Large Scale Acquisition of Paraphrases for Learning Surface Patterns

Rahul Bhagat[*]
Information Sciences Institute
University of Southern California
Marina del Rey, CA
rahul@isi.edu

Deepak Ravichandran
Google Inc.
1600 Amphitheatre Parkway
Mountain View, CA
deepakr@google.com

Abstract

Paraphrases have proved to be useful in many applications, including Machine Translation, Question Answering, Summarization, and Information Retrieval. Paraphrase acquisition methods that use a single monolingual corpus often produce only syntactic paraphrases. We present a method for obtaining surface paraphrases, using a 150GB (25 billion words) monolingual corpus. Our method achieves an accuracy of around 70% on the paraphrase acquisition task. We further show that we can use these paraphrases to generate surface patterns for relation extraction. Our patterns are much more precise than those obtained by using a state of the art baseline and can extract relations with more than 80% precision for each of the test relations.

1 Introduction

Paraphrases are textual expressions that convey the same meaning using different surface words. For example consider the following sentences:

Google *acquired* YouTube. (1)

Google *completed the acquisition of* YouTube. (2)

Since they convey the same meaning, sentences (1) and (2) are sentence level paraphrases, and the phrases "*acquired*" and "*completed the acquisition of*" in (1) and (2) respectively are phrasal paraphrases.

Paraphrases provide a way to capture the variability of language and hence play an important role in many natural language processing (NLP) applications. For example, in question answering, paraphrases have been used to find multiple patterns that pinpoint the same answer (Ravichandran and Hovy, 2002); in statistical machine translation, they have been used to find translations for unseen source language phrases (Callison-Burch et al., 2006); in multi-document summarization, they have been used to identify phrases from different sentences that express the same information (Barzilay et al., 1999); in information retrieval they have been used for query expansion (Anick and Tipirneni, 1999).

Learning paraphrases requires one to ensure identity of meaning. Since there are no adequate semantic interpretation systems available today, paraphrase acquisition techniques use some other mechanism as a kind of "pivot" to (help) ensure semantic identity. Each pivot mechanism selects phrases with similar meaning in a different characteristic way. A popular method, the so-called distributional similarity, is based on the dictum of Zelig Harris "you shall know the words by the company they keep": given highly discriminating left and right contexts, only words with very similar meaning will be found to fit in between them. For paraphrasing, this has been often used to find syntactic transformations in parse trees that preserve (semantic) meaning. Another method is to use a bilingual dictionary or translation table as pivot mechanism: all source language words or phrases that translate to a given foreign word/phrase are deemed to be paraphrases of one another. In this paper we call the paraphrases that contain only words as surface paraphrases and those

[*]Work done during an internship at Google Inc.

that contain paths in a syntax tree as syntactic paraphrases.

We here, present a method to acquire surface paraphrases from a single monolingual corpus. We use a large corpus (about 150GB) to overcome the data sparseness problem. To overcome the scalability problem, we pre-process the text with a simple parts-of-speech (POS) tagger and then apply locality sensitive hashing (LSH) (Charikar, 2002; Ravichandran et al., 2005) to speed up the remaining computation for paraphrase acquisition. Our experiments show results to verify the following main claim:

Claim 1: *Highly precise surface paraphrases can be obtained from a very large monolingual corpus.*

With this result, we further show that these paraphrases can be used to obtain high precision surface patterns that enable the discovery of relations in a minimally supervised way. Surface patterns are templates for extracting information from text. For example, if one wanted to extract a list of company acquisitions, "⟨ACQUIRER⟩ *acquired* ⟨ACQUIREE⟩" would be one surface pattern with "⟨ACQUIRER⟩" and "⟨ACQUIREE⟩" as the slots to be extracted. Thus we can claim:

Claim 2: *These paraphrases can then be used for generating high precision surface patterns for relation extraction.*

2 Related Work

Most recent work in paraphrase acquisition is based on automatic acquisition. Barzilay and McKeown (2001) used a monolingual parallel corpus to obtain paraphrases. Bannard and Callison-Burch (2005) and Zhou et al. (2006) both employed a bilingual parallel corpus in which each foreign language word or phrase was a pivot to obtain source language paraphrases. Dolan et al. (2004) and Barzilay and Lee (2003) used comparable news articles to obtain sentence level paraphrases. All these approaches rely on the presence of parallel or comparable corpora and are thus limited by their availability and size.

