Using Dependency-Based Features to Take the "Para-farce" out of Paraphrase

Stephen Wan[†‡] **Mark Dras**[†] **Robert Dale**[†]
[†]Center for Language Technology
Div of Information Communication Sciences
Macquarie University
Sydney, NSW 2113
swan,madras,rdale@ics.mq.edu.au

Cécile Paris[‡]
[‡]Information and Communication Technologies
CSIRO
Sydney, Australia
Cecile.Paris@csiro.au

Abstract

As research in text-to-text paraphrase generation progresses, it has the potential to improve the quality of generated text. However, the use of paraphrase generation methods creates a secondary problem. We must ensure that generated novel sentences are not inconsistent with the text from which it was generated. We propose a machine learning approach be used to filter out inconsistent novel sentences, or *False Paraphrases*. To train such a filter, we use the Microsoft Research Paraphrase corpus and investigate whether features based on syntactic dependencies can aid us in this task. Like Finch et al. (2005), we obtain a classification accuracy of 75.6%, the best known performance for this corpus. We also examine the strengths and weaknesses of dependency based features and conclude that they may be useful in more accurately classifying cases of False Paraphrase.

1 Introduction

In recent years, interest has grown in paraphrase generation methods. The use of paraphrase generation tools has been envisaged for applications ranging from abstract-like summarisation (see for example, Barzilay and Lee (2003), Daumé and Marcu (2005), Wan et al. (2005)), question-answering (for example, Marsi and Krahmer (2005)) and Machine Translation Evaluation (for example, Bannard and Callison-Burch (2005) and Yves Lepage (2005)). These approaches all employ a loose definition of paraphrase attributable to Dras (1999), who defines a 'paraphrase pair' operationally to be "a pair of units of text deemed to be interchangeable". Notably, such a definition of paraphrase lends itself easily to corpora based methods. Furthermore, what the more modern approaches share is the fact that often they generate new paraphrases from raw text not semantic representations. The generation of paraphrases from raw text is a specific type of what is commonly referred to as *text-to-text generation* (Barzilay and Lee, 2003).

As techniques for generating paraphrases improve and basic concerns such as grammaticality are less of an issue, we are faced with an additional concern. That is, we must validate whether or not the generated novel sentence is in fact a paraphrase. It may be detrimental in some applications, for example abstract-like summarisation, to allow a novel sentence that is inconsistent with the content of the input text to be presented to the end user.

As an example of the type of inconsistencies that can arise from paraphrase generation, in Figure 1, we present two examples of generated sentences. In each example, a sentence pair is presented in which the second sentence was generated from an input news article statistically using a four-gram language model and a probabilistic word selection module. Although other paraphrase generation approaches differ in their underlying mechanisms[1], most generate a novel sentence that cannot be found verbatim in the input text.

The generated second sentence of the example is intended to be a paraphrase of the article headline. One might be convinced that the first exam-

[1]The details of the generation algorithm used for this example are peripheral to the focus of this paper and we direct the interested reader to Wan et al. (2005) for more details.

Proceedings of the 2006 Australasian Language Technology Workshop (ALTW2006), pages 131–138.

Example 1:
Original Headline:
European feeds remain calm on higher dollar.

Generated Sentence:
The European meals and feeds prices were firm on a stronger dollar; kept most buyers in this market.

Example 2:
Original Headline:
India's Gujral says too early to recognise Taleban.

Generated Sentence:
Prime Minister Inder Kumar Gujral of India and Pakistan to recognise the Taleban government in Kabul.

Figure 1: Two examples of generated novel sentences. Articles from the Reuters corpus were fed as input to the statistical summary generation system.

ple passes as a paraphrase, however the second is clearly inconsistent. We would like to identify this sentence pair as a false paraphrase. In addition to the ambiguity in the subject noun phrase (The Prime Minister of Pakistan is not the same as that of India), the generated sentence seems to ignore the adverbial phrase "too early" resulting in a farcical sentence that is almost the polar opposite of the headline.

