BIRZEIT UNIVERSITY

Faculty of Engineering and Technology

Electrical & Computer Engineering Department

Introduction to graduation project ENCS5200

Graduation project report

Approaches to semantic similarity in Arabic language

Prepared by:

Name: Mahmoud Nobani **Id:**1180729

Name: Ahmad Raddad **Id:**1180814

Name: Masoud Ajjouli **Id:**1181621

Instructor: Dr. Adnan Yahya

Supervisor: Dr. Wasel Ghannem

Section: 3

Date:24/7/2023

Abstract:

Studying Arabic semantic similarity involves analyzing the meaning and connotations of words and phrases in the Arabic language. This can be done for a lot of purposes, such as improving natural language processing (NLP) algorithms, developing language learning materials, conducting linguistic research, improve web search/information retrieval or plagiarism detection.

In this project we aim to research various semantic similarity algorithms in Arabic text and perform comparative study for them,

For our study focused on the new approach of transformers, mainly BERT and the different approaches derived like from it ARABERT, we tried utilize it to find the semantic similarity in Arabic text with the help of finetuning, other appraohes we utilized are RoBERTa based approaches which is an improved version of BERT and AraVec, the Arabic word2vec equivalent

To train and test these results, a special dataset was constructed and collected, while also using an external dataset from the internet.

The test conducted proved fruitful with a best result of 0.805 achieved by paraphrase-multilingual-mpnet a derived version of RoBERTa.

المستخلص:

ت تعتبر دراسة التشابه الدلالي العربي مهمة جدا، فهي تساعدنا في تحليل معنى ودلالات الكلمات، العبارات والنصوص في اللغة العربية، ويهدف القيام بذلك لمجموعة متنوعة من الأغراض، مثل تحسين خوارزميات معالجة اللغة الطبيعية، تطوير مواد تعلم اللغة، أو إجراء بحوثات لغوية، تحسين البحث على الويب / استرجاع المعلومات او كشف السرقات الادبية.

في هذا المشروع، نهدف إلى دراسة خوارزميات التشابه الدلالي المختلفة في النص العربي وإجراء بحث مقارن مدروس بينهم.

ولتحقيق هذا الهدف، سنحاول التركيز على الطريقة الجديدة لمعالجة اللغة الطبيعية ترانسفورمرز (المتحولات)، بالتحديد (بيرت) بالإضافة الي أدوات الذكاء الصناعي الماخودة منها ك ارابيرت و روبيرتيا .

ولدراسة وتدريب الالات الذكاء الاصطناعي, تم جمع مجموعة بيانات خاصة بنا بالإضافة الي استخدام ما هو متوافر على الانترنيت وبعد القيام بالتجارب, قدرنا على الوصول على نتيجة 0.805 سبيرمان كوريلاشن بفضل احد الطرق المستحدثة من روبيرتا

Contents

List of figures:

List of tables:

List of abbreviations:

NLP - natural language processing

BERT - Bidirectional Encoder Representations from transformers

NER - named entity recognition

QA - question answering

CBOW - continuous bag of words

GloVe - global vectors

LSA - Latent semantic analysis

ESA - explicit semantic analysis

SVD - singular value decomposition

RNN - Recurrent Neural Networks

LSTM - Long Short Term Memory

S.O.T.A - state-of-the-art

MLM - masked language modeling

NSP - next sentence prediction

MNLI - multi genre natural language interence dataset aim to test sentiment analysis

MRPC - Microsoft Research Paraphrase Corpus to test if paraphrase or not

STS-B - Semantic textual similarty benchmark to test semantic similarity

HARD - The Hotel Arabic Reviews Dataset (for sentiment analysis task)

ASTD - arabic sentiment twitter dataset

ArSenTD-Lev - The Arabic Sentiment Twitter Dataset for Levantine dialect

LABR - The Large-scale Arabic Book Reviews Dataset

AJGT - The Arabic Jordanian General Tweets dataset

RoBERTa - robustly optimized BERT approach

SNLI - Stanford natural language inference

MNLI - multi-genre natural language inference

ANLI - adversarial natural language inference which is a hard NLI dataset

XNLI - cross-lingual natural language inference

AraVec - Arabic vectors

DS1 - dataset we found on the internet

DS2 - dataset we collected

Chapter 1: Introduction and Motivation

1.1) Introduction:

Arabic semantic similarity approaches involve the study of how words and phrases in the Arabic language are related to one another in terms of meaning. These approaches are often utilized in natural language processing (NLP) and computational linguistics, as well as in other fields such as psychology and education. There are several different methods that are commonly used in Arabic semantic similarity research, including corpus-based approaches, word embedding methods, and lexical resource-based approaches [1]**.** One popular method that is currently being used in Arabic semantic similarity research is BERT or Bidirectional Encoder Representations from Transformers, which is a language model developed by Google. BERT has shown promising results in a plethora of natural languages processing tasks, like NER (named entity recognition), QA (question answering) and semantic similarity tasks, and has the capability to figure out complex relations between words and phrases in the Arabic language. Other methods that are commonly used in Arabic semantic similarity research include word embedding methods such as Word2Vec and GloVe, as well as lexical resource-based approaches such as the use of dictionaries and thesauri. These approaches are all useful for understanding how words and phrases in the Arabic language are related to one another in terms of meaning, and they provide valuable insights into the structure and use of the language [2]**.**

1.2) Motivation:

There are several reasons for studying Arabic semantic similarity, we can summarize them in the following points:

1) For researchers, it is an interesting and challenging problem that can lead to new insights and techniques for NLP. For administrators, improving the performance of NLP tools in Arabic can have practical benefits such as increased efficiency and cost savings for businesses and organizations that rely on translation services to communicate with Arabic-speaking populations. Finally, for users, improved NLP tools can lead to a better user experience and more accurate information, we summarize the motivation [3]**.**

2) There is a need for improved natural language processing (NLP) tools and techniques for the Arabic language, which is spoken by over 400 million person worldwide. The Arabic language has a unique structure and grammar, and developing NLP tools that can effectively handle these characteristics is essential for a numerous application, including information retrieval, machine translation, and text classification.
3) There is increasing demand for NLP applications in the Arabic-speaking world, where there is a growing need for automated language processing in areas such as social media analysis, customer service, and natural language generation. Improving the performance of NLP tools in Arabic can lead to more accurate and effective solutions for these tasks, which can have practical benefits such as increased efficiency and cost savings.
4) Studying Arabic semantic similarity can contribute to our understanding of the structure and meaning of the Arabic language and how it can be represented and processed by computers. This can lead to new insights and techniques for NLP that can be applied to other languages and domains.
5) Improving the performance of NLP tools in Arabic can have practical benefits such as increased efficiency and cost savings. For example, improved information retrieval systems can help users find the information they need more quickly, while improved machine translation systems can decrease the resources needed for translating documents form other languages. These benefits can be particularly important for administrators and organizations that rely on NLP tools in their daily operations.
6) Studying Arabic semantic similarity can have personal rewards such as the satisfaction of making a positive impact on the field of NLP and the Arabic-speaking community. It can also provide opportunities for collaboration and networking with other researchers and practitioners in the field.

1.3) Semantic Similarity:

Before we dive into the history and methods of calculating Semantic similarity, we need to understand what is semantic similarity, and why did we choose this type of similarity?

Semantic similarity: is a way for describing how similar two words or phrases meanings are to one another, It is typically determined by comparing the meanings of the words or phrases in a particular context, using some advanced model and NLP techniques [4]**.**

There are also other types of similarity such as:

Lexical similarity: This is a measure of the similarity between words or phrases based on their spelling and pronunciation. The semantics of the words or phrases are not considered, for example, the words "cat" and "bat" are lexically similar because they sound similar, but they have different meanings and would not be considered semantically similar [5]**.**

Structural similarity: This is a measure of the similarity between the structures or shapes of objects. It does not take into account the meanings of the objects or their functions. For example, a chair and a table might be structurally similar because they both have four legs, but they have different functions and would not be considered semantically similar [6]**.**

Functional similarity: This is a measure of the similarity between the functions or purposes of two things. It does not necessarily take into account the meanings or structures of the things. For example, a hammer and a screwdriver might be functionally similar because they are both tools used for construction, but they have different structures and might not be considered semantically similar.

