Paraphrase Recognition via Dissimilarity Significance Classification

Long Qiu, Min-Yen Kan and **Tat-Seng Chua**
Department of Computer Science
National University of Singapore
Singapore, 117543
{qiul,kanmy,chuats}@comp.nus.edu.sg

Abstract

We propose a supervised, two-phase framework to address the problem of paraphrase recognition (PR). Unlike most PR systems that focus on sentence similarity, our framework detects dissimilarities between sentences and makes its paraphrase judgment based on the significance of such dissimilarities. The ability to differentiate significant dissimilarities not only reveals what makes two sentences a non-paraphrase, but also helps to recall additional paraphrases that contain extra but insignificant information. Experimental results show that while being accurate at discerning non-paraphrasing dissimilarities, our implemented system is able to achieve higher paraphrase recall (93%), at an overall performance comparable to the alternatives.

1 Introduction

The task of sentence-level paraphrase recognition (PR) is to identify whether a set of sentences (typically, a pair) are semantically equivalent. In such a task, "equivalence" takes on a relaxed meaning, allowing sentence pairs with minor semantic differences to still be considered as paraphrases.

PR can be thought of as synonym detection extended for sentences, and it can play an equally important role in natural language applications. As with synonym detection, applications such as summarization can benefit from the recognition and canonicalization of concepts and actions that are shared across multiple documents. Automatic construction of large paraphrase corpora could mine alternative ways to express the same concept, aiding machine translation and natural language generation applications.

In our work on sentence-level PR, we have identified two main issues through observation of sample sentences. The first is to identify all discrete *information nuggets*, or individual semantic content units, shared by the sentences. For a pair of sentences to be deemed a paraphrase, they must share a substantial amount of these nuggets. A trivial case is when both sentences are identical, word for word. However, paraphrases often employ different words or syntactic structures to express the same concept. Figure 1 shows two sentence pairs, in which the first pair is a paraphrase while the second is not. The paraphrasing pair (also denoted as the $+pp$ class) use different words. Focusing just on the matrix verbs, we note differences between "injured" and "hurt". A paraphrase recognition system should be able to detect such semantic similarities (despite the different syntactic structures). Otherwise, the two sentences could look even less similar than two non-paraphrasing sentences, such as the two in the second pair. Also in the paraphrasing pair, the first sentence includes an extra phrase "Authorities said." Human annotators tend to regard the pair as a paraphrase despite the presence of this extra information nugget.

Paraphrase (+pp):
Authorities said a young man *injured* Richard Miller.
Richard Miller was *hurt* by a young man.
Non-Paraphrase (-pp):
The technology-laced Nasdaq Composite Index .IXIC *added* 1.92 points, or 0.12 percent, at 1,647.94.
The technology-laced Nasdaq Composite Index .IXIC *dipped* 0.08 of a point to 1,646.

Figure 1: Examples: Paraphrasing & Non-paraphrasing

This leads to the second issue: how to recognize when such extra information is extraneous with respect to the paraphrase judgment. Such paraphrases are common in daily life. In news articles describing the same event, paraphrases are widely used, possibly with extraneous information.

We equate PR with solving these two issues, presenting a natural two-phase architecture. In the first phase, the nuggets shared by the sentences are identified by a pairing process. In the second phase, any unpaired nuggets are classified as significant or not (leading to $-pp$ and $+pp$ classifications, respectively). If the sentences do not contain unpaired nuggets, or if all unpaired nuggets are insignificant, then the sentences are considered paraphrases. Experiments on the widely-used MSR corpus (Dolan et al., 2004) show favorable results.

We first review related work in Section 2. We then present the overall methodology and describe the implemented system in Section 3. Sections 4 and 5 detail the algorithms for the two phases respectively. This is followed with our evaluation and discussion of the results.

