Sjournals
Scientific Journals
Scientific Journal of Review (2014) 3(3) 94-106
ISSN 2322-2433
doi: 10.14196/sjr.v3i3.1259

Contents lists available at Sjournals
Scientific Journal of REVIEW
Journal homepage: www.Sjournals.com

Original article

Efficient calculation of sentence semantic similarity: a proposed scheme based on machine learning approaches and NLP techniques

M. Roostaee, S.M. Fakhrahmad, M.H. Sadreddini*, A. Khalili
Department of Computer Science and Engineering and IT, School of Electrical Engineering and Computer, Shiraz, Iran.

*Corresponding author; Department of Computer Science and Engineering and IT, School of Electrical Engineering and Computer, Shiraz, Iran.

ARTICLE INFO

Article history:
Received 06 March 2014
Accepted 24 March 2014
Available online 30 March 2014

Keywords:
Sentence semantic similarity
Natural language processing
Machine learning classification

ABSTRACT

Aim of Study Sentence semantic similarity plays a crucial role in a variety of applications such as Machine Translation, Information Retrieval, Question Answering and Multi-document Summarization. Considering the variability of natural language expression, sentence semantic similarity detection is not a trivial task. This paper tries to make use of Natural Language Processing (NLP) as well as machine learning techniques in order to propose a scheme for sentence semantic similarity. Materials and Methods In the first part of the proposed scheme, i.e., the NLP section, different sets of linguistic features including string-based, semantic-based, Named Entity-based and syntax-based features are extracted. In the second part, machine learning algorithms are used to construct classification models on the extracted set of features. Results Experimental results in the first part indicate that extracted features are valid for sentence semantic similarity. Moreover, by comparing the performance of different classification algorithms in the second part, KNN seems to be the most successful algorithm. Overall conclusion Overall, experimental results indicate that the proposed approach can be used to improve the performance of sentence semantic similarity detection especially in terms of accuracy.

1. Introduction

Sentence semantic similarity detection has a wide range of applications such as paraphrase recognition, textual entailment recognition and question answering. Paraphrase recognition systems try to detect paraphrases, sentences and natural language expressions. These terms convey similar or almost similar meaning (Bhagat and Hovy, 2013). Textual entailment recognition systems receive pairs <T, H> and determine whether H is most likely to T or not (Iftene et al., 2012). On the other hand, question answering applications need to specify the similarity of question-answer or question-question pairs (Sangeetha and Arok, 2012). The most important problem is the difficulty to determine semantic similarity for short sentences. The reason is that context information is not provided adequately and natural language expression can be stated in variety of forms. For example, sentences A and B have similar meanings while they have different forms:

A: They stay with their mother for over a year.

B: For over twelve months, they remain with their mother.

In this paper, NLP and machine learning algorithms are employed to determine the semantic similarity of short sentences. First, NLP extracts four different sets of linguistic features from sentences including String-based features, Semantic-based features, Named-Entity-based features and Syntax-based features. Moreover, WordNet is used to extract semantic information. Next, machine learning classification algorithms such as K-Nearest Neighbor (KNN), Logistic Regression and Support Vector Machines (SVM) construct models on extracted feature sets. Finally, models are evaluated on three datasets to specify the most successful classification algorithm. Evaluation datasets are Microsoft Research Paraphrase Corpus (MSRP), Third Recognition Textual Entailment (RTE3) and TREC9 Question Variant Key (TREC9). Each dataset targets a different application of sentence semantic similarity detection. Experimental results indicate that proposed approach improves the performance of current methods, tools and techniques for sentence semantic similarity detection.

A number of different methods have already been proposed to determine the semantic similarity of long sentences. Researchers try to improve these methods for similarity detection in short sentences. Since a sentence is a syntactic form of word sequence expressing a fact or a piece of information compared to documents, the performance of these methods is not acceptable for similarity detection (Achananuparp and In, 2010). Generally, semantic similarity detection methods are categorized into four main categories; word co-occurrence based methods, corpus based methods, hybrid methods and syntax based methods.

Word co-occurrence based methods also called as vector based methods or bag-of-words methods are the most widely used methods in information retrieval (Meadow et al., 1999). These methods have poor performances in similarity detection for short sentences. The reason is that short sentences have much less similar words compared to documents. In addition, natural language expression can be stated with variety of forms. Several techniques are introduced to resolve the problem such as pattern matching methods in text mining (Chiang and Yu, 2005). These techniques express the meaning as a set of patterns. As a result, pattern matching can be used to compute the similarity. The main disadvantage of these techniques is the lack of automated methods to extract the complete set of patterns for each meaning.

Corpus-based methods use statistical information in large corpus for similarity detection. Latent Semantic Analysis (LSA) is the most well-known method in this category. LSA analyses the corpus and forms the word by context matrix which reflects the presence of words in the text. In fact, this matrix computes the similarity between words and pieces of text. Singular Value Decomposition (SVD) reduces the dimensionality of this matrix. For each sentence, a reduced vector is computed. LSA uses these reduced vectors to calculate the similarity of sentences (Foltz et al., 1998). LSA does not consider syntactic information and has a poor performance on short sentences. Another work in this category is done by Islam et al. (Islam and Inkpen, 2008) which uses semantic and syntactic information to compute the

sentence semantic similarity. This work combines string similarity and semantic word similarity. It also uses common word order similarity in order to represent syntactic information.

Several methods use both corpus-based and knowledge-based techniques (Li et al., 2006; Mihalcea et al., 2006). Li et al. (2006) proposed hybrid methods which combines lexical database information and corpus statistics to construct a semantic vector. Moreover, a word order vector is computed using lexical database. Word order and semantic vectors are combined to compute sentence semantic similarity. Mihalcea et al. (2006) combines the results of six WordNet and two corpus-based measures. Since this method combines eight measures, it is not computationally efficient.

Although Islam and Inkpen (2008) and Mihalcea et al. (2006) leveraged word order similarity to compute semantic similarity, this approach was not effective. Syntax-based methods used deep parsers to obtain syntactic information. Oliva et al. (2011) proposed SyMSS to compute the semantic similarity based on semantic and syntax information. SyMSS used WordNet to calculate the semantic similarity of words and extracting syntactic information from a deep parsing process. The results indicate that its similarity measure should be reinvestigated.

Overall experimental results of described techniques indicate that different features should be considered for semantic similarity detection. Machine learning techniques try to extract different semantic, syntactic and other features to improve the performance. They build prediction and classification models to determine the similarity. Several studies employed machine learning techniques for semantic similarity detection. The problem is that the majority of these techniques build specific learning models. Stated differently, they do not compare the performance of different machine learning algorithms in different applications of semantic similarity detection (Wan et al., 2006; Zhang et al., 2011; Glava et al., 2012).

2. Materials and methods

This paper tries to improve the sentence semantic similarity using NLP and machine learning. First of all, NLP tools are used to preprocess the datasets and extract the feature sets. Preprocessing includes Normalization, Tokenization, POS Tagging, Named Entity Recognition and Syntactic parsing. In the next step, the result of preprocessing is used to extract four feature sets: String-based features, Semantic-based features, Named Entity-based features and Syntax-based features. Finally, several machine learning algorithms use these feature sets to determine the semantic similarity of sentences. This process is demonstrated in Figure 1.

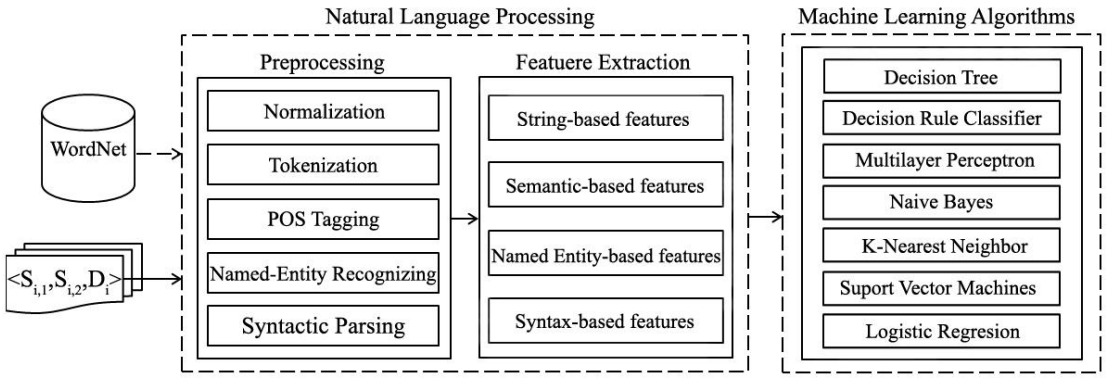

Fig. 1. The process of sentence semantic similarity Detection.

As illustrated in figure 1, the dataset contains n records in the form of $< S_{i,1}, S_{i,2}, D_i >$. $S_{i,1}$ and $S_{i,2}$ represent the first and the second sentences respectively. $D_i = 1$ implies that sentences are similar while $D_i = 0$ shows that they are not similar. Firstly, records are preprocessed and feature sets are extracted. As a result, each record is converted to $< \vec{V}_i, D_i >$ where $\vec{V}_i$ represents the extracted features. Finally, the set of $\{< \vec{V}_1, D_1 >, \ldots, < \vec{V}_n, D_n >\}$ is used to compare the performance of different machine learning algorithms in semantic similarity detection.

2.1. Preprocessing

In the preprocessing step, the following operations are performed:

Normalization: In order to normalize the data, three types of reformation including String normalization, Date and Time normalization and number normalization are carried out. String normalization stands for converting shortened strings to their primary form. Date and Time normalization uniforms the values of date and time. Number normalization converts textual representations of numbers to a standard numerical string.

Tokenization: The words are tokenized based on Penn Treebank compatible tokenizer.

POS Tagging: A POS Tagger compatible to Penn Treebank is used to tag the words.

Named Entity Recognition

Syntactic and dependency parsing

Normalization is handled using BIU normalizer that is available at http://www.cs.biu.ac.il/~nlp/downloads. Stanford CoreNLP performs Tokenization, POS Tagging, Named Entity Recognition and Syntactic parsing. A copy of the tool is available at http://nlp.stanford.edu/software/corenlp.shtml.

2.2. Feature extraction

Four different feature sets are extracted from the preprocessed data: String-based features, Semantic-based features, Named Entity-based features and Syntax-based features. These feature sets cover different characteristics of linguistic information.

2.2.1. String-based features

This set of features includes n-gram overlap features and string similarity measures. Bigram and Trigram are the most widely used n-gram overlap features. String similarity measures use string sequences to extract features. In this category, Longest Common Subsequence based features are the most widely used. Longest Common Subsequence is defined as the longest similar sequence of common words. $\mathrm{Recall}_{\mathrm{LCS}}$, $\mathrm{Precision}_{\mathrm{LCS}}$ and $\mathrm{F-Measure}_{\mathrm{LCS}}$ are computed using Longest Common Subsequence Measure defined in Lin et al. (2004).

$$Bigram_{S_1S_2} = 2 \times \frac{|B_1 \cap B_2|}{|B_1| + |B_2|}$$

$$Tigram_{S_1S_2} = 2 \times \frac{|T_1 \cap T_2|}{|T_1| + |T_2|}$$

2.2.2. Semantic-based features

In this category, features are extracted based on semantic information. Semantic similarity between concepts is drawn from the well-known tool, WordNet. Different measures have already been proposed to calculate semantic similarity of concepts including: Path-based measures, Information content measures and Gloss-based measures (Oliva et al., 2011). Path-based measures take the advantage of WordNet hierarchical structure. Information content measures focus on concept specificity and Gloss-based measures use the gloss of concepts. In this paper, we use a path-based measure proposed by Zhibiao-Wu et al. (1994). Results of different studies such as Oliva et al. (2011) indicate that this measure which considers the depth of Least Common Subsumer (LCS) to calculate path similarity is more accurate than measures in other categories:

$$Sim_{WUP}(c_1, c_2) = 2 \times \frac{depth(LCS)}{depth(c_1) + depth(c_2)}$$

Since path-based measures use hierarchical structure of WordNet they can be used only for nouns and verbs. This is because adjectives and adverbs do not have hierarchical structure. As stated in Oliva et al. (2011), the only measures which can be for adjectives and adverbs are Gloss-based measures. In this work, Extended Gloss overlap measure introduced by Banerjee and Pedersen (2003) is used.

$WNBigram_{S_1S_2}$ and $WNTrigram_{S_1S_2}$ are two measures of the n-gram overlap category. The difference between $Bigram_{S_1S_2}$ and $WNBigram_{S_1S_2}$ is that the first measure uses exact matching to compute the similarity, while the later employs WordNet based measures. More specifically, for $WNBigram_{S_1S_2}$, two concepts are similar if their similarity is larger than a predefined threshold. This is also the case for $\text{Trigram}_{S_1S_2}$ and $\text{WNTrigram}_{S_1S_2}$. Recall_{WNLCS}, Precision_{WNLCS} and F-Measure_{WNLCS} are also calculated using the same approach as $WNBigram_{S_1S_2}$ and $WNTrigram_{S_1S_2}$.

$$WNBigram_{S_1S_2} = 2 \times \frac{|WNB_1 \cap WNB_2|}{|WNB_1| + |WNB_2|}$$

$$WNTigram_{S_1S_2} = 2 \times \frac{|WNT_1 \cap WNT_2|}{|WNT_1| + |WNT_2|}$$

Vector space based features are calculated using word overlapping. Sim_{S_1} measures the similarity of sentence S_1 to S_2 and Sim_{S_2} measures the similarity of sentence S_2 to S_1. $Sim_{S_1S_2}$ represents the harmonic mean of Sim_{S_1} and Sim_{S_2}. Since similar sentences contain similar concepts, the words in a sentence which are not present in the other sentence make the sentences different (Oliva et al., 2011). This issue is covered by $Penalty_{S_1}$, $Penalty_{S_2}$ and $Penalty_{S_1S_2}$. $Penalty_{S_1}$ calculates the differences of sentence S_1 to S_2 and $Penalty_{S_2}$ calculates the differences of sentence S_2 to S_1. $Penalty_{S_1S_2}$ represents the harmonic mean of $Penalty_{S_1}$ and $Penalty_{S_2}$. TotalSim calculates the overall similarity between S_1 to S_2:

$$Sim_{S_1} = \frac{\sum_{w_i \in S_1} MaxSim(w_i, S_2)}{|S_1|}$$

$$Sim_{S_1S_2} = 2 \times \frac{Sim_{S_1} \times Sim_{S_2}}{Sim_{S_1} + Sim_{S_2}}$$

$$Penalty_{S_1} = \frac{\ln(\sum_{w_i \in (S_1 \ S_2)} P_i + 1)}{|S_1|}$$

$$Penalty_{S_1S_2} = 2 \times \frac{penalty_{S_1} \times penalty_{S_2}}{Penalty_{S_1} + penalty_{S_2}}$$

$$TotazlSim = Sim_{S_1S_2} - Penalty_{S_1S_2}$$

2.2.3. Named Entity-based features

These features are proposed based on Named entities extracted from preprocessing step. Considering Named Entity similarities and differences, 11 features on entities such as person, organization and date are extracted. For example, NE_{person} calculates person overlapping in sentences S_1 and S_2. In this equation, P_1 is the set of person entities in S_1 and P_2 is the set of person entities in S_2.

$$NE_{person} = 2 \times \frac{|P_1 \cap P_2|}{|p_1| + |P_2|}$$

2.2.4. Syntax-based features

As stated by Oliva et al. (2011) and Wiemer-Hastings (2004), syntax is also important in semantic similarity detection. In this paper, syntactic information is used to calculate the similarity of tokens with identical functional role. Extracted features are similarity of verbs in two sentences (S_V), similarity of subjects in two sentences (S_S), similarity of objects in two sentences (S_O), similarity of adverbial complements in two sentences (S_A) and finally similarity of other roles (S_R). Moreover, dependency relation of sentences is also used to extract R_{S_1} , R_{S_2} and $R_{S_1S_2}$ which are defined as:

$$R_{S_1} = \frac{\left|D_1 \cap D_2\right|}{\left|D_1\right|}$$

$$R_{S_1S_2} = 2 \times \frac{R_{S_1} \times R_{S_2}}{R_{S_1} + R_{S_2}}$$

R_{S_1} represents the dependency recall of S_1 , S_2 and R_{S_2} represents the dependency recall of S_2 , S_1 . $R_{S_1S_2}$ is the harmonic mean of R_{S_1} and R_{S_2} . Moreover, part of speech n-grams and stop n-grams (Stamatatos, 2011) are also used as features. In addition to these features, "the difference between the heights of parse trees", "difference between numbers of edges in parse trees" and "difference between numbers of leaves in parse trees" are also considered as features.

2.3. Machine learning algorithms

In this section, seven well-known supervised machine learning algorithms used in this work are described. The algorithms include Decision Tree, PART Decision Rule Classifier, Multilayer Perceptron, Naïve Bayes, K-Nearest Neighbor (KNN), Support Vector Machines (SVM) and Logistic Regression.

2.3.1. Decision tree

Decision Tree is a rule based classifier which uses tree structure to build the prediction model. Starting from the root of tree, each path to leaves is a different rule. Decision tree calculates the information gain of each feature for branching. Feature with the highest score branches the decision tree on the first level and the leaves have the least information gain. Since leaves are labeled, decision tree rules can be used for classification. The advantages of decision tree are the high scalability and interpretability but over fitting is the most important problem.

2.3.2. Decision rule classifier

Decision rule classifiers use inductive rule learning approach to building classifier. In these methods each class is represented by a Disjunctive Normal Form (DNF). The aim of decision rule is to make the smallest set of rules which are consistent to the training data. Unlike decision trees which are constructed in a top-down approach, bottom-up approach is used to build DNF rules (Sebastiani, 2002). In this paper, a well-known decision rule classifier PART (Partial Decision Tree) is used. PART generates the rules by frequently generating partial decision trees. The main advantage of PART is its ability to generate sufficiently strong rules.

2.3.3. Multilayer perceptron

Perceptron based techniques introduce another category of algorithms used in different applications such as classification and prediction. Perceptron can be briefly described as follows:

Assume that $X_1..X_n$ are input features values and $W_1..W_n$ are vectors of synaptic weights. Then, perceptron calculates the sum of net; net is defined as $W_i \times X_i$ .This process is formulated as:

$$V = \varphi(\eta) \qquad \eta = \sum_i W_i X_i$$

η calculates the sum of nets and φ is activation function which decides whether V (output) is 0 or 1. If input samples are linearly separable, perceptron can truly classify all samples. Multilayer perceptron consists of several layers of perceptrons to resolve the problem of linearly inseparable classes. Multilayer perceptron has a feed forward structure which is demonstrated in Figure 2. As illustrated in this figure, multilayer perceptron is divided into: input layer, output layer and hidden layers. Determining the number of hidden layers is an important issue which depends on the application (Ayodele, 2010). Assuming that layers and activation function are defined, the net weight is calculated by the training algorithm.

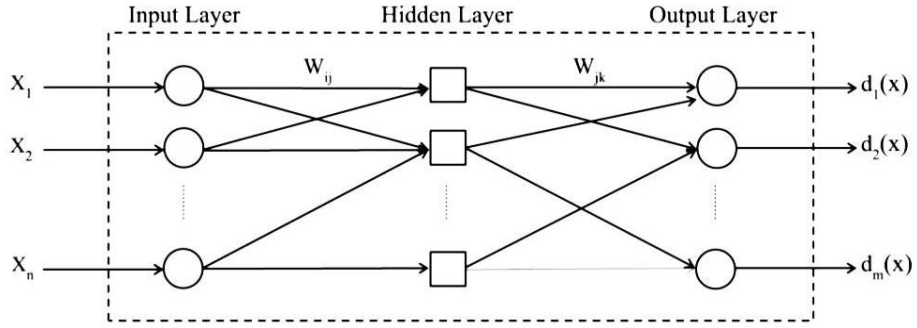

Fig. 2. Feed forward multilayer perceptron.

2.3.4. Naïve bayes

Probabilistic classifiers assume that the variables are generated by a probabilistic mechanism. They also consider the dependency between the class label and other features. Naïve Bayes is a simple probabilistic classifier which uses the Bayes theorem with strong conditional independence between features, given class labels (Baharudin et al., 2010). Naïve Bayes calculates the posterior probability of sample X in each class $P(X \mid C)$ and chooses the class with highest posterior probability as X label. $P(X \mid c)$ shows the Naïve Bayes approach to calculate posterior probabilities. $x_1, x_2, ..., x_L$ are the feature values of the sample X:

$$P(X \mid c) = P(x_1, x_2, ..., x_L \mid c) = \prod_{j=1}^{L} P(x_j, c)$$

The main advantage of Naïve Bayes is the low runtime complexity caused by conditional independence assumption. However, this assumption reduces the performance.

2.3.5. K-nearest neighbor (knn)

Lazy learners do not construct models for classification. They use the samples in training set to classify test instances. Lazy learners use a variety of distance measures to calculate the distance of test sample to training samples. The main advantage of these algorithms is the low runtime complexity in training phase. In addition, they do not need prior knowledge. However, their runtime complexity in test phase is high. The reason is that the distance of each test sample is calculated to all training sample. This approach for large datasets with large number of features and training samples is not feasible (Guo et al., 2003). One of the most successful lazy learners is KNN. It is proven that (Kuncheva, 2004):

$$error_{Bayes} < error_{1\ NN} < 2 \times error_{Bayes}$$

KNN calculates the distance of each test sample to all training samples. K nearest neighbors determine the label of test sample using majority vote (Kwon and Lee 2003).

2.3.6. Support vector machines (svm)

SVM is a supervised classification algorithm. It tries to calculate the maximum margin classifier between two classes. Maximum margin classifier is a hyperplane with maximum distance to both classes. Figure 3 demonstrates the hyperplane and margin in SVM. Assuming that y is the class label, training

samples have $y = 1$ or $y = \ 1$. Calculating the separating hyperplane leads to an optimization problem with following constraint:

$$a_i(y_i(wx_i) \ \ 1) = 0 \qquad w = \sum_{i=1}^{n} a_i y_i x_i$$

w is the separating hyperplane. For sample with $y_i(wx_i) = 1$, a_i will be positive. These samples are called support vectors. Other samples have $a_i = 0$. Above equation concludes that merely support vectors determine the separating hyperplane (Zhang et al., 2008). Assuming that z is a test sample, its label is determined using $y = wz$. If $y \geq 1$, z is labeled as class 1 else if $y \leq \ 1$ it is labeled as class -1.

When the training samples are not linearly separable, they are mapped to a high dimensional space using kernel functions. In this new space, samples are linearly separable.

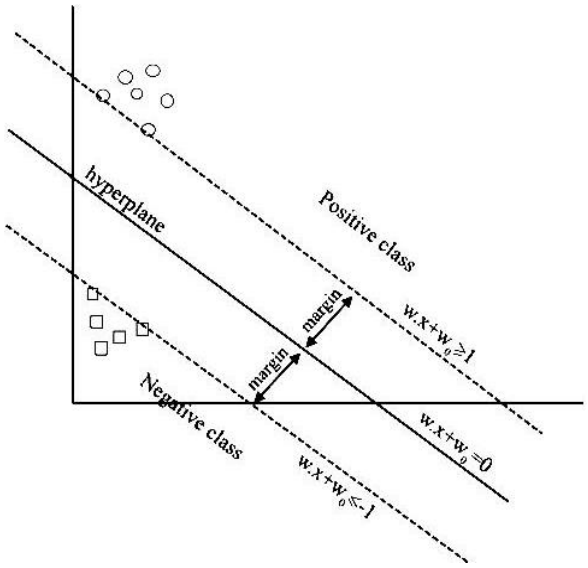

Fig. 3. SVM Algorithm.

2.3.7. Logistic regression

Logistic regression is a statistical algorithm for classification. It tries to find the weighting vector to predict the label of new sample. Assuming that θ is the weighting vector, logistic regression model can be formulated as (Komarek and Calvet, 2004):

$$\log \frac{P}{1 \ P} = \theta_0 + \theta_1 x_1 + \ldots + \theta_m x_m$$

n is the number of features and $\theta_1 .. \theta_n$ are the corresponding coefficients of features $x_1 .. x_n$. $\theta_1 .. \theta_n$ are learned using Maximum Likelihood (ML) approach. Assuming that z is the test sample, it is probability is calculated using:

$$V = \log \frac{\widehat{P}}{1 \ \widehat{P}} = \theta_0 + \theta_1 x_1 + \ldots + \theta_m x_m$$

If V is greater than a predefined threshold, X is labeled as a positive sample else it is labeled as a negative sample. Logistic regression is easily interpreted and computationally efficient.

3. Results and discussion

In this section, machine learning tool, datasets, evaluation metrics and results are explained. Waiko Environment for Knowledge Analysis (WEKA) is a widespread machine learning tool which implements the machine learning algorithms (Anon, 2013).

Microsoft Research Paraphrase Corpus (MSRP) (Dolan et al., 2004), Third Recognition Textual Entailment (RTE3) (Dagan et al., 2006) and TREC9 Question Variants Key (Achananuparp et al., 2008) are used to evaluate the proposed approach. Accuracy, Recall, Precision, Rejection, F1 and f1 measures are used for evaluation.

3.1. Datasets

Three different standard datasets are used to evaluate the performance of the proposed approach. Each dataset targets a different application of semantic similarity detection. The datasets are MSRP, RTE3 and TREC9. MSRP contains 4076 training pairs and 1725 testing pairs. Each pair is labeled by human annotators. The labels indicate whether the sentences are semantically equivalent or not. Linguistic complexity of MSRP is high since human evaluators agreed with each other in almost 83% of cases. Disagreements were resolved by another expert. RTE3 includes 800 sentence pairs. Each pair contains a Text sentence and a Hypothesis sentence. For each pair <Text, Hypothesis>, human evaluators read the Text and determine whether Hypothesis is entailed from Text or not. TREC9 contains 193 paraphrase question pair constructed by human evaluators using originally questions extracted from query log of user submitted questions. Similar to the work done in Achananuparp et al. (2008), another 193 dissimilar question pairs are constructed using randomly pairs originally question with non-paraphrase questions. Thus, TREC9 includes 386 question pairs with low linguistic complexity. The details of datasets are given in Table1.

Table 1
Summary of the datasets used in this experiment.

Summary	MSRP	RTE3	TREC9
Number of sentence pairs	1725	800	386
Number of similar sentence pairs	1147	410	193
Number of dissimilar sentence pairs	578	390	193
Number of unique words (After removing stop words)	7683	5219	287
Average sentence length without change (in characters)	115.1	113.9	39.2
Average difference of sentences in each pair (in characters)	19.6	132.8	9.4
Percent of words covers by WordNet	64.2	69.3	83.6
Linguistic complexity	High	Very high	Low

3.2. Evaluation metrics

Accuracy is the percent of correctly predicted sentences from all sentences. Recall is defined as the number of correctly predicted similar sentences compare to all similar sentences. Precision is the percent of correctly predicted similar sentences from all predicted similar sentences. Rejection is a metric to measure the performance of method to determine dissimilar sentences. Rejection is defined as the proportion of correctly dissimilar sentences compared to all dissimilar sentences and is of high importance in machine translation. F1 and f1 represent the uniform harmonic mean precision-recall and uniform harmonic mean rejection-recall, respectively. The metrics are calculated using following formulas:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}, \quad Precision = \frac{TP}{TP + FP},$$

$$Recall = \frac{TP}{TP + FN}, \quad Rejection = \frac{TN}{TN + FP},$$

$$F_1 = \frac{2 \times precision \times recall}{precision + recall}, \quad f_1 = \frac{2 \times rejection \times recall}{rejection + recall},$$

In the above formulas, TP denotes True Positive, TN denotes True Negative, FP denotes False Positive and FN denotes False Negative.

3.3. Experimental results

Evaluation was performed for three applications: Paraphrase Recognition, Textual Entailment Recognition and Question Paraphrase Recognition. MSRP dataset is used to evaluate the performance of the proposed method in Paraphrase Recognition. RTE3 dataset is used for Textual Entailment Recognition and TREC9 for Question Paraphrase Recognition.

3.3.1. Paraphrase recognition

Semantic similarity detection is necessary for paraphrase recognition. Table 2 demonstrates the result of machine learning algorithms described in section 2-3 compared to the results of the methods proposed in Islam and Inkpen (2008) and Mihalcea et al. (2006). Moreover, machine learning algorithms are also compared to phrasal overlap, TF-IDF vector overlap and Sem+WO introduced by Achananuparp et al (2008). Since MSRP contains 4076 training pairs and 1725 testing pairs, machine learning algorithms are trained using training pairs and tested on testing pairs. As illustrated in Table 2, machine learning algorithms especially KNN and Logistic Regression outperform other methods considering accuracy, precision, rejection, F1 and f1 measures.

Table 2
Performance of different machine learning algorithms compared to other similarity measure on MSRP.

Similarity Measure	Accuracy	Precision	Recall	Rejection	F1	f1
Decision Tree	0.687	0.748	0.798	0.467	0.772	0.589
Part Decision Rule	0.696	0.711	0.916	0.260	0.800	0.404
Multilayer Perceptron	0.717	0.752	0.858	0.438	0.801	0.580
Naïve Bayes	0.685	0.812	0.685	0.685	0.743	0.685
K-nearest neighbor	0.730	0.832	0.744	0.702	0.785	0.722
Support Vector Machine	0.741	0.761	0.891	0.445	0.821	0.593
Logistic Regression	0.752	0.778	0.878	0.503	0.825	0.640
Mihalcea et al.	0.703	0.696	0.977	-	0.813	-
Islam and Inkpen	0.726	0.746	0.891	0.400	0.813	0.552
Phrasal Overlap	0.675	0.700	0.892	0.244	0.785	0.383
TF-IDF vector similarity	0.690	0.734	0.836	0.398	0.782	0.539
Sem+WO	0.671	0.674	0.977	0.064	0.800	0.120

3.3.2. Textual entailment recognition

Another application of semantic similarity detection is textual entailment recognition. In this application, system needs to recognize the pairs <Text, Hypothesis> such that a human would infer Hypothesis from Text. Experiments are performed on RTE3 which includes 800 sentence pairs. Since RTE3 is not categorized to training and test sets, 10-fold cross validation is used. Results illustrated in Table 3 indicate that machine learning algorithms also outperform other methods in textual entailment recognition. As in the case for paraphrase recognition, KNN is the most successful algorithm in textual entailment recognition.

The question is the difference between the performance of machine learning algorithm on MSRP and RTE3. The reason is that the training records in MSRP are much more than RTE3. As illustrated in Table 1, the number of training records in MSRP is 4076 while RTE3 contains 800 records. The number of unique words in MSRP is 7683 while this value for RTE3 is 5219. In addition, average difference between lengths of two sentences in MSRP is 19.6 characters while this value for RTE3 is 132.8.

3.3.3. Question paraphrase recognition

TREC9 is used to evaluate the performance of machine learning algorithms in question paraphrase recognition. This dataset cannot be used directly. As a result, the same approach as Achananuparp and In (2010) is used in this paper. Since dissimilar question pairs are randomly chosen, the dataset is

constructed 10 times to reduce the bias. Table 4 demonstrates the average of performance metrics for each algorithm. Results indicate that KNN is also superior on this dataset.

Table 3
Performance of different machine learning algorithms compared to other similarity measures on RTE3.

Similarity Measure	Accuracy	Precision	Recall	Rejection	F1	f1
Decision Tree	0.586	0.590	0.632	0.538	0.610	0.581
Part Decision Rule	0.586	0.557	0.937	0.218	0.699	0.354
Multilayer Perceptron	0.580	0.591	0.588	0.572	0.589	0.580
Naïve Bayes	0.613	0.624	0.612	0.613	0.618	0.613
K-nearest neighbor	0.655	0.645	0.727	0.579	0.683	0.645
Support Vector Machine	0.628	0.608	0.768	0.479	0.679	0.590
Logistic Regression	0.629	0.625	0.690	0.564	0.656	0.621
Phrasal Overlap	0.58	0.638	0.417	0.751	0.504	0.536
TF-IDF vector similarity	0.553	0.644	0.283	0.836	0.393	0.423
Sem+WO	0.620	0.614	0.695	0.541	0.652	0.608

Table 4
Performance of different machine learning algorithms compared to other similarity measures on TREC9.

Similarity Measure	Accuracy	Precision	Recall	Rejection	F1	f1
Decision Tree	0.930	0.941	0.917	0.943	0.929	0.930
Part Decision Rule	0.956	0.949	0.964	0.948	0.956	0.956
Multilayer Perceptron	0.938	0.942	0.933	0.943	0.938	0.938
Naïve Bayes	0.915	0.888	0.948	0.881	0.917	0.913
K-nearest neighbor	0.977	0.974	0.979	0.974	0.977	0.977
Support Vector Machine	0.938	0.938	0.938	0.938	0.938	0.938
Logistic Regression	0.946	0.943	0.948	0.943	0.946	0.946
Phrasal Overlap	0.819	1.000	0.637	1.000	0.778	0.778
TF-IDF vector similarity	0.881	1.000	0.762	1.000	0.865	0.865
Sem+WO	0.948	0.963	0.933	0.964	0.948	0.948

4. Conclusions and Future works

In this paper, a new solution to resolve the problem of sentence semantic similarity detection was proposed. In the first part, proposed solution uses NLP techniques to extract different categories of linguistic features. These features represent the semantic and syntactic information of sentences. In the second part, machine learning algorithms construct classification models on extracted features. Evaluations on three standard datasets of different applications of semantic similarity detection indicated that KNN was the most successful one. Moreover, experimental results showed that proposed approach can be used to improve the performance of semantic similarity detection especially in the terms of accuracy.

In the future, we will focus on extraction of other categories of linguistic features to improve the performance. In addition, the proposed approach will be evaluated on other applications of semantic similarity detection.

References

Achananuparp, P., Hu, X., Shen, X., 2008. The evaluation of sentence similarity measures. Data Warehousing Knowl. Dis., pp. 305–316.

Achananuparp, P., In., 2010. Similarity measures and diversity rankings for query-focused sentence extraction. Ph.d. thesis. Drexel Univ., p 39.

Anon., 2013. Machine Learning Group at University of Waikato. Weka3: data mining software in Java. URL: http://www.cs.waikato.ac.nz/~ml/weka/

Ayodele, T.O., 2010. New Advances in Machine Learning. Y. Zhang., pp. 19–49.

Baharudin, B., Lee, L.H., Khan, K., 2010. A Review of Machine Learning Algorithms for Text-Documents Classification. J. Adv. Inform. Technol., 1(1), pp.4–20.

Banerjee, S., Pedersen, T., 2003. Extended gloss overlaps as a measure of semantic relatedness. In proceeding of IJCAI. pp. 805–810.

Bhagat, R., Hovy, E., 2013. What is a paraphrase. Computational Linguistics. pp. 1–16.

Chiang, J.H., Yu, H.C., 2005. Literature extraction of protein functions using sentence pattern mining. Knowl. Data Eng.IEEE Transactions on., 17(8), pp. 1088–1098.

Dagan, I., J.H., YU, H.C., 2006. The pascal recognising textual entailment challenge, In Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification. Rec. Tectual Entailment. Springer., pp. 177–190.

Dolan, B., Quirk, C., Brockett, C., 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In Proceedings of the 20th international conference on Computational Linguistics., p. 350.

Foltz, P.W., Kintsch, W., Landauer, T.K., 1998. The measurement of textual coherence with latent semantic analysis. Dis. Proc., 25(2-3), pp.285–307.

Glava, G., Sari, F., Karan, M., Snajder, J., 2012. TakeLab: Systems for Measuring Semantic Text Similarity. First Joint Conf. Lexical Comput. Semant., pp. 441–448.

Guo, G., WANG, H., BELL, D., BI, Y., GREER, K., 2003. KNN model-based approach in classification. In On The Move to Meaningful Internet Systems: CoopIS, DOA, and ODBASE. Springer., pp. 986–996.

Iftene, A., Gînscă, A., Moruz, A., Trandabăț, A., Moruz, M., Boroș, E., 2012. Enhancing a Question Answering system with Textual Entailment for Machine Reading Evaluation. Online Work., Notes/Labs/Workshop.

Islam, A., Inkpen, D., 2008. Semantic text similarity using corpus-based word similarity and string similarity. ACM Transact. Knowl. Dis. From. Data., 2(2), pp.1–25.

Komarek, P., Calvet, A., 2004. Logistic regression for data mining and high-dimensional classification. Ph.d. thesis, Carnegie Mellon University, Pittsburgh., p. 222.

Kuncheva, L.I., 2004. Combining Pattern Classifiers: Methods and Algorithms. John Wiley & Sons, Inc.

Kwon, O., Lee, J., 2003. Text categorization based on k -nearest neighbor approach for Web site classification. Inform. Process. Manag., 39, pp. 25–44.

Li, Y., Mclean, D., Bandar, Z.A., O'shea, J.D., Crockett, K., 2006. Sentence similarity based on semantic nets and corpus statistics. IEEE Transact. Knowl. Data Eng., pp. 1138–1150.

Lin, C., Och, F.J., Rey, M., 2004. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In Proceedings of the 42nd Annual Meeting on Assoc. Comput. Ling., p. 605.

Meadow, C.T., Kraft, D.H., Boyce, B.R., 1999. Text information retrieval systems, Academic Press, Inc.

Mihalcea, R., Corley, C., Strapparava, C., 2006. Corpus-based and knowledge-based measures of text semantic similarity. Proc. Amer. Assoc. Artificial Intell., pp. 775–780.

Oliva, J., Serrano, J.I., Del Castillo, M.D., Iglesias, Á., 2011. SyMSS: A syntax-based measure for short-text semantic similarity. Data Knowl. Eng., 70(4), pp. 390–405.

Sangeetha, S., Arock, M., 2012. Recognising sentence similarity using similitude and dissimilarity features. Int. J. Adv. Intell. Parad., 4(2), pp.120-131.

Sebastiani, F., 2002. Machine learning in automated text categorization. ACM Comput. Surveys., 34(1), pp.1–47.

Stamatatos, E., 2011. Plagiarism detection using stopword n-grams. J. Amer. Soc. Inform. Sci. Technol., 62(12), pp. 2512–2527.

Wan, S., Dras, M., Dale, R., 2006. Using Dependency-Based Features to Take the " Para-farce " out of Paraphrase. In proc. Australas. Language Technol. Workshop. pp.131–138.

Wiemer-Hastings, P., 2004. All parts are not created equal: SIAM-LSA. In poceedings of 26th Annu. Conf. Cogn. Sci. Soc.

Zhang, Y., Xu, X., Liu, C., Wang, X., Xu, R., Chen, Q., Wang, X., Hou, Y., Tang, B., 2011. ICRC_HITSZ at RITE: Leveraging Multiple Classifiers Votingfor Textual Entailment Recognition., In Proceedings of NTCIR-9 Workshop Meeting., pp. 325-329.

Zhang, W., Yoshida, T., Tang, X., 2008. Text classification based on multi-word with support vector machine. Knowl-Based System., 21(8), pp. 879–886.

Zhibiao, W.u., Palmer, M., Wu, Z., 1994. Verbs semantics and lexical selection. Proc. annu. meet. Assoc. Comput. Ling., pp. 133–138.