A Statistical Model for Near-Synonym Choice

DIANA INKPEN
University of Ottawa

We present an unsupervised statistical method for automatic choice of near-synonyms when the context is given. The method uses the Web as a corpus to compute scores based on mutual information. Our evaluation experiments show that this method performs better than two previous methods on the same task. We also describe experiments in using supervised learning for this task. We present an application to an intelligent thesaurus. This work is also useful in machine translation and natural language generation.

Categories and Subject Descriptors: I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*; I.2.6 [**Artificial Intelligence**]: Learning—*Induction, Knowledge acquisition*

General Terms: Algorithms, Languages, Performance

Additional Key Words and Phrases: Lexical choice, near-synonyms, semantic similarity, Web as a corpus, intelligent thesaurus

ACM Reference Format:

Inkpen, D. 2007. A statistical model for near-synonym choice. ACM Trans. Speech Lang. Process. 4, 1, Article 2 (January 2007), 17 pages. DOI = 10.1145/1187415.1187417 http://doi.acm.org/10.1145/1187415.1187417.

1. INTRODUCTION

When writing a text, a poorly chosen word can convey unwanted connotations, implications, or attitudes. Similarly, in machine translation and natural language generation systems, the choice among near-synonyms needs to be made carefully. By *near-synonyms* we mean words that have the same meaning but differ in lexical nuances. For example, *error*, *mistake*, and *blunder* all mean a generic type of error, but *blunder* carries an implication of *accident* or *ignorance*. In addition to paying attention to lexical nuances, when choosing a word, we need to make sure it fits well with the other words in a sentence. In this article, we investigate how the collocational properties of near-synonyms can help

This work is funded by the Natural Sciences and Engineering Research Council of Canada and the University of Ottawa.

Author's address: School of Information Technology and Engineering, University of Ottawa, 800 King Edward, Ottawa, ON, Canada, K1N 6N5; email: diana@site.uottawa.ca.

with choosing the best word in each context. This problem is difficult because near-synonyms have senses that are very close to each other, and, therefore, they occur in similar contexts.

The work we present here is needed in two of our applications. The first one is an intelligent thesaurus. A writer can access a thesaurus to retrieve words that are similar to a given word when there is a need to avoid repeating the same word or when the word does not seem to be the best choice in the context. A standard thesaurus does not offer any explanation about the differences in nuances of meaning between the possible word choices. Moreover, a standard thesaurus tool does not attempt to order the choices to suit a particular writing context. Our intelligent thesaurus offers explanations and orders the choices using their collocational properties relative to the writing context.

The second application is a natural language generation (NLG) system [Inkpen and Hirst 2003] that uses symbolic knowledge of near-synonym differences. This knowledge was acquired by applying information extraction techniques to entries in various dictionaries. We included a preliminary collocation module that reduces the risk of choosing a near-synonym that does not fit with the other words in a generated sentence (i.e., violates collocational constraints). The work presented in this article allows for a more comprehensive near-synonym collocation module.

More specifically, the task we address in this article is the selection of the best near-synonym that should be used in a particular context. The natural way to validate an algorithm for this task would be to ask human readers to evaluate the quality of the algorithm's output, but this kind of evaluation would be very laborious. Instead, we validate the algorithms by deleting selected words from sample sentences to see whether the algorithms can restore the missing words, that is, we create a *lexical gap* and evaluate the ability of the algorithms to fill the lexical gap. Two examples are presented in Figure 1. All the near-synonyms of the original word, including the word itself, become the choices in the solution set (see the figure for two examples of solution sets). The task is to automatically fill the gap with the best choice in the particular context. We present a method of scoring the choices. The highest scoring near-synonym will be chosen. In order to evaluate how well our method works, we consider that the only correct solution is the original word. This will cause our evaluation scores to underestimate the performance of our method as more than one choice will sometimes be a perfect solution. Moreover, what we consider to be the best choice is the typical usage in the corpus, but it may vary from writer to writer. Nonetheless, it is a convenient way of producing test data in an automatic way. To verify how difficult the task is for humans, we perform experiments with human judges on a sample of the test data. The statistical scoring method that we propose here is based on mutual information scores of each candidate with the words in the context. We explore how far such a method can go when using the Web as a corpus. We estimate the counts by using the Waterloo MultiText System [Clarke and Terra 2003b] with a corpus of about one terabyte of text collected by a Web crawler.

Sentence: This could be improved by more detailed consideration of the processes of .... propagation inherent in digitizing procedures.
Original near-synonym: error
Solution set: mistake, blooper, blunder, boner, contretemps, error, faux pas, goof, slip, solecism

Sentence: The day after this raid was the official start of operation strangle, an attempt to completely destroy the .... lines of communication.
Original near-synonym: enemy
Solution set: opponent, adversary, antagonist, competitor, enemy, foe, rival

Fig. 1. Examples of sentences with a lexical gap and candidate near-synonyms to fill the gap.

2. RELATED WORK

The idea of using the Web as a corpus of texts has been exploited by many researchers. Radev and McKeown [1997] acquired different ways of referring to the same named entity from the Web. Grefenstette [1999] used the Web for example-based machine translation; Kilgarriff [2001] investigated the type of noise in Web data; Mihalcea and Moldovan [1999] and Agirre and Martinez [2000] used it as an additional resource for word sense disambiguation; Resnik [1999] mined the Web for bilingual texts; Turney [2001] used Web frequency counts to compute information retrieval-based mutual-information scores. In a *Computational Linguistics* special issue which focused on the Web as a corpus [Kilgarriff and Grefenstette 2003], Keller and Lapata [2003] show that Web counts correlate well with counts collected from a balanced corpus: the size of the Web compensates for the noise in the data. In this article, we are using a very large corpus of Web pages to address a problem that has not been successfully solved before.

