A deep learning based technique for plagiarism detection: a comparative study

El Mostafa Hambi, Faouzia Benabbou
Information Technology and Modeling Laboratory Science, Faculty of science Ben M'sik, Hassan II University of Casablanca, Morocco

Article Info

Article history:

Received Aug 27, 2019
Revised Oct 20, 2019
Accepted Nov 5, 2019

Keywords:

Deep Learning
Doc2vev
Neural network
Plagiarism
Pre-processing
Word2vec

ABSTRACT

The ease of access to the various resources on the web enabled the democratization of access to information but at the same time allowed the appearance of enormous plagiarism problems. Many techniques of plagiarism were identified in the literature, but the plagiarism of idea steels the foremost troublesome to detect, because it uses different text manipulation at the same time. Indeed, a few strategies have been proposed to perform the semantic plagiarism detection, but they are still numerous challenges to overcome. Unlike the existing states of the art, the purpose of this study is to give an overview of different proposition for plagiarism detection based on the deep learning algorithms. The main goal of these approaches is to provide high quality of worlds or sentences vector representation. In this paper, we propose a comparative study based on a set of criterions like: Vector representation method, Level Treatment, Similarity Method and Dataset. One result of this study is that most of researches are based on world granularity and use the word2vec method for word vector representation, which sometimes is not suitable to keep the meaning of the whole sentences. Each technique has strengths and weaknesses; however, none is quite mature for semantic plagiarism detection.

Corresponding Author:

El Mostafa Hambi,
Information Technology and Modeling Laboratory Science, Faculty of science Ben M'sik,
Hassan II University of Casablanca,
Casablanca 20000, Morocco.
Email: hambimostafa91@gmail.com

1. INTRODUCTION

The advancement of information technology (IT) and particularly the Web has impressively expanded the accessibility of data and leads thus to the rising of plagiarism. Plagiarism is a practice of taking someone else's work or ideas and passing them off as one's own. Several plagiarism techniques are performed by some dishonest authors, and here bellow some of them [1-2]:

- Copy-paste, textually (word by word): the content of the text is copied from one or more sources and could be slightly modified.
- Paraphrasing: the grammar of the text is changed the, words are changed by their synonyms. The sentences are reorganized from the original work and some parts of the text are deleted.
- False references, references are changed and sometimes are false or that do not even exist.
- Plagiarism with translation, the contents are translated and used without reference to the original work.

– Plagiarism of ideas, it is the most difficult plagiarism to detect because it is more complicated than the previous types, because it is not simple manipulations made on the text, but a more advanced form which could include all the other techniques.

In general, we can classify the plagiarism techniques on three strategies: lexical, syntaxial and semantic methods. The plagiarism of ideas most often incorporates reformulations as well as semantic and lexical changes which make it very hard to detect [3]:

The Lexical methods consider text as a sequence of characters or terms [4]. The pre-processing technique includes tokenization, lowercasing, punctuation removal and stemming [5]. The more common terms the documents have, the more similar they are. Methods such as longest common subsequence, n-grams and fingerprint are considered as this kind of methods. The comparison units adopted include words, sentences, human defined sliding window or an n-gram [6-12]. The Syntactical methods use text's syntactical units for comparing the similarity between documents. Implicitly, we consider that similar documents would have similar syntactical structure. This method makes use of characteristics such as POS tag to compare the similarity between different documents [13]. The Semantic methods use a semantic similarity for comparing documents. In this approach, different semantic features which include (Synonyms, hyponyms, hypernyms, semantic dependencies) [2-3] are extracted from the source documents and then used to trace out the plagiarism case from the corpus. The plagiarism detection is considered as a part of Natural Language Processing (NLP). Hence, based on NLP techniques many solutions have been proposed for lexical or Syntactical plagiarism, and most are based on the concept extraction using a corpus like WordNet [14-16].

With the classical approaches, two documents that share the same words are considered similar, and the word order is not respected which will make loss of the true meaning of a document. In recent years, deep learning techniques have been the subject of several researches and in different domains, from pattern recognition to NLP problems. The high performance obtained are very encouraging and make it possible to consider the use of these techniques in the field of plagiarism detection [17-18]. The techniques based on Deep Learning for plagiarism detection, include not only the contextual (semantic) level of the document but olso the syntactical and lexical level in vector representation. The remainder of this paper is organized as follows. The first section presents background concept. The second section defines related work. The third section contains a deep analyse concerning our comparison study. The last section introduces the conclusion and future work.

2. RESEARCH METHOD

In this section we will mention the different techniques used by the plagiarism detection approaches, whether in terms of its representation of its texts or the methods those calculate the similarity:

a. Neural network based models

Word embeddings are a type of word representation which stores the contextual information in a low-dimensional vector. This approach gained extreme popularity with the introduction of Word2Vec in 2013, groups of models to learn the word embeddings in a computationally efficient way. And Doc2Vec can be seen an extension of Word2Vec whose goal is to create a representational vector of a document or paragraph.