We propose that an automatic classifier be employed to identify and filter out inconsistent novel sentences. To do so, we couch Paraphrase Classification as a supervised machine learning task and train a classifier on the Microsoft Research Paraphrase (MSR) Corpus (Dolan et al., 2004), a corpus specifically collected for this task. In particular, we are especially interested in exploring the use of syntactic dependency information in making this classification.

In this paper, we present our findings in training, testing and evaluating a paraphrase classifier. Section 2, we describe the research problem and outline related work in paraphrase classification. In Section 3, we present the features used in our classifier. Our classification experiments and results are described in Section 4. Before concluding, we discuss the strengths and weaknesses of dependency-based features in Section 5.

2 Paraphrase Classification and Related Work

In general, our task is to compare two sentences and produce a binary classification indicating if one is interchangeable with the other. To do so, we adopt the 'entailment decision' problem as put forward by the Pascal Recognising Textual Entailment (RTE) challenge (Dagan et al., 2005). The challenge requires participant systems to decide, given a pair of sentences, if the first sentence (referred to as the *hypothesis*) is entailed by the second sentence (referred to as the *text*). Although the task is that of logical entailment, participant systems are free to use any method, logic-based or not, to decide if sentence pairs are entailments. Crucial to this exercise is the simplification that the entailment decision be made on the basis of information within the sentences alone, and not on extensive representations of extensive world knowledge.

Similarly in our task, two sentences are checked for 'entailment'. In contrast to the RTE challenge, the MSR corpus has been collected based on a definition of paraphrase pairs as bi-directional entailment. That is, we must decide if one sentence is 'entailed' by its paired sentence, and vice versa. Sentence pairs were annotated as being True paraphrases if they were judged to be 'more or less semantically equivalent'. Otherwise, sentence pairs were annotated as being False Paraphrases.

Previous approaches to this classification task have focused on semantic equivalence at both the word and syntax level. Papers standardly report classification accuracy which is defined as the number of correctly classified test cases divided by the total number of test cases. Corley and Mihalcea (2005) use word equivalence features resulting in a classification accuracy of 71.5%. Zhang and Patrick (2005) examine string edit distance features and ngram overlap features collected on pairs of sentences in their canonical form. An overall accuracy of 71.9% is obtained.

Qiu et al. (2006) also focus on the detection of False Paraphrases. In this work, features based on predicate-argument information designed to indicate dissimilarity between the two sentences are collected. Using a support vector machine, this method results in an accuracy of 72.0%.

The best published result for this classification task is obtained by Finch et al. (2005) who obtained a classification accuracy of 74.96% using a

1 unigram recall
2 unigram precision
3 lemmatised unigram precision
4 lemmatised unigram recall
5 Bleu precision
6 Bleu recall
7 lemmatised Bleu precision
8 lemmatised Bleu recall
9 fmeasure
10 dependency relation precision
11 dependency relation recall
12 lemmatised dependency relation precision
13 lemmatised dependency relation recall
14 tree-edit distance (Zhang and Sasha algorithm)
15 lemmatised tree-edit distance (Zhang and Sasha algorithm)
16 difference in sentence length (in words)
17 absolute difference in sentence length (in words)

Figure 2: A list of all possible features

support vector machine trained on relatively simple features based on ngram overlap.

3 Features

In this paper, we decided to explore features encoding information about the relative difference between the structures of the two sentence. We thus experimented with a range of features ranging from differences in sentence length, to word overlap, to syntax dependency tree overlap, where the latter approximately represent predicate and argument structure. Figure 2 presents an overview of our features. We now describe each of these features.

3.1 *N*-gram Overlap: Features 1 to 9

We used variety of features based on word overlap and word-sequence overlap, where tokenisation is delimited by white space. We considered unigram overlap and explored two metrics, recall (feature 1) and precision (feature 2), where a precision score is defined as:

$$\text{p}recision = \frac{\textit{word-overlap}(sentence_1, sentence_2)}{\textit{word-count}(sentence_1)}$$

and recall is defined as:

$$\text{r}ecall = \frac{\textit{word-overlap}(sentence_1, sentence_2)}{\textit{word-count}(sentence_2)}$$

For each of the unigram overlap features described, we also computed a lemmatised variant. Both sentences were parsed by the Connexor parser[2] which provides lemmatisation information. For both sentences, each original word is replaced by its lemma. We then calculated our unigram precision and recall scores as before (features 3 and 4).

