Applied Artificial Intelligence
An International Journal

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/uaai20

Hybrid Attention-based Approach for Arabic Paraphrase Detection

Adnen Mahmoud & Mounir Zrigui

To cite this article: Adnen Mahmoud & Mounir Zrigui (2021) Hybrid Attention-based Approach for Arabic Paraphrase Detection, Applied Artificial Intelligence, 35:15, 1271-1286, DOI: 10.1080/08839514.2021.1975880

To link to this article: https://doi.org/10.1080/08839514.2021.1975880

APPLIED ARTIFICIAL INTELLIGENCE
2021, VOL. 35, NO. 15, 1271–1286
https://doi.org/10.1080/08839514.2021.1975880

Taylor & Francis
Taylor & Francis Group

Hybrid Attention-based Approach for Arabic Paraphrase Detection

Adnen Mahmoud [a,b] and Mounir Zrigui [a]

[a]Research Laboratory in Algebra, Numbers Theory and Intelligent Systems RLANTIS, University of Monastir, Monastir, Tunisia; [b]Higher Institute of Computer Science and Communication Techniques ISITCom, University of Sousse, Hammam Sousse, Tunisia

ABSTRACT

The growth of data across the web and the ambiguous structure of Arabic language have favored the act of paraphrase. It is defined as a restatement of the original text, giving the same meaning in another form without mentioning its source. Its detection requires calculating semantic textual similarity, which is an important research area in Natural Language Processing (NLP) tasks. Following the literature, deep neural network models have gained satisfactory results in sentence modeling and similarity computing. In this context, a hybrid Siamese neural network architecture is proposed that is composed of the following main components: First, salient features are extracted by applying Global Vectors Representation (GloVe). Then, Convolutional Neural Networks (CNN) capture and learn the contextual meaning of words due to their outstanding performance that has been achieved in different NLP tasks. Then, the output of CNN is combined with an attention model to distinguish the most important words representing the meaning of the sentence. The similarity score between sentences was subsequently computed by applying the cosine measure. Experiments were carried out on a proposed Arabic paraphrased corpus using the Open-Source Arabic Corpora (OSAC). To validate its quality, the SemEval benchmark is used.

ARTICLE HISTORY
Received 5 February 2021
Revised 19 July 2021
Accepted 27 August 2021

Introduction

The abundance of online resources and ease of internet access has increased the act of plagiarism. It consists of copying original texts without asking permission or indicating the source. Nowadays, paraphrase has become one of the famous types of plagiarism. It consists of rewriting ideas in your own words by changing grammar, substituting synonyms or rearranging sentences. Its detection represents important opportunities and challenges in various Natural Language Processing (NLP) applications such as machine translation, plagiarism detection, question answering, etc. (Mahmoud and Zrigui 2021a). It is necessary to measure the degree of semantic similarity between sentences, which is very difficult due to the lack of common lexical features and the

CONTACT Adnen Mahmoud mahmoud.adnen@gmail.com Research Laboratory in Algebra, Numbers Theory and Intelligent Systems RLANTIS, University of Monastir, Monastir, Tunisia

linguistic variation problems. Our research is oriented toward finding an approach to detect Arabic paraphrase because of its specific semantic and syntactic structures compared to Western languages like English.

Recently, deep learning has ushered in amazing technological advances in learning word vector representations through natural language models and composition over word vectors (Du, Gui, and Xu (2017)). In this paper, a hybrid Siamese neural network architecture is proposed by combining Convolutional Neural Network (CNN) with an attention model for modeling Arabic documents, further joining their resulted vectors by a similarity function. The proposed approach is composed of the following main components: First, salient features are extracted by applying Global Vectors Representation (GloVe). Then, Convolutional Neural Networks (CNN) capture and learn the contextual meaning of words due to their outstanding performance that has been achieved in different NLP tasks. Then, the output of CNN is combined with an attention model to distinguish the most important words representing the meaning of the sentence. Thereafter, a join function is used to compute the score of a sentence-pair similarity. For experiments, an Arabic paraphrased corpus is proposed using the Open-Source Arabic Corpora (OSAC). To validate its quality, the SemEval 2017 benchmark is used.

The rest of this paper is structured as follows: Section 2 presents a literature review on paraphrase detection systems. Section 3 briefly describes the components of the suggested approach. Subsequently, section 4 details the dataset preparation and parameter settings, and it discusses thereafter the experimental results compared to the state-of-the-art methods. Finally, section 5 concludes the paper and presents some future works.

Literature Review

Several methods have been put forward for extracting features and subsequently detecting different forms of reuse between multilingual and monolingual documents. There are those that are identified lexical matching between texts, in which the similarity score is computed according to the number of terms belonging to both textual segments. In contrast, these measures cannot compute the similarity beyond a trivial level and can only estimate the textual similarity but not the semantic one (Shajalal and Aono 2018). In recent years, word-embedding-based models have gained competitive results in capturing contextual semantic meaning of words. Thus, words with similar contexts have embeddings close to each other in the high-dimensional space (Zuo et al. 2018).

