Paraphrase Identification Using Weighted Dependencies and Word Semantics

Mihai C. Lintean and Vasile Rus
Department of Computer Science
The University of Memphis
Memphis, TN, 38120, USA
E-mail: {mclinten,vrus}@memphis.edu

Keywords: paraphrasing, syntactic dependencies, word semantics

Received: March 24, 2009

We present in this article a novel approach to the task of paraphrase identification. The proposed approach quantifies both the similarity and dissimilarity between two sentences. The similarity and dissimilarity is assessed based on lexico-semantic information, i.e., word semantics, and syntactic information in the form of dependencies, which are explicit syntactic relations between words in a sentence. Word semantics requires mapping words onto concepts in a taxonomy and then using word-to-word similarity metrics to compute their semantic relatedness. Dependencies are obtained using state-of-the-art dependency parsers. One important aspect of our approach is the weighting of missing dependencies, i.e., dependencies present in one sentence but not the other. We report experimental results on the Microsoft Paraphrase Corpus, a standard data set for evaluating approaches to paraphrase identification. The experiments showed that the proposed approach offers state-of-the-art results. In particular, our approach offers better precision when compared to other approaches.

Povzetek: Prispevek se ukvarja z vsebinsko primerjavo dveh stavkov, tj. parafrazami.

1 Introduction

We present in this paper a novel approach to the task of paraphrase identification. Paraphrase is a text-to-text relation between two non-identical text fragments that express the same idea in different ways. As an example of a paraphrase we show below a pair of sentences from the Microsoft Research (MSR) Paraphrase Corpus [5] in which Text A is a paraphrase of Text B and vice versa.

Text A: *York had no problem with MTA's insisting the decision to shift funds had been within its legal rights.*

Text B: *York had no problem with MTA's saying the decision to shift funds was within its powers.*

Paraphrase identification is the task of deciding whether two given text fragments have the same meaning. We focus in this article on identifying paraphrase relations between sentences such as the ones shown above. It should be noted that paraphrase identification is different from paraphrase extraction. Paraphrase extraction [1, 2] is the task of extracting fragments of texts that are in a paraphrase relation from various sources. Paraphrase could be extracted, for instance, from texts that contain redundant semantic content such as news articles from different media sources that cover the same topic, or multiple English translations, by different translators, of same source texts in a foreign language. Recognizing textual entailment [4, 20] is another task related to paraphrase identification. Entailment is a text-to-text relation between two texts in which one text entails, or logically infers, the other. Entailment defines an asymmetric relation between two texts, meaning that one text is entailed by the other text, while paraphrase requires a symmetric relation between the two texts, i.e. one text can be entailed from the other and viceversa. Rus and colleagues [20] showed that approaches to textual entailment can be extended to handle paraphrase identification.

In this paper, we focus on the problem of paraphrase identification. Paraphrase identification is an important task in a number of applications including Question Answering [9], Natural Language Generation [10], and Intelligent Tutoring Systems [6, 15]. In Natural Language Generation, paraphrases are a method to increase diversity of generated text [10]. In Question Answering, multiple answers that are paraphrases of each other could be considered as evidence for the correctness of the answer [9]. For Intelligent Tutoring Systems with natural language input [6, 15] paraphrases are useful to assess whether student's articulated answers to deep questions (e.g. conceptual physics questions) are similar-to/paraphrases-of ideal answers.

We propose in this article a fully automated approach to the task of paraphrase identification. The basic idea is that two sentences are in a paraphrase relation if they have many similarities (at lexico-semantic and syntactic levels) and few or no dissimilarities. For instance, the two sentences shown earlier from the MSR paraphrase corpus have many similarities, e.g., common words such as *York* and common syntactic relations such as the *subject* relationship between *York* and *have*, and only a few dissimilarities, e.g., Text A contains the word *saying* while Text B contains the word *insisting*. Thus, we can confidently deem the two sentences

as being paraphrases of each other. Following this basic idea, to identify paraphrases we first compute two scores: one reflecting the similarity and the other the dissimilarity between the two sentences. A paraphrase score is generated by taking the ratio of the similarity and dissimilarity scores. If the ratio is above a certain threshold, the two sentences are judged as being paraphrases of each other. The threshold is obtained by optimizing the performance of the proposed approach on training data.

There are several key features of our approach that distinguish it from other approaches to paraphrase identification. *First*, it considers both similarities and dissimilarities between sentences. This is an advantage over approaches that only consider the degree of similarity [19] because the dissimilarity of two sentences can be very important to identifying paraphrasing, as shown by [18] and later in this article. *Second*, the similarity between sentences is computed using word-to-word similarity metrics instead of simple word matching or synonymy information in a thesaurus as in [19, 18]. The word-to-word similarity metrics can identify semantically related words even if the words are not identical or synonyms. We use the similarity metrics from the WordNet similarity package [17]. These metrics rely on statistical information derived from corpora and lexico-semantic information from WordNet [16], a lexical database of English. The basic idea behind the WordNet similarity metrics is that the closer the distance in WordNet between words/concepts is, the more similar they are. For instance, in the earlier example the semantic relationship between the words *insist* and *say* cannot be established using simple direct matching or synonymy. On the other hand, there is a relatively short path of three nodes in WordNet from *say* to *insist* via *assert*, indicating *say* and *insist* are semantically close. *Third*, we weight dependencies to compute dissimilarities between sentences as opposed to simple dependency overlap methods that do no weighting (see [13, 20]). The weighting allows us to make fine distinctions between sentences with a high similarity score that are paraphrases and those that are not due to the strength of the few dissimilarities. For instance, two sentences that are almost identical except their subject relations are likely to be non-paraphrases as opposed to two highly similar sentences that differ in terms of, say, determiner relations. We weight dependencies using two features: (1) the type/label of the dependency, and (2) the depth of a dependency in the dependency tree. To extract dependency information we used two parsers, Minipar [11] and the Stanford parser [14]. We report results with each of the parsers.