Lin and Pantel (2001) and Szpektor et al. (2004) proposed methods to obtain entailment templates by using a single monolingual resource. While both differ in their approaches, they both end up finding syntactic paraphrases. Their methods cannot be used if we cannot parse the data (either because of scale or data quality). Our approach on the other hand, finds surface paraphrases; it is more scalable and robust due to the use of simple POS tagging. Also, our use of locality sensitive hashing makes finding similar phrases in a large corpus feasible.

Another task related to our work is relation extraction. Its aim is to extract instances of a given relation. Hearst (1992) the pioneering paper in the field used a small number of hand selected patterns to extract instances of hyponymy relation. Berland and Charniak (1999) used a similar method for extracting instances of meronymy relation. Ravichandran and Hovy (2002) used seed instances of a relation to automatically obtain surface patterns by querying the web. But their method often finds patterns that are too general (e.g., X *and* Y), resulting in low precision extractions. Rosenfeld and Feldman (2006) present a somewhat similar web based method that uses a combination of seed instances and seed patterns to learn good quality surface patterns. Both these methods differ from ours in that they learn relation patterns on the fly (from the web). Our method however, pre-computes paraphrases for a large set of surface patterns using distributional similarity over a large corpus and then obtains patterns for a relation by simply finding paraphrases (offline) for a few seed patterns. Using distributional similarity avoids the problem of obtaining overly general patterns and the pre-computation of paraphrases means that we can obtain the set of patterns for any relation instantaneously.

Romano et al. (2006) and Sekine (2006) used syntactic paraphrases to obtain patterns for extracting relations. While procedurally different, both methods depend heavily on the performance of the syntax parser and require complex syntax tree matching to extract the relation instances. Our method on the other hand acquires surface patterns and thus avoids the dependence on a parser and syntactic matching. This also makes the extraction process scalable.

3 Acquiring Paraphrases

This section describes our model for acquiring paraphrases from text.

3.1 Distributional Similarity

Harris's distributional hypothesis (Harris, 1954) has played an important role in lexical semantics. It states that words that appear in similar contexts tend to have similar meanings. In this paper, we apply the distributional hypothesis to phrases i.e. word n-grams.

For example, consider the phrase "*acquired*" of the form "X *acquired* Y". Considering the context of this phrase, we might find {Google, eBay, Yahoo,...} in position X and {YouTube, Skype, Overture,...} in position Y. Now consider another phrase "*completed the acquisition of*", again of the form "X *completed the acquisition of* Y". For this phrase, we might find {Google, eBay, Hilton Hotel corp.,...} in position X and {YouTube, Skype, Bally Entertainment Corp.,...} in position Y. Since the contexts of the two phrases are similar, our extension of the distributional hypothesis would assume that "*acquired*" and "*completed the acquisition of*" have similar meanings.

3.2 Paraphrase Learning Model

Let p be a phrase (n-gram) of the form $X\ p\ Y$, where X and Y are the placeholders for words occurring on either side of p. Our first task is to find the set of phrases that are similar in meaning to p. Let $P = \{p_1, p_2, p_3, ..., p_l\}$ be the set of all phrases of the form $X\ p_i\ Y$ where $p_i \in P$. Let $S_{i,X}$ be the set of words that occur in position X of p_i and $S_{i,Y}$ be the set of words that occur in position Y of p_i. Let V_i be the vector representing p_i such that $V_i = S_{i,X} \cup S_{i,Y}$. Each word $f \in V_i$ has an associated score that measures the strength of the association of the word f with phrase p_i; as do many others, we employ pointwise mutual information (Cover and Thomas, 1991) to measure this strength of association.

$$pmi(p_i; f) = \log \frac{P(p_i,f)}{P(p_i)P(f)} \tag{1}$$

The probabilities in equation (1) are calculated by using the maximum likelihood estimate over our corpus.