Cosma (2011) introduced a semantic-based approach for detecting and investigating source-code plagiarism using Latent Semantic Analysis (LSA). This technique was integrated with the PlaGate plagiarism detection tool to extract semantics between source code fragments. For estimating similarities,

a parse-tree kernel method was applied to give the structure of the source code functionality. To detect plagiarism in students' programming assignments based on semantics, multimedia e-learning-based smart assessment methodology was propounded by Ullah et al. (2020). It was processed as follows: Source codes were converted to tokens. Next, the Document Term Frequency Matrix (DTFM) was prepared and weighted according to the terms used in the source code. Then, they extracted the semantic features of each token using the LSA technique. It efficiently measured the semantic similarity without a parser requirement for any programming language.

Ezzikouri, Errital, and Oukessou (2017) presented a fuzzy-semantic-based approach for multilingual plagiarism detection. This was in accordance with the WordNet lexical database. Thus, different pre-processing methods were used including lemmatization, stop word removal and Part Of Speech (POS) tagging as well as n-gram segmentation for Arabic and English languages. Afterward, the Wu and Palmer similarity measures evaluated the resemblance between texts using fuzzy-semantic similarity measures. Similarly, Alzahrani and Salim (2008) introduced a fuzzy information retrieval model for detecting verbatim reproductions. It was extended by Salha, Alzahrani and Salim (2010) to identify rewording with the shingling algorithm and Arabic WordNet. To do this, the Jaccard coefficient selected the candidate documents, which were compared with suspect documents through a fuzzy-semantic-based string similarity. Subsequently, Alzahrani (2015) retrieved a list of candidate source documents using n-gram fingerprinting and the Jaccard coefficient. Then, a k-overlapping approach was applied to compare source and suspect documents. Finally, consecutive n-grams were joined to form the united plagiarized segments.

For Bengali language, Shajalal and Aono (2018) developed a semantic-similarity-based approach. Indeed, word-level semantics were extracted from a pre-trained word-embedding model (word2vec) trained on Bengali Wikipedia texts. Thereafter, the semantic similarity score was computed by applying the cosine similarity technique. To test the performance of this method, a Bengali dataset was prepared using the same approach as SemEval employed in the Semantic Text Similarity workshop (STS 2017). In contrast, Nagoudi et al. (2018) proposed an Arabic plagiarism detection approach. It detected verbatim and complex reproductions using fingerprinting and word embedding techniques. Next, word alignments, Inverse Document Frequency (IDF) and POS weighting identified the most descriptive words in each textual unit.

Deep neuronal architectures have improved semantic analysis and semantic similarity prediction. CNNs have achieved promising outcomes. Indeed, convolutional filters were effectively applied to identify the most descriptive n-grams of different semantic aspects (Mahmoud and Zrigui 2019). To extract different granularities, sentence embedding was

generated using a Siamese CNN by He, Gimpel, and Lin (2015). Multiple convolutions were employed using filters with various window sizes. Afterward, max, min and mean pooling operations were studied. For similarity computation, horizontal and vertical comparisons were applied in local regions of sentence representation. In the same vein, the semantic textual similarity in Arabic language was studied using word2vec for feature extraction and CNN model for sentence modeling and classification based on different window sizes and max pooling operations (Mahmoud and Zirgui 2017)

Other works have focussed on recurrent neural networks (RNNs) models. However, they risked vanishing or exploding gradient problems (Bsir and Zrigui 2018). Therefore, a Long Short-Term Memory (LSTM) architecture was introduced to efficiently learn long-term dependencies (Pontes et al. 2018). Based on the advantages of this model, Siamese LSTM was employed by Mueller and Aditya (2016). It generated sentence vector representation and predicted thereafter similarities using Manhattan distance. The un-labeled short text similarity was measured by Yao, Pan, and Ning (2019). The objective was to avoid gradient vanishing problems in the process of backward propagation faster after normalization. To do this, the training stage leveraged the inception module to extract the features of different dimensions and improved the LSTM encoder to process the relationships of word sequences. The evaluation stage employed the cosine distance to calculate semantic similarity. To reduce the parameters in the update and reset gates, multiple GRUs were introduced by Dey and Fathi (2017). The first (GRU1) computed each gate using only the previous hidden state and the bias. The second (GRU2) computed each gate utilizing just the previous hidden state. The last (GRU3) computed only the bias. Experiments showed that GRU1 and GRU2 had indistinguishable performances, whereas GRU3 frequently lagged in performance, especially for relatively long sequences and required more execution time.