We used the MSR Paraphase Corpus [5], an industry standard for paraphrase identification, to evaluate our approach. The corpus is divided into two subsets: training and test data. The training subset was used to obtain the optimal threshold above which a similarity/dissimilarity ratio would indicate a paraphrase or a non-paraphrase, otherwise. We report state-of-the-art results on the testing data (72.06% accuracy, with Minipar), which are significantly better (Fisher's exact test yields a $p = 0.00005$) than the baseline approach of always predicting the most frequent class in the training data (66.49% accuracy) and than a simple dependency overlap method ($p<0.001$; with Minipar). Compared to results obtained using the Stanford parser (71.01% accuracy), Minipar led to statistically significant better results ($p = 0.004$).

Following this introductory part, in the next section, *What is a paraphrase?*, we offer a broader view of the concept of paraphrase. The article continues with a section on *Related Work*. The *Approach* section describes in detail how our similarity-dissimilarity method works. The following *Summary of Results* section provides details of the experimental setup, results, and a comparison with results obtained by other research groups. The *Discussion* section offers further insights into our approach and the MSR Paraphrase Corpus. The *Summary and Conclusions* section ends the article.

2 What is a paraphrase?

A quick search with the query *What is a paraphrase?* on a major search engine reveals many definitions for the concept of paraphrase. Table 1 presents a small sample of such definitions. From the table, we notice that the most common feature in all these definitions is *different/own words*. That is, a sentence is a paraphrase of another sentence if it conveys the same meaning using different words. While these definitions seem to be quite clear, one particular type of paraphrases, sentence-level paraphrases (among texts the size of a sentence), do not seem to follow the above definitions as evidenced by existing data sets of such paraphrases.

For sentential paraphrases, the feature of "different words" seems to be too restrictive, although not impossible. As we will show later in the article, the MSR Paraphrase corpus supports this claim as the paraphrases in the corpus tend to have many words in common as opposed to using *different words* to express the same meaning. While the high lexical overlap of the paraphrases in the MSR corpus can be explained by the protocol used to create the corpus - same keywords were used to retrieve same stories from different sources on the web, in general, we could argue that avoiding the high word overlap issue in sentential paraphrasing would be hard. Given an isolated sentence it would be quite challenging to omit/replace some core concepts when trying to paraphrase. Here is an example of a sentence (instance 735 in MSR corpus), *Counties with population declines will be Vermillion, Posey and Madison.*, which would be hard to paraphrase using many other/different words. The difficulty is due to the large number of named entities in the sentence. Actually, its paraphrase in the corpus is *Vermillion, Posey and Madison County populations will decline.*, which retains all the named entities from the original corpus as it is close to impossible to replace them with other words. It is beyond the

Table 1: Definitions of paraphrases from various sources.

Source	Definition. A paraphrase (is)...
Wikipedia	a restatement of a text or passage *using different words*.
Wordnet	express the same message in *different words*; rewording for the purpose of clarification.
Purdue's OWL	*your own rendition* of essential information and ideas expressed by someone else, presented in a new form.
Bedford/St.Martin's	a prose restatement of the central ideas of a poem, in *your own language*.
Pearson's Glossary	to record someone else's words in the *writer's own words*.
LupinWorks	restating the meaning *in own words*, retaining all of the ideas without making an interpretation or evaluation.

scope of this article to provide a final answer with respect to whether high lexical overlap should be acceptable or not in sentential paraphrases.

Another interesting aspect of sentential paraphasing is the fact that there seem to be two different ways to judge them. On one hand, two sentences are considered paraphrases of each other if and only if they are *semantically equivalent*, i.e. they both convey the same message with no additional information present in one sentence but not the other. An example of two sentences in a semantic equivalence was given in the previous section. Thus, in order to detect whether two sentences are *not* paraphrases of each other, we only need to find one concept that is present in one sentence but not in the other. On the other hand, two sentences can be judged as forming a paraphrase if they convey *roughly* the same message (minor details being different is acceptable). In this case, the paraphrase relation can be looked at as a bidirectional entailment relation [19]. To exemplify such loose paraphrases, we show below a pair of sentences that has been tagged as paraphrase in the MSR Paraphrase Corpus:

Text A: Ricky Clemons' brief, troubled Missouri basketball career is over.

Text B: Missouri kicked Ricky Clemons off its team, ending his troubled career there.

In this example, the first sentence specifies that the career of Mr. Clemons was brief, while the second sentence specifies the reason why Mr. Clemons' career is over. The MSR Paraphrase corpus, our experimental data set, contains both types of sentential paraphrases, i.e. precise and loose paraphrases. This characteristic of the MSR corpus impacts the performance of general approaches, such as ours, to paraphrase identification that are not biased towards judging styles. A general approach to paraphrase identification assumes that two sentences are paraphrases of each other if they have exactly the same meaning.