Semantic relatedness: This is a measure of how two words are related to each other, It's kind of similar to semantic similarity, but it is not necessarily a measure of how similar the meanings are. Instead, it is a measure of how closely the meanings of the words or phrases are related, For example, the words "doctor" and "nurse" might be considered semantically similar because they both have to do with the medical field, but they are not identical in meaning. On the other hand, the words "doctor" and "patient" might be considered semantically related because they are both related to the medical field, but they are not necessarily similar in meaning.

So, semantic similarity is a particular type of similarity that focuses on the meanings of words or phrases, rather than their pronunciation, structure, or function.

1.4) Applications of Semantic similarity:

Semantic similarity has many applications that can benefit from it such as:

1.4.1) Biomedical fields:

In recent years semantic similarity has become more apparent in biomedical researches, Semantic similarity has emerged into a powerful tool for assessing the findings drawn from biomedical researches, this is done by finding the semantic similarity between the biomedical ontologies which provides a formal representation for the knowledge in biomedical domain [7]**.**

1.4.2) Plagiarism:

In academic and scientific study, plagiarism is a serious problem when someone presents someone else's work as their own. Traditional plagiarism detection techniques frequently rely on textual similarity tests, which can be readily gotten around by making minor changes or paraphrasing, Semantic similarity enable the comparison of texts based on their semantic content, syntactic structures, and relationships between concepts [8].

Chapter 2: Background

The concept of semantic similarity has a long history and has been studied in disciplines such as linguistics, psychology, and computer science. Aristotle, the Greek philosopher, was among the first to explore the relationships between words with similar meanings [9]. In the 19th and early 20th centuries, linguists and philosophers systematically studied word meanings and developed theories on word relationships. Charles Osgood's semantic differential scale in the 1950s was an early attempt to measure semantic similarity [10].

In Arabic linguistics, probably the first major contributions to the idea of semantic similarity was established by Islamic scholar (specifically Quran interpreters) and them trying to understand how similar words differs in different contexts.

Modern scholars, including linguist Noam Chomsky, have continued to contribute to the field, particularly in syntax, which is closely related to semantic similarity. Additionally, research in Arabic semantic similarity has advanced our understanding of the Arabic language's structure and meaning, leading to improved NLP techniques applicable to other languages and domains.

Historically, there has been a lot of methods that aim to find the semantic similarity and the semantic relations between word, Sentences and texts, and they have evolved every passing year.

In this section we will go on a journey through that progression:

2.1) Knowledge-based methods:

These methods attempt to determine how similar two items (e.g., words, phrases, or paragraphs) are by using external knowledge sources such as dictionaries, thesauri, and ontologies. These methods rely on the idea that the meaning of an item can be represented by its relationships to other items in the knowledge source [11]**.**

For example, consider the words "apple" and "orange." A knowledge-based semantic similarity method might compare the two words by looking at the relationships between them in a thesaurus. If the thesaurus includes the relationship "apple is a type of fruit" and "orange is a type of fruit," the method might conclude that the words are semantically similar because they are both types of fruit.

There are many different ways to implement knowledge-based techniques, and they can be utilized for a huge range of tasks such as natural language processing, information retrieval, and machine translations, few of the methods used in Knowledge-based semantic similarity are:

2.1.1) Edge-counting methods:

This method estimates the similarity between two items by counting the number of edges that connect them in a knowledge graph. A knowledge graph is a graph-based representation of a domain of knowledge, where the nodes represent concepts or entities and the edges represent relationships between them [12].

For example, suppose we have a knowledge graph representing the domain of fruit. In this graph, the nodes might represent specific types of fruit (e.g., "apple," "orange," "banana") and the edges might represent relationships such as "is a type of" or "is related to." To determine the semantic similarity between two fruits using an edge-counting method, we might count the number of edges that connect them in the graph. The more edges that connect the two fruits, the more similar they are considered to be.

2.1.2) Feature-based methods:

This method compares the similarity between two items by examining their shared features or attributes. These methods typically involve representing the meaning of an item as a collection of features or attributes, then comparing the degree to which the sets of features for the two objects under comparison overlap [11].

For example, consider the words "cat" and "dog." A feature-based method might represent the meanings of these words using a set of features such as "has fur," "has paws," "is a pet," etc. To find out the semantic similarity between the two words, the method would compare the overlap between the sets of features. If the sets have a lot of overlap (e.g., both words have the features "has fur" and "is a pet"), the method would conclude that the words are semantically similar. If the sets have little overlap (e.g., one word has the feature "can fly" while the other does not), the method would conclude that the words are not semantically similar [13].

2.1.3) Information content (IC)-based methods:

This method compares the similarity involving two items by measuring the volume of information that they contain. These methods typically rely on a pre-computed measure of the information content of each item in a knowledge base, and compare the information content of the two items [13].

One common way to compute the information content of an item is to use the frequency of the item in a large corpus of text. The idea is that more frequent items convey less information (because they are more predictable), while less frequent items convey more information (because they are less predictable). For example, a word like "the" is very common and conveys very little information, while a word like "quark" is less common and conveys more information.

To determine the similarity between two items using an IC-based method, we compare the information content of the items. If the information content of the items is similar (i.e., both items are equally frequent or infrequent), the method would conclude that the items are semantically similar. If the information content of the items is dissimilar (i.e., one item is more frequent than the other), the method would conclude that the items are not semantically similar.

But using frequency alone is not a reliable method to determine semantic similarity because frequency alone does not take into consideration the relative importance or meaning of the words being compared.

For example, two words might have the same frequency of occurrence in a corpus, but one word may have a much broader range of meanings than the other. In this case, the words would not be semantically similar even though their frequency of occurrence is the same.

To address this limitation, more sophisticated methods are used to determine semantic similarity, such as taking into account not only the frequency of occurrence but also adding an extra feature to find the information content of the words being compared, such as word co-occurrence, context, or syntactic structure.

2.2) Corpus-based methods

These are approaches to evaluating the semantic similarity between two items by analyzing the context in which the items appear in a large collection of text, also known as a corpus. These methods rely on the idea that the meaning of an item is determined by how it is used in a language, and that similar items will sometimes appear in similar contexts [14].

For example, look at the two words "apple" and "orange." A corpus-based semantic similarity method might compare the two words by analyzing the context in which they are found in a huge collection of text. If the method finds that the words often appear in similar contexts (e.g., both words are frequently used to describe types of fruit), it might conclude that the words are semantically similar. On the other hand, if the method finds that the words appear in very different contexts (e.g., one word is used to describe a type of fruit while the other is used to describe a color), it might conclude that the words are not semantically similar.

There are many different ways to implement corpus-based similarity methods, some of the methods used in Corpus-based semantic similarity are:

2.2.1) Word embeddings:

This method converts words (or phrases) from the vocabulary into vectors. The idea behind word embeddings is to figure out the meaning of the words by their contextual usage in a large corpus of text [15].

In word embedding models, each word is represented as a point in a high-dimensional space, and the position of the word in the space is determined by its statistical context within the corpus. Words that are used in similar contexts are positioned closer to each other, while words that are used in dissimilar contexts are positioned farther apart. This enables the model to figure the semantic relationships between words, such as synonymy (words that are synonyms are positioned close to one another in the space) and analogy (words that are related by an analogy, such as "man is to woman as king is to queen," are positioned in similar positions in the space).