In fact, the only work that addresses exactly the same task is that of Edmonds [1997] as far as we are aware. Edmonds gives a solution based on a lexical co-occurrence network that included second-order co-occurrences. We use a much larger corpus and a simpler method, and we obtain much better results.

Our task has similarities to the word sense disambiguation task. Our near-synonyms have senses that are very close to each other. In Senseval, some of the fine-grained senses are also close to each other, so they might occur in similar contexts, while the coarse-grained senses are expected to occur in distinct contexts. In our case, the near-synonyms are different words to choose from, not the same word with different senses.

Turney et al. [2003] addressed the multiple-choice synonym problem: given a word, choose a synonym for that word among a set of possible solutions. In this case, the solutions contain one synonym and some other (unrelated) words. They achieve high performance by combining classifiers. Clarke and Terra [2003a] addressed the same problem as Turney et al., using statistical associations measures computed with counts from the Waterloo terabyte corpus. In our case, all the possible solutions are synonyms of each other, and the task is to choose one that best matches the context: the sentence in which the original synonym is replaced with a gap. It is much harder to choose between words that are near-synonyms because the context features that differentiate a word from other words might be shared among the near-synonyms. Therefore, the choice is done on the basis of a few features that are discriminant.

3. A NEW STATISTICAL METHOD FOR NEAR-SYNONYM CHOICE

Our method computes a score for each candidate near-synonym that could fill in the gap. The near-synonym with the highest score is the proposed solution. The score for each candidate reflects how well a near-synonym fits in with the context. It is based on the mutual information scores between a near-synonym and the content words in the context (we filter out the stopwords).

The *pointwise mutual information* (*PMI*) between two words x and y compares the probability of observing the two words together (their joint probability) to the probabilities of observing x and y independently (the probability of occurring together by chance) [Church and Hanks 1991].

$$PMI(x, y) = \log_2 \frac{P(x, y)}{P(x)P(y)}.$$

The probabilities can be approximated by $P(x) = C(x)/N$, $P(y) = C(y)/N$, $P(x, y) = C(x, y)/N$, where C denotes frequency counts and N is the total number of words in the corpus. Therefore,

$$PMI(x, y) = \log_2 \frac{C(x, y) \cdot N}{C(x) \cdot C(y)},$$

where N can be ignored in comparisons since is it the same in all the cases.

We model the context as a window of size 2k around the gap (the missing word), k words to the left and k words to the right of the gap. If the sentence is $s = \ldots w_1 \ldots w_k \; Gap \; w_{k+1} \ldots w_{2k} \ldots$, for each near-synonym NS_i from the group of candidates, the score is computed by the following formula:

$$Score(NS_i, s) = \Sigma_{j=1}^{k} PMI(NS_i, w_j) \; + \Sigma_{j=k+1}^{2k} PMI(NS_i, w_j).$$

We also experimented with the same formula when the sum is replaced with maximum to see whether a particular word in the context has higher influence than the sum of all contributions.

Because we are using the Waterloo terabyte corpus and we issue queries to its search engine, we have several possibilities of computing the frequency counts. $C(x, y)$ can be the number of co-occurrences of x and y when y immediately follows x, or the distance between x and y can be up to q. We call q the query frame size. The tool for accessing the corpus allows us to use various values for q in queries. We used queries of the type $[q] > (x..y)$, which asks how many times x is followed by y in a frame of size q.

The search engine also allows us to approximate words counts with document counts. If the counts $C(x)$, $C(y)$, and $C(x, y)$ are approximated as the number of document in which they appear, we obtain the *PMI-IR* formula [Turney 2001]. The queries we need to send to the search engine are the same but they are restricted to document counts: $C(x)$ is the number of document in which x occurs; $C(x, y)$ is the number of documents in which x is followed by y in a frame of size q; the query is formulated as $(\langle doc\rangle..\langle /doc\rangle) > [q] > (x..y)$.

Other statistical association measures, such as log-likelihood, could be used. We tried only *PMI* because it is easy to compute on a Web corpus and because Clarke and Terra [2003a] showed that *PMI* performed better than other measures in their experiments.

Table I. Examples of Collocations and Anticollocations, The ∗ Indicates the Anticollocations

ghastly mistake	spelling mistake
∗ghastly error	spelling error
ghastly blunder	∗spelling blunder
∗ghastly faux pas	∗spelling faux pas
∗ghastly blooper	∗spelling blooper
∗ghastly solecism	∗spelling solecism
∗ghastly goof	∗spelling goof
∗ghastly contretemps	∗spelling contretemps
∗ghastly boner	∗spelling boner
∗ghastly slip	∗spelling slip

We present the results in Section 6.1, where we compare our method to a baseline algorithm that always chooses the most frequent near-synonyms and to Edmonds's method for the same task on the same data set. First, however, we present two other methods to which we compare our results.

4. THE ANTICOLLOCATIONS METHOD

For the task of near-synonym choice, another method that we implemented is the anticollocations method. By *anticollocation* we mean a combination of words that a native speaker would not use and therefore should not be used when automatically generating text. This method uses a knowledge-base of collocational behavior of near-synonyms that we acquired in previous work [Inkpen and Hirst 2002]. To build this knowledge base, we acquired collocations of the near-synonyms from a corpus. For each word that collocated with a near-synonym, we used a t-test (computed with Web counts collected through the AltaVista search engine) to learn whether the word forms a collocation or an anticollocation with other near-synonyms in the same group. A fragment of the knowledge base is presented in Table I for the near-synonyms of the word *error* and two collocate words *ghastly* and *spelling*. The lines marked by ∗ represent anticollocations, and the rest represent strong collocations.