2 Related Work

Possibly the simplest approach to PR is an information retrieval (IR) based "bag-of-words" strategy. This strategy calculates a cosine similarity score for the given sentence set, and if the similarity exceeds a threshold (either empirically determined or learned from supervised training data), the sentences are paraphrases. PR systems that can be broadly categorized as IR-based include (Corley and Mihalcea, 2005; Brockett and Dolan, 2005). In the former work, the authors defined a *directional similarity* formula reflecting the semantic similarity of one text "with respect to" another. A word contributes to the directional similarity only when its counterpart has been identified in the opposing sentence. The associated word similarity scores, weighted by the word's specificity (represented as inverted document frequency, *idf*), sum to make up the directional similarity. The mean of both directions is the overall similarity of the pair. Brockett and Dolan (2005) represented sentence pairs as a feature vector, including features (among others) for sentence length, edit distance, number of shared words, morphologically similar word pairs, synonym pairs (as suggested by WordNet and a semi-automatically constructed thesaurus). A support vector machine is then trained to learn the $\{+pp, -pp\}$ classifier.

Strategies based on bags of words largely ignore the semantic interactions between words. Weeds et al. (2005) addressed this problem by utilizing parses for PR. Their system for phrasal paraphrases equates paraphrasing as distributional similarity of the partial sub-parses of a candidate text. Wu (2005)'s approach relies on the generative framework of Inversion Transduction Grammar (ITG) to measure how similar two sentences arrange their words based on edit distance.

Barzilay and Lee (2003) proposed to apply multiple-sequence alignment (MSA) for traditional, sentence-level PR. Given multiple articles on a certain type of event, sentence clusters are first generated. Sentences within the same cluster, presumably similar in structure and content, are then used to construct a lattice with "backbone" nodes corresponding to words shared by the majority and "slots" corresponding to different realization of arguments. If sentences from different clusters have shared arguments, the associated lattices are claimed to be paraphrase. Likewise, Shinyama et al. (2002) extracted paraphrases from similar news articles, but use shared named entities as an indication of paraphrasing. It should be noted that the latter two approaches are geared towards acquiring paraphrases rather than detecting them, and as such have the disadvantage of requiring a certain level of repetition among candidates for paraphrases to be recognized.

All past approaches invariably aim at a proper similarity measure that accounts for all of the words in the sentences in order to make a judgment for PR. This is suitable for PR where input sentences are precisely equivalent semantically. However, for many people the notion of paraphrases also covers cases in which minor or irrelevant information is added or omitted in candidate sentences, as observed in the earlier example. Such extraneous content should not be a barrier to PR if the main concepts are shared by the sentences. Approaches that focus only on the similarity of shared contents may fail when the (human) criteria for PR include whether the unmatched content is significant or not. Correctly addressing this problem should increase accuracy. In addition, if extraneous portions of sentences can be identified, their confounding influence on the sentence similarity judgment can be removed,

leading to more accurate modeling of semantic similarity for both recognition and acquisition.

3 Methodology

As noted earlier, for a pair of sentences to be a paraphrase, they must possess two attributes:

1. *similarity*: they share a substantial amount of information nuggets;

2. *dissimilarities are extraneous*: if extra information in the sentences exists, the effect of its removal is not significant.

A key decision for our two-phase PR framework is to choose the representation of an information nugget. A simple approach is to use representative words as information nuggets, as is done in the SimFinder system (Hatzivassiloglou et al., 2001).

Instead of using words, we choose to equate information nuggets with *predicate argument tuples*. A predicate argument tuple is a structured representation of a verb predicate together with its arguments. Given a sentence from the example in Figure 1, its predicate argument tuple form in PropBank (Kingsbury et al., 2002) format is:

target(predicate):	hurt
arg0:	a young man
arg1:	Richard Miller

We feel that this is a better choice for the representation of a nugget as it accounts for the action, concepts and their relationships as a single unit. In comparison, using fine-grained units such as words, including nouns and verbs may result in inaccuracy (sentences that share vocabulary may not be paraphrases), while using coarser-grained units may cause key differences to be missed. In the rest of this paper, we use the term *tuple* for conciseness when no ambiguity is introduced.

An overview of our paraphrase recognition system is shown in Figure 2. A pair of sentences is first fed to a syntactic parser (Charniak, 2000) and then passed to a semantic role labeler (ASSERT; (Pradhan et al., 2004)), to label predicate argument tuples. We then calculate normalized tuple similarity scores over the tuple pairs using a metric that accounts for similarities in both syntactic structure and content of each tuple. A thesaurus constructed from corpus statistics (Lin, 1998) is utilized for the content similarity.