More recently, researches have incorporated attention mechanisms into semantic features. They have demonstrated great success in distinguishing between the most important words insentences. Johnson and Tong (2014) expressed sentences by an attention pooling-based CNN. Then, the attention weight was obtained by an intermediate sentence representation. It was generated through Bi-directional LSTM (Bi-LSTM). In contrast, Ma et al. (2019) proposed a gated attentive-auto encoder (GATE) model for content-aware recommendation. It exploited neighboring relations between items to help infer users' preferences. Word-level attention learned the item hidden representations from word sequences of items while favoring informative words by assigning larger attention weights. Neighbor-level attention learned the hidden representation of the neighborhood of an item by considering its neighbors in a weighted manner

Following the state of the art, few works have been proposed for paraphrase detection, especially in Arabic language. This is due to its complex specificities that represent challenges in lexical, syntactic and semantic analysis. It needs a deep understanding of text (Haffar, Hkiri, and Zrigui 2020). Indeed, Arabic script is not only used for writing but also used in several other languages in Asia and Africa like Urdu, Persian, Azerbaijani, and others (Abdellaoui and Zrigui 2018). This language has large variations in textual representations (Hkiri, Mallat, and Zrigui 2020). It is non-vocalized, non-concatenative, homographic, agglutinative and derivational. This richness of features has made its processing more difficult than other languages (Batita and Zrigui 2018; Maraoui, Terbeh, and Zrigui 2018). On the other hand, deep neural networks have outperformed traditional techniques (e.g., LSA, TF-IDF, LDA, etc.) for semantic similarity computation. Inspired by the role of context in the attention model, a hybrid convolutional attentional neural network model is proposed. The aim is to extract salient features from the text and subsequently improve the performance of Arabic paraphrase detection.

Proposed Approach

Arabic paraphrase detection aims to pre-process textual documents and extract their discriminant features. Then, similarity computation consists in determining the score of semantic relatedness between train and test corpora. For this purpose, a Siamese Neural Network (SNN) architecture is proposed that is efficient to optimize the matching task. Given two input vectors, SNN consists of two identical neural networks sharing the same weights. The resulting output vectors are fed to a join function based on the distance computation between them. This section details the components of the proposed approach, as shown in Figure 1. It is based on NLP and data mining techniques, as follows:

(1) First, documents are transformed into a more understandable form so that further processing can perform better.
(2) Next, global word vector representations (GloVe) are generated by mapping sentences into an interpretable geometric space.
(3) Subsequently, the pre-trained vectors are used as inputs in a hybrid convolutional attentional neural network.
(4) Finally, a paraphrase identification layer based on the cosine measure is used to produce a prediction score.

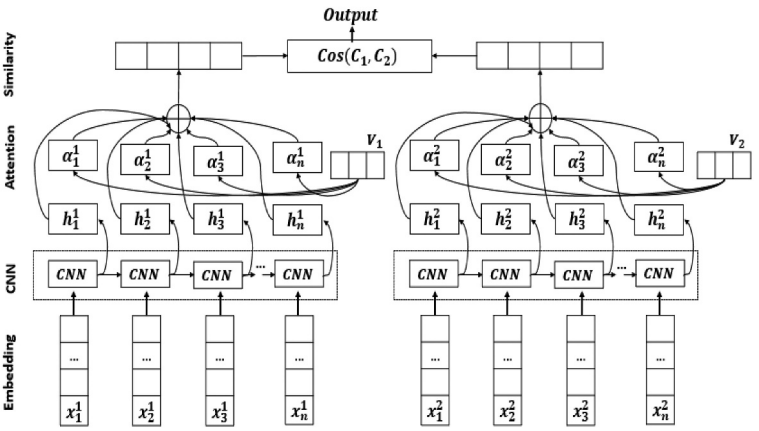

Figure 1. Proposed architecture for Arabic paraphrase detection.

Pre-processing

Pre-processing operations are essential in Arabic NLP systems for reducing lexical parsimony and storing texts into comprehensible and simple formats. In order to facilitate the process of Arabic paraphrase detection, different pre-processing techniques are applied:

First, remove unnecessary data that have little or no semantic meaning associated, such as diacritics, extra white spaces, title numeration, duplicated letters and non-Arabic words.

Second, normalize some writing forms. For example, all forms of alif "أ" and hamza "ء" are converted to a single form "ا." Likewise, Taa Marboutah "ة" is converted to ha "ه."

Finally, split words regarding the white space between them. This operation is called tokenization to reduce the lexical parsimony problem.

Global Word Embedding

GloVe is employed for efficiently capturing the contextual relationships between words (Mahmoud and Zrigui 2021b). It learns semantics and grammar information, taking into account the context of words and the information of the global corpus. Formally, GloVe builds a matrix M_{ij} of word–word co-occurrences by estimating the probability of appearance of a word w_i in the context of another word w_j. It is based on an objective function J to produce vectors with a fixed dimension according to

a vocabulary size V, scalar biases b_i and b_j and a weighting co-occurrence function f(x) for rare and frequent words. It is defined as follows in Equation (1):

$$J = \sum_{i,j=1}^{V} f\left(X_{ij}\right)\left(w_i^T w_j + b_i + b_j - \log\left(X_{ij}\right)\right)^2 \quad (1)$$

Given a sentence $S = [w_1, w_2, \ldots, w_n]$ of length N, in which w_i is the i-th word of the sentence represented by an incorporation x_i. It is a line of dimension K in a matrix $M = [x_1, x_2, \ldots, x_n]$ of size $N \times K$.

