A Technical Word and Term Translation Aid using Noisy Parallel Corpora across Language Groups

Pascale Fung and Kathleen McKeown
Computer Science Department
Columbia University

Abstract.
Technical term translation represents one of the most difficult tasks for human translators since (1) most translators are not familiar with term and domain specific terminology and (2) such terms are not adequately covered by printed dictionaries. This paper describes an algorithm for translating technical words and terms from noisy parallel corpora across language groups. Given any word which is part of a technical term in the source language, the algorithm produces a ranked candidate match for it in the target language. Potential translations for the term are compiled from the matched words and are also ranked. We show how this ranked list helps translators for technical term translation. Most algorithms for lexical and term translation focus on Indo-European language pairs, and most use a sentence-aligned clean parallel corpus without insertion, deletion or OCR noise. Our algorithm is language and character-set independent, and is robust to noise in the corpus. We show how our algorithm requires minimum preprocessing and is able to obtain technical word translations without sentence boundary identification or sentence alignment, from the English/Japanese AWK manual corpus with noise arising from text insertions or deletions and on the English/Chinese HKUST bilingual corpus. We obtain a precision of 55.35% from the AWK corpus for word translation including rare words, counting only the best candidate and direct translations. Translation precision of the the best candidate translation is 89.93% from the HKUST corpus. Potential term translations produced by the program help bilingual speakers to get a 47% improvement in translating technical terms.

1. Introduction

Technical terms often cannot be translated on a word by word basis. While the individual words of the term may have many possible translations, the term itself typically has a unique translation that may or may not correspond to the translation of its components. The word *Governor* can be translated into different Chinese words such as 總督, 主管(*top manager*), 總裁(*chief*), 州長(*of a State*). However, *The Hong Kong Governor* has only one translation—香港 總督 [1]. Likewise, *house* is normally translated as *maison* in French, but *House of Commerce* should be translated as *Chambre de Commerce*. In addition, domain-specific terms such as *Basic Law* or *Green Paper* cannot be translated literally word by word into Chinese. Every word of these terms has multiple translations in dictionaries, while the correct translation of the term sometimes cannot be found in any dictionary.

Human translators of technical materials, not being experts in every technical or regional domain, have difficulty producing the correct translations of techni-

[1] The translation of *Hong Kong Governor* in Chinese has two forms, the full term 香港 總督 and the acronymic form 港督, which consists of the second character of *Hong Kong* and the second character of *Governor*.

cal terms effectively. In addition, neither machine-translation systems nor machine-aided translation systems can perform robustly without a domain-specific bilingual lexicon. Meanwhile, human compiled dictionaries do not cover the jargon of technical and scientific areas which are ever evolving, producing new terms. Updating existing dictionaries or compiling new technical dictionaries would require a tremendous amount of human effort. Therefore, automated domain term translation algorithms are in great demand.

However, just as any bilingual person who has come to Hong Kong would have no trouble translating *Governor* as 總督 from reading newspapers and journals, many *unknown* domain or regional bilingual terms can be learned from texts produced in the specific domain or region. This is due to the relative *consistency* in the mapping of domain terms. The consistency of domain term mapping in different languages gives rise to certain patterns of usage of these terms and their translations. It is possible to deduce term translations from such patterns by using statistical analysis. With the increasing availability of large quantities of electronic texts, much statistical work has been done in the area of translation and lexicon compilation. In our work, we propose a novel statistical learning paradigm for translating technical terms.

Most previous statistical algorithms for compiling bilingual lexicon entries use statistical information derived from clean, sentence-aligned translated documents in a specific domain such as the Canadian Hansard Corpus. Given a list of paired sentences in the two languages, these approaches derive co-occurrence patterns of words across the languages using various correlation metrics. However, bilingual corpora which are sentence-to-sentence translations of each other are not common and are usually limited to transcripts of government debate or government documents.

Many bilingual documents in other domains do not contain sentence-to-sentence translations. For example, the electronic form of the AWK manual [Aho *et al.*1980] in English and its Japanese translation contain many *noisy* translations. One English sentence may be translated as several Japanese sentences, different programming examples are included for the same AWK command, and relevant picture files in the Japanese version are sometimes replaced by a simple pointer to the English version. This results in many non-parallel segments. While such noisy corpora are basically parallel, they cannot be handled by traditional alignment programs which typically work on a sentence by sentence basis and thus, cannot ignore a chunk of text in either source or target. Other bilingual corpora which, like the AWK manual, are noisy or do not contain clear sentence delimiters, include texts available in optical character recognition (OCR) form and Asian/Romance language pairs, where the structures of the languages are so different that their translation is often not done sentence by sentence.

In this paper, we describe an algorithm we developed for translating technical terms given a noisy parallel corpus as input[Fung1995]. The algorithm features a novel method to find word correlations and from these builds technical terms translations. To find individual word correlations, the system first finds a small lexicon to serve as reliable anchor points in the texts using a dynamic time warping algorithm. It is based on the notion that while similar words will not occur at the exact same position in each half of the corpus, distances between instances of the same word will be similar across languages. Dynamic time warping is used to find a mapping between the distances. These reliable anchor points are then used to segment the corpus and correlation metrics can be used to find pairs of words that tend to occur in the same segments.