We utilize this metric to greedily pair together the most similar predicate argument tuples across

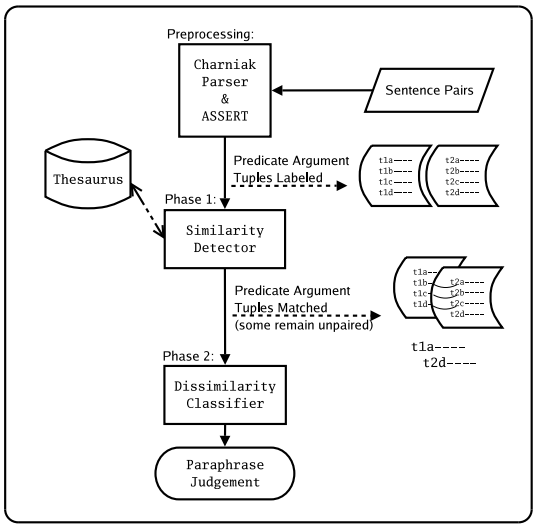

Figure 2: System architecture

sentences. Any remaining unpaired tuples represent extra information and are passed to a dissimilarity classifier to decide whether such information is significant. The dissimilarity classifier uses supervised machine learning to make such a decision.

4 Similarity Detection and Pairing

We illustrate this advantage of using predicate argument tuples from our running example. In Table 1, one of the model sentences is shown in the middle column. Two edited versions are shown on the left and right columns. While it is clear that the left modification is an example of a paraphrase and the right is not, the version on the left involves more changes in its syntactic structure and vocabulary. Standard word or syntactic similarity measures would assign the right modification a higher similarity score, likely mislabeling one or both modifications.

In contrast, semantic role labeling identifies the dependencies between predicates and their arguments, allowing a more precise measurement of sentence similarity. Assuming that the arguments in predicate argument tuples are assigned the same role when their roles are comparable[1], we define the similarity score of two tuples T_a and T_b as the weighted sum of the pairwise similarities of all their shared constituents C=$\{(c_a, c_b)\}$ (c being either the *target* or one of the *arguments* that both

[1] ASSERT, which is trained on the Propbank, only guarantees consistency of **arg0** and **arg1** slots, but we have found in practice that aligning **arg2** and above arguments do not cause problems.

Sentence	Modification 1: **paraphrase**	**Model Sentence**	Modification 2: **non-paraphrase**
	Richard Miller was hurt by a young man.	Authorities said a young man injured Richard Miller.	Authorities said Richard Miller injured a young man.
(Paired) Tuples		**target**: said **arg0**: Authorities **arg1**: a young man injured Richard Miller	**target**: said **arg0**: Authorities **arg1**: Richard Miller injured a young man
	target: hurt **arg0**: a young man **arg1**: Richard Miller	**target**: injured **arg0**: a young man **arg1**: Richard Miller	**target**: injured **arg0**: Richard Miller **arg1**: a young man

Table 1: Similarity Detection: pairing of predicate argument tuples

tuples have):

$$Sim(T_a, T_b) = \frac{1}{\alpha} \sum_{c=\{target, arg_{shared}\}} Sim(c_a, c_b) * \ w_{target}^{c==target} \quad (1)$$

where normalization factor α is the sum of the weights of constituents in C, *i.e.*:

$$\alpha = \|\{arg_{shared}\}\| + w_{target}. \quad (2)$$

In our current implementation we reduce targets and their arguments to their syntactic headwords. These headwords are then directly compared using a corpus-based similarity thesaurus. As we hypothesized that targets are more important for predicate argument tuple similarity, we multiply the target's similarity by a weighting factor w_{target}, whose value we have empirically determined as 1.7, based on a 300-pair development set from the MSR training set.

We then proceed to pair tuples in the two sentences using a greedy iterative algorithm. The algorithm locates the two most similar tuples from each sentence, pairs them together and removes them from futher consideration. The process stops when subsequent best pairings are below the similarity threshold or when all possible tuples are exhausted. If unpaired tuples still exist in a given sentence pair, we further examine the copular constructions and noun phrases in the opposing sentence for possible pairings[2]. This results in a one-to-one mapping with possibly some tuples left unpaired. The curved arrows in Table 1 denote the correct results of similarity pairing: two tuples are paired up if their target and shared arguments are identical or similar respectively, otherwise they remain unpaired even if the "bag of words" they contain are the same.