Word2vec: is a model using neural network used to produce a distributed representation of word. Some researcher says that is not deep learning technique, because it is simple bi-layered neural network architecture. This model is shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus [19].

Doc2vec: Doc2vec is an unsupervised algorithm to generate vectors representation of sentences, paragraphs and documents [20]. Its model is based on Word2Vec, with only adding another vector (paragraph ID) to the input. The architecture of Doc2Vec model is shown Figure 1.

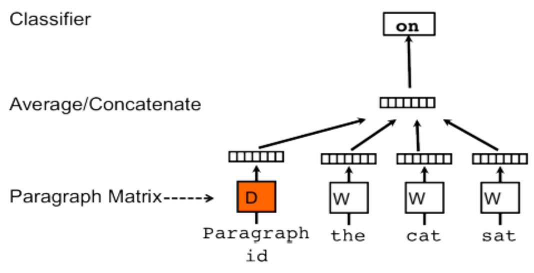

Figure 1. Doc2vec architecture

Instead of using just nearby words to predict the word, we also added another feature vector, which is document-unique.

b. Deep learning based models

Deep learning is a set of learning methods attempting to model data with complex architectures combining different non-linear transformations. The elementary bricks of deep learning are the neural networks that are combined to form the deep neural networks. There exist several types of architectures for neural networks:

Recursive neural networks (RNN): have been successful, for instance, in learning sequence and tree structures in natural language processing, mainly phrase and sentence continuous representations based on word embedding [21].

Siamese LSTM for Learning documents Similarity: LSTM is a king of recurrent neural network and it is great when we have an entire sequence of words or sentences. This is because RNNs can model and remember the relationships between different words and sentences. Manhattan LSTM models have two networks LSTMleft and LSTMright which process one of the sentences in a given pair independently. Siamese LSTM, a version of Manhattan LSTM where both LSTMleft and LSTMright have same tied weights such that LSTMleft = LSTMright. Such a model is useful for tasks like duplicate query detection and query ranking. Here, duplicate detection task is performed to find if two documents are similar or not. Similar model can be trained for query ranking using hit data for a given query and its matching results as a proxy for similarity [21].

Convolutional neural network: CNN is a class of deep, feed-forward artificial neural networks that uses a variation of multilayer perceptions designed to require minimal preprocessing. These are inspired by animal visual cortex. CNNs are generally used in computer vision; however, they have recently been applied to various NLP tasks like a text classification [21].

Deep Structured Semantic Model (DSSM): DSSM stands for Deep Structured Semantic Model, or more general, Deep Semantic Similarity Model. It is a deep neural network (DNN) modelling technique for representing text strings (sentences, queries, predicates, entity mentions, etc.) in a continuous semantic space and modelling semantic similarity between two text strings.

c. Other models

Other methods used to construct a vector representation of a given text can be found:

GLOVE: is an unsupervised learning algorithm for obtaining vector representations for words. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space [22].

InferSent: is a sentence embeddings method that provides semantic representations for English sentences. It is trained on natural language inference data and generalizes well to many different tasks [22].

d. Similarity methods

Finding similarity between elements is the core of sentence similarity. In the literature, there are many metrics for calculating similarity. This section shows different approaches used to calculate similarity between elements:

Cosine similarity: is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them. The cosine of 0° is 1, and it is less than 1 for any angle in the interval $[0, \pi]$ radians [23].

Jaccard index: also known as Intersection over Union and the Jaccard similarity coefficient (originally given the French name coefficient de community by Paul Jaccard), is a statistic used for gauging the similarity and diversity of sample sets. The Jaccard coefficient measures similarity between finite sample sets and is defined as the size of the intersection divided by the size of the union of the sample sets [23].

Euclidean Distance: refers to Euclidean distance. When data is dense or continuous, this is the best proximity measure. The Euclidean distance between two points is the length of the path connecting them, and it is obtained with the Pythagorean Theorem [23].

Longest common subsequence (LCS) method: consists of finding the longest subsequence common to all sequences in a set of sequences. The longest common subsequence problem is a classic computer science problem, the basis of data comparison programs such as the diff utility and has applications in computational linguistics and bioinformatics [24].

Word Mover's Distance (WMD): uses word embeddings to calculate the similarities, and precisely, it uses normalized Bag-of-words and word Embeddings to calculate the distance between documents [25].

