Comparison of Deep Learning Methods Used to Detect the Similarity Between Two Texts

El Mostafa HAMBI and Faouzia Benabbou

University of Hassan 2, Faculty of Sciences Ben M’sik, Casablanca, Morocco

Summary
Recently deep learning has proposed several methods used in several domains. Among these domains we found its use in the texts processing, since they have given relevant results at the level of text classification, detection of text similarity and translation detection, that’s why we will use these algorithms in our study. In this paper we will compare the different algorithms used to detect semantics similarity between two given texts. This comparison will give us a global vision to contribute a relevant system that can detect the different types of similarity proposed by a Corpus. In our study we will compare the use of siamese lstm which uses word2vec to have a vector representation of words and siamese lstm which uses doc2vec for the vector representation of sentences to perform the comparison between two texts.

Key words:
Deep Learning, Preprocessing; Doc2vev; Word2vec; neural network; Long short-term memory (LSTM); Convolutional neural network (Cnn); Siamese neural network.

1. Introduction

In this paper, we focus on the problem of detecting whether two text contain common semantic information, to do this a comparison was made between two approaches that use the principle of deep learning to detect similarity. The basis of our approach is the use of the Pan corpus to build in the first place our learning base which will contain the different types of plagiarism that will be able to detect them in its tests. And then we will use the siamese lstm algorithm to compare two documents to perform our learning system.

Lstm on the other hand is great when you have an entire sequence of words or sentences. This is because it can model and remember the relationships between different words and sentences. CNNs can only detect these relations but overall they model the sentence as a bag of words and lose context information and word ordering.

In fact, we did a study to get an idea of the representation that we need to put in place for our documents so that it is understandable in our learning system. Currently they are several approaches that transform a document into vectors, more precisely, there are many techniques used to carry out a vector representation of a document that can offer us several tracks for using this principle at the level of the detection of the plagiarism between the documents. Among these techniques, we find the deep learning which is an important component of computational intelligence which has the core domain machine learning research in it. Concerning text mining applications, deep learning methods represent words as a vector of numerical values. This new representation contains a major part of syntactic as well as semantic rules of the text data in applications such as: “similarity detection and text classification,”- much larger units such as: “phrases, sentences and documents” should be described as a vector. Vector representation of text data makes easier to compare words and sentences as well as minimizing the need to use lexicons [1].

Among the approaches used for the vector representation of a word based on deep learning, the word2vec model, proposed by Mikolov and al., is very popular and has attracted great attention over the last two years. It has been shown that vector representations of words learned by word2vec models have a semantic meaning and are useful in various tasks of NLP. We also find the doce2vec which is inspired by word2vec to have a vector representation of a sentence.

In this paper we propose a plagiarism detection system based on the algorithms proposed by deep learning to train our system to detect the different types of plagiarism in a dataset. The remainder of this paper is organized as following: The first section is dedicated to the definition of basic concept. Additionally, the second section is about defining related work. Concerning the third and fourth Sections they are devoted to illustrate deep learning and the approach of using it in NLP applications and define an overall illustration of our comparison; and the last section introduces our future work to be carried out in the conclusion.

2. Related Works

In NLP tasks such as document classification, x typically encodes features like words or characters occurring in the text. Bag-of-words approaches, and extensions considering n-grams, are arguably the most commonly used representations, treating words and/or phrases as unique discrete symbols, and weighting their contributions

Manuscript received October 5, 2019
Manuscript revised October 20, 2019

through heuristics such as the Term Frequency multiplied by the Inverse Document Frequency (TF-IDF)[2].

More recently, noting that bags-of-words often fail in capturing similarities between words, methods using neural networks to learn distributed vector representations of words (i.e., word embeddings) have gained popularity, through methods like word2vec [13] or GloVe. Das and Smith [3][4] present a probabilistic model for paraphrase detection based on syntactic similarity, semantics, and hidden loose alignment between syntactic trees of the two given sentences. Socher et al. [3][5] present an approach based on recursive autoencoders for paraphrase detection. Heilman and Smith [3][6] propose a tree edit model for paraphrase identification based on syntactic relations among words. Wang et al. [3][7] decompose the sentence similarity matrix into a similar component matrix and a dissimilar component matrix, and train a two-channel convolutional neural network to compose these components into feature vectors. Xu et al. [3][8] propose a latent variable model that jointly infer the correspondence between words and sentences.