Word embeddings are often used in natural language processing tasks such as language modeling, machine translation, and information retrieval, they are useful because they enable the system to encode word meaning in a continuous, numerical space, which makes it possible to perform mathematical operations on the representations (e.g., addition and subtraction). This makes it

possible to use word embeddings as the basis for more much more complex models such as neural networks.

And there are multiple types of word embedding techniques such as:

A. Word2vec:

Word2vec is a family of algorithms for learning word embeddings from a large corpus of text. It was designed by researchers at Google in 2013 and has since become one of the most famous techniques for learning word embeddings [1]**.**

There are two main variants of word2vec: continuous bag of words (CBOW) and skip-gram. CBOW predicts the current word given the context (i.e., the words around it), while skip-gram predicts the context given the current word. Both variants use a shallow neural network with a single hidden layer to learn the word embeddings, or sometimes they can use n-grams to capture the statical properties and pattern of a text, as the texts will be broken to n chunks of continues items and frequency will be calculated, and the embedding will be calculated.

Word2vec is particularly efficient at learning word embeddings from large corpora because it uses an efficient learning algorithm called negative sampling. This allows the model to learn high-quality word embeddings even from very large corpora.

Word2vec is used in a lot of NLP tasks like language modeling, machine translation, and information retrieval. It has also been used for a big range of other tasks such as detecting bias in language and identifying the authorship of texts [1]**.**

B. GloVe:

GloVe (short for "Global Vectors") is a corpus-based semantic representation that maps words (or phrases) from a vocabulary to dense vectors of real numbers. It was developed by researchers at Stanford University in 2014 as an alternative to the word2vec embedding [2]**.**

Like word2vec, GloVe represents each word as a point in a high-dimensional space, and the position of the word in the space is determined by its statistical context within a corpus of text. The key difference between GloVe and word2vec is the way in which the word embeddings are learned. While word2vec uses a shallow neural network to learn the embeddings, GloVe uses a

different approach called matrix factorization. This allows GloVe to learn word embeddings that are more interpretable, although it is less efficient than word2vec at learning from large corpora.

C. fastText*:*

fastText is a library for efficient learning of word embeddings and text classification. It was developed by researchers at Facebook in 2016 and has since become one of the most widely used tools for these tasks [16]**.**

One of the essential elements of fastText is its ability to learn high-quality word embeddings from very large corpora very quickly. This is possible through the use of techniques such as subword information and hierarchical softmax, which allow the model to learn meaningful representations of rare and out-of-vocabulary words.

In addition to learning word embeddings, fastText can also be used for text classification tasks such as sentiment analysis and spam detection. It does this by using the learned word embeddings as features in a simple linear classifier.

fastText is considered to be one of the best document similarity methods.

2.2.2) Latent Semantic Analysis (LSA):

LSA is a statistical technique for extracting and representing the underlying meaning of a collection of documents in a low-dimensional vector space. It is based on the idea that the meaning of a word is determined by the sentence/text it is in, and that similar words will tend to appear in similar contexts [17]**.**

In LSA, a document is represented as a vector of weights, where each weight corresponds to the importance of a particular word in the document. The weights are calculated using a matrix factorization technique known as singular value decomposition (SVD). The resulting vectors are called latent semantic vectors, as they capture the underlying meaning of the documents in a low-dimensional space.

LSA has been generally used in natural language processing tasks such as information retrieval, document classification, and machine translation. It has also been used for a huge range of other tasks like improving the accuracy of spell checkers and detecting plagiarism. One of the key advantages of LSA is that it is very efficient at learning latent semantic vectors from large corpora.

2.2.3) Explicit Semantic Analysis (ESA):

ESA is a method for representing the meaning of words and documents in a high-dimensional space. It is based on the notion that the meaning of a word can be represented by the concepts it is related to, and that similar words will tend to be related to similar concepts [18]**.**

In ESA, the concepts are represented as vectors in a high-dimensional space, and the words are represented as weighted combinations of these vectors. The weights are computed using a large encyclopedia or knowledge base, such as Wikipedia, which is used to define the relationships between words and concepts.

ESA is used in natural language processing tasks such as information retrieval, document classification, and machine translation. It has also been used for a variety other task including improving the accuracy of spell checkers and detecting plagiarism. One of the key limitations of ESA is the fact that it struggles with words with multiple meanings, and it being computationally intensive due to the potential high dimensionality for some vectors.

2.3) Neural Network-Based Semantic Similarity Methods:

The next step in the natural progression of semantic similarity methods is the usage of the deep learning tools and neural networks, what makes these methods special is the fact they are able to extract the features by themselves, without the help of the model designer, and in general Deep learning/ neural network-based methods most of the time perform better than traditional methods, so we will present some deep learning methods for semantic similarity:

2.3.1) LSTM networks:

LSTM is a variation of Recurrent Neural Networks (RNN), and while RNN has the ability to recall previous words to help capture its contexts, this produces an issue called the long-term dependency (where it may not be able to keep up with the full context of the sentences from one layer to another), LSTM was designed to able to overcame this issue, thus enabling LSTM to hold information's for longer periods of time across the model layers, what's impressive about LSTM is how versatile it is, here are some of its different variations:

- MaxLSTM-CNN: in 2018 a variation of LSTM combined with CNN called MaxLSTM-CNN was introduced to learn the embedding in order to find the similarity in sentences, this is done by first using CNN to learn the word embedding from a variety of pretrained word embedding (like word2vec and fastText), then using max-pooling scheme and LSTM to create the sentence representation, and lastly to measure the similarity a multi-level comparison which consists of a word vs word level, sentence vs sentence level and word vs sentence level was suggested by the author, figure 1 provides us with the overall structure of the mode [19]**.**

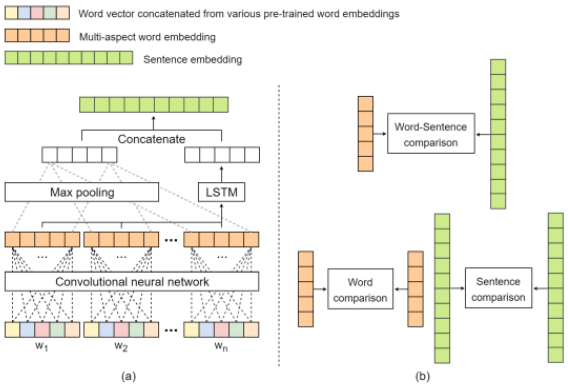

Figure 1: MaxLSTM-CNN model: (a) MaxLSTM-CNN embedding; (b) multiple-level comparison.

- Tree-LSTM: Another variation worth mentioning was introduced in 2015 called Tree-LSTM, which is a generalization from the normal LSTM, while the standard LSTM is a chain structure which constructs its hidden states from the input of the current time step and the previous one, Tree-LSTM is tree structured and thus constructs its hidden state of the input vector and hidden state of any number of child's, we can see the difference Cleary in figure 2 [20].

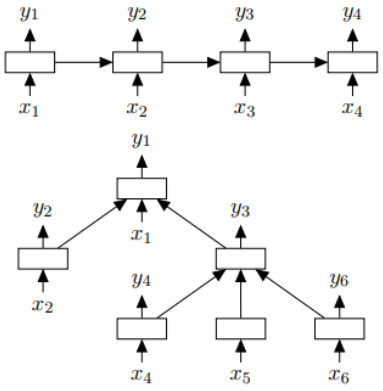

Figure 2: Top: A chain-structured LSTM network. Bottom: A tree-structured LSTM network

The Tree-LSTM is used produce a sentence representation for each sentence, which then are fed to a multi-layer neural network that computes the distance between the vectors and the angle to find the score for the similarity.