[2]Copular constructions are not handled by ASSERT. Such constructions account for a large proportion of the semantic meaning in sentences. Consider the pair "Microsoft rose 50 cents" and "Microsoft was up 50 cents", in which the second is in copular form. Similarly, NPs can often be equivalent to predicate argument tuples when actions are nominalized. Consider an NP that reads "(be blamed for) frequent attacks on soldiers" and a predicate argument tuple: "(be blamed for) attacking soldiers". Again, identical information is conveyed but not captured by semantic role labeling. In such cases, they can be paired if we allow a candidate tuple to pair with the *predicative argument* (e.g., *50 cents*) of a copula, or (the head of) an NP in the opposing sentence. As these heuristic matches may introduce errors, we resort to these methods of matching tuple only in the contingency when there are unpaired tuples.

5 Dissimilarity Significance Classification

If some tuples remain unpaired, they are dissimilar parts of the sentence that need to be labeled by the dissimilarity classifier. Such unpaired information could be extraneous or they could be semantically important, creating a barrier for paraphrase. We frame this as a supervised machine learning problem in which a set of features are used to inform the classifier. A support vector machine, SVM^{Light}, was chosen as the learning model as it has shown to yield good performance over a wide application range. We experimented with a wide set of features of unpaired tuples, including internal counts of *numeric expressions* , *named entities*, *words*, *semantic roles*, whether they are similar to other tuples in the same sentence, and contextual features like *source/target sentence length* and *paired tuple count*. Currently, only two features are correlated in improved classification, which we detail now.

Syntactic Parse Tree Path: This is a series of features that reflect how the unpaired tuple connects with the context: the rest of the sentence. It models the syntactic connection between the constituents on both ends of the path (Gildea and Palmer, 2002; Pradhan et al., 2004). Here, we model the ends of the path as the unpaired tuple and the paired tuple with the closest shared ancestor, and model the path itself as a sequence of constituent category tags and directions to reach the destination (the paired target) from the source (the

unpaired target) via the the shared ancestor. When no tuples have been paired in the sentence pair, the destination defaults to the root of the syntactic parse tree. For example, the tuples with target "injured" are unpaired when the model sentence and the non-paraphrasing modification in Table 1 are being compared. A path "$\uparrow_{VBD}, \uparrow_{VP}, \uparrow_{S}, \uparrow_{SBAR}, \uparrow_{VP}, \downarrow_{VBD}$" links a target "injured" to the paired target "said", as shown in Figure 3.

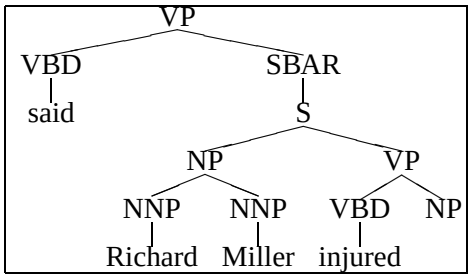

Figure 3: Syntactic parse tree path

The syntactic path can act as partial evidence in significance classification. In the above example, the category tag "$_{VBD}$" assigned to "injured" indicates that the verb is in its past tense. Such a predicate argument tuple bears the main content of the sentence and generally can not be ignored if its meaning is missing in the opposing sentence. Another example is shown in Figure 4. The second sentence has one unpaired target "proposed" while the rest all find their counterpart. The path we get from the syntactic parse tree reads "$[\uparrow_{VBN}, \uparrow_{NP}, \uparrow_{S}, ...]$", showing that the unpaired tuple (consisting of a single predicate) is a modifier contained in an NP. It can be ignored if there is no contradiction in the opposing sentence.

We represent a syntactic path by a set of n-gram ($n \leq 4$) features of subsequences of category tags found in the path, along with the respective direction. We require these n-gram features to be no more than four category tags away from the unpaired target, as our primary concern is to model what role the target plays in its sentence.

Sheena Young of Child, the national infertility support network, hoped the guidelines would lead to a more "fair and equitable" service for infertility sufferers.

Sheena Young, a spokesman for Child, the national infertility support network, said the **proposed** guidelines should lead to a more "fair and equitable" service for infertility sufferers.

Figure 4: Unpaired predicate argument tuple as modifier in a paraphrase

Predicate: This is the lexical token of predicate argument tuple's target, as a text feature. As this feature is liable to run into sparse data problems, the semantic category of the target would be a more suitable feature. However, verb similarity is generally regarded as difficult to measure, both in terms of semantic relatedness as well as in finding a consistent granularity for verb categories. We investigated using WordNet as well as Levin's classification (Levin, 1993) as additional features on our validation data, but currently find that using the lexical form of the target works best.