We tested the algorithm on a noisy parallel corpus of English and Chinese texts, and another one of English and Japanese texts. In the English/Chinese version we collected 626 word translations and suggested 95 term translations. Word translations were evaluated as 89.9% accurate. To evaluate the usefulness of the system's output for technical terms, we compared the ability of human translators to translate the English terms with and without the aid of system output. By checking the list of candidate words suggested by the system as translations for technical terms, their accuracy doubled, increasing from 40% to 80% in the best case and 31% to 77% in the worst, thus dramatically showing the advantage of this algorithm as a translator aid.

In the following sections, we first describe related work on statistical analysis for translation. After describing the corpora we use and their characteristics, we overview our algorithm and then present each of the steps, describing the extraction of technical terms from the English side, finding reliable anchor points using dynamic time warping, determining word correspondences from the resulting segments, and finally, formation of the list of candidate words for term translation. After presenting our results, we describe directions for future work.

2. Related work

Of the large body of previous work using bilingual parallel texts for machine translation, our work is most closely related to research on alignment of parallel texts and research on acquisition of bilingual lexical information for subsequent use by a translator or translation system. Previous work in this area primarily relies on clean, parallel text as input and is based on the premise that sentences in these texts are aligned first. In the following sections, we first present previous work on sentence alignment and show how the notion of sentence alignment can be extended to alignment of segments, which are not strictly parallel, if we can find suitable anchor points. We then turn to research on the acquisition of lexical information,

2.1. Sentence alignment

There are two main approaches for sentence alignment, namely text-based and length-based alignment. The former makes use of lexical information in the sentences and the latter makes use of the total number of characters or words in a sentence.

Length-based algorithms make the assumption that translated sentences in the parallel corpus will be of approximately the same, or constantly related, length. Two main approaches were developed at about the same time, both achieving similar results. [Brown *et al.*1991] base length on the number of words within a sentence, while [Gale & Church1993] base length on number of characters. These methods have been heavily used by other researchers and work quite well with corpora that have cleanly delimited sentence boundaries and where the translation closely parallels the source. More recent research has shown that for cases where sentence boundaries are not clearly marked, as is the case in OCR input [Church1993], or where the languages are less parallel, as is the case with Asian/Romance language pairs [Wu1994, Fung & McKeown1994, Fung1995], these algorithms do not perform as well.

Text-based algorithms use paired lexical indicators across the languages to find matching sentences. One type of indicator is derived from an *a priori* bilingual lexicon. [Warwick-Armstrong & Russell1990] propose using a bilingual dictionary to select word pairs in sentences from a parallel corpus, and then align the sentence pairs containing such word pairs. This can work even when the lexicon is incomplete; thus, [Wu1994] augments a length-based algorithm similar to [Gale & Church1993] by using a small lexicon (e.g. translations of numerals, weekdays) to provide additional anchor points. [Chen1993] also uses lexical information to align sentences but as part of an EM-based parameter estimation method. Again, it requires *a priori* information for bootstrapping; in this case, manual alignment of 100 sentences.

A second type of indicator is suggested by [Simard *et al.*1992] who noted that language-specific knowledge would be helpful for alignment, and proposed using common cognates between language pairs as alignment anchors. In European language pairs, cognates often share character sequences, e.g., *government* and *gouvernement*. Such anchors are more robust to insertion and deletion than length-based algorithms. [Simard *et al.*1992] show, however, that this method could be used in combination with length-based algorithm, not alone.

[Kay & Röscheisen1993] starts with a partial alignment of words, along with sentence boundary delimiters, to induce a maximum likelihood alignment of the sentence level, which is in turn used to refine the word level estimation in the next iteration. The algorithm converges to the correct sentence alignment in a few iterations. The similarity of two sentences is estimated by the number of correlated constituent words, which in turn is estimated on the basis of the similarity of word distributions and the total number of occurrences.

2.2. Segment Alignment

All of the work on sentence alignment requires clear, well-defined sentence boundaries in the texts and assumes one-to-one translation of sentences. [Shemtov1993] has developed an algorithm to handle translations between languages where insertions and deletions are common (in his case, between Hebrew and English). His algorithm first tries to align paragraphs, taking into account insertions and deletions of paragraphs. He then aligned sentences within aligned paragraphs, also using clear, well-defined sentence boundaries. [Church1993] was the first to show that we can also align a text by using delimiters other than sentence or paragraph boundaries. He implements a cognate-based tool, *char_align*, to align texts, producing segments that are bounded by common character sequences between the languages. [Church *et al.*1993] report preliminary success in aligning the English and Japanese versions of the AWK manual [Aho *et al.*1980] using this technique. This was possible since the AWK manual happens to contain a large number of examples and technical words that are the same in the English source and target Japanese. In general, however, the *char_align* method works better for European language pairs since languages such as English and Japanese or Chinese are written in different alphabets.

Like Church, we want to align texts based on segments other than sentences or paragraphs. The problem is finding reliable anchor points that can be used for Asian/Romance language pairs. Segment alignment is more appropriate for aligning noisy corpora since an insertion can be identified as a target segment that is larger than the source segment. If a finer alignment can be made, noise can be identified as a segment whose start and end correspond to the same delimiter in the other language. However, we try to avoid using paragraph boundaries as segment boundries as in [Shemtov1993]. This is because in many translated texts such as the AWK manual, the inserted or deleted segments are not necessarily an entire paragraph or a whole sentence. Thus, paragraph boundaries are not necessarily reliable anchor points.