2.3.2) Transformer based models:

In general transformer-based models are the best kind of model to handle linguistics, Its introduction to the scene and the models built thanks to it are considered groundbreaking, and that thanks to its encoder decoder architecture and self-attention properties, on famous model that follow this architecture is BERT:

2.3.2.1) BERT:

BERT is a state-of-the-art (SOTA) deep learning model developed by google AI researchers, BERT in itself was created to be an improvement to other traditional NLP models which all are unidirectional [21].

To further understand the model Let's start by breaking down the term "BERT" and explaining its components, BERT is an acronym that stands for "Bidirectional Encoder Representations from Transformers." Each part of the acronym is represented as a key aspect of the model:

- **Bidirectional**:

  Bidirectional means that we incorporate the context of the sentences from both directions, whereas unidirectional means that we take the context from one direction (either right to left or left to right), which can produce inconsistency solving various NLP tasks, especially semantic similarity. This groundbreaking property was implemented by pretraining the model using MLM or (masked language modeling) task which randomly masks words from the inputs and using what's left the model should predict the original vocabulary, this enable of finding a representation the fuses both the left and right contexts [21]**.**

- **Encoder**:

  Unlike transformer architecture which contained both encoder and decoder (which can be very useful for translation purposes), BERT model only has an encoder. The encoder serves the purpose of getting the meaning behind the words/sentence, BERT achieves this by leveraging its self-attention mechanism, which allows it to examine other positions within the input sentence for contextual cues that aid in generating more informative encodings for each token. This self-attention property facilitates the establishment of connections between different words, as depicted in Figure 5. Once these connections are established, BERT employs the self-attention mechanism to compute the proximity or closeness between tokens, thereby capturing the interrelationships and semantic similarities among them.

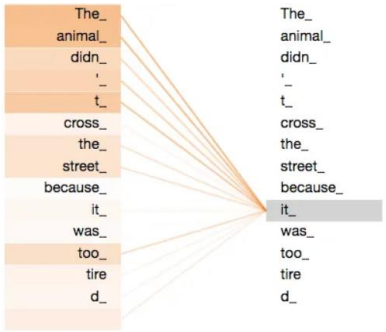

Figure 3: representation of the self-attention property where the darker the color, the stronger the connection.

- **Pretraining of BERT:**

BERT utilizes a two-step pretraining process to enhance its performance. In the initial step, BERT is pretrained on two distinct tasks. The first task, known as Masked Language Modeling (MLM), involves predicting the missing words within a sentence. By training on this task, BERT learns to understand the contextual relationships between words and effectively fill in the gaps. The second task is Next Sentence Prediction (NSP), which focuses on discerning the relationship between two consecutive sentences. By training on NSP, BERT gains the ability to comprehend sentence-level connections and improve its understanding of discourse. The parameters obtained from pre-training serve as the initial values for the subsequent fine-tuning, Fine-tuning involves training BERT on specific downstream tasks using labeled data specific to each task. This process enables BERT to adapt its knowledge to the nuances of the target task, allowing for more accurate predictions or classifications, this structure can be seen in figure 6.

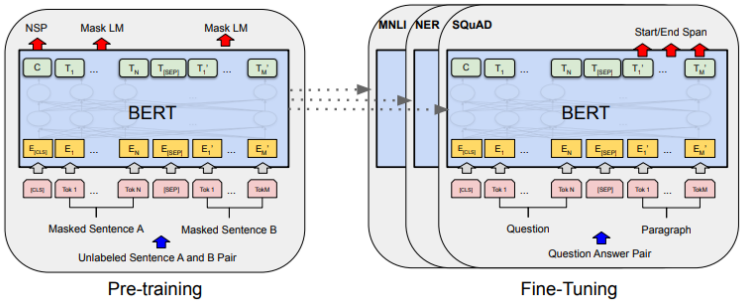

Figure 4: BERT construction for question answering task (example).

It's important to note that BERT was Trained of 16GB of data (books and Wikipedia articles)

- **Original BERT model specification:**

On release BERT had two models, each with its own unique specifications: [21]**.**

- **BERTbase:**
  - 12 encoder layers
  - 12 attention heads
  - 768 hidden units
  - ~110million parameter

- **BERTlarge:**
  - 24 encoder layers
  - 16 attention heads
  - 1024 hidden units
  - ~330 million parameters

The parameters mentioned above can be described:

The encoded layers represent the number of layers in the model, each layer has a number of attention heads (12 attention head for BERTbase or 16 for BERTlarge) where the hidden units (the sentences representation) are divided on them

- **BERT vs older models:**

BERT's utilization of transformers significantly enhances its ability to comprehend linguistic ambiguity and context. By analyzing a word in relation to other words within a sentence, both preceding and succeeding it, BERT gains a holistic understanding of the word's contextual surroundings, thereby unraveling its underlying meaning.

This stands in stark contrast to traditional shallow language processing models, such as word embeddings, where each word is converted into a vector representation that captures only a limited aspect of its true meaning, devoid of context. In contrast, BERT's contextual analysis empowers the model to grasp the intricate nuances of language, making it proficient in tackling more complex challenges, including QA [22].

We also would like to showcase the result the first iteration of BERT got on GLUE (General Language Understanding Evaluation) benchmark [21]**:**

System	MNLI-(m/mm)	STS-B	MRPC
number	392k	5.7k	**3.5k**
Pre-OpenAI SOTA	80.6/80.1	81.0	**86.0**
$BERT_{BASE}$	84.6/83.4	85.8	**88.9**
$BERT_{LARGE}$	**86.7/85.9**	**86.5**	**89.3**

Table 1: original BERT results on common GLUE tasks.

where MNLI (Multi-Genre Natural Language Inference) is a task that aims to predict the relationship between two sentences, determining whether they exhibit entailment, contradiction, or are neutral in meaning. It involves analyzing sentence pairs and making inference judgments

based on their semantic relationship. Please note that the "m/mm" abbreviation refers to the "matched/mismatched" split of the dataset. MNLI provides two separate evaluation sets: the matched set (m) and the mismatched set (mm).

MRPC (Microsoft Research Paraphrase Corpus) is a dataset comprising pairs of sentences collected from online news sources. Each pair is labeled by human annotators to indicate whether the sentences are semantically equivalent or not. The goal of MRPC is to assess the ability of models to identify paraphrases accurately.

STS-B (Semantic Textual Similarity Benchmark) is a widely used benchmark dataset for evaluating the performance of models in measuring semantic similarity between pairs of sentences. The dataset includes sentence pairs with assigned labels ranging from 0 to 5, indicating the degree of similarity between the sentences.

The result BERT achieved in table 1, incentivized us to focus more on it and the approach derived from it as it was able to achieve SOTA when it came surpassing the previous existing approaches (like the ones mention in the background)

- **Applications of BERT:**

The original BERT model was revolutionary, and thanks to that a lot of finetuned models spawned from it like [22]**:**

1) patentBERT - a BERT model fine-tuned to perform patent classification.
2) docBERT - a BERT model fine-tuned for document classification.
3) bioBERT - a pre-trained biomedical model for biomedical text mining.
4) VideoBERT - a joint visual-linguistic model for process unsupervised learning of an abundance of unlabeled data on YouTube.
5) SciBERT - a pretrained BERT model for scientific text
6) ARABERT: which aversion of BERT that was specifically train on Arabic tasks with Arabic dataset.

Chapter 3: Methodology:

Before going through the general approach, some concepts and model we will be using should be first explained:

3.1) Transfer learning:

Transfer learning is a machine learning technique where a model designed for a goal is reused as the starting point for another model that aims to perform a different but a close goal in essence we basically "transfer the knowledge" from the first model to the second model.

What is interesting about that is that in the case of ARABERT, it itself is a product of transfer learning of the original BERT model.