3. RELATED WORK

Our study focuses on the detection of semantic plagiarism more precisely the identification of the plagiarism of ideas between two given texts, as illustrated below we dug on methods that detect this type of plagiarism:

In [26] proposed a plagiarism detection system, which rely on use sentences comparison in two phases. They first extract word vectors by word2vec algorithm, and then remove Persian stop words while text pre-processing. After that, for each sentence an average of all word vectors is calculated. After feature extraction, in phase 1, each sentence in a suspicious document is compared with all the sentences in the source documents. Cosine similarity is used as a comparison metric. After this step which helps to find the nearest sentences in real time, in phase 2, lexical similarity of two sentences is evaluated by the Jaccard similarity measure. Two sentences which pass Jaccard similarity threshold considered as plagiarism at final step. In [27] proposed the use word2vec model in order to compute vector of features for every word. They choose documents from the corpus itself, however the documents used for testing was processed and the pre-processing that was made is stop words removal. The similarity between vectors was computed by using cosine similarity. [24] The aim of this approach is evaluating the validity of using the distributed representation to define the word similarity. They introduce three methods based on the following three document similarities: for two documents: The length of the longest common subsequence (LCS) divided by the length of the shorter document, the local maximal value of the length of LCS, and the local maximal value of the weighted length of LCS. The distributed representation was obtained from no particular data by word2vec.

Another approach uses the principle of Deep Structured Semantic Model (DSSM) proposed by [28]. DSSM is a deep learning-based technique that is proposed for semantic understanding of textual data. It maps short textual strings, such as sentences, to feature vectors in a low-dimensional semantic space. Then the vector representations are utilized for document retrieval by comparing the similarity between documents and queries. After obtaining the semantic feature vectors for each paired snippets of text, cosine similarity is utilized to measure the semantic similarity between the pair. Similarly, with the previous methods, in [29] deep learning documents or texts can be represented as vectors by the using document to vector technique (doc2vec). And the detection of plagiarism will be done by a simple comparison between all sentences of each two documents analysed.

The approach proposed in [30] is based on converting a paragraph to vectors and it's inspired by the methods for learning the word vectors. The inspiration is that the word vectors are asked to contribute to a prediction task about the next word in the sentence. So, despite the fact that the word vectors are initialized randomly, they can eventually capture semantics as an indirect result of the prediction task. It will use this idea in their paragraph vectors in a similar manner. The paragraph vectors are also asked to contribute to the prediction task of the next word given many contexts sampled from the paragraph.

These approaches [29-30] are used to perform similarity detection between the document vectors but also use the cosine to compare the vectors. In paper [31] they represent each word w by a vector. It constructs these word vectors using GloVe. This approach uses the recursive neural networks algorithm to have a vector representation of a sentence and use the cosine for calculate the similarity. In [32] two input sentences are processed in parallel by identical neural networks, outputting sentence representations. The sentence representations are compared by the structured similarity measurement layer. The similarity features are then passed to a fully-connected layer for computing the similarity score. Cosine distance measures the distance of two vectors according to the angle between them. The use of cosine to detect similarity between sentences remains a solution that carries many risks. InferSent [22] is an NLP technique for universal sentence representation developed by Facebook that uses supervised training to produce high transferable representations. They used a Bi-directional LSTM with attention that consistently surpassed many unsupervised training methods such as the SkipThought vectors. They also provide a Pytorch implementation that they used to generate sentence embedding. So, this approach needs to define a similarity measure to compare two vectors, and for that goal, it'll be the cosine similarity.

The authors in [33] used word embedding, vector representations of terms, computed from unlabelled data, that represent terms in a semantic space in which proximity of vectors can be interpreted as semantic similarity. They propose to go from word-level to text-level semantics by combining insights from methods based on external sources of semantic knowledge with word embedding. They derive multiple types of meta-features from the comparison of the word vectors for short text pairs, and from the vector means of their respective word embedding. The features representing labelled short text pairs are used to train a supervised learning algorithm. In [25] present the Word Mover's Distance (WMD), a novel distance function between text documents. This work is based on recent results in word embedding that learn semantically meaningful representations for words from local co-occurrences in sentences. The WMD distance measures the dissimilarity between two text documents as the minimum amount of distance that the embedded words of one document need to "travel" to reach the embedded words of another document. This article [34] proposed an innovative word embedding-based system devoted to calculating the semantic similarity in Arabic sentences. The main idea is to exploit vectors as word representations in a multidimensional space in order to capture the semantic and syntactic properties of words. IDF weighting and Part-of-Speech tagging are applied on the examined sentences to support the identification of words that are highly descriptive in each sentence.