2.3. Word and term translation

Some of the algorithms used for alignment[2] produce a small bilingual lexicon as a by product [Kay & Röscheisen1993, Chen1993]. Some other algorithms use sentence-aligned parallel texts to further compile a bilingual lexicon [Gale & Church1991, Dagan *et al.*1993, Kupiec1993, Wu & Xia1994, Dagan & Church1994, Smadja & McKeown1993]. Note that all of the following algorithms, with the exception of

[2] Some of the work cited actually finds the correct word ordering in the translation, while others do not. It was argued that word alignment should only refer to those who deal with word ordering in the translation. However, we follow the convention in the literature of including word correspondence work in the alignment work.

[Dagan *et al.*1993, Dagan & Church1994], require clean, sentence-aligned parallel text input.

2.4. Word alignment

[Brown *et al.*1990, Brown *et al.*1993] are the first to use a stochastic sentence translation model. Estimation-Maximization is used to estimate the parameters for the model. Their model produces word alignment from clean, sentence-aligned parallel corpora during EM estimation. [Gale & Church1991] propose using mutual information and t-scores to find word correspondences as an alternative to the IBM model, noting that there is an explosive number of parameters to be estimated in the EM model when the vocabulary size is large. They use a progressive deepening method, starting with a small region of the training corpus and find the best pairs of words according to their mutual information and t-score. At subsequent iterations, the training sample is increased and more word pairs are found. This work laid the basis for other translation algorithms using correlation scores.

Yet another approach to word alignment is presented in [Dagan *et al.*1993]; they describe a tool, *word_align*, which uses the aligned output from *char_align* to further refine the alignment at the word level. Words are matched across the two languages within a window size around the anchor points found by *char_align*. The final matching between words is found by an EM-based learning method. This tool is an improvement on the IBM word alignment model because it only requires a rough initial alignment.

[Wu & Xia1994] computed a bilingual Chinese-English lexicon from a strictly one-to-one, sentence-aligned and selected bilingual corpus processed by the algorithm in [Wu1994]. The estimation of the word matching is an EM-based model. Various filtering techniques are used to improve the matching.

2.5. Phrase Translation

Others have translated phrases above the word level for inclusion in a bilingual lexicon or as an aid for translators. Translation of a group of words as a whole is important because idiomatic phrases, collocations, and technical terms often cannot be translated on a word by word basis. While all the work described in this section assumes a clean, parallel corpus as input, we have extended some of the ideas developed here for use with noisy corpora.

[Kupiec1993] uses as input a sentence-aligned, tagged French-English corpus, and produces a list of translated noun phrases. Simple noun phrases are extracted from both source and target halves of the corpus and grouped as single terms. The noun phrases which occur in the same sentences are considered to have strong correlations and thus translations of each other.

[Smadja & McKeown1993] provide a collocation translation algorithm using sentence-aligned, parallel texts as input. French words which are highly correlated with the English collocation are determined using the Dice coefficient and then the translation formed by iteratively identifying combinations of the French words which are highly correlated with the source collocation. Unlike the other work described within this section, their system, *Champollion*, translates both compounds (including noun phrases as well as other categories) and word pairs which may be separated by other words and can appear in different orders.

[Dagan & Church1994] use the output from *word_align* and a part-of-speech tagger[Church1988] to develop a tool for aiding human translators. *Termight* finds noun phrases in English and then uses *word_align* to align the head and tail words of the noun phrases to the words in the other language. The word sequence in the other language starting and ending with the aligned words of the English head and tail nouns is produced as the translated noun phrase. The fully implemented editor environment in which it is embedded enhances *Termight*'s practicality and efficiency as translator aid. Unlike other work described in this section, it can produce translations of low frequency terms. While its accuracy is somewhat low, it has proved quite useful as a translator's aid where the human translator can select the correct translation from a candidate list.

3. Noisy parallel corpora across language groups

As seen from the description of related work in bilingual word translation algorithms, the main weakness of these approaches is lack of robustness—against structural noise in parallel corpora, and against language pairs which do not share etymological roots.

There are many bilingual texts which are translations of each other but are not translated sentence by sentence. Some texts such as the AWK manual in English and its translation in Japanese do not contain the same programming examples or postscript figures. Their online versions have many mismatched sentences and paragraphs, suggesting a discontinuous mapping between the parallel texts. Most algorithms cannot deal with deletion and insertions in parallel corpora efficiently and therefore cannot be applied to such noisy corpora. In addition, texts scanned in by optical character recognition(OCR) tools are often visually noisy, with missing sentence boundaries or additional ink marks [Church1993]. Algorithms relying on clean sentence boundary cannot be applied to OCR inputs.

Another issue is language robustness. Tools like *char_align* [Church1993] do not rely on sentence boundaries. Instead, *char_align* aligns segments of English texts to French or other language texts by using shared character sequences between an English word and a French word as anchor points. Such tools can be applied to any pairs of languages which share a linguistic common root and therefore have the common cognate phenomena. However, such phenomena do not exist between, say English and Chinese, which are from different language groups and have completely different character sets.

We use two noisy parallel corpora for our experiments: (1) English version of the AWK manual and its Japanese translation[3], and (2) parts of the HKUST English-Chinese Bilingual Corpora for our experiments on noisy-parallel corpus.

The AWK manual was translated by a human translator into Japanese. Its English version contains 131,669 tokens including words and punctuations, special symbols for the AWK language, etc. As noted in the previous section, the Japanese version contains some English cognates which helped alignment in [Church *et al.*1993]. However, this is an incidental case. To test our algorithm for noisy corpora, this cognate information is not used. The Japanese version contains 95,695 tokens, including words, compound words, punctuations, and special symbols. The 1.37 ratio of the size of English text to that of the Japanese text is due to two factors: (1) Japanese texts do not have word boundaries and a morphological analyzer was used to insert these boundaries. More often than not, a single Japanese "word" can correspond to multiple words in English, such as *dehtasoosa/data-manipulation*; (2) there are many tables in the English text which are simply replaced by a pointer in the Japanese translation; (3) There are many text formatting special symbols which are different in the two language versions.