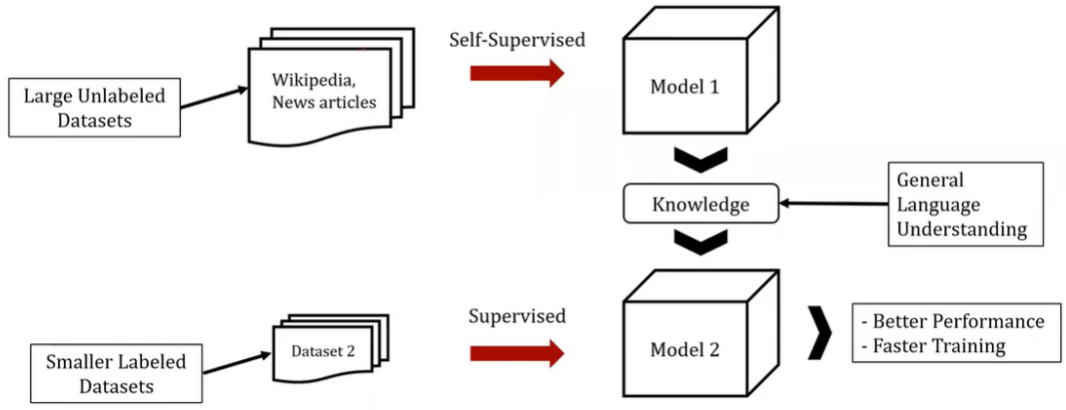

Figure 5:flowchart representing transfer learning method

We can see the Figure 5 basically describes the whole process we will go through, model1 represents ARABERT or another pretrained model which was trained using a very huge set of Arabic Wikipedia and news articles online, and with that existing general understanding of the Arabic language, we will **finetune** the model to match our needs using the dataset we collected; thus, we hope that we will get better results, faster training time and better performance.

Please note that finetuning involve taking a pretrained model and training the last layer in it, to match the task to aim to achieve, which is more computationally efficient and faster to train, with the added benefit of the already existing data in the previous layers.

3.2) Models used:

In this section we will describe the models we will use in our experiment.

3.2.1) Arabert:

ARABERT is a State-of-the-art arabic language model based on the BERT architecture, designed specifically for the Arabic language by three researchers from the American University of Beirut who are Wissam Antoun, Fady Baly, and Hazem Hajj in the year 2020

The ARABERT model was initially created from BERTbase model, having the same attributes while holding up to 512 maximum sequence length, the ARABERT model was trained on ~70m sentences collected from Wikipedia and news articles [23]**.**

3.2.1.1) ARABERT pretraining:

Same with the original BERT, ARABERT was first pretrain using MLM task with words instead of tokens, this forces the model to predict a full word rather than a part of it, also the model was pretrained on Next Sentence Prediction (NSP) which improves the model ability to figure out the relations between two sentences.

3.2.1.2) ARABERT Version:

There are two general versions of AraBERT with the difference being the how the tokenization work:

1) ARABERT v0.1/v1: Original, with wordpiece tokenization with the original words.
2) ARABERT v0.2/v2: better vocabulary, more data, more training, and the word are pre-segmented using Farasa Segmenter (will dive deeper on it in the next part).

3.2.1.3) Handling Arabic in ARABERT:

Handling Arabic in general is a bit tricky, so some changes had to be made before the training starts in the ARABERT model like:

First, they started by cleaning the data:

1) Removed diacritics
2) Removed elongations
3) Keep English characters

As mentioned above, ARABERT has mainly two versions, each model performed tokenization in a different way, the v1 model performed wordpiece tokenizer, which is the tokenizer supported by the original BERT, it works by dividing words either into the full forms (e.g., one word becomes one token) or into word pieces — where one word can be split into multiple tokens, depending on how frequent the tokens are.

Example:

Word	Token(s)
surf	['surf']
surfing	['surf', '##ing']
surfboarding	['surf', '##board', '##ing']
surfboard	['surf', '##board']
snowboard	['snow', '##board']
snowboarding	['snow', '##board', '##ing']
snow	['snow']
snowing	['snow', '##ing']

Figure 6: tokenization using wordpiece in ARABERT v1.

This method might work well with English, but it isn't efficient using Arabic, as it will make the vocabulary set for the model much larger.

Example:

("يكتب") = يكتب

("يكتب## , #س") = سيكتب

This made the word يكتب being repeated twice in the vocabulary file one time as

يكتب

And the second time as:

##يكتب

so instead, AraBERTv2 introduced the pre-segmentation using Farasa Segmenter (a java library dedicated to segmenting Arabic words) alongside the wordpiece tokenizer.

Example on pre-segmentation:

يكتب = ("يكتب")

سيكتبون = س+ , يكتب , +ون

The word يكتب will be repeated once, thus reducing the vocabulary size used, and thus increasing the unique words in vocabulary of the model.

3.2.1.4) The Performance of ARABERT:

ARABERT was tested on several tasks as shown in figure 8 [24].

Task	prev. SOTA	mBERT	AraBERTv1	AraBERTv2
HARD	95.7 ElJundi et.al.	95.7	**96.2**	96.1
ASTD	86.5 ElJundi et.al.	80.1	92.2	**92.6**
ArsenTD-Lev	52.4 ElJundi et.al.	51	58.9	**59.4**
AJGT	93 Dahou et.al.	83.6	93.1	**93.8**
LABR	**87.5** Dahou et.al.	83	85.9	86.7

Figure 7: ARABERT in accuracy on different tasks.

Where is **HARD**: The Hotel Arabic Reviews Dataset with Reviews are split into positive, negative reviews, neutral, which represents sentiment analysis task.

ASTD: The Arabic Sentiment Twitter Dataset, representing a sentiment analysis task.

ArSenTD-Lev: The Arabic Sentiment Twitter Dataset for Levantine dialect, considered to be a very hard dataset.

LABR: The Large-scale Arabic Book Reviews Dataset The reviews are rated between 1 and 5.

AJGT: The Arabic Jordanian General Tweets dataset with positive or negative rating.

We can see the ARABERT with its two-version achieved better results the previous SOTA in each task it was tested one except the last by less than 1% (still impressive)

3.2.2) RoBERTa:

RoBERTa or robustly optimized BERT approach is Another variant of BERT developed by Facebook AI researcher, using the same architecture of BERT but a more efficient training procedure [25].

3.2.2.1) RoBERTa pretraining:

This is the part where the Facebook AI researcher improved on BERT, as they concluded BERT was actually undertrained [26], and to face that, a new pretraining method was implemented [27]:

1) And increase on the dataset was seen (from 16GB with BERT to 160GB with RoBERTa)
2) RoBERTa focused on MLM pretraining and didn't train on NSP (like BERT) as they concluded that it didn't affect the later result a lot.
3) In MLM pretraining, a dynamic masking procedure was implemented, unlike BERT who used a static dynamic procedure (masking was done only in preprocessing), this reduced the number of duplications in the pretraining procedure (dynamic masking is done by duplicating the training data 10 times to make sure each sequence had a different pattern

- **RoBERTa results:**

System	MNLI-(m/mm)	STS-B	MRPC
RoBERTa	90.8/90.2	92.2	92.3

Table 2: RoBERTa results on the GLUE benchmark *[27]*.

From table 2, and if we compare it to table 1 that shows the result from BERT, we can see how much RoBERTa improved on BERT with these small changes on the pretraining procedure.

3.2.2.2) Multilingual RoBERTa:

These results intrigued us, but sadly there is no Arabic version of RoBERTa, so we will use the multilingual version of it, specifically, two finetuned model built from it:

1) xlm-roberta-base-snli-mnli-anli-xnli: a multilingual version of RoBERTa finetuned on SNLI (Stanford natural language inference), MNLI (multi-genre natural language inference), ANLI (adversarial natural language inference which is a hard NLI dataset) and lastly XNLI (cross-lingual natural language inference) [28].
2) paraphrase-multilingual-mpnet: a mapped sentences and paragraphs version of RoBERTa specifically trained on paraphrasing [29].