Once we have the vectors for each phrase $p_i \in P$, we can find the paraphrases for each p_i by finding its nearest neighbors. We use cosine similarity, which is a commonly used measure for finding similarity between two vectors.

If we have two phrases $p_i \in P$ and $p_j \in P$ with the corresponding vectors V_i and V_j constructed as described above, the similarity between the two phrases is calculated as:

$$sim(p_i; p_j) = \frac{V_i . V_j}{|V_i| * |V_j|} \tag{2}$$

Each word in V_i (and V_j) has with it an associated flag which indicates weather the word came from $S_{i,X}$ or $S_{i,Y}$. Hence for each phrase p_i of the form $X\ p_i\ Y$, we have a corresponding phrase $-p_i$ that has the form $Y\ p_i\ X$. This is important to find certain kinds of paraphrases. The following example will illustrate. Consider the sentences:

Google *acquired* YouTube. (3)

YouTube *was bought by* Google. (4)

From sentence (3), we obtain two phrases:

1. p_i = *acquired* which has the form "X *acquired* Y" where "X = Google" and "Y = YouTube"
2. $-p_i$ = $-$*acquired* which has the form "Y *acquired* X" where "X = YouTube" and "Y = Google"

Similarly, from sentence (4) we obtain two phrases:

1. p_j = *was bought by* which has the form "X *was bought by* Y" where "X = YouTube" and "Y = Google"
2. $-p_j$ = $-$*was bought by* which has the form "Y *was bought by* X" where "X = Google" and "Y = YouTube"

The switching of X and Y positions in (3) and (4) ensures that "*acquired*" and "$-$*was bought by*" are found to be paraphrases by the algorithm.

3.3 Locality Sensitive Hashing

As described in Section 3.2, we find paraphrases of a phrase p_i by finding its nearest neighbors based on cosine similarity between the feature vector of p_i and other phrases. To do this for all the phrases in the corpus, we'll have to compute the similarity between all vector pairs. If n is the number of vectors and d is the dimensionality of the vector space, finding cosine similarity between each pair of vectors has time complexity $O(n^2d)$. This computation is infeasible for our corpus, since both n and d are large.

To solve this problem, we make use of Locality Sensitive Hashing (LSH). The basic idea behind LSH is that a LSH function creates a fingerprint for each vector such that if two vectors are similar, they are likely to have similar fingerprints. The LSH function we use here was proposed by Charikar (2002). It represents a d dimensional vector by a stream of b bits ($b \ll d$) and has the property of preserving the cosine similarity between vectors, which is exactly what we want. Ravichandran et al. (2005) have shown that by using the LSH nearest neighbors calculation can be done in $O(nd)$ time.[1].

4 Learning Surface Patterns

Let r be a target relation. Our task is to find a set of surface patterns $S = \{s_1, s_2, ..., s_n\}$ that express the target relation. For example, consider the relation r = "*acquisition*". We want to find the set of patterns S that express this relation:
S = $\{\langle$ACQUIRER$\rangle$ *acquired* $\langle$ACQUIREE$\rangle$, $\langle$ACQUIRER$\rangle$ *bought* $\langle$ACQUIREE$\rangle$, $\langle$ACQUIREE$\rangle$ *was bought by* $\langle$ACQUIRER$\rangle$,...$\}$.

The remainder of the section describes our model for learning surface patterns for target relations.

4.1 Model Assumption

Paraphrases express the same meaning using different surface forms. So if one knew a pattern that expresses a target relation, one could build more patterns for that relation by finding paraphrases for the surface phrase(s) in that pattern. This is the basic assumption of our model.