5.1 Classifier Training Set Acquisition

Currently, no training corpus for predicate argument tuple significance exists. Such a corpus is indispensable for training the classifier. Rather than manually annotating training instances, we use an automatic method to construct instances from paraphrase corpora. This is possible as the paraphrase judgments in the corpora can imply which portion of the sentence(s) are significant barriers to paraphrasing or not. Here, we exploit the similarity detector implemented for the first phase for this purpose. If unpaired tuples exist after greedy pairing, we classify them along two dimensions: whether the sentence pair is a (non-)paraphrase, and the source of the unpaired tuples:

1. **[PS] p**araphrasing pairs and unpaired predicate argument tuples are only from a **s**ingle sentence;
2. **[NS] n**on-paraphrasing pairs and only one **s**ingle unpaired predicate argument tuple exists;
3. **[PM] p**araphrasing pairs and unpaired predicate argument tuples are from **m**ultiple (both) sentences;
4. **[NM] n**on-paraphrasing pairs and **m**ultiple unpaired predicate argument tuples (from either one or both sentences) exist.

Assuming that similarity detector pairs tuples correctly, for the first two categories, the paraphrasing judgment is directly linked to the unpaired tuples. PS tuple instances are therefore used as $insignificant$ class instances, and NS as $significant$ ones. The last two categories cannot be used for training data, as it is unclear which of the unpaired tuples is responsible for the (non-) paraphrasing as the similarity measure may mistakenly leave some similar predicate argument tuples unpaired.

6 Evaluation

The goal of our evaluation is to show that our system can reliably determine the cause(s) of non-

MSR Corpus Label	+pp		-pp		
system prediction correct?	T	F	T	F	total
# sentence pairs (*s-ps*)	85	23	**55**	37	200
# *s-ps* with H-C-agreed labelings	80	19	53	35	187
# tuples paired (*t-ps*) (S)	80	6	36	35	**157**
# correct *t-ps* (H)	74	6	34	30	**144**
# missed *t-ps* (H)	11	10	5	5	**31**
# significant unpaired tuples(H)	6	4	69	51	130
# significant unpaired tuples(S)	0	32	70	0	102
# H-S-agreed sig. unpaired tuples	0	4	**43**	0	47
# -pp for other reasons	0	0	5	2	7

Table 2: (H)uman annotations vs. (C)orpus annotations and (S)ystem output

paraphrase examples, while maintaining the performance level of the state-of-the-art PR systems.

For evaluation, we conduct both component evaluations as well as a holistic one, resulting in three separate experiments. In evaluating the first tuple pairing component, we aim for high precision, so that sentences that have all tuples paired can be safely assumed to be paraphrases. In evaluating the dissimilarity classifier, we simply aim for high accuracy. In our overall system evaluation, we compare our system versus other PR systems on standard corpora.

Experimental Data Set. For these experiments, we utilized two widely-used corpora for paraphrasing evaluation: the MSR and PASCAL RTE corpora. The Microsoft Research Paraphrase coupus (Dolan et al., 2004) consists of 5801 newswire sentence pairs, 3900 of which are annotated as semantically equivalent by human annotators. It reflects ordinary paraphrases that people often encounter in news articles, and may be viewed as a typical domain-general paraphrase recognition task that downstream NLP systems will need to deal with. The corpus comes divided into standard training (70%) and testing (30%) divisions, a partition we follow in our experiments. ASSERT (the semantic parser) shows for this corpus a sentence contains 2.24 predicate argument tuples on average. The second corpus is the *paraphrase acquisition* subset of the PASCAL Recognizing Textual Entailment (RTE) Challenge corpus (Dagan et al., 2005). This is much smaller, consisting of 50 pairs, which we employ for testing only to show portability.

