A Hybrid Model for Paraphrase Detection Combines pros of Text Similarity with Deep Learning

Mohamed I. El Desouki
Information Systems Department
Prince Sattam Bin Abdulaziz University

Wael H. Gomaa
Beni-Suef University
Shaqra University

Hawaf Abdalhakim
Department of Information Systems,
College of computers Engineering and and Artificial Intelligence,
Helwan University
Cairo, Egypt

ABSTRACT
Paraphrase detection (PD) is a very essential and important task in Natural language processing. The goal of paraphrase detection is to check whether two statements written in natural language have the identical semantic or not. Its importance appears in many fields like plagiarism detection, question answering, document clustering and information retrieval, etc. This paper proposes a hybrid model that combines the text similarity approach with deep learning approach in order to improve paraphrase detection. This model verified results with Microsoft Research Paraphrase Corpus (MSPR) dataset, shows that accuracy measure is about 76.6% and F-measure is about 83.5%.

General Terms
Paraphrase detection, Skip thought vector, Text similarity

Keywords
Paraphrase detection, Deep Learning, Skip thought vector, Text similarity

1. INTRODUCTION
Paraphrase Detection (PD) is the ability to check whether two sentences that are written in a natural language are similar or not. The goal of PD is to decide whether the two sentences have the exact meaning or not. In other words PD assess whether the two sentences have the identical semantic or not. This task is not a trivial or a simple task but it is an important and a basic step in many NLP tasks, and it is a basic component that must exist in software programs that are used in plagiarism detection, information retrieval, text mining, text recapitulation, machine translation, document clustering, etc. PD typically follow two approaches, unsupervised approach which depends on text similarity algorithms, and the supervised approach which depends on machine learning (ML) and deep learning algorithms.

Words can be similar in two ways lexically and semantically. Semantics is the study of what does a word or an expression means. Words are said to be lexically similar when they have the same sequence of characters. Words are said to be semantically similar when they have the same meaning, used in the same way, used in the same context or one is a type of another. Usually, different String-Based algorithms are used to check the Lexical similarity. While the Semantic similarity is introduced in Corpus-Based and Knowledge-Based algorithms [1].

Another approach that is applied in solving the PD task is the Machine learning (ML) approach which depends on applying the ML algorithms that deal with the PD task as a regular text classification problem that makes use of syntactic and/or linguistic features. ML is categorized in two techniques; supervised learning that is defined as inferring a function from labeled training data. and Unsupervised learning that is defined as inferring a function to describe hidden structure from unlabeled data i.e. a classification or categorization is not included in the observations. Distinguishing unsupervised learning from supervised learning and reinforcement learning [2] can be observed from the examples that are given to the learner, unlabeled examples, and accuracy evaluation of the output-structure may be considered. Deep learning is a class of machine learning that is based on learning data representations, as opposed to task-specific algorithms. Learning category can be supervised and semi-supervised, unsupervised [3] or reinforcement learning.

In brief, the main purpose of this paper is to represent a new model by combining different text similarity algorithms with deep learning approach named skip thought vector. The proposed model noticeably advances the results of the most important researches in the area of paraphrase detection for the Microsoft Research Paraphrase Corpus.

The rest of this paper is organized as follows. The relevant related work on paraphrase detection techniques is represented in Section 2. Section 3 gives a brief overview on skip-thought vectors and text similarity algorithms. The proposed hybrid model will be illustrated in details in section 4.The experiment results will be listed in Section 5. The paper conclusion and suggested future work comes in Section 6.

2. RELATED WORK
This section represents a review on the previous work done in PD, the previous work can be categorized into two approaches; the first is the unsupervised approach that depends on Text Similarity (TS) while the other is the supervised approach that basically depends on (ML) and (DL) methods. Unsupervised learning is a class of Machine Learning techniques to find the patterns in data and it aims to describe the hidden structure from "unlabeled" data. It depends on the various approaches of text similarity such as string-based, corpus-based and knowledge-based, while the supervised learning infers a function from labeled training data. Two measures are used to evaluate the performance of the selected articles. The first is the accuracy, second is the F-measure in identifying paraphrase in Microsoft Research Paraphrase Corpus (MSPR) dataset.

