Detection of Translational Equivalence

Noah A. Smith
Department of Computer Science
University of Maryland, College Park
nasmith@cs.umd.edu

15 May 2001

Abstract

I propose a general algorithm for detecting translational equivalence between text samples in different languages. This algorithm is based on current approaches to duplicate detection, and it relies on information which can be automatically learned from parallel text. I also show experimental results which support the hypothesis that translational equivalence is empirically observable. In addition, these results suggest profitable directions for improving performance on this recognition task.[1]

This work is submitted in partial fulfillment of the requirements for the degree of **Bachelor of Science in Computer Science with High Honors** at the **University of Maryland, College Park**.

Thesis advisor: Professor Philip S. Resnik, Departments of Linguistics and Computer Science and Institute for Advanced Computer Study.

[1]The research presented here was supported in part by the National Science Foundation, Johns Hopkins University, and DARPA/ITO Cooperative Agreement N660010028910.

Contents

List of Tables

List of Figures

1 Introduction

While the task of automatic translation is one that has received a great deal of attention in current literature (e.g., [WS99]), the complementary task of recognizing translations is largely unexplored. One may ask the question, "given two linguistic samples E and F, does the predicate $\mathit{Translation}(E, F)$ hold true?" More generally, one might produce a confidence score for the translational equivalence of E and F.

One candidate for such a score is $\Pr(E, F)$ for some statistical generative model of translation. Melamed [Mel00] describes three symmetric word-to-word models which are learnable from text in parallel translation (bi-texts). However, given E and F, the computation of the true value of $\Pr(E, F)$ is expensive because it requires a summation over all possible word-to-word assignments. In order to avoid the full cost of this computation, Melamed utilizes the *maximum a posteriori* (MAP) approximation in his model estimation methods. However, an additional problem presents itself for this particular application of such models: the value of $\Pr(E, F)$ diminishes exponentially as the lengths of E and F increase. This is problematic for two reasons; one might wish to find translation pairs in sets where the strings are either very long (e.g., documents) or highly variable in length.

Consider an example in which we wish to choose the best French translation for the English sentence "John buys shirts in Paris" (E). In this simple case, suppose we have two options: "Jean mange" (F_1) and "Jean achète souvent des chemises à Paris" (F_2). Note the lengths of these strings:

$$\begin{aligned} |E| &= 5 \\ |F_1| &= 2 \\ |F_2| &= 7 \end{aligned}$$

In the Melamed models, a single word may have either one or zero corresponding words in a generated translation pair. Words are generated in pairs, one from each language, in which one word or the other in a pair may be null, but not both [Mel00]. This means that, if E and F_1 were generated as a pair, there were between 5 and 7 word pairs generated; if E and F_2 were generated as a pair, there were between 7 and 12 word pairs generated. Clearly, by such a model, $\Pr(E, F_1)$ stands a good chance of being significantly greater than $\Pr(E, F_2)$. If that probability distribution is the scoring function for translational equivalence, performance is predicted to be quite horrendous.

This work seeks to build on the idea of a symmetric translation model as a useful tool in recognizing translational equivalence in text while avoiding the sentence-length problem and keeping computational feasibility in mind.

Consideration of the translation detection task begs the question of what it means for two text-strings to be mutual translations. It has been argued [Who73] that translation between languages is not possible. This investigation seeks to show that empirically measurable properties of strings suffice to learn automatic classifiers (or, more generally, scoring functions) to support the hypothesis that "translation-ness" is an observable property of some bi-texts. Following an intuition made explicit by Alshawi et al. [ABD00], I suggest that a profitable approach may be to avoid artificial meaning representations in favor of the most natural ones — the strings themselves.

1.1 Potential Applications

I suggest four practical applications of such a scoring function. The first is in parallel corpus construction using systems like STRAND [Res99] and Nie et al.'s system [NSID99]. STRAND is a a tool which automatically discovers World-Wide Web pages which may be mutual translations (in a given language pair), then filters the candidates based on structural similarity evidenced by the language-independent markup tags present in the documents. While the precision of STRAND is extremely high, experiments show that the filter is overly strict, giving a yield with room for improvement.

STRAND carries out two types of search, both of which could benefit from a translation scoring function. In the first, document-pairs are classified based on their markup tags. A correlation score determines the likelihood that the two documents are parallel text, but in some cases one document in an actual translation pair will have extra text. This results in the entire document pair being discarded. If the text chunks (between markup tags) could be evaluated on their own, portions of such asymmetric documents might be salvaged to increase the yield of STRAND without affecting precision.

The other search task STRAND addresses is the pairing up items from two lists of candidate pages. Given two sets of documents, STRAND attempts to generate an assignment[2] between them, but because of the high computational cost of comparing the contents of each possible document pair, STRAND produces candidates based solely on the URL strings. (Nie et al. [NSID99] and Chen and Nie [CN00] used a similar approach.) The markup filter is then applied, but if the candidates are wrongly paired, translation pairs may be lost simply because they weren't paired based on URL string similarity.

A second application considers text-strings of shorter length; computing a *maximum a posteriori* (MAP) word-to-word assignment where some syntactic information is available is a task faced, e.g., in translation modeling. If, for example, NP-bracketing is available for both sides of a bi-text, determining which contiguous chunks (NPs or extra-NP chunks) correspond with each other using a general classifier would help to guide the search. This application could be part of a framework involving bootstrapping of more complex translation models from simpler ones (e.g. [MS00], [BB94]).

A third application involves comparable corpora. A comparable corpus contains text in multiple languages that is known to contain some similar content, such as news from the same time period. While comparable corpora do not contain direct translations, a comparable corpus might be assumed to contain some translationally-equivalent material. Extracting this material might be a technique for parallel corpus construction. More generally, ranking potentially translationally equivalent portions of the corpus could provide a means to weight examples for some other learning tasks.

Finally, in multilingual information retrieval, one would prefer to avoid returning translationally-equivalent duplicates. If two documents can be classified as duplicates (i.e., translations of each other), there exists a potential for improved recall in N-best systems. At the same time, translation detection could allow a cost savings when the translation of a document is desired; if the translation exists in a database and can be identified, the task of translating the document is not necessary [Oard01].

[2] I use the term "assignment" to refer to chunk-to-chunk mappings which respect no ordering restrictions, and I use the term "alignment" to refer to such mappings which do not allow "cross-over."

1.2 Background: Duplicate Detection

Broder et al. [BGMZ97] sought to detect copies of documents in a single language. They propose two document similarity scores, resemblance (r) and containment (c).

In order to compute these scores, a document D is viewed as a set of shingles [Dam95] (a "shingling"), where a shingle is an n-gram type (i.e., a contiguous subsequence of length n) contained in D. For example, the trigram shingling of the next sentence is {"denote the shingling", "the shingling of", "shingling of D", "of D as", "D as $S(D)$"}. Denote the shingling of D as $S(D)$. Resemblance is a measure in $[0, 1]$; a higher score indicates a higher degree of similarity between two documents. It is defined for documents D_1 and D_2 as follows:

$$r(D_1, D_2) = \frac{|S(D_1) \cap S(D_2)|}{|S(D_1) \cup S(D_2)|} \tag{1}$$

Containment is also in $[0, 1]$. It indicates a level of confidence that D_1 is contained within D_2. It is defined as:

$$c(D_1, D_2) = \frac{|S(D_1) \cap S(D_2)|}{|S(D_1)|} \tag{2}$$

Broder et al. used a sampling technique for estimating the shingling of documents; for present purposes I shall not address this issue, since most of my discussion is directed toward smaller text segments for which sampling is unnecessary.

Another approach to similarity is given by Lin [Lin98]. Lin gives a theoretically-motivated general description of similarity:

$$sim(D_1, D_2) = \frac{\log \Pr[\text{common}(D_1, D_2)]}{\log \Pr[\text{description}(D_1, D_2)]} \tag{3}$$

That is, the similarity between D_1 and D_2 is measured by the ratio between the amount of information needed to state the commonality of D_1 and D_2 and the information needed to fully describe what D_1 and D_2 are.

This definition is in fact quite similar (no pun intended) to the one offered by [BGMZ97]. If we view D_1 as $S(D_1)$ and D_2 as $S(D_2)$, then the Lin similarity measure is given as:

$$sim(D_1, D_2) = \frac{2 \times \sum_{s \in S(D_1) \cap S(D_2)} \log \Pr(s)}{\sum_{s \in S(D_1)} \log \Pr(s) + \sum_{s \in S(D_2)} \log \Pr(s)} \tag{4}$$

For purposes of comparison with the [BGMZ97] r score, note that, trivially:

$$|S(D_1) \cap S(D_2)| = \sum_{x \in S(D_1) \cap S(D_2)} 1 \tag{5}$$

$$|S(D_1) \cup S(D_2)| = \sum_{x \in S(D_1) \cup S(D_2)} 1 \tag{6}$$

While the two similarity measures are by no means mathematically equivalent, one notes that:

- The domain of the summed items is $\{0, 1\}$ for [BGMZ97] and $(-\infty, 0]$ (continuous) for [Lin98].
- The denominator values, while not identical, are related. (Note that $|S(D_1) \cup S(D_2)| = |S(D_1)| + |S(D_2)| - |S(D_1) \cap S(D_2)|$.)

- The factor of 2 in [Lin98] is irrelevant in a competitive scoring framework.

The key difference between the two approaches, for practical purposes, is that *sim* assumes a probability model, while r assumes discrete sets.

2 Translational Equivalence as a Function Over Sets

The *sim* measure requires a probability distribution over shingle (n-gram) types s. When considering documents in a single language, it is straightforward to estimate the parameters to an n-gram model. However, it becomes less clear how to define such a probability model over shingles when dealing with multilingual documents; yet this is required for computing the intersection of shingle sets in different languages. The concept of shingle *equality* must first be addressed.

Let us suppose that there is a set of language-independent concepts C which is universal. In the text production process, these concepts are lexicalized and ordered according to language-specific parameters. Let each concept c have a set $X_{\mathcal{L}}(c)$ of lexical items in language $\mathcal{L}$ which are candidates in the lexicalization process.

We may consider two elements e and f in two languages $\mathcal{L}_1$ and $\mathcal{L}_2$, respectively, to be translationally equivalent if and only if there exists a concept c such that $e \in X_{\mathcal{L}_1}(c)$ and $f \in X_{\mathcal{L}_2}(c)$[3]

Unfortunately, the set of concepts C is not directly observable (and arguably may not exist). Curtailing this issue, let us suppose that we have some means to estimate a confidence score for the following statement:

$$\exists c \in C : e \in X_{\mathcal{L}_1}(c) \wedge f \in X_{\mathcal{L}_2}(c) \tag{7}$$

Let the confidence score be in the domain $[0, 1]$: 0 indicates an assertion that e and f are not at all translationally equivalent, while 1 indicates an assertion that they are. Note that this is not a probability distribution. (The discussion of deriving such a confidence score from data follows in later sections.)

A desirable property of such a score is that a shingle e (in $\mathcal{L}_1$) may hold translational equivalence with any number of shingles f (in $\mathcal{L}_2$). For example, English unigram *the* is generally considered to hold a high degree of equivalence with French unigrams *le*, *la*, *les*, and *l'*. One-to-many (and many-to-many) relationships need not affect strengths of association when dealing with confidence scores.

Such confidence scores, however, do not yield the information needed to determine the probability distribution over language-independent concepts as required by the *sim* measure. Rather than attempting to develop a generative model for entirely unobservable concepts, I utilize the [BGMZ97] measure of resemblance as an indicator of text similarity.

Resemblance, however, comes with its own difficulties. At first blush, it appears that one must define operations of set-union and set-intersection over language-independent shingles. This is, however not the case. The operations need not be defined at all; it is only the *cardinality* of the sets that is of interest. Given the confidence scores, we may estimate the

[3]This discussion ignores entirely the problems of polysemy and context. A word that is polysemous has multiple meanings (usually related, such as English "chicken" the animal and "chicken" the food). In different contexts, words which are, by my definition, "translationally equivalent," may not have the same meaning at all. I assume that the cases where incorrect conclusions are drawn about two tokens' translational equivalence, due to lack of attention to context, will add only minor noise. This is an area requiring further consideration in the future.

cardinality of a set-intersection based on the confidence that each element in one set shares a concept with *any* element in the other set.

I use the notation E and F to refer to two text samples in $\mathcal{L}_1$ and $\mathcal{L}_2$ respectively whose similarity we seek to estimate. Let $S(E)$ and $S(F)$ be the (language-specific) shingle sets for the two text samples. Hence if:

$$\begin{aligned} E &= \text{“Philip drinks coffee with sugar”} \\ F &= \text{“Philippe boit du café avec du sucre”} \end{aligned}$$

then for trigram shingles:

$$\begin{aligned} S(E) &= \{\text{“Philip drinks coffee”}, \text{“drinks coffee with”}, ...\} \\ S(F) &= \{\text{“Philippe boit du”}, \text{“boit du café”}, ...\} \end{aligned}$$

Unigram shingle sets of E and F are:

$$\begin{aligned} S(E) &= \{\text{“Philip”}, \text{“drinks”}, \text{“coffee”}, \text{“with”}, \text{“sugar”}\} \\ S(F) &= \{\text{“Philippe”}, \text{“boit”}, \text{“du”}, \text{“café”}, \text{“avec”}, \text{“sucre”}\} \end{aligned}$$

For a shingle set $S(x)$, denote the set of underlying concepts lexicalized in the set by $\lambda(S(x))$. Next, define the function $\overset{t}{\cap}\ (S(E), S(F))$ as the set of concepts lexicalized in both E and F. This is equivalent to $\lambda(S(E)) \cap \lambda(S(F))$. Likewise, $\overset{t}{\cup}\ (S(E), S(F))$ is the set of concepts lexicalized in either E or F (equivalent to $\lambda(S(E)) \cup \lambda(S(F))$). I emphasize that this approach does not seek to estimate the contents either set. Because the $\overset{t}{\cap}$ and $\overset{t}{\cup}$ functions are so similar to normal set intersection and union, I use the notation $S(E) \overset{t}{\cap} S(F)$ and $S(E) \overset{t}{\cup} S(F)$. Note next that we may define $\overset{t}{\cup}$ in terms of $\overset{t}{\cap}$:

$$\left| S(E) \overset{t}{\cup} S(F) \right| = |\lambda(S(E))| + |\lambda(S(F))| - \left| S(E) \overset{t}{\cap} S(F) \right| \tag{8}$$

This is analogous to an intuition noted in the previous section about the standard $\cup$ and $\cap$ functions over sets.

Before defining $|S(E) \overset{t}{\cap} S(F)|$, I note the intuition that:

$$\left| S(E) \overset{t}{\cap} S(F) \right| \leq \min\left(|\lambda(S(E))|, |\lambda(S(F))|\right) \tag{9}$$

That is, the intersection may be no greater in cardinality than either of the argument sets. I assume that $|\lambda(S(x))| = |S(x)|$, that is, that exactly one unique concept is lexicalized by each shingle in the set.

Finally, define the notation $e \overset{t}{=} f$ as the translational equivalence between shingle e and shingle f. In this discussion, the generic notation for the confidence value of the truth of “$e \overset{t}{=} f$” will be $t(e, f)$.

Consider first a simple case where $|S(E)| = 1$ and $|S(F)| = 1$. In this scenario, $|S(E) \overset{t}{\cap} S(F)|$ is 1 if $e \overset{t}{=} f$ and 0 otherwise. where e is the sole shingle in $S(E)$ and f the sole shingle in $S(F)$. Therefore, whatever confidence we have for the truth of “$e \overset{t}{=} f$” (i.e., $t(e, f)$) will be the estimate for $\left| S(E) \overset{t}{\cap} S(F) \right|$.

The presence of e in $S(F)$ is the degree of confidence that $e \stackrel{t}{=} f$ for *any* f. I consider this confidence score to be additive; if $t(e, f_1) = 0.3$ and $t(e, f_2) = 0.2$ then $|\{e\} \stackrel{t}{\cap} \{f_1, f_2\}|$ is 0.5. This becomes problematic when the summation of confidence values is greater than one. Confidence must fall in the domain $[0, 1]$, and further, a value greater than $|S(E)| = 1$ is inconsistent with the intuition in (9). For this reason, I define the presence π of e in $S(F)$ as follows:

$$\pi[e, S(F)] = \min\left(1, \sum_{f \in S(F)} t(e, f)\right) \tag{10}$$

Finally, the general case definition of $\left|S(E) \stackrel{t}{\cap} S(F)\right|$ is:

$$\begin{aligned} \left|S(E) \stackrel{t}{\cap} S(F)\right| &= \min\left(\sum_{e \in S(E)} \pi[e, S(F)], \sum_{f \in S(F)} \pi[f, S(E)]\right) \\ &= \min\left(\sum_{e \in S(E)} \min\left(1, \sum_{f \in S(F)} t(e, f)\right), \sum_{f \in S(F)} \min\left(1, \sum_{e \in S(E)} t(e, f)\right)\right) \end{aligned} \tag{11}$$

The outermost minimum forces the intuition stated in (9).

We now redefine the resemblance score r using $\stackrel{t}{\cap}$ and $\stackrel{t}{\cup}$:

$$r(D_1, D_2) = \frac{\left|S(D_1) \stackrel{t}{\cap} S(D_2)\right|}{\left|S(D_1) \stackrel{t}{\cup} S(D_2)\right|} \tag{12}$$

An illustrative example is given in Table 1 and Figure 1.

	e			
f	Philip	doesn't	drink	tea
Philippe	1	0	0	0
ne	0	1	0	0
boit	0	0	1	0
pas	0	1	0	0
de	0	0	0	0
thé	0	0	0	1

Table 1: A sample boolean t function.