The anticollocations method simply ranks the strong collocations higher than the anticollocations. In case of ties, it chooses the most frequent near-synonym. In Section 6.2, we present the results from comparing this method to the method from the previous section.

5. A SUPERVISED LEARNING METHOD

We can also apply supervised learning techniques to our task. It is easy to obtain labelled training data, the same way we collected test data for the two unsupervised methods presented previously. We train classifiers for each group of near-synonyms. The classes are the near-synonyms in the solution set. Each sentence is converted into a vector of features to be used for training the supervised classifiers. We used two types of features. One type of features are the scores of the left and right context with each class (i.e., with each near-synonym from the group). The number of features of this type is equal to twice the number of classes: one feature for the score between the near-synonym and the part of the sentence at the left of the gap, and one feature for the score between the near-synonym and the part of the sentence at the right of the gap. The second type of features are the words in the context windows. For each group of near-synonyms, we used as features the 500 most frequent words situated close to the gaps in a development set. The value of a word feature for each training example is 1 if the word is present in the sentence (at the left or at the right of the gap), and 0 otherwise. We trained classifiers using several machine learning algorithms to see which one is best at discriminating among the near-synonyms. In Section 6.3, we present the results of several classifiers.

A disadvantage of the supervised method is that it requires training for each group of near-synonyms. Additional training is required whenever we add more near-synonyms to our knowledge base. An advantage of this method is that we could improve the accuracy by using a combination of classifiers and by trying other possible features. We think that part-of-speech features of the content words in the context may not be very useful since all the possible solutions have the same part-of-speech and might have similar syntactic behavior. Maybe some function words immediately before the gaps could discriminate among the near-synonyms in some groups.

6. EVALUATION

6.1 Comparison to Edmonds's Method

In this section, we present results of the statistical method explained in Section 3. We compare our results with those of Edmonds [1997] whose solution used the texts from the Wall Street Journal (WSJ) for the year 1989 to build a lexical co-occurrence network for each of the seven groups of near-synonyms from Table II. The network included second-order co-occurrences. Edmonds used the WSJ 1987 texts for testing and reported accuracies only a little higher than the baseline. The near-synonyms in the seven groups were chosen to have low polysemy. This means that some sentences with wrong senses of near-synonyms might be in the automatically produced test set, but hopefully not many.

For comparison purposes, in this section, we use the same test data (WSJ 1987) and the same groups of near-synonyms. Our method is based on mutual information not on co-occurrence counts. Our counts are collected from a much larger corpus. The seven groups of near-synonyms used by Edmonds are listed in the first column of Table II. If we would have used groups of synonyms from WordNet, we would probably obtain similar results because the words in the seven groups differ only a little. Here are the words from the WordNet synsets:

mistake, error, fault
job, task, chore
duty, responsibility, obligation
difficult, hard
material, stuff
put up, provide, offer
decide, settle, resolve, adjudicate.

Table II. Comparison Between the New Statistical Method from Section 3, Baseline Algorithm, and Edmonds's Method (See details about Experiment 1 in Section 6.1.)

Test Set (Exp1 Data)	Number of Cases	Accuracy			
		Base-Line	Edmonds's Method	New Method (Docs)	New Method (Words)
difficult, hard, tough	6,630	41.7%	47.9%	61.0%	59.1%
error, mistake, oversight	1,052	30.9%	48.9%	66.4%	61.5%
job, task, duty	5,506	70.2%	68.9%	69.7%	73.3%
responsibility, burden, obligation, commitment	3,115	38.0%	45.3%	64.1%	66.0%
material, stuff, substance	1,715	59.5%	64.6%	68.6%	72.2%
give, provide, offer	11,504	36.7%	48.6%	52.0%	52.7%
settle, resolve	1,594	37.0%	65.9%	74.5%	76.9%
ALL (average)	31,116	44.8%	55.7%	65.1%	66.0%

Before we look at the results, we mention that the accuracy values we compute are the percentage of correct choices when filling in the gap with the winning near-synonym. The expected solution is the near-synonym that was originally in the sentence, and it was taken out to create the gap. This measure is conservative; it does not consider cases when more than one solution is correct.

Table II presents the comparative results for seven groups of near-synonyms. The last row averages the accuracies for all of the test sentences. The second column shows how many test sentences we collected for each near-synonym group. The third column is for the baseline algorithm that always chooses the most frequent near-synonym. The fourth column presents the results reported in Edmonds [1997]. The fifth column presents the result of our method when using document counts in *PMI-IR*, and the last column is for the same method when using word counts in *PMI*. We show in bold the best accuracy figure for each dataset. We notice that the automatic choice is more difficult for some near-synonym groups than for others.

To fine-tune our statistical method, we used the dataset for the near-synonyms of the word *difficult* collected from the WSJ 1989 corpus as a development set. We varied the context window size k and the query frame q, and determined good values for the parameter k and q. The best results were obtained for small window sizes, $k = 1$ and $k = 2$ (meaning k words to the left and k words to the right of the gap). For each k, we varied the query frame size q. The results are best for a relatively small query frame, $q = 3, 4, 5$, when the query frame is the same or slightly larger then the context window. The results are worse for a very small query frame, $q = 1, 2$. The results presented in the rest of the article are for $k = 2$ and $q = 5$. For all the other datasets used in this article (from WSJ 1987 and BNC), we use the parameter values as determined on the development set.