CNN Based-attention Model

Convolutional Neural Network (CNN) Layer

Given the pre-trained vectors of an input sequence $X = [x_1, x_2, \ldots, x_n]$, CNN captures invariant contextual features through convolutional, pooling and fully connected layers. It is used to extract the most descriptive and influential n-grams of different semantic aspects from the text. Given a window size $w_s = 3$, a convolution $h_i h_i$ is based on a filter weight $F_i \in R$ of size 64. It is defined in Equation (2) as the dot product between F_i and each sequence of $(n - w_s + 1)$ words, where the ReLU is a nonlinear activation function, and $[b_1, \ldots, b_n]$ are the bias functions:

$$h_i = \mathrm{ReLU}(F_i.x_{i:n-w_s+1} + b_i) \quad (2)$$

After applying the filter $F \in R^{w_s \times K}$ to each possible window of words w_s in the embedding sequence, a new feature map is produced as defined in Equation (3):

$$c_i = [c_1, \ c_2, \ldots, c_{n-w_s+1}] \quad (3)$$

A max pooling layer captures the most descriptive and common words, as defined in Equation (4). The main objective is to produce; thereafter, K relevant and useful feature maps $H = \{h_1, \ldots, h_k\}$ that are concatenated afterward to enhance the generalization ability of the model.

$$h_i = \mathrm{Max}_{1 \le i \le n-w_s+1} c_i \quad (4)$$

Attention Based CNN Layer

The attention mechanism is useful at highlighting the important words representing the sentence in the form of a fixed sentence vector representation. It is computed as the weighted sum of all words using the attention weights.

For each time step t, h_t is fed through a fully connected network to get e_t as a hidden representation. It is based on a Tangent Hyperbolic function (Tanh). It helps to identify relevant features by generating high scores for them. It is

the core idea of the attention mechanism. Then, weights are calculated by applying Softmax function on the scores, which are called attention weights representing the relevance of each word in the sentence. After that, a context vector is constructed applying a multiplication of weights and the features generated from the CNN. Finally, a fixed representation r of the whole sentence is computed as the weighted sum of all hidden states h_t using the attention weights a_t. Formally, the attention mechanism process is denoted in Equations (5)–(7):

$$e_{\mathrm{t}} = \tanh(W_h h_{\mathrm{t}} + b_h), e_{\mathrm{i}} \in [-1, 1] \tag{5}$$

$$a_{\mathrm{t}} = \frac{\exp\left(e_{\mathrm{t}}^T u_h\right)}{\sum^{\mathrm{t}=1} T \exp(e_t^T u_h)}, \sum_{\mathrm{t}=1}^{T} a_{\mathrm{t}} = 1 \tag{6}$$

$$r = \sum_{\mathrm{t}=1}^{T} a_{\mathrm{t}} h_{\mathrm{t}}, \ r \in R^{2L} \tag{7}$$

Where: $W_h, b_h and\ u_h$ are the layer's weights.

Similarity Computation

The hybrid SNN consists of two identical sub-networks that extract semantic features from two disparate sentences. Then, the cosine similarity is used as the join function. It measures the semantic similarity between the obtained two hidden vectors. It determines how similar real-valued vectors of sentences are irrespective of their sizes by calculating the angle or correlation between them. It is defined in Equation (8):

$$z = \mathrm{Cos}(\mathrm{x}, \mathrm{y}) \tag{8}$$

Where: x and y are the output word vectors of the Siamese sub-networks. The cosine similarity z takes values between [−1, +1] which are converted into probabilities P as follows in Equation (9):

$$\frac{\mathrm{z}}{10} = \frac{(\mathrm{z} \times 5)}{100} + \frac{50}{100} \tag{9}$$

For Arabic paraphrase detection, the output of the cosine similarity is predicted according to a threshold $\beta = 30\%$ as defined in Equation (10):

$$\begin{cases} \mathrm{ifP}(\mathrm{x}, \mathrm{y}) \leq \beta, \mathrm{thensentencesaresimilar} \\ \mathrm{otherwise}, \mathrm{sentencesaredissimilar} \end{cases} \tag{10}$$

Experiments

Experiments are carried out on a developed paraphrased corpus in Arabic language. Its performance is validated and compared to the Arabic dataset of SemEval STS task. The remainder of this section presents the details of the dataset collection, the experimental setup and the comparative discussion.