Further, we might also redefine *sim* using notions defined in this section:

$$sim(D_1, D_2) = \frac{2 \times \sum_{s \in S(D_1) \stackrel{t}{\cap} S(D_2)} \log \Pr(s)}{\sum_{s \in \lambda(S(D_1))} \log \Pr(s) + \sum_{s \in \lambda(S(D_2))} \log \Pr(s)} \tag{13}$$

It was previously noted that this score requires probability distributions over shingles that might be in different languages. By using language-independent concepts instead of the shingles themselves, the problem of computing the intersection between two text samples' shingles is avoided. A new problem presents itself, however: how can we estimate a probability model over unobservable language-independent concepts? Seeking to define, let alone

$$
\begin{array}{rclcl}
\pi[\text{Philip}, S(F)] & = & \min(1, 1+0+0+0+0+0) & = & 1 \\
\pi[\text{doesn't}, S(F)] & = & \min(1, 0+1+0+1+0+0) & = & 1 \\
\pi[\text{drink}, S(F)] & = & \min(1, 0+0+1+0+0+0) & = & 1 \\
\pi[\text{tea}, S(F)] & = & \min(1, 0+0+0+0+0+1) & = & 1 \\
\pi[\text{Philippe}, S(E)] & = & \min(1, 1+0+0+0) & = & 1 \\
\pi[\text{ne}, S(E)] & = & \min(1, 0+1+0+0) & = & 1 \\
\pi[\text{boit}, S(E)] & = & \min(1, 0+0+1+0) & = & 1 \\
\pi[\text{pas}, S(E)] & = & \min(1, 0+1+0+0) & = & 1 \\
\pi[\text{de}, S(E)] & = & \min(1, 0+0+0+0) & = & 0 \\
\pi[\text{thé}, S(E)] & = & \min(1, 0+0+0+1) & = & 1
\end{array}
$$

$$
\begin{aligned}
\sum_{e \in S(E)} \pi(e, S(F)) &= 1+1+1+1 \\
&= 4 \\
\sum_{f \in S(F)} \pi(f, S(E)) &= 1+1+1+1+0+1 \\
&= 5 \\
\left| S(E) \overset{t}{\cap} S(F) \right| &= \min(4,5) \\
&= 4 \\
\left| S(E) \overset{t}{\cup} S(F) \right| &= |\lambda(S(E))| + |\lambda(S(F))| - 4 \\
&= 4+6-4 \\
&= 6 \\
r &= \frac{4}{6} = \frac{2}{3}
\end{aligned}
$$

Figure 1: An example of computing the r score. The shingles here are unigrams. The values of $t(e,f)$ come from Table 1. The text samples are “Philip doesn't drink tea” and “Philippe ne boit pas de thé.”

model, these concepts, goes against the intuition that the best linguistic representation of a sample is the text string itself.

The newly-defined resemblance score allows us to avoid explicitly estimate the concepts present in a text sample because it deals only with cardinalities of sets. Given a word-to-word translational equivalence function (generically, t), we can compute r for pairs of text strings without ever resorting to artificial linguistic representations. Some t functions on which these definitions might be based will be addressed in Section 3.

3 Word-to-Word Equivalence Functions

In order to use the similarity measure described in Section 2 to estimate the degree of translational equivalence between a bilingual pair of test samples, we must define a measure of translational equivalence between shingles in the two languages.

I consider here shingles of length one (i.e., unigrams). This approach has several advantages:

- It affords the fewest problems with sparse data.
- It is likely to closely mirror actual translational equivalence between text items, since the majority of empirically observable correspondences are one (word) to one (word) [Mel00].
- It fits most cleanly with available data sources.
- It fits most cleanly with current translation models.

I consider three informing functions which may be used as an estimate of the $\stackrel{t}{=}$ function: electronic *a priori* bilingual dictionaries, automatically-learned probabilistic word-to-word translation models, and cognate-ness scores. The first two are described in this section; a discussion of cognates is in Section 4.

3.1 Bilingual Dictionaries

For some language pairs, electronic bilingual dictionaries are available. Previous work (e.g., [BD93]) has viewed such resources as exploitable data, and they are particularly appropriate for the task of identifying translational equivalence.

A bilingual dictionary may be adapted to suit this purpose in a straightforward way. Let $t(e, f)$ be a boolean predicate such that $t(e, f) = 1$ if the entry (e, f) is present in the dictionary and 0 otherwise. A comprehensive dictionary would be expected to list most translationally equivalent terms, though a few shortcomings are to be expected:

- Many bilingual dictionary entries will not be one-to-one. How to exploit these entries in a reasonable way when the element of interest is the unigram is an open question.
- Bilingual dictionaries may not contain domain-specific words and terms; such terms are often highly informative.
- Morphological variants are not typically listed in dictionaries, so without lemmatization of the text samples, some potentially informative open class terms will not be found in the dictionary.

3.2 Translation Models

It has been shown that performance on multilingual tasks involving word-level translational equivalence (e.g., cross-language information retrieval) can benefit from the use of automatically induced translation lexicons [ROL01] [NSID99].

For this purpose, I utilize Melamed's [Mel00] Model A. Model A assumes a generative process in which pairs of lexical items are generated in turn according to a distribution $\Pr(e, f)$, producing parallel bags of words. The parameters to this model are learned from a corpus of parallel text, aligned at approximately the sentence level.

For a pair of words e and f, the value of $\Pr(e, f)$ could as well be taken as an estimator t of $e \stackrel{t}{=} f$. This relies on the assumption that the probability of generating a pair of type (e, f) is directly related to the confidence that e and f are translationally equivalent. Alternately, any non-zero entry (e, f) in the translation model might be assigned $t(e, f) = 1$, and for any word pair (e, f) for which $\Pr(e, f) = 0$, $t(e, f) = 0$. This would create a boolean t function that might be merged with other such resources[4].

Using a learned statistical translation model helps to overcome the problems with manually-constructed dictionaries:

- Model A assumes generation of concepts, which are assumed to be in a one-one relationship with word-pairs. Therefore, for any $\Pr(e, f)$, e and f are both unigrams.
- The coverage of Model A is determined by the domain of the training corpus.
- Morphological variation is unknown to Model A; the distribution ranges over lexeme pairs.

It is important to highlight that translation models do require a resource: aligned parallel text. In general, this type of resource is more readily available than broad-domain electronic dictionaries (e.g., the Bible exists in electronic form in nearly every language for which any electronic resources are available), though its preparation is sometimes non-trivial.

One can imagine a framework in which a highly precise parallel text is obtained using the STRAND system [Res99], then used to train a translation model. This model could then be used to identify additional text samples that are translationally equivalent. By adding these samples to the parallel corpus in an iterative manner, retraining the translation model at each step, a larger parallel corpus might be extracted from the set of candidates. This sort of framework is the motivation for the evaluation task described in Section 5, though I did not undertake its exploration.

4 Cognates

The term "cognate" refers to a word in one language that is orthographically or phonetically similar to a semantically related word in another language. An example is the cognate pair English "calendar" and French "calendrier." By identifying cognates in a pair of text samples, we may hope to more accurately detect translational equivalence. The key advantage to cognates over dictionary and translation model[5] methods is that the detection

[4] Merging with a dictionary was the motivation for this approach, but experimental results in Section 7 show that this t function can outperform the weighted version.

[5] Melamed and Smith [MS00] have designed and implemented a generative translation model which includes cognate information.

of cognates need not rely on having previously seen the pair of words in translationally equivalent contexts prior to the detection task, or even having seen either word at all.

Previous work has shown the usefulness of cognates. Simard et al. [SFI92] used cognates to improve sentence alignment in parallel corpora. Knight and Graehl [KG97] explored transliteration of English characters to Japanese katakana using a generative source-channel model; the performance their system attained was better than that of human judges. Melamed [Mel95] used a string similarity measure based on character identity and string length to identify cognates; this was extended by Tiedemann [Tie99]. Work by Smith and Jahr [WS99] showed that cognate classifiers like Tiedemann's could be learned from a bilingual dictionary, and that these classifiers could improve translation models like that in the Candide system [BB94]. This approach is discussed and generalized here[6].

4.1 Background

String similarity metrics have proven useful in the extraction of cognates from text. This section describes generalizations on a method presented in Tiedemann [Tie99], which automatically constructed language-specific string matching functions from a set of known cognate pairs. My approach constructs such a function from a sentence-aligned parallel corpus, and it assumes very little linguistic similarity between the two languages[7].

4.1.1 LCSR and HSCR

Tiedemann [Tie99] explored ways in which language-independent versions of the Least Common Substring Ratio (LCSR, [Mel95]) could be derived from a set of known cognates in the language pair of interest. The LCSR is the ratio of the longest substring of the characters which are common to the two types in the pair (LCS)—this subset need not be consecutive—to the length (in characters) of the longer word in the pair.

The LCSR can be calculated using a dynamic programming technique. This is best illustrated by an example (based on [Tie99]). Consider the English word *seismic* (E) and its Czech cognate *seismický* (C). Note that the lengths, respectively, are seven characters and nine characters. Let $l_{i,j}$ denote the length of the least common substring (LCS) of the first i characters of *seismic* and the first j characters of *seismický*. For some character-matching function m, the dynamic programming equations are as follows:

$$\forall i, 0 \leq i \leq 7, l_{i,0} = 0$$
$$\forall j, 0 \leq j \leq 9, l_{0,j} = 0$$
$$l_{i,j} = \max[l_{i-1,j}, l_{i,j-1}, l_{i-1,j-1} + m(E_i, C_j)]$$

(The m function is boolean for the LCSR: it is 1 if the argument characters are the same and 0 if they are not.) The LCSR is then computed by dividing that value by $\max[|\textit{seismic}|, |\textit{seismický}|]$. The algorithm is illustrated in the following matrix, where the rows correspond to values of i and the columns to values of j:

[6] While this approach may appear to be over-kill for languages that use the same script in similar ways, like English and French (the languages it was tested on in Section 7.5), this technique was developed with an eye toward application to dissimilar orthographic systems, e.g., Arabic, Greek, Cyrillic, and Hangul. For language pairs like English and French, LCSR might yield acceptable (or even better) results.

[7] Development of the techniques presented here was carried out in part using computing resources at the University of Edinburgh, Edinburgh, Scotland.

		s	e	i	s	m	i	c	k	ý
	0	0	0	0	0	0	0	0	0	0
s	0	1	1	1	1	1	1	1	1	1
e	0	1	2	2	2	2	2	2	2	2
i	0	1	2	3	3	3	3	3	3	3
s	0	1	2	3	4	4	4	4	4	4
m	0	1	2	3	4	5	5	5	5	5
i	0	1	2	3	4	5	6	6	6	6
c	0	1	2	3	4	5	6	7	7	**7**

The LCS is 7; the LCSR is $\frac{7}{9}$. For languages that have similar orthographic systems, like Swedish and English, this is an effective way of recognizing cognates.

In [Tie99], three types of independent character matching functions were suggested for m, so that instead of computing the LCS length, the algorithm computes the highest score of correspondence (HSC). These independent matching functions were generated empirically based on a list of known cognates. Each function m was defined for each pair of units, with a unit being either a character or an n-gram of characters. The HSC ratio (HSCR) for two words is computed by dividing the HSC by the maximum length of the two words, in units.

HSCR is then taken to be a score of cognate-ness for a pair of words in two languages.

4.2 Modifications for this Application

Rather than a list of known cognate pairs (an unlikely resource), I used as a training set the non-zero entries in a statistical translation model (see Section 3.2) trained on aligned parallel bi-text. These pairs are weighted with scores (probabilities) in $[0, 1]$ that are taken to indicate confidence in translational equivalence. Unlike Tiedemann, I exploit these scores in estimating the matching function m. The details of the training algorithm for m are as follows.

4.2.1 Filters

Tiedemann [Tie99] utilized several filters on bilingual word pairs (found in a corpus) which might be scored for similarity. Two of these filters may be generalized so that minimal assumptions are made about the language pair in question.

The first filter is a minimal token length, which Tiedemann set to 4 for both languages. If a language like Japanese was being considered, where a single atomic character may contain several phonemes, this would be inappropriate when the other language is written in an alphabetic script. Preferably, the minimal length would be customized for each language. The remedy I used was to set the minimum at 4 for English (E below) and estimate a minimum for the other language L based on each language's average type length:

$$length_{min}(L) = \frac{\overline{typeLength_L}}{\overline{typeLength_E}} \cdot length_{min}(E) \tag{14}$$

The other constraint of interest is a minimal length difference ratio. Tiedemann [Tie99] argues that, because "cognates should be of comparable length," a ratio of the shorter string's length to the longer string's length should be required to above a certain value, which he set at 0.7. The approach here does the same, but normalizes each string's length by the average type length for the types in the string's language. The ratio of the normalized lengths is then required to be in the range $[\frac{7}{10}, \frac{10}{7}]$.

The score for any word pair which does not meet these two criteria is zero, and any word pair in the training list which does not meet the criteria is not considered.

4.2.2 Counting character cooccurrences

Counts of cooccurring characters from all of the training pairs are then taken. [Tie99] used an "estimated position" value to determine which characters should be counted, for example:

$$EP(E_i) = round\left(i \cdot \frac{|E|}{|C|}\right) \tag{15}$$

This assumes a linear relationship between the characters in the strings, but may be overly strong when the training example is like the *seismic/seismický* example above, in which the term in one language contains an affix. Instead of limiting the character pairs that were counted, my approach biases in favor of those characters which are in similar positions relative to the beginning and end of the string:

$$count(E_i, C_j) = 1 - \left\lfloor \left| \frac{i - \frac{1}{2}}{length(E)} - \frac{j - \frac{1}{2}}{length(C)} \right| \right\rfloor^2 \tag{16}$$

i and j are decreased by $\frac{1}{2}$ so as to place the characters between the integers $\{0, 1, \ldots n\}$ for an n-length type; this means that the relative positions of all characters are influenced by the length of the string containing them (i.e., the first character is not at the absolute beginning of the string and the last is not at the absolute end). Hence in my method, for a given training word pair, all pairs in the cross-product of characters are counted, but the counts are weighted by the relative nearness of the positions.

Each count is further weighted by the score (probability) of the word pair from which it comes. Therefore, low-scoring word pairs from the translation model do not impact the character matching function as greatly as high-scoring word pairs.

4.2.3 Character classes

Tiedemann [Tie99] used a further restriction on counting in this particular method: vowels were counted only with vowels and consonants only with consonants. Each character was assigned to one of these disjunct sets. While this was quite helpful for Swedish-English, it would be inappropriate for two languages which use the letters differently (e.g., w is a vowel in Welsh, but a consonant in English), or which use different letters (e.g., Russian-English). Further, in some scenarios, the script may be entirely unknown, so that even if such classes (and bilingual correspondences between classes) exist, the information is unavailable. My approach assumes no prior knowledge of character classes which might help in the task of building a character matching function.

4.2.4 Matching function m

Like Tiedemann [Tie99], this approach computes a Dice score for each character pair after collecting the counts. The equation is given below. This score is then used as the m function.

$$m(x, y) = Dice(x, y) = \frac{2c_{x,y}}{c_x + c_y} \tag{17}$$

4.2.5 Cognateness as t

The HSCR may ultimately be used as a t function, i.e., $t(e, f) = \mathrm{HSCR}(e, f)$. Low HSCR scores, however, are uninformative, since the score is really intended to separate cognate pairs from the vast majority of word pairs that are not cognates. For this reason, and because every pair e, f has an HSCR score (and for e and f above the minimum length, the score is non-zero), it makes sense to apply a threshold τ, so that $t(e, f) = \mathrm{HSCR}(e, f)$ if $\mathrm{HSCR}(e, f) \geq \tau$ and 0 otherwise. Similarly, the value of the HSCR could be ignored altogether to create a boolean t function: $t(e, f) = 1$ if $\mathrm{HSCR}(e, f) \geq \tau$ and 0 otherwise.

5 Evaluation Task

I propose the following task to evaluate the usefulness of a given t function in detecting translational equivalence. Begin with a segmented corpus $\mathcal{C}_1$ in language $\mathcal{L}_1$ and another segmented corpus $\mathcal{C}_2$ in language $\mathcal{L}_2$. Let both $\mathcal{C}_1$ and $\mathcal{C}_2$ consist of n segments. Let k of the segments in $\mathcal{C}_1$ be known to be translationally equivalent respectively to k segments in $\mathcal{C}_2$. The remaining $n - k$ segments in $\mathcal{C}_1$ and $\mathcal{C}_2$ are noise and assumed not to be translationally equivalent to any segments in the opposite corpus. It is assumed, therefore, that every segment in either corpus has either one or zero translationally equivalent elements in the other corpus.

Using the t function as an estimate of shingle translational equivalence ($\stackrel{t}{=}$), a resemblance score may be computed for each of the $n \times n$ potentially translationally equivalent pairs in $\mathcal{C}_1 \times \mathcal{C}_2$. Resemblance for this purpose exploits the set-theoretic functions defined in Section 2:

$$r(E, F) = \frac{\left| S(E) \stackrel{t}{\cap} S(F) \right|}{\left| S(E) \stackrel{t}{\cup} S(F) \right|} \tag{18}$$

where $E \in \mathcal{C}_1, F \in \mathcal{C}_2$ and $S(E), S(F)$ are the sets of unigram types (n-length shingles) present in E and F, respectively.

The problem is now reducible to the maximum weighted bipartite matching problem [Mel00]: given a bipartite graph $G(V, E)$ (let $V = \mathcal{C}_1 + \mathcal{C}_2$) with weighted edges (let $w(E, F) = r(E, F)$ for all E, F), find a matching M between the bipartite sets that maximizes $\sum_{(E,F) \in M} w(E, F)$. The lowest currently known upper bound on the computational complexity of this problem is $O(ve + v^2 \log v)$ for v vertices and e edges [AMO93]. This reduces to $O(n^3)$ for this task, since all $n \times n$ pairs are scored, creating n^2 edges.