The HKUST English-Chinese Bilingual Corpora consists of transcriptions of the Hong Kong Legislative Council debates in both English and Chinese [Wu1994]. The topic of these debates varies though is to some extent confined to the same domain, namely the political and social issues of Hong Kong. While the corpus contains many formal and legal phrases, it also contains slang and many idiomatic expressions which are typical of transcribed, rather than written, texts. This to some extent a test of the

[3] Access to the AWK bilingual corpus is provided by Lucent Bell Laboratories.

algorithmic robustness since idioms are usually translated non-literally. We ran the algorithm on the transcriptions of nine days of Legislative Council debates consisting of 516,554 tokens in Chinese, 566,465 tokens in English. The tokens include words, punctuations and mark-up symbols. Noise in this case comes from three sources. First, similar to Japanese, the Chinese text was tokenized by a segmenter. There are many $n : m$ word matchings in the Chinese and English texts. Second, non-literal translations of slang and idioms cause discontinuities, including even translations of common words such as *christening* translated as 首次/*first time* where the latter is not always mapped to the former. Third, $n : m$ sentence correspondences such as shown in Figure 1. There are, for example, 408 Chinese sentences but 424 English sentences in the October 24th, 1992 transcriptions.

sentence-start
Hong Kong, as you know, now accounts for over 60% of total foreign investment in China and 36% of China's worldwide trade.
sentence-end
sentence-start
China accounts for 32% of our worldwide trade.
sentence-end

sentence-start
誠 如 各位 所 知 , 本 港 佔 中國 外來 投資 總 額 的 60% 以上 ,
同 時 , 亦 佔 中國 的 世界 貿易 的 36%,
相比 之下 , 中國 佔 本 港 的 世界 貿易 的 32% 。
sentence-end

Figure 1. An example of 2 to 1 sentence mapping in the HKUST corpus

4. Algorithm overview

We treat the domain word translation problem as a pattern matching problem—each word shares some common features with its counterpart in the translated text. We try to find the best representations of these features and the best ways to match them.

Our algorithm computes from no *a priori* information a small bilingual lexicon of high frequency words by a pattern matching method. The words from the source language are part of technical terms extracted. The algorithm then backtracks over the pattern matching paths, using them to yield high precision word positions as anchor points. These anchor points are highly reliable and take into account text deletion or insertion in a parallel corpus. The algorithm then iterates and recomputes a lexicon of both high and low frequency words by another type vector representation and pattern matching.

This algorithm bootstraps off of corpus alignment procedures we developed previously [Fung & Church1994, Fung & McKeown1994]. Our previous alignment procedures attempted to align texts by finding matching word pairs and we have demonstrated their effectiveness for Chinese/English and Japanese/English. The main focus then was accurate alignment, but the procedure produced a small number of word translations as a by-product. In contrast, our new algorithm performs a rough alignment, to facilitate compiling a much larger bilingual lexicon.

The paradigm for [Fung & Church1994, Fung & McKeown1994] is based on four main steps—find a small bilingual *primary lexicon*, use the text segments which contain some of the word pairs in the lexicon as anchor points for alignment, align the text, and compute a better *secondary lexicon* from these partially aligned texts. This paradigm can be seen as analogous to the Estimation-Maximization steps in [Brown *et al.*1991, Dagan *et al.*1993, Wu & Xia1994].

For a noisy corpus without sentence boundaries, the primary lexicon accuracy depends on the robustness of the algorithm for finding word translations given no *a priori* information. The reliability of the anchor points will determine the accuracy of the secondary lexicon. We also want an algorithm that bypasses a long, tedious sentence or text alignment step. So the purposes of our new algorithm are to compile a reliable primary lexicon and to use minimal anchor points for secondary lexicon estimation.

The outline of this algorithm is:

```
    Term translation aid:
    begin
1     Tag the English half of the parallel text.
2     Extract technical terms from the English text.
3     Tokenize the target Japanese and Chinese texts, and form a
      word list.
4     Compile an English word list for translation from technical
      terms.
5     Compile non-linear segment boundaries with high frequency
      word pairs.
      begin
        5.1 Compute the dynamic recency vector of each word
        in the word lists.
        5.2 Match pairs of recency vectors, giving scores.
        5.3 Select a primary lexicon using the scores with thresholds.
        5.4 Retrace word position points using the primary lexicon.
        5.5 Find most reliable anchor points by unsupervised filtering.
        5.6 Segment the bilingual texts by the anchor points.
      end
6     Compile bilingual word lexicon.
      begin
        6.1 Compute the non-linear segment binary vector of
        each word.
        6.2 Match binary vectors with correlation measure to yield
        a secondary lexicon.
      end
7     Suggest a word list for each technical term to the translator.
    end
```

In Section 5 and 6, we describe stages 1-4 where the corpus is pre-processed. Section 7, we describe the first four steps in stage 5 in our algorithm, cumulating in a primary lexicon for anchoring. Section 8 describes the next anchor point finding stage. Section 9 contains the procedure for compiling the secondary and final lexicon. Section 10 shows the result of this lexicon compilation. Section 11 shows how the secondary lexicon is used as translator-aid for term translations. Section 12 describes the effectiveness of this translator-aid.