3.2.3) AraVec:

We wanted to also test another approach outside of BERT based approaches, so we decided to test AraVec (Arabic vectors), which an Arabic word embedding model developed by Abu Bakr Soliman, Kareem Eisa, and Samhaa R. El-Beltagy, from Nile University, Egypt [30].

3.2.3.1) Development of the model:

Similar to Word2Vec, AraVec was developed using N-gram modeling with either CBOW or SkipGram on a training set either collected from twitter of Wikipedia, this enabled the developer to produce the following versions of AraVec shown in figure 8:

Model	Docs No.	Vocabularies No.	Vec-Size
Twitter-CBOW	66,900,000	1,476,715	300
Twitter-CBOW	66,900,000	1,476,715	100
Twitter-SkipGram	66,900,000	1,476,715	300
Twitter-SkipGram	66,900,000	1,476,715	100
Wikipedia-CBOW	1,800,000	662,109	300
Wikipedia-CBOW	1,800,000	662,109	100
Wikipedia-SkipGram	1,800,000	662,109	300
Wikipedia-SkipGram	1,800,000	662,109	100

Figure 8: different versions of AraVec available.

Where Docs No. is the dataset, Vocabularies No. is the number of vocabularies deduced from the dataset and Vec-size is the length of embedding.

For our experiment we will use Twitter-CBOW with a Vec-Size of 300.

3.3) Datasets:

No experiment is possible without a Dataset, so we First, started by trying to find a dataset online that is similar to our desire, and luckily, we found one, a dataset with over 1400 pair of sentences, with ratings ranging from 0 to 5 [31].

Preview of the data:

	ترجمة العمود B إلى الفصحى	ترجمة العمود C إلى الفصحى
2.5	فتاة تصفف شعرها	فتاة تمشط شعرها
3.6	مجموعة من الرجال يلعبون كرة القدم على الشاطئ	مجموعة من الأولاد يلعبون كرة القدم على الشاطئ
5	امرأة تقيس كاحل امرأة أخرى	امرأة تقيس كاحل امرأة أخرى
4.2	رجل يقطع ثمرة خيار	رجل يقطع ثمرة خيار إلى شرائح
1.5	رجل يعزف على القيثارة.	رجل يعزف على بيانو الكتروني.
1.8	امرأة تقطع البصل.	امرأة تقطع التوفو
3.5	رجل يركب دراجة كهربائية.	رجل يركب دراجة هوائية.
2.2	رجل يقرع على الطبول.	رجل يعزف على الجيتار.
2.2	رجل يعزف على الجيتار.	سيدة تعزف على الجيتار.
1.714	رجل يعزف على الجيتار.	رجل يعزف على يوق.
1.714	رجل يعزف على الجيتار.	رجل يعزف على يوق.
5	رجل يقطع بصلة.	رجل يقطع بصلة.
0.6	رجل يركب دراجة هوائية.	رجل يتكلم.
4.4	رجل يشق بطن سمكة.	رجل يقطع سمكة.
2	رجل يقطع طماطم.	رجل يقطع كمكة.
1.8	رجل يعزف على الجيتار.	رجل يعزف على بيانو إلكتروني.
4.4	صغير البانداإ يسقط على منحدر	بندا يتزلق على منحدر
3.6	رجل يغني ويعزف على الجيتار.	رجل يعزف على الجيتار.
3.6	رجل يهاجم امرأة.	رجل يصفع امرأة.
1.2	رجل يقود سيارة.	رجل يركب حصانًا.
2.4	امرأة تقطع التوفو.	امرأة تقطع بصلة.
0.2	المرأة تصفف شعرها.	المرأة تقطع الأعشاب.
4.2	حماران وحشيان يلعبان في حقل مفتوح	حماران وحشيان يلعبان في أحد الحقول
4.4	رجل يقطع حبة بطاطس.	رجل يقطع بعض البطاطس.
2.25	رجل يقطع بصلة.	امرأة تقطع البقطين إلى شرائح.
2	رجل يرقص.	رجل وامرأة يرقصان.
0.75	رجل يركب دراجة نارية.	امرأة تركب حصانًا.
2.2	امرأة تقطع الثوم شرائح	امرأة تقطع بصلة شرائح
0.8	رجل يتكلم.	رجل يطبخ.
2.2	طفل صغير يغني ويعزف على الجيتار.	رجل يغني ويعزف على الجيتار.
3.2	سلحفاة تسبح في الماء.	سلحفاة تمشي تحت الماء.
4.8	امرأة شابة تضع ملصقات على وجهها بأكمله	امرأة تضع ملصقات على وجهها.
1.4	امرأة تغلف التوفو.	امرأة تكور العجينة
4.25	قطة تأكل بعض الذرة.	قطة تأكل الذرة الموجود على كوز الذرة
3.4	رجل يأكل طعاما.	رجل يأكل قطعة خبز.

Figure 9: sample of the online dataset.

We will call this Dataset from now on DS1.

Secondly, we started preparing our own data, each member in the team was tasked with gathering 250 pairs of sentences, and now have close to 800 pairs [32].

sample of the data:

S1	S2
الطائرة بدأت بالطيران	الطائر يطير في السماء
احمد يلعب البيانو	احمد يعزف الناي
محمود يعجن عجينة البيتزا	محمود يأكل البيتزا
انا احب ان العب الشطرنج	الشطرنج لعبة مملة
مسعود يرسم لوحة بالالوان المائية	اشترى مسعود لوحة مرسومة بالالوان المائية
رأيت مجموعة من الرجال تتحدث بجانب البنك	ذهبت الى البنك لتتحدث مع المدير
ليث يدخن السجائر	ليث يحب السجائر
لدي حساسية من القطط	لدي حساسية من الفول السوداني
وقعت العصا على الارض	ضرب الاستاذ الطالب بالعصى ثم القاها على الارض
ذهب الى الحديقة لرؤية الاسد	رأيت اسد في الجامعة
سلم احمد على عبدالله	سلم احمد الكرة لعبدالله
لبنان تقع شمال فلسطين	لا يجب ان تأكل الطعام بشمالك
حطت الطائرة في المطار	هبطت الطائرة في المطار
تكسر الصحن	حطمت الفتاة الصحن عندما وقعت

Figure 10: sample of our dataset.

For our rating range, we decided to stick with a range of 0 to 5

where 0 indicates close to 0% similarity

where 1 indicates close to 20% similarity

where 2 indicates close to 40% similarity

where 3 indicates close to 60% similarity

where 4 indicates close to 80% similarity

where 5 indicates close to 100% similarity

to make sure our dataset is usable, inter-judge agreement calculation was performed, in two steps:

1) Percent agreement with the aim of finding the different combination of each pair
2) Interclass correlation which represent the agreement for all the rating.

To make things simpler to calculate we mapped the scores for the dataset to 3 different levels:

1) Level 0 (close to not similar): sentences with a score of 0 or 1
2) Level 1 (in between): sentences with a score of 2 or 3
3) Level 2 (close to similar): sentences with a score of 4 or 5

This will help us getting better results with the percent agreement.