Using the maximum weighted bipartite matching algorithm is reasonable when the sets of text samples are small. I report results using this algorithm (see discussion of precision and recall below).

Following Melamed [Mel00], I also utilize a greedy approximation algorithm to maximum weighted bipartite matching called competitive linking. This technique operates in the following manner. Begin with the set of candidate pairs $\mathcal{C}_1 \times \mathcal{C}_2$. At each step, select the highest-scored pair $(E \in \mathcal{C}_1, F \in \mathcal{C}_2)$[8]. Mark (E, F) as a translationally equivalent pair and remove it from the set of candidates, adding it to $\mathcal{T}$, a set of pairs believed to be translationally equivalent. Continue until some stopping condition is met or no more links

[8] Melamed and Smith (in progress) describe a random tie-breaking method when multiple candidates have the same score, which I use.

are possible. When the scored pairs are maintained in a priority queue implemented by a heap, this algorithm runs in $O(n \log n)$; it is suitable for scenarios where n is large or only the top pairs are desired.

At each step in the competitive linking algorithm, precision and recall scores may be calculated for the set of marked pairs (E, F). Let $\mathcal{T}_c$ be the subset of marked pairs $(E, F) \in \mathcal{T}$ where E and F are known in advance to be translationally equivalent.

$$\text{precision} = \frac{|\mathcal{T}_c|}{|\mathcal{T}|} \quad (19)$$

$$\text{recall} = \frac{|\mathcal{T}_c|}{n} \quad (20)$$

I report precision and recall at all stages of completion of the algorithm using precision-recall plots.

6 Shrinking the Search Space

Scoring a text sample pair (E, F) for resemblance requires $O(|E| \cdot |F|)$ steps. Exhaustive pairwise scoring, then, is a highly expensive endeavor. Chen and Nie [CN00] note that pairs (E, F) that are highly disparate in length are unlikely to be translational pairs. In other words, a positive correlation between the lengths of E and F where E and F are translationally equivalent is to be expected.

If this is the case, the space of pairs that to be scored may be significantly narrowed by filtering out pairs where, e.g., E is relatively short and F is relatively long, or vice versa.

I trained a linear regression model for sentences in an aligned parallel corpus of Hong Kong Laws [Ma00] in English and Chinese. (Information about the training and test corpora for this experiment are shown in Table 2.)

Woods et al. [WFH86] describe how to compute a confidence interval for an independent variable value within such a model. For example, we might like to know with some level of confidence the range of values for the length of the Chinese translation for a given English sentence. Let p denote the probability that the Chinese translation will not be within that range.

I applied the confidence interval as a filter to the set of candidates for resemblance scoring. Given E (an English segment), the set of candidates involving E are only those pairs (E, F) where $|F|$ is in the confidence interval for $|E|$, for some p. A higher value of p yields a more strict filter, eliminating pairs whose lengths are not close to the regression line; the benefit of such a filter is a significantly reduced search space. A lower value of p, however, is more conservative, sacrificing search space reduction in favor of potentially higher recall. Such a p eliminates less pairs, reducing the chances that some translationally-equivalent pairs will be ruled out before resemblance scoring.

The size of the unfiltered search space (i.e., the number of resemblance scores that must be computed) for the test corpus is $191^2 = 36,481$. Figure 2 shows the size of the search space remaining after this length filter is applied for varying values of p. Figure 3 shows the number of translationally equivalent pairs remaining in the filtered search space for varying values of p.

Interestingly, as p increases, the search space size decays approximately exponentially, while the number of correct pairs eliminated increases only linearly. This is encouraging; it shows that computational savings are to be had without necessarily affecting performance in direct relation to the savings.

	size (segments)	English tokens	English types	Chinese tokens	Chinese types	mean English segment size	mean Chinese segment size
training	5,622	204,388	5,612	208,163	4,517	32.44	33.04
test	191	6,192	1,026	6,326	951	30.96	31.63

Table 2: Hong Kong Laws parallel corpus [Ma00]. The English text was tokenized using a script included in the Egypt distribution [WS99], written by Dan Melamed and Yaser Al-Onaizan. The Chinese text was segmented by Clara Cabezas using ch_seg [Chen95] [LO94].

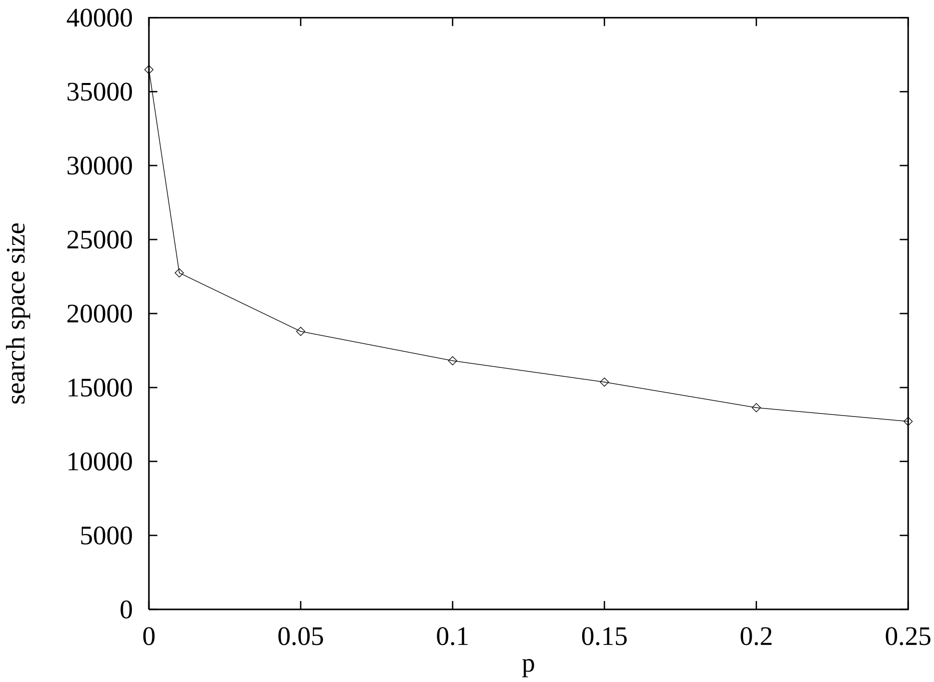

Figure 2: The benefit of length filter is a reduction in search space.

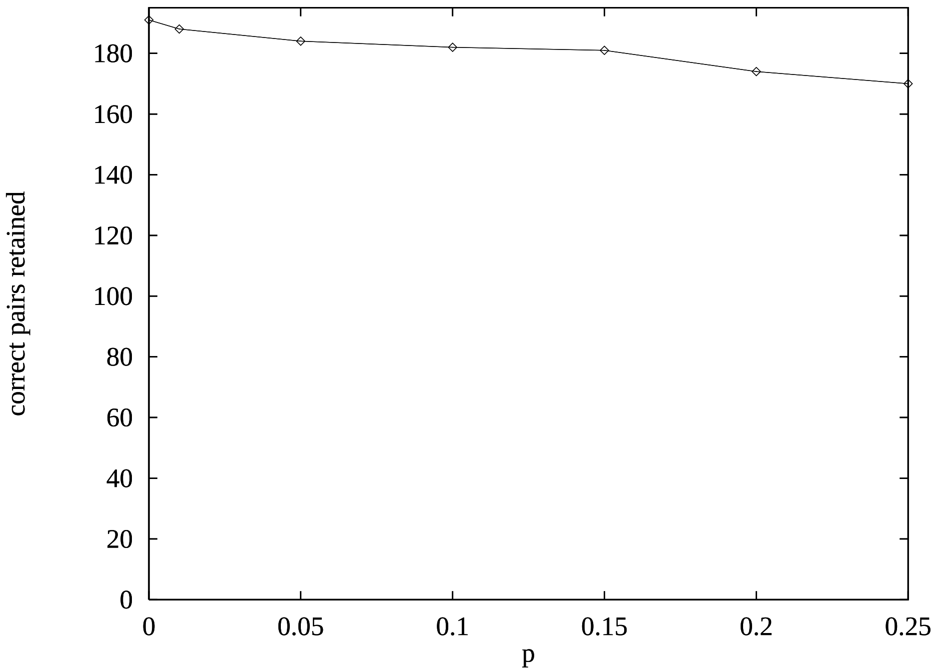

Figure 3: The length filter eliminates few correct pairs.

7 Experimental Results

A number of experiments with the methods described were carried out. First, comparisons were made among different t functions, most notably between functions constructed from electronic dictionaries and functions derived from statistical translation models. The actual effects of the length filter (Section 6) are shown. Noise, in the form of additional text samples without their respective translations, was added to the test corpus and evaluation carried out. The usefulness of cognates as a supplement to other t functions was tested next. Finally, preliminary experiments were carried out on document pair candidates discovered by the STRAND system.

7.1 Human vs. Machine: English-Chinese

In this experiment, several different t functions were compared. Three of them were defined as boolean functions using a bilingual dictionary as an information source. Another was fuzzy (i.e., ranges over $[0, 1]$), defined using a statistical translation model learned from parallel text [Mel00]. The training and test corpora used are those described in Table 2.

7.1.1 Electronic bilingual dictionary

Starting with two directional lexicons[9], three different t functions were induced. Recall that the t function is defined (for this discussion) over pairs of unigram types. This dictionary, in its original form, contained many entries involving more than one English word and/or

[9]Thanks to Clara Cabezas and Gina Levow for help in procuring these resources.

more than one Chinese word. The three approaches used here are the most obvious ways of inducing the function from a dictionary.

To begin, the Chinese-to-English dictionary[10] contained 341,187 entries, and the English-to-Chinese dictionary[11] contained 394,969 entries.

One-to-one entries (OO)

1. All entries involving more than one English word or more than one Chinese word were removed. The Chinese-to-English dictionary now contained 232,518 entries. The English-to-Chinese dictionary now contained 288,709 entries.

2. The two lexicons were merged into a single listing. 291,454 pairs were in the union of the two dictionaries.

3. Remove any entries including words not found in the corpora (this step was carried out for efficiency reasons; it could not have affected performance in any way). The final dictionary contained 9,263 entries.

The t_{OO} function is:

$$t_{\mathrm{OO}}(e, f) \equiv (e, f) \in \text{Dictionary}$$

Cross-product of non-one-to-one entries (CP)

1. All non-one-to-one entries from both dictionaries were collected; the total was 214,947 entries.

2. Any entries containing (any) words not present in the corpus were removed. The remaining set contained 6,060 entries.

3. The entries were expanded into multiple entries by taking the cross-product of English words with Chinese words. All elements in the cross-product were included; the result was 15,549 entries. 5,910 of these entries were not present in the OO dictionary.

4. These entries were merged with the OO dictionary. The new dictionary totaled 15,173 one-to-one entries.

The t_{CP} function is:

$$t_{\mathrm{CP}}(e, f) \equiv (...e..., ...f...) \in \text{Dictionary}$$

Stoplisted cross-product of non-one-to-one entries (SLCP) The items added to OO in the cross-product process have a high potential for noise, since many non-one-to-one dictionary entries contain function words that do not hold an equivalence relation to corresponding words on the other side of the entry. (An example of this is ‘to’ in English infinitives.) The noise might degrade the dictionary’s performance; in order to lessen this effect, the following dictionary preprocessing was applied:

- All non-one-to-one entries from both dictionaries were collected; the total was 214,947 entries.

[10] The creation of this dictionary is described in [LOC00]. Thanks to Gina Levow, Doug Oard, and Clara Cabezas for allowing its use.

[11] The creation of this dictionary is described in [WS00]. Thanks to Gina Levow for offering it for use here.

- Any entries containing words not present in the corpus were removed. The remaining set contained 6,060 entries.
- A stoplist filter was applied to the set. The English stoplist contained 238 common closed-class words; the Chinese stoplist contained 188 function words[12]. After the filter was applied, any entries where the English or Chinese side was empty were removed. The result was 5,555 entries.
- The non-one-to-one entries were expanded into multiple entries by taking the cross-product of English words with Chinese words. All elements in the cross-product were included; the result was 8,285 entries. 2,842 of these entries were not present in the OO dictionary.
- These entries were merged with the OO dictionary. The new dictionary totaled 12,105 one-to-one entries.

Let $\sigma(x)$ be a boolean predicate taking the value 'true' if x is present on its language's stoplist. The t_{SLCP} function is:

$$t(e, f) \equiv (\neg\sigma(e) \wedge \neg\sigma(f) \wedge (...e..., ...f...) \in \text{Dictionary}) \vee (e, f) \in \text{Dictionary}$$

7.1.2 Method A translation model

Using Method A [Mel00], a symmetric translation model was induced on the Hong Kong Laws (see Table 2) training corpus. One change was made to the training method described in [Mel00] for the model: rather than use competitive linking, I used the maximum weighted bipartite matching algorithm implemented in the Library of Efficient Data Types (LEDA) [MN99]. This provides a closer approximation to the maximum likelihood estimation for Method A. A Method A distribution contains the probability of generating a token in one language with a "null-word" in the other; a null-word is simply an empty element not observable in the string. After the model was trained, all entries involving a null-word were removed. The translation model used contained 3,767 non-zero entries of the form $\Pr(e, f)$ where e is an English word and f a Chinese word. The t_{A} function is:

$$t_{\mathrm{A}}(e, f) = \Pr(e, f)$$

7.1.3 Results

The evaluation method described in Section 5 was applied to all four t functions on the test corpus of 191 segment pairs. For this experiment, $k = n$; that is, there is no added noise. The results are shown in Table 3.

Figure 4 shows precision-recall plots for the entire competitive linking process on each of these t functions as well as the maximum weighted bipartite matching performance for each. Early iterations correspond to the left side of the plots (note that recall can never decrease, so the algorithm will only move leftward across the plot). The degradation in precision as competitive linking proceeds is due to erroneous links made between text samples. This is expected; each successive competitive linking step links the next-highest scored pair, so the confidence of the decision made at each iteration diminishes. The increase in recall, of course, is due to correct matches accumulated.

[12] Thanks to Dan Melamed for the use of these stoplists.

t function	competitive linking precision	recall	MWB matching precision	recall
OO dictionary	0.5236	0.5236	0.6607	0.5812
CP dictionary	0.4660	0.4660	0.6176	0.5497
SLCP dictionary	0.5550	0.5550	0.6845	0.6021
Method A translation model	0.7016	0.7016	0.8372	0.7539

Table 3: The translation model outperforms the dictionary, and maximum weighted bipartite matching outperforms competitive linking.

The translation model clearly outperforms all three dictionaries, substantially. Note that the SLCP dictionary achieved higher recall and precision than the other two (OO and CP). In addition, maximum weighted bipartite matching offers much better results than competitive linking in all cases.

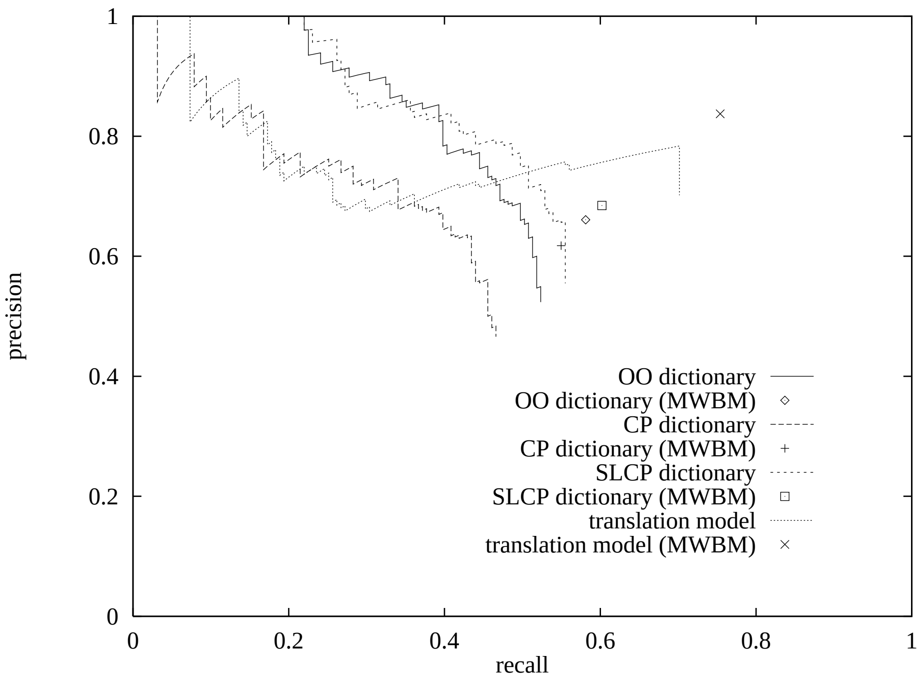

Figure 4: Precision and recall of four t functions through the competitive linking process and under maximum weighted bipartite matching.

7.1.4 Further Exploration

The following discussion seeks to investigate why the translation model was more successful than the dictionaries. Simultaneously, the difference in performance of the two matching algorithms—competitive linking and maximum weighted bipartite matching—is considered.

Preprocessing effects Degradation of the dictionary due to preprocessing, while possible, is unlikely. The most obvious uses of the dictionary were all applied. Both CP and

SLCP significantly increased the size of the dictionary used (by 64% in the first case, by 31% in the second). Yet neither offered benefits that pushed the performance of this t function to the level of the statistical translation model's performance. It is of course possible that more clever, less obvious ways to exploit a dictionary to this end exist.