5. Extracting technical terms from English text

As we are interested in finding domain-specific terms, we can concentrate our effort on those words which are part of such terms. For this purpose, we tagged the English part of the corpus by a modified POS tagger, extracted noun phrases which are most likely to be technical terms and apply our algorithm to find the translations for words which are part of these terms only.

We used a multilingual noun phrase bracketer, Brat [Fung *et al.*1996], which takes a syntactic regular expression, a tagged corpus, and produces a list of desired terms and a bracketed corpus. The regular expression for technical terms is based on Justeson and Katz's technical term finder [Justeson & Katz1995] which is $((A|N)+|((A|N)*(NP)?)(A|N)*)N$. They found that a technical term is usually a multi-word noun phrase, containing either a string of nouns and/or adjective, ending in a noun; or a concatenation of two such strings, separated by a preposition. The tagger used is described in [Church1988]. We made a few minor changes based on errors arising from interaction between POS and the regular expression. For example, we found that the POS tagger tends to tag possessive pronouns as adjectives and Brat would yield terms such as "his new idea". To discourage this, terms starting with possessive pronouns were subsequently not included for translation. We applied two other filters in the term extractor: (1) proper names starting with titles such as "Mr.", "Ms.", "Dr." are not considered as technical terms; (2) only terms with occurrence frequency above a threshold are considered. For the AWK corpus, the frequency threshold is zero, i.e. all technical terms are included in the list. It is seven for the HKUST corpus.

575 English technical terms were extracted from the AWK manual, a corpus containing 2212 unique words, and 95 were extracted after filtering from the HKUST corpus, which contains 3545 unique words. Note that frequency thresholding was used to select these technical terms from the full list, e.g. there were a total of 20,209 terms in the HKUST corpus. The large difference in the number of pre and post filtered English technical terms in the HKUST corpus shows that there are a huge number of rare terms (e.g. 16,703 terms with frequency one) and very many names and titles. The larger number of selected technical terms found in the AWK manual is due to the much lower frequency threshold. When compiling a technical word list from the AWK terms, another threshold was imposed; in this case, only words occurring more than once are considered. A 327 technical word list is obtained from the AWK corpus and a 626 technical word list is obtained from the HKUST corpus.

corpus	word list	English technical terms	selected technical terms	technical word list
AWK	2212	2949	575	327
HKUST	3545	20209	95	626

Figure 2. Technical terms and word list from both corpora

6. Tokenization of Chinese and Japanese texts

The target texts in Japanese and Chinese were only tokenized, not tagged. Tagging the target language and subsequently extracting technical terms from the target language can reduce the search space for the translation algorithm as only the words in these terms can become matches for the words in the English word list. This can be easily done if one has a reliable tagger and technical term extractor for the target language. However, both the Japanese tagger, JUMAN [Matsumoto & Nagao1994] and the Chinese tagger have a relatively low accuracy compared to the English tagger. Moreover, although we have developed a Japanese technical term extractor [Fung *et al.*1996], our effort in developing a Chinese technical term extractor is still in progress.

Our experimental results show that our translation algorithms yield reliable candidate words without yet specifying that those words need to be part of technical terms in the target language.

Tokenization of the Chinese text is done by using a statistically augmented dictionary-based tokenizer[Fung & Wu1994, Wu & Fung1994] which is able to recognize frequent domain words. This tokenizer is able to put character strings like 基本法/*Basic Law* together as a word. There are, however, still tokenization errors with low frequency words.

The Japanese text is tokenized by JUMAN without domain word augmentation. Tokenization errors occur mostly with such words in the AWK corpus.

7. A rough word pair based alignment

We treat translation as a pattern matching task where words in one language are the templates against which words in the other language are matched. The word pairs which are considered to be most similar by some measurement are taken to be translations of each other and form anchor points for the alignment. At stage 5 of our algorithm we have a noisy parallel corpus with no *a prior* sentence knowledge. Our task is to find a representation and similarity measurement which can find word pairs to serve as anchor points. These anchor points can then be used for a rough alignment.

7.1. Dynamic recency vectors

The similarity between patterns can be measured in a mathematically rigorous manner if the patterns are represented in a vector space [Rabiner & Juang1993]. We propose to represent the occurrence information of the words in vector forms. In looking at the text shown in Figure 3, we can see that words like *Governor* and 總督, which are translations of each other, may not necessarily occur within the same segments, nor do they occur at the same byte positions in the texts. To use an extreme case example, word A and its translation word B may occur in Chapters 1, 2 and 10 but they might not be in linearly corresponding byte segments. Chapters in the two parallel documents do not necessarily start or end at the same byte positions. This shows that the absolute byte position representation is not robust for measuring the smiliarity between two words occuring in the binary text.

