EXPLORING STRATEGIES FOR MEASURING SEMANTIC SIMILARITY IN SHORT ARABIC TEXTS

Mohamed Abd-Elnabi I. I. Gabr
Computer and Systems Engineering Department, Faculty of Engineering, Ain Shams University, Cairo, Egypt.
1902186@eng.asu.edu.eg

Ahmed Z. Badr
Computer and Systems Engineering Department, Faculty of Engineering, Ain Shams University, Cairo, Egypt.
ahmed.z.badr@eng.asu.edu.eg

Hani M. K. Mahdi
Computer and Systems Engineering Department, Faculty of Engineering, Ain Shams University, Cairo, Egypt.
hani.mahdi@eng.asu.edu.eg

Abstract

Measuring the semantic similarity of short Arabic texts is a challenging task due to the distinct linguistic characteristics of Arabic and the scarcity of resources for natural Arabic language processing. In this study, we investigate different strategies to accurately assess the semantic similarity between brief Arabic texts found in public domains and question-answering systems. We explore recent embedding techniques in NLP and propose deep learning approaches to understand the text's semantic meaning. However, analyzing short texts can be difficult because they typically contain terse sentences. Despite these challenges, measuring semantic similarity is crucial for various applications such as information retrieval, plagiarism detection, information extraction, and machine translation. Our study sheds light on the challenges and potential solutions for measuring semantic similarity in short Arabic texts.

***Keywords*: Deep learning; Natural Language Processing (NLP); Similarity Measure; Transformers; Arabic Text Similarity.**

1. Introduction

Arabic was the fourth most popular language on the web in 2022, with 237 million Arab Internet users [1]. Unprocessed Arabic data increased on many different means, such as social media and companies that offer a variety of services including chatbots and online markets. The study of this data has been particularly crucial in many related Natural Language Processing (NLP) areas, including sentiment analysis, named entity recognition, question answering, and identifying semantic similarity between phrases, the study of this data has been particularly crucial.

The subject of this study is Semantic Textual Similarity (STS), which is the foundation of innumerable applications and a crucial component in a variety of fields including information retrieval, plagiarism detection, information extraction, and machine translation [2]. Deep learning approaches have recently been successful at understanding the text's semantic meaning, particularly in the English language. This could hold the key to identifying the similarities between brief Arabic texts. Due to the complexity, diversity, ambiguity, and richness of the Arabic language's morphological-logical structure, applying these techniques can be difficult. Additionally, analyzing them can be challenging because short texts typically contain terse sentences. This study will examine and present methods for determining how semantically short Arabic sentences are similar.

Because determining the efficiency and efficacy of evaluating sentence similarity is a critical topic, it has drawn academics' attention in recent years [3]. Lexical or semantic similarity between sentences is possible. If two words share the same character arrangement, they are said to have a similar lexical meaning. For example, the sentences "**The dog sat on the mat**" and "**The cat sat on the mat**" have high lexical similarity because they share many words and the same character arrangement. When two sentences have the same meaning but are employed in diverse ways, such as "**I love eating pizza**" and "**pizza is my favorite food**", they are said to be semantically comparable because they convey the same idea that the speaker enjoys pizza. We have seen that sentences with completely different lexical structures can have the same semantic meaning.

The need to comprehend semantic meaning was initially sparked by two fundamental difficulties with natural language. The first difficulty is the vocabulary mismatch or the reality that the same meaning can be

communicated in many ways. The second difficulty is the ambiguity in natural language which is the possibility of several interpretations for a single term. We can divide traditional semantic understanding methods measurement into two main categories Corpus-based and Knowledge-based. Corpus-based means using statistics computed over large collections of texts. There are popular techniques like Latent Semantic Analysis (LSA) which is a counter-based model [4]. On the other hand, knowledge-based means using some pre-defined linguistic/semantic resource like WordNet [5].

Text vectorization, commonly referred to as "vector embedding," is the first step in NLP and is frequently used when words, sentences, or even greater units of text are displayed as vectors. Utilizing distance measures like Euclidean, Manhattan, and cosine similarity, these vectors can be utilized to find similarities. Luhn made the initial attempt to obtain word embedding [6]. It is Count-Based Text Vectorization, which counts distinct word terms in a document without considering the grammar or word order and assigns a weight to each word based on how frequently it appears in the document.

The representation of documents based solely on term frequency is considered inadequate since frequently used words such as "and, the, to" typically have the highest frequency in documents. Therefore, the presence of a high-frequency term does not necessarily signify its importance in the document. The Bag Of Words (BOW) algorithm was used to implement the term frequency method [7]. To address the limitation of the term frequency method, an Inverse Document Frequency Factor (IDF) was introduced by Sparck Jones [8], which reduces the weight of frequently occurring terms in the document set and increases the weight of rarely occurring terms as shown in the following equation.

$$tf - idf(t,d) = tf(t,d) * idf(t) \quad (1)$$

Where *tf(t,d)* represents how many times the term *t* appeared in document *d,* and *idf(t)* represents the inverse document frequency value. Although TF-IDF is a simple and effective means of calculating document similarity, it has the disadvantage of being unable to capture the semantic meaning of terms.