F-measure is a measure of a test's accuracy. It considers both the precision p which is the number of correct positive results divided by the number of all positive results returned by the classifier and the recall r which is the number of correct positive results divided by the number of all relevant samples of the test to compute the score . It is the harmonic average of the precision

and recall, F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0.

In Table 1, a list of 20 articles is summarized to show the most important methods that were applied in paraphrase detection. The table organized in five columns, the 1st column shows the reference number of the article, the 2nd column lists the category of learning approach which is one of three categories (Unsupervised, Machine learning and Deep learning). The 3rd column represents the paraphrase detection method, and the 4th and 5th columns show the accuracy and the F-measure values of each method [27].

Table 1: Paraphrase Detection Methods

Ref. No.	Learning Approach category	Methods	Accuracy	F-measure
[5]	Unsupervised	- Cosine similarity with tf-idf weighting - Combination of several word similarity measures	64.5 % 70.3 %	75.3 % 81.3 %
[6]	Unsupervised	- Explicit semantic space - Salient semantic space	67 % 72.5 %	79.3 % 81.4 %
[7]	Unsupervised	- Graph subsumption	70.6 %	80.5 %
[8]	Unsupervised	- Combination of semantic and string similarity	72.6 %	81.3 %
[9]	Unsupervised	-Additive composition of vectors and cosine distance	73 %	82 %
[10]	Unsupervised	JCN WordNet similarity with matrix	74.1 %	82.4 %
[11]	Machine Learning	Sentence dissimilarity classification	72 %	81.6 %
[12]	Machine Learning	PI using semantic heuristic features	74.4 %	81.8 %
[13]	Machine Learning	Combination of lexical and semantic features	76.6 %	79.6 %
[14]	Machine Learning	Combination of MT evaluation measures as features	75 %	82.7 %
[15]	Machine Learning	Product of experts	76.1 %	82.7 %
[16]	Machine Learning	Dependency-based features	75.6 %	83 %
[17]	Machine Learning	Combination of eight machine translation metrics	77.4 %	84.1 %
[18]	Machine Learning	Matrix factorization with supervised reweighting	80.4 %	85.9 %
[19]	Machine Learning	Combination of Convolution Kernels and similarity scores	79.1 %	85.2 %
[20]	Deep Learning	Recursive autoencoder with dynamic pooling	76.8 %	83.6 %
[21]	Deep Learning	Simple distributional semantic space	73 %	82.3 %
[22]	Deep Learning	Multi-perspective Convolutional NNs and structured similarity layer	78.6 %	84.7 %
[23]	Deep Learning	Recursive NNs using syntax-aware multi-sense word embeddings	78.6 %	85.3 %
[24]	Deep Learning	Sentence Similarity Learning by Lexical Decomposition and Composition	78.4 %	**2.7 %**

3. Skip-thought Vector and Text Similarity Methodologies

3.1 Skip-thought Vector

One of the most important unsupervised approaches in implementing the sentence embedding is the skip-thought vector [4]. Sentence embedding can be considered as an umbrella that covers a set of techniques in natural language processing (NLP) where sentences are mapped to vectors of real numbers. The idea is to encode sentences in fixed-length dense vectors to highly improve the processing of textual data. The sentence embedding is considered an extension to the word embedding, which is a representation of words in a n-dimensional vector space so that semantically similar words (e.g. "boat"—"ship") or semantically related (e.g. "boat"—"water") words come closer depending on the training method.

For the sentence embedding, according to their purposes, they generally fall into two categories: task-specific sentence embeddings and general-purpose sentence embeddings [26]. The first category focuses on training sentence embeddings for a particular task using supervised learning methods. The other category focuses on global sentence embeddings, which are usually built using unsupervised or semi-supervised learning and can be served as features for different NLP tasks like text classification and semantic textual similarity.