Non-boolean scores and ties One possible reason for degradation under competitive linking may be traceable to the boolean nature of a t function derived from a dictionary. Because the entries are unweighted, the values that the r score may take on are limited to rational numbers with denominators $\leq q$, where q is equal to the maximum length of an English text sample plus the maximum length of a Chinese text sample. If the set of possible r values is finite, then many tied scores are to be expected. When multiple pairings are tied, the competitive linking algorithm chooses from them at random until no more linkings may be made. Therefore, if a correct pairing (e, f) is tied with many other pairings involving e or f, the probability of correctly linking e with f shrinks. The number of unique scores observed in the set of all 36,481 pairwise r-scorings is shown in Table 4. (Note that precision and recall are given at the completion of competitive linking; because there was no noise and each pair was linked to exactly one other pair, the total number of linkings was 191. As a result, final precision and final recall were equivalent.)

dictionary	number of unique r values	CL precision, recall
OO	779	0.5236
SLCP	808	0.5550
CP	939	0.4660

Table 4: Tied scores probably do not account for degraded performance on competitive linking.

These data are inconsistent with the hypothesis that low score-variability resulting in bad tie-breaking is a factor in the performance of competitive linking. Competitive linking performance does not correlate with the number of distinct scores. It is left to conclude that competitive linking is an approximation that simply fails to fails the nuances discovered by maximum weighted bipartite matching.

The next question to be asked is whether the weights in the t function derived from the statistical translation model play a role in its success in this framework.

Method A translation model without weights The 3,767 non-zero entries in the translation model distribution were stripped of their probability values, giving an unweighted translation lexicon[13]. This was used as a boolean t function on the English-Chinese experimental task:

$$t_{\mathrm{A}_{\mathrm{unweighted}}}(e, f) = 1 \text{ if } \Pr(e, f) \geq 0, \text{ else } 0$$

Results are shown in Table 5; precision-recall plots for the weighted and unweighted t functions are shown in Figure 5.

[13]It is worth noting that applying a threshold to the translation model entries before removing weights, so that low-probability pairs are not included in the set of translationally-equivalent word pairs, was not attempted, though it might offer some benefit.

t function	competitive linking precision	competitive linking recall	MWB matching precision	MWB matching recall
Method A translation model	0.7016	0.7016	0.8372	0.7539
unweighted translation model	0.8115	0.8115	0.9075	0.8220

Table 5: Performance of the translation model with and without weights.

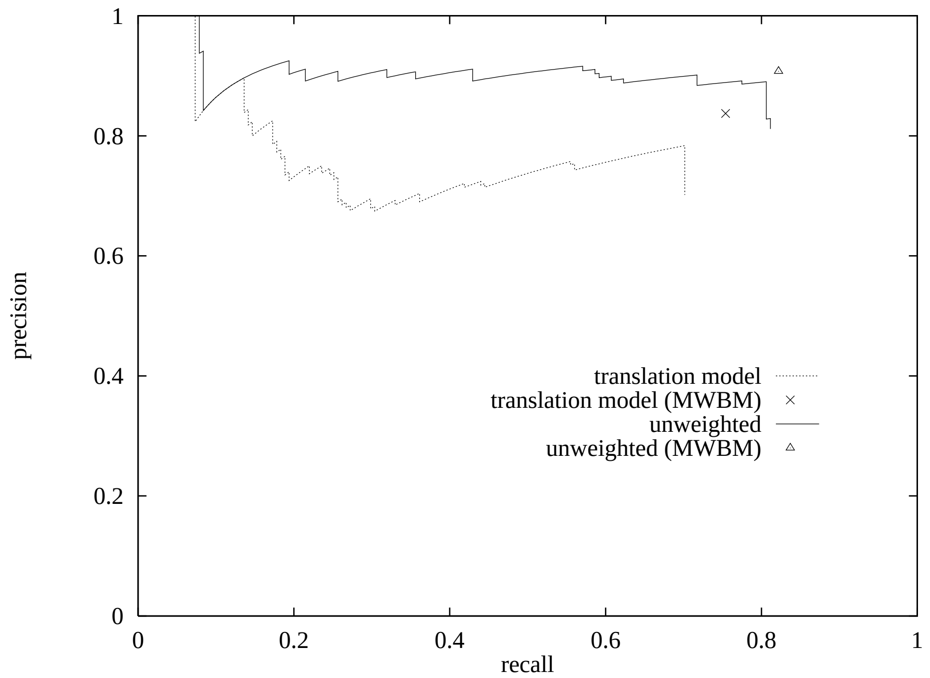

Figure 5: Precision and recall of the translation model with and without weights through the competitive linking process and under maximum weighted bipartite matching.

This t function is without a doubt noisy. Many of the entries learned from the corpus are the result of frequent collocation rather than translational equivalence[14]. Yet removing weights improves performance under maximum weighted bipartite matching and under competitive linking. It is unclear why this might be the case, though it might be conjectured that the weights in the translation model result in "undue caution": when computing the resemblance score, a word pair's t value might be quite low, though the word pair is in fact translationally equivalent.

These experiments suggest that the translation model detected, in the training corpus, a significant number of term equivalences that were not present in the dictionary. In fact, the intersection of listed entries (weights aside) of the OO dictionary with the translation model is only 485 entries. The intersection with the CP dictionary contained 535 entries, and the intersection with the SLCP dictionary contained 533 entries.

It is therefore likely that the translation model's performance is attributable in part to domain specificity. Of the words in the corpus, 41% of English types (63% of tokens) and 72% of Chinese types (87% of tokens) were listed in at least one dictionary entry (in the SLCP dictionary). Yet the translation model is in stark contrast: of the words in the corpus, 55% of English types, accounting for a mere 12% of tokens, were listed in at least one translation model probability. The same is true of only 61% of Chinese types, accounting for only 11% of tokens. This shows that the translation model, unsurprisingly, induced translational equivalence relations between highly informative words not found in the dictionary.

Merge of dictionary and unweighted translation model Finally, because the intersection of the dictionary and the translation model entries was so small, and because the weights in the translation model seem not to play a major role in performance, I merged the best-performing electronic dictionary (SLCP) with the unweighted translation model to create a t function that might have the advantages of each. Detection results on the test corpus are shown in Table 6; precision-recall plots are shown in Figure 6.

	competitive linking		MWB matching	
t function	precision	recall	precision	recall
SLCP dictionary	0.5550	0.5550	0.6845	0.6021
unweighted translation model	0.8115	0.8115	0.9075	0.8220
union of translation model and SLCP	0.8220	0.8220	0.9048	0.8953

Table 6: Performance of the best-performing preprocessed dictionary merged with the unweighted translation model.

These results are exactly what one might hope to find. The high precision of the unweighted translation model t function is achieved, and recall reaches almost 90%, attributable to additional coverage from the dictionary.

7.2 Human vs. Machine: English-Spanish

A similar experiment was carried out on English-Spanish data to verify that these findings might be generalizable. Table 7 shows data about the corpus used.

[14] For this task, however, this information may be useful nonetheless, since collocated, translationally inequivalent words may tend to collocate frequently.

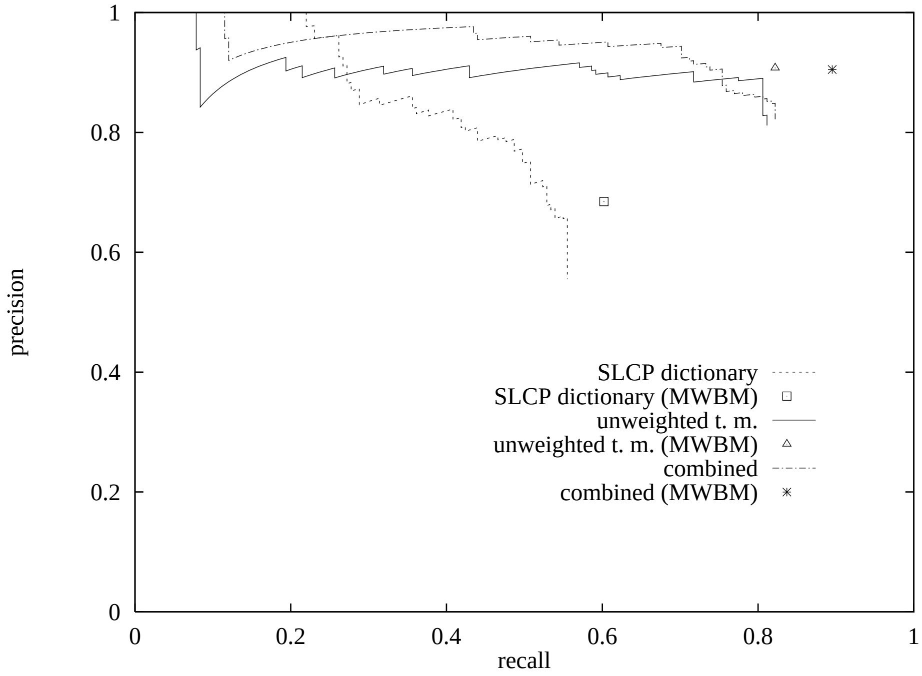

Figure 6: Precision and recall of the translation model, dictionary, and their union through the competitive linking process and under maximum weighted bipartite matching.

	size (segments)	English tokens	English types	Spanish tokens	Spanish types	mean English segment size	mean Spanish segment size
training	4,695	141,125	9,872	158,094	11,484	29.43	32.97
test	200	6,637	1,751	7,549	1,795	33.19	37.75

Table 7: English-Spanish parallel corpus. The original text from which this was taken is a portion of United Nations proceedings [Graff94]. The alignment was done by Clara Cabezas using MXTERMINATOR [RR97]. The English text was tokenized using a script included in the Egypt distribution [WS99], written by Dan Melamed and Yaser Al-Onaizan. The Spanish text was tokenized using a program written by Nizar Habash and Bonnie Dorr, with kind permission of the authors.

7.2.1 Electronic bilingual dictionary

A single bilingual lexicon[15] consisting of 58,413 one-to-one entries from English to Spanish was used. Of those entries, 6,053 contained words found in the corpora. The t_{Dict} function is simply:

$$t_{\text{Dict}}(e, f) \equiv (e, f) \in \text{Dictionary}$$

7.2.2 Method A translation model

As in the Chinese/English experiment, a symmetric translation model (Model A [Mel00]) was induced on the training corpus, using maximum weighted bipartite matching instead of competitive linking. (See discussion in 7.1.2.) Entries involving a null-word were removed, and the resulting model contained 10,227 non-zero entries. The t_{A} function is defined as:

$$t_{\text{A}}(e, f) = \Pr(e, f)$$

As in the English-Chinese experiment, an unweighted (boolean) version was also computed, such that $t_{\text{A}_{\text{unweighted}}}(e, f) = 1$ if $\Pr(e, f) \geq 0$ and 0 otherwise.

7.2.3 Combined t

The intersection of the dictionary with the translation model was only 1,449 entries. The two were therefore merged to create a new t function such that:

$$t_{\text{combined}}(e, f) \equiv t_{\text{Dict}}(e, f) \vee t_{\text{A}_{\text{unweighted}}}(e, f)$$

This combined function held true for 14,831 pairs relevant to the corpus.

7.2.4 Results

The evaluation method described in Section 5 was applied to all four t functions on the test corpus of 200 segment pairs. For this experiment, $k = n$; that is, there is no added noise. Table 8 compares precision and recall results from the four functions; precision-recall plots are shown in Figure 7.

	competitive linking		MWB matching	
t function	precision	recall	precision	recall
Dictionary	0.9050	0.9050	0.9282	0.9050
Method A translation model	0.7500	0.7500	0.8971	0.7850
unweighted translation model	0.8200	0.8200	0.9600	0.8400
union of dictionary and translation model	0.9500	0.9500	0.9442	0.9300

Table 8: Some dictionaries perform well as t functions; the union of a dictionary and a translation model achieves even better results.

These results are highly similar to the English-Chinese results, except that the dictionary outperforms the translation model. The most plausible reason for this is simply that the

[15]Thanks to Gina Levow for providing this resource. The dictionary consists of unigram English terms from the Brown Corpus [FK82] and their equivalents returned by the LOGOS machine translation system [Logos]. Some heuristics were employed in the process to force certain inflections on nouns, adjectives, and verbs.

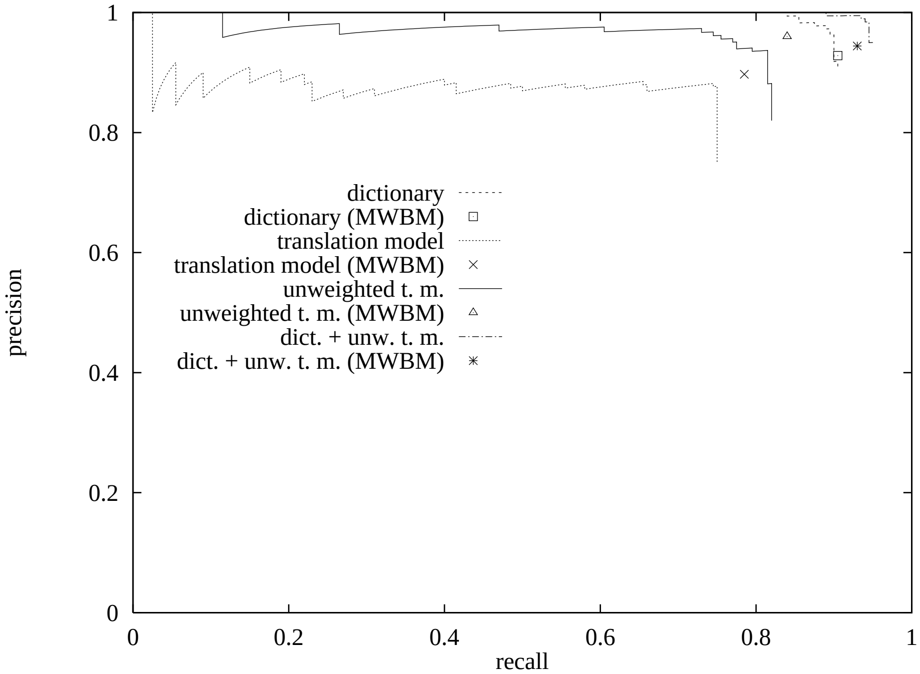

Figure 7: Precision and recall of four t functions through the competitive linking process and under maximum weighted bipartite matching.

English-Spanish dictionary used was either well-suited to the genre of this test corpus or had more extensive coverage than the dictionaries used in the English-Chinese experiments.

These experiments on English-Chinese and English-Spanish corpora show that translational equivalence is detectable in text, and that statistical models offer a means to carry out that detection. This is a useful result, as parallel text (required for translation model training) is a less expensive resource than bilingual dictionaries, which, though in some cases may offer even higher levels of performance on the detection task, are variable in their effectiveness.

7.3 Length Filter

Using three of the English-Chinese t functions from the experiment in Section 7.1, I applied the length filter from Section 6 at various values of p. This technique reduces the amount of time required for scoring text sample pairs (computing the resemblance score r) by eliminating pairs where the length of the Chinese segment is outside some confidence interval for the length of the English segment, given a linear regression model. The linear regression model is learned from the same training corpus as the translation model.

Table 9 shows precision and recall after application of the length filter at various values of p for the SLCP dictionary, and Figure 8 shows precision and recall throughout competitive linking and under maximum weighted bipartite matching. Table 10 and Figure 9 show the same for the Method A translation model (unweighted). Table 11 and Figure 10 show the same for the union of the SLCP dictionary and the unweighted translation model. The values of p chosen are small, since the benefit (search space size reduction) levels out as p increases (see Figure 2).

It is important to note that, because the filters eliminate some correct pairs, recall is bounded by the number of correct pairs that pass the filter. For example, for the $p = 0.05$ length filter, 184 correct pairs pass the filter, so recall is limited to $188/191 = 0.9843$. Similarly, the limits for $p = 0.05, p = 0.1, p = 0.15$ are 0.9634, 0.9529, and 0.9476, respectively. For this reason, competitive linking results are shown for these experiments at that critical iteration step; for each filter, the step denoted as "max" is the last one for which precision may potentially be equal to 1 (i.e., 188 for $p = 0.01$, etc.). Beyond that iteration, precision cannot increase. It is recognized that a user of this technique cannot know the number of correct pairs eliminated by the length filter, though under competitive linking, the maximum precision score is observable.

	competitive linking					
	max		completion		MWB matching	
length filter	precision	recall	precision	recall	precision	recall
none	0.5550	0.5550	0.5550	0.5550	0.6845	0.6021
$p = 0.01$	0.5638	0.5550	0.5550	0.5550	0.6864	0.6073
$p = 0.05$	0.5815	0.5602	0.5654	0.5654	0.6923	0.6126
$p = 0.1$	0.5934	0.5654	0.5707	0.5707	0.6568	0.5812
$p = 0.15$	0.5856	0.5550	0.5550	0.5550	0.6627	0.5864

Table 9: Effects of the length filter on translation detection using a bilingual dictionary (SLCP).