For example, consider the seed pattern "$\langle$ACQUIRER$\rangle$ *acquired* $\langle$ACQUIREE$\rangle$" for the target relation "*acquisition*". The surface phrase in the seed pattern is "*acquired*". Our model then assumes that we can obtain more surface patterns for "*acquisition*" by replacing "*acquired*" in the seed pattern with its paraphrases i.e. $\{$*bought*, $-$*was bought by*[2],...$\}$. The resulting surface patterns are:
$\{\langle$ACQUIRER$\rangle$ *bought* $\langle$ACQUIREE$\rangle$, $\langle$ACQUIREE$\rangle$ *was bought by* $\langle$ACQUIRER$\rangle$,...$\}$

4.2 Surface Pattern Model

Let r be a target relation. Let $SEED = \{seed_1, seed_2, ..., seed_n\}$ be the set of seed patterns that express the target relation. For each $seed_i \in SEED$, we obtain the corresponding set of new patterns PAT_i in two steps:

1. We find the surface phrase, p_i, using a seed and find the corresponding set of paraphrases, $P_i = \{p_{i,1}, p_{i,2}, ..., p_{i,m}\}$. Each paraphrase, $p_{i,j} \in P_i$, has with it an associated score which is similarity between p_i and $p_{i,j}$.

2. In seed pattern, $seed_i$, we replace the surface phrase, p_i, with its paraphrases and obtain the set of new patterns $PAT_i = \{pat_{i,1}, pat_{i,2}, ..., pat_{i,m}\}$. Each pattern has with it an associated score, which is the same as the score of the paraphrase from which it was obtained[3] . The patterns are ranked in the decreasing order of their scores.

After we obtain PAT_i for each $seed_i \in SEED$, we obtain the complete set of patterns, PAT, for the target relation r as the union of all the individual pattern sets, i.e., $PAT = PAT_1 \cup PAT_2 \cup ... \cup PAT_n$.

5 Experimental Methodology

In this section, we describe experiments to validate the main claims of the paper. We first describe paraphrase acquisition, we then summarize our method for learning surface patterns, and finally describe the use of patterns for extracting relation instances.

5.1 Paraphrases

Finding surface variations in text requires a large corpus. The corpus needs to be orders of magnitude larger than that required for learning syntactic variations, since surface phrases are sparser than syntactic phrases.

For our experiments, we used a corpus of about 150GB (25 billion words) obtained from Google News[4] . It consists of few years worth of news data.

[1]The details of the algorithm are omitted, but interested readers are encouraged to read Charikar (2002) and Ravichandran et al. (2005)

[2]The "$-$" in "$-$*was bought by*" indicates that the $\langle$ACQUIRER$\rangle$ and $\langle$ACQUIREE$\rangle$ arguments of the input phrase "*acquired*" need to be switched for the phrase "*was bought by*".

[3]If a pattern is generated from more than one seed, we assign it its average score.

[4]The corpus was cleaned to remove duplicate articles.

We POS tagged the corpus using Tnt tagger (Brants, 2000) and collected all phrases (n-grams) in the corpus that contained at least one verb, and had a noun or a noun-noun compound on either side. We restricted the phrase length to at most five words.

We build a vector for each phrase as described in Section 3. To mitigate the problem of sparseness and co-reference to a certain extent, whenever we have a noun-noun compound in the X or Y positions, we treat it as bag of words. For example, in the sentence "Google Inc. *acquired* YouTube", "Google" and "Inc." will be treated as separate features in the vector[5].

Once we have constructed all the vectors, we find the paraphrases for every phrase by finding its nearest neighbors as described in Section 3. For our experiments, we set the number of random bits in the LSH function to 3000, and the similarity cut-off between vectors to 0.15. We eventually end up with a resource containing over 2.5 million phrases such that each phrase is connected to its paraphrases.