The Bleu metric (Papineni et al., 2002), which uses the geometric average of unigram, bigram and trigram precision scores, is implemented as feature 5. The score was obtained using the original Bleu formula[3] with a brevity penalty set to 1 (that is, the brevity penalty is ignored). Note that in our usage, there is only one 'reference' sentence. By reversing which sentence was considered the 'test' sentence and which was considered the 'reference', a recall version of Bleu was obtained (feature 6). Lemmatised versions provided features 7 and 8.

Finally, because of the bi-directionality property of paraphrase, the F-Measure[4], which combines both precision and recall into a single score using the harmonic mean, was implemented as feature 9.

3.2 Dependency Relation Overlap: Features 10 to 13

Overlap of dependency tuples has been cited by other researchers as being a useful approximate representation of sentence meaning (Mollá, 2003). Indeed, Rouge-BE (Hovy et al., 2005), a recall-based metric similar to this feature, is currently being used in summarisation evaluations to measure the content overlap of summaries with source documents.

We again make use of the Connexor parser, this time to provide a dependency structure analysis of a sentence. Each sentence was parsed resulting in a set of dependency relations (one set per sentence). A relation is simply a pair of words in a parent-child relationship within the dependency tree[5], refered to as head-modifier relationships. In this paper, we ignored the label of the relationships which indicates the semantic role. The next series of features examines the use of features based on an overlap of such head-modifier relations (hereafter, *relations*) between sentences.

Feature 10 is the precision score calculated from the overlap according to the following formula:

[2] see http://www.connexor.com/software/syntax/
[3] http://www.ics.mq.edu.au/∼szwarts/downloads/Bleu.cpp
[4] http://www.ics.mq.edu.au/∼szwarts/downloads/FMeasure.cpp
[5] That is, an edge and the two nodes on either side

$$\text{precision}_d = \frac{|\textit{relations}(sentence_1) \cap \textit{relations}(sentence_2)|}{|\textit{relations}(sentence_1)|}$$

where $precision_d$ stands for *dependency precision* and $relations(sentence_i)$ is the set of head-modifier relations for some sentence.

A recall variant of this feature was also used (feature 11) and is defined as:

$$\text{recall}_d = \frac{|\textit{relations}(sentence_1) \cap \textit{relations}(sentence_2)|}{|\textit{relations}(sentence_2)|}$$

Lemmatised versions of these features were used in feature 12 and 13.

3.3 Dependency Tree-Edit Distance: Features 14 and 15

As another measure of how alike or different the two sentences are from each other, we decided to examine how similar their respective dependency trees were. Ordered tree-edit distance algorithms are designed to find the least costly set of operations that will transform one tree into another. In our case, we want to find the cost of transforming dependency parse trees.

Our implementation is based on the dynamic programming algorithm of Zhang and Shasha (1989). The algorithm finds the optimum (cheapest) set of tree-edit operations in polynomial time. This algorithm has been used in the past in Question-Answering as a means of scoring similarity between questions an candidate answers (Punyakanok et al., 2004). In a similar vein to our work here, it has also been used in the RTE challenge (Kouylekov and Magnini, 2005).

We calculated the tree-edit distance over the syntactic dependency parse trees returned by the Connexor parser. Inserting, deleting and renaming nodes, or words, into a dependency tree, were all given an equal cost.

The cost returned by the algorithm is simply the sum of all operations required to transform one tree into the other. This cost was normalised by the number nodes in the target dependency tree to produce a value between 0 and 1 (feature 14). A lemmatised variant of this feature was obtained by first lemmatising the two dependency trees (feature 15).

3.4 Surface Features: Features 16 and 17

Finally, we looked at the difference in length of the two sentences as measured in words by subtracting one length from the other. This difference (feature 16) could be a negative or positive integer. An absolute variant was used in Feature 17.