In recent times, advancements in neural network techniques, in conjunction with innovative research, such as the incorporation of attention mechanisms and transformers, have given rise to several novel innovations in the field of natural language processing. These include the Word2vec model developed by Google [9], the Glove model developed by Stanford University [10], and the Fasttext model developed by Facebook AI Research [11], in addition to transformers-based models that have gained considerable attention in the research community [12]. In this paper, we intend to leverage these recent innovative embedding mechanisms to vectorize the words and sentences of short Arabic texts, and subsequently measure the similarity between them.

2. Related Work

Based on Knowledge-based concepts Almarsoomi *et* al. [13] show how to use the Arabic knowledge base to calculate how similar two Arabic terms are semantically identical. In the hierarchy semantic net, the common and unique features between Arabic words are combined to determine the semantic similarity. To improve and assess the Arabic measure, a previously created Arabic word benchmark dataset is employed. The results of the experimental evaluation show that the Arabic measure is functioning well. Also, Malallah *et* al. [14] designed Arabic Semantic Network to store keywords related to computer science. This network used specific equations to find semantic similarities between words.

Using the corpus-based concept, Hussein [15], [16] talked about the study and visualization of document similarity using a content-based approach. By utilizing morphological analysis and lexical lookup, the suggested strategy is based on modeling the relationship between documents and their n-gram phrases produced from the normalized text. A technique for pair-wise matching that considers lexical and syntactic changes is used to build the TF-IDF model for the studied documents. Then, Latent Semantic Analysis (LSA) [4] is used to investigate the undiscovered relationships between the documents and their distinct n-gram phrases.

Also, Moatez *et* al. [2] suggested a technique to capture the semantic and syntactic characteristics of words by using vectors as word representations in a multidimensional space. To aid in the recognition of terms that are highly descriptive in each phrase, IDF weighting, and part-of-speech tagging are applied to the investigated sentences. The proposed system's effectiveness is verified by the Pearson correlation between the given semantic similarity scores and human assessments.

Hammad *et* al. [17] applied models to check Arabic question similarity. The authors used machine learning besides deep learning techniques to address the issue of understanding the semantic text similarity between Arabic question sentences. Three models were used: a supervised machine learning model using XGBoost, an adapted Siamese-based deep learning recurrent architecture, and a pre-trained deep bidirectional transformer based on BERT model. Using a reference Arabic dataset from the mawdoo3.com company, proposed models were assessed. BERT-based model performs better than the other two models, according to the evaluation results.

3. Dataset

The two datasets that were used in this study and in the primary preprocessing methods are described in this section.

3.1. *Data collection*

To train and evaluate the hereafter proposed models, we combine two datasets related to the public domain and question-answering system. The first dataset is a collection of Arabic question pairs made available by Mawdoo3 [18]. Mawdoo3's data annotation team manually annotated it, and it had 11997 entries with similarity labels of 1 or 0. The second dataset is the Arabic paraphrasing benchmark (MSRvid) which consists of 1010 labeled rows [19]. The full dataset structure is (13007) labeled rows divided into (6745) for non-similar pairs and (6262) for similar pairs. To ensure data will not be biased in training we downsampled it to non-similar pairs and the full structure becomes (12524) pairs distributed as shown in Table 1. We will divide our data to train the models mentioned below. It will be divided into training, validation, and testing records as 9270, 2005, and 2505 records, respectively.

Table 1. Dataset Distribution

Dataset	Positive /similar records	Negative/ non-similar records	Total records numbers	Balanced
Mawdoo3	5397	6600	11997	no
MSRvid	865	145	1010	no
Merged	**6262**	6745	13007	no
Downsampled	**6262**	**6262**	**12524**	**yes**

3.2. *Data Preprocess*

The dataset was prepared for training using a number of Arabic pre-processing approaches to increase accuracy and decrease noise. Those are:

- Removing of non-Arabic words.
- Removing hashtags and hyperlinks.
- Eliminating Arabic elongation and diacritics.
- Eliminating punctuation and symbols like "? (,),' ! @ \$ \# —".
- Normalization, which is used to remove "ء" from the "أإآ".

Arabic characters are normalized, and Arabic diacritics and punctuation are removed using specialized Python routines. The Natural Language Toolkit (NLTK) [20] library was used to remove stop words and, if necessary, it was used to perform the tokenization process, as will be demonstrated later.

4. Models Training and Evaluating

4.1. *Manhattan Long Short-Term Memory (MaLSTM) based on Aravec embedding*

Following data pre-processing, Arabic words are first embedded using the Word2vec word embedding mechanism. Using straightforward vector arithmetic, the Word2vec method may capture various levels of similarity between words. Through arithmetic operations, it is possible to derive patterns such as "man" to "woman" and "brother" to "sister." Because it is a shallow model and can build the vector representation using a Continuous Bag Of Words (CBOW) or Skip-Gram, it is capable of syntactic relationships like the present, past tense, and semantic relationships. The embedding vector's size is compact and adaptable.