In paper [35] they address the issue of finding an effective vector representation for a very short text fragment. By effective they mean that the representation should grasp most of the semantic information in that fragment. For this, they use semantic word embedding to represent individual words, and we learn how to weigh every word in the text through the use of tf-idf (term frequency-inverse document frequency) information to arrive at an overall representation of the fragment comparing two tf-idf vectors is done through a standard cosine similarity. [36] This paper investigates the effectiveness of several such naive techniques, as well as traditional tf-idf similarity, for fragments of different lengths. This main contribution is a first step towards a hybrid method that combines the strength of dense distributed representations-as opposed to sparse term matching-with the strength of tf-idf based methods to automatically reduce the impact of less informative terms. This approach outperforms the existing techniques in a toy experimental set-up, leading to the conclusion that the combination of word embedding and tf-idf information might lead to a better model for semantic content within very short text fragments. Between two such representations they then calculate the cosine similarity.

In the architecture proposed in [37], word embedding is first trained on API documents, tutorials, and reference documents, and then aggregated in order to estimate semantic similarities between documents where the similarity between vectors is usually defined as cosine similarity. In paper [38], they propose to combine explicit semantic analysis (ESA) representations and word2vec representations as a way to generate denser representations and, consequently, a better similarity measure between short texts. In [39] they proposed a semantic similarity approach for paraphrase identification in Arabic texts by combining different techniques of Natural Language Processing NLP such as: Term Frequency Inverse Document Frequency TF-IDF technique. The goal is to represent a word vector using word2vec. And also, to generate a sentence vector representation and after applying a similarity measurement operation based on different metrics of comparison, such as: Cosine Similarity and Euclidean Distance. This approach was evaluated on the Open Source Arabic Corpus OSAC and obtained a promising rate.

[40] This paper proposes a novel deep neural network-based approach that relies on coarse-grained sentence modelling using a convolutional neural network and a long short-term memory model, combined with a specific fine-grained word-level similarity matching model. In this component, they represent every sentence using their joint CNN and LSTM architecture. The CNN is able to learn the local features from words to phrases from the text, while the LSTM learns the long-term dependencies of the text. More specifically, they firstly take the word embedding as input to their CNN model, in which various types of convolutions and pooling techniques are applied to capture the maximum information from the text. Next, the encoded features are used as input to the LSTM network. Finally, the long-term dependencies learned by the LSTM becomes the semantic sentence representation.

[41] This approach proposes to explicitly model pairwise word interactions and present a novel similarity focus mechanism to identify important correspondences for better similarity measurement. They used GloVe word embeddings for vector representation of word and their model contains four major components: 1. Bidirectional Long Short-Term Memory Net-works (Bi-LSTMs) are used for context modeling of input sentences. 2. A novel pairwise word interaction modeling technique encourages direct comparisons between word contexts across sentences. Cosine distance (cos) measures the distance of two vectors by the angle between them, while L2Euclidean distance (L2Euclid) and dotproduct distance (DotProduct) measure magnitude differences. We use three similarity functions for richermeasurement. 3. A novel similarity focus layer helps the model identify important pairwise word interactions across sentences.4. A layer deep convolutional neural network (ConvNet) converts the similarity measurement problem into a pattern recognition problem for final classification.

The model of [42] is applied to assess semantic similarity between sentences. For these applications, they provide word-embedding vectors word2vec to the LSTMs, which use a fixed size vector to encode the underlying meaning expressed in a sentence (irrespective of the particular wording/syntax). By restricting subsequent operations to rely on a simple Manhattan metric, they compel the sentence representations learned by their model to form a highly structured space whose geometry reflects complex semantic relationships. [43] This paper proposes a model for com-paring sentences that uses a multiplicity of perspectives. We first model each sentence using a convolutional neural network that extracts features at multiple levels of granularity and uses multiple types of pooling. We then compare our sentence representations at several granularities using multiple similarity metrics (cos, LEuclid). We apply our model to three tasks, including the Microsoft Research paraphrase identification task and two SemEval semantic textual similarity tasks.

In this paper [44], they present convolutional neural network architecture for reranking pairs of short texts, where they learn the optimal representation of text pairs and a similarity function to relate them in a supervised way from the available training data. Their network takes only words in the input, thus requiring minimal preprocessing. In particular, they consider the task of reranking short text pairs where elements of the pair are sentences. They test our deep learning system on two popular retrieval tasks from TREC: Question Answering and Microblog Retrieval. [45] This system combines convolution and recurrent neural networks to measure the semantic similarity of sentences. It uses a convolution network to take account of the local context of words and an LSTM to consider the global context of sentences. This combination of networks helps to preserve the relevant information of sentences and improves the calculation of the similarity between sentences. According to this state of the art we have been able to detect the strengths and weaknesses of each approach that helped us to build our approach. The Table 1 represents a summary compared to the methods above:

Table 1. Comparative table

Approach	Vector representation		Level treatment	Similarity method	Dataset/resources	Critical
	Word	Sentence				
[26]	Word2vec	Average	sentence	Cosine, Jaccard	PAN 2016	Loss of the meaning of the sentence.
[27]	Word2vec	-	word	Cosine	OSAC Arabic corpus	Two documents share the same vectors could be non-plagiarized.
[32]	Word2vec	-	word	Cosine	Microsoft Research Paraphrase Corpus	The use of cosine to detect similarity between sentences remains a solution that carries many risks.
[33]	Word2vec	-	word	Cosine	Microsoft Research Paraphrase Corpus data set	Use cosine similarity to compute a similarity between sentences.
[34]	Word2vec	-	word	Cosine	Microsoft Research Video Description Corpus	
[35]	Word2vec	-	word	Cosine	Wikipedia dataset	
[36]	Word2vec tf-idf	-	word	Cosine	Wikipedia dataset	
[37]	Word2vec	-	word	Cosine	Wiki corpus	
[38]	Word2vec	-	word	Cosine	-	
[39]	Word2vec	-	word	Cosine Euclidean Distance	Arabic Corpus OSAC	
[24]	Word2vec	-	word	LCS	PAN 2013	LCS problem seeks a longest subsequence of every member of a given set of vectors, lose the semantic aspect.
[28]	Deep Structured Semantic Model (DSSM)	-	word	Cosine	SemEval 2015 English STS	The treatment is at the level of sentence or small texts.
[29]	-	Doc2vec	sentence	Cosine	-	Slowness of the system. The semantic aspect of a paragraph is lost because the comparison is done sentence by sentence.
[30]	-	Doc2vec	sentence	Cosine	-Stanford sentiment treebank dataset - IMDB dataset	
[31]	GloVe	Recursive neural networks	sentence	Cosine	SemEval-2015 Task 2	Use of doc2vec is better then uses RNN. The semantic aspect of a paragraph is lost.
[22]	Word2vec	InferSent	sentence	Cosine	-	The use of cosine to detect similarity between sentences remains a solution that carries many risks. The comparison is done at the sentence level, so we always encounter the problem of loss of the semantic aspect of the paragraph or text analysed. According to the study done by [31], [32] he found that the use of doc2vec gives trampling results.

Table 1. Comparative table *(Continue)*

Approach	Vector representation		Level treatment	Similarity method	Dataset/resources	Critical
	Word	Sentence				
[25]	Word2vec	-	word	WMD	BBCSPOR-T	This method is used just to detect the similarity between small sentences.
[40]	Glove	-	word	CNN-RNN	SemEval 2015	These methods are based on the vector representation of words, so they are used only for the detection of similarity between sentences but not texts.
[45]	Word2vec	-	word	CNN-LSTM	SICK	
[41]	Glove	-	word	Lstm Cnn DotProduct L2Euclid	-2014 SemEval -Microsoft Video Paraphrase Corpus -WikiQA	
[42]	Word2vec	-	word	LSTM	SICK	
[43]	Word2vec	-	word	CNN	-SemEval -Microsoft Research paraphrase	Always we encounter the problem of level representation of the analysed data; the representation by word poses the problem that we can just analyse the small sentences. CNN's use of treating the similarity between list of word poses several problems like the loss of semantics level of the sentence construct.
[44]	Word2vec	-	word	CNN	TREC : Answering and Microblog Retrieval	

In addition to that we could detect the most powerful methods used for the representation of a text. It has been found that the use of the doc2vec principle remains the most relevant solution from the [29-30] study, and then we went further and took inspiration from it to build our learning system that detects plagiarism between the documents.

4. RESULTS AND DISCUSSION

In this part we will analyse the results found in the study carried out above, first we will illustrate the most important comparison criteria defined:

Vector representation: This is a treatment performed on a text that will transform it to list of vectors which keep the semantic and syntactic aspect offered by the use of deep learning algorithms.

Level treatment: this criterion defines the level of the treatment of a text, more exactly if the text is treated by word or by sentence.

Similarity method: This part deals with the approaches used for calculating the similarity between the vectors that represent the texts, which will give us a global visibility to detect the strengths and weaknesses of each method. In addition, we are going to talk about the critical point for each approach illustrated in the paragraph above. Starting from the methods used for the vector representation of a text, according to the analysis it turns out that most of the approaches use either the word2vec or the doc2vec for its vector transformation, so we distinguish that the mikolov representations are the best methods used to keep the semantic aspect of a given text. In Revenge, Each Approach treats the text with its own way, there are some who transform it into a list of words and someone into a list of sentences, these representations yield results that differ from one approach to another but the transformation of a text to a list of sentences in our opinion remains the most relevant since the meaning of the text treated remains in consideration. With regard to the methods used for the similarity calculation, the preceding paragraphs mention the different ways used to detect whether there is a similarity or not between the analysed texts. There are also many approaches that work with CNN and RNN on its plagiarism detection architecture, but most of them use the word level for its vector representation, so they are used only for the detection of similarity between sentences but not texts.