In this paper, we focus on using deep learning algorithms to develop a robust system for detecting the most difficult kind of similarity, and as above we found lot of approaches which use the word representation in their algorithms for detecting similarity, in our approach we will compare the method that use the word representation with our method that use a sentence representation. The both methods will use the siamse lstm in our training phase.

3. Deep Learning Algorithms Used

Deep learning models have presented very good performance on complex tasks that require deep understanding of text like translation, question answering, summarization, natural language inference etc. so it seemed like an excellent approach but deep learning is usualy trained on hundreds of thousands or even millions of labeled data points. Usually, we need big datasets for deep learning to avoid over-fitting. Deep neural networks have many parameters thus usually if they don’t have enough data, they tend to memorize the training set and perform poorly on the test set. To avoid this phenomenon without big data we need to use special techniques [10].

3.1 Word2vec

The word2vec model and its application by Mikolov and al [11] was the inspiration for a great advancement over the last two years. It has been shown that vector representations of words learned by word2vec models have a semantic meaning and are useful in various tasks of NLP. Words that represent synonyms have similar vectors. Even more surprising, word vectors accept the laws of analogy. For example, consider the analogy "The woman is to the queen as the man is to the king" as follow:

$$v_{queen} – v_{women}+vman \sim v_{king}$$

Note that there are two main word2vec models: a continuous word bag (CBOW) and a skip-gram. In the CBOW model, they predict a word in a context (a context may look like a sentence). Skip-Gram is the opposite: predict the context from an input word.

3.2 Doc2vec

The goal of doc2vec is to create a numeric representation of a document, regardless of its length. But unlike words, documents do not come in logical structures such as words, so another method has to be found. The concept that Mikilov and Le have used was simple, yet clever: they have used the word2vec model, and added another vector (Paragraph ID below) [11], like so:

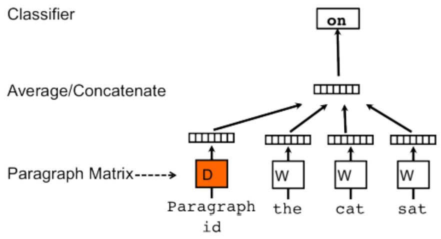

Fig. 1 Doc2vec model.

3.3 LSTM

We use lstm and not cnn because cnn is a feed forward neural network that is generally used for Image recognition and object classification.While RNN works on the principle of saving the output of a layer and feeding this back to the input in order to predict the output of the layer. CNN considers only the current input while RNN considers the current input and also the previously received inputs. It can memorize previous inputs due to its internal memory. RNN can handle sequential data while CNN cannot. In RNN, the previous states are fed as input to the current state of the network. RNN can be used in NLP, Time Series Prediction, Machine Translation, etc. The Long Short-Term Memory, or LSTM, network is perhaps the most successful RNN because it overcomes the problems of training a recurrent network and in turn has been used on a wide range of applications. We have also based on the study carried out at the level of this article [12] which illustrates the strong point of using the lstm compared to cnn in the treatment of a sequence of words.

4. Proposed Approaches

In this section we will propose two approaches; one that uses the wor2vec to represent the document words as vectors and another that uses doc2vec to transform the sentences of the document to vectors. These two representations are both used in the same learning phase that will take a pair of documents that must represent either the word2vec or doc2vec; this phase should learn all types of plagiarism offered by the 75% of the pairs of documents proposed by the Pan corpus which is a corpus for the evaluation of automatic plagiarism detection algorithms.