To assess the component performance, we need additional ground truth beyond the $\{+pp, -pp\}$ labels provided by the corpora. For the first evaluation, we need to ascertain whether a sentence pair's tuples are correctly paired, misidentified or mispaired. For the second, which tuple(s) (if any) are responsible for a $-pp$ instance. However, creating ground truth by manual annotation is expensive, and thus we only sampled the data set to get an indicative assessment of performance. We sampled 200 random instances from the total MSR testing set, and first processed them through our framework. Then, five human annotators (two authors and three volunteers) annotated the ground truth for tuple pairing and the semantic significance of the unpaired tuples, while checking system output. They also independently came up with their own $\{$*+pp,-pp*$\}$ judgment so we could assess the reliability of the provided annotations.

The results of this annotation is shown in Table 6. Examining this data, we can see that the similarity detector performs well, despite its simplicity and assumption of a one-to-one mapping. Out of the 157 predicate argument tuple pairs identified through similarity detection, annotators agreed that 144 (92%) are semantically similar or equivalent. However, 31 similar pairs were missed by the system, resulting in 82% recall. We defer discussion on the other details of this table to Section 7.

To assess the dissimilarity classifier, we focus on the $-pp$ subset of 55 instances recognized by the system. For 43 unpaired tuples from 40 sentence pairs (73% of 55), the annotators' judgments agree with the classifier's claim that they are significant. For these cases, the system is able to both recognize that the sentence pair is not a paraphrase and further correctly establish a cause of the non-paraphrase.

In addition to this ground truth sampled evaluation, we also show the effectiveness of the classifier by examining its performance on PS and NS tuples in the MSR corpus as described in Section 5. The test set consists of 413 randomly selected PS and NS instances among which 145 are significant (leading to non-paraphrases). The classifier predicts predicate argument tuple significance at an accuracy of 71%, outperforms a majority classifier (65%), a gain which is marginally statistically significant ($p < .09$).

significant	insignificant	
112	263	insignificant by classifier
33	5	**significant** by classifier

We can see the classifier classifies the majority of insignificant tuples correctly (by outputting a score greater than zero), which is effectively a

Algorithm	709 Sentence Pairs Without Unpaired Tuples (41.1% of Test set)			1016 Sentence Pairs With Unpaired Tuples (58.9% of Test set)			Overall (100% of Test set)			
	Acc	R	P	Acc	R	P	Acc	R	P	F1
Majority Classifier	79.5%	100%	79.5%	57.4%	100%	57.4%	66.5%	100%	66.5%	79.9%
SimFinder	82.2%	92.2%	86.4%	66.3%	84.9%	66.1%	**72.9%**	88.5%	**75.1%**	81.3%
CM05	-	-	-	-	-	-	71.5%	92.5%	72.3%	81.2%
Our System	79.5%	100%	79.5%	66.7%	87.0%	66.0%	72.0%	**93.4%**	72.5%	**81.6%**

Table 3: Results on MSR test set

Algorithm	17 Sentence Pairs Without Unpaired Tuples (34% of Test set)			33 Sentence Pairs With Unpaired Tuples (66% of Test set)			Overall (100% of Test set)			
	Acc	R	P	Acc	R	P	Acc	R	P	F1
Majority Classifier	65%	100%	65%	42%	100%	42%	50%	100%	50%	67%
SimFinder	71%	91%	71%	42%	21%	27%	52%	52%	52%	52%
Our System	65%	100%	65%	48%	64%	43%	54%	80%	53%	64%

Table 4: Results on PASCAL PP test set

98% recall of insignificant tuples. However, the precision is less satisfatory. We suspect this is partially due the tuples that fail to be paired up with their counterpart. Such noise is found among the automatically collected PS instances used in training.

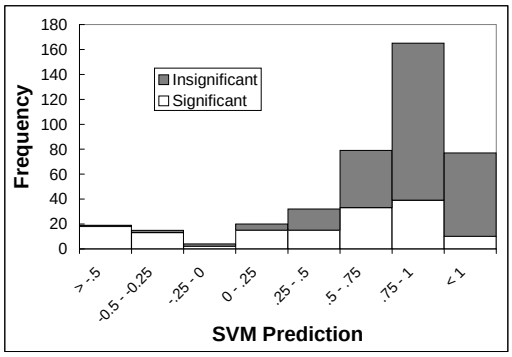

Figure 5: Dissimilarity classifier performance

For the final system-wide evaluation, we implemented two baseline systems: a majority classifier and SimFinder (Hatzivassiloglou et al., 2001), a bag-of-words sentence similarity module incorporating lexical, syntactic and semantic features. In Table 3, precision and recall are measured with respect to the paraphrasing class. The table shows sentence pairs falling under the column "pairs without unpaired tuples" are more likely to be paraphrasing than an arbitrary pair (79.5% versus 66.5%), providing further validation for using predicate argument tuples as information nuggets. The results for the experiment benchmarking the overall system performance are shown under the "Overall" column: our approach performs comparably to the baselines at both accuracy and paraphrase recall. The system performance reported in (CM05; (Corley and Mihalcea, 2005)), which is among the best we are aware of, is also included for comparison.