Datasets

A semi-automatic approach is proposed to construct an obfuscated corpus. Different datasets are collected to form the source and vocabulary corpora:

- Open-Source Arabic Corpora (OSAC) are collected from various categories (e.g., history, economics, sports, etc.). These datasets represented the source corpus from which their contents are paraphrased. (Saad and Ashour 2010)
- More than 2.3 billion vocabulary words are collected from different resources including the King Saud University Corpus of Classical Arabic (KSUCCA) (Alrabiah et al. 2014), the Arabic Corpora resource (AraCorpus[1]) and a set of Arabic papers from Wikipedia.[2]

The degree of paraphrase P is configured randomly using a random uniform function. It is fixed between 0.45 and 0.75. Using P, the number of words to replace S in the source corpus of size N is calculated as follows in Equation (11):

$$\mathrm{S} = \mathrm{P} \times \mathrm{N} \tag{11}$$

Distributed word vector representation (word2vec) is efficient for analogy reasoning offering an expressive representation of words with low-dimensional vectors. In our work, we use it to extract the synonyms of each source word from the vocabulary of size V. For its training, the Skip-gram model is employed with a window size w_s of 3. It covered three words w_k behind and ahead of the current word w_t, a minimum frequency of 5 and a vector dimension of 300, defined as follows in Equation (12) (Mahmoud and Zrigui 2018):

$$\frac{1}{V}\sum_{j=1}^{V}\sum_{k=-w_s}^{w_s}\log p(w_{j+k}|w_j) \tag{12}$$

For conserving the syntactic and semantic properties of Arabic sentences, each word from the source corpus is replaced by its most similar one, which has the same grammatical class (POS tag). This is done by using the random shuffle function. Table 1 represents an example of a paraphrased sentence construction

Table 1. Arabic paraphrased sentence generation.

Source Tokens	Arabic / Translation	اتمنى لك النجاح في العمل I wish you success in business Synonyms		
Arabic	Translation	POS	Arabic, Cos	Translation
	wish	Verb	0.83))	hope
			| 0.84))	|trust
			| 0.78))	|request
	you	Preposition		you
	the success	Noun	0.86))	success
			|0.83))	|success
			| 0.77))	|accord
	in	Preposition		in
	the business	Noun	0.85))	the career
			| 0.83))	|the work
			| 0.87))	|the job
P_1	Arabic			
	Translation	I wish you would succeed in a job		
P_2	Arabic			
	Translation	I hope you success in the job		

Table 2. Experimental datasets.

Corpora	Models	# Total Sentence-pairs	# Paraphrased Sentence-pairs	# Different Sentence-pairs
Proposed corpus	Train	720	480	240
	Test	300	200	100
SemEval	Train	510	357	153
	Test	210	147	63

The evaluation of the proposed approach is conducted on the OSAC source corpus. For each category (i.e., health, history, sport, etc.), some of its contents are randomly paraphrased. This dataset contains 1020 sentence pairs divided into 720 pairs for training and 300 pairs for testing. Each one is annotated with a relatedness label [1, 6] corresponding to the average relatedness judged by different individuals. Its effectiveness is tested against the standard reference corpus SemEval. It contains a monolingual Arabic paraphrased corpus comprising 250 pairs of sentences with their corresponding semantic similarity scores ranging [1, 5]. The experiments are carried out on the following train and test corpora as illustrated in Table 2:

Parameter's Settings

GloVe and CNN models are employed to perform the proposed method. Table 3 details their parameter settings:

Results and Evaluation

In this section, the impact of the attention mechanism with CNN is studied in terms of precision, recall and F1 score as represented in Figure 2

Table 3. Parameter settings.

Models	Parameters	Values
GloVe	Size of co-occurrence matrix	1,119,436 × 1,119,436 words
	Vector dimension	300
	Window size	3
	Min_count	20
	Learning rate	0.05
	Batch size	512
CNN	Filters number	64
	Kernel size	3
	Activation function	ReLU
	Pooling size	4
	Dropout	0.5

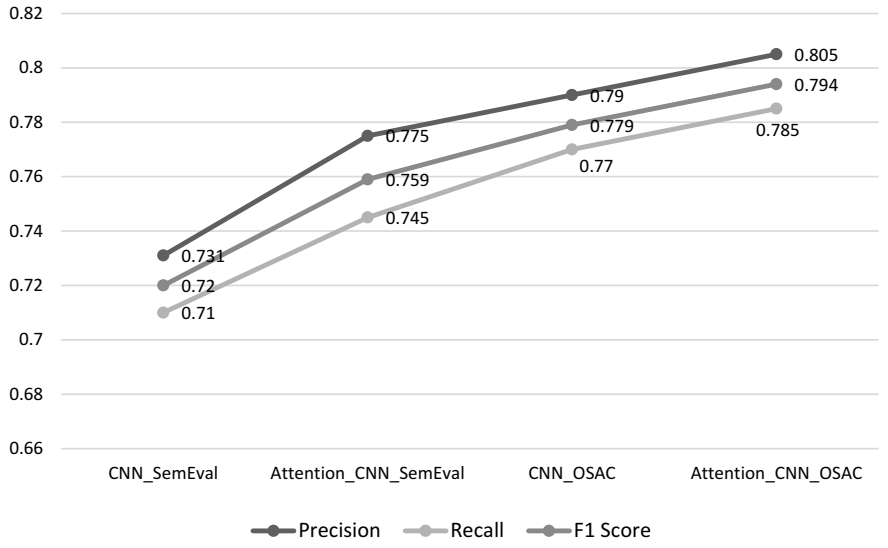

Figure 2. Experimental results of the proposed models.