The initial phase in this model involved applying the dataset preprocessing, NLTK tokenization, and calculating IDF weights. Then, we used AraVec [21], an open-source project that aims to deliver powerful and free word embedding models to the Arabic NLP research community through pre-trained distributed word representation. It utilized the Word2Vec algorithm on top of three distinct Arabic content domains, including Tweets, Web pages, and Arabic Wikipedia articles. AraVec provides different word embedding models. We used for the embedding phase Twitter-CBOW n-gram model that has (**1,476,715**) words token with a vector dimension of **300** for each token. These word embedding vectors of dim 300 can be used directly to measure the similarity of sentences by averaging all sentence word vectors and computing cosine similarity between two sentence vectors or any other distance metric. Also, the TF-IDF value for each word can be used to maximize important word embedding vectors and minimize less important word embedding vectors.

Suppose we have 2 sentences like "سافر أبي إلى تونس أمس" and "سافر أبي إلى تونس البارحة" and after processing steps sentences words having TF-IDF weights as shown in Table 2. Hence, we can compute the similarity between two sentences.

Table 2. TF-IDF Weights Of Words

word	TF-IDF weight
سافر	0.5617
ابى	0.4530
تونس	0.5759
امس	0.5941
البارحه	0.4869

v1= [(vector$_{dim300}$(سافر)* 0.5617+ vector$_{dim300}$ (ابى)*0.4530+ vector$_{dim300}$(تونس) * 0.5759+ vector$_{dim300}$(امس) *0.5941)]/4
v2= [(vector$_{dim300}$ (سافر)* 0.5617+ vector$_{dim300}$ (ابى)*0.4530+ vector$_{dim300}$(تونس) * 0.5759+ vector$_{dim300}$(البارحه) *0.4869)]/4
similarity (v1, v2) = cos (v1, v2) = 0.97

This method is not the best one for measuring sentence similarity because it doesn't give attention to the semantics of the sentence. So Long Short-Term Memory (LSTM) is used to avoid this weakness. LSTM is a Recurrent Neural Network (RNN)-based system. RNN is a particular kind of network where memory is formed via recurrent connections. All inputs in RNN are connected to each other, i.e., it is not a traditional feedforward network where the inputs are not reliant on one another. This enables the network to demonstrate robust temporal behavior for a time period, making it advantageous for sequential categorization such as text analysis [22].

RNN suffers from some drawbacks like vanishing or exploding gradient. Additionally, RNN basic structure has a limited memory, which makes it difficult for it to remember knowledge from earlier time steps in larger sequential data. These issues can be readily resolved by Long Short-Term Memory (LSTM) [23] and Gated Recurrent Unit (GRU) [24] as they are capable of remembering long periods of information. Repeating modules are present in LSTM as well as RNN, but the structure is different. LSTM uses a modified recurrent cell with an input gate, forget gate, and output gate in place of a typical recurrent cell like a Sigma or Tanh cell as shown in Fig. 1. This cell structure aids LSTM in maintaining long-term memory. This structure proved the ability to solve sequential issues like machine translation, speech synthesis, speech recognition, and handwriting recognition.

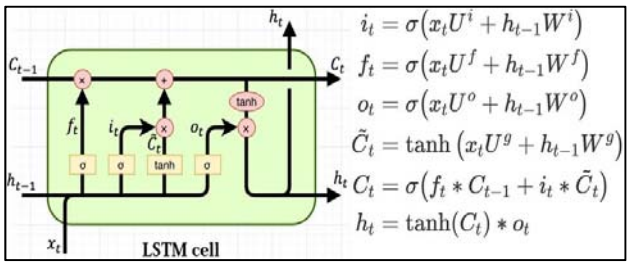

Fig. 1 LSTM cell [25].

Where x_t represents the input vector to the LSTM unit; i_t represents the input/update gate's activation vector; f_t represents forget gate's activation vector; o_t represents the output gate's activation vector; $\tilde{C}_t$ represents the cell input activation vector; C_t represents the cell state vector; h_t represents the hidden state vector or output vector of the LSTM unit.

The proposed model architecture shown in Fig. 2 adapted from [26] will be used for checking the similarity between two Arabic text pairs. The purpose of using LSTM is to capture the semantic meaning of sentences. The Two branches of LSTM shown in Fig. 2 are called Siamese networks. Siamese networks [27] are dual-branch or more networks with connected weights, meaning the weight are shared. Depending on the case, this network can be trained with loss functions like Mean Squared Error (MSE), binary cross-entropy, and ranking loss functions.

In this model, Manhattan distance is used to tries maximize the distance between different pairs and reduce the distance between similar pairs in an embedding space. Hence, we have absolute value the result will always be positive, and the exponential having negative absolute value the results will ∈ [0,1]. The Mean Squared Error (MSE) is used as a loss function to train the Siamese network in backpropagation through time which updates the weight in the network and Adadelta optimizer [28] adjusted with gradient clipping value (1.25) to avoid the exploding gradient problem.

We used early stopping when there was no change in validation loss for three epochs to prevent overfitting during training. There are 50 Long Short-Term Memory units (LSTM). We trained our model and tested it. we have results with an accuracy of 83% and F1-score of 82% on the training set. When evaluating the model on the test set with threshold 0.5 the accuracy result is 68% and the F1-score is 67%. The plot of accuracy and loss for training and confusion matrix for the test phase is shown in Fig. 3 and Table 3, consequently.