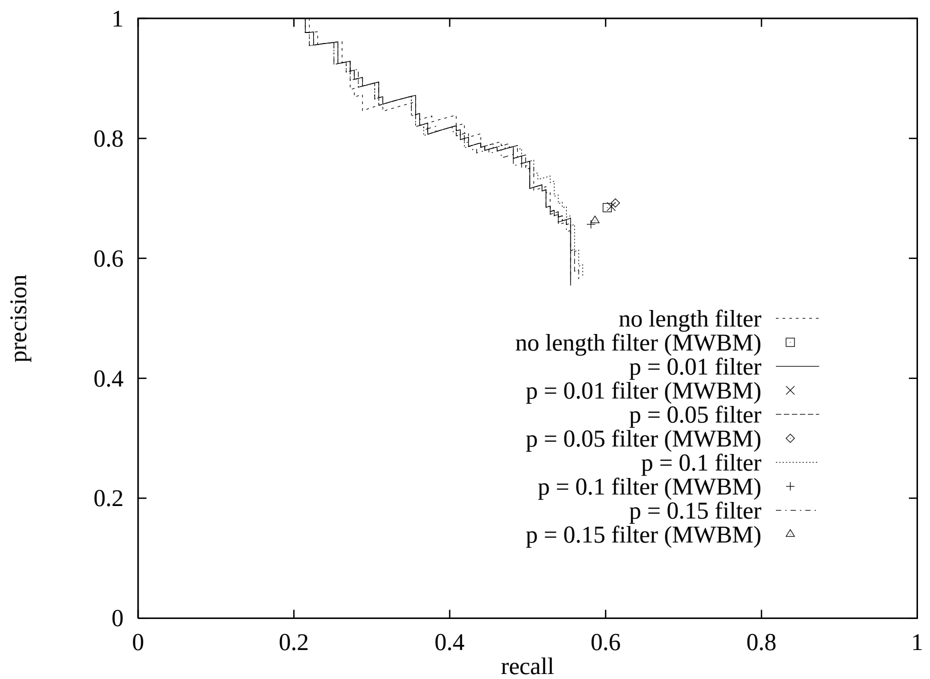

Figure 8: Precision and recall of the SLCP dictionary are hardly affected by length filtering.

Under both matching algorithms, the filter did not affect performance of the SLCP dictionary greatly. In many cases performance improved, and in cases where it was degraded, the decrease was less than 0.03 for both precision and recall.

	competitive linking					
	max		completion		MWB matching	
length filter	precision	recall	precision	recall	precision	recall
none	0.7016	0.7016	0.7016	0.7016	0.8372	0.7539
$p = 0.01$	0.8032	0.7906	0.7906	0.7906	0.9017	0.8168
$p = 0.05$	0.8043	0.7749	0.7749	0.7749	0.8830	0.7906
$p = 0.1$	0.8242	0.7853	0.7853	0.7853	0.8772	0.7853
$p = 0.15$	0.8232	0.7801	0.7801	0.7801	0.8764	0.7801

Table 10: Effects of the length filter on translation detection using an unweighted statistical translation model.

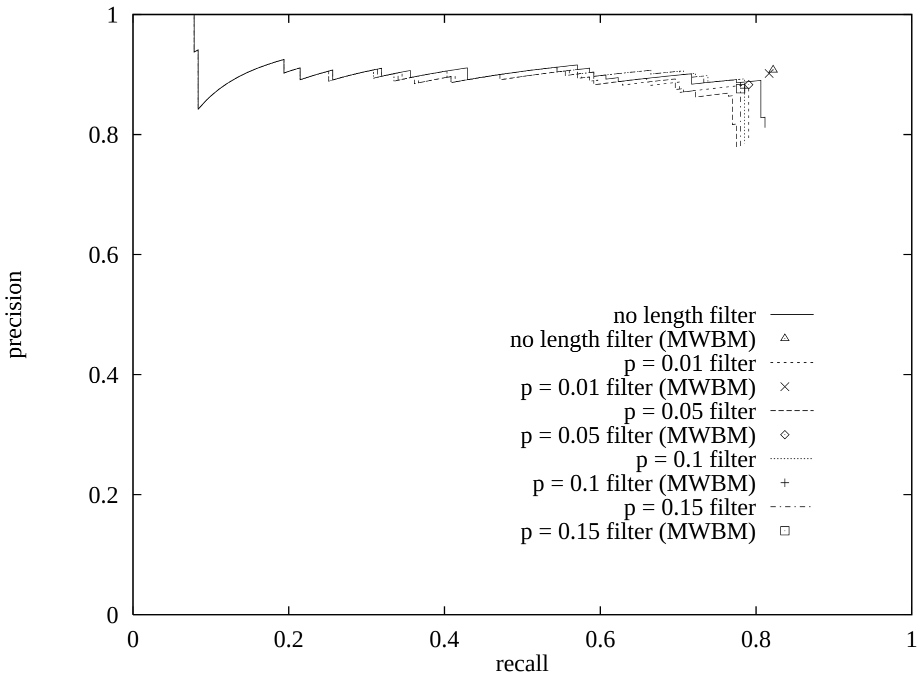

Figure 9: Precision and recall of the unweighted translation model are hardly affected by length filtering.

Performance of the unweighted translation model was improved under both matching algorithms for all filters. The results thus far suggest that the length filter not only offers a computational savings, but it also guides the matching process by eliminating pairs unlikely to be linked.

The merged t function's performance did not experience the same benefits through application of a length filter. Some slight increase in precision was to be had (at very slight and unsurprising loss in recall) under competitive linking.

It is clear that reducing the search space (and therefore run time) using a length filter does *not* greatly diminish performance when the text samples are sentence-sized, particularly under competitive linking. In fact, under some conditions it improves performance. It

length filter	competitive linking max precision	competitive linking max recall	competitive linking completion precision	competitive linking completion recall	MWB matching precision	MWB matching recall
none	0.8220	0.8220	0.8220	0.8220	0.9048	0.8953
$p = 0.01$	0.8298	0.8168	0.8168	0.8168	0.8783	0.8691
$p = 0.05$	0.8370	0.8063	0.8370	0.8370	0.8670	0.8534
$p = 0.1$	0.8407	0.8010	0.8115	0.8115	0.8457	0.8325
$p = 0.15$	0.8398	0.7958	0.8063	0.8063	0.8670	0.8534

Table 11: Effects of the length filter on translation detection using an unweighted statistical translation model merged with the SLCP dictionary.

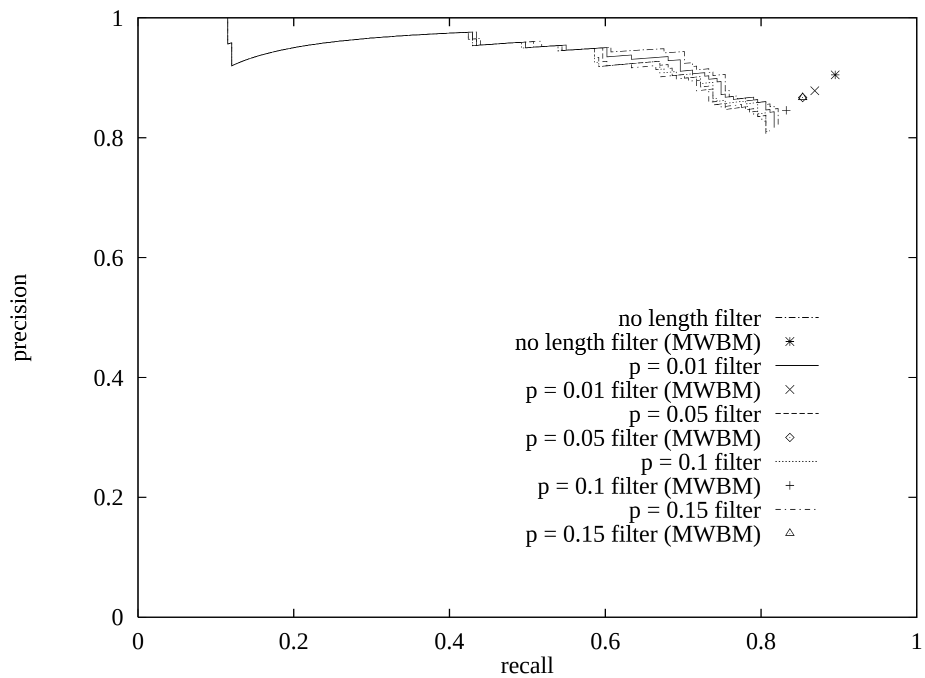

Figure 10: Precision and recall of the merged t function are hardly affected by length filtering.

is concluded that this is a useful technique allowing for faster completion of the detection task.

7.4 Robustness to Candidate Noise

It is unrealistic to suppose that the task of selecting translation pairs from a set of candidates will typically involve sets of text samples in two languages where every sample is known to have exactly one translationally equivalent partner in the opposite set. It is to be expected that there will be some noise, e.g., text samples which are not part of translation pairs. These make the task more difficult by increasing the number of candidates quadratically and possibly affecting both precision and recall.

In order to test the robustness of this method to such noise, I randomly selected 573 English text samples and 573 Chinese text samples from the same Hong Kong Laws corpus that the test and training corpus came from. These samples were of comparable length (34.64 average sample length for English, 36.73 for Chinese) and genre to the corpus in Table 2. The English samples were taken from a different region of the corpus than the Chinese samples to avoid choosing samples that might be translationally equivalent.

Incrementally, the noise samples were added to the test corpus of $k = 191$ pairs[16], and the resulting candidate pairs were scored. Maximum weighted bipartite matching is not a useful algorithm for this task because it seeks to link as many text sample pairs as possible to maximize the sum of the scores. Doing this will inevitably hurt precision; if $k = 191$, then generating more than n links will affect precision for the worse.

In order to lessen the computational cost, a length filter with $p = 0.1$ was applied. Table 12 shows the reduction in the search space through the application of this filter. This filter was selected because it offered, overall, the greatest improvement in performance under competitive linking (see results of the length filter experiments in Section 7.3).

n ($n - k$)	number of candidates	number of candidates after filter
287 (96)	82,369	37,573
382 (191)	145,924	66,173
573 (382)	328,329	144,950
764 (573)	583,696	246,431

Table 12: Reduction in candidates via length filter.

Three t functions were used: the SLCP dictionary, the Method A translation model (unweighted), and the union of the two. Results with the dictionary are shown in Table 13, and Figures 11 and 12 show precision and recall (respectively) throughout the competitive linking process. Precision and recall are separated and each is plotted against competitive linking iteration to highlight the importance of stopping the competitive linking process at the appropriate time. Analogous results with the translation model are shown in Table 14 and Figures 13–14, and results for the union are shown in Table 15 and Figures 15–16.

Note that after 191 links are made, it is unlikely that further iterations (in the competitive linking process) will improve precision or recall, as only 191 of the original possible links are correct. For this reason, competitive linking results are shown only for 191 iterations. Iteration 182 is shown because it is known (in the experiment) to be the point at which

[16] Following the discussion in Section 5, k refers to the number of translationally equivalent pairs, and n refers to the total number of text samples in each language. The number of noise added is $n - k$.

precision cannot increase, since it is known that only 182 correct pairs passed the $p = 0.1$ length filter. Therefore, recall cannot pass 0.9529.

	iteration 182		iteration 191 (k)	
noise level ($n - k$)	precision	recall	precision	recall
0% (0)	0.5934	0.5654	0.5707	0.5707
33% (96)	0.5824	0.5550	0.5550	0.5550
50% (191)	0.5879	0.5602	0.5602	0.5602
67% (382)	0.5824	0.5550	0.5550	0.5550
75% (573)	0.5879	0.5602	0.5602	0.5602

Table 13: Effects of noise on the SLCP dictionary t function.

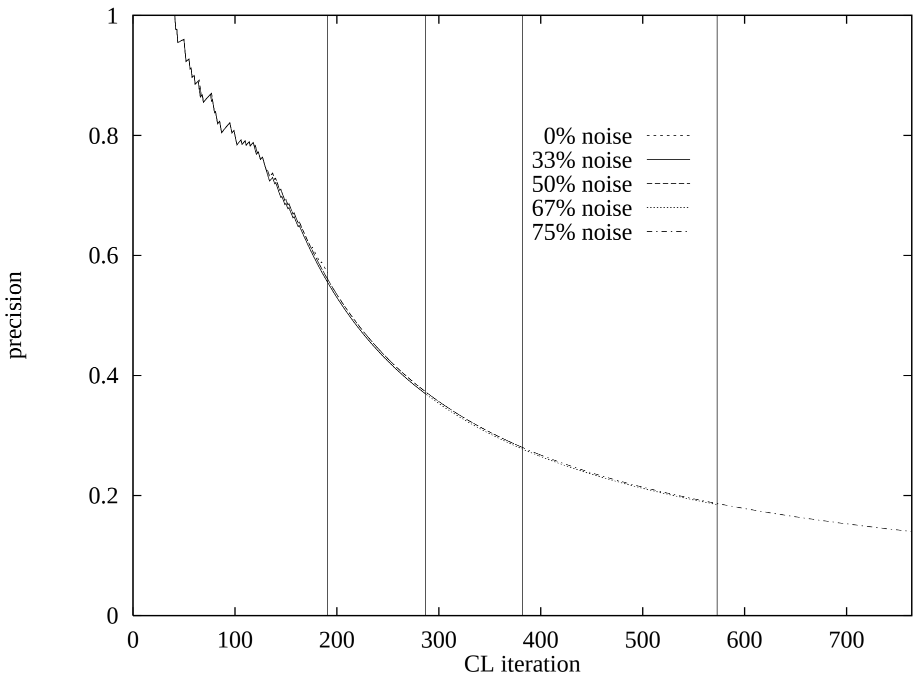

Figure 11: Precision of the SLCP dictionary is hardly affected by noise. The vertical lines mark the points where competitive linking ceases for successively noisier scenarios; e.g., linking stops at iteration 287 when there are total 287 (191 good plus 96 noise) pairs to be linked.

These results show that the SLCP t function is robust to noise, even when the amount of noise is three times the amount of correct pairs present. Note that linking past 182 results only in decreased precision. Of course, in practical applications, the number of correct pairs that passed the filter will be unknown, as will be k, the number of translationally equivalent pairs present. Determining when to stop the linking process is not addressed here, though it follows from the competitive linking results that choosing a threshold will suffice. Such a threshold could be chosen empirically by finding the range of r scores for known translation pairs. Alternately, the percentage of noise ($\frac{n-k}{n}$) in the candidates could be estimated and only the top k links taken.

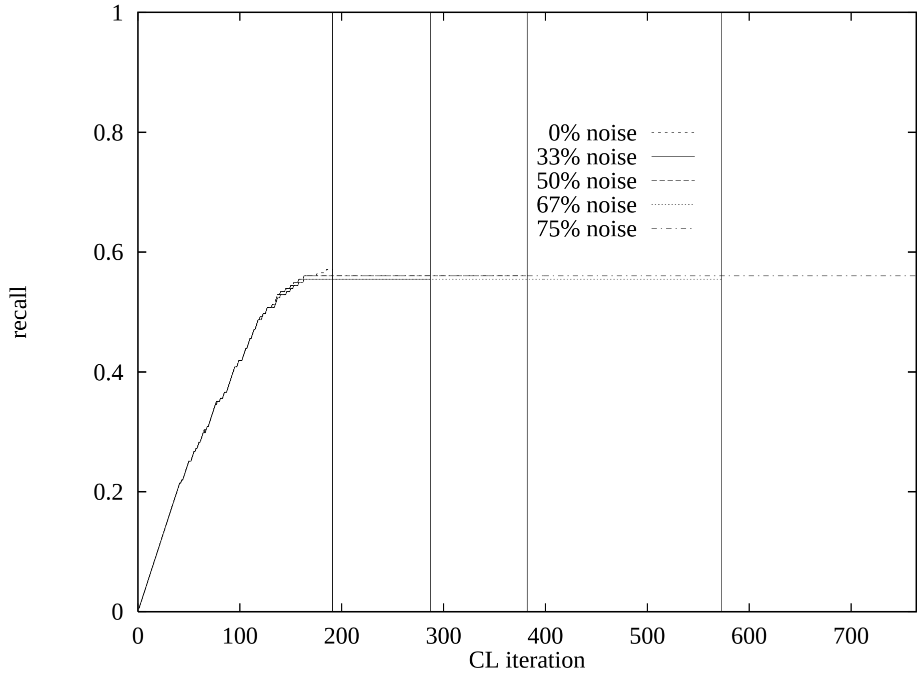

Figure 12: Recall of the SLCP dictionary is hardly affected by noise. The vertical lines mark the points where competitive linking ceases for successively noisier scenarios.

	iteration 182		iteration 191 (k)	
noise level ($n-k$)	precision	recall	precision	recall
0% (0)	0.8242	0.7853	0.7853	0.7853
33% (96)	0.5659	0.5393	0.5550	0.5550
50% (191)	0.5550	0.5288	0.5340	0.5340
67% (382)	0.5550	0.5388	0.5288	0.5288
75% (573)	0.5495	0.5236	0.5236	0.5236

Table 14: Effects of noise on the translation model (unweighted) t function.

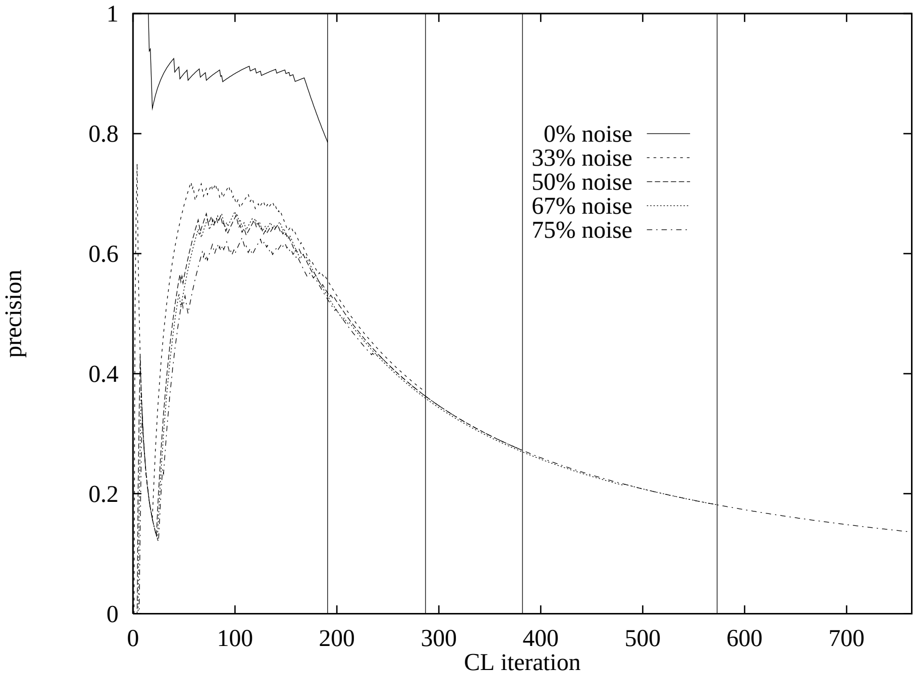

Figure 13: Precision of the unweighted translation model is heavily affected by noise. The vertical lines mark the points where competitive linking ceases for successively noisier scenarios.

