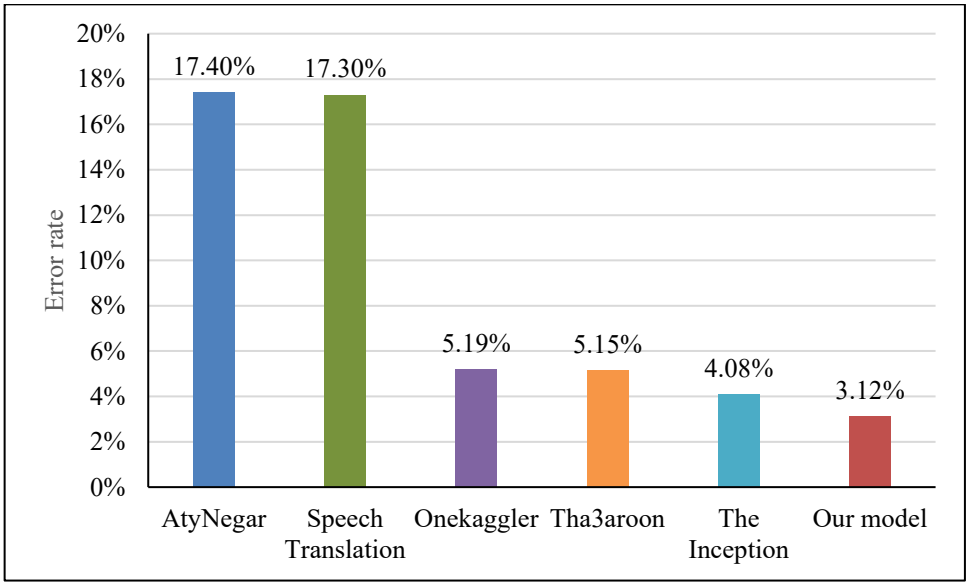


Figure 38. Comparison of error rates among all researchers who worked on this task

Table 18. Comparison with the results of trams participated in solving the task

No.	Team name	Approach	Dataset	Error rate
1	AtyNegar	1) Use SelectKBest Algorithm 2) SVM used for sentences classification	Mawdoo3	17.4%
2	Speech Translation	LSVM classifier	Mawdoo3	17.3%
3	Onekaggler	bidirectional GRU layers	Mawdoo3	5.19%
4	Tha3aroon	1) data augmentation techniques. 2)ELMo pre-trained	Mawdoo3	5.152%
5	The Inception	2) RNN, CNN, Multi-Head Attention, and Bidirectional Encoder Representations from Transformers (BERT) models	Mawdoo3	4.079%
6	Our model	1) Data augmentations 2) AraBERT v0.2	Mawdoo3 Back-translation Quora	3.12%

Chapter VI: Conclusions and Future Work

In this work, we address the question similarity for the Arabic language. Different datasets were pre-trained on the proposed model before training the main dataset. The model that is used for the task is AraBERT and an ensemble model of AraBERT where BERT was trained on a huge dataset and was ready to use with Arabic tasks. Also, they could use the finetuning BERT-Base Multilingual cased or uncased for dealing with Arabic tasks like this team (Al-Theiabat and Al-Sadi, 2019).

Our first main experiment is divided into two parts: 1) pre-training the model by using Mawdoo3 dataset, back-translation for Mawdoo3 dataset and translated Quora question pairs, 2) training the main dataset with the data augmentation. Based on these two parts, we achieved the best accuracy of 96.74% when enlarging the dataset by the reflexive method.

The second main experiment used the ensemble model that outperforms all other experiments on this task by achieving 96.88% accuracy with less error rate than previous work. This performance ranks first place among other teams.

Many factors contributed to these results, pre-trained the model on more datasets and an ensemble learning model that more data augmentations can boost.