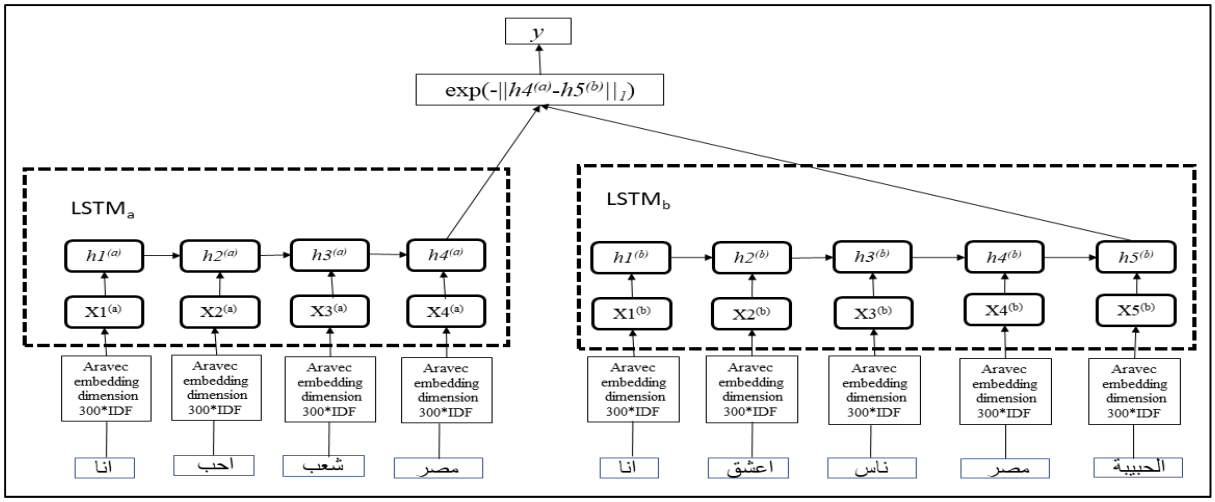

Fig. 2 The Proposed MaLSTM Architecture.

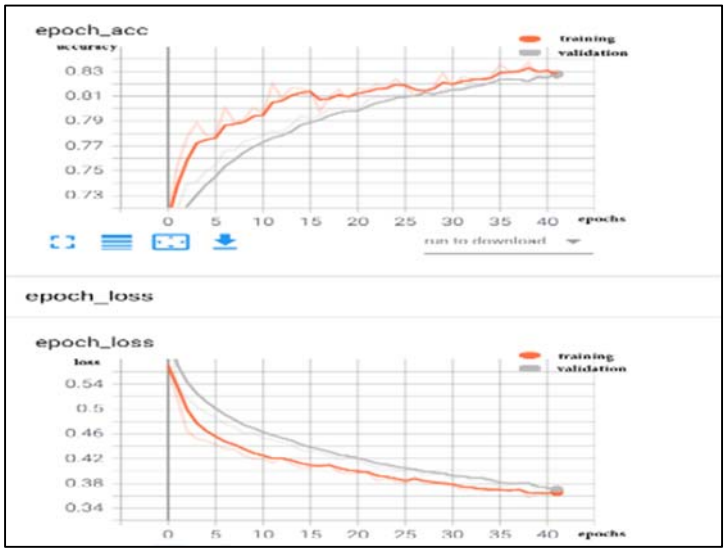

Fig. 3 The training accuracy and loss of the MALSTM vs. epochs

Table 3. The confusion matrix for evaluating the MaLSTM model on the test dataset

		Predicted classes	
		Similar pairs	Non-similar pairs
Actual classes	Similar pairs	TP=984	FN=276
	Non-similar pairs	FP=538	TN=707

4.2. *Marbert model*

In 2018, Google researchers unveiled Bidirectional Encoder Representations from Transformers (BERT) [29] which is a new language representation model that attained state-of-the-art performance on a variety of Natural Language Processing (NLP) tasks. It is a deep bidirectional text transformer model that has been pre-trained on a massive unlabeled text dataset that consists of 3.3 billion words. The Masked Language Modeling (MLM) objective is used to train BERT, a method for teaching language models to detect missing words in a phrase. The model can acquire contextual representations of words and phrases, which can then be refined for NLP tasks.

BERT architecture is based on the transformer paradigm, which was first introduced in the paper "Attention Is All You Need" [12]. Transformers are a type of neural network that can learn long-distance word dependencies. This is accomplished through self-attention, a mechanism that enables each word in a sentence to pay attention to

all other words in the sentence. With twelve encoder layers and twelve bidirectional self-attention heads totaling 110 million parameters, BERT model learns contextual representations of words and phrases. The positional encoding technique used In BERT model adds vector representations of the input sequence's order, the position of each word within the sequence, and the spacing between words to the embedding layer. These vectors aid in the capture of contextual data from the input sequence.

BERT is designed to learn from both the left and right context of a word in a sentence. It does this by pre-training deep bidirectional representations from the unlabeled text. This means that BERT can understand the context of a word in a sentence by looking at both the words that come before it and the words that come after it. After being pre-trained, BERT can be fine-tuned for specific NLP tasks. For example, BERT can be fine-tuned for question answering, natural language inference, and sentiment analysis. BERT has been shown to achieve state-of-the-art results on a wide range of NLP tasks [29].