S1	S2	judge A ratings	basher rating	karam ratings	judge A range reduction	judge B range reduction	judge C range reduction	A/B	A/C	B/C	interjudge agreement
الطائره بدأت بالطيران	الطائر يطير في السماء	1	3	1	0	1	0	0	1	0	1
احمد يلعب البيانو	احمد يعزف الناي	2	3	3	1	1	1	1	1	1	2
محمود يحجن عجينة البيتزا	محمود يأكل البيتزا	3	4	3	1	2	1	0	1	0	1
انا احب ان العب الضفدع	الضفدع لجة مملة	1	1	2	0	0	1	1	0	0	1
مسعود يرسم لوحة بالالوان المائية	اشترى مسعود لوحة مرسومة بالالوان المائية	4	3	4	2	1	2	0	1	0	1
رأيت مجموعة من الرجال تحدث بجانب البنك	ذهبت الى البنك لتحدث مع المدير	1	2	2	0	1	1	0	0	1	1
ليث يدخن السجائر	ليث يحب السجائر	5	4	4	2	2	2	1	1	1	2
لدي حساسية من القطط	لدي حساسية من القول السوداني	2	2	3	1	1	1	1	1	1	2
وقعت العصا على الارض	ضرب الاستاذ الطالب بالعصى ثم القاها على الارض	3	1	2	1	0	1	0	1	0	1
ذهب الى الحديقة لرؤية الاسد	رأيت اسد في الحديقة	0	1	3	0	0	1	1	0	0	1
سلم احمد على عدالله	سلم احمد الكره لعبدالله	1	1	3	0	0	1	1	0	0	1
لبنان تقع شمال فلسطين	لا يجب ان تأكل الطعام بشمالك	0	0	0	0	0	0	1	1	1	2
حملت الطائره في المطار	هبطت الطائره في المطار	5	4	5	2	2	2	1	1	1	2
تكسر الصحن	حطمت القطاه الصحن عندما وقعت	5	2	4	2	1	2	0	1	0	1
لدى العرب ضغينة تجاه العرب	حقد العرب على العرب بعد الهزيمة	5	3	4	2	1	2	0	1	0	1
لقد الولد عقله حماله	حل الطالب المسالة	0	1	2	0	0	1	1	0	0	1
اكل محمد هو اخته	أرسل الرسول محمد بالحق من ربه	0	0	0	0	0	0	1	1	1	2
صان الصحابة القرآن	حفظ الصحابة القرآن	5	3	5	2	1	2	0	1	0	1
فسر الطالب المشكلة لزميله	حلل المهندس المشكلة	1	2	4	0	1	2	0	0	0	0
رمى لاعب كره السلة الكره	رمى الولد القمامة	0	0	2	0	0	1	1	0	0	1
بدأ المذاع عن افضل الاطعمة	اعطى لك الغذاء العامل	0	2	1	0	1	0	0	1	0	1
توسع العداء	توت الوسع العداء	5	3	5	2	1	2	0	1	0	1
رسم المهندس المبنى	صمم المهندس العماره	4	3	5	2	1	2	0	1	0	1
اسبانيا لديها طعام ممتاز	اكلت الطعام الفاخر في اسبانيا	1	3	4	0	1	2	0	0	0	0
وصل الاسلام لكل اماكن الجزيره العربية	رقعة الفتوحات الاسلامية كان كبيره وواسعة	4	3	4	2	1	2	0	1	0	1
العصر الحضاري امتاز بالتقدم العلمي	عهد الحضاريين امتاز بالتطور العلمي	5	4	5	2	2	2	1	1	1	2
يتصح الناجحون بقراءه الكتب	يتصح الناجحون بمطالعه الكتب	5	5	5	2	2	2	1	1	1	2
لم يحقق الطالب الجامعي إنجاز كثيره	انجازات الطالب الجامعي إنجاز كثيره	4	4	4	2	2	2	1	1	1	2
ركب الفارس على مرج الحصان	صعد الفارس على ظهر الجبل	5	4	4	2	2	2	1	1	1	2
اخبرني اهي عن مبارزه البارحة	حدثني اهي عن مبارزه البارحة	5	5	5	2	2	2	1	1	1	2
اعتذ العمال من التاجر	ذهبت العمال الى التاجر	2	2	2	1	1	1	1	1	1	2
ودعني ابي قبل ذهابه	تركني ابي في البيت وحيدا	2	2	2	1	1	1	1	1	1	2

Figure 11: sample of the data after performing percent agreement.

The summary of the results of the tests where: 14 pair that required changing (with 0 judges agreed), 361 pair with two judge agreement and lastly 403 pair were agreed on by three judges, these results are promising proving the usability of our dataset.

After that Interclass correlation was performed to make the agreement for all the ratings and we got the following results:

Judges	Correlation
A, B, C	0.78
A, B	0.78
B, C	0.76
A, C	0.8

Table 3: interclass correlation between judges.

The outcome we got indicates that the dataset we have is sound and suitable for training, as it has a higher correlation percentage then 0.66 (if only two judges agreed out of three) we will call it from now on DS2.

3.4) Preprocessing:

Before testing can start, we should first preprocess our data and for that we followed these steps:

1) Filtering non-arabic content.
2) Normalization, and by that, I mean letters like ة replaced with ه, and letters like أ آ replaced with ا and so on.
3) Removed tashkeel.
4) Follow the models' guidelines for preprocessing (if exist).
5) And this is the most important step, using an NER model to replace names with محمد , to perform NER we will be using a finetuned version BERT to identify a person named entities, this model had a F1 score of 87% [33].

3.5) Evaluation criteria:

For our criteria of evaluation and after a lot to studying, we decided to use the following metric:

Metric	Formula
spearman correlation	$\frac{6 * \sum d^2}{n(n^2-1)}$

Table 4: evaluation metrics.

Where n is the number of the pair of sentences, and d is the squared difference between two points (score of similarity), what makes spearman correlation the best metric for our task is the fact that it's good for assessing relationships with non-linear patterns and is less sensitive to outliers compared to other correlation measures like Pearson correlation.

3.6) General approach:

Our general approach for this experiment is as follow:

1) Preprocess the dataset following the instruction explained above
2) Fine tune the pretrained model on the given dataset (by training the last layer to fit our task)
3) Use the model to produce embedding or vector representing the input data
4) Calculate the similarity between the two vectors by using cosine similarity.
5) Repeat the same experiment with NER added to the preprocessing step

3.7) Current S.O.T.A:

For our task the current S.O.T.A results are 0.81 spearman correlation on DS1, produced by a model that takes an input sentences, compute multiple embedding for that sentence and compares them in a Siamese architecture model, where this procedure is guided by manual estimation scores to find the best embedding then comparing these embedding using cosine similarity [31].

Chapter4: Experiments and results

4.1) Results:

model	DS1	DS2
paraphrase-multilingual-mpnet	0.80	0.66
Arabert v1	0.58	0.60
Arabert v2	0.52	0.54
xlm-roberta-base-snli-mnli-anli-xnli	0.64	0.58
AraVec	0.44	0.38
paraphrase-multilingual-mpnet + NER	0.805	0.65
Arabert v1 + NER	0.58	0.59
Arabert v2 + NER	0.523	0.52
xlm-roberta-base-snli-mnli-anli-xnli + NER	0.641	0.57
AraVec + NER	0.431	0.35

Table 5: experiments results.

Looking at table 5 we can see that paraphrase-multilingual-mpnet was the best performing model by far on DS1 with a correlation result of 0.8 without NER and 0.805 with NER, while AraVec is by far the worst performing model showing the limitations of the typical NLP models compared to deep learning approaches, what's interesting is that ARABERT performed way worse than we expected (with V1 doing better than V2), and we may contribute it to a problem with the finetuning or the fact that it follows the guideline of the typical BERT model unlike paraphrase-multilingual-mpnet which follow RoBERTa, another interesting observation is the fact that all the models performed better on DS1 compared to DS2 which is a harder dataset as it has a wider range plus a discrete rating (which reduced the overall correlation), lastly we can see that an increase in correlation when using NER in DS1 even if it's a small margin, while we see a decrease when using NER in DS2 and we can contribute that to the fact that DS2 didn't have any named entities (we avoided them) and the identified ones contributed in the decrease of correlation, after our

various trails we couldn't pass the S.O.T.A results, but we were able to shown a considerable improvement using NER.