In the future, to improve the result, we can investigate more techniques such as pre-training AraBERT on more datasets that only consist of Arabic question pairs with their labels. Moreover, the second technique is by using AraBERT fine-tuning with effective mechanisms. The second technique depends on the work used to improve BERT (i.e., Base and Large) fine-tuning via self-Ensemble and self-Distillation mechanisms without

any external data or knowledge. Also, their experiments on the text classification and NLI tasks achieved significant improvement.

References

Al-Theiabat, H., & Al-Sadi, A. (2019), The Inception Team at NSURL-2019 Task 8:Semantic Question Similarity in Arabic. **Proceedings of The First International Workshop on NLP Solutions for Under-Resourced Languages**.

Antoun, W., Baly, F., & Hajj, H (2021), AraBERT: Transformer-based Model for Arabic Language Understanding. **Proceedings of the Twelfth International Conference on Language Resources and Evaluation (LREC 2020)**, France.

Bdour, W. N., & Gharaibeh, N. K. (2013), Development of Yes/No Arabic Question Answering System. **International Journal of Artificial Intelligence & Applications (IJAIA)**, 4(1).

Brini, W., Ellouze, M., Mesfar, S., & Belguith, L. H. (2009), An Arabic Question-Answering system for Factoid Questions. **Proceedings of IEEE International Conference on Natural Language Processing and Knowledge Engineering**, China, pp. 1-7.

Bromley, J., Bentz, J. W., Bottou, L., Guyon, I., LeCun, Y., Moore, C., ... & Shah, R. (1993), Signature Verification using a "Siamese" Time Delay Neural Network. **International Journal of Pattern Recognition and Artificial Intelligence**, 7(04), 669-688.

Brownlee, Jason (2020), How to Develop Voting Ensembles with Python. Retrieved August 1st, 2021, from **https://machinelearningmastery.com/voting-ensembles-with-python/**.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014), Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. **Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP)**, pp. 1724–1734.

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019), BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. **ArXiv preprint arXiv:1810.04805**.

Ezzeldin, A. M., & Shaheen, M. (2012), A survey of Arabic Question Answering: challenges, tasks, approaches, tools, and future trends. **Proceedings of the 13th International Arab journal on Information Technology.** pp. 1-8.

Fadel, A., Tuffaha, I., & Al-Ayyoub, M. (2019), Tha3aroon Team at NSURL-2019 Task 8: Semantic Question Similarity in Arabic. **Proceedings of The First International Workshop on NLP Solutions for Under-Resourced Languages**.

Gomaa, W. H., & Fahmy, A. A. (2013), A Survey of Text Similarity Approaches. **International Journal of Computer Applications**, 68(13).

Graves, Alex (2012), **Supervised Sequence Labelling with Recurrent Neural Networks**. Berlin, Springer.

HuggingFace, Transformer: Preprocessing data . Retrieved May 20th, 2021, from **https://huggingface.co/transformers/preprocessing.html**.

HuggingFace, AraBERT v1 & v2: Pre-training BERT for Arabic Language Understanding. Retrieved June 7th, 2021, from **https://huggingface.co/aubmindlab/bert-base-arabertv02**.

I. Shankar, D. Nikhil, and C. Kornél (2017), First Quora Dataset Release: Question Pairs. Retrieved July 7th, 2021, from **https://quoradata.quora.com/First-Quora-Dataset-Release-Question-Pairs**.

Kenter, T., Borisov, A., & De Rijke, M (2016), Siamese CBOW: Optimizing Word Embeddings for Sentence Representations. **Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics**, Germany, pp. 941-951.

Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., & Fidler, S (2015), Skip-Thought Vectors. **Proceedings of Advances in Neural Information Processing Systems Conference**.

Le, Q., & Mikolov, T. (2014), Distributed Representations of Sentences and Documents. **Proceedings of the 31st International Conference on Machine Learning**, China, pp.1188-1196.