A pre-trained model for the Arabic language based on Transformers called MARBERT [30] was fine-tuned to check semantic sentence similarity. Dialectal Arabic (DA) and Modern Standard Arabic (MSA) are both included in the Arabic dataset used by MARBERT model, which uses the same network architecture as BERT-base. MARBERT model was trained using a random sample of one billion Arabic tweets drawn from a sizable internal dataset of six billion tweets. The first phase is a tokenization procedure as shown in Fig. 5. A [CLS] token is added to the input word tokens at the beginning of the first sentence, and a [SEP] token is added at the end of each sentence. Segment embeddings are another example, where each token has a marker indicating either Sentence A or Sentence B. As a result, the encoder is able to distinguish between various phrases. The next step is to give each token positional embeddings to indicate where in the phrase it fits.

Because MARBERT model input max sequence is 512 token length a zero padding is added to the original tokens to apply a fixed length. These Tokens go through 12 layers of transformers encoder with an attention mechanism to learn the context of words. BERT model can give different embedding of the same word based on the contextual meaning of words like "**الوقت من ذهب وقد ذهب**". The word "**ذهب**" has a different contextual meaning and different embedding representation. In the end [CLS] embedding can be used for the classification process. The fine-tuned layer which consists of the linear layer maps the dimension of [CLS] embedding and then passes the values to the sigmoid function to give the probability of similar pairs or not.

We fine-tuned MARBERT model for the similarity classification problem using the Adam optimizer with a learning rate of 2e-5 and binary cross entropy loss function, 4 epochs training, and 128 for the maximum length of the tokenized input sentence pair and 16 for batch size. The experimental results showed that BERT base model gives significant improvements on the training dataset with an accuracy of 95% and an F1-score of 95% and an accuracy of 97.8% and an F1-Score of 97.7% on the test dataset evaluation. We computed similarity for the test dataset at threshold 0.5 and plot the confusion matrix between predicted and actual values. The result is shown in Table 4.

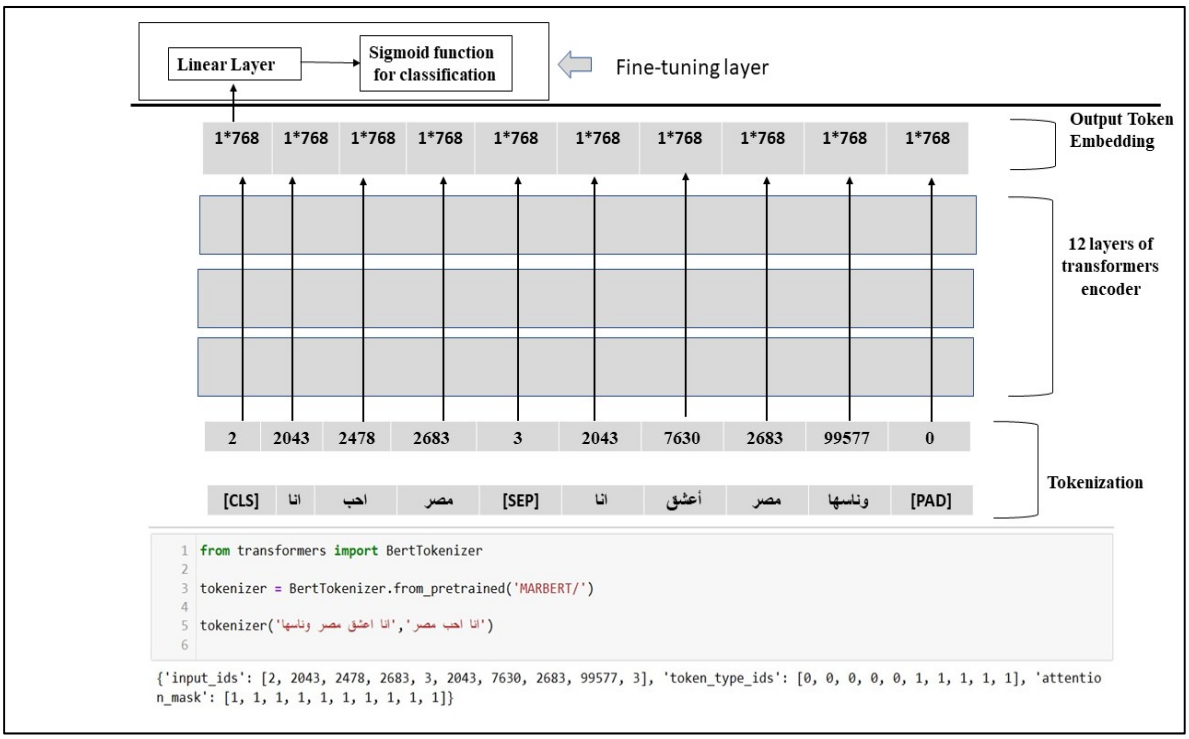

Fig. 5 The fine-tuned MARBERT model architecture.

Table 4. The confusion matrix for evaluating MARBERT model on the test dataset

		Predicted classes	
		Similar pairs	Non-similar pairs
Actual classes	Similar pairs	TP=1234	FN=26
	Non-similar pairs	FP=27	TN=1218

4.3. *Smarbert model*

The finetuned MARBERT model introduces significant enhancements compared to MaLSTM model due to its attention mechanism's ability to understand the semantic meaning. However, MARBERT model has a scalability issue. For instance, if we have a new text and want to query the most similar entry in our database consisting of 50K entries, it will require 50K tokenization steps and feedforward passes. The main problem arises from the fact that BERT mechanism outputs the embedding vector of sentence words instead of sentences.