Chapter 5: Conclusion and future work

This project aimed to study and research the various approaches possible to calculate the semantic similarity in Arabic language, we first started by understanding the concept of semantic similarity and comparing it the various types of similarity available, then we researched the progression of semantic similarity estimation method, starting with the more traditional approaches like knowledge based and corpus based approaches, then made our way to the more recent and popular approaches, especially transformers based ones, with BERT being the leading model.

And after researching BERT, we noticed that it achieved on release a S.O.T.A on the glue benchmark, which prompt us to focus on it and on the approaches derived from it like ARABERT and RoBERTa.

And with that in mind, the search for a dataset started with us finding a dataset online DS1 and collecting out own dataset with and interjudge agreement of 0.8, and using these dataset we started our experimenting on the various model we decided to use, with paraphrase-multilingual-mpnet proving to be the best model for our task with 0.8 spearman correlation on DS1 and after adding NER to it a raise was noticed in the correlation with 0.805 spearman score, the idea of NER proved to be usefull in some cases, but not with DS2 as it didn't have any person name entity.

For our future work, we aim to try using NER with the current S.O.T.A to see if an improvement can be made to the model overall results, and we also see the need for a new embedding tool for Arabic languages so pretraining a more modern model like GPT-4 on Arabic specific task (not in a multilingual fashion) is a good option.

References

[1] T. Mikolov, K. Chen, G. Corrado and J. Dean, *Efficient Estimation of Word Representations in Vector Space,* arXiv, 2013.

[2] J. Pennington, R. Socher and C. Manning, "GloVe: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, 2014.

[3] H. Lahlou, "The Motivations for the Semantic Change in the Category Green in Arabic: A Synchronic and Diachronic Corpus-Based Analysis," *Open Journal of Social Sciences,* vol. 8, pp. 18-28, 2020.

[4] F. Couto and A. Lamurias, "Semantic Similarity Definition," 2019, p. 870–876.

[5] N. Pradhan, M. Gyanchandani and R. WadhvanI, "A Review on Text Similarity Technique used in IR and its Application," *International Journal of Computer Applications,* vol. 120, pp. 29-34, June 2015.

[6] R.L. Goldstone, A. Kersten and P. Carvalho, "Concepts and categorization.," 2013.

[7] C. Pesquita, D. Faria, A. Falcão, P. Lord and F. Couto, "Semantic Similarity in Biomedical Ontologies.," *PLoS Comput. Biol.,* vol. 5, 2009.

[8] T. Vrbanec and A. Mestrović, "The struggle with academic plagiarism: Approaches based on semantic similarity," *2017 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO),* pp. 870-875, 2017.

[9] J. Bradac, "Aristotle's semantic theory and some implications for research," *Review of Communication,* vol. 3, no. 1, pp. 41-52, (2003).

[10] C.E. Osgood, G. Suci and P.H Tannenbaum, The Measurement of Meaning, University of Illinois Press, 1957.

[11] D. Sánchez, M. Batet, D. Isern and A. Valls, "Ontology-based semantic similarity: A new feature-based approach," *Expert Systems with Applications,* vol. 39, pp. 7718-7728, March 2012.

[12] R . Rada, M. Roy Hafedh, E. Bicknell and M. Blettner, "Development and application of a metric on semantic nets," *Systems, Man and Cybernetics, IEEE Transactions on,* vol. 19, pp. 17-30, February 1989.

[13] D. Sáncheza and M. Batet, "A semantic similarity method based on information content exploiting multiple ontologies," *Expert Systems with Applications,* vol. 40, pp. 1393-1399, March 2013.

[14] J. Gorman and J. Curran, "Scaling Distributional Similarity to Large Corpora," in *Annual Meeting of the Association for Computational Linguistics*, 2006.

[15] T. Schnabel, I. Labutov, D. Mimno and T. Joachims, "Evaluation methods for unsupervised word embeddings," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lisbon, 2015.

[16] P. Bojanowski, E. Grave, A. Joulin and T. Mikolov, "Enriching Word Vectors with Subword Information," *Transactions of the Association for Computational Linguistics,* vol. 5, p. 135–146, 2017.

[17] T.K. Landauer and T.S. Dumais, "A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge," *Psychological Review,* vol. 104, p. 211–240, 1997.

[18] E. Gabrilovich and SH. Markovitch, "Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis," 2007.

[19] N. Tien, M. Le, Y. Tomohiro and I. Tatsuya, "Sentence Modeling via Multiple Word Embeddings and Multi-level Comparison for Semantic Textual Similarity," 2018.

[20] K. Sheng Tai, R. Socher and C.D. Manning, *Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks,* arXiv, 2015.

[21] J. Devlin, M. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, 2019.

[22] B. Lutkevich, "BERT language model," January 2020. [Online]. Available: https://www.techtarget.com/searchenterpriseai/definition/BERT-language-model#:~:text=BERT%20is%20an%20open%20source,surrounding%20text%20to%20establish%20context. [Accessed 27 2 2023].

[23] W. Antoun, F. Baly and H. Hajj, "AraBERT: Transformer-based Model for Arabic Language Understanding," in *Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection*, Marseille, 2020.

[24] W. Antoun, F. Baly and H. Hajj, "AraBERT : Pre-training BERT for Arabic Language Understanding," 28 Feb 2020. [Online]. Available: https://sites.aub.edu.lb/mindlab/2020/02/28/arabert-pre-training-bert-for-arabic-language-understanding/#:~:text=AraBERT%20is%20an%20Arabic%20pretrained,versions%20of%20the%20model%20AraBERTv0. [Accessed 27 2 2023].

[25] pawangfg, "Overview of ROBERTa model," geeksforgeeks, [Online]. Available: https://www.geeksforgeeks.org/overview-of-roberta-model/.

[26] D. Sharma, "A Gentle Introduction to RoBERTa," Analytics Vidhya, 22 oct 2022. [Online]. Available: https://www.analyticsvidhya.com/blog/2022/10/a-gentle-introduction-to-roberta/.

[27] Y. Stoyanov, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer and Veselin, *RoBERTa: A Robustly Optimized BERT Pretraining Approach,* arXiv.org, 2019.

[28] "symanto/sn-xlm-roberta-base-snli-mnli-anli-xnli · Hugging Face," huggingface.co, [Online]. Available: https://huggingface.co/symanto/sn-xlm-roberta-base-snli-mnli-anli-xnli.

[29] "sentence-transformers/paraphrase-multilingual-mpnet-base-v2," huggingface.co, [Online]. Available: https://huggingface.co/sentence-transformers/paraphrase-multilingual-mpnet-base-v2.

[30] A. Soliman, K. Eisa and S. R. El-Beltagy, "AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP," in *in proceedings of the 3rd International Conference on Arabic Computational Linguistics*, dubai, 2017.

[31] M. Sulaiman, A. Moussa, S. Abdou, H. Elgibreen, M. Faisal and M. Rashwan, "Semantic textual similarity for modern standard and dialectal Arabic using transfer learning," *PLOS ONE,* vol. 17, pp. 1-14, August 2022.

[32] M. Nobani, A. Raddad and M. Ajjouli, "Dataset," bzu, 28 2 2023. [Online]. Available: https://docs.google.com/spreadsheets/d/1_6aQ96t-H-qJ82h4PCiOGxHanRVzBYgqJuJFsDelWiA/edit?usp=sharing. [Accessed 28 2 2023].

[33] "hatmimoha/arabic-ner," huggingface.co, [Online]. Available: https://huggingface.co/hatmimoha/arabic-ner. [Accessed 22 7 2023].

[34] N. Tien, M. Le, Y. Tomohiro and I. Tatsuya, "Sentence Modeling via Multiple Word Embeddings and Multi-level Comparison for Semantic Textual Similarity," 2018.