As illustrated in Figures 3 and Figures 4, the configuration of the window size and pooling has an influence on the final experimental results of Attention-CNN model based on OSAC dataset. The highest precision of 0.790 and recall of 0.770 are obtained when using the window size $w_s w_s$ = 3 and when having a max-pooling operation. The window size was efficient to capture the contextual information and detect morpho-syntactic properties of Arabic sentences. Moreover, max pooling was useful for producing the most common and significant features than min and mean operations. Furthermore, the results are improved through the application of the attention mechanism with 0.805 precision, 0.785 recall and 0.794 F1 score.

Compared to the state-of-the-art-based methods, the performance results are displayed in Table 4 and Figure 5. Overall experiments demonstrated the effectiveness of the proposed approach.

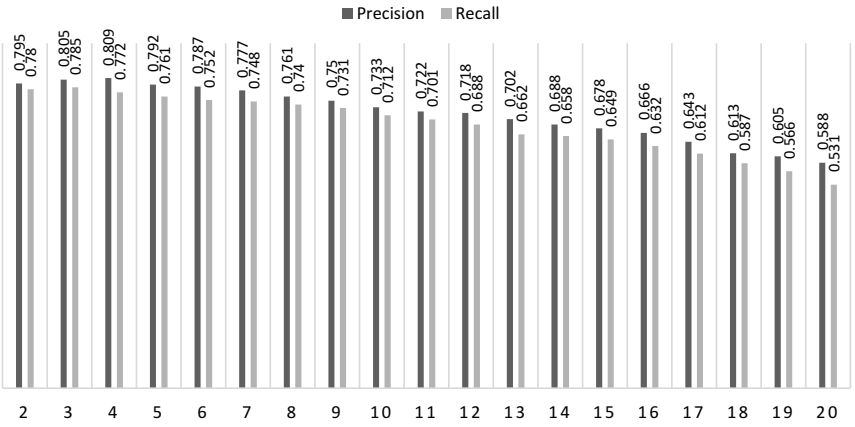

Figure 3. Attention-CNN-based approach performances according to the window sizes using OSAC corpus.

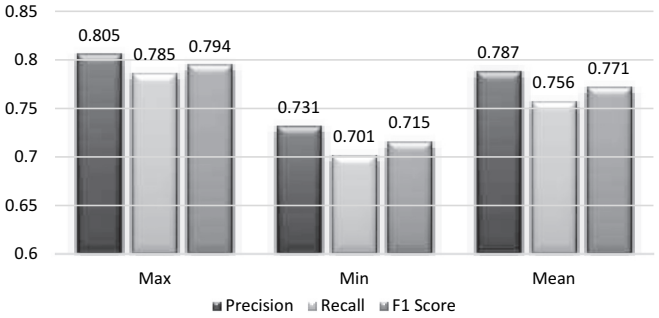

Figure 4. Attention-CNN-based Approach performances according to the pooling operations using OSAC Corpus.

Table 4. Comparison between the proposed approach and the state-of-the-art-based methods.

Models	Dataset	Model	Precision	Recall	F1 score
Ours	SemEval	CNN	0.731	0.710	0.720
		Attention CNN	0.775	0.745	0.759
	Paraphrased OSAC	CNN	0.790	0.770	0.779
		Attention CNN	0.805	0.785	0.794
Xie et al. (2020)	817,216 codes sources pairs	Word2vec + Cosine	0.790	0.420	0.550
		WICE	0.690	0.600	0.640
		WICE-SNN	0.670	0.830	0.740

As demonstrated by Xie et al. (2020), experimental results are improved when using word2vec to obtain real-valued word vectors and compute the cosine similarity based on their average weights. It achieved 0.790 precision, 0.420 recall and 0.55 F1 score higher than TF-IDF weights-based model. Thereafter, the authors combined them into a Siamese Neural Network (SNN), called Word Information for Code Embedding (WICE-SNN). It mapped codes into continuous space vectors and captured their semantic