5.2 Surface Patterns

One claim of this paper is that we can find good surface patterns for a target relation by starting with a seed pattern. To verify this, we study two target relations[6]:

1. ***Acquisition:*** *We define this as the relation between two companies such that one company acquired the other.*

2. ***Birthplace:*** *We define this as the relation between a person and his/her birthplace.*

For "*acquisition*" relation, we start with the surface patterns containing only the words *buy* and *acquire*:

1. "⟨ACQUIRER⟩ *bought* ⟨ACQUIREE⟩" (and its variants, i.e. *buy*, *buys* and *buying*)

2. "⟨ACQUIRER⟩ *acquired* ⟨ACQUIREE⟩" (and its variants, i.e. *acquire*, *acquires* and *acquiring*)

This results in a total of eight seed patterns.

For "*birthplace*" relation, we start with two seed patterns:

1. "⟨PERSON⟩ *was born in* ⟨LOCATION⟩"

2. "⟨PERSON⟩ *was born at* ⟨LOCATION⟩".

We find other surface patterns for each of these relations by replacing the surface words in the seed patterns by their paraphrases, as described in Section 4.

5.3 Relation Extraction

The purpose of learning surface patterns for a relation is to extract instances of that relation. We use the surface patterns obtained for the relations "*acquisition*" and "*birthplace*" to extract instances of these relations from the LDC North American News Corpus. This helps us to extrinsically evaluate the quality of the surface patterns.

6 Experimental Results

In this section, we present the results of the experiments and analyze them.

6.1 Baselines

It is hard to construct a baseline for comparing the quality of paraphrases, as there isn't much work in extracting surface level paraphrases using a monolingual corpus. To overcome this, we show the effect of reduction in corpus size on the quality of paraphrases, and compare the results informally to the other methods that produce syntactic paraphrases.

To compare the quality of the extraction patterns, and relation instances, we use the method presented by Ravichandran and Hovy (2002) as the baseline. For each of the given relations, "*acquisition*" and "*birthplace*", we use 10 seed instances, download the top 1000 results from the Google search engine for each instance, extract the sentences that contain the instances, and learn the set of baseline patterns for each relation. We then apply these patterns to the test corpus and extract the corresponding baseline instances.

6.2 Evaluation Criteria

Here we present the evaluation criteria we used to evaluate the performance on the different tasks.

[5]This adds some noise in the vectors, but we found that this results in better paraphrases.

[6]Since we have to do all the annotations for evaluations on our own, we restricted our experiments to only two commonly used relations.

Paraphrases

We estimate the quality of paraphrases by annotating a random sample as correct/incorrect and calculating the accuracy. However, estimating the recall is difficult given that we do not have a complete set of paraphrases for the input phrases. Following Szpektor et al. (2004), instead of measuring recall, we calculate the average number of correct paraphrases per input phrase.

Surface Patterns

We can calculate the precision (P) of learned patterns for each relation by annotating the extracted patterns as correct/incorrect. However calculating the recall is a problem for the same reason as above. But we can calculate the relative recall (RR) of the system against the baseline and vice versa. The relative recall $RR_{S|B}$ of system S with respect to system B can be calculated as:

$$RR_{S|B} = \frac{C_S \cap C_B}{C_B}$$

where C_S is the number of correct patterns found by our system and C_B is the number of correct patterns found by the baseline. $RR_{B|S}$ can be found in a similar way.

Relation Extraction

We estimate the precision (P) of the extracted instances by annotating a random sample of instances as correct/incorrect. While calculating the true recall here is not possible, even calculating the true relative recall of the system against the baseline is not possible as we can annotate only a small sample. However, following Pantel et al. (2004), we assume that the recall of the baseline is 1 and estimate the relative recall $RR_{S|B}$ of the system S with respect to the baseline B using their respective precision scores P_S and P_B and number of instances extracted by them $|S|$ and $|B|$ as:

$$RR_{S|B} = \frac{P_S * |S|}{P_B * |B|}$$

6.3 Gold Standard

In this section, we describe the creation of gold standard for the different tasks.

Paraphrases

We created the gold standard paraphrase test set by randomly selecting 50 phrases and their corresponding paraphrases from our collection of 2.5 million phrases. For each test phrase, we asked two annotators to annotate its paraphrases as correct/incorrect. The annotators were instructed to look for strict paraphrases i.e. equivalent phrases that can be substituted for each other.