A cross-encoder is employed, in which BERT receives two sentences and calculates a similarity score between them. However, this requires $n(n-1)/2$ computations, i.e., complexity of $O(n^2)$ [31]. When the number of sentences being compared surpasses hundreds or thousands of sentences due to brute force comparisons; that is comparing every sentence with all other sentences. To address this issue, if BERT architecture is used to generate meaningful embedding vectors at the sentence level. These sentence-level embeddings can be saved and used when searching for the most similar new text embedding vector using similarity metric functions such as cosine similarity which is a faster method.

The SBERT paper [31] solved this issue by using Siamese and triplet network architectures to derive the semantic meaning of sentences. They added a pooling layer at the end of BERT that outputs a fixed sentence representation embedding vector of size 768, which is based on three strategies that utilize the output embedding tokens of words. The first strategy uses [CLS] token embedding, the second strategy calculates the mean of all output vectors of words which is called the mean strategy and the last strategy calculates max-over-time for the embedding vectors of words which is called the MAX strategy. Fig. 7 below shows the architecture.

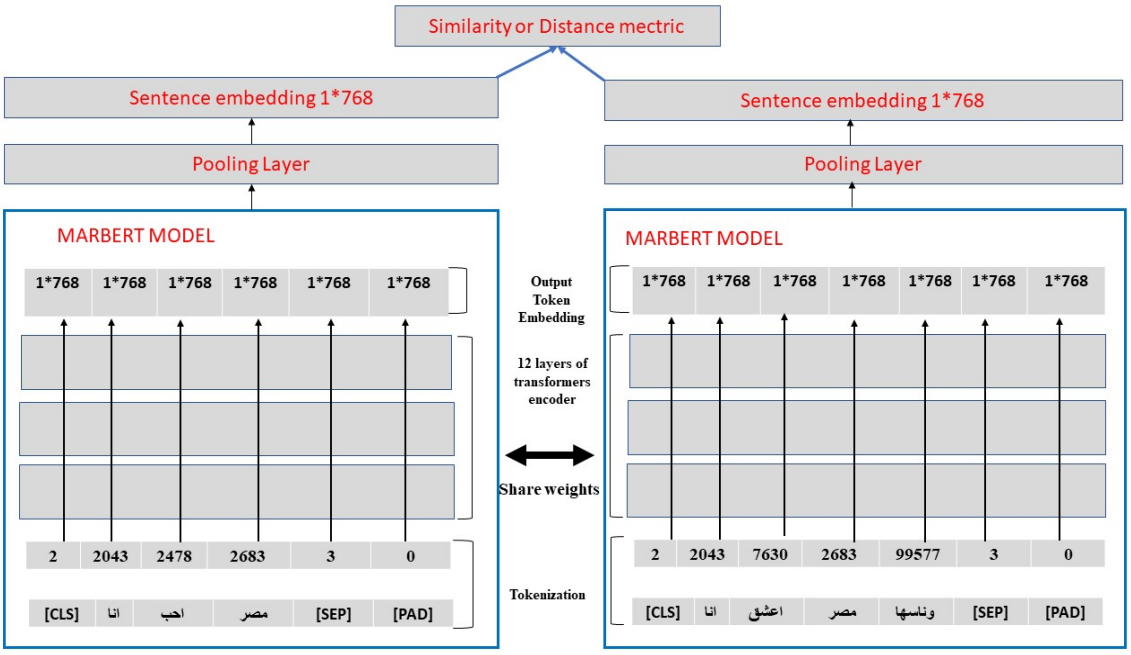

Fig. 7 Siamese MARBERT architecture

We trained SMARBERT using MARBERT model for embedding Arabic sentences with max sequence 128 for each sentence, batch sizes were 16, 4 epochs, Adam optimizer with $1e^{-6}$learning rate and apply a pooling layer to derive fixed-size sentence embedding based on mean strategy. The Contrastive loss [32] function presented in Eq. 2 with the Manhattan distance metric was chosen as the loss function because our input label consists of binary values. This function was first introduced in dimensionality reduction by learning an invariant mapping paper [33].

$$L(x_i, x_j, y) = y * \left(1 - d(x_i, x_j)\right)^2 + (1 - y) * d(x_i, x_j)^2 \qquad (2)$$

Where x_i and x_j are the feature vectors of the two texts in the pair; y is the label in the truth table and $d(x_i, x_j)$ is the distance between the two feature vectors which is Manhattan in our case.

The experimental results showed that SMARBERT model gives significant improvements with the training dataset with an accuracy of 98% and an F1-score equal to 98%. The results for the test dataset have an accuracy of 99.5%, and an F1-Score of 99.5%. We computed similarity for the test dataset at threshold 0.5 and plot the confusion matrix between predicted and actual values. The result is shown in Table 5.

Table 6 showed that SMARBERT model is the best one able to check the similarity between two Arabic sentences. Our results compared with Hammad *et* al.[17] because the same question dataset is used beside the public domain dataset. They proposed 3 models as mentioned above to check the similarity between Arabic questions. The results are shown in Table 7. MARBERT and SMARBERT models introduce significant enhancements over the three models used by the Hammad team.