Li, Y., McLean, D., Bandar, Z. A., O'shea, J. D., & Crockett, K. (2006), Sentence Similarity based on semantic nets and corpus statistics. **IEEE Transactions on Knowledge and Data Engineering**, 18(8), 1138–1150.

Ling, W., Tsvetkov, Y., Amir, S., Fermandez, R., Dyer, C., Black, A. W., ... & Lin, C. C. (2015), Not all Contexts are Created Equal: Better Word Representations with Variable Attention. **Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing**, Portugal, pp. 1367–1372.

Liu, Y., Sun, C., Lin, L., & Wang, X. (2016), Learning Natural Language Inference using Bidirectional LSTM Model and Inner-Attention. **ArXiv preprint arXiv:1605.09090**, China.

Marie-Sainte, S. L., Alalyani, N., Alotaibi, S., Ghouzali, S., & Abunadi, I. (2018), Arabic Natural Language Processing and Machine Learning-Based Systems. **IEEE Access**, 7(99), 7011 –7020.

Mawdoo3, About us. Retrieved June 10, 2019, from **https://ai.mawdoo3.com/**.

Mohammad, A. S., Jaradat, Z., Mahmoud, A. A., & Jararweh, Y. (2017), Paraphrase Identification and Semantic Text Similarity Analysis in Arabic News Tweets using Lexical, Syntactic, and Semantic Features. **Elsevier**, 53(3), 640–652.

Mueller, J., & Thyagarajan, A. (2016), Siamese Recurrent Architectures for Learning Sentence Similarity. **Proceedings of Thirtieth AAAI Conference on Artificial Intelligence**, pp. 2786–2792.

Neculoiu, P., Versteegh, M., & Rotaru, M. (2016), Learning Text Similarity with Siamese Recurrent Networks. **Proceedings of the 1st Workshop on Representation Learning for NLP**, Germany, pp. 148–157.

Neptune.ai blog, Data Augmentation in NLP: Best Practices from a Kaggle Master. Retrieved July 26th, 2021, from **https://neptune.ai/blog/data-augmentation-nlp**.

NSURL website (2019), Task 8: Semantic Question Similarity in Arabic. **NLP Solutions for Under-Resourced Languages**.

Okazaki, N., Matsuo, Y., Matsumura, N., & Ishizuka, M. (2003), Sentence Extraction by Spreading Activation Through Sentence Similarity. **IEICE Transactions on Information and Systems**, E86D, pp. 1686–1694.

Schwab, D. (2017), Semantic Similarity of Arabic Sentences with Word Embeddings. **In Third Arabic Natural Language Processing Workshop**, France, pp. 18–24.

Seelawi, H., Mustafa, A., Al-Bataineh, H., Farhan, W., & Al-Natsheh, H. T. (2019), NSURL-2019 Shared Task 8: Semantic Question Similarity in Arabic. NLP Solutions for Under-Resourced Languages, **ArXiv preprint arXiv: 1909.09691v1**.

Shaheen, M., & Ezzeldin, A. M. (2014), Arabic Question Answering: System, Resources, Tools, and Future Trends. **Arabian Journal for Science and Engineering,** 39(6), 4541- 4564.

Sharifi, A. , Hassanpoor, H., & Maduyieh, N. Z.(2019), AtyNegar Team at NSURL-2019 Task 8: Semantic Question Similarity in Arabic. **Proceedings of The First International Workshop on NLP Solutions for Under-Resourced Languages**.

T. Darryl K. (2010), JetBrains Strikes Python Developers with PyCharm 1.0 IDE. Retrieved from **https://www.eweek.com/development/jetbrains-strikes-python-developers-with-pycharm-1.0-ide.**

Tai, K. S., Socher, R., & Manning, C. D. (2015), Improved Semantic Representations from Tree-Structured Long Short-Term Memory Networks. **Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing**, China, pp. 1556–1566.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017), Attention Is All You Need. **ArXiv preprint arXiv:1706.03762v5**.