Skip-thought vector can be though as the equivalent for sentences of the skip-gram model developed for word embeddings: rather than predicting the words surrounding a word, it tries to predict the surroundings sentences of a given sentence [4]. The model consists of an RNN-based encoder-decoder model to first encode a sentence into a vector and then decode that representation into the surrounding sentences. Skip-thought vectors not take the ordering of both words and sentences into account. This allows it to encode rich information into the embedding. The skip-thought model has been proven to be effective at learning sentence representations and capturing sentence semantics. Skip-thought vectors have achieved outstanding results in many complicated tasks including image-sentence ranking, question-type classification, paraphrase detection, semantic relatedness and sentiment analysis.

3.2 Text Similarity

As mentioned in the in introduction, Text similarity measures play an increasingly important role in text related research and applications in many useful tasks such as information retrieval, text classification and many other tasks. Words can be similar in two ways lexically and semantically. Lexical similarity usually introduced as String-Based algorithms while Semantic similarity is introduced through Corpus-Based and Knowledge-Based algorithms.

String similarity measures operate on string sequences and character composition. A string metric is a metric that measures similarity or dissimilarity (distance) between two text strings for approximate string matching or comparison. An excellent and more detailed overview of text similarity measures can be found in [25]. This research handles 14 types of String-based similarity algorithms: Block Distance, Cosine similarity, Dice's coefficient, Euclidean distance, Jaccard similarity, Jaro, Jaro-Winkler, Levenshtein, Matching Coefficient, MongeElkan, Needleman-Wunsch, Overlap Coefficient, N-gram Similarity and Smith-Waterman.

Corpus-Based similarity is a semantic similarity measure that determines the similarity between words according to information gained from large corpora [25]. This research handles 4 types of Corpus-based similarity algorithms; two of them are based on DIStributionally similar words using CO-occurrences (DISCO): DISCO1 and DISCO2. The other two are based on topic modeling: Latent semantic analysis (LSA) and Latent Dirichlet allocation (LDA).

Knowledge-Based similarity is a semantic similarity measure that determines the degree of similarity between words using information derived from semantic networks. This research handles Lin algorithm that considers the information content of lowest common subsumer and the two compared concepts. A lowest common subsumer is a concept in a lexical semantic network (e.g. WordNet), which has the shortest distance from the two concepts compared.

All similarity measures are normalized to output similarity value between 0 and 1. The work of text similarity is performed using SimAll tool introduced in [26].

3.3 The proposed Hybrid Approach

As shown in Figure.1, the proposed model checks the Paraphrase detection between two sentences task through 3 steps.

Step 1 Measure the quotation between the two sentences:

i) The two sentences are entered into our algorithm to assess the degree of similarity through applying text similarity algorithms. Each algorithm is trained and tested using 50 different classifiers in Weka tool with 10-fold cross-validation method.

ii) The algorithm produces a degree of similarity between the two sentences, ranging from 0 to 1. While zero does not quote at all, one means that they are identical.

iii) At the same level the two sentences are entered as an input to a pre-trained skip-thought vector [4] which depends on a training corpus of contiguous text that was extracted from a large collection of novels, and free books namely the BookCorpus dataset. This pre-trained model focuses on converting the two sentences into semantic vectors using skip-thought approach to get the semantic vector of each sentence and then measure the vector similarity between the resulted semantic vectors. Vectors is an array with as many rows as the length of X, and each row is 4800 dimensional (combine-skip model). The first 2400 dimensions is the uni-skip model, and the last 2400 is the bi-skip model. Bi-skip model contains two encoders with different parameters: one encoder is given the sentence in correct order, while the other is given the sentence in reverse. The combine-skip vectors are highly recommended; as they are almost universally the best performing in previous research [4].