However, if we define arrival interval to be the difference between successive byte positions of a word in the text, then Figure 3 shows that the arrival intervals are *more* similar between word pairs than their absolute positions. This property of recency vector representation is more robust for insertions and deletions than the absolute position representation. We will show in the following section how this representation facilitates noise-elimination in matching. Going back to our previous example, although the exact byte positions of word A in chapters 1, 2 and 10 is not similar to those of word B, the fact that they each occur 3 times first at a distance of 1 chapter and then at a distance of 8 chapters is significant. This arrival interval can be regarded as the recency information of words and can be used as a feature in the pattern matching of word pairs.

posi-tion	inter-val	sentence		
2380		MISS EMILY LAU :	Governor	- - I think I would
2390	10	would address you as	Governor	rather than Mr
2463	73	life will be protected .	Governor	, after reading these
2565	102	tion of human rights ,	Governor	, I could not find
2667	102	about other laws ,	Governor	? So will you please
2758	91	better to address me as	Governor	rather than Mr
3681	923	PETER WONG : Mr	Governor	, you have adopted a
4679	998	I do not think , as	Governor	of Hong Kong trying
5144	465	(in Cantonese) : Mr	Governor	, Meeting Point
5439	295	proposals because the	Governor	of Hong Kong and
90		席者 :	總督	彭 定 康 先
2021	1931	定 和 以往 歷任	總督	與 本 局 所
2150	129	問 (譯 文) :	總督	先生 (我 想
2158	8	(我 想 稱呼 你 為	總督	先生 較
2238	80	的 社會 。	總督	先生 , 讀 畢
2329	91	滿 。 同 時 ,	總督	先生 , 在 保
2416	87	歡迎 。 但	總督	先生 , 其他
2490	74	未來 ?	總督	答 (譯 文) :
2510	20	為 你 稱呼 我 為	總督	先生 較 主
3238	728	這 是 前任	總督	領導 下 的

Figure 3. Part of the concordances of the word *Governor* in English and Chinese

More concretely, the word positions of *Governor* form a vector $\langle 2380, 2390, 2463, 2565, ...\rangle$ of length 212. We compute the recency vector for *Governor* to be $\langle 10, 73, 102, 102, 91, 923, 998, ...\rangle$ with length 211. The position vector of 總督 is $\langle 90, 2021, 2150, 2158, 2238, ...\rangle$ of length 254. Its recency vector is $\langle 1931, 129, 8, 80, 91, 87, 74, 20, 728, ...\rangle$ with 253 as its length. The recency vectors of these two words have different lengths because their frequencies are different. If there is an insertion such as 總督 at position 90 (*Governor* starts at 2380), the corresponding recency vector would have a "glitch" at 1931. Such insertion and deletion noise, as we will show in the following section, is accounted for by our matching algorithm.

We can compute the recency vector for any word in the bilingual texts in this way. The values of the vector are integers with lower bound of 1, and upper bound being the length of the text.

Figuratively, for each word, we have a signal such as is shown in Figure 4 with the horizontal axis representing the word position in the text and the vertical axis the values of the recency vectors. Note the striking similarity of the two signals which represent *Governor* in English and Chinese. The word *Bill* in Chinese is not a translation of *Governor* and its signal is clearly uncorrelated with that of *Governor*. The signal for *President* is also very different. These signal patterns of the words can be used as the features for matching to give us most similar pairs.

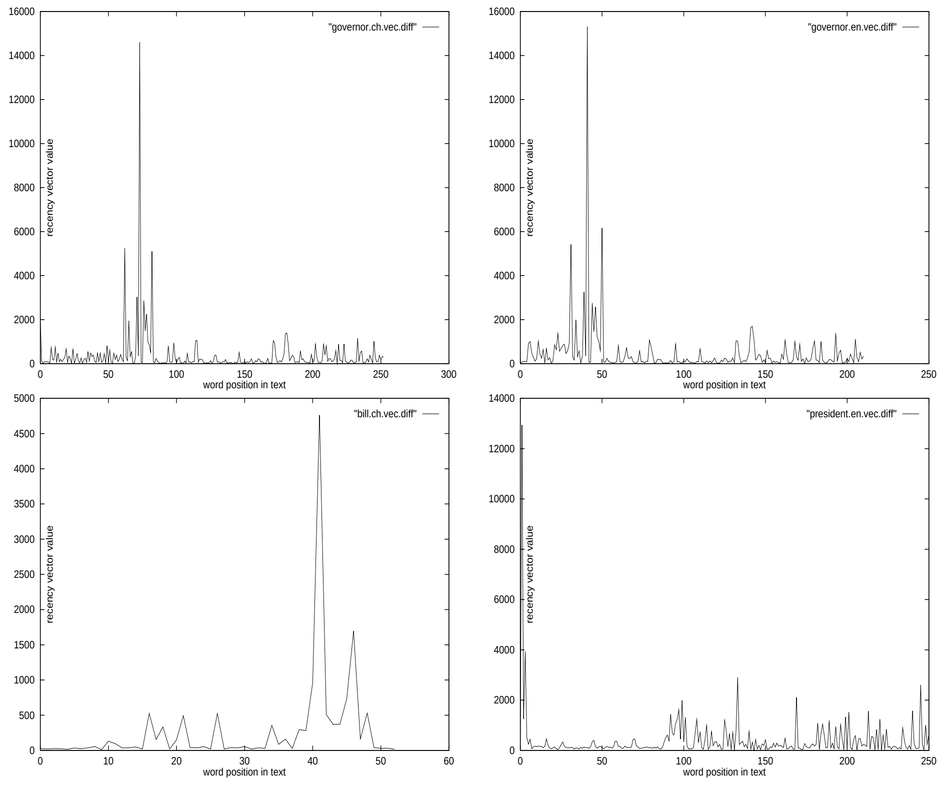

Figure 4. Recency vector signals showing similarity between *Governor* in English and Chinese, contrasting with *Bill* and *President* in English

Looking at the values of individual recency vectors, it is hard for us to tell which pairs are indeed similar to each other. However, the signals show that there is a characteristic shape of each vector which can be used for pattern matching. Just as it is hard for people tell if the spectral signals of two utterances represent the same word, so it is hard for humans to match the patterns of the recency vectors. But pattern matching has been successfully used in signal processing to match the spectral patterns for two words; we use a similar technique to "recognize" translation of words.