In conclusion, we found that almost of these approaches use the cosine to calculate the similarity between documents, so it was found that these methods perform its similarity analyses in word-by-word or sentence-by-sentence, which will pose after reliability problem of these results, since we can find two documents that share the same word or the same sentences but they are not semantically similar, in addition to that we can lose the semantic aspect when the documents are treating via a list of sentences or words. So, you have to think of a method that manages this problem by proposing an approach that will represent a text by a list of sentences that will eventually be transformed into a list of vectors, and in addition to that we must use a treatment that keeps the semantic aspect of this list of sentences, so it well be a manipulation that processes a list of sentences to detect a similarity using an algorithm like the RNN that will keep the semantic aspect of a text.

5. CONCLUSION

In this paper, we have mentioned many different methods used in detection of plagiarism of ideas that stand for the principal of Deep Learning, and by this brilliant study we could construct our critical base of the previous weaknesses which we have seen during our study. This helped us to get a general idea about the different methods of deep learning used for plagiarism detection or especially semantic plagiarism detection. In addition to this, this study has given us the paths to follow for the construction of our approach by benefiting from the strengths of each method and bypassing the weak points of each method. Concerning the future work consists of construct and putting into practice our approach and comparing it with the other methods used at the level of the phase related work.

REFERENCES

[1] Tuomo Kakkonen, Maxim Mozgovoy. Hermetic and Web Plagiarism Detection Systems for Student Essaysan Evaluation Of The State-Of-The-Art. *Journal of Educational Computing Research*, v42 n2 p135-159 2010. University of Joensuu, Finland, University of Aizu, Japan [en ligne] 2010.

[2] Ahmed Jabr Ahmed Muftah. Document Plagiarism Detection Algorithm Using Semantic Networks. A project report submitted in partial fulfillment of the requirements for the award of the degree of Master of Science (Computer Science). Faculty of Computer Science and Information Systems University Technology Malaysia (2009).

[3] Erfaneh Gharavi, Kayvan Bijari et Kiarash ZahirniaA Deep Learning Approach to Persian Plagiarism Detection. *Journal of Machine Learning Research* (2011).

[4] S. M. Alzahrani, N. Salim, and A. Abraham, "Understanding plagiarism linguistic patterns, textual features, and detection methods," Trans. Sys.Man Cyber Part C, vol. 42, no. 2, pp. 133–149, Mar. 2012. [Online]. Available: http://dx.doi.org/10.1109/TSMCC.2011.2134847.

[5] M. Chong and L. Specia, "Lexical generalisation for word-level matching in plagiarism detection," in RANLP, 2011, pp. 704–709.

[6] S. Brin, J. Davis, and H. Garcia-Molina, "Copy detection mechanisms for digital documents," in SIGMOD Conference, 1995, pp. 398–409.

[7] D. R. White and M. Joy, "Sentence-based natural language plagiarism detection," *ACM Journal of Educational Resources in Computing*, vol. 4, no. 4, pp. 1–20, 2004.

[8] S. Niezgoda and T. P. Way, "Snitch: a software tool for detecting cut and paste plagiarism," in SIGCSE, 2006, pp. 51–55.

[9] A. Barr´ on-Cedeno and P. Rosso, "On automatic plagiarism detection based on n-grams comparison," in ECIR, 2009, pp. 696–700.

[10] M. S. Pera and Y.-K. Ng, "A naıve bayes classifier for web document summaries created by using word similarity and significant factors," *International Journal on Artificial Intelligence* Tools, vol. 19, no. 4, pp. 465–486, 2010.

[11] E. Stamatatos, "Plagiarism detection using stopword n-grams," JASIST, vol. 62, no. 12, pp. 2512–2527, 2011.

[12] J. Grman and R. Ravas, "Improved implementation for finding text similarities in large sets of data-notebook for pan at clef 2011," in CLEF (Notebook Papers/Labs/Workshop), 2011.

[13] Uzuner, O., and Katz, B., and Nahnsen, T.: *Using Syntactic Information to Identify Plagiarism*. In: 2nd Workshop on Building Educational Applications using NLP (2005).

[14] Ahmed Hamza Osman, Naomie Salim, and Albaraa Abuobieda. Survey of Text Plagiarism Detection. Computer Engineering and Applications Vol. 1, No. 1, June 2012. Universiti Teknologi Malaysia, Faculty of Computer Science and Information Systems, Skudai, Johor, Malaysia, International University of Africa, Faculty of Computer Studies, Khartoum, Sudan.

[15] Vani Kanjirangat, Deepa Gupta. A Study on Extrinsic Text Plagiarism Detection Techniques and Tools. Journal of Engineering Science and Technology Review 9 (5) (2016) 9 – 23. Department of Computer Science & Engineering, Amrita School of Engineering, Amrita University, Amrita Vishwa Vidyapeetham, Bangalore, India. Department of Mathematics, Amrita School of Engineering, Amrita University, Amrita Vishwa Vidyapeetham, Bangalore, India.

