Statistical Analysis of ML-Based Paraphrase Detectors with Lexical Similarity Metrics

El-Sayed M. El-Alfy
College of Computer Sciences and Engineering
King Fahd University of Petroleum and Minerals
Dhahran 31261, Saudi Arabia
Email: alfy@kfupm.edu.sa

***Abstract*—Paraphrase detection has several important applications in natural language processing. Examples of such applications include language translation, text summarization, question answering, plagiarism detection, and online information retrieval. A number of metrics have been proposed in the literature to quantify the textual similarity between two sentences. However, the accuracy of utilizing each similarity metric alone in detecting paraphrases is very low. Though some machine learning (ML) techniques have been deployed for paraphrase detection, there is no known study that intensively benchmarks their performance on this problem under similar conditions. In this paper, we evaluate the utility of integrating five lexical similarity metrics with three standard machine learning paradigms to detect paraphrases. We apply statistical tests to compare and benchmark the relative significance of the adopted ML-based paraphrase detectors on different datasets.**

I. Introduction

The process of expressing the same information content differently in a pair of sentences is known as paraphrasing [1]; which can occur in a variety of ways. Paraphrasing is also known as bidirectional textual entailment since each sentence can be inferred from the other. Recognizing pairs of paraphrased sentences is essential in many applications in natural language processing. Examples of such applications include multi-language translation, plagiarism detection, question answering, tutoring systems, text generation, multiple document summarization, and information retrieval using search engines.

In the past decade, there has been increasing interest on problems related to paraphrasing and text entailment. In [2], the authors presented an extensive survey of the work related to paraphrasing and textual entailment applications, recognition, generation and extraction. Considering paraphrase detection, a number of approaches have been suggested in the literature using lexical and contextual similarity metrics. Among these approaches is the work described in [3] using a support vector machine and a set of machine translation (MT) related metrics. Applying it on Microsoft research paraphrase corpus [4], an identification accuracy of 74.96% is achieved on the test dataset. In [5], the authors described a technique to construct a corpus of sentential paraphrase pairs from news clusters found on the world-wide web. In [6], a neural network approach is introduced for paraphrase recognition. In [7], string similarity measures were combined to recognize paraphrases among sentences. Another approach is presented in [8] to detect similarity between pairs of short textual units by combining primitive and composite linguistic features with the rule-based machine learning algorithm, RIPPER.

Though a number of similarity metrics and paraphrase detection methods exist in the literature, there is no intensive statistical comparison made among them. This paper aims at evaluating and comparing the performance of state-of-the-art machine learning paradigms on different datasets. We present three frequently used machine learning methods and statistically compare their significance using t-tests on four datasets in terms of a number of performance measures.

The rest of the paper is organized as follows. Section II describes the methodology followed in this study. Section III discusses the empirical work and the results. Finally, Section IV concludes the paper.

II. Methodology

In this paper, we selected three paradigms for supervised machine learning that are well-known of their effectiveness and applied them to the problem of sentence-level paraphrase recognition. We compare their performance using t-test and significance analysis. A high-level outline of the methodology is illustrated in Figure 1.

A. Textual Similarity Metrics

1) Word Edit Distance: A word-level modified version of the well-know Levenshtein edit distance between two character strings [9]. It is computed from the number of deletion, insertion, and substitution operations of words that are needed to convert one sentence into the other. The computation of edit distance is performed using a dynamic programming algorithm which has a complexity of $O(|s_1| \times |s_2|)$ where $|s_i|$ is the length of sentence i [10]. Mathematically, it can be expressed recursively as follows $\forall i = 1 : |s_1|, \forall j = 1 : |s_2|$,

$$d(i,j) = \begin{cases} i, & \text{if } j = 0 \\ j, & \text{if } i = 0 \\ min\{d(i-1,j-1)+ & \\ \ (s_1[i] = s_2[j]?0:1), & \\ \ d(i-1,j)+1, & \\ \ d(i,j-1)+1\} & \text{otherwise} \end{cases} \quad (1)$$

where the expression $(s_1[i] = s_2[j]?0 : 1)$ is equal to 0 if $s_1[i] = s_2[j]$; otherwise it is 1.

When there are a few changes, i.e. very similar pair of sentences, the value of this metric will be small. As more

978-1-4799-4441-5/14/$31.00 ©2014 IEEE

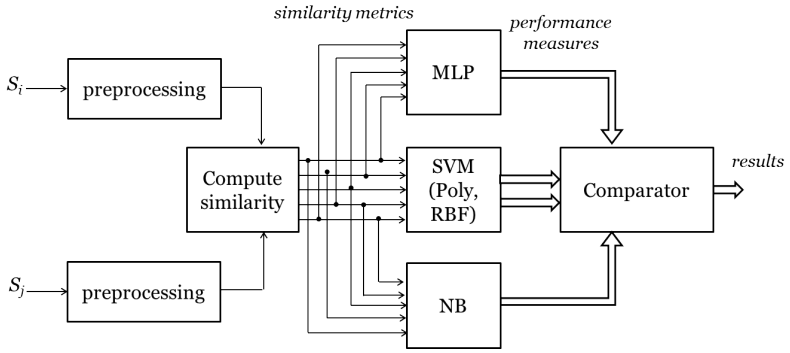

Fig. 1. Schematic diagram for the processing steps for paraphrase detection and comparisons

changes are introduced, the edit distance will increase. The effectiveness of this metric depends on the level of paraphrasing. Although edit distance has been widely used in earlier publications [9], [11], [12]; it fails to detect paraphrases with too many lexical changes.

2) Simple Word N-Gram Overlap: This metric uses the word n-gram overlap for $n = 1, 2, \ldots, N$ and compute the average ratio as follows,

$$sim_o(S_1, S_2; N) = \frac{1}{N} \times \sum_{n=1}^{N} \frac{S_1^n \cap S_2^n}{min(S_1^n, S_2^n)} \quad (2)$$

where $S_1^n \cap S_2^n$ is the number of common n-grams in sentences S_1 and S_2, and $min(S_1^n, S_2^n)$ is the number of n-grams in the shorter sentence.

3) Exclusive LCP N-Gram Overlap: This metric is similar to sim_o but it counts the prefix overlapping of uni-gram, bi-gram, $\ldots N-$gram considering the Longest Common Prefix (LCP) [12],

$$sim_{exc}(S_1, S_2; N) = \max_{n \in \{1,2,\ldots,N\}} \frac{S_1^{n,exc} \cap S_2^{n,exc}}{min(S_1^{n,exc}, S_2^{n,exc})} \quad (3)$$

4) BLEU Metric: BLEU is an abbreviation of BiLingual Evaluation Understudy. It has been one of the first devised methods for automatic quality evaluation of machine translation (MT) of a natural language in terms of the semantic equivalence to reference translations made by professional humans [13]. It is widely used for evaluating machine translation and benchmarking other MT related metrics. Later, it has been used in some studies for paraphrase generation and identification as well, e.g. [3], [7], [14]. The computation of the BLEU metric is based on counting the matches of the word n-grams for the input sentences (for $n \leq N$ where N is a pre-specified number which is frequently taken as 4).

5) Sumo Metric: This metric is devised in [12] to alleviate some of the limitations faced by other metrics. The computation of this metric is based on exclusive lexical links between the two sentences. It was designed as reliable and fast means to minimize the human intervention in corpus construction. It penalizes equal and almost equal sentences and considers pairs with different syntactic structure and high degree of lexical reordering,

B. Machine Learning Paradigms

We examined Bayesian learning, artificial neural networks (ANN) and two examples of kernel machines [15], [16]. In the following, we briefly describe these methods.

1) Bayesian Learning: This machine learning paradigm is common and is frequently applied to a variety of classification applications. An example of this paradigm is the Naive Bayes (NB) classifier which is also used to benchmark other methods. The idea is to handle the uncertainty in the data using probabilistic relations based on the Bayes' theorem. It starts by estimating the posterior and prior probabilities using a training dataset. Then, the resulting model is used to determine the class for each given instance by finding the class that maximizes the posterior probability. Naive Bayes assumes that the attributes are conditionally independent given the class label. Thus, for a feature vector $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, the conditional probability $p(\mathbf{x}|y)$ is simply expressed as the product of the conditional probabilities for each feature as given by:

$$p(\mathbf{x}|y) = \prod_{i=1}^{n} p(x_i|y) \quad (4)$$

2) Artificial Neural Networks: A typical structure of a feed-forward multi-layer perceptron (MLP) is shown in Fig. 2, where there are three layers of processing units. The input layer receives the feature vector, which is a vector of real-valued data representing the similarity metrics between the two sentences. The output of the input layer is fed into a hidden layer. Subsequently, the output from the hidden layer goes into the output layer which has only one neuron giving a predicted class, paraphrased or not, corresponding to the given input vector. Each neuron multiplies the input vector by a weight vector and adds a bias, then the net output is passed through an activation or transfer function. Normally, the neural network uses supervised learning with previously labeled examples of the positive and negative classes to determine the the network parameters (a.k.a. weights and biases). To combine bias and weights in one vector, an additional input x_0 is set to +1 and thus the corresponding weight represents the bias. The adjustment of network parameters is achieved by the frequently used back-propagation training algorithm.

TABLE I. SUMMARY OF DATASETS SIZES AND NUMBERS OF POSITIVE AND NEGATIVE PAIRS

Designation	Total	Positive	Negative	Remarks
DS1	4076	2753	1323	MSRPC train dataset
DS2	1725	1147	578	MSRPC test dataset
DS3	750	376	374	STS train dataset with original MS labels
DS4	750	554	196	STS train dataset with GS rating thresholded at 2.8

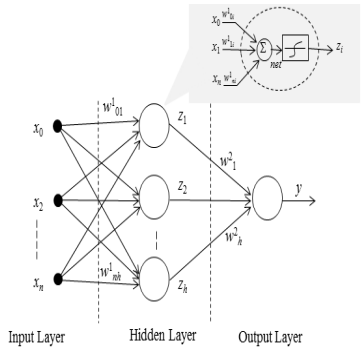

Fig. 2. A typical structure of a feed-forward multi-layer perceptron

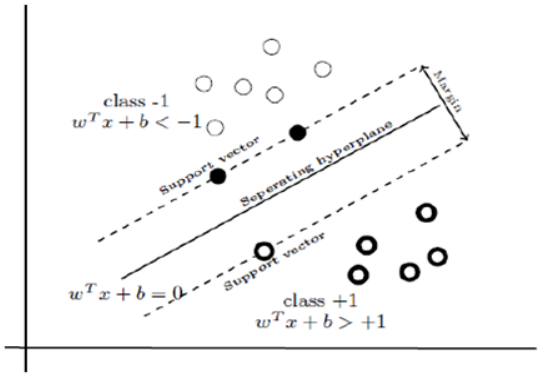

Fig. 3. Illustration of a support vector machine decision boundary for a two-dimensional feature space

3) Kernel Machines: SVM has gained great importance in pattern recognition in a variety of fields [17], [18]. The underlying idea of SVM is to map the feature space into a higher-dimensional space where data points become linearly separable using a kernel function. The training algorithm of SVM constructs a maximum margin hyper-plane in the new space of mapped features to separate the data points; as illustrated for a 2D example in Fig. 3. The closest points to the hyper-plane are called support vectors. The optimal separation hyper-plane is found by solving a constrained optimization problem using the Lagrangian multipliers.

For the linearly separable case, SVM looks for a hyper-plane $H : w^T y+b = 0$ and two hyper-planes $H_1 : w^T y+b = 1$ and $H_2 : w^T y + b = -1$ parallel to, and with equal distances to, H with the condition that there are no data points between H_1 and H_2 and the distance between H_1 and H_2 is maximized (see Fig. 3). The variables w and b are the parameters to be optimized. For the non-linearly separable case, input feature vectors are transformed to a higher-dimensional space, by using a kernel function, where a linear hyper-plane is located. We examined two types of kernels: polynomial and radial-basis function (RBF) kernels.

III. EXPERIMENTAL WORK

A. Datasets

We adopted four datasets that contain positive (paraphrased) and negative (non-paraphrased) pairs of sentences. A summary of the various datasets is given in Table I. The first two datasets correspond to the training and testing datasets of the Microsoft Research Paraphrase Corpus (MRPC) [4]. This corpus is one of the largest and complex paraphrase corpora that is widely used in the literature in related work. The other two datasets correspond to the training dataset published as part of SemEval-2012 workshop under Task 6 entitled Semantic Textual Similarity (STS) [19]. This dataset is a subset drawn from MRPC containing 750 sentence pairs. In the first version, DS3, the dataset uses the same labels as the corresponding sentence pairs in the original MRPC. The second version, DS4, uses the Gold Standard rating which has been disclosed as part of Task 6 of SemEval-2012. This rating provides a real value on a continuous scale from 0 to 5 representing the degree of graded bidirectional semantic equivalence for each sentence pair (with 0 means different sentences and 5 means fully semantically equivalent sentences).

B. Evaluation Measures

The performance of various classifiers is evaluated in terms of the following measures: recognition rate, precision, recall, F_1 measure, Area under the ROC curve (AUC), kappa, and model construction time. These performance measures are summarized in Table II.

TABLE II. EVALUATION MEASURES

Measure	Formula
Recognition Rate	$rr = (tp + tn)/(tp + tn + fp + fn)$
Precision	$prc = tp/(tp + fp)$
Recall	$rec = tp/(tp + fn)$
F_1 Measure	$F_1 = 2 \times prc \times rec/(prc + rec)$
AUC	–
Kappa	–
Train Time	–

C. Experiments and Results

In order to conduct our study, we adopted the implementations existing in Weka [15] for multi-layer perceptrons, support vector machines with both second-order polynomial and radial-basis function kernels, and naive Bayes classifiers. We used 10-fold stratified cross validation and repeated the experiment 20 times for each algorithm on each dataset. Thus, the total number of runs is $4 \times 4 \times 10 \times 20 = 3200$. For each performance measure, we used the statistical t-test with 95% two-tailed significance value to assess various methods on different datasets.

In our experiments, we utilized naive Bayes (NB), an example of the Bayesian learning paradigm, as a baseline classifier to benchmark other paradigms. For kernel machines,

TABLE III. PERFORMANCE COMPARISON OF THE FOUR METHODS FOR VARIOUS DATASETS BASED ON PERCENTAGE CLASSIFICATION ACCURACY AND THE STATISTICAL SIGNIFICANCE AS IT COMPARES TO NB (b: SIGNIFICANTLY BETTER THAN NB, w: SIGNIFICANTLY WORSE THAN NB, s: SIGNIFICANTLY SAME AS NB).

Dataset	#Runs	NB	MLP	SVM.Poly	SVM.RBF
DS3	200	55.97±6.09	62.19±6.16b	57.17±5.83s	50.45±2.48w
DS1	200	68.85±2.05	71.67±2.29b	70.58±1.38b	67.54±0.10s
DS2	200	69.32±3.14	74.16±3.13b	68.30±1.54s	66.49±0.20w
DS4	200	75.41±4.67	78.77±4.07b	79.17±4.13b	73.87±0.65s
Count($b/s/w$)		Baseline	(4/0/0)	(2/2/0)	(0/2/2)

TABLE IV. PERFORMANCE COMPARISON OF THE FOUR METHODS FOR VARIOUS DATASETS BASED ON PRECISION RATE AND THE STATISTICAL SIGNIFICANCE AS IT COMPARES TO NB (b: SIGNIFICANTLY BETTER THAN NB, w: SIGNIFICANTLY WORSE THAN NB, s: SIGNIFICANTLY SAME AS NB).

Dataset	#Runs	NB	MLP	SVM.Poly	SVM.RBF
DS3	200	0.55±0.05	0.63±0.07b	0.58±0.07s	0.50±0.03w
DS2	200	0.77±0.03	0.79±0.03b	0.69±0.01w	0.66±0.00w
DS1	200	0.77±0.02	0.77±0.03s	0.72±0.01w	0.68±0.00w
DS4	200	0.85±0.03	0.83±0.03w	0.83±0.03w	0.74±0.01w
Count($b/s/w$)		Baseline	(2/1/1)	(0/1/3)	(0/0/4)

we employed SVM with two types of kernels (second-order degree polynomial and radial-basis function kernels). The training of SVM is carried out using John Platt's sequential minimal optimization (SMO) algorithm [20] and normalized features. For artificial neural networks, we used a 3-layer back-propagation MLP with sigmoid activation function units. The number of processing units in hidden layer is three. Attributes were normalized and the learning rate and momentum were set to 0.3 and 0.2, respectively. The maximum number of training epoches was set to 1000.

Tables III to IX show the statistical t-test results for various performance measures considering NB as the baseline classifier. In each table, the number of times the algorithm performs better (b), same (s), or worse (w) is indicated and counted. The count is given in the last row. For example, in Table III MLP has performed better than NB for all datasets; hence it was in the table as (4/0/0). The other tables use similar notations. for We can observe that all methods performed better on DS4 as it compares to the other three datasets. We also observed that MLP has mostly performed significantly better than NB but it needed more training time. Figure 4 shows the performance measures for ranking different learning algorithms on dataset DS4.

IV. CONCLUSION

This paper examines the performance of three machine learning paradigms on four paraphrase detection datasets using 10-fold stratified cross validation. The analysis is conducted using t-test and 95% confidence as it compares to the naive Bayes approach as a baseline classifier. It is found that the multi-layer perceptron performed significantly better than naive Bayes in terms of accuracy, precision, F_1 measure and AUC. However, MLP requires more training time. As future work, other machine learning algorithms and datasets can be examined and ranked on this particular and very important problem.

Acknowledgments: The author would like to acknowledge the support provided by the Deanship of Scientific Research (DSR) at King Fahd University of Petroleum & Minerals (KFUPM) for funding this work through the Intelligent Systems Research Group (ISRG) under project No. RG1113-1&2. Special thanks to R. Abdel-Aal, W. Al-Khatib and F. Alvi at KFUPM for providing invaluable comments on the initial draft of this paper.

TABLE V. PERFORMANCE COMPARISON OF THE FOUR METHODS FOR VARIOUS DATASETS BASED ON RECALL RATE AND THE STATISTICAL SIGNIFICANCE AS IT COMPARES TO NB (b: SIGNIFICANTLY BETTER THAN NB, w: SIGNIFICANTLY WORSE THAN NB, s: SIGNIFICANTLY SAME AS NB).

Dataset	#Runs	NB	MLP	SVM.Poly	SVM.RBF
DS3	200	0.67±0.08	0.63±0.15s	0.55±0.09w	0.97±0.13b
DS1	200	0.77±0.02	0.83±0.06b	0.93±0.02b	1.00±0.00b
DS2	200	0.78±0.04	0.84±0.05b	0.97±0.02b	1.00±0.00b
DS4	200	0.81±0.05	0.91±0.04b	0.91±0.04b	1.00±0.00b
Count($b/s/w$)		Baseline	(3/1/0)	(3/0/1)	(4/0/0)

TABLE VI. PERFORMANCE COMPARISON OF THE FOUR METHODS FOR VARIOUS DATASETS BASED ON F_1 MEASURE AND THE STATISTICAL SIGNIFICANCE AS IT COMPARES TO NB (b: SIGNIFICANTLY BETTER THAN NB, w: SIGNIFICANTLY WORSE THAN NB, s: SIGNIFICANTLY SAME AS NB).

Dataset	#Runs	NB	MLP	SVM.Poly	SVM.RBF
DS3	200	55.97±6.09	62.19±6.16b	57.17±5.83s	50.45±2.48w
DS1	200	68.85±2.05	71.67±2.29b	70.58±1.38b	67.54±0.10s
DS2	200	69.32±3.14	74.16±3.13b	68.30±1.54s	66.49±0.20w
DS4	200	75.41±4.67	78.77±4.07b	79.17±4.13b	73.87±0.65s
Count($b/s/w$)		Baseline	(4/0/0)	(2/2/0)	(0/2/2)

TABLE VII. PERFORMANCE COMPARISON OF THE FOUR METHODS FOR VARIOUS DATASETS BASED ON KAPPA STATISTIC AND THE STATISTICAL SIGNIFICANCE AS IT COMPARES TO NB (b: SIGNIFICANTLY BETTER THAN NB, w: SIGNIFICANTLY WORSE THAN NB, s: SIGNIFICANTLY SAME AS NB).

Dataset	#Runs	NB	MLP	SVM.Poly	SVM.RBF
DS3	200	0.12±0.12	0.24±0.12b	0.14±0.12s	0.01±0.05w
DS1	200	0.29±0.05	0.32±0.08s	0.20±0.04w	0.00±0.00w
DS2	200	0.31±0.07	0.40±0.08b	0.11±0.05w	0.00±0.00w
DS4	200	0.39±0.11	0.39±0.12s	0.40±0.12s	0.00±0.00w
Count($b/s/w$)		Baseline	(2/2/0)	(0/2/2)	(0/0/4)

REFERENCES

[1] R. Bhagat and E. Hovy, "What is a paraphrase?" *Computational Linguistics*, vol. 39, no. 3, pp. 462–472, 2013.

TABLE VIII. PERFORMANCE COMPARISON OF THE FOUR METHODS FOR VARIOUS DATASETS BASED ON AUC AND THE STATISTICAL SIGNIFICANCE AS IT COMPARES TO NB (b: SIGNIFICANTLY BETTER THAN NB, w: SIGNIFICANTLY WORSE THAN NB, s: SIGNIFICANTLY SAME AS NB).

Dataset	#Runs	NB	MLP	SVM.Poly	SVM.RBF
DS3	200	0.59±0.07	0.67±0.06b	0.57±0.06s	0.50±0.02w
DS2	200	0.71±0.04	0.78±0.03b	0.54±0.02w	0.50±0.00w
DS1	200	0.72±0.02	0.76±0.03b	0.58±0.02w	0.50±0.00w
DS4	200	0.78±0.06	0.79±0.06s	0.68±0.06w	0.50±0.00w
Count($b/s/w$)		Baseline	(3/1/0)	(0/1/3)	(0/0/4)

TABLE IX. PERFORMANCE COMPARISON OF THE FOUR METHODS FOR VARIOUS DATASETS BASED ON CPU TRAINING TIME AND THE STATISTICAL SIGNIFICANCE AS IT COMPARES TO NB (b: SIGNIFICANTLY BETTER THAN NB, w: SIGNIFICANTLY WORSE THAN NB, s: SIGNIFICANTLY SAME AS NB).

Dataset	#Runs	NB	MLP	SVM.Poly	SVM.RBF
DS1	200	0.01±0.01	7.44±0.28w	31.21±4.56w	39.94±8.78w
DS2	200	0.00±0.01	3.05±0.08w	4.25±1.38w	4.81±1.20w
DS3	200	0.00±0.01	1.33±0.04w	0.31±0.05w	0.48±0.13w
DS4	200	0.00±0.00	1.31±0.02w	0.21±0.03w	0.44±0.14w
Count($b/s/w$)		Baseline	(0/0/4)	(0/0/4)	(0/0/4)

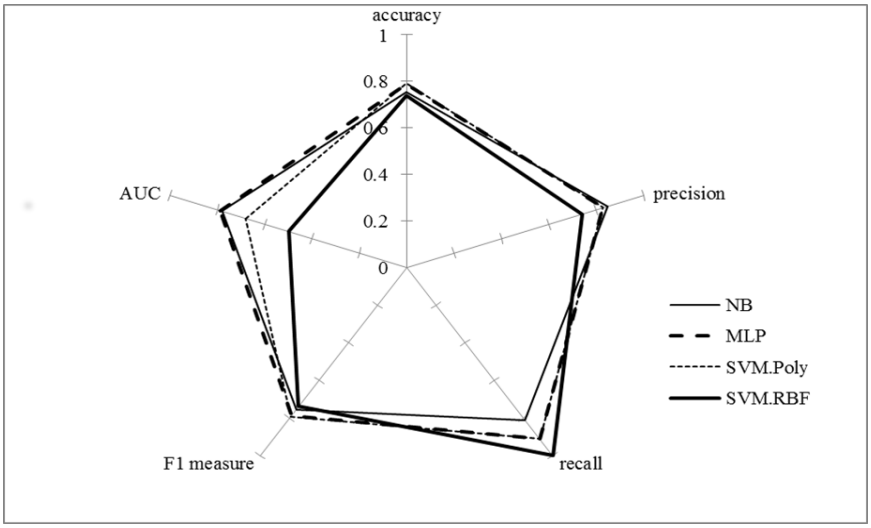

Fig. 4. Comparing the learning methods on DS4 in terms of accuracy, precision, recall, F_1 measure and AUC

[2] I. Androutsopoulos and P. Malakasiotis, "A survey of paraphrasing and textual entailment methods," *Journal of Artificial Intelligence Research*, vol. 38, no. 1, pp. 135–187, may 2010.

[3] A. Finch, Y.-S. Hwang, and E. Sumita, "Using machine translation evaluation techniques to determine sentence-level semantic equivalence," in *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005, pp. 17–24.

[4] "Microsoft research paraphrase corpus," http://research.microsoft.com/en-us/downloads/.

[5] C. Brockett and W. B. Dolan, "Support vector machines for paraphrase identification and corpus construction," in *Proceedings of the 3rd International Workshop on Paraphrasing*, 2005, pp. 1–8.

[6] A. Rajkumar and A. Chitra, "Paraphrase recognition using neural network classification," *International Journal of Computer Applications*, vol. 1, no. 29, pp. 42–47, February 2010.

[7] P. Malakasiotis, "Paraphrase recognition using machine learning to combine similarity measures," in *Proceedings of the ACL-IJCNLP 2009 Student Research Workshop*, 2009.

[8] V. Hatzivassiloglou, J. L. Klavans, and E. Eskin, "Detecting text similarity over short passages: Exploring linguistic feature combinations via machine learning," in *Proceedings of the 1999 joint sigdat conference on empirical methods in natural language processing and very large corpora*, 1999, pp. 203–212.

[9] G. Navarro, "A guided tour to approximate string matching," *ACM Computing Surveys*, vol. 33, no. 1, pp. 31–88, 2001.

[10] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2009.

[11] B. Dolan, C. Quirk, and C. Brockett, "Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources," in *Proceedings of the 20th International Conference on Computational Linguistics*, 2004.

[12] J. Cordeiro, G. Dias, and P. Brazdil, "A metric for paraphrase detection," in *Proceedings of the International Multi-Conference on Computing in the Global Information Technology (ICCGI)*, march 2007.

[13] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: A method for automatic evaluation of machine translation," in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 2002.

[14] N. Madnani, J. Tetreault, and M. Chodorow, "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*, 2012.

[15] H. Witten, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Series in Data Management Systems, 2005.

[16] P.-N. Tan *et al.*, *Introduction to data mining*. Pearson Education Inc., 2007.

[17] N. Cristianini and J. Shawe-Taylor, *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[18] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data mining and knowledge discovery*, vol. 2, no. 2, pp. 121–167, 1998.

[19] "STS MSR train corpus," http://www.cs.york.ac.uk/semeval/task6/.

[20] J. C. Platt, "Advances in kernel methods," B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds. MIT Press, 1999, ch. Fast Training of Support Vector Machines Using Sequential Minimal Optimization, pp. 185–208.