Step 2: Learning and verification are as follows:

Using the Weka tool, a classical machine learning algorithms are applied on the resulted similarity values from both Skip-thought pre-trained model and from text similarity algorithms to evaluate the paraphrase detection system. 10-fold cross-validation is used for all of the experiments.

Step 3: System evaluation:

We have submitted two methods, called CombineALL and CombineBest. The CombineALL method is performed by training on all of the obtained similarity values from all of the 20 algorithms that were tested separately. While In the CombineBest method, 7 algorithms were selected using CfsSubsetEva attribute evaluator; it evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them. The search method used was BestFirst; it searches the space of attribute subsets by greedy hillclimbing augmented with a backtracking facility. The selected algorithms were Skip-thought, Jaro, Matching Coefficient, MongeElkan, Needleman-Wunch, N-gram and Lin.

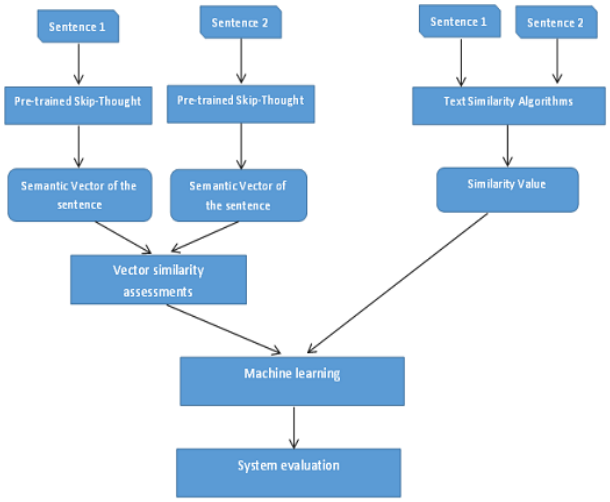

Figure 1: The proposed Model

4. Experiments Results and Analysis

The proposed model is evaluated on MSRP dataset; it is a well-known dataset in the field of paraphrase detection, text similarity and other NLP tasks, it is provided by Microsoft, it includes 5800 pairs of sentences which have been extracted from news sources on the web, along with human annotations indicating whether each pair captures a paraphrase/semantic equivalence relationship. No more than one sentence has been extracted from any given news article. There are 1725 pairs of sentences are used for testing all approaches, while the rest pairs are used for training purposes for only skip-thought approach. Table 2 shows the accuracy and F-Measure values of all measures that are tested separately. As mentioned above, different Weka classifiers are tested on vs text similarity algorithms. The fourth column in table 2 shows these classifiers that outputs the best results. The best values of the accuracy and F-measure were 75.7 and 82.9 respectively, which resulted from the skip-thought approach

Table 2: Comparing the results of Skip-thought and different text similarity algorithms

Seq	Text-similarity algorithms	Accuracy	F-Measure	Weka Classifier
1	Skip-thought	**75.7**	**82.9**	-
2	Block Distance	72.34	80.1	bayes.BayesNet
3	Cosine similarity	72.1	80.1	meta.LogitBoost
4	Dice's coefficient	72.4	80.2	bayes.BayesNet
5	Euclidean distance	70.37	77.6	bayes.BayesNet
6	Jaccard similarity	72.34	80.3	bayes.BayesNet
7	Jaro	70.5	79.3	function.multilayerPerceptron
8	Jaro-Winkler	70	78.5	fuction.multilayerPerceptron
9	Levenshtein	69	77.4	meta.LogitBoost
10	Matching Coefficient	72	80.9	bayes.BayesNet
11	MongeElkan	70.1	79.3	bayes.BayesNet
12	Needleman-Wunsch	68.5	77.5	lazy.kstar
13	Overlap Coefficient	71	79	bayes.BayesNet
14	N-gram Similarity	73.2	81.6	meta.Random subspace
15	Smith-Waterman	68.2	77.1	bayes.NaiveBayes
16	DISCO 1	69.6	78.3	lazy.kstar
17	DISCO 2	69.2	78.3	bayes.NaiveBayes
18	LSA	69.1	78.2	bayes.BayesNet
19	LDA	68.7	77.8	lazy.kstar
20	LIN	69.7	79.5	bayes.NaiveBayes