Thus, the recency vector representation assumes that if two words are translations of each other, they are more likely to occur a similar number of times at similar arrival intervals. Our next task is thus to determine what distance metric to use to measure the arrival intervals and do pattern matching.

7.2. Matching recency vectors

Many similarity measures could be used to compare vectors of variable lengths in one language against vectors in the other language. We need to use a measure which is both mathematically tractable and linguistically meaningful for the task. We propose using dynamic time warping (DTW), which has been used extensively for matching spectral signals in speech recognition tasks. DTW is a type of dynamic programming algorithm with embedded time normalization[4]. DTW takes two vectors of lengths N and M, finds an optimal path through the N by M trellis, starting from $(1,1)$ to (N,M).

DTW has the following implicit properties desirable for our segment alignment and word matching task:

– **Endpoint constraints**: the starting points and the end points of two vectors are considered as *a priori* anchor points. We assume that the noise due to insertion and deletion of paragraphs occurr mostly within the range between the start and end anchor points[5]. In our algorithm, the starting points are $V1[1]$ and $V2[1]$, the end points are $V1[N]$ and $V2[M]$ where n and m are the lengths of the two vetors.

– **Monotonicity**: our DTW algorithm only allows rightward (insertion),upward (deletion), and diagonally right-and-upward (alignment) running of the path. This ensures there is no cross-alignment between word pairs in a bilingual text, i.e. if the s-th occurrence of a word w_1 corresponds to the t-th occurrence of another word w_2, then the $(s+1)$-th occurrence of w_1 can only correspond to the $(t+i)$-th occurrence of w_2 where $i \geq 1$. We believe that there is very little cross-alignment between the same word pairs in bilingual texts and the monotonicity of our algorithm helps to make the computation tractable.

– **Normalization**: DTW score is the final running score normalized by the length of the path. Without normalization, DTW score for shorter vectors, i.e. words with lower frequencies, would always be better than that of longer vectors, thereby favoring words with lower frequencies. This is clearly not desirable for our task.

Consider the two vectors for *Governor*/總督, vector1 of length 212 and vector2 of length 254. If we plot vector2 against vector1, we get a trellis of length 212 and height 254. If the two texts were perfectly aligned translations of each other, then the interword arrival differences would be linearly proportional to the slope of the trellis, so that we could draw a straight line from the origin to the point (212, 254) to represent this correspondence between the two vectors. Since they are not, however, we need to measure the distortion of the one vector relative to the other. The way we do this is by incrementally computing

[4] The *time* axis in segment alignment refers to the offset of the text from its beginning.

[5] This constraint is not unlike the end point constraint in[Kay & Röscheisen1993]. DTW renders this constraint implicit.

the optimal path from the origin to point (212,254) as close as possible to the idealized diagonal. The divergence from the ideal gives a measure of the distortion. DTW traces the correspondences between all points in $V1$ and $V2$. When the path takes a rightward step, two words in one text are aligned to the same word in the other text, i.e. an insertion has occurred. When the path goes one step upward, a deletion. If it goes diagonally upward, two words in one text are aligned to two words in the other text. Each of these steps has an associated cost. This cost is dependent on the difference between the values of recency vectors at those points. The combined cost of insertion and deletion is greater than that of alignment. Moreover, DTW paths can be reconstructed during a backtracking step. The algorithm can then automatically choose the best points on these DTW paths as anchor points.

Our DTW algorithm is as follows:

–1 Initialization

Costs are initialized according to recency vector values:

$$\begin{aligned}
&\varphi(1,1)=\zeta(1,1)\\
&\varphi(i,1)=\zeta(i,1)+\varphi(i-1,1)\\
&\varphi(1,j)=\zeta(1,j)+\varphi(1,j-1)\\
\text{where}\ \ &\varphi(a,b)=\text{minimum cost of moving}\\
&\qquad\quad \text{from } a \text{ to } b\\
&\zeta(c,d)=|V1[c]-V2[d]|\\
&\ \text{for}\ \ i=2,3,\ldots,N\\
&\qquad\ \ j=2,3,\ldots,M\\
&\qquad N=\dim(V1)\\
&\qquad M=\dim(V2)
\end{aligned}$$

–2 Recursion

The accumulated cost φ of the DTW path is defined recursively:

$$\varphi(i,j)=\min\begin{cases}\varphi(i-1,j)+\zeta(i,j) & \text{rightward(insertion)}\\ \varphi(i-1,j-1)+\zeta(i,j) & \text{right-and-up}\\ \varphi(i,j-1)+\zeta(i,j) & \text{up(deletion)}\end{cases}$$

$$\xi(i,j)=\begin{cases}1 \text{ if } \varphi(i,j)=\varphi(i-1,j)+\zeta(i,j)\\ 2 \text{ if } \varphi(i,j)=\varphi(i-1,j-1)+\zeta(i,j)\\ 3 \text{ if } \varphi(i,j)=\varphi(i,j-1)+\zeta(i,j)\end{cases}$$

$$\begin{aligned}&\text{for}\ \ i=2,3,\ldots,N\\ &\text{and}\ \ j=2,3,\ldots,M\end{aligned}$$