Table II shows that the performance is generally better for word counts than for document counts. Therefore, we prefer the method that uses word counts (which is also faster in our particular setting). The difference between them is not statistically significant. Our statistical method performs significantly

(1) benefit, advantage, favor, gain, profit
(2) low, gush, pour, run, spout, spurt, squirt, stream
(3) deficient, inadequate, poor, unsatisfactory
(4) afraid, aghast, alarmed, anxious, apprehensive, fearful, frightened, scared, terror-stricken
(5) disapproval, animadversion, aspersion, blame, criticism, reprehension
(6) mistake, blooper, blunder, boner, contretemps, error, faux pas, goof, slip, solecism
(7) alcoholic, boozer, drunk, drunkard, lush, sot
(8) leave, abandon, desert, forsake
(9) opponent, adversary, antagonist, competitor, enemy, foe, rival
(10) thin, lean, scrawny, skinny, slender, slim, spare, svelte, willowy, wiry
(11) lie, falsehood, fib, prevarication, rationalization, untruth

Fig. 2. Near-synonyms used in the evaluation experiments in Section 6.2.

better than both Edmond's method and the baseline algorithm. For all the results presented in this article, statistical significance tests were done using the paired t-test as described in Manning and Schütze [1999, p. 209].

In summary, the results are better for smaller context windows for the sum of all the *PMI*s with the words in the context window not for taking the maximum contribution. The performance decreases with larger query frames $q = 5, 6, \ldots, 20$ and degrades sharply when q is unlimited (the words are in the same document no matter at what distance). Error analysis reveals that incorrect choices happen more often when the context is weak, that is, very short sentences or sentences with very few content words.

On average, our method performs 22 percentage points better than the baseline algorithm and 10 percentage points better than Edmonds's method. Its performance is similar to that of the supervised method (see Section 6.3). An important advantage of our method is that it works on any group of near-synonyms without training, whereas Edmonds's method required a lexical co-occurrence network to be built in advance for each group of near-synonyms and the supervised method required training for each near-synonym group.

6.2 Comparison to the Anticollocations Method

In a second experiment, we compare the results of our methods with the anticollocations method described in Section 4. We use the dataset from Inkpen and Hirst [2002], which contains sentences from the first half of the British National Corpus with near-synonyms from the eleven groups listed in Figure 2. The number of near-synonyms in each group is higher compared with WordNet synonyms because they are taken from Hayakawa [1994], a dictionary that explains differences between near-synonyms. Moreover, we retain only the sentences in which at least one of the context words is in our previously acquired knowledge base of near-synonym collocations. That is, the anticollocations method works only if we know how a word in the context collocates with the near-synonyms from a group. For the sentences that do not contain collocations or anticollocations, it will perform no better than the baseline because the information needed by the method is not available in the knowledge base. Even if we increase the coverage of the knowledge base, the

Table III. Comparison Between the New Statistical Method from Section 3 and the Anticollocations Method from Section 4 (See details about Experiment 2 in Section 6.2.)

Test Set (Exp2 Data)	Number of Cases	Accuracy			
		Baseline	Anticollocations Method	New Method (Docs)	New Method (Words)
TestSample	171	57.0%	63.3%	75.6%	73.3%
TestAll	332	48.5%	58.6%	70.0%	75.6%

Table IV. Comparative Results for the Supervised Learning Method Using Various ML Learning Algorithms (Weka), Averaged Over the Seven Groups of Near-Synonyms From the Experiment 1 Dataset

ML method (Weka)	Features	Accuracy (averaged)
Decision Trees	PMI scores	65.4%
Decision Rules	PMI scores	65.5%
Naive Bayes	PMI scores	52.5%
K-Nearest Neighbor	PMI scores	64.5%
Kernel Density	PMI scores	60.5%
Boosting (Decision Stumps)	PMI scores	67.7%
Naive Bayes	500 word features	68.0%
Decision Trees	500 word features	67.0%
Naive Bayes	PMI + 500 word features	66.5%
Boosting (Decision Stumps)	PMI + 500 word features	69.2%

anticollocations method might still fail too often due to words that were not included.

Table III presents the results of the comparison. We used two datasets: TestSample, which includes at most two sentences per collocation (the first two sentences from the corpus) and TestAll, which includes all the sentences with collocations as they occurred in the corpus. The reason we used these two tests was not to bias the results due to frequent collocations.

The last two columns are the accuracies achieved by our new method. The second to the last column shows the results of the new method when the word counts are approximated with document counts. The improvement over the baseline is 16 to 27 percentage points. The improvement over the anticollocations method is 10 to 17 percentage points.

6.3 Comparison to Supervised Learning

We present the results of the supervised method from Section 5 on the datasets used in Section 6.1. As explained before, the datasets contain sentences with a lexical gap. For each of the seven groups of near-synonyms, the class to choose from in order to fill in the gaps is one of the near-synonyms in each cluster. We implemented classifiers that use as features either the PMI scores of the left and right context with each class, or the words in the context windows, or both types of features combined. We used as features the 500 most frequent words for each group of near-synonyms. We report accuracies for tenfold cross-validation. We used the Weka collection of machine learning algorithms [Witten and Frank 2000].

Table IV presents the results, averaged for the seven groups of near-synonyms, of several classifiers from the Weka package. The classifiers that

Table V. Comparison Between the Unsupervised Statistical Method from Section 3 and the Supervised Method Described in Section 5, on the Experiment 1 Datasets. (The results of two of the best supervised classifiers are presented.)