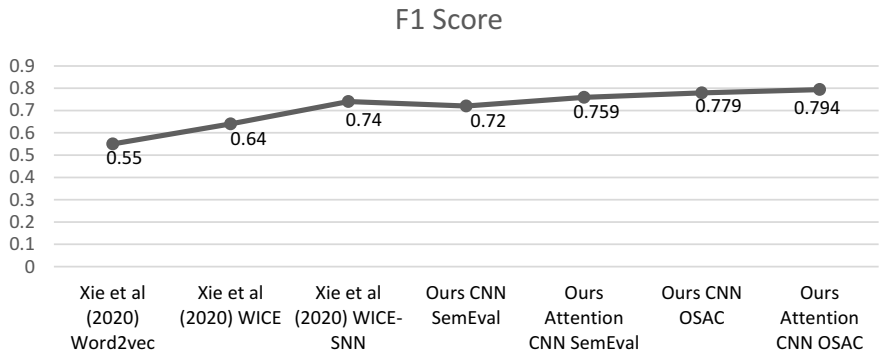

Figure 5. Summary of experimental results according to F1 score.

meaning. First, a word2vec model based on Continuous Bag Of Words (CBOW) algorithm was employed for features extraction. The weights of the series were fitted by the Term Frequency-Inverse Document Frequency (TF-IDF). Then, a CNN-SNN model was constructed to learn the semantic vector representation of code snippets. It was based on a cosine function for measuring the similarity score between pairs of code snippets. This method achieved the highest results in terms of 0.670 precision, 0.830 recall and 0.740 F1 score.

Overall experiments demonstrated that the application of GloVe with attentional CNN improved results with 0.775 precision, 0.745 recall and 0.759 F1 score using SemEval dataset. These values were further increased using the generated paraphrased OSAC corpus (0.805 precision, 0.785 recall and 0.794 F1 score), for the following reasons: First, GloVe was more beneficial than word2-vec for capturing the contextual relations between words taking into account the context of words and the information of the global corpus. Then, the use of the outputs of CNN in an attention model has improved the quality of the returned vectors. It captured the most useful local features from the generated vectors according to a given context. It could select the important words in several sentence-pairs. Furthermore, the use of cosine similarity was efficient for detecting semantic similarity between pairs of sentences.

Conclusion and Future Work

An Arabic paraphrase detection system is proposed combining the advantages of the feed-forward model and the attention mechanism. CNN was efficient in capturing salient contextual features and binary classification. Thus, the use of their outputs in an attention model has improved the quality of the returned vectors. It was useful in distinguishing the most important words representing the meaning of the sentence. The similarity score between sentences was subsequently computed by applying the cosine measure. We have semi-automatically developed a paraphrased

corpus and judged it manually. POS and local word embedding (word2vec) were efficient in conserving the morpho-syntactic properties of sentences with other semantically similar words. To validate its quality, the SemEval benchmark was used. The overall results and evaluation have denoted that our suggested methodology has achieved a promising performance compared to the state-of-the-art with 0.794 F1 score. For future work, we will improve the accuracy of our approach as well as the long-term dependencies. We will study the effectiveness of recurrent neural network-based methods and how they can lead to a better attention-based models.

Notes

1. http://aracorpus.e3rab.com/
2. https://fr.wikipedia.org/wiki/Wikip%C3%A9dia_en_arabe

Disclosure statement

No potential conflict of interest was reported by the author(s).

ORCID

Adnen Mahmoud http://orcid.org/0000-0003-2425-6230

References

Abdellaoui, H., and M. Zrigui. 2018. Using tweets and emojis to build TEAD: An Arabic dataset for sentiment analysis. *Computación y Sistemas* 22 (3):777–86. doi:10.13053/cys-22-3-3031.

Alrabiah, M., A. Alsalman, E. Atwell, and N. Alhelewh. 2014. KSUCCA: A key to exploring Arabic historical linguistics. *International Journal of Computational Linguistics (IJCL)* 5 (2):27–36.

Alzahrani, S. 2015. Arabic plagiarism detection using word correlation in N-Grams with K-overlapping approach. In *Working Notes for PAN-ArabPlagDet at FIRE*, 123–25. Gandhinagar.

Alzahrani, S., and N. Salim. 2008. Plagiarism detection in Arabic scripts using fuzzy information retrieval. In *Student Conference on Research and Development*, 281–85. Johor Bahru, Malaysia.

Alzahrani, S., and N. Salim. 2010. Fuzzy semantic-based string similarity for extrinsic plagiarism detection Lab report for PAN at CLEF 2010. In *Conference on Multilingual and Multimodal Information Access Evaluation*. Padua.

Batita, M. A., and M. Zrigui. 2018. Derivational relations in Arabic wordnet. In *9th Global WordNet Conference (GWC)*, 137–44. Singapore.

Bsir, B., and M. Zrigui. 2018. Enhancing deep learning gender identification with gated recurrent units architecture in social text. *Computación y Sistemas* 22 (3):757–66. doi:10.13053/cys-22-3-3036.

Cosma, G. 2011. An approach to source-code plagiarism detection and investigation using latent semantic analysis. *IEEE Transactions on Computers* 61 (3):379–94. doi:10.1109/TC.2011.223.

Dey, R., and M. S. Fathi. 2017. Gate-variants of gated recurrent unit (GRU) neural networks. In *IEEE 60th international midwest symposium on circuits and systems (MWSCAS)*, 1597–600. USA.