We also ran our system (trained on the MSR corpus) on the 50 instances in the PASCAL paraphrase acquisition subset. Again, the system performance (as shown in Table 4) is comparable to the baseline systems.

7 Discussion

We have just shown that when two sentences have perfectly matched predicate argument tuples, they are more likely to be a paraphrase than a random sentence pair drawn from the corpus. Furthermore, in the sampled human evaluation in Table 2, among the 88 non-paraphrasing instances with whose MSR corpus labels our annotators agreed (53 correctly and 35 incorrectly judged by our system), the cause of the $-pp$ is correctly attributed in 81 cases to one or more predicate argument tuples. The remaining 7 cases (as shown in the last row) are caused by phenomenon that are not captured by our tuple representation. We feel this justifies using predicate argument tuples as *information nuggets*, but we are currently considering expanding our representation to account for some of these caes.

The evaluation also confirms the difficulty of making paraphrase judgements. Although the MSR corpus used strict means of resolving inter-

rater disagreements during its construction, the annotators agreed with the MSR corpus labels only 93.5% (187/200) of the time.

One weakness of our system is that we rely on a thesaurus (Lin, 1998) for word similarity information for predicate argument tuple pairing. However, it is designed to provide similarity scores between pairs of individual words (rather than phrases). If a predicate argument tuple's target or one argument is realized as a phrase (*borrow* → *check out*, for instance), the thesaurus is unable to provide an accurate similarity score. For similarity between predicate argument tuples, negation and modality have yet to be addressed, although they account for only a tiny fraction of the corpus. potentially responsible for only ignorable amount of incorrect system judgements (we identified less than 10 cases of 1725 where negation matters).

We further estimated how the similarity detector can be affected when the semantic role labeler makes mistakes (by failing to identify arguments or assigning incorrect role names). Checking 94 pairs ground-truth similar tuples, we found that the system mislabels 43 of them. The similarity detector failed to pair around 30% of them. In comparsion, all the tuple pairs free of labeling errors are correctly paired. A saving grace is that labeling errors rarely lead to incorrect pairing (one pairing in all the examined sentences). The labeling errors impact the whole system in two ways: 1) they caused similar tuples that should have been paired up to be added as noise in that dissimilarity classifier's training set and 2) incorrectly paired tuples erroneously increase the target weight in Equation (1).

Some example paraphrasing cases that are problematic for our current system are:

1. Non-literal language issues such as implicature, idiom, metaphor, *etc.* are not addressed in our current system. When predicate argument tuples imply each other, they are less similar than what our system currently is trained for, causing the pairing to fail:

+pp, Later in the day, a standoff developed between French soldiers and a Hema battlewagon that attempted to pass the UN compound.

French soldiers later threatened to open fire on a Hema battlewagon that tried to pass near the UN compound.

2. A paraphrasing pair may exceed the systems' threshold for syntactic difference:

+pp, With the exception of dancing, physical activity did not decrease the risk.

Dancing was the only physical activity associated with a lower risk of dementia.

3. One or more unpaired tuples exist, but their significance is not inferred correctly:

+pp, Inhibited children tend to be timid with new people, objects, and situations, while uninhibited children spontaneously approach them.

Simply put, shy individuals tend to be more timid with new people and situations.

In the MSR corpus, the first error case is more frequent than the other two. We identify these as challenging cases where idiomatic processing is needed.

Below we show some unpaired predicate argument tuples (underlined) that are significant enough to be paraphrase barriers. These examples give an indicative categorization of what a significant tuples are and their corpus frequency (when predicate argument tuples are the reasons; we examined 40 such cases for this estimation). There is one thing in common: every case involves substantial information is difficult to infer from context. Such tuples appear as:

1. (40%) The nucleus of the sentence (often the matrix tuple):

Michael Hill, a Sun reporter who is a member of the Washington-Baltimore Newspaper Guild's bargaining committee, estimated meetings to last late Sunday.