Table 5. The confusion matrix for evaluating SMARBERT model on the test dataset

		Predicted classes	
		Similar pairs	Non-similar pairs
Actual classes	Similar pairs	TP=1258	FN=2
	Non-similar pairs	FP=9	TN=1236

Table 6. Results

	Training Dataset		**Test Dataset**	
	Accuracy	**F1-score**	**Accuracy**	**F1-score**
Siamese MaLSTM	83%	82%	68%	67%
MARBERT	95%	95%	97.8%	97.6%
SMARBERT	**98%**	**98%**	**99.9%**	**99.9%**

Table 7. Hammad Team Results

Model	**XGBoost**	**Siamese-based**	**BERT-Based**
F1	86.086%	89.048%	92.99%

5. Conclusions and Future Work

In this study, we have explored the challenges of measuring semantic similarity in short Arabic texts and proposed deep learning approaches to address these challenges. Our experiments show that these approaches can accurately assess the semantic similarity of short Arabic texts. Nevertheless, there is still space for improvement, especially in addressing the complexity and diversity of Arabic language structures.

Future research could concentrate on constructing more complex models capable of handling the ambiguity and complexity of Arabic language structures. In addition, additional research is required to determine the efficacy of these models on various categories of brief Arabic texts, such as social media posts and news headlines. In addition, it would be intriguing to investigate how these models can be incorporated into existing natural language processing systems in order to enhance their performance in a variety of applications.

In conclusion, our research casts light on the difficulties and potential solutions associated with measuring semantic similarity in brief Arabic texts. We anticipate that our findings will inspire additional research in this area and contribute to the development of more effective systems for processing Arabic natural language.

Conflicts of interest

The authors have no conflicts of interest to declare.

References

[1] "Top Ten Internet Languages in The World - Internet Statistics." https://www.internetworldstats.com/stats7.htm (accessed Dec. 02, 2022).