		Accuracy			
Test Set	Number of Cases	Base-Line	Supervised Boosting (PMI)	Supervised Boosting (PMI + Words)	Unsuper-vised Method
difficult, hard, tough	6,630	41.7%	55.8%	57.3%	59.1%
error, mistake, oversight	1,052	30.9%	68.1%	70.8%	61.5%
job, task, duty	5,506	70.2%	86.5%	86.7%	73.3%
responsibility, burden, obligation, commitment	3,115	38.0%	66.5%	66.7%	66.0%
material, stuff, substance	1,715	59.5%	70.4%	71.0%	72.2%
give, provide, offer	11,504	36.7%	53.0%	56.1%	52.7%
settle, resolve	1,594	37.0%	74.0%	75.8%	76.9%
ALL (average)	31,116	44.8%	67.7%	69.2%	66.0%

use PMI features are Decision Trees, Decision Rules, Naive Bayes, K-Nearest Neighbor, Kernel Density, and Boosting, a weak classifier (Decision Stumps which are shallow decision trees). Then a Naive Bayes classifier that uses only the word features is presented, and the same type of classifiers with both types of features. The other classifiers from the Weka package were also tried but the results did not improve, and these algorithms had difficulties in scaling up. In particular, when using the 500-word features for each training example, only the Naive Bayes algorithm was able to run in reasonable time. We noticed that the Naive Bayes classifier performs very poorly on PMI features only (55% average accuracy) but performs very well on word features (68% average accuracy). In contrast, the Decision Tree classifier performs well on PMI features, especially when using boosting with Decision Stumps. When using both the PMI scores and the word features, the results are slightly higher. It seems that both types of features are sufficient for training a good classifier, but combining them adds value.

Table V presents the detailed results of two of the supervised classifiers, and, for easier comparison, repeats the results of the unsupervised statistical method from Section 6.1. The supervised classifier that uses only PMI scores performs similar to the unsupervised method. The best supervised classifier that uses both types of features performs slightly better than the unsupervised statistical method, but the difference is not statistically significant. We conclude that the results of the supervised methods and the unsupervised statistical method are similar. An important advantage of our unsupervised method is that it works on any group of near-synonyms without training.

6.4 Comparison to Language Models

Since one of the main applications of our methods is lexical choice in machine translation and statistical NLG, we need to compare our methods with the current mainstream method for lexical choice, language modeling. The sum of mutual information scores bears some similarity to a bigram language model.

For example, $\mathit{PMI}(w_1, w_2) + \mathit{PMI}(w_2, w_3)$ can be rewritten as:

$$\mathit{PMI}(w_1, w_2) + \mathit{PMI}(w_2, w_3) = \log_2 \frac{P(w_1, w_2)}{P(w_1)P(w_2)} + \log_2 \frac{P(w_2, w_3)}{P(w_2)P(w_3)} =$$

$$= \frac{\log_2 P(w_1, w_2)P(w_2, w_3)}{P(w_1)P(w_2)^2 P(w_3)} = \frac{\log_2 P(w_2|w_1)P(w_3|w_2)}{P(w_2)^2 P(w_3)^2}$$

Lexical choice in most statistical Machine Translation (MT) systems today is heavily determined by the language model, and it has proven very difficult in practice to apply state-of-the-art Word Sense Disambiguation/lexical choice models to obtain improvements over a baseline statistical MT with a language model [Carpuat and Wu 2005].

We do not go into details here, but in previous work [Inkpen and Hirst 2006], we compared the results of the anticollocations method to the results of using a 3-gram language model. This language model is part of the HALogen NLG system [Langkilde and Knight 1998], and it was built on a 250 million-word newsarticle text: AP Newswire, year 1988 and 1990; SJM, year 1991; and WSJ, years 1987–1988 and 1990–1994. The comparison of the two methods was done on different groups of near-synonyms, including some of the groups from the Experiment 1 dataset. The anticollocation method performed better on the task of choosing the best near-synonym in a context, therefore, so would the statistical method based on PMI (because we showed in Section 6.2 that our PMI-based method achieves higher accuracy than the anticollocations method).

6.5 Experiments with Human Judges

We asked two human judges, native speakers of English, to guess the missing word in a random sample of the Experiment 1 dataset (50 sentences for each of the 7 groups of near-synonyms, 350 sentences in total). The results in Table VI show that the agreement between the two judges is high (78.5%), but not perfect. This means the task is difficult even if some wrong senses in the automatically-produced test data might have made the task easier in a few cases.

The human judges were allowed to choose more than one correct answer when they were convinced that more than one near-synonym fit well in the context. They used this option sparingly, only in 5% of the 350 sentences. In future work, we plan to allow the system to make more than one choice when appropriate (e.g., when the second choice has a very close score to the first choice).

Taking the accuracy achieved by the human judges as an upper limit, we conclude that the automatic method has room for improvement of approximately 10–15 percentage points.

7. APPLICATIONS

7.1 Intelligent Thesaurus

The intelligent thesaurus that we are developing is an interactive application that presents the user with a list of alternative near-synonyms, and, unlike standard thesauri, it orders the choices to match the writing context and explains the differences between the possible choices.