Du, J., L. Gui, and R. Xu. 2017. A convolutional attentional neural network for sentiment classification. In *International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, Shenzhen, China, 445–50.

Ezzikouri, H., M. Errital, and M. Oukessou. 2017. Fuzzy-semantic similarity for automatic multilingual plagiarism detection. *International Journal of Advanced Computer Science and Applications (IJACSA)* 8 (9):86–90.

Haffar, N., E. Hkiri, and M. Zrigui. 2020. Enrichment of Arabic TimeML corpus. In *International Conference on Computational Collective Intelligence (ICCCI)*, 655–67. Da Nang, Vietnam.

He, H., K. Gimpel, and J. Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Conference on empirical methods in natural language processing*, 1576–86. Pennsylvania.

Hkiri, E., S. Mallat, and M. Zrigui. 2020. Semantic and contextual enrichment of Arabic query leveraging NLP resources and association rules model. International Business Information Management Association (IBIMA), Granada, Spain.

Johnson, R., and Z. Tong. 2014. Effective use of word order for text categorization with convolutional neural networks. *arXiv Preprint arXiv* 1412:1058.

Ma, C., P. Kang, B. Wu, Q. Wang, and X. Liu. 2019. Gated attentive-autoencoder for content-aware recommendation. In *12th ACM International Conference on Web Search and Data Mining*, 519–27. Australia.

Mahmoud, A., and M. Zirgui. 2017. Semantic similarity analysis for paraphrase identification in Arabic texts. In *31st Pacific Asia Conference on Language, Information and Computation (PACLIC)*, 274–81. Philippine.

Mahmoud, A., and M. Zrigui. 2018. Artificial method for building monolingual plagiarized Arabic corpus. *Computacion Y Systemas* 22 (3):767–76.

Mahmoud, A., and M. Zrigui. 2019. Sentence embedding and convolutional neural network for semantic textual similarity detection in Arabic language. *Arabian for Engineering and Science Journal* 44 (11):9263–74. doi:10.1007/s13369-019-04039-7.

Mahmoud, A., and M. Zrigui. 2021a. Semantic similarity analysis for corpus development and paraphrase detection in Arabic. *International Arab Journal of Information Technology (IAJIT)* 18 (1):1–7.

Mahmoud, A., and M. Zrigui. 2021b. BLSTM-API: Bi-LSTM recurrent neural network-based approach for Arabic paraphrase identification. *Arabian for Engineering and Science Journal* 46 (4):4163–74. doi:10.1007/s13369-020-05320-w.

Maraoui, M., N. Terbeh, and M. Zrigui. 2018. Arabic discourse analysis based on acoustic, prosodic and phonetic modeling: Elocution evaluation, speech classification and pathological speech correction. *International Journal of Speech Technology* 21 (4):1071–90. doi:10.1007/s10772-018-09566-6.

Mueller, J., and T. Aditya. 2016. Siamese recurrent architectures for learning sentence similarity. In *AAAI Conference on Artificial Intelligence*, 2786–92. Arizona USA.

Nagoudi, E. B., A. Khorsi, H. Cherroun, and D. Schwab. 2018. A two-level plagiarism detection system for Arabic documents. *Cybernetics and Information Technologies* 18 (1):1–17. In Press.

Pontes, E. L., S. Huet, A. C. Linhares, and J. Torres-Moreno. 2018. Predicting the semantic textual similarity with Siamese CNN and LSTM. *arXiv E-prints10641* 1810 (3):1810.

Saad, M., and W. Ashour. 2010. OSAC: Open source Arabic corpora. In *6th ArchEng Internaional Symposiums on Electrical and Electronics Egineering and Computer Science (EEECS)*, 1–6. Lefke, North Cyprus.

Shajalal, M., and M. Aono. 2018. Semantic textual similarity in Bengali text. In *International Conference on Bangla Speech and Language Processing (ICBSLP)*, 1–5. New Jersey.

Ullah, F., J. Wang, M. Farhan, S. Jabbar, Z. Wu, and S. Khalid. 2020. Plagiarism detection in students' programming assignments based on semantics: Multimedia e-learning based smart assessment methodology. *Multimedia Tools and Applications* 79 (13–14):8581–98. doi:10.1007/s11042-018-5827-6.

Xie, C., X. Wang, C. Qian, and M. Wang. 2020. A source code similarity based on Siamese neural network. *Applied Sciences* 10 (21):1–12. doi:10.3390/app10217519.

Yao, L., Z. Pan, and H. Ning. 2019. Unlabeled short text similarity with LSTM encoder. *IEEE Access* 7 (1):3430–37. doi:10.1109/ACCESS.2018.2885698.

Zuo, F., X. Li, P. Young, L. Luo, Q. Zeng, and Z. Zhang. 2018. Neural machine translation inspired binary code similarity comparison beyond function pairs. In *Network and Disctributed Systems Security (NDSS) Sympsium*, San Diego, California, 1–15.