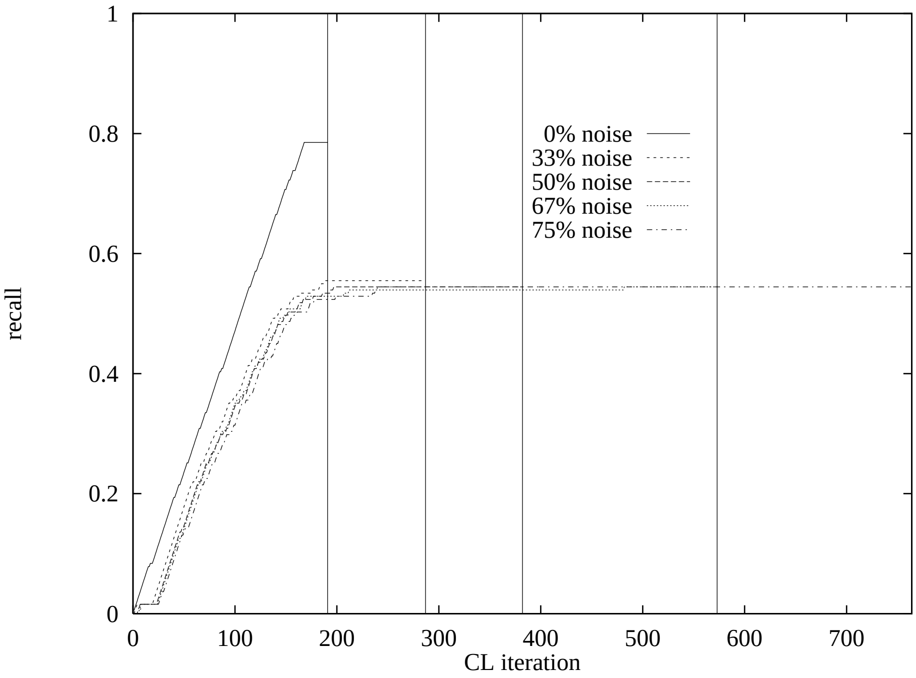

Figure 14: Recall of the unweighted translation model is heavily affected by noise. The vertical lines mark the points where competitive linking ceases for successively noisier scenarios.

The translation model's performance is greatly affected by the addition of noise. Both precision and recall are markedly diminished early in the competitive linking process and remain so throughout. However, performance does not diminish greatly as the amount of noise increases.

One possibility for the decrease in precision and recall of this t function is the "noise" in the function itself. Recall that his function was formed by removing weights from statistical translation entries. While many entries in the translation model are translationally equivalent pairs, some are not. Presumably higher weights will correlate with translational equivalence. Therefore, by removing the weights (i.e., making t a boolean function), incorrect word pairings will be given the same attention as their correct counterparts. It is possible that this method is robust to either noise in the candidates *or* noise in the t function, but not both. To test this, I carried out the same experiment with the weighted translation model as t function, including the baseline case with no noise. The results are not given here because the hypothesis proved incorrect: not only did the weighted translation model fail to achieve the same performance, as the unweighted version (as in the original experiment with no length filter, see Section 7.1.3), but it degraded sharply as candidate noise increased.

	iteration 182		iteration 191 (k)	
noise level ($n-k$)	precision	recall	precision	recall
0% (0)	0.8407	0.8010	0.8115	0.8115
33% (96)	0.4670	0.4450	0.4555	0.4555
50% (191)	0.4121	0.3927	0.3927	0.3927
67% (382)	0.3352	0.3194	0.3246	0.3246
75% (573)	0.2747	0.2618	0.2618	0.2618

Table 15: Effects of noise on the translation model (unweighted) t function merged with the SLCP dictionary t function.

Performance of the merged t function degrades much more quickly than the translation model alone, as noise increases. In addition, precision and recall never reach the same level as either the SLCP dictionary or the translation model alone.

Three key conclusions may be drawn from this experiment. First, some t functions may be chosen that are robust to noise in the candidate data (e.g., SLCP English-Chinese dictionary); such relations perform almost as well in noisy conditions as in noiseless ones. Second, some such functions do not diminish in performance as the level of noise increases (e.g., SLCP dictionary and the unweighted Method A translation model). Finally, the translation model (unweighted) performed almost as well as the SLCP dictionary in noisy conditions. While the results from this section suggest that the best-performing t functions are not necessarily resilient enough to be useful in noisy conditions, there is promise that noise in the candidates need not result in performance loss. Finding functions that perform well *and* hold up to noise is left for future research, though the English-Spanish results in Section 7.2 suggest that some bilingual dictionaries may fit the bill.

7.5 Cognates: A Bonus

In Section 4, a method for deriving a score of cognate-ness between words (HSCR) was described. It was suggested that using that score as a t function was worth consideration.

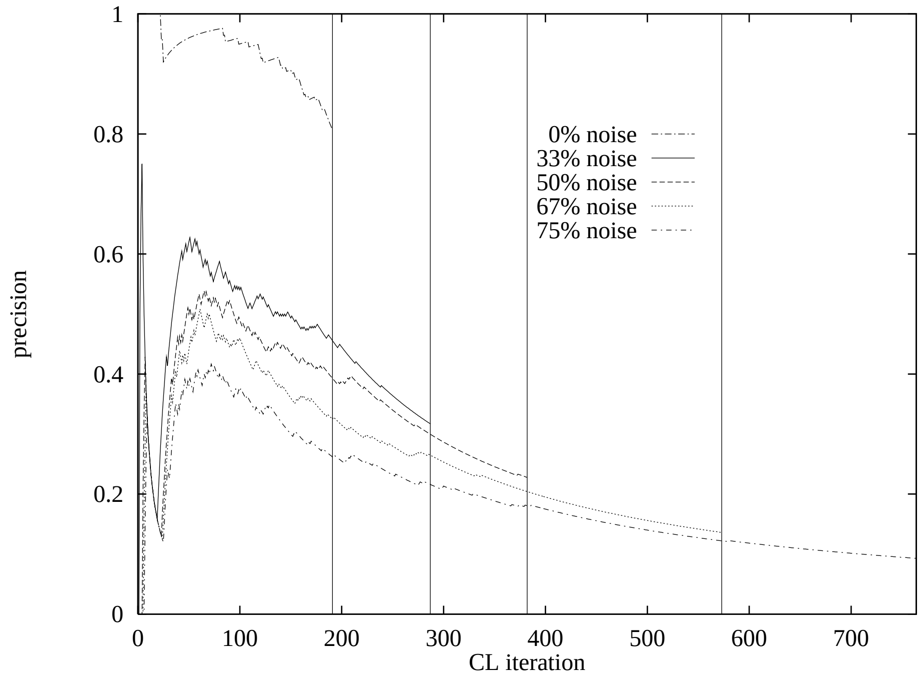

Figure 15: Precision of the merged t function is heavily affected by noise. The vertical lines mark the points where competitive linking ceases for successively noisier scenarios.

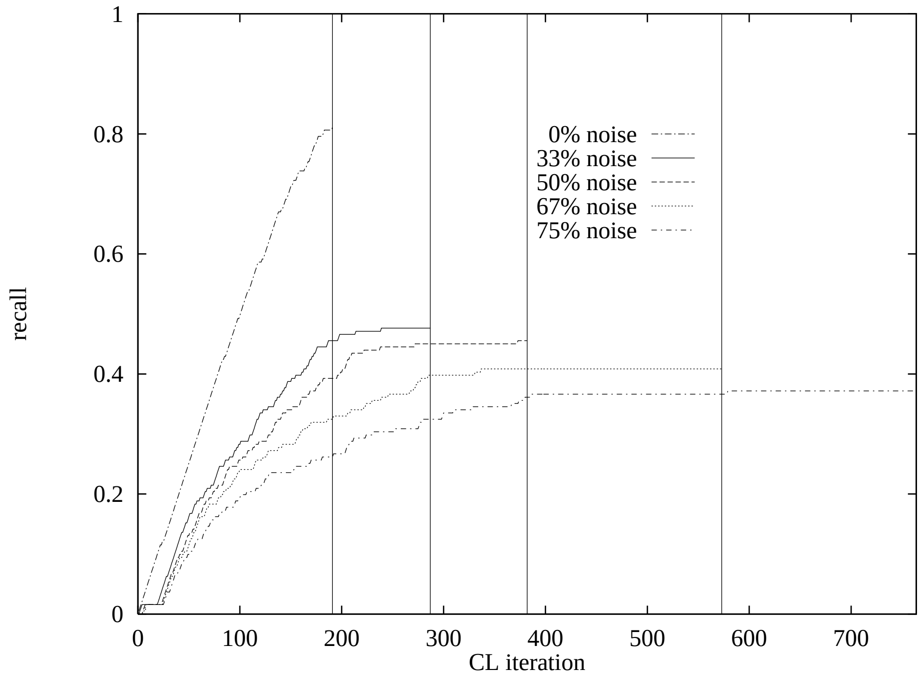

Figure 16: Recall of the merged t function is heavily affected by noise. The vertical lines mark the points where competitive linking ceases for successively noisier scenarios.

Here an experiment on English-French (a cognate-rich pair) is described and results reported. The evaluation task here is the same as that in the other experiments; no noise was added and no length filter was applied.

Facts about the corpus used for training the translation model (details below) and the corpus used for testing are shown in Table 16.

	size (segments)	English tokens	English types	French tokens	French types	mean English segment size	mean French segment size
training	5,963	153,199	7,993	168,361	10,529	25.53	28.06
test	200	3,788	917	3,858	920	18.21	18.55

Table 16: English-French Bible parallel corpus. The English and French text was tokenized using scripts included in the Egypt distribution [WS99], written by Dan Melamed and Yaser Al-Onaizan.

7.5.1 Electronic bilingual dictionary

A t function was derived from an electronic English-French dictionary, using the same methods described in Section 7.1.1. To begin, the French-to-English dictionary[17] contained 34,804 entries. The entries were reversed to be English-to-French for this purpose, and each entry was tokenized using tokenization scripts for these languages included in the Egypt distribution [WS99].

One-to-one entries (OO)

1. All entries involving more than one English word or more than one French word were removed. 30,783 entries remained.

2. Any entries including words not found in the candidate pairs were removed (this step was carried out for efficiency reasons; it could not have affected performance in any way). The final OO dictionary contained 4,462 entries.

Stoplisted cross-product of non-one-to-one entries (SLCP)

- All non-one-to-one entries from the dictionary were collected; the total was 4,021 entries.

- Any entries containing words not present in the corpus were removed. The remaining set contained 837 entries.

- A stoplist filter was applied to the set. The English stoplist contained 238 common closed-class words; the French stoplist contained 311 words[18]. After the filter was applied, any entries where the English or French side was empty were removed. The result was 673 entries.

[17] This dictionary was derived by Gina Levow from an English-to-French one freely available at http://-www.freedict.com.

[18] Thanks to Dan Melamed for the use of these stoplists.

- The non-one-to-one entries were expanded into multiple entries by taking the cross-product of English words with French words. All elements in the cross-product were included; the result was 1,210 entries. 111 of these entries were not present in the OO dictionary.
- These entries were merged with the OO dictionary. The new dictionary totaled 5,561 one-to-one entries.

The SLCP dictionary, in preliminary tests, outperformed the OO dictionary, so it was the only one used in this experiment.

7.5.2 Automatically learned t functions

Another set of t functions came from the set of non-zero entries from a Method A translation model trained on 5,963 aligned English-French verses of the Bible. These were selected at regular intervals[19] throughout the Bible from the unannotated section of the Blinker corpus [Mel98]. Table 16 shows facts about the training corpus for the translation model. There were 9,593 non-zero entries in the translation model. As before, one t function was defined as $t(e, f) = \Pr(e, f)$. The same 9,593 entries from the translation model were also used without weights (i.e., $t(e, f) = 1$ if and only if $\Pr(e, f)$ is non-zero. As before, a merged t function, the union of the unweighted translation model with the SLCP dictionary, was created; it contained 14,446 non-zero entries.

Using the technique from Section 4, an m function was induced using the translation model. This, in turn, was used to compute HSCR cognate-ness scores for all word pairs in the cross-product of the set of English types with the set of French types. An *ad hoc* threshold of 0.2 was selected; this left a set of 340,213 scored word pairs in the set of candidates for which $t > 0$.

Though this set was highly noisy (i.e., many pairs were not cognates), preliminary comparisons with higher thresholds showed that 0.2 was, for this m function and data set, reasonable[20] The t function was $t(e, f) = \mathrm{HSCR}(e, f)$ if $\mathrm{HSCR}(e, f) \geq 0.2$ and 0 otherwise, and the unweighted version was $t(e, f) = \mathrm{HSCR}(e, f)$ if $\mathrm{HSCR}(e, f) \geq 0.2$ and 0 otherwise. Table 17 shows some sample word pairs from the test corpus and their HSCR scores. Because biblical text is rich in names, the m function adeptly captures character-to-character mappings. Note the many name pairs from the test corpus with high scores.

7.5.3 Results

Table 18 shows, under both competitive linking and maximum weighted bipartite matching, precision and recall of these t functions on the detection task. Precision-recall plots for these t functions are shown in Figures 17 and 18.

For this data set, the cognate function is the best-performing t function, though the translation model (without weights) nearly reaches its level of precision under maximum weighted bipartite matching. The SLCP dictionary's performance was strikingly low; this may be due in part to the absence of accents in the dictionary (accents were present in the corpus).

[19] For this task I used the Whittle tool by Mike Jahr included in the Egypt distribution [WS99].

[20] Note that a "good" HSCR score for a word pair is entirely dependent on the m function, which in turn depends entirely on the training corpus. This threshold cannot be said to be generally applicable. Choosing a good threshold is an empirical question left unexplored here.

rank	HSCR	English	French
1	0.5022	Jobab	Jobab
2	0.4814	Jabal	Jabal
3	0.4809	Jarib	Jarib
4	0.4790	Joram	Joram
1,001	0.3551	Jorah	Jaera
1,002	0.3550	Menahem	Menahem
1,003	0.3550	Mahath	Maath
1,004	0.3550	Hadid	Hadid
8,001	0.2968	Ahuzzam	Achuzzam
8,002	0.2968	Dishon	Disent
8,003	0.2968	Jorai	Joanan
8,004	0.2968	Jaddai	Joanan
64,001	0.2430	vapor	évapore
64,002	0.2430	Jabesh	Jaroach
64,003	0.2430	Caraway	Conania
64,004	0.2430	Branches	Barabbas
340,210	0.2000	Dedication	variation
340,211	0.2000	Separate	égarants
340,212	0.2000	separate	égarants
340,213	0.2000	Pagiel	Publie

Table 17: Examples of English-French cognate-ness (HSCR) scores. These examples were chosen arbitrarily to show a range of scores and noisiness.

	competitive linking		MWB matching	
t function	precision	recall	precision	recall
SLCP dictionary	0.2400	0.2400	0.3393	0.2850
Method A translation model	0.8800	0.8800	0.9677	0.9000
unweighted translation model	0.9100	0.9100	0.9840	0.9200
union of SLCP and unweighted t. m.	0.8750	0.8750	0.9538	0.9300
HSCR (0.2 threshold)	0.9450	0.9450	0.9949	0.9800
unweighted HSCR (0.2 threshold)	0.9400	0.9400	0.9695	0.9550

Table 18: Performance of various t functions on the English-French test corpus.

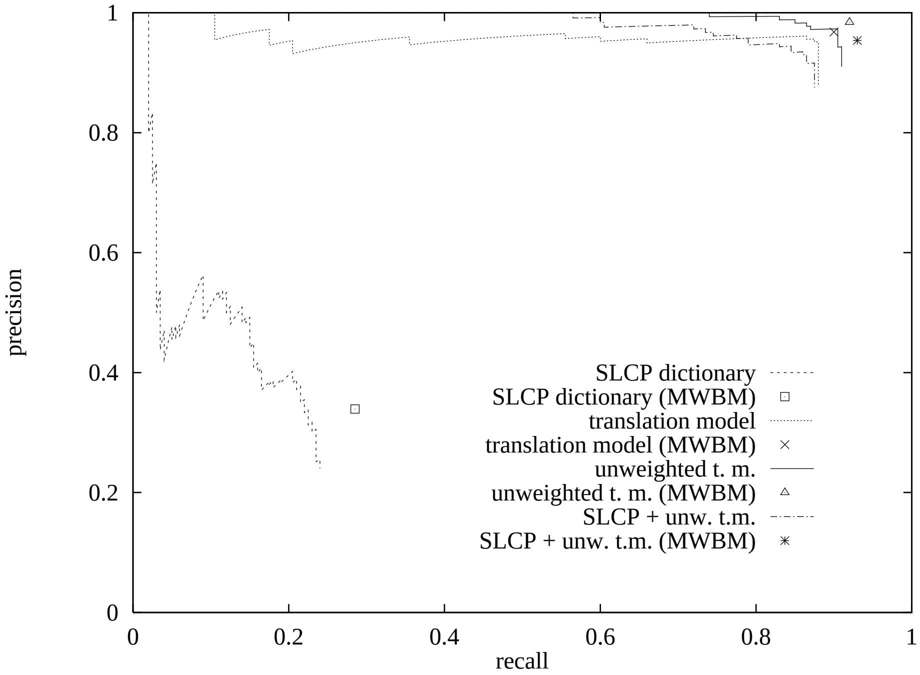

Figure 17: Performance of the SLCP dictionary, translation model (with and without weights) and the two combined.