4 Experiments

4.1 Data and Software

The Microsoft Paraphrase Corpus (MSR) (Dolan et al., 2004) is divided into a training set and a test set. In the original training set, there were 2753 True Paraphrase pairs and 1323 False Paraphrase pairs. The original test set contained 1147 True Paraphrases pairs and 578 False Paraphrases pairs.

We first parsed the MSR paraphrase corpus using the Connexor parser. While Connexor is by no means a perfect parser, it usually produces partial parses if a more complete one is not possible. Our experience with Connexor is that these partial parses have tended to be useful. We are currently comparing Connexor to other dependency parsers to see what kinds of errors it introduces. However, due to time constraints, utilising this information is left for future work.

Because there were several cases which broke our parsing scripts (due to an occasional non-XML character), our training and test sets were slightly smaller. These included 2687 True Paraphrase pairs and 1275 False Paraphrase pairs in our training set, and 1130 True Paraphrase pairs and 553 False Paraphrases pairs in our test set.

We used the open source WEKA Data Mining Software (Witten and Frank, 2000). A selection of commonly used techniques was experimented with including: a Naive Bayes learner (bayes.NaiveBayes), a clone of the C4.5 decision tree classifier (trees.J48), a support vector machine with a polynomial kernel (functions.SMO), and K-nearest neighbour (lazy.IBk). Each machine learning technique was used with the default configurations provided by WEKA. The baseline learning technique (rules.ZeroR) is simply the performance obtained by choosing the most frequent class. We report only the results obtained with the support vector machine as this machine learning method consistently outperformed the other methods for this task.

Finally, we tested for significance between correct and incorrect classifications of the two systems being compared using the Chi-squared test

Features	Acc.	C1-prec.	C1-recall	C1-Fmeas.	C2-prec	C2-recall	C2-Fmeas.
lemma'd 1-grams	0.69	0.52	0.56	0.54	0.78	0.75	0.76
1-grams	0.73	0.63	0.39	0.49	0.75	0.89	0.81
ZeroR	0.66	0	0	0	0.67	1	0.80
Finch	0.75	0.69	0.46	0.55	0.77	0.90	0.83
Best Features	0.75	0.70	0.46	0.55	0.77	0.90	0.83

Table 1: Classification performance of the best feature vector found. C1 denotes False Paraphrase pairs, C2 denotes True Paraphrase pairs. C1 scores for the best system in Finch et al. (2005) were calculated from the C2 scores published.

Features	Acc.	C1-prec.	C1-recall	C1-Fmeas.	C2-prec	C2-recall	C2-Fmeas.
Dependencies	0.75	0.67	0.45	0.54	0.77	0.89	0.82
Bleu	0.75	0.69	0.45	0.55	0.77	0.90	0.83

Table 2: Classification performance comparison between dependency features and *n*-gram features. C1 denotes False Paraphrase pairs, C2 denotes True Paraphrase pairs.

implemented in the R-Statistical package.

4.2 Best Performing Feature Set

Through experimentation, we found the best performing classifier used all features except for lemmatised unigrams[6]. The results on the test set are presented in Table 1. Accuracy is the number of correctly classified test cases (regardless of class) divided by the total number of test cases. Recall for True Paraphrase class is defined as the number of cases correctly classified as True Paraphrase divided by the total number of True Paraphrase test cases. Precision differs in that the denominator is the total number of cases (correct or not) classified as True Paraphrase by the system. The F-Measure is the harmonic mean of recall and precision. Likewise, the recall, precision and f-measure for the False Paraphrase class is defined analogously.

We note an improvement over majority class baseline, a unigram baseline and a lemmatised unigram baseline. In particular, the addition of our features add a (3%) improvement in overall accuracy compared to the best performing baseline using (unlemmatised) unigram features. Improvement over this baseline (and hence the other baselines) was statistically significant (χ-squared = 4.107, df = 1, p-value = 0.04271). Our performance was very close to that reported by (Finch et al. (2005) is not statistically significant. The system employed by Finch et al. (2005) uses features that are predominantly based on the Bleu metric.