Table 3 shows the results of the proposed hybrid model; we only illustrated the classifiers that enhanced the results obtained before the hybrid method. The best values of the accuracy and F-measure were 76.6 and 83.5 respectively,which resulted from the Voted Perceptron classifier using CombineBest method.

Table 3: Results of proposed hybrid model

Method	Accuracy	F-Measure	Weka Classifier
CombineALL	76.2	83.1	rules.JRip
CombineALL	76.2	83.2	functions.Logistic
CombineBest	76.1	82.4	functions.Logistic
CombineBest	76.1	82.5	functions.SGD
CombineBest	76.5	83.3	functions.SMO
CombineBest	**76.6**	**83.5**	**functions.VotedPerceptron**
CombineBest	76.1	82.6	lazy.LWL
CombineBest	76.3	82.6	meta.AdaBoostM1
CombineBest	76.3	82.7	meta.ClassificationViaRegression
CombineBest	76.1	82.4	meta.FilteredClassifier
CombineBest	76.1	82.3	meta.MultiClassClassifier
CombineBest	76.1	82.3	trees.DecisionStump

5. CONCLUSION, LIMITATIONS AND FUTURE WORK

This paper has examined paraphrase detection recent approaches. It established a new paraphrase detection hybrid model. The developed work is done within Microsoft Research Paraphrase Corpus dataset. The proposed model is verified throughout three-level stages. In the first stage the pre-trained skip-thought vector is used, where the pre-trained skip-thought vector converts the sentences into semantic vectors using skip-thought approach to get the semantic vector of each sentence then measures the vector similarity between the resulted semantic vectors. The accuracy and F-measure are 75.7 and 82.9 respectively. In the second stage 19 different string-based, corpus-based and knowledge-based are tested using text-similarity-algorithms. Each algorithm is trained and tested using 50 different classifiers with 10-fold cross-validation method. In the third stage the classical machine algorithms is exercised to obtain similarity values from both Skip-thought pre-trained model and text similarity algorithms, and to evaluate this paraphrase detection. Also this work evolved two new methods methods, called CombineALL and CombineBest. They are tested, where the CombineALL method testing is performed via training all of the obtained similarity values from all of the 20 algorithms that is tested separately. In the CombineBest method, 7 algorithms are automatically selected using attribute evaluator algorithm. In brief this proposed model achieves effective results values of 76.6 and 83.5 paraphrase detection accuracy and F-measure respectively.

The main limitation for the proposed model is that it can only be applied on English language; as the pre-trained model was trained using English texts.

Our future work will focus on applying the proposed model architecture to other languages as Arabic. Furthermore, different sentence embedding approaches will be tested such as Paragraph Vector, Siamese CBOW and FastSent.

6. REFERENCES

[1] Gomaa, W. H., & Fahmy, A. A. (2011). Tapping into the power of automatic scoring. In The Eleventh International Conference on Language Engineering, Egyptian Society of Language Engineering (ESOLEC).

[2] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural computation, 9(8):1735– 1780, 1997.

[3] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436.

[4] Kiros, R., Zhu, Y., Salakhutdinov, R. R., Zemel, R., Urtasun, R., Torralba, A., & Fidler, S. (2015). Skip-thought vectors. In Advances in neural information processing systems (pp. 3294-3302).

[5] Mihalcea, R., Corley, C., & Strapparava, C. (2006, July). Corpus-based and knowledge-based measures of text semantic similarity. In *AAAI* (Vol. 6, pp. 775-780).

[6] Hassan, S. (2011). Measuring semantic relatedness using salient encyclopedic concepts. University of North Texas.