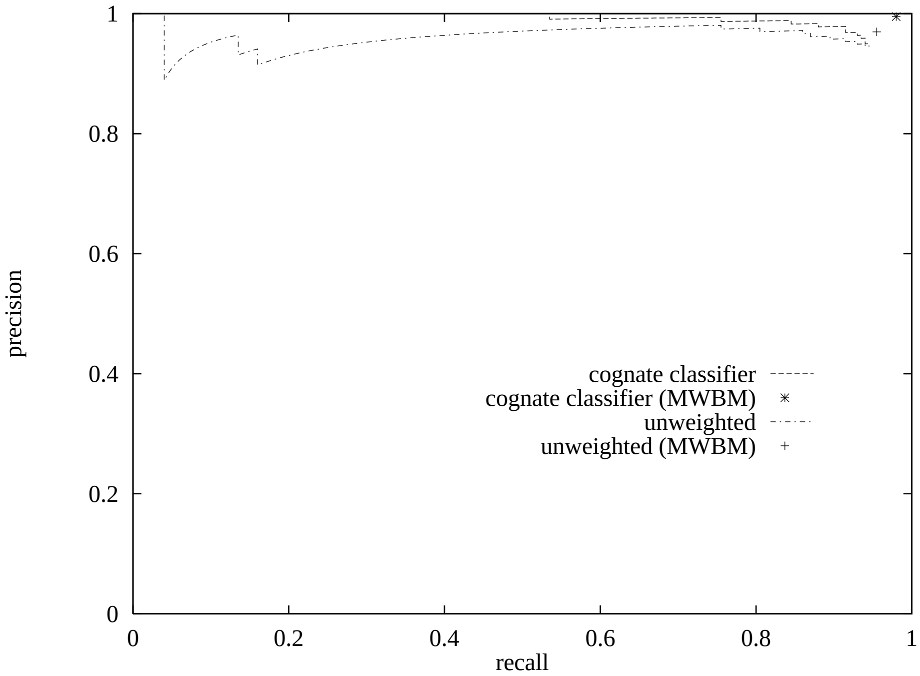

Figure 18: Performance of the HSCR (0.2 threshold) t function with and without weights.

The unweighted translation model and the unweighted HSCR t functions were merged to create a new t function such that $t(e, f) = 1$ if $\Pr(e, f) > 0$ or $\mathrm{HSCR}(e, f) > 0.2$ and 0 otherwise. The result contained 348,724 entries. This t function offered precision and recall of 0.99 under both competitive linking and maximum weighted bipartite matching. The precision-recall plots showing this in comparison to both of the components merged are shown in Figure 19.

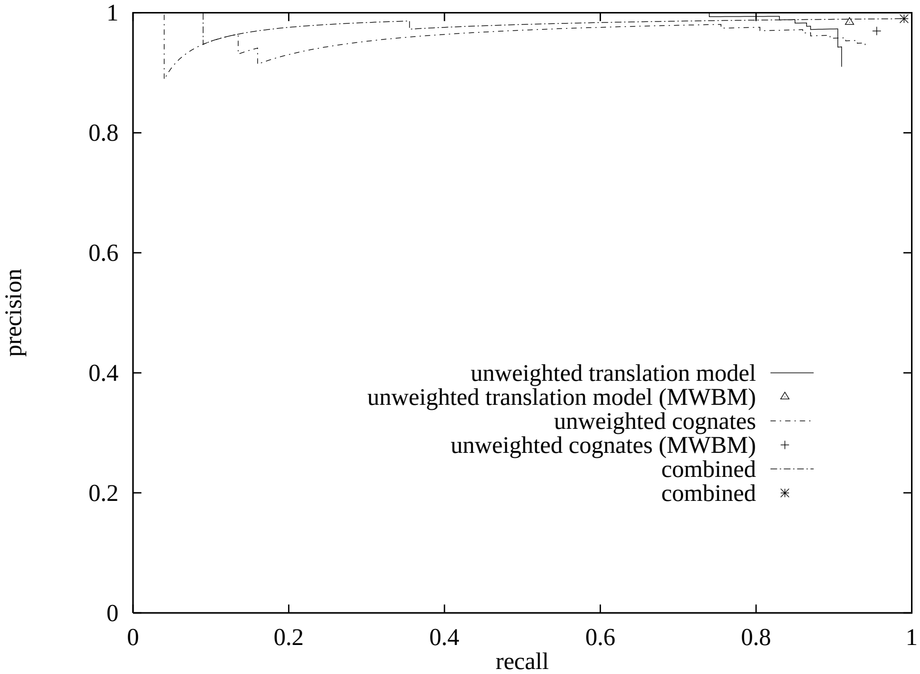

Figure 19: Performance of the HSCR (0.2 threshold) t function combined with the unweighted translation model.

Taking the union of the SLCP dictionary, the unweighted translation model, and the unweighted cognate classifier (a total of 352,739 non-zero entries) yielded precision of 1 with recall of 1 under both competitive linking and maximum weighted bipartite matching. (This result is not shown in any of the precision-recall plots.) In this particular instance, combining three sources of information in the simplest way (union) yielded perfect classification.

It can be concluded that even noisy cognate classifiers are useful for translation detection.

7.6 Performance at the Document Level: Comparison with STRAND

One of the potential applications for this technique discussed in Section 1.1 was using content-based scores of translational equivalence to boost performance of the STRAND system [Res99], which uses structural information only to locate translation pairs at the document level. Here I offer preliminary results that show the capability of the resemblance scoring technique to classify document pairs by translational equivalence in comparison with STRAND's capability.

Resnik [Res99] ran an experiment with two human judges in which STRAND classifications of candidate document pairs were compared to human ratings of translational equivalence. In all, there were 233 English-French document pairs[21] for which the human judges agreed (see Table 19 for data on these pairs). Of these, 84 were ruled as translation pairs, and the other 149 were ruled as inequivalent. The STRAND system correctly identified 66 of the 84 good pairs as translations (i.e., recall = 0.7857), with precision of 0.9429. The STRAND result is used here as a benchmark.

	size (pairs)	English tokens	English types	French tokens	French types	mean English segment size	mean French segment size
good	84	56,388	21,378	64,730	28,492	671.29	770.60
bad	148	180,912	30,247	120,810	33,940	1222.38	816.28
all	232	237,300	34,470	185,540	38,318	1022.84	799.74

Table 19: English-French STRAND candidate pairs. Thanks to Philip Resnik for use of the documents and details of the STRAND experiment. The English and French text was tokenized using scripts included in the Egypt distribution [WS99], written by Dan Melamed and Yaser Al-Onaizan. "Good" refers to candidate pairs agreed to be translationally equivalent by the human judges; "bad" refers to the candidate pairs agreed not to be translationally equivalent by the human judges.

The base t functions used in this experiment were:

- An SLCP dictionary derived from the same original dictionary in Section 7.5, retaining all relevant entries (total size was 7,637 entries).
- Method A tranlsation model from Section 7.5 with weights.
- Method A tranlsation model from Section 7.5 without weights.
- HSCR (cognate-ness) scores were computed for every word co-occurrence pair in the set of candidate documents[22] using the m function from Section 7.5. As before, these were thresholded at HSCR 0.2; this left 36,137 pairs for which $t > 0$. The t function was defined as $t(e, f) = \text{HSCR}(e, f)$ if $\text{HSCR}(e, f) \geq 0.2$ and 0 otherwise.
- A boolean t function derived from the HSCR: $t(e, f) = 1$ if $\text{HSCR}(e, f) \geq 0.2$ and 0 otherwise.

The genre of the translation model training data (biblical verses) is in sharp contrast to the genre of the candidate documents (World-Wide Web documents), so it is expected that the coverage of the translation model t function will be limited for this purpose. For this reason cognates are expected to increase the performance of the translation model.

Table 20 shows some sample word pairs from the cross-product of words in the English and French documents and their HSCR scores. Note that English words appear in French text and vice versa, and that Web text often contains errors. Capital letters tend to receive

[21] For this experiment, one pair was not used because it contained a great deal of non-linguistic material (i.e., a compressed file), resulting in an excessively high number of "terms," most of which were not words but pieces of compressed data. This pair was ruled as "bad" by both judges and STRAND.

[22] A small error was made in this process. The minumum length of a French type was computed to be 5, but because of a small error in the code, only pairs involving French words of length 6 or greater were actually used. At the time of printing, there was not time to correct this; it stands to reason that more cognates would be found and used if the error was corrected. This error did not affect cognate scoring in the experiments described in Section 7.5.

high m scores because they are less common than lower-case letters, so many all-capital words receive high HSCR scores even though they are not cognates.

rank	HSCR	English	French
1	0.8315	CANADA	CANADA
2	0.7926	COMMAND	COMMAND
3	0.7871	NACIONAL	NACIONAL
4	0.7812	ANGLAIS	ANGLAIS
201	0.5238	MAISON	ACTION
202	0.5228	VEHICLES	VÉHICULE
203	0.5226	ARGENTINA	ARGENTINES
204	0.5225	BORDEAUX	BORDEAUX
1,001	0.3595	Bettina	Bettina
1,002	0.3595	Corporations	Corporation
1,003	0.3595	Corportation	Corporation
1,004	0.3595	VERSION	EDITION
8,001	0.2549	BASICLY	AIDENT
8,002	0.2549	TRAFFIC	AIDENT
8,003	0.2549	Limited.	Limitée
8,004	0.2549	Walter	Walter
20,001	0.2174	information	orientation
20,002	0.2174	Panama	Paraphé
20,003	0.2174	Politicised	politiciens
20,004	0.2173	Cherney	Culture
36,134	0.2000	LABELLING	SOMMAIRE
36,135	0.2000	associate	soixante
36,136	0.2000	Choosing	Coordonner
36,137	0.2000	transition	association

Table 20: Examples of English-French cognate-ness (HSCR) scores. These examples were chosen arbitrarily to show a range of scores and noisiness.

7.6.1 Structure vs. Content

The candidate generation component of STRAND selects candidate document *pairs* rather than candidate documents. This means that it is unnecessary to score the cross-product of English documents with French documents; only the 232 candidate pairs needed to be scored. Competitive linking degenerates to simple thresholding by resemblance score in this scenario, since there is no "competition" among documents of the same language.

In order to test content-based comparison only, structural information (i.e., HTML tags) was completely removed from the documents. The only preprocessing applied was tokenization.

Results are reported as precision-recall plots. Each plotted point corresponds to a threshold. The highest thresholds offer high precision and low recall, and as the threshold decreases, precision also decreases while recall increases. The points in the upper left corner of each plot correspond, then, to the highest thresholds. The STRAND benchmark is marked as a reference point.

The SLCP dictionary's performance (see Figure 20) came very close to that of STRAND,

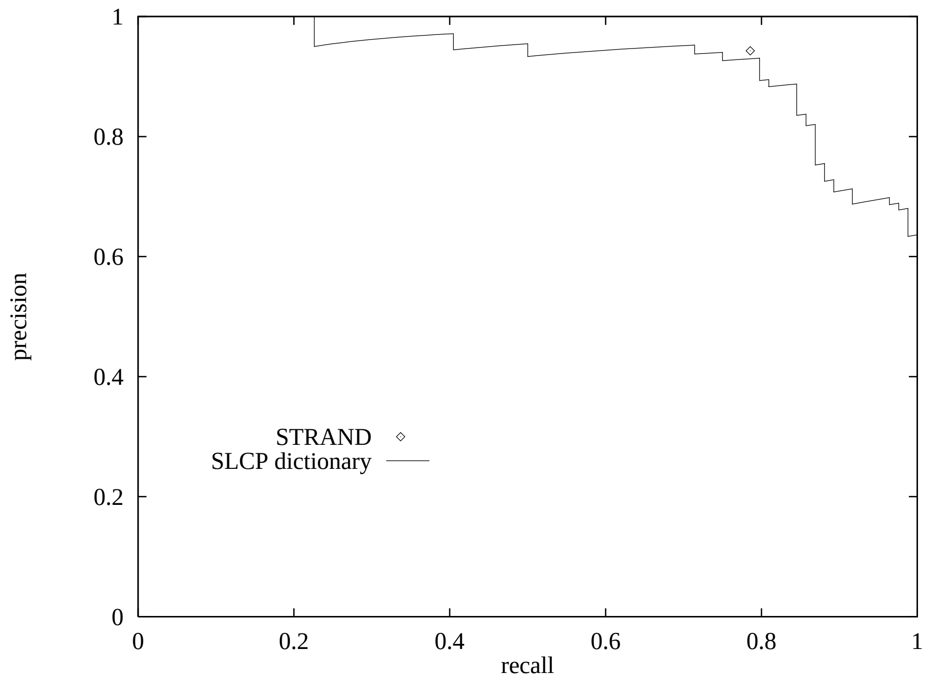

Figure 20: Performance of the SLCP dictionary as compared to STRAND.

but did not surpass it. This is rather surprising, considering this dictionary's dismal performance on the Blinker Bible corpus in the experiment reported in Section 7.5. It is possible that the absence of accents in the dictionary was less of a problem in this case, since the documents themselves may lack accents.

As before, the performance of the unweighted translation model was significantly greater than that of the weighted one (see Figure 21). Neither, however, obtained the level of performance of STRAND or the SLCP dictionary. This is not surprising, given the genre difference of the training and test data.

Another likely effect is the size of the documents. Recall that the resemblance score does not take into account word type frequency in a text sample. A word type may be present once or one hundred times in a document, and that document's resemblance score with any other document will remain the same. This is more important at the document level than the sentence level, since documents are more likely to contain many duplicates of word types, some of which may be highly informative. Further investigation is necessary to determine the robustness of this technique to candidate text sample size.

As with the translation model, the cognate t function performed better without weights than with weights (see Figure 22). The cognate classifier is not terribly useful on its own, but given that only a small subset of word pairs will actually be cognates, it is expected to serve mainly to supplement other t functions that are domain-limited.

It might be expected, for example, that the cognate classifier would complement the translation model, adding additional word pairs that, though not in the Bible corpus, are useful for Web documents. Such cognates might include proper names and technical terms. I tested the performance of the union of the two, and it performed consistently worse than the translation model alone. It is unclear why this would be the case; further investigation is required to understand what is happening in this situation.

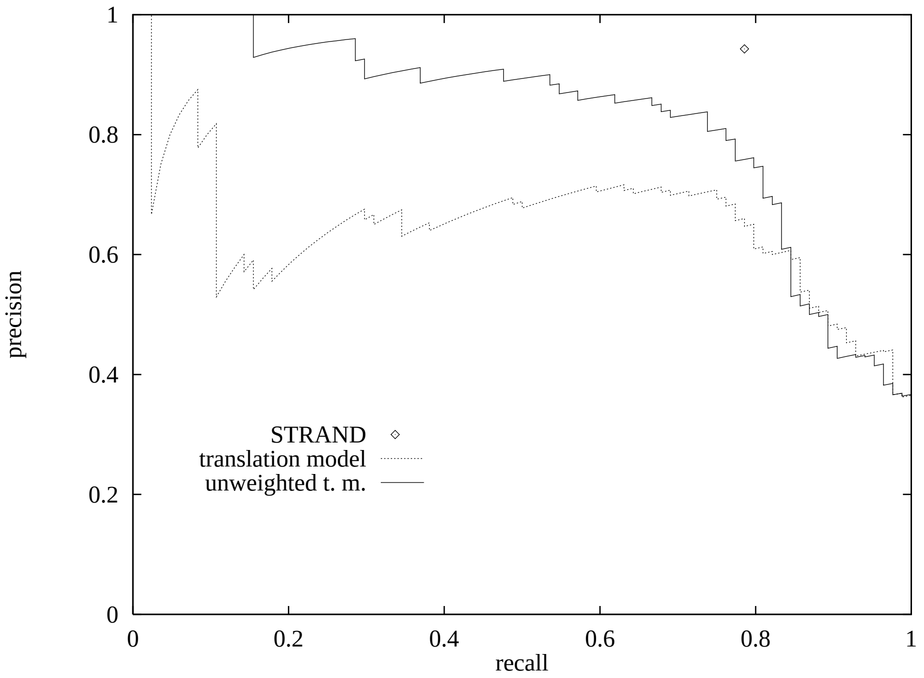

Figure 21: Performance of the translation model with and without weights.