Wali, W., & Gargouri, B. (2015), Supervised Learning to Measure the Semantic Similarity between Arabic Sentences. **In Computational Collective Intelligence**, 158–167.

Wikipedia, PyCharm Page, Last accessed August 1st, 2021, a. Retrieved from **https://en.wikipedia.org/wiki/PyCharm**.

Xu, Y., Qiu, X., Zhou, L., & Huang, X. Xuanjing (2020), Improving BERT Fine-Tuning *via* Self-Ensemble and Self-Distillation. **ArXiv preprint arXiv:2002.10345v1**.

Zhao, J., Zhu, T., & Lan, M. (2014), ECNU: One Stone Two Birds: Ensemble of Heterogeneous Measures for Semantic Relatedness and Textual Entailment. **Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)**, Ireland, pp. 271–277.

التشابه الدلالي للأسئلة العربية باستخدام حلول تعلم الآلة

إعداد
شروق محمود العواودة
المشرف
الأستاذ الدكتور غيث علي عبندة

ملخـــص

اللغة العربية هي اللغة الأم لأكثر من 400 مليون ناطق في جميع أنحاء العالم. اللغة العربية غنية لأنها يمكن أن تنقل معاني العبارات، ويمكن وصف كلمة واحدة بعدة طرق من المعنى والتعريف والتعبير.

حاليًا، هناك المزيد من التطبيقات التي تدعم اللغة الإنجليزية مقارنة باللغة العربية. لذلك يجب زيادة الاهتمام باللغة العربية لأنها تستخدم على نطاق واسع على وسائل التواصل الاجتماعي ومحركات البحث مثل Facebook Google engine. يستخدم العديد من الأشخاص محركات البحث للبحث عن أسئلة معينة في سياق مختلف للعثور على حل.

تم اقتراح عدة طرق لحل مشكلة التشابه الدلالي بين سؤالين أو جمل. سمحت التطورات الأخيرة في التعلم العميق وتوافر نماذج اللغة الإنجليزية والعربية المدربة مسبقًا بتدريب أنواع مختلفة من مجموعات البيانات باستخدام إجراء الضبط الدقيق. أبرز هذه النماذج هي المحولات ثنائية الاتجاه العميقة المدربة مسبقًا لفهم اللغة (BERT). تزيد هذه الأساليب من الدقة مع معدل خطأ منخفض وتحصل بسرعة على نتائج جديدة على أحدث طراز في العديد من مشكلات معالجة اللغة الطبيعية.

تهدف هذه الأطروحة إلى إيجاد نموذج يمكنه إيجاد تشابه السؤال الدلالي بين سؤالين عربيين بمعدل خطأ منخفض. مجموعة البيانات التي لدينا ليست كافية لتدريب نموذج ما قبل التدريب بكفاءة. لذلك، نعتمد أساليب لتوسيع مجموعة بيانات التدريب مثل زيادة البيانات (متعدية ، متماثلة ، انعكاسية).

تم استخدام مجموعة البيانات الأصلية في مسابقة تم تنظيمها في ورشة عمل حول حلول معالجة اللغة الطبيعية للغات قليلة الموارد (NSURL) في عام 2019. لذلك، في هذا المشروع، نحاول تحسين الدقة ومقارنة نتائجنا بالفرق الأخرى التي شاركت في هذه المسابقة.

أرشيف AraBERT v0.2 لأحدث النتائج. نستخدم الدقة لتقييم النتائج بعد تدريبها مسبقًا على مجموعات البيانات المعززة. أولاً، قمنا بتدريب النموذج مسبقًا على مجموعة بيانات Quora ، وكانت الدقة بعد التدريب مع جميع أنواع زيادة البيانات 96.39٪. بعد ذلك، مع مجموعة بيانات الترجمة الخلفية ، تم تحسين الدقة إلى 96.74٪. أخيرًا، تم تحسين الدقة من خلال مجموعة تصل إلى 96.88٪.