To obtain the inter-annotator agreement, the two annotators annotated the test set separately. The kappa statistic (Siegal and Castellan Jr., 1988) was $\kappa = 0.63$. The interesting thing is that the annotators got this respectable kappa score without any prior training, which is hard to achieve when one annotates for a similar task like textual entailment.

Surface Patterns

For the target relations, we asked two annotators to annotate the patterns for each relation as either "precise" or "vague". The annotators annotated the system as well as the baseline outputs. We consider the "precise" patterns as correct and the "vague" as incorrect. The intuition is that applying the vague patterns for extracting target relation instances might find some good instances, but will also find many bad ones. For example, consider the following two patterns for the "*acquisition*" relation:

⟨ACQUIRER⟩ *acquired* ⟨ACQUIREE⟩ (5)

⟨ACQUIRER⟩ *and* ⟨ACQUIREE⟩ (6)

Example (5) is a precise pattern as it clearly identifies the "*acquisition*" relation while example (6) is a vague pattern because it is too general and says nothing about the "*acquisition*" relation. The kappa statistic between the two annotators for this task was $\kappa = 0.72$.

Relation Extraction

We randomly sampled 50 instances of the "*acquisition*" and "*birthplace*" relations from the system and the baseline outputs. We asked two annotators to annotate the instances as correct/incorrect. The annotators marked an instance as correct only if both the entities and the relation between them were correct. To make their task easier, the annotators were provided the context for each instance, and were free to use any resources at their disposal (including a web search engine), to verify the correctness of the instances. The annotators found that the annotation for this task was much easier than the previous two; the few disagreements they had were due to ambiguity of some of the instances. The kappa statistic for this task was $\kappa = 0.91$.

Annotator	Accuracy	Average # correct paraphrases
Annotator 1	67.31%	4.2
Annotator 2	74.27%	4.28

Table 1: **Quality of paraphrases**

are being distributed to	approved a revision to the
have been distributed to	unanimously approved a new
are being handed out to	approved an annual
were distributed to	will consider adopting a
−are handing out	approved a revised
will be distributed to all	approved a new

Table 2: **Example paraphrases**

6.4 Result Summary

Table 1 shows the results of annotating the paraphrases test set. We do not have a baseline to compare against but we can analyze them in light of numbers reported previously for syntactic paraphrases. DIRT (Lin and Pantel, 2001) and TEASE (Szpektor et al., 2004) report accuracies of 50.1% and 44.3% respectively compared to our average accuracy across two annotators of 70.79%. The average number of paraphrases per phrase is however 10.1 and 5.5 for DIRT and TEASE respectively compared to our 4.2. One reason why this number is lower is that our test set contains completely random phrases from our set (2.5 million phrases): some of these phrases are rare and have very few paraphrases. Table 2 shows some paraphrases generated by our system for the phrases "*are being distributed to*" and "*approved a revision to the*".

Table 3 shows the results on the quality of surface patterns for the two relations. It can be observed that our method outperforms the baseline by a wide margin in both precision and relative recall. Table 4 shows some example patterns learned by our system.

Table 5 shows the results of the quality of extracted instances. Our system obtains very high precision scores but suffers in relative recall given that the baseline with its very general patterns is likely to find a huge number of instances (though a very small portion of them are correct). Table 6 shows some example instances we extracted.

acquisition	birthplace
X *agreed to buy* Y	X *, who was born in* Y
X *, which acquired* Y	X *, was born in* Y
X *completed its acquisition of* Y	X *was raised in* Y
X *has acquired* Y	X *was born in NNNN*[a] *in* Y
X *purchased* Y	X *, born in* Y

[a] Each "N" here is a placeholder for a number from 0 to 9.

Table 4: **Example extraction templates**

acquisition	birthplace
1. *Huntington Bancshares Inc.* agreed to acquire *Reliance Bank*	1. *Cyril Andrew Ponnamperuma* was born in *Galle*
2. *Sony* bought *Columbia Pictures*	2. *Cook* was born in NNNN in *Devonshire*
3. *Hanson Industries* buys *Kidde Inc.*	3. *Tansey* was born in *Cincinnati*
4. *Casino America inc.* agreed to buy *Grand Palais*	4. *Tsoi* was born in NNNN in *Uzbekistan*
5. *Tidewater inc.* acquired *Hornbeck Offshore Services Inc.*	5. *Mrs. Totenberg* was born in *San Francisco*

Table 6: **Example instances**

6.5 Discussion and Error Analysis

We studied the effect of the decrease in size of the available raw corpus on the quality of the acquired paraphrases. We used about 10% of our original corpus to learn the surface paraphrases and evaluated them. The precision, and the average number of correct paraphrases are calculated on the same test set, as described in Section 6.2. The performance drop on using 10% of the original corpus is significant (11.41% precision and on an average 1 correct paraphrase per phrase), which shows that we indeed need a large amount of data to learn good quality surface paraphrases. One reason for this drop is also that when we use only 10% of the original data, for some of the phrases from the test set, we do not find any paraphrases (thus resulting in 0% accuracy for them). This is not unexpected, as the larger resource would have a much larger recall, which again points at the advantage of using a large data set. Another reason for this performance drop could be the parameter settings: We found that the quality of learned paraphrases depended greatly on the various cut-offs used. While we adjusted our model

Relation	Method	# Patterns	**Annotator** 1		**Annotator** 2	
			P	RR	P	RR
Acquisition	Baseline	160	55%	13.02%	60%	11.16%
	Paraphrase Method	231	**83.11%**	**28.40%**	**93.07%**	**25%**
Birthplace	Baseline	16	31.35%	15.38%	31.25%	15.38%
	Paraphrase Method	16	**81.25%**	**40%**	**81.25%**	**40%**

Table 3: **Quality of Extraction Patterns**

Relation	Method	# Patterns	**Annotator** 1		**Annotator** 2	
			P	RR	P	RR
Acquisition	Baseline	$1,261,986$	6%	**100%**	2%	**100%**
	Paraphrase Method	3875	**88%**	4.5%	**82%**	12.59%
Birthplace	Baseline	$979,607$	4%	**100%**	2%	**100%**
	Paraphrase Method	1811	**98%**	4.53%	**98%**	9.06%

Table 5: **Quality of instances**

parameters for working with smaller sized data, it is conceivable that we did not find the ideal setting for them. So we consider these numbers to be a lower bound. But even then, these numbers clearly indicate the advantage of using more data.

We also manually inspected our paraphrases. We found that the problem of "antonyms" was somewhat less pronounced due to our use of a large corpus, but they still were the major source of error. For example, our system finds the phrase "*sell*" as a paraphrase for "*buy*". We need to deal with this problem separately in the future (may be as a post-processing step using a list of antonyms).

Moving to the task of relation extraction, we see from table 5 that our system has a much lower relative recall compared to the baseline. This was expected as the baseline method learns some very general patterns, which are likely to extract some good instances, even though they result in a huge hit to its precision. However, our system was able to obtain this performance using very few seeds. So an increase in the number of input seeds, is likely to increase the relative recall of the resource. The question however remains as to what good seeds might be. It is clear that it is much harder to come up with good seed patterns (that our system needs), than seed instances (that the baseline needs). But there are some obvious ways to overcome this problem. One way is to bootstrap. We can look at the paraphrases of the seed patterns and use them to obtain more patterns. Our initial experiments with this method using handpicked seeds showed good promise. However, we need to investigate automating this approach. Another method is to use the good patterns from the baseline system and use them as seeds for our system. We plan to investigate this approach as well. One reason, why we have seen good preliminary results using these approaches (for improving recall), we believe, is that the precision of the paraphrases is good. So either a seed doesn't produce any new patterns or it produces good patterns, thus keeping the precision of the system high while increasing relative recall.

7 Conclusion

Paraphrases are an important technique to handle variations in language. Given their utility in many NLP tasks, it is desirable that we come up with methods that produce good quality paraphrases. We believe that the paraphrase acquisition method presented here is a step towards this very goal. We have shown that high precision surface paraphrases can be obtained by using distributional similarity on a large corpus. We made use of some recent advances in theoretical computer science to make this task scalable. We have also shown that these paraphrases can be used to obtain high precision extraction patterns for information extraction. While we believe that more work needs to be done to improve the system recall (some of which we are investigating), this seems to be a good first step towards developing a minimally supervised, easy to implement, and scalable relation extraction system.

References

P. G. Anick and S. Tipirneni. 1999. The paraphrase search assistant: terminological feedback for iterative information seeking. In *ACM SIGIR*, pages 153–159.

C. Bannard and C. Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Association for Computational Linguistics*, pages 597–604.

R. Barzilay and L. Lee. 2003. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *In Proceedings North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 16–23.

R. Barzilay and K. R. McKeown. 2001. Extracting paraphrases from a parallel corpus. In *In Proceedings of Association for Computational Linguistics*, pages 50–57.

R. Barzilay, K. R. McKeown, and M. Elhadad. 1999. Information fusion in the context of multi-document summarization. In *Association for Computational Linguistics*, pages 550–557.

M. Berland and E. Charniak. 1999. Finding parts in very large corpora. In *In Proceedings of Association for Computational Linguistics*, pages 57–64.

T. Brants. 2000. Tnt – a statistical part-of-speech tagger. In *In Proceedings of the Applied NLP Conference (ANLP)*.

C. Callison-Burch, P. Koehn, and M. Osborne. 2006. Improved statistical machine translation using paraphrases. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 17–24.

M. S. Charikar. 2002. Similarity estimation techniques from rounding algorithms. In *In Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388.

T.M. Cover and J.A. Thomas. 1991. *Elements of Information Theory*. John Wiley & Sons.

B. Dolan, C. Quirk, and C. Brockett. 2004. Unsupervised construction of large paraphrase corpora: exploiting massively parallel news sources. In *In Proceedings of the conference on Computational Linguistics (COLING)*, pages 350–357.

Z. Harris. 1954. Distributional structure. *Word*, pages 10(23):146–162.

M. A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the conference on Computational linguistics*, pages 539–545.

D. Lin and P. Pantel. 2001. Dirt: Discovery of inference rules from text. In *ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 323–328.

P. Pantel, D. Ravichandran, and E.H. Hovy. 2004. Towards terascale knowledge acquisition. In *In Proceedings of the conference on Computational Linguistics (COLING)*, pages 771–778.

D. Ravichandran and E.H. Hovy. 2002. Learning surface text for a question answering system. In *Association for Computational Linguistics (ACL)*, Philadelphia, PA.

D. Ravichandran, P. Pantel, and E.H. Hovy. 2005. Randomized algorithms and nlp: using locality sensitive hash function for high speed noun clustering. In *In Proceedings of Association for Computational Linguistics*, pages 622–629.

L. Romano, M. Kouylekov, I. Szpektor, I. Dagan, and A. Lavelli. 2006. Investigating a generic paraphrase-based approach for relation extraction. In *In Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*.

B. Rosenfeld and R. Feldman. 2006. Ures: an unsupervised web relation extraction system. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 667–674.

S. Sekine. 2006. On-demand information extraction. In *In Proceedings of COLING/ACL*, pages 731–738.

S. Siegal and N.J. Castellan Jr. 1988. *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill.

I. Szpektor, H. Tanev, I. Dagan, and B. Coppola. 2004. Scaling web-based acquisition of entailment relations. In *In Proceedings of Empirical Methods in Natural Language Processing*, pages 41–48.

L. Zhou, C.Y. Lin, D. Munteanu, and E.H. Hovy. 2006. Paraeval: using paraphrases to evaluate summaries automatically. In *In Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 447–454.