4.1 Preprocessing Phase

This part consists of transforming each document of the corpus into a list of vectors; the following figure illustrates the steps used for each approach:

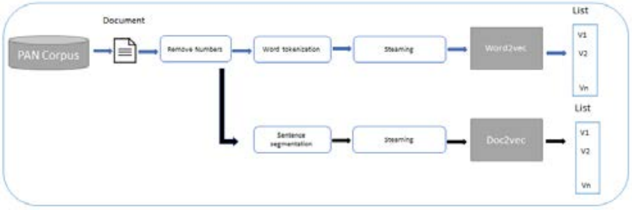

Fig. 2 Preparation phase

These steps start with a cleaning of the document by deleting the special characters as well as the numbers, then there will be two processing that will be performed separately, the application of the steaming that converts each word into their basic dictionary forms for easy comparisons and then either apply the word2vec or doc2vec to finally have a vector representation of the document.

4.2 Training Phase

This phase consists in making our system learn the different types of plagiarism proposed by the corpus pan, we use the siamese lstm algorithm which takes as input two documents which are either represented by the word2vec or by doc2vec. These two documents are labeled 1 if they are similar 0 if they are not. The following figure illustrates the learning phase of our system:

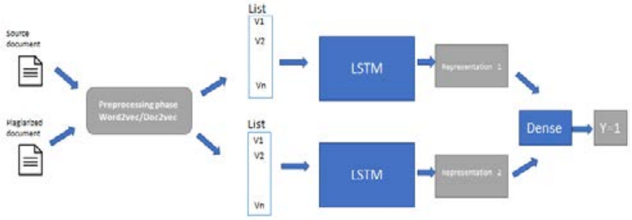

Fig. 3 Training phase

5. Validation and Comparison

Regarding the validation we worked with python framework which offered us several functionalities. nltk gensim pickle and keras are used to set up our validation.

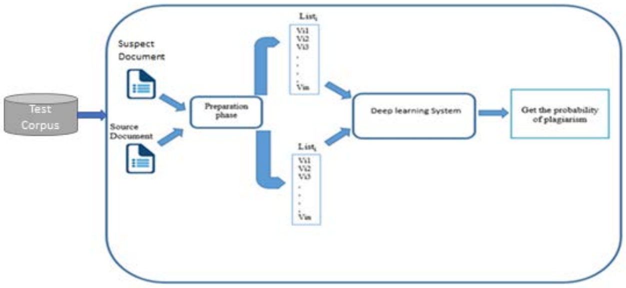

Fig. 4 Testing phase

The figure above gives an overall view of our approach. The system take two documents in the input, the first is the suspect document and the second is the source document from 25 % of the PAN corpus used for the test. These two documents will be prepared at the preprocessing phase. And later, these documents will be represented by a list of vectors that will subsequently be as a base of a deep learning system.

This vector list will be provided by either the word2vec application or doc2vec. Our goal is to have the statistics needed to detect the most relevant method among these two vector representations. For each method we calculated the value of the accuracy and the loss value as illustrated below:

Found result when using word2vec:

```
Epoch 000: Loss: 1.145, Accuracy: 28.294%
Epoch 050: Loss: 1.232, Accuracy: 66.021%
Epoch 100: Loss: 0.429, Accuracy: 88.943%
Epoch 150: Loss: 0.293, Accuracy: 90.000%
Epoch 200: Loss: 0.105, Accuracy: 91.602%
```

Found result when using doc2vec:

```
Epoch 000: Loss: 1.948, Accuracy: 10.958%
Epoch 050: Loss: 1.001, Accuracy: 74.394%
Epoch 100: Loss: 0.300, Accuracy: 90.000%
Epoch 150: Loss: 0.103, Accuracy: 96.400%
Epoch 200: Loss: 0.092, Accuracy: 98.291%
```

The measure of the accuracy across the entire test set using word2vec method is as follow.

```
Test set accuracy: 90.271%
```

Also when we use the doc2vec method we found the result bellow:

```
Test set accuracy: 97.934%
```

The system will detect later the two probabilities of similarity provided by the application of our approaches. As shown in the table below we find the results found for each method used:

Table 1: Result of our test

Couple of documents	Probability of similarity when we use Word2vec	Probability of similarity when we use Doc2vec
Pair of documents 1	0.82	0.93
Pair of documents 2	0.32	0.61
Pair of documents 3	0.61	0.60
Pair of documents 4	0.44	0.62
Pair of documents 5	0.57	0.75
Pair of documents 6	0.85	0.93
Pair of documents 7	0.84	0.88
Pair of documents 8	0.45	0.62
Pair of documents 9	0.65	0.71
Pair of documents 10	0.72	0.78

Finally, it is noticed that the results provided by the use of the doc2vec for the vector representation of a document are more relevant compared when using word2vec. So we conclude that the list of sentence vectors that represents a document correctly and keeps the semantic part that we will use at the comparison level.

6. Conclusion

In this paper, we have mentioned many different methods used in detection of plagiarism of ideas that are based on the Deep Learning principal. This study showed us the interest of the use of deep learning in the detection of plagiarism. We have proposed a system for the detection of plagiarism based on the deep learning method. Its interest is the extraction of characteristics without losing the sense of the document. It was possible to detect the best vector representation of a document through a validation that was carried out on the PAN corpus. So we have noticed that the use of doc2vec in our approach is the best solution. Indeed, we did not use a precise method to calculate the similarity between the documents by using for example a computation of distance between the vectors of the documents but we could detect these measurements through the result of our neuron network which offers us probabilities of similarity between the paragraphs of a couple of analyzed documents thing that offers us the reliable results. Concerning the future work consists of consolidate our approach by adding other treatment that helps us to detect better the plagiarism.

References

[1] Bela Gipp State-of-the-art in detecting academic plagiarism. International Journal for Educational Integrity. University of California, Berkeley and University of Magdeburg, Department of Computer Science.

[2] LUÍS BORGES, BRUNO MARTINS, PÁVEL CALADO. Combining Similarity Features and Deep Representation Learning for Stance Detection in the Context of Checking Fake News. ACM Journal of Data and Information Quality, Vol. 9, No. 4, Article 39. Publication date: April 201. Instituto Superior Técnico, Universidade de Lisboa, Portugal.

[3] Basant Agarwal, Helge Langseth, 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.

[4] D. Das, N. A. Smith, Paraphrase Identification As Probabilistic Quasi-synchronous Recognition, in: Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1, ACL '09, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 978-1-932432-45-9, 468–476, 2009.

[5] R. Socher, E. H. Huang, J. Pennington, A. Y. Ng, C. D. Manning, Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection, in: Proceedings of the 24th International Conference on Neural Information Processing Systems, NIPS'11, Curran Associates Inc., USA, ISBN 978-1-61839-599-3, 801–809, 2011.

[6] M. Heilman, N. A. Smith, Tree Edit Models for Recognizing Textual Entailments, Paraphrases, and Answers to Questions, in: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, HLT '10, Association for Computational Linguistics, Stroudsburg, PA, USA, ISBN 1-932432-65-5, 1011–1019, 2010jkh

[7] Z. Wang, H. Mi, A. Ittycheriah, Sentence Similarity Learning by Lexical Decomposition and Composition, in: Proceedings of the 26th International Conference on Computational Linguistics, COLING 2016 Technical Papers, 1340–1349, 20.

[8] W. Xu, A. Ritter, C. Callison-Burch, W. Dolan, Y. Ji, Extracting Lexically Divergent Paraphrases from Twitter,

Transactions of the Association for Computational Linguistics 2 (2014) 435–448.

[9] Martin Potthast Benno Stein Andreas Eiselt, Alberto Barr´on-Cede˜no Paolo Rosso. Overview of the 1st International Competition on Plagiarism Detection. Stein, Rosso, Stamatatos, Koppel, Agirre (Eds.): PAN'09, pp. 1-9, 2009. Web Technology & Information Systems Group Bauhaus-Universit¨ at Weimar, Natural Language Engineering Lab, ELiRF Universidad Polit´ecnica de Valencia.

[10] yonatan hadar. Lessons Learned from Applying Deep Learning for NLP Without Big Data.

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

[12] 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.