[2] E. Moatez, B. Nagoudi, and D. Schwab, "Semantic Similarity of Arabic Sentences with Word Embeddings," 2017. [Online]. Available: https://sites.google.com/site/mohazahran/data
[3] M. Alian and A. Awajan, "Arabic sentence similarity based on similarity features and machine learning," *Soft comput*, vol. 25, no. 15, pp. 10089–10101, Aug. 2021, doi: 10.1007/s00500-021-05754-w.
[4] T. K. Landauer and S. T. Dumais, "A Solution to Plato's Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge We thank Karen Lochbanm for valuable help in analysis; George Fumas for early ideas and inspiration; Peter Foltz, Walter Kintsch," *Psychol Rev*, vol. 104, no. 2, pp. 211–240, 1997, [Online]. Available: http://www.indiana.edu/~pcl/rgoldsto/courses/concepts/landauer.pdf%0Ahttp://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.333.7403&rep=rep1&type=pdf
[5] Y. Li, D. McLean, Z. A. Bandar, J. D. O'Shea, and K. Crockett, "Sentence similarity based on semantic nets and corpus statistics," *IEEE Trans Knowl Data Eng*, vol. 18, no. 8, pp. 1138–1150, 2006, doi: 10.1109/TKDE.2006.130.
[6] H. P. Luhn, "A Statistical Approach to Mechanized Encoding and Searching of Literary Information*."
[7] W. A. Qader, M. M. Ameen, and B. I. Ahmed, "An Overview of Bag of Words;Importance, Implementation, Applications, and Challenges," *Proceedings of the 5th International Engineering Conference, IEC 2019*, no. July, pp. 200–204, 2019, doi: 10.1109/IEC47844.2019.8950616.
[8] K. SPARCK JONES, "A STATISTICAL INTERPRETATION OF TERM SPECIFICITY AND ITS APPLICATION IN RETRIEVAL," *Journal of Documentation*, vol. 28, no. 1, pp. 11–21, Jan. 1972, doi: 10.1108/eb026526.
[9] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *1st International Conference on Learning Representations, ICLR 2013 - Workshop Track Proceedings*, pp. 1–12, 2013.
[10] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation." [Online]. Available: http://nlp.
[11] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," *Trans Assoc Comput Linguist*, vol. 5, pp. 135–146, 2017, doi: 10.1162/tacl_a_00051.
[12] A. Vaswani *et al.*, "Attention is all you need," *Adv Neural Inf Process Syst*, vol. 2017-Decem, no. Nips, pp. 5999–6009, 2017.
[13] F. A. Almarsoomi, J. D. O'Shea, Z. Bandar, and K. Crockett, "AWSS: An algorithm for measuring Arabic word semantic similarity," *Proceedings - 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013*, no. October, pp. 504–509, 2013, doi: 10.1109/SMC.2013.92.
[14] S. Malallah, A. Qassim, and A. Alameer, "Finding the Similarity between Two Arabic Text," *Iraqi Journal of Science*, vol. 58, no. 1, pp. 152–162, 2017.
[15] A. S. Hussein, "Arabic document similarity analysis using n-grams and singular value decomposition," *Proceedings - International Conference on Research Challenges in Information Science*, vol. 2015-June, no. June, pp. 445–455, 2015, doi: 10.1109/RCIS.2015.7128906.
[16] A. S. Hussein, "Visualizing document similarity using n-grams and latent semantic analysis," *Proceedings of 2016 SAI Computing Conference, SAI 2016*, pp. 269–279, 2016, doi: 10.1109/SAI.2016.7555994.
[17] M. Hammad, M. Al-Smadi, Q. B. Baker, and S. A. Al-Zboon, "Using deep learning models for learning semantic text similarity of Arabic questions," *International Journal of Electrical and Computer Engineering*, vol. 11, no. 4, pp. 3519–3528, Aug. 2021, doi: 10.11591/ijece.v11i4.pp3519-3528.
[18] "NSURL 2019: Task8 | Kaggle." https://www.kaggle.com/competitions/nsurl-2019-task8/data (accessed May 11, 2023).
[19] "GitHub - marwah2001/Arabic-Paraphrasing-Benchmark: Arabic paraphrasing benchmark consists of 1010 Arabic sentence pairs with label of similarity and paraphrasing." https://github.com/marwah2001/Arabic-Paraphrasing-Benchmark (accessed Aug. 06, 2022).
[20] "NLTK :: Natural Language Toolkit." https://www.nltk.org/ (accessed Nov. 27, 2022).
[21] A. B. Soliman, K. Eissa, and S. R. El-Beltagy, "AraVec: A set of Arabic Word Embedding Models for use in Arabic NLP," *Procedia Comput Sci*, vol. 117, no. September, pp. 256–265, 2017, doi: 10.1016/j.procs.2017.10.117.
[22] A. Aziz Sharfuddin, M. Nafis Tihami, and M. Saiful Islam, "A Deep Recurrent Neural Network with BiLSTM model for Sentiment Classification," *2018 International Conference on Bangla Speech and Language Processing, ICBSLP 2018*, no. October, 2018, doi: 10.1109/ICBSLP.2018.8554396.
[23] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.
[24] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *EMNLP 2014 - 2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pp. 1724–1734, 2014, doi: 10.3115/v1/d14-1179.
[25] S. Varsamopoulos, K. Bertels, and C. G. Almudever, "Designing neural network based decoders for surface codes Accelerated BWA-MEM View project hartes View project Designing neural network based decoders for surface codes," *arXiv:1811.12456 [quant-ph]*, no. November, pp. 1–12, 2018, [Online]. Available: https://www.researchgate.net/publication/329362532
[26] J. Mueller and A. Thyagarajan, "Siamese recurrent architectures for learning sentence similarity," *30th AAAI Conference on Artificial Intelligence, AAAI 2016*, no. 2014, pp. 2786–2792, 2016, doi: 10.1609/aaai.v30i1.10350.
[27] D. Chicco, "Siamese Neural Networks: An Overview," *Methods in Molecular Biology*, vol. 2190, pp. 73–94, 2021, doi: 10.1007/978-1-0716-0826-5_3.
[28] "Adadelta." https://keras.io/api/optimizers/adadelta/ (accessed Apr. 06, 2023).
[29] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, vol. 1, no. Mlm, pp. 4171–4186, 2019.
[30] M. Abdul-Mageed, A. R. Elmadany, and E. M. B. Nagoudi, "ARBERT & MARBERT: Deep bidirectional transformers for Arabic," *ACL-IJCNLP 2021 - 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, no. ii, pp. 7088–7105, 2021, doi: 10.18653/v1/2021.acl-long.551.
[31] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," Aug. 2019, [Online]. Available: http://arxiv.org/abs/1908.10084
[32] "Losses — Sentence-Transformers documentation." https://www.sbert.net/docs/package_reference/losses.html (accessed Apr. 06, 2023).
[33] 0R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 1735–1742, 2006, doi: 10.1109/CVPR.2006.100.

Mohamed Abdelnabi Ibrahim is currently pursuing a master's at the faculty of engineering at Ain shams university. Also, he obtained his B.C.A degree from a military-technical college in 2013 and he worked as a freelancer engineer in the fields of programming and artificial intelligence.

Prof. Dr. Ahmed Zaki Badr is a prominent academic and politician in Egypt. He is currently a Professor of Computer and System Engineering at Ain Shams University, where he conducts research and teaches. He is also the Chairman of the Board of Trustees of October 6 University. Prof. Badr has a Bachelor's degree in Electrical Engineering and a Ph.D. in Computer and Automatic Control Engineering. His research interests include control systems, optimization, and artificial intelligence. In addition to his academic work, Prof. Badr has held several key positions in the Egyptian government, including Minister of Education and Minister of Local Development. He played a crucial role in implementing reforms in the education sector and promoting decentralization and community empowerment.

Prof. Dr. Hani M. K. Mahdi is an Emeritus Professor of Computer Systems at Ain Shams University in Cairo, Egypt. He received his Doctorate from Technische Universität Braunschweig in West Germany in 1984 and was a Post-Doctoral Research Fellow at The Pennsylvania State University from 1988-1989. He has also been a visiting professor at the University of Louisville. Prof. Mahdi is an IEEE Life Senior Member and has received numerous accolades throughout his distinguished career, including the Distinguished University Award from Ain Shams University in 2015. His research interests include pattern recognition and artificial intelligence, and he has published extensively in these areas. Prof. Mahdi has served on the editorial boards of several international journals and has been a keynote speaker at numerous conferences.