3 Related work

Paraphrase identification has been explored in the past by many researchers, especially after the release of the MSR Paraphrase Corpus [5]. We describe in this section four previous studies that are most related to our approach and leave others out, e.g., [8, 21] due to space reasons.

Rus and colleagues [19] addressed the task of paraphrase identification by computing the degree of subsumption at lexical and syntactic level between two sentences in a bidirectional manner: from Text A to Text B and from Text B to Text A. The approach relied on a unidirectional approach that was initially developed to recognize the sentence-to-sentence relation of entailment [20]. Rus and colleagues' approach only used similarity to decide paraphrasing, ignoring dissimilarities which could be important to the final decision. The similarity was computed as a weighted sum of lexical matching, i.e. direct matching of words enhanced with synonymy information from WordNet, and syntactic matching, i.e., dependency overlap. Dependencies were derived from a phrase-based parser which outputs the major phrases in a sentence and organizes them hierarchically into a parse tree. Our approach has a better lexical component based on word semantics and a finer syntactic analysis component based on weighted dependencies. Furthermore, the use of phrase-based parsing in [19] limits the applicability of the approach to free-order languages for which dependency parsing is more suitable.

Corley and Mihalcea [3] proposed an algorithm that extends word-to-word similarity metrics into a text-to-text semantic similarity metric based on which they decide whether two sentences are paraphrases or not. To obtain the semantic similarity between individual words, they used the same WordNet similarity package as we do. Our approach has the advantage that it considers syntactic information, in addition to word semantics, to identify paraphrases.

Qiu and colleagues [18] proposed a two-phase architecture for paraphrase identification. In the first phase, they identified similarities between two sentences, while in the second phase the dissimilarities were classified with respect to their relevance in deciding the presence of paraphrase. Their approach uses predicate argument tuples that capture both lexical and syntactic dependencies among words to find similarities between sentences. The first

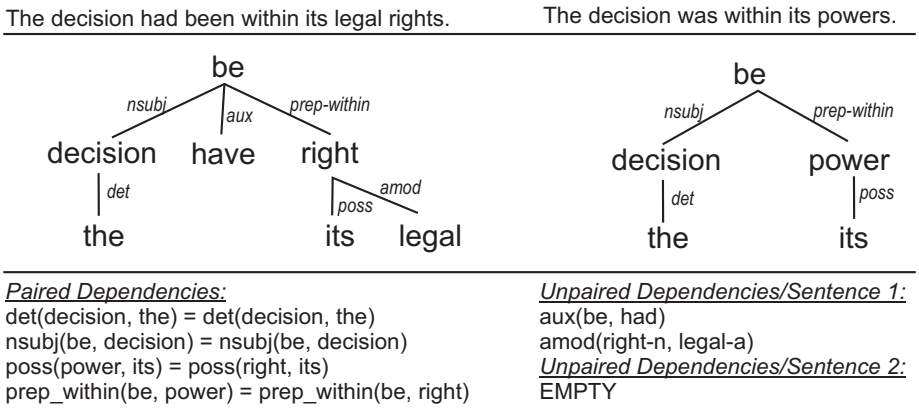

Figure 1: Example of dependency trees and sets of paired and non-paired dependencies.

phase is similar to our approach for detecting common dependencies. In the second phase, they used a supervised classifier to detect whether the dissimilarities are important. There are two advantages of our approach compared to Qiu and colleagues' approach (1) we use word semantics to compute similarities, (2) we take advantage of the dependency types and position in the dependency tree to weight dependencies as opposed to simply using non-weighted/unlabeled predicate-argument relations.

Zhang and Patrick [22] offer another ingenious solution to identify sentence-level paraphrase pairs by transforming source sentences into canonicalized text forms at the lexical and syntactic level, i.e. generic and simpler forms than the original text. One of the surprising findings is that a baseline system based on a supervised decision tree classifier with simple lexical matching features leads to best results compared to more sophisticated approaches that were experimented by them or others. They also revealed limitations of the MSR Paraphrase Corpus. The fact that their text canonicalization features did not lead to better than the baseline approach supports their findings that the sentential paraphrases, at least in the MSR corpus, share more words in common than one might expect given the standard definition of a paraphrase. The standard definition implies to use different words when paraphrasing. Zhang and Patrick used decision trees to classify the sentence pairs making their approach a supervised one as opposed to our approach which is minimally supervised - we only need to derive the value of the threshold from training data for which it is only necessary to know the distribution of true-false paraphrases in the training corpus and not the individual judgment for every instance in the corpus. They rely only on lexical and syntactic features while we also use semantic similarity factors.

We will compare the results of our approach on the MSR corpus with these related approaches. But first, we must detail the innerworkings of our approach.

4 Approach

As mentioned earlier, our approach is based on the observation that two sentences express the same meaning, i.e., are paraphrases, if they have all or many words and syntactic relations in common. Furthermore, the two sentences should have few or no dissimilar words or syntactic relations. In the example below, we show two sentences with high lexical and syntactic overlap. The different information, *legal rights* in the first sentence and *powers* in the second sentence, does not have a significant impact on the overall decision that the two sentences are paraphrases, which can be drawn based on the high degree of lexical and syntactic overlap.