2. (30%) A part of a coordination:

Security lights have also been installed and police have swept the grounds for booby traps.

3. (13%) A predicate of a modifying clause:

Westermayer was 26 then, and a friend and former manager who knew she was unhappy in her job tipped her to another position.

4. (7%) An adjunct:

While waiting for a bomb squad to arrive, the bomb exploded, killing Wells.

5. (7%) An embedded sentence:

Dean told reporters traveling on his 10-city "Sleepless Summer" tour that he considered campaigning in Texas a challenge.

6. (3%) Or factual content that conflicts with the opposing sentence:

Total sales for the period declined 8.0 percent to USD1.99 billion from a year earlier.

Wal-Mart said sales at stores open at least a year rose 4.6 percent from a year earlier.

8 Conclusions

We have proposed a new approach to the paraphrase recognition (PR) problem: a supervised, two-phase framework emphasizing dissimilarity classification. To emulate human PR judgment in which insignificant, extraneous *information nuggets* are generally allowed for a paraphrase, we estimate whether such additional information nuggets affect the final paraphrasing status of a sentence pair. This approach, unlike previous PR approaches, has the key benefit of explaining the cause of a non-paraphrase sentence pair.

In the first, similarity detection module, using predicate argument tuples as the unit for comparison, we pair them up in a greedy manner. Unpaired tuples thus represent additional information unrepresented in the opposing sentence. A second, dissimilarity classification module uses the lexical head of the predicates and the tuples' path of attachment as features to decide whether such tuples are barriers to paraphrase.

Our evaluations show that the system 1) obtains high accuracy for the similarity detector in pairing predicate argument tuples, 2) robust dissimilarity classification despite noisy training instances and 3) comparable overall performance to current state-of-the-art PR systems. To our knowledge this is the first work that tackles the problem of identifying what factors stop a sentence pair from being a paraphrase.

We also presented corpus examples that illustrate the categories of errors that our framework makes, suggesting future work in PR. While we continue to explore more suitable representation of unpaired predicate argument tuples, we plan to augment the similarity measure for phrasal units to reduce the error rate in the first component. Another direction is to detect semantic redundancy in a sentence. Unpaired tuples that are semantically redundant should also be regarded as insignificant.

References

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

Chris Brockett and Bill Dolan. 2005. Support vector machines for paraphrase identification and corpus construction. In *Proceedings of the 3rd International Workshop on Paraphrasing*.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL'2000)*.

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*, pages 13–18, Ann Arbor, USA.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *PASCAL Proceedings of the First Challenge Workshop–Recognizing Textual Entailment*, Southampton,UK.

Bill Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, Switzerland.

Daniel Gildea and Martha Palmer. 2002. The necessity of parsing for predicate argument recognition. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, USA.

Vassileios Hatzivassiloglou, Judith Klavans, Melissa Holcombe, Regina Barzilay, Min-Yen Kan, and Kathleen McKeown. 2001. Simfinder: A flexible clustering tool for summarization. In *Proceedings of the NAACL Workshop on Automatic Summarization*, pages 41–49.

Paul Kingsbury, Martha Palmer, and Mitch Marcus. 2002. Adding semantic annotation to the penn treebank. In *Proceedings of the Human Language Technology Conference*, San Diego, USA.

Beth Levin. 1993. English verb classes and alternations: A preliminary investigation. University of Chicago Press.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL '98*, pages 768–774, Montreal, Canada.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Dan Jurafsky. 2004. Shallow semantic parsing using support vector machines. In *Proceedings of HLT/NAACL*, Boston, USA.

Yusuke Shinyama, Satoshi Sekine, Kiyoshi Sudo, and Ralph Grishman. 2002. Automatic paraphrase acquisition from news articles. In *Proceedings of the Human Language Technology Conference*, pages 40–46, San Diego, USA.

Julie Weeds, David Weir, and Bill Keller. 2005. The distributional similarity of sub-parses. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, pages 7–12, Ann Arbor, USA.

Dekai Wu. 2005. Recognizing paraphrases and textual entailment using inversion transduction grammars. In *Proceedings of the ACL Workshop on Empirical Modeling of Semantic Equivalence and Entailment*, Ann Arbor, USA.