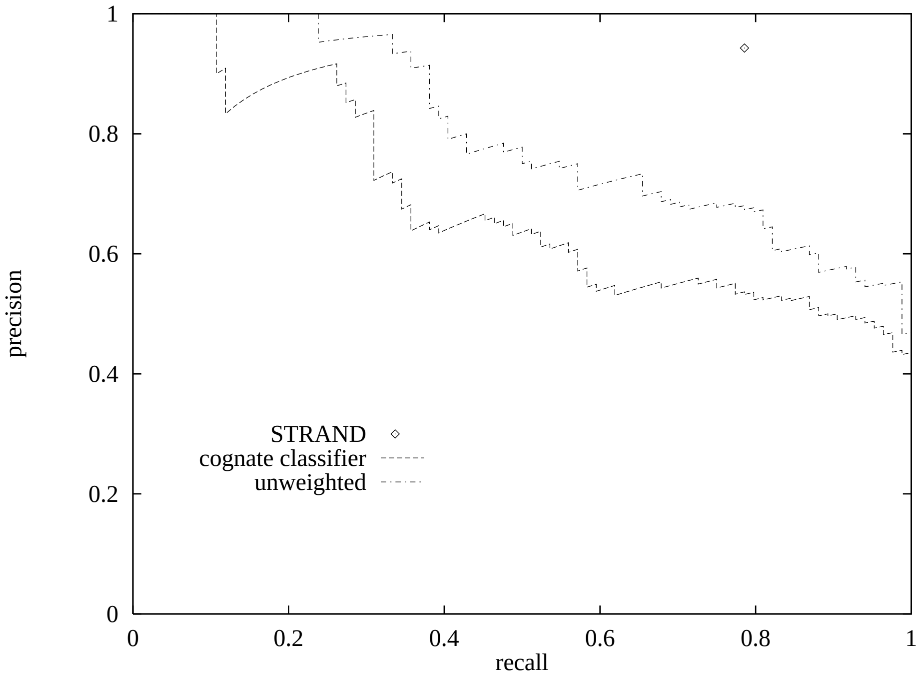

Figure 22: Performance of a cognate classifier with and without weights, in comparison to STRAND.

Finally, the SLCP t function was merged with the unweighted cognate classifier (yielding 43,485 non-zero entries), and also with both the unweighted cognate classifier and unweighted translation model (yielding 52,561 non-zero entries).

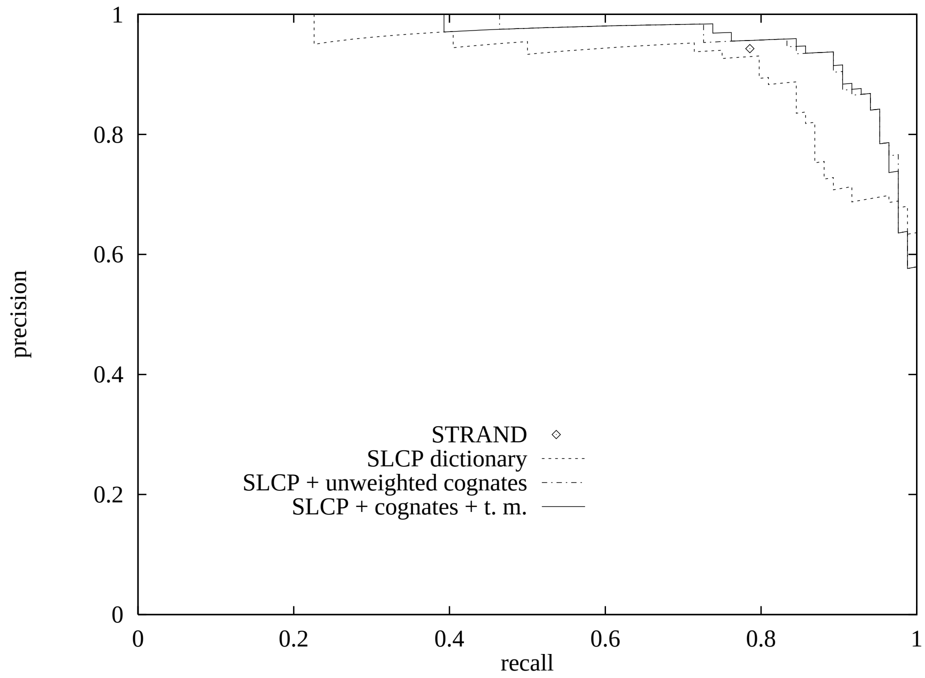

Figure 23: Performance of the SLCP dictionary compounded with the unweighted cognate classifier, and with both the Method A translation model without weights *and* the cognate classifier in comparison to STRAND.

As shown in Figure 23, this method outperforms STRAND when the t function is the union of the SLCP dictionary with the cognate classifier (unweighted). There are slight improvements beyond that when the translation model is added in as well[23]. By adding in cognates, the number of non-zero pairs for t was increased to 43,485, an increase of 522%. Adding the translation model entries as well resulted in 52,561 pairs. Table 21 shows this t function's precision and recall at STRAND's levels of recall and precision.

technique	precision	recall
STRAND [Res99] (structure)	0.9429	0.7857
resemblance scoring (content)	**0.9565**	0.7857
	0.9429	**0.8571**

Table 21: Content-based translation detection wins out over structure-based translation detection.

8 Future Work

A number of extensions to this research may be suggested; some are described briefly here.

[23]The translation model added to the SLCP dictionary by itself did not result in any improvement.

8.1 Theoretical modifications

One of the key shortcomings of the resemblance-scoring method described here is its failure to take into account multiple occurrences of a word type in a text sample. This information could be important, for example, when dealing with samples on the same topic in which a word type is used frequently, sometimes more than once in a sample. The number of times that word type is seen could be an indicator of its importance in a sample; that information in turn should be matched with samples in the other language.

The technique presented here views text samples as sets of words. I did not explore the effectiveness of shingles of size greater than 1, since dictionaries and translation models will not generally be useful for the straightforward derivation of t functions over pairs of such shingles. It might be worthwhile to examine ways to exploit such (mostly) one-to-one resources to develop bigram or trigram t functions.

Along the same lines, I did not use a generative probability model for pairs of word-sets, mainly because of the problem of exponential decay in probability of an item as length increases. Eisner [Eis01] suggested an approach to this which might be considered in further work. By constructing a model M for translationally equivalent pairs *and* a model $\overline{M}$ for other pairs, the probability that a pair (E, F) was generated by M is:

$$\Pr(M|E,F) = \frac{\Pr(E,F|M) \cdot \Pr(M)}{\Pr(E,F|M) \cdot \Pr(M) + \Pr(E,F|\overline{M}) \cdot (1 - \Pr(M))}$$

8.2 Enhancements and Further Experimentation

The noisy scenario in Section 7.4 is one likely to be encountered in many applications. Frequently the amount of noise present (or, equivalently, the number of translationally equivalent pairs present) in the data to be processed will be unknown. A generic means to estimate the amount of noise or a threshold for the r threshold would be useful in such scenarios.

It might be fruitful to explore ways to deal with polysemy, such as word-sense disambiguation as a preprocessing step.

I did not carry out extensive testing of the robustness of this technique to text sample size. This could be done relatively easily using an aligned parallel corpus by segmenting it at multiple levels of granularity and evaluating the precision and recall at different sample sizes and different levels of variability in sample size.

The containment score c was not used at all in this technique. This score estimates the extent to which one sample is "contained" in the other. It is possible that this score might be exploited in some way.

One direction that might lead to increased performance is to consider an iterative framework in which the classifier is used to select the unlabeled pairs it is most confident are translations, and then these pairs are added to the training corpus. More robust empirical classifiers (translation models and/or cognate classifiers) might then be learned from the enhanced corpus. Similar techniques have proven successful in work by Yarowsky [Yar95] and Blum and Mitchell [BM98].

Other than the STRAND evaluation, this research involved no reliance on human judgement. Although the aligned texts used may be considered reliable, some pairs "wrongly" matched might be considered close in meaning by human judges. Further evaluation involving bilingual speakers would more strongly support the positive findings presented here.

It goes without saying that better t functions are to be obtained, perhaps empirically like the translation models. A key feature in such functions is their ability to hold up to noisy conditions. Interestingly, my findings suggest that neither the size of the t function (i.e., number of non-zero entries) nor weights (i.e., a non-boolean function) offer performance advantages.

Most importantly, better means of combining multiple sources of word translational equivalence information need to be considered. In many cases, combining two high-performing boolean t functions (through union) did not result in a significantly better t function, even though the intersection was small. It is unclear why this is the case, but clearly there can be benefits of combining, for example, a dictionary with cognates.

In particular, an interesting direction would be to combine the content-based equivalence information offered by this technique with the structure-based equivalence information used by systems like STRAND. It is likely that using cross-lingual resemblance scoring in tandem with structural comparison would increase the capabilities of parallel text mining applications.

9 Conclusions

This discussion has presented a general algorithm which can reliably classify whether two pieces of text are translations of each other. To my knowledge, no such technique has been developed based on text content alone. This technique has shown success on three language pairs: English-Chinese, English-Spanish, and English-French.

Because many scenarios are likely to involve large search spaces for translational pairs, I have offered a filter to limit the search without affecting performance. I have shown that, depending on the resources used to define word-to-word translational equivalence, this approach can be robust to noise. Another useful finding is that, in this framework, automatically-learned translation models (learned from parallel text) and cognate-scoring functions (learned from translation models or dictionaries) can in some instances be used either on their own or as supplements to *a priori* resources (i.e., electronic bilingual dictionaries) to achieve excellent performance.

Finally, I have shown that a content-based classifier of translational equivalence can compare favorably to one that uses only structural information. It is expected that the technique presented here will prove useful in supplementing systems for parallel corpus construction and other multilingual tasks.

10 Acknowledgements

This work is dedicated to the memory of my mother, Lorie E. Steinberg, who planted seeds but tragically never saw them grow. In addition, I dedicate this work to my second mother, Kathryn G. Smith. Her confidence in my decisions has always encouraged me, her academic devotion and excellence have always motivated me, and her love has never failed me.

The list of individuals who have contributed to my education in Computer Science is a long one. I thank the great teachers who have maintained my excitement about the field: Philip Resnik, Clyde Kruskal, William Gasarch, James Reggia, and Ben Bederson (University of Maryland); and John Hallam (University of Edinburgh). Several concurrent Computer Science students at Maryland have left a mark on my thinking and direction, most notably Dan Lake '02, Greg Marton '99, Dav Clark '99, Lisa Pearl '02, and Pete

Schwartz '01. Conversations with various researchers have helped me focus this project and frame the problem; I thank Doug Oard, Dan Melamed, Rebecca Hwa, Kareem Darwish, and Jason Eisner for their time and insight. Throughout my time at the University of Maryland, I have benefited also from conversations with Amy Weinberg, Nizar Habash, Mona Diab, Bonnie Dorr, Gina Levow, and David Traum.

Several individuals at the Johns Hopkins Language Engineering Workshop in 1999 were instrumental in my intellectual development: Kevin Knight, Dan Melamed, David Yarowsky, John Lafferty, and Fred Jelinek. I credit these individuals with winning me over to statistical approaches in natural language processing.

It is due to the patience and guidance of my advisors that I have discovered my love for research and for natural language processing. I owe a great debt (again) to Dan Melamed, who offered the original concept for this work as well as advice inestimable in quantity and quality.

Finally and most importantly, my advisor Philip Resnik has shown me an optimism and an enthusiasm that cannot help but be emulated. For his many conversations, technical and otherwise, for the opportunities he has made available to me since I knocked timidly on his door as a freshman, and for the genuine faith he has had in my abilities, I thank him.

References

[AMO93] Ahuja, Ravindra K., Thomas L. Magnati, and James B. Orlin (1993). *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, Englewood Cliffs, New Jersey.

[WS99] Al-Onaizan, Yaser, Jan Curin, Michael Jahr, Kevin Knight, John Lafferty, Dan Melamed, Franz-Josef Och, David Purdy, Noah A. Smith, and David Yarowsky (1999). Statistical Machine Translation: Final Report. Johns Hopkins University 1999 Summer Workshop on Language Engineering, Center for Speech and Language Processing, Baltimore, Maryland.

[ABD00] Alshawi, Iliyan, Srinivas Bangalore, and Shona Douglas (2000). Learning Dependency Translation Models as Collections of Finite State Head Transducers. *Computational Linguistics 26(1)*, January.

[BB94] Berger, Adam L., Peter F. Brown, Stephen A. Della Pietra, Vincent J. Della Pietra, John R. Gillett, John D. Lafferty, Robert L. Mercer, Harry Printz, Luboš Ureš (1994). The Candide System for Machine Translation. *Human Language Technology* 1994, pp. 157–162.

[BM98] Blum, Avrim and Tom Mitchell (1998). Combining Labeled and Unlabeled Data with Co-Training. Eleventh Annual Conference on Computational Learning Theory, Madison, Wisconsin.

[BGMZ97] Broder, Andrei Z., Steven C. Glassman, Mark S. Manasse, and Geoffrey Zweig (1997). Syntactic Clustering of the Web. Sixth International World-Wide Web Conference, Santa Clara, California.

[BD93] Brown, Peter F., Stephen A. Della Pietra, Vincent J. Della Pietra, Meredith J. Goldsmith, Jan Hajic, Robert L. Mercer, and Surya Mohanty (1993). But Dictionaries are Data Too. *Human Language Technology* 1993, pp. 202–205.

[CN00] Chen, Jiang and Jian-Yun Nie (2000). Web Parallel Text Mining for Chinese-English Cross-Language Information Retrieval. International Conference on Chinese Language Computing, Chicago, Illinois.

[Chen95] Chen, Lei (1995). A Chinese Text Display Supported by the Chinese Segmentation Algorithm. Master Thesis, Department of Computer Science, New Mexico State University.

[Dam95] Damashek, Marc (1995). Gauging Similarity with n-Grams: Language-Independent Categorization of Text. In *Science 267*, pp. 843–848.

[Eis01] Eisner, Jason. Personal communication, February 2001.

[FK82] Francis, W. Nelson and Henry Kucera (1982). *Frequency Analysis of English Usage: Lexicon and Grammar.* Houghton Mifflin Company, Boston, Massachusetts.

[Graff94] Graff, David (1994). United Nations Parallel Text. Available from the Linguistic Data Consortium at http://www.ldc.upenn.edu.

[KG97] Knight, Kevin and Jonathan Graehl (1998). Machine Transliteration. *Computational Linguistics 24(4)*, September.

[LO94] Leisher, M. and Bill Ogden (1994). Multi-Attribute Text: The Low Level Library and Text Widget, Technical Report, Computing Research Laboratory, New Mexico State University.

[LOC00] Levow, Gina-Anne, Douglas W. Oard, and Clara I. Cabezas (2000). Translingual Topic Tracking with PRISE. Topic Detection and Tracking (TDT-3) Workshop, Tysons Corner, Virginia.

[Lin98] Lin, Dekang (1998). An Information-Theoretic Definition of Similarity. Fifteenth International Conference on Machine Learning, Madison, Wisconsin.

[Logos] Logos Corporation, 111 Howard Boulevard, Suite 214, Mount Arlington, New Jersey. http://www.logos-usa.com.

[Ma00] Ma, Xiaoyi (2000). Hong Kong Laws Parallel Text. Available from the Linguistic Data Consortium at http://www.ldc.upenn.edu.

[MN99] Mehlhorn, Kurt and Stefan Näher (1999). *LEDA: A Platform for Combinatorial and Geometric Computing.* Cambridge University Press, Cambridge, United Kingdom.

[Mel95] Melamed, I. Dan (1995). Automatic Evaluation and Uniform Filter Cascades for Inducing N-best Translation Lexicons. *Third Workshop on Very Large Corpora*, Boston, Massachusetts.

[Mel98] Melamed, I. Dan (1998). Manual Annotation of Translational Equivalence: The Blinker Project. Institute for Research in Cognitive Science Technical Report #98-07. University of Pennsylvania, Philadelphia, Pennsylvania.

[Mel00] Melamed, I. Dan (2000). Models of Translational Equivalence among Words. *Computational Linguistics 26(2)*, June.

[MS00] Melamed, I. Dan and Noah A. Smith (2000). Augmenting Models of Translational Equivalence with Orthographic Cognate Information. West Group Technical Report, Eagan, Minnesota.

[WS00] Meng, Helen, Berlin Chen, Erika Grams, Sanjeev Khudanpur, Gina-Anne Levow, Wai-Kit Lo, Douglas W. Oard, Patrick Schone, Hsin-min Wang, and Jianqiang Wang (2000). Mandarin-English (MEI): Investigating Translingual Speech Retrieval. Johns Hopkins University 2000 Summer Workshop on Language Engineering, Center for Speech and Language Processing, Baltimore, Maryland.

[NSID99] Nie, Jianyn, Michel Simard, Pierre Isabelle, and Richard Durand (1999). Cross-Language Information Retrieval Based on Parallel Texts and Automatic Mining Parallel Texts from the Web. ACM-SIGIR Conference, Berkeley, California.

[Oard01] Oard, Douglas. Personal communication, February 2001.

[Res98] Resnik, Philip (1998). Parallel Strands: A Preliminary Investigation into Mining the Web for Bilingual Text. Third Conference of the Association for Machine Translation in the Americas, Langhorne, Pennsylvania.

[Res99] Resnik, Philip (1999). Mining the Web for Bilingual Text. Thirty-Seventh Annual Meeting of the Association for Computational Linguistics, College Park, Maryland.

[ROL01] Resnik, Philip, Douglas Oard, and Gina Levow (2001). Improved Cross-Language Retrieval using Backoff Translation. *Human Language Technology* 2001.

[RR97] Reynar, Jeffrey C. and Adwait Ratnaparkhi (1997). A Maximum Entropy Approach to Identifying Sentence Boundaries. Fifth Conference on Applied Natural Language Processing, Washington, D.C.

[SFI92] Simard, Michel, George F. Foster, and Pierre Isabelle (1992). Using Cognates to Align Sentences in Bilingual Corpora. Fourth International Conference on Theoretical and Methodological Issues in Machine Translation, Montreal, Canada.

[Tie99] Tiedemann, Jörg (1999). Automatic Construction of Weighted String Similarity Measures. Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, College Park, Maryland.

[Who73] Whorf, Benjamin L. (1973). Linguistic Relativity. In *Reading in Applied Linguistics*. Oxford University Press.

[WFH86] Woods, Anthony, Paul Fletcher, and Arthur Hughes (1986). *Statistics in Language Studies*. Cambridge University Press.

[Yar95] Yarowsky, David (1995). Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. Thirty-Third Annual Meeting of the Association for Computational Linguistics, Cambridge, Massachusetts.