Table VI. Experiments with Two Human Judges on a Random Subset of the Experiment 1 Dataset

Test Set	J1-J2 Agreement	J1 Accuracy	J2 Accuracy	System Accuracy
difficult, hard, tough	72%	70%	76%	53%
error, mistake, oversight	82%	84%	84%	68%
job, task, duty	86%	92%	92%	78%
responsibility, burden, obligation, commitment	76%	82%	76%	66%
material, stuff, substance	76%	82%	74%	64%
give, provide, offer	78%	68%	70%	52%
settle, resolve	80%	80%	90%	77%
ALL (average)	78.5%	79.7%	80.2%	65.4%

Table VII. Accuracies for the First Two Choices as Ordered by an Interactive Intelligent Thesaurus

Test Set	Accuracy for the First Choice	Accuracy for the First Two Choices
Experiment 1 Data, ALL	66.0%	88.5%
Experiment 2 Data, TestSample	73.3%	94.1%
Experiment 2 Data, TestAll	75.6%	87.5%

The new statistical method presented in this article (Section 3) allows us to order the near-synonyms according to how well they fit into the writing context. When composing a text, if the writer is unhappy with a word, he/she can select that word and ask for a better substitute. The intelligent thesaurus provides alternative near-synonyms in a context-dependent manner. Our experiments show that the accuracy of the first choice as the best choice is 66 to 75%; therefore, there will be cases when the writer will not choose the first alternative. But the accuracy for the first two choices is quite high, around 90%, as presented in Table VII.

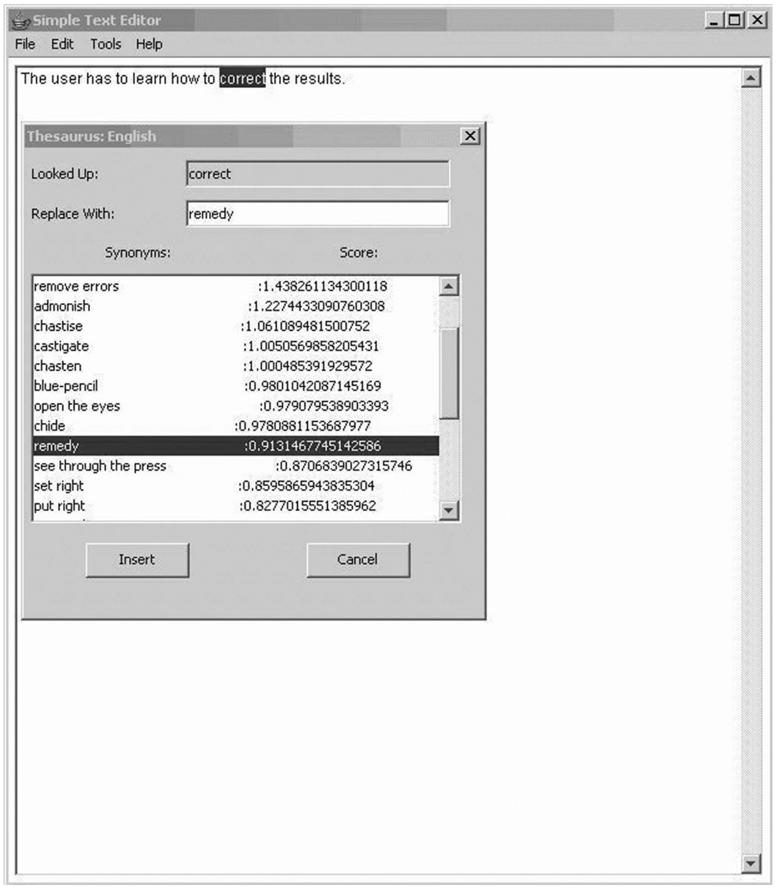
If the writer is in the process of writing and selects the last word to be replaced with a near-synonym proposed by the thesaurus, then only the context on the left of the word can be used for ordering the alternatives. Our method can be easily adapted to consider only the context on the left of the gap. The results of this case are presented in Table VIII, for the datasets used in the previous sections. The accuracy values are lower than in the case when both the left and the right context are considered (Table VII). This is due in part to the fact that some sentences in the test sets have very little left context or no left context at all. On the other hand, many times the writer composes a sentence or paragraph and then she/he goes back to change a word that does not sound right. In this case, both the left and right context will be available.

In the intelligent thesaurus, we could combine the supervised and unsupervised method by using a supervised classifier when the confidence in the classification is high and by using the unsupervised method otherwise. Also the unsupervised statistical method would be used for the groups of near-synonyms for which a supervised classifier was not previously trained.

Figure 3 presents a screen shot of the current implementation of the intelligent thesaurus application. The system allows the user to specify which

Table VIII. Results of the Statistical Method When Only the Left Context is Considered, for the datasets from Section 6.1

Test Set	Accuracy for the First Choice	Accuracy for the First Two Choices
Experiment 1 Data, ALL	58.05%	84.82%
Experiment 2 Data, TestSample	57.4%	75.1%
Experiment 2 Data, TestAll	56.1%	77.4%



The user has to learn how to correct the results.

Looked Up: correct

Replace With: remedy

Synonyms:	Score:
remove errors	:1.438261134300118
admonish	:1.2274433090760308
chastise	:1.061089481500752
castigate	:1.0050569858205431
chasten	:1.000485391929572
blue-pencil	:0.9801042087145169
open the eyes	:0.979079538903393
chide	:0.9780881153687977
remedy	:0.9131467745142586
see through the press	:0.8706839027315746
set right	:0.8595865943835304
put right	:0.8277015551385962

Fig. 3. Screen shot of the intelligent thesaurus application.

synonym inventory to use. We integrated Roget's thesaurus [Roget 1852] in order to increase coverage (the number of words for which alternatives are available). We used the interface provided by Jarmasz and Szpakowicz [2001]. Roget's thesaurus contains among the related words several phrases (as seen in Figure 3). In future work, we need to find new ways to score these phrases, because the PMI formula that we used gives them scores that are too high.