[16] Christina Kraus. Plagiarism Detection-State-of-the-art systems (2016) and evaluation methods. arXiv:1603.03014v1 [cs.IR] 8 Mar 2016. Technische Universität Berlin Database Systems and Information Management Group.

[17] Collobert, R. and Weston, J. *A unified architecture for natural language processing: Deep neural networks with multitask learning*. In Proceedings of the 25th international conference on Machine learning ACM, 160-167 (2008).

[18] Chong, M.Y.M.,. A study on plagiarism detection and plagiarism direction identification using natural language processing techniques (2013).

[19] Mikolov, T., Chen, K., Corrado, G., and Dean, J., 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

[20] Quoc Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. Google Inc, 1600 Amphitheatre Parkway, Mountain View, CA 94043.

[21] Shaojie Bai, J. Zico Kolter, Vladlen Koltun. An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling. arXiv:1803.01271v2 [cs.LG] 19 Apr 2018.

[22] Christian S. Perone. Privacy-preserving sentence semantic similarity using InferSent embeddings and secure two-party computation.

[23] Mamdouh Farouk. Measuring Sentences Similarity: A Survey. Indian Journal of Science and Technology, Vol 12(25), DOI: 10.17485/ijst/2019/v12i25/143977, July 2019. Department of Thermotechnik Computer Science, Assiut University, Markaz El-Fath, Assiut Governorate 71515, Egypt.

[24] Kensuke Baba, Tetsuya Nakatoh and Toshiro Minami. *Plagiarism detection using document similarity based on distributed representation.* 8th International Conference on Advances in Information Technology, IAIT2016, 19-22 December 2016, Macau, China. Fujitsu Laboratories, Kawasaki, Japan Kyushu University, Fukuoka, Japan.

[25] Matt J. Kusner, Yu Sun, Nicholas I. Kolkin and Kilian Q. *Weinberger. From Word Embeddings To Document Distances*. Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 2015. JMLR: W&CP volume 37. Washington University in St. Louis, 1 Brookings Dr., St. Louis, MO 63130.

[26] Erfaneh Gharavi, Kayvan Bijari and Kiarash Zahirnia. *A Deep Learning Approach to Persian Plagiarism Detection*. DOI: 10.1109/ICTCS.2017.42 Conference: Conference: The International Conference on new Trends in Computing Sciences (ICTCS2017). University of Tehran Faculty of new Science and Technology Data & Signal processing Lab 2017.

[27] Dima Suleiman, Arafat Awajan and Arafat Awajan. Deep Learning Based Technique for Plagiarism Detection in Arabic Texts. 2017 International Conference on New Trends in Computing Sciences. Computer Science Department Princess Sumaya University for Technology 2017.

[28] Naveed Afzal, Yanshan Wang and Hongfang Liu. MayoNLP at SemEval-2016 Task 1: Semantic Textual Similarity based on Lexical Semantic Net and Deep Learning Semantic Model. Proceedings of SemEval-2016, pages 674–679, San Diego, California, June 16-17, 2016. 2016 Association for Computational Linguistics. Department of Health Sciences Research Mayo Clinic, Rochester, MN.

[29] Tedo Vrbanec and Ana Mestrovic. The Struggle with Academic Plagiarism: Approaches based on Semantic Similarity. MIPRO 2017, May 22- 26, 2017, Opatija, Croatia. Faculty of Teacher Education, University of Zagreb, Croatia Department of [nformatics, University of Rijeka, Croatia.

[30] Quoc Le and Tomas Mikolov. Distributed Representations of Sentences and Documents. Google Inc, 1600 Amphitheatre Parkway, Mountain View, CA 94043.

[31] Adrian Sanborn and Jacek Skryzalin. Deep Learning for Semantic Similarity. MIPRO 2017, May 22- 26, 2017, Opatija, Croatia. Department of Computer Science Stanford University.

[32] Hua He, Kevin Gimpel,and Jimmy Lin. Multi-Perspective Sentence Similarity Modeling with Convolutional Neural Networks. DOI: 10.18653/v1/D15-1181 Conference: Conference: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. Department of Computer Science, University of Maryland, College Park, Toyota Technological Institute at Chicago and David R. Cheriton School of Computer Science, University of Waterloo.

[33] Tom Kenter and Maarten de Rijke. *Short Text Similarity with Word Embeddings*. CIKM '15 Proceedings of the 24th ACM International on Conference on Information and Knowledge Management Pages 1411-1420. University of Amsterdam, Amsterdam, The Netherlands.