The improvement of the unigram-based classifier is 6 percentage points above the majority class is also significant (χ-squared = 11.4256, df = 1, p-value = 0.0007244). Interestingly, results from using just precision and recall unigram features[7] without lemmatisation are comparable to Finch et al. (2005). Indeed, a principal components analysis showed that unigram features were the most informative accounting for 60% of cases.

Oddly, the results for the lemmatised unigram features are poorer even the majority class baseline, as demonstrated by a lower True Paraphrase F-Measure. Why this is so is puzzling as one would expect lemmatisation, which abstracts away from morphological variants, to increase the similarity between two sentences. However, we note that two sentences can differ in meaning with the inclusion of a single negative adverb. Thus, an increased similarity for all training cases may simply make it much harder for the machine learning algorithm to differentiate effectively between classes.

5 The Strengths and Weaknesses of Dependency Features

The previous experiment showed that together, Bleu-based features and dependency-based features were able to achieve some improvement. We were also interested in comparing both feature types to see if one had any advantage over the other.

We note that bigrams and dependencies in ac-

[6] features: 1,2,5,6,7,8,9,10,11,12,13,14,15,16,17

[7] features: 1,2

2685	True		False		1275
	Bleu	Dep	Bleu	Dep	
A: 44	F	T	F	T	B: 41
C: 2342	T	T	T	T	D: 654
E: 50	T	F	T	F	F: 49
G: 249	F:	F	F	F	H: 531

Table 3: Error Analysis showing the number of cases in which the Bleu-based classifier disagreed with the Dependency-based classifier. 'T' and 'F' stand for predicted 'TRUE' and predicted 'FALSE'. The other capital letters A to H are cell labels for ease of reference.

tuality encode very similar types of information, that is a pairing of two words. In the case of dependency relations, the words are connected via some syntactic dependency structure, whereas word pairs in bigrams (for example) are merely 'connected' via the property of adjacency. However, Collins (1996) points out that in English, around 70% of dependencies are in fact adjacent words. Thus one would think that Bleu and dependency features have similar discriminative power.

Two versions of the classifier were trained based on two separate sets of features that differed only in that one included four Bleu features[8] whereas the other included four dependency overlap features[9]. All other features were kept constant.

The results obtained on the test set are presented in Table 2. As expected, the two seem to perform at the same level of performance and were not statistically different. This is consistent with the same levels of performance observed between our system and that of Finch et al. (2005) in Table 1. However, it would also be interesting to know if each feature might be more suitable for different types of paraphrase phenomena.

5.1 Differences in Predictions

To understand the strengths and weaknesses of *n*-gram and dependency features, we performed an analysis of the cases where they differed in their classifications. We tested the two classifiers in Table 2 on the training set to gives us an indication of the ideal situation in which the training data reflects the testing data perfectly. Table 3 presents this analysis. For example, Cell A indicates that there were 44 true paraphrase cases that were correctly classified by the dependency-based classifier but misclassified by the bleu-based classifier.

For the dependency-based classifier to outperform the Bleu-based classifier in classifying True Paraphrases, Cell A must be greater than Cell E. That is, the number of cases in which the Dependency-based classifier improves the true positive count must outweigh the false negative count. Unfortunately, this isn't the case.

Correspondingly, for the dependency-based classifier to outperform the Bleu-based classifier in classifying True Paraphrases, Cell F must be greater than Cell B. In this case, the dependency-based classifier does performs better than the Bleu-based classifier.

One could summarise this analysis by saying that the dependency-based classifier tended make pairs look more dissimilar than the Bleu-based classifier. To gain some insight as to how to create features that build on the strengths of the two feature types, for example using dependency based features to better classify False Paraphrase cases, we manually examined the sentence pairs from the training set in which the two classifiers disagreed in the hopes of identifying reasons for the erroneous classifications.