The system also allows us to specify from which corpus to collect the counts (including from the Web, through the Google and Yahoo API). The system performs part-of-speech tagging (using QTAG[1]) on the text written in the text editor in order to be able to suggest only synonyms with the right part-of-speech.

Our intelligent thesaurus can present in a separate window explanations of how the near-synonyms differ in the lexical nuances they carry. This kind of knowledge is available in a limited form from our previous work [Inkpen and Hirst 2001] about extracting information from a special dictionary of synonym discrimination [Hayakawa 1994]. The coverage is limited in the sense that this knowledge base contains only 5452 words in 909 groups of near-synonyms. The explanations include denotational nuances (what do the near-synonyms imply or suggest), how formal or informal they are, and how positive or negative they might sound; also examples of usage are provided. We do not expand on this aspect of our intelligent thesaurus here because the focus of this article is on the collocational properties of the near-synonyms.

7.2 Natural Language Generation

In previous work [Inkpen and Hirst 2006], we presented and evaluated an NLG system that pays particular attention to choosing the right near-synonyms. This system extended HALogen [Langkilde and Knight 1998] with two extra modules: a symbolic and a statistical module for near-synonym choice.

Without going into details, we mention that the symbolic module deals with nuances of meaning in an explicit way. Connotations or implications of certain concepts are represented using an interlingual representation language and concepts from an ontology. The user can specify what nuances of meaning are preferred in the generated sentence, in addition to the main meaning. Alternatively, the preferences could come from the sentence in the source language if our NLG system is used as part of an interlingual machine translation system. The system knows which near-synonyms carry which nuances of meaning because it contains the knowledge automatically acquired from the special dictionary of synonym discrimination [Hayakawa 1994].

The statistical module uses the anticollocations method to choose near-synonyms that collocate well with the other words in the context. We plan to replace it with the PMI-based method in order to increase coverage and accuracy.

8. CONCLUSION

We presented a statistical method of choosing the best near-synonym in a context. We compared this method with two previous methods (Edmonds's method and the anticollocations method) and showed that the performance improves considerably. We also show that our unsupervised statistical method performs comparably to a supervised learning method.

Our method based on PMI scores performs well despite the well-known limitations of PMI when used with corpora. PMI tends to have problems mostly on

[1] http://www.english.bham.ac.uk/staff/omason/software/qtag.html.

very small counts, but it works reasonably with larger counts. Our web corpus is quite large, therefore the problem of small counts does not appear.

We combine symbolic and statistical knowledge in two applications, an intelligent thesaurus and a natural language generation system that has knowledge of nuances of meaning of near-synonyms.

The accuracy of 66 to 75% is not enough for automatic choice but helps complement the choices made by using symbolic knowledge. In the intelligent thesaurus, we do not make the near-synonym choice automatically, but we let the user choose. The first choice offered by the thesaurus is quite often the best one and if we consider the first two choices, they are correct 90% of the time.

In this article we focused on idiomatic usage of near-synonyms, while, in previous work, we looked at nuances of meaning and differences between near-synonyms in terms of connotations and implications [Inkpen and Hirst 2006]. There are, though, some implications that are captured by idiomatic usage. For example, Church et al. [1991] presented associations (collocations, but not necessarily between adjacent words) for the near-synonyms *ship* and *boat*; they suggest that a lexicographer looking at these associations can infer that *boats* are generally smaller than *ships* because they are found in rivers and lakes and are used for small jobs (e.g., fishing, police, pleasure), whereas ships are found in seas and are used for serious business (e.g., cargo, war).

In future work, we plan to investigate the possibility of automatically inference for this kind of knowledge or to validate already acquired knowledge. Words that do not associate with a near-synonym but associate with all the other near-synonyms in a cluster could tell us something about its nuances of meaning. For example *terrible slip* is an anti-association, whereas *terrible* associates with *mistake, blunder, error*. This is an indication that *slip* is a minor error. By further generalization, the associations could become conceptual associations. This may allow the automatic learning of denotational distinctions between near-synonyms from free text. The concepts that are common to all the near-synonyms in a cluster could be part of their main meaning, while those that associate only with one near-synonym could be part of their implied nuances of meaning.

Future work includes a near-synonym sense disambiguation module to ensure that the intelligent thesaurus does not offer alternatives for wrong senses of words. In addition to the groups of synonyms from WordNet, Roget, and dictionaries of synonyms, we could acquire synonyms from corpora so that the intelligent thesaurus can offer alternatives for a very large number of words. There is research done on acquiring distributionally similar words [Lin 1998], but they include, in addition to near-synonyms, words that have other relations. Lin et al. [2003] looked at filtering out the antonyms, using specific co-occurrence patterns. Words that have other relations could also be filtered out. One way to do this could be to collect signatures for each potential near-synonym—words that associate with it in many contexts. For two candidate words, if one signature is contained in the other, the words are probably in an IS-A relation; if the signatures overlap totally, it is a true near-synonymy relation; if the signatures overlap partially, it is a different kind of relation.

ACKNOWLEDGMENTS

We are most grateful to Egidio Terra, Charlie Clarke, and the School of Computer Science of the University of Waterloo, for allowing us to use their terabyte webpage corpus and the MultiText system. We thank Peter Turney and his colleagues at IIT/NRC Ottawa for giving us access to their local copy of the corpus. We also thank Peter Turney for sharing his Perl code for remote access to the corpus and for his comments on the draft of this article. We thank Graeme Hirst for comments on earlier versions of this article. We thank Jayakumar Balasingham and Apoorve Chokshi for their contribution to the implementation of the Intelligent Thesaurus application and Mario Jarmasz for allowing us to use his software interface to Roget's thesaurus.