[34] El Moatez Billah Nagoudi and Didier Schwab. *Semantic Similarity of Arabic Sentences with Word Embeddings*. Proceedings of The Third Arabic Natural Language Processing Workshop (WANLP), pages 18–24, Valencia, Spain, April 3, 2017.©, 2017 Association for Computational Linguistic. LIM-Laboratoire d'Informatique et de Mathématiques, Université Amar Telidji de Laghouat, Algérie. LIG-GETALP Univ. Grenoble Alpes France.

[35] Cedric De Boom, Steven Van Canneyt, Thomas Demeester and Bart Dhoedt. Representation learning for very short texts using weighted word embedding aggregation. *Journal Pattern Recognition Letters* archive Volume 80 Issue C, September 2016 Pages 150-156. Department of Information Technology, Technologiepark 15, 9052 Zwijnaarde, Belgium.

[36] Cedric De Boom, Steven Van Canneyt, Steven Bohez, Thomas Demeester and Bart Dhoedt. Learning Semantic Similarity for Very Short Texts. *2015 IEEE International Conference on Data Mining Workshop (ICDMW).* Ghent University – iMinds Gaston Crommenlaan 8-201, 9050 Ghent, Belgium.

[37] Xin Ye, Hui Shen, Xiao Ma, Razvan Bunescu, and Chang Liu. From Word Embeddings To Document Similarities for Improved Information Retrieval in Software Engineering. CSE '16, May 14-22, 2016, Austin, TX, USA. School of Electrical Engineering and Computer Science, Ohio University Athens, Ohio 45701, USA.

[38] Yangqiu Song and Dan Roth. *Unsupervised Sparse Vector Densification for Short Text Similarity*. DOI: 10.3115/v1/N15-1138 Conference: Conference: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Department of Computer Science University of Illinois at Urbana-Champaign Urbana, IL 61801, USA.

[39] Adnen Mahmoud and Mounir Zrigui. *Semantic Similarity Analysis for Paraphrase Identification in Arabic Texts*. Conference: Conference: The 31st Pacific Asia Conference on Language, Information and Computation PACLIC 31 (2017), At University of the Philippines Cebu, Cebu, Philippines. LATICE Laboratory Research Department of Computer Science University of Monastir, Tunisia.

[40] Basant Agarwala, Heri Ramampiaroa, Helge Langsetha, Massimiliano Ruocco. A Deep Network Model for Paraphrase Detection in Short Text Messages. arXiv:1712.02820v1 [cs.IR] 7 Dec 2017. Dept. of Computer

Science, Norwegian University of Science and Technology, Norway Swami Keshvanand Institute of Technology, India Telenor Research, Trondheim, Norway.

[41] Hua He and Jimmy Lin. *Pairwise Word Interaction Modeling with Deep Neural Networks for Semantic Similarity Measurement*. Proceedings of NAACL-HLT 2016, pages 937–948, San Diego, California, June 12-17, 2016.c©2016 Association for Computational Linguistics. Department of Computer Science, University of Maryland, College Park David R. Cheriton School of Computer Science, University of Waterloo.

[42] Jonas Mueller and Aditya Thyagarajan. *Siamese Recurrent Architectures for Learning Sentence Similarity.* Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16). Computer Science& Artificial Intelligence Laboratory Massachusetts Institute of Technology. Department of Computer Scienceand EngineeringM. S. Ramaiah Institute of Technology.

[43] Hua He, Kevin Gimpel, and Jimmy Lin. *Multi-Perspective Sentence Similarity Modelingwith Convolutional Neural Networks.* Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 1576–1586, Lisbon, Portugal, 17-21 September 2015.c©2015 Association for Computational Linguistics. Department of Computer Science, University of Maryland, College Park2Toyota Technological Institute at Chicago. David R. Cheriton School of Computer Science, University of Waterloo.

[44] Aliaksei Severyn, Alessandro Moschitti. Learning to Rank Short Text Pairs with Convolutional DeepNeural Networks. Google Inc. Qatar Computing Research Institute.

[45] Elvys Linhares Pontes, Stéphane Huet, Andréa Carneiro Linhares, Juan-Manuel Torres-Moreno. Predicting the Semantic Textual Similarity with Siamese CNN and LSTM. LIA, Université d'Avignon et des Pays de Vaucluse, Avignon, 84000 France Universidade Federal do Ceará, Sobral, Ceará Brazil École Polytechnique de Montréal, Montréal, Canada.

BIOGRAPHIES OF AUTHORS

El Mostafa HAMBI is a Ph.D. student of Computer Science. His research areas include data

Faouzia Benabbou is a professor of Computer Science and member of Compute Science and Information Processing laboratory. She is Head of the team "Cloud Computing, Network and Systems Engineering (CCNSE)". She received his Ph.D. in Computer Science from the Faculty of Sciences, University Mohamed V, Morocco, 1997. His research areas include cloud Computing, data mining, machine learning, and Natural Language Processing. She has published several scientific articles and book chapters in these areas.