5.2 Wrongly Predicting 'True' on False Paraphrase cases

Table 3 suggest that dependency-features might improve the precision and recall of the False Paraphrase class. Thus, we focused on the cases where the dependency-based classifier incorrectly classified False Paraphrase cases. We found several situations where this was the case. Often, some portion of both sentences would share a high degree of word overlap that we suspect was confusing our classifier.

In the case of Sentence Pair 1, a title is quoted in both increasing the textual similarity. However, on closer inspection the clauses are different, specifically the main clause verb and subject. In Sentence Pair 2, we notice this also happened with long noun phrases relating to organisations.

Sentence Pair 1:

Details of the research appear in the Nov. 5 issue of the Journal of the American Medical Association.

The results, published in the Journal of the American Medical Association, involved just 47 heart attack patients.

Sentence Pair 2:

The Securities and Exchange Commission has also initiated an informal probe of Coke.

[8] features: 1,2,5,6,7,8,16,17

[9] features: 1,2,10,11,12,13,16,17

That federal investigation is separate from an informal inquiry by the Securities and Exchange Commission.

Similarly, despite high overlap in both words and dependency relations, some sentences pairs simply differed in the focus of the main clause as in Sentence Pair 3. We see a similar problem in Sentence Pair 4 in which the main clause of the first sentence matches the subordinate clause of the second but the focus of each is different.

Sentence Pair 3:
He replaces Ron Dittemore, who announced his resignation in April.
Dittemore announced his plans to resign on April 23.

Sentence Pair 4:
Peterson told police he fished alone in San Francisco Bay on Christmas Eve, returning to an empty house.
Peterson told police he left his wife at about 9:30 a.m. on Dec. 24 to fish alone in San Francisco Bay.

5.3 Follow-on Experiment

One of the reasons why our use of dependencies leads to the problem exemplified by Sentence Pairs 1 to 4, is that all dependency pairs are treated equal. However, clearly, some are more equal than others. Dependency relations concerning the main verb and subject ought to count for more.

The simplest way to model this inequality is to give more weight to relations higher up in the tree as these will tend to express the semantics of the main clause.

Our extra set of features represent the weighted dependency precision, the weighted dependency recall, and the lemmatised versions of both those feature types. In total four new features were added.

To begin with nodes were scored with the size of their subtrees. We then traversed the tree breadth-first where siblings were traversed in decreasing order with respect to the size of their respective subtrees. Nodes were given a position number according to this traversal. Each node was then weighted by the inverse of its position in this ordering. Thus, the root would have weight 1 and it's heaviest child node would receive a weight of 0.5. The relation weight is simply the product of the weights of the nodes.

The overall score for the sentence pair is simply the sum of relation weights normalised accordingly to yield precision and recall scores.

The results on the test set are presented in Table 4. Note that this result differs drastically from all the previous systems reported. In contrast to these systems, our last classifier seems to produce good precision results (83%) for the True Paraphrase class at the expense of recall performance. Consequently, it has the best performing recall for False Paraphrase (71%) out of all the systems tested. This gain in recall, while compensated by a loss in precision, ultimately leads to the highest F-Measure observed for this class (61%), an improvement on Finch et al. (2005). This seems to suggest that our additional features are doing what we hoped they would, improve the classification of the False Paraphrase class. However, this effect also has an overall harmful effect on our classifier which may be over-classifying cases as False Paraphrase. Thus, a drop in accuracy is observed. Avenues to integrate the benefits of these new features without harming our overall accuracy remain further work.

6 Conclusion

In this paper, we presented work on Paraphrase Classification with the Microsoft Research Paraphrase Corpus. We show that dependency-based features in conjunction with bigram features improve upon the previously published work to give us the best reported classification accuracy on this corpus, equal with Finch et al. (2005). In addition, using weighted dependency overlap seems to provide promise, yielding the best F-Measure for False Paraphrase classification seen so far. We conclude that dependency features may thus be useful in more accurately classifying cases of False Paraphrase. In future work, we will build upon the strengths of the weighted dependency features to improve the classifier further.

We also argue that Paraphrase Classification be used as a means to validate whether or not, in the context of abstract-like summarisation, a generated paraphrase reflected the source material. For this purpose, performance of precision and recall of the False Paraphrase classification seems more important, as we do not want to waste the end user's time by generation misleading information.