–3 Termination

$$\varphi(N,M)=\min\begin{cases}\varphi(N-1,M)+\zeta(N,M) & \text{rightward}\\ \varphi(N-1,M-1)+\zeta(N,M) & \text{right-and-up}\\ \varphi(N,M-1)+\zeta(N,M) & \text{up}\end{cases}$$

$$\xi(N,M)=\begin{cases}1 \text{ if } \varphi(N,M)=\varphi(N-1,M)+\zeta(N,M)\\ 2 \text{ if } \varphi(N,M)=\varphi(N-1,M-1)+\zeta(N,M)\\ 3 \text{ if } \varphi(N,M)=\varphi(N,M-1)+\zeta(N,M)\end{cases}$$

Final cost of the DTW path is normalized by the length of the path:

$$\mathrm{DTW}(N,M)=\frac{\varphi(N,M)}{N+M}$$

–4 Path reconstruction

In our algorithm, we reconstruct the DTW path and obtain the points on the path for later use. The DTW path for *Governor*/總督 is as shown in Figure 5.

$$\begin{aligned}&\text{optimal path}=(l(I,J),l(p(I,J)),l(p(p(I,J))),\ldots,1)\\ &\text{where } p(i,j)=\begin{cases}(i-1,j) & \text{if } \xi(i,j)=1\\ (i-1,j-1) & \text{if } \xi(i,j)=2\\ (i,j-1) & \text{if } \xi(i,j)=3\end{cases}\\ &\qquad l(i,j)=(V1[i],V2[j])\\ &\qquad \text{for } \ i=2,3,\ldots,N\\ &\qquad \text{and } \ j=2,3,\ldots,M\end{aligned}$$

In step (2), to choose among the three directions, a distortion or distance function is computed. So to decide whether the path at (i,j) should go to (i+1,j), (i,j+1) or (i+1, j+1), we compute the absolute difference between V1[i+1] and V2[j], V1[i] and V2[j+1], and between V1[i+1] and V2[j+1]. The direction is determined by the smallest absolute difference among the three pairs.

As the path is constructed, the absolute differences at each step are accumulated into a running score which is finalized when the path reaches the point (212, 254). This score is then normalized by the total number of *steps* in the path. This is the score for the pair of vectors we compared. For every word vector in language A, the word vector in language B which gives it the lowest DTW score $DTW(V1[N],V2[M])$ is taken to be its translation.

We thresholded the bilingual word pairs obtained from above stages in the algorithm and stored the more reliable pairs as our primary bilingual lexicon.

7.3. Statistical filters

If we have to exhaustively match all source words against all target words in the corpus, the matching would be very expensive since it involves computing all possible paths between two vectors, and then backtracking to find the

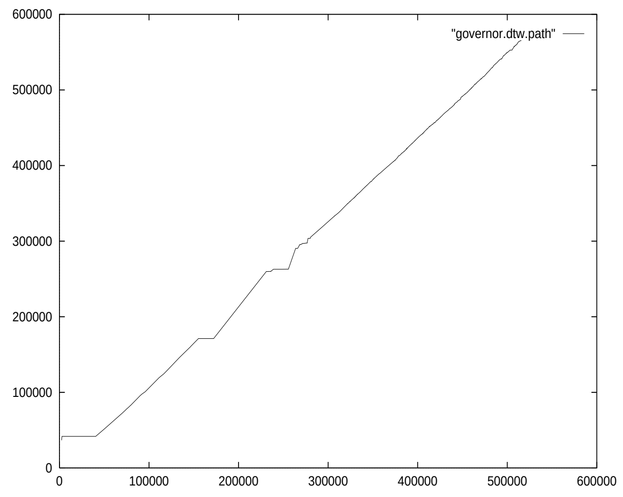

Figure 5. Dynamic Time Warping path for *Governor* in English and Chinese

optimal path, and doing this for all source/target word pairs in the texts. The complexity of DTW is $\Theta(NM)$ and the complexity of the matching algorithm is $\Theta(IJNM)$ where I is the number of technical words in the source text, J is the number of unique words in the target text, N is the occurrence count of one source word and M the occurrence count of one target word. To avoid this complexity we incorporated a number of heuristic constraints to filter the set of possible pairs:

– Starting point constraints - the value of the first dimension in each vector is its starting word position (minus zero). So vectors whose first value is at least half the text length apart are not considered in DTW since it means they start to occur at least half a text apart and cannot be translations of each other. This is a position constraint.

– Length constraint - the length of a vector is actually the frequency of that word minus 1. So vectors whose lengths vary by at least half the length of the vector represent cases where one word occurs at least twice as often as the other word; thus they cannot be translations of each other. They are not considered for DTW either. This is a frequency constraint.

– Means/standard deviation constraint - to improve the computation speed, we constrain the vector pairs further by looking at the Euclidean distance $\mathcal{E}$ of their means and standard deviations:

$$\mathcal{E} = \sqrt{(m_1 - m_2)^2 + (\sigma_1 - \sigma_2)^2}$$

If their Euclidean distance is higher than a certain empirically determined threshold, we filter the pair out and do not use DTW matching on them. This process eliminated most word pairs. Note that this Euclidean distance function helps to filter out word pairs which are very different from each other, but it is not discriminative enough to pick out the best translation of a word. So for word pairs whose Euclidean distance is below the threshold, we still need to use DTW matching to find the best translation. However, this Euclidean distance filtering greatly improved the speed of this stage of bilingual lexicon compilation.

8. Finding anchor points and eliminating noise

The primary lexicon is used for the sole purpose of aligning the