[7] Rus, V., McCarthy, P. M., Lintean, M. C., McNamara, D. S., & Graesser, A. C. (2008, May). Paraphrase Identification with Lexico-Syntactic Graph Subsumption. In *FLAIRS conference*(pp. 201-206).

[8] Islam, A., & Inkpen, D. (2009). Semantic similarity of short texts. *Recent Advances in Natural Language Processing V*, *309*, 227-236.

[9] Milajevs, D., Kartsaklis, D., Sadrzadeh, M., & Purver, M. (2014). Evaluating neural word representations in tensor-based compositional settings. *arXiv preprint arXiv:1408.6179*.

[10] Fernando, S., & Stevenson, M. (2008, March). A semantic similarity approach to paraphrase detection. In *Proceedings of the 11th Annual Research Colloquium of the UK Special Interest Group for Computational Linguistics* (pp. 45-52).

[11] Qiu, L., Kan, M. Y., & Chua, T. S. (2006, July). Paraphrase recognition via dissimilarity significance classification. In Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (pp. 18-26). Association for Computational Linguistics.

[12] Ul-Qayyum, Z., & Altaf, W. (2012). Paraphrase identification using semantic heuristic features. *Research Journal of Applied Sciences, Engineering and Technology*, *4*(22), 4894-4904.

[13] Kozareva, Z., & Montoyo, A. (2006). Paraphrase identification on the basis of supervised machine learning techniques. In Advances in natural language processing (pp. 524-533). Springer, Berlin, Heidelberg.

[14] Finch, A., Hwang, Y. S., & Sumita, E. (2005). Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*.

[15] Das, D., & Smith, N. A. (2009, August). 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-Volume 1 (pp. 468-476). Association for Computational Linguistics.

[16] Wan, S., Dras, M., Dale, R., & Paris, C. (2006). Using dependency-based features to take the'para-farce'out of paraphrase. In *Proceedings of the Australasian Language Technology Workshop 2006* (pp. 131-138).

[17] Madnani, N., Tetreault, J., & Chodorow, M. (2012, June). Re-examining machine translation metrics for paraphrase identification. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 182-190). Association for Computational Linguistics.

[18] Ji, Y., & Eisenstein, J. (2013). Discriminative improvements to distributional sentence similarity. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing* (pp. 891-896).

[19] Filice, S., Da San Martino, G., & Moschitti, A. (2015). Structural representations for learning relations between pairs of texts. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (Vol. 1, pp. 1003-1013).

[20] Socher, R., Huang, E. H., Pennin, J., Manning, C. D., & Ng, A. Y. (2011). Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Advances in neural information processing systems* (pp. 801-809).

[21] Blacoe, W., & Lapata, M. (2012, July). A comparison of vector-based representations for semantic composition. In Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning (pp. 546-556). Association for Computational Linguistics.

[22] He, H., Gimpel, K., & Lin, J. (2015). Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (pp. 1576-1586).

[23] Cheng, J., & Kartsaklis, D. (2015). Syntax-aware multi-sense word embeddings for deep compositional models of meaning. *arXiv preprint arXiv:1508.02354*.

[24] Wang, Z., Mi, H., & Ittycheriah, A. (2016). Sentence similarity learning by lexical decomposition and composition. *arXiv preprint arXiv:1602.07019*.

[25] Gomaa, W. H., & Fahmy, A. A. (2013). A survey of text similarity approaches. International Journal of Computer Applications, 68(13), 13-18.

[26] Wael H. Gomaa and Aly A. Fahmy (2017). SimAll: A flexible tool for text similarity. The Seventeenth Conference On Language Engineering ESOLEC' 2017 17 (1), 122-127, Ain Shams University, Cairo, Egypt.

[27] Mohamed El I Desouki and Wael H Gomaa. Exploring the Recent Trends of Paraphrase Detection. International Journal of Computer Applications 182(46):1-5,March 2019