7 Acknowledgements

This work was funded by the Centre for Language Technology at Macquarie University and the CSIRO Information and Communication Technology Centre. We would like to thank the research groups of both organisations as well as the anonymous reviewers for useful comments and

Features	Acc.	C1-prec.	C1-recall	C1-Fmeas.	C2-prec	C2-recall	C2-Fmeas.
Best Features	75.63	0.70	0.46	0.55	0.77	0.90	0.83
All Features	71.00	0.55	0.71	0.61	0.83	0.72	0.77

Table 4: Classification performance of the best feature vector found and the feature vector including weighted dependency overlap. C1 denotes False Paraphrase pairs, C2 denotes True Paraphrase pairs.

feedback.

References

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, Ann Arbor, Michigan.

Regina Barzilay and Lillian Lee. 2003. Learning to paraphrase: An unsupervised approach using multiple-sequence alignment. In *HLT-NAACL 2003: Main Proceedings*.

Michael John Collins. 1996. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the Thirty-Fourth Annual Meeting of the Association for Computational Linguistics*, San Francisco.

Courtney Corley and Rada Mihalcea. 2005. Measuring the semantic similarity of texts. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, Ann Arbor, Michigan.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*.

Hal Daumé and Daniel Marcu. 2005. Bayesian multi-document summarization at mse. In *Proceedings of the Workshop on Multilingual Summarization Evaluation (MSE)*, Ann Arbor, MI.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of Coling 2004*, Geneva, Switzerland.

Mark Dras. 1999. *Tree Adjoining Grammar and the Reluctant Paraphrasing of Text*. Ph.D. thesis, Macquarie University, Australia.

Andrew Finch, Young Sook Hwang, and Eiichiro Sumita. 2005. Using machine translation evaluation techniques to determine sentence-level semantic equivalence. In *Proceedings of The 3rd International Workshop on Paraphrasing (IWP2005)*.

Eduard Hovy, Chin-Yew Lin, and Liang Zhou. 2005. Evaluating duc 2005 using basic elements. In *Proceedings of Document Understanding Conference (DUC 2005)*, Vancouver, B.C. Canada.

Milen Kouylekov and Bernardo Magnini. 2005. Recognizing textual entailment with tree edit distance algorithms. In *Proceedings of the PASCAL Challenges Workshop: Recognising Textual Entailment Challenge*.

Erwin Marsi and Emiel Krahmer. 2005. Explorations in sentence fusion. In *The Proceedings of the European Workshop on Natural Language Generation 2005*, Aberdeen, Scotland.

Diego Mollá. 2003. Towards semantic-based overlap measures for question answering. In *Proceedings of the Australasian Language Technology Workshop (ALTW 2003)*, Melbourne.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, Philadelphia, July.

V. Punyakanok, D. Roth, and W. Yih. 2004. Mapping dependencies trees: An application to question answering. In *Proceedings of AI & Math*.

Long Qiu, Min-Yen Kan, and Tat-Seng Chua. 2006. Paraphrase recognition via dissimilarity significance classification. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, Sydney, Australia.

Stephen Wan, Mark Dras, Robert Dale, and Cécile Paris. 2005. Towards statistical paraphrase generation: preliminary evaluations of grammaticality. In *Proceedings of The 3rd International Workshop on Paraphrasing (IWP2005)*, Jeju Island, South Korea.

Ian H. Witten and Eibe Frank. 2000. *Data mining: practical machine learning tools and techniques with Java implementations*. San Francisco, CA, USA.

Etienne Denoual Yves Lepage. 2005. Automatic generation of paraphrases to be used as translation references in objective evaluation measures of machine translation. In *Proceedings of The 3rd International Workshop on Paraphrasing (IWP 2005)*.

Yitao Zhang and Jon Patrick. 2005. Paraphrase identification by text canonicalization. In *Proceedings of the Australasian Language Technology Workshop 2005*, Sydney, Australia.

K. Zhang and D. Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal of Computing*, 18(6).