REFERENCES

Agirre, E. and Martinez, D. 2000. Exploring automatic word sense disambiguation with decision lists and the Web. In *Proceedings of the Workshop on Semantic Annotation And Intelligent Content (COLING00.)* Saarbrücken/Luxembourg/Nancy.

Carpuat, M. and Wu, D. 2005. Evaluating the word sense disambiguation performance of statistical machine translation. In *Proceedings of the 43th Annual Meeting of the Association for Computational Linguistics*. Ann Arbor, MI. 120–125.

Church, K., Gale, W., Hanks, P., and Hindle, D. 1991. Using statistics in lexical analysis. In U. Zernik, Ed. *Lexical Acquisition: Using Online Resources to Build a Lexicon*. Lawrence Erlbaum, 115–164.

Church, K. and Hanks, P. 1991. Word association norms, mutual information and lexicography. *Comput. Linguis. 16*, 1, 22–29.

Clarke, C. L. A. and Terra, E. 2003a. Frequency estimates for statistical word similarity measures. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL03)*. Edmonton, Canada, 165–172.

Clarke, C. L. A. and Terra, E. 2003b. Passage retrieval vs. document retrieval for factoid question answering. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Toronto, Canada, 427–428.

Edmonds, P. 1997. Choosing the word most typical in context using a lexical co-occurrence network. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*. Madrid, Spain, 507–509.

Grefenstette, G. 1999. The World Wide Web as a resource for example-based machine translation tasks. In *Proceedings of the ASLIB Conference on Translating and Computers*. London, UK.

Hayakawa, S. I., Ed. 1994. *Choose the Right Word* 2nd Ed. (Revised by Eugene Ehrlich). Harper Collins Publishers.

Inkpen, D. Z. and Hirst, G. 2001. Building a lexical knowledge-base of near-synonym differences. In *Proceedings of the Workshop on WordNet and Other Lexical Resources, Second Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL01)*. Pittsburgh, PA. 47–52.

Inkpen, D. Z. and Hirst, G. 2002. Acquiring collocations for lexical choice between near-synonyms. In *Proceedings of the Workshop on Unsupervised Lexical Acquisition 40th Annual Meeting of the Association for Computational Linguistics (ACL02)*. Philadelphia, PA. 67–76.

Inkpen, D. Z. and Hirst, G. 2003. Near-synonym choice in natural language generation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing RANLP'03*. Borovets, Bulgaria, 204–211.

Inkpen, D. Z. and Hirst, G. 2006. Building and using a lexical knowledge-base of near-synonym differences. *Comput. Linguis. 32*, 2.

JARMASZ, M. AND SZPAKOWICZ, S. 2001. The design and implementation of an electronic lexical knowledge base. In *Proceeding of the 14th Biennial Conference of the Canadian Society for Computational Studies of Intelligence (AI01)*. Ottawa, Canada, 325–334.

KELLER, F. AND LAPATA, M. 2003. Using the Web to obtain frequencies for unseen bigrams. *Comput. Lingui. 29*, 3, 459–484.

KILGARRIFF, A. 2001. Web as corpus. In *Proceedings of the Corpus Linguistics Conference*. Lancaster, UK, 342–345.

KILGARRIFF, A. AND GREFENSTETTE, G. 2003. Introduction to the special issue on the Web as a corpus. *Computa. Linguis. 29*, 3, 333–347.

LANGKILDE, I. AND KNIGHT, K. 1998. The practical value of N-grams in generation. In *Proceedings of the 9th International Natural Language Generation Workshop*. Niagara-on-the-Lake, Canada, 248–255.

LIN, D. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics Joint with 17th International Conference on Computational Linguistics (ACL-COLING'98)*. Montreal, Quebec, Canada, 768–774.

LIN, D., ZHAO, S., QIN, L., AND ZHOU, M. 2003. Identifying synonyms among distributionally similar words. In *Proceedings of the 18th Joint International Conference on Artificial Intelligence (IJCAI03)*. Acapulco, Mexico.

MANNING, C. AND SCHÜTZE, H. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, MA.

MIHALCEA, R. AND MOLDOVAN, D. 1999. A method for word sense disambiguation from unrestricted text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*. 152–158.

RADEV, D. AND MCKEOWN, K. R. 1997. Building a generation knowledge source using Internet-accessible newswire. In *Proceedings of the 5th ACL Conference on Applied Natural Language Processing (ANLP'97)*. Washington, DC, 221–228.

RESNIK, P. 1999. Mining the Web for bilingual text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*. 527–534.

ROGET, P. M., Ed. 1852. *Roget's Thesaurus of English Words and Phrases*. Longman Group Ltd., Harlow, Essex, UK.

TURNEY, P. 2001. Mining the Web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the Twelfth European Conference on Machine Learning (ECML 2001)*. Freiburg, Germany, 491–502.

TURNEY, P., LITTMAN, M., BIGHAM, J., AND SHNAYDER, V. 2003. Combining independent modules to solve multiple-choice synonym and analogy problems. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing RANLP'03*. Borovets, Bulgaria, 482–489.

WITTEN, I. H. AND FRANK, E. 2000. *Data Mining: Practical Machine Learning Tools with Java Implementations*. Morgan Kaufmann, San Francisco, CA.

Received September 2004; revised March 2006, October 2006; accepted November 2006 by Dekai Wu