Text A: *The decision was within its legal rights.*

Text B: *The decision was within its powers.*

On the other hand, there are sentences that are almost identical, lexically and syntactically, and yet they are not paraphrases because the few dissimilarities make a big difference. In the example below, there is a relatively "small" difference between the two sentences. Only the subject of the sentences is different. However, due to the importance of the subject relation to the meaning of any sentence the high similarity between the sentences is sufficiently dominated by the "small" dissimilarity to make the two sentences non-paraphrases.

Text A: *CBS is the leader in the 18 to 46 age group.*

Text B: *NBC is the leader in the 18 to 46 age group.*

Thus, it is important to assess both similarities and dissimilarities between two sentences S_1 and S_2 before making a decision with respect to them being paraphrases or not. In our approach, we capture the two aspects, similarity or dissimilarity, and then find the dominant aspect by computing a final paraphrase score as the ratio of the similarity and dissimilarity scores: Paraphrase(S_1, S_2)=Sim(S_1, S_2)/Diss(S_1, S_2). If the paraphrase score is above a learned threshold T the sentences are deemed paraphrases. Otherwise, they are non-paraphrases.

The similarity and dissimilarity scores are computed

based on dependency relations [7], which are asymmetric relationships between two words in a sentence, a *head* or modifee, and a *modifier*. A sentence can be represented by a set of dependency relations (see the bottom half of Figure 1). An example of dependency is the *subject* relation between *John* and *drives* in the sentence *John drives a car.* Such a dependency can be viewed as the triple *subj(John, drive)*. In the triplets the words are lemmatized, i.e., all morphological variations of a word are mapped onto its base form. For instance, *go, went, gone, going* are all mapped onto *go*.

The Sim(S_1, S_2) and Diss(S_1,S_2) scores are computed in three phases: (1) map the input sentences into sets of dependencies, (2) detect common and non-common dependencies between the sentences, and (3) compute the Sim(S_1, S_2) and Diss(S_1,S_2) scores. Figure 2 depicts the general architecture of the system in which the three processing phases are shown as the three major modules.

In the first phase, the set of dependencies for the two sentences is extracted using a dependency parser. We use both Minipar [11] and the Stanford parser [14] to parse the sentences. Because these parsers do not produce perfect output the reader should regard our results as a lower bound, i.e. results in the presence of parsing errors. Should the parsing been perfect, we expect our results to look better. The parser takes as input the raw sentence and returns as output a dependency tree (Minipar) or a list of dependencies (Stanford). In a dependency tree, every word in the sentence is a modifier of exactly one word, its head, except the head word of the sentence, which does not have a head. The head word of the sentence is the root node in the dependency tree. Given a dependency tree, the list of dependencies can be easily derived by traversing the tree and for each internal node, which is head of at least one dependency, we retrieve triplets of the form *rel(head, modifier)* where *rel* represents the type of dependency that links the node, i.e., the *head*, to one of its children, the *modifier*. Figure 1 shows the set of dependencies in the form of triplets for the dependency trees in the top half of the figure.

In this phase, we also gather positional information about each dependency in the dependency tree as we will need this information later when weighting dependencies in Phase 3. The position/depth of a dependency within the dependency tree is calculated as the distance from the root of the node corresponding to the head word of the dependency. Because the Stanford parser does not provide the position of the dependencies within the tree, we had to recursively reconstruct the tree based on the given set of dependency relations and calculate the relative position of each relation from the root.

The second phase in our approach identifies the common and non-common dependencies of the sentences, based on word semantics and syntactic information. Three sets of dependencies are generated in this phase: one set of *paired*/common dependencies and two sets of *unpaired* dependencies, one corresponding to each of the two sentences. To generate the paired and unpaired sets a two-step procedure is used. In the first step, we take one dependency from the shorter sentence in terms of number of dependencies (a computational efficiency trick) and identify dependencies of the same type in the other sentence. In the second step, we compute a dependency similarity score (d2dSim) using the word-to-word similarity metrics applied to the two heads and two modifiers of the matched dependencies. Heads and modifiers are mapped onto all the corresponding concepts in WordNet, one concept for each sense of the heads and modifiers. The similarity is computed among all senses/concepts of the two heads and modifiers, respectively, and then the maximum similarity is retained. If a word is not present in WordNet exact matching is used. The word-to-word similarity scores are combined into one final dependency-to-dependency similarity score by taking the weighted average of the similarities of the heads and modifiers. Intuitively, more weight should be given to the similarity score of heads and less to the similarity score of modifiers because heads are the more important words. Surprisingly, while trying to learn a good weighting scheme from the training data we found that the opposite should be applied: more weight should be given to modifiers (0.55) and less to heads (0.45). We believe this is true only for the MSR Paraphrase Corpus and this weighting scheme should not be generalized to other paraphrase corpora. The MSR corpus was built in such a way that favored highly similar sentences in terms of major content words (common or proper nouns) because the extraction of the sentences was based on keyword searching of major events from the web. With the major content words similar, the modifiers are the heavy lifters when it comes to distinguishing between paraphrase and non-paraphrase cases. Another possible approach to calculate the similarity score between dependencies is to rely only on the similarity of the most disimilar items, either heads or modifiers. We also tried this alternative approach, but it gave slightly poorer results (around 2% decrease in performance), and therefore, using a weighted scheme to calculate the similarity score for dependencies proved to be a better choice. The dependency-to-dependency similarity score needs to exceed a certain threshold for two matched dependencies to be deemed similar. Empirically, we found out from training data that a good value for this threshold would be *0.5*. Once a pair of dependencies is deemed similar, we place it into the paired dependencies set, along with the calculated dependency-to-dependency similarity value. All the dependencies that could not be paired are moved into the unpaired dependencies sets.

$$sim(S_1, S_2) = \sum_{d_1 \in S_1} max_{d_2 \in S_2^*}[d2dSim(d_1, d_2)]$$

$$diss(S_1, S_2) = \sum_{d \in \{unpairedS_1, unpairedS_2\}} weight(d)$$

In the third and final phase of our approach, two scores are calculated from the three dependency sets obtained

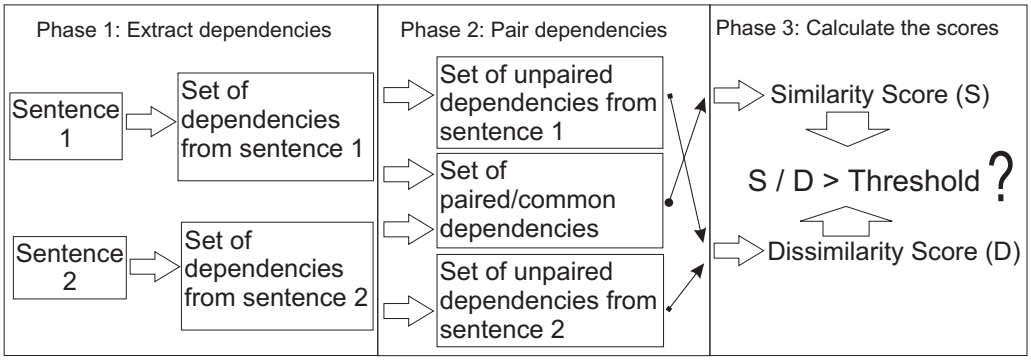

Figure 2: Architecture of the system.

Table 2: Performance and comparison of different approaches on the MS Paraphrase Corpus.

System	Accuracy	Precision	Recall	F-measure
Uniform baseline	0.6649	0.6649	1.0000	0.7987
Random baseline [3]	0.5130	0.6830	0.5000	0.5780
Lexical baseline (from Zhang et. al.)[22]	0.7230	0.7880	0.7980	0.7930
Corley and Mihalcea [3]	0.7150	0.7230	0.9250	0.8120
Qiu [18]	0.7200	0.7250	0.9340	0.8160
Rus - average [19]	0.7061	0.7207	0.9111	0.8048
Simple dependency overlap (Minipar) [13]	0.6939	0.7109	0.9093	0.7979
Simple dependency overlap (Stanford) [13]	0.6823	0.7064	0.8936	0.7890
Optimum results (Minipar)	0.7206	0.7404	0.8928	0.8095
Optimum results (Stanford)	0.7101	0.7270	0.9032	0.8056
No word semantics (Minipar)	0.7038	0.7184	0.9119	0.8037
No word semantics (Stanford)	0.7032	0.7237	0.8954	0.8005
No dependency weighting (Minipar)	0.7177	0.7378	0.8928	0.8079
No dependency weighting (Stanford)	0.7067	0.7265	0.8963	0.8025
No penalty for extra info (Minipar)	0.7067	0.7275	0.8936	0.8020
No penalty for extra info (Stanford)	0.7032	0.7138	0.9241	0.8055

in Phase 2: a cumulative *similarity score* and a cumulative *dissimilarity score*. The cumulative similarity score $\text{Sim}(S_1, S_2)$ is computed from the set of paired dependencies by summing up the dependency-to-dependency similarity scores (S_2^* in the equation for similarity score represents the set of remaining unpaired dependencies in the second sentence). Similarly, the dissimilarity score $\text{Diss}(S_1, S_2)$ is calculated from the two sets of unpaired dependencies. Each unpaired dependency is weighted based on two features: the depth of the dependency within the dependency tree and type of dependency. The depth is important because an unpaired dependency that is closer to the root of the dependency tree, e.g., the main verb/predicate of sentence, is more important to indicate a big difference between two sentences. In our approach, each unpaired dependency is initially given a perfect weight of 1.00, which is then gradually penalized with a constant value (*0.20* for the Minipar output and *0.18* for the Stanford output), the farther away it is from the root node. The penalty values were derived empirically from training data. Our tests show that this particular feature works well only when applied to the sets of unpaired dependencies. The second feature that we use to weight dependencies is the type of dependency. For example a *subj* dependency, which is the relation between the verb and its subject, is more important to decide paraphrasing than a *det* dependency, which is the relation between a noun and its determiner. Each dependency type is assigned an importance level between 0 (no importance) and 1 (maximum importance). The importance level for each dependency type has been established by the authors based on their linguistic knowledge and an analysis of the role of various dependency types in a subset of sentences from the training data.

Before comparing the similarity and dissimilarity scores, we consider one more feature that will affect the disimilarity score. This improvement, of a more statistical nature, is based on the idea that if one sentence contains a significant amount of extra information compared to the other

sentence although they do refer to the same action or event, then the relation between the two sentences is not a bidirectional relation of paraphrase, but rather a unidirectional relation of entailnment, so they should be evaluated as non-paraphrases. This extra information is recorded in our dependency sets by the fact that the set of unpaired dependencies from the longer, more detailed sentence is larger than the set of unpaired dependencies from the shorter sentence. To account for this statistical feature, we add an absolute value to the dissimilarity score, which was empirically chosen to be *14*, for every case when the set of unpaired dependencies from the longer sentence has more than *6* extra dependencies compared to the set of unpaired dependencies from the shorter sentence. We chose these optimal constants values to tweak this feature, based on a series of tests made on the MSR Paraphrase Corpus, and because of that, by including it into the system, the performance was improved significantly.

Once the Sim(S_1, S_2) and Diss(S_1, S_2) scores are available, the paraphrase score is calculated by taking the ratio between the similarity score, S, and the disimilarity score, D, and compare it to the optimum threshold T learned from training data. Formally, if $S/D > T$ then the instance is classified as paraphrase, otherwise is a non-paraphrase. To avoid division by zero for cases in which the two sentences are identical ($D = 0$) the actual implementation tests for $S > T * D$. To find the optimum threshold, we did an exhaustive search on the training data set, looking for the value which led to optimum accuracy. This is similar to the sigmoid function of the simple voted perceptron learning algorithm used in [3].

5 Summary of results

We experimented with our approach on the MSR Paraphrase Corpus [5]. The MSR Paraphrase Corpus is the largest publicly available annotated paraphrase corpus which has been used in most of the recent studies that addressed the problem of paraphrase identification. The corpus consists of 5801 sentence pairs collected from newswire articles, 3900 of which were labeled as paraphrases by human annotators. The whole set is divided into a training subset (4076 sentences of which 2753 are true paraphrases) which we have used to determine the optimum threshold T, and a test subset (1725 pairs of which 1147 are true paraphrases) that is used to report the performance results. We report results using four performance metrics: accuracy (percentage of instances correctly predicted out of all instances), precision (percentage of predicted paraphrases that are indeed paraphrases), recall (percentage of true paraphrases that were predicted as such), and f-measure (harmonic mean of precision and recall).

In Table 1 three baselines are reported: a uniform baseline in which the majority class (paraphrase) in the training data is always chosen, a random baseline taken from [3], and a lexical baseline taken from [22] which uses a supervised learning decision tree classifier with various lexical-matching features. We next show the results of others including results obtained using the simple dependency overlap method in [13]. The simple dependency overlap method computes the number of common dependency relations between the two sentences divided by the average number of relations in the two sentences. Our results are then presented in the following order: our best/state-of-the-art system, that uses all three features described in the previous section: word semantics, weighted dependencies and penalties for extra information, then a version of the proposed approach without word semantics (similarity in this case is 1 if words are identical, case insensitive, or 0 otherwise), then one without weighted dependencies, and finaly, one version where the instances with extra information found in one of their sentences are not penalized. The conclusion based on our best approach is that a mix of word semantics and weighted dependencies leads to better accuracy and in particular better precision. The best approach leads to significantly better results than the naive baselines and the simple dependency overlap ($p<0.001$ for the version with Minipar). The comparison between our best results and the results reported by [3] and [13] is of particular importance. These comparisons indicate that weighted dependencies and word semantics leads to better accuracy and precision than using only word semantics [3] or only simple dependency overlap [13].

All results in Table 1 were obtained with the *lin* measure from the WordNet similarity package, except the case that did not use WordNet similarity measures at all – the *No word semantics* row. This *lin* measure consistently led to the best performance in our experiments when compared to all the other measures offered by the WordNet similarity package.

For reference, we report in Table 3 results obtained when various word-to-word similarity metrics are used with an optimum threshold calculated from the *test data set*. For *lin* measure we report results with optimum test thresholds when using both parsers, Minipar and Stanford, while for the rest of the measures we only report results when using Minipar. We deem these results as one type of benchmark results for approaches that rely on WordNet similarity measures and dependencies as they were obtained by optimizing the approach on the testing data. As we can see from the table, the results are not much higher than the results in Table 1 where the threshold was derived from training data.

One important advantage that our system has over other approaches ([18], [22]) is that it does not rely too much on the training. The training data is used merely to tune the parameters, rather than for training a whole classifier. Since the only parameter whose value fully depends on the training data is the final threshold value, we've made another set of experiments where the threshold value depends only on one piece of information about the caracteristic of the test data set: the percentage of paraphrase instances within the data set. In other words, when calculating the threshold value, the system needs to know only what is the

Table 3: Accuracy results for different WordNet metrics with optimum test threshold values

Metric	Acc.	Prec.	Rec.	F
$\text{Lin}_{Minipar}$	.7241	.7395	.9032	.8132
$\text{Lin}_{Stanford}$	.7130	.7387	.8797	.8030
Path	.7183	.7332	.9058	.8105
L & C	.7165	.7253	.9233	.8124
W & P	.7188	.7270	.9241	.8138
J & C	.7217	.7425	.8901	.8097
Lesk	.7148	.7446	.8692	.8021
Vector	.7200	.7330	.9093	.8117
Vector pairs	.7188	.7519	.8614	.8029

probability of finding a paraphrase within the given data set. The system then tries to find a threshold value that splits the instances into two sets with the same distribution of instances as the given data set. For the testing part of the MSR Paraphrase data corpus the distribution value is 0.6649. We used this information to decide on a threshold and the results were no more than 2.09 percent below the optimum performance scores (for example on Minipar output and when excluding the WordNet similarity feature, the accuracy performance was only 0.06 percent less than when the threshold is calculated from the training data).

6 Discussion

One item worth discussing is the annotation of the MSR Paraphrase Corpus. Some sentences are intentionally labeled as paraphrases in the corpus even when the small dissimilarities are extremely important, e.g. different numbers. Below is a pair of sentences from the corpus in which the "small" difference in both the numbers and the anonymous *stocks* in Text A are not considered important enough for the annotators to judge the two sentences as non-paraphrases.

Text A: *The stock rose $2.11, or about 11 percent, to close on Friday at $21.51 on the New York Stock Exchange.*

Text B: *PG&E Corp. shares jumped $1.63 or 8 percent to $21.03 on the New York Stock Exchange on Friday.*

This makes the corpus more challenging and the fully-automated solutions look less powerful than they would on a paraphrase corpus that followed the standard interpretation of what a paraphrase is, i.e. the two texts have exactly the same meaning.

Another item worth discussing is the comparison of the dependency parsers. Our experimental results show that Minipar consistently outperforms Stanford, in terms of accuracy of our paraphrase identification approach. Minipar is also faster than Stanford, which first generates the phrase-based syntactic tree for a sentence and then extracts the corresponding sets of dependencies from the phrase-based syntactic tree. For instance, Minipar can parse 1725 pairs of sentences, i.e. 3450 sentences, in 48 seconds while Stanford parser takes 1926 seconds, i.e. 32 minutes and 6 seconds. A faster parser means it could be used in interactive environments, such as Intelligent Tutoring Systems, where a fast response is needed.

Finally, we would like to discuss the impact of word weighting on our method. We weighted words by their importance as derived from Wikipedia. The reason we did not mention the IDF feature in previous sections of this article is because the results are less accurate, at least on the MSR corpus. However, we think it is informative to discuss these results as they provide more insights on the problem of paraphrase identification. In particular it highlights the difficulty of the problem and the challenging nature of sentential paraphases in general.

Corley and Mihalcea [3] suggested that word weighting could improve methods to paraphrase identification. Translated into our approach, the idea is to weight words according to their importance (or specificity) when calculating the similarity and dissimilarity scores. In general, a word is more important if it is more specific. The specificity of a word can be approximated by its IDF (Inverted Document Frequency) value calculated from a large collection of documents. The theoretical assumption for using IDF on the problem of paraphrase identification is that when a word is considered highly specific (e.g. an unusual name or a very uncommon noun), this word should play an important role when deciding paraphrasing. To further motivate this assumtion, we show below a pair of sentences extracted from the MSR test data (instance #89), where by using IDF, our method succesfully classifies an otherwise failed instance:

Text A: *Emily Church is London bureau chief of CBS.MarketWatch.com.*

Text B: *Russ Britt is the Los Angeles Bureau Chief for CBS.MarketWatch.com.*

Notice that even though the predicates are the same and there is a rather long common noun phrase, which results in a significant number of identical dependencies between the two sentences, the subjects and the locations are completely different. Because there are two different pairs of named entities, which have high IDF values, this will put a significant weight on the dissimilarity score, which in the end will lead to the decision that the two sentences are in

fact not paraphrases.

We used Wikipedia, one of the largest and most diverse collection of documents freely available on the Internet, as the source for IDF values. IDF values are calculated from the DF (document frequency) of words which was extracted from over 2.2 million Wikipedia documents. To account for the data sparseness factor raised by the very high number of documents available, we calculated the IDF values from a maximum of 1 million (10^6) documents in the original collection. All DF values that exceeded the maximum number of documents were reduced to the maximum accepted value of 10^6. This means that the very few words that appeared in more than 1 million documents in Wikipedia will have the same minimal IDF value of 0. This means that the maximum absolute IDF value, for words that appeared in only one document is log(10^6) = 6. In the equations below, these values are normalized.

We experimented with two aproaches with IDF weights: 1) apply IDF weights to both paired and unpaired dependencies 2) apply IDF weights only to unpaired dependencies. We adjusted our previously presented scores such that they consider the IDF values of words. We added IDF-based weights on the paired dependencies in the similarity score and IDF-based weights on the unpaired dependencies in the dissimilarity score. The weights for paired and unpaired dependencies, respectively, are calculated according to the following formulae:

$$W_{idf}(d_{(w_1,w_2)}, d_{(w_3,w_4)}) = [\sum_{i=1}^{4} idf(w_i)]/(4 * 6)$$

$$W_{idf}(d_{(head,mod)}) = [idf(head) + idf(mod)]/(2 * 6)$$

Table 4 shows results with these two IDF-based methods when used with both dependency parsers (Minipar and Stanford). We present the same performance scores as in the previous section using optimum thresholds derived from both the training and the testing data sets. An interesting observation drawn from these results is that the first IDF method works better when used on the Minipar parser, while the second method works better on the Stanford parser. Another interesting effect of IDF values can be noted by comparing the IDF-based results with results in Table 1. It seems that when only unpaired dependencies are IDF-weighted the precision increases, at the expense of lower recall.

7 Summary and conclusions

In this article, we presented a novel approach to solve the problem of paraphrase identification. The approach uses word semantics and weighted dependencies to compute degrees of similarity at word/concept level and at syntactic level between two sentences. Based on the degree of similarity, sentences are being judged as paraphrases or not. The proposed approach offers state of the art performance. In particular, the approach offers high precision due to the use of syntactic information.

References

[1] Barzilay, R., and Lee, L. 2003. Learning to Paraphrase: An Unsupervised Approach Using Multiple Sequence Alignment. In *Proceedings of NAACL 2003*.

[2] Brockett, C., and Dolan, W. B. 2005. Support Vector Machines for Paraphrase Identification and Corpus Construction. In *Proceedings of the 3rd International Workshop on Paraphrasing*.

[3] Corley, C., and Mihalcea, R. 2005. Measuring the Semantic Similarity of Texts. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*. Ann Arbor, MI.

[4] Dagan, I., Glickman, O., and Magnini B. 2006. The PASCAL Recognising Textual Entailment Challenge. In Quiñonero-Candela, J.; Dagan, I.; Magnini, B.; d'Alché-Buc, F. (Eds.), *Machine Learning Challenges. Lecture Notes in Computer Science , Vol. 3944*, 177–190, Springer, 2006.

[5] Dolan, B.; Quirk, C.; and Brockett, C. 2004. Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources. In *Proceedings of COLING*, Geneva, Switzerland.

[6] Graesser, A.C.; Olney,A.; Haynes, B.; and Chipman, P. 2005. *Cognitive Systems: Human Cognitive Models in Systems Design.* Erlbaum, Mahwah, NJ. chapter AutoTutor: A cognitive system that simulates a tutor that facilitates learning through mixed-initiative dialogue.

[7] Hays, D. 1964. Dependency Theory: A Formalism and Some Observations. *Languages*, 40: 511–525.

[8] Kozareva, Z., and Montoyo, A. 2006. *Lecture Notes in Artificial Intelligence: Proceedings of the 5th International Conference on Natural Language Processing (Fin-TAL 2006)*. chapter Paraphrase Identification on the basis of Supervised Machine Learning Techniques.

[9] Ibrahim, A.; Katz B.; and Lin, J. 2003. Extracting Structural Paraphrases from Aligned Monolingual Corpora. in *Proceeding of the Second International Workshop on Paraphrasing*, (ACL 2003).

[10] Iordanskaja, L.; Kittredge, R.; and Polgere, A. 1991. Natural Language Generation in Artificial Intelligence and Computational Linguistics. *Lexical selection and paraphrase in a meaning-text generation model*, Kluwer Academic.

Table 4: Performance scores when using IDF values from Wikipedia.

Method	Parser	Optimum training threshold				Optimum testing threshold			
		Acc	Prec	Recall	F-score	Acc	Prec	Recall	F-score
Sim & Diss	Minipar	0.6922	0.7133	0.8980	0.7951	0.6957	0.7418	0.8317	0.7842
	Stanford	0.7049	0.7228	0.9024	0.8026	0.7101	0.7289	0.8980	0.8047
Diss only	Minipar	0.7049	0.7450	0.8457	0.7922	0.7113	0.7246	0.9128	0.8079
	Stanford	0.7043	0.7323	0.8753	0.7975	0.7072	0.7242	0.9041	0.8042

[11] Lin, D. 1993. Principle-Based Parsing Without Overgeneration. In *Proceedings of ACL*, 112–120, Columbus, OH.

[12] Lin, D. 1995. A Dependency-based Method for Evaluating Broad-coverage Parsers. In *Proceedings of IJCAI-95*.

[13] Lintean, M.; Rus, V.; and Graesser, A. 2008. Using Dependency Relations to Decide Paraphrasing In *Proceedings of the Society for Text and Discourse Conference 2008*.

[14] Marneffe, M. C; MacCartney, B.; and Manning, C. D. 2006. Generating Typed Dependency Parsers from Phrase Structure Parses. In *Proceeding of LREC 2006*.

[15] McNamara, D.S.; Boonthum, C.; Levinstein, I. B.; and Millis, K. 2007. *Handbook of Latent Semantic Analysis.* Erlbaum, Mahwah, NJ. chapter Evaluating self-explanations in iSTART: comparing word-based and LSA algorithms, 227–241.

[16] Miller, G. 1995 WordNet: A Lexical Database of English. Communications of the ACM, v.38 n.11, p.39-41.

[17] Patwardhan, S.; Banerjee, S.; and Pedersen, T. 2003. Using Measures of Semantic Relatedness for Word Sense Disambiguation. in *Proceeding of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics*, Mexico City, February.

[18] Qiu, L.; Kan M. Y.; and Chua T. S. 2006. Paraphrase Recognition via Dissimilarity Significance Classification. In *Proceeding of EMNLP*, Sydney 2006.

[19] Rus, V.; McCarthy, P. M.; Lintean, M.; McNamara, D. S.; and Graesser, A. C. 2008. Paraphrase Identification with Lexico-Syntactic Graph Subsumption. In *Proceedings of the Florida Artificial Intelligence Research Society International Conference (FLAIRS-2008).*

[20] Rus, V.; McCarthy, P. M.; McNamara, D. S.; and Graesser, A. C. 2008. A Study of Textual Entailment. *International Journal of Artificial Intelligence Tools*, August 2008.

[21] Wu, D. 2005. Recognizing Paraphrases and Textual Entailnment using Inversion Transduction Grammars. in *Proceeding of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailnment*, Ann Arbor, MI.

[22] Zhang, Y., and Patrick, J. 2005. Paraphrase Identification by Text Canonicalization In *Proceedings of the Australasian Language Technology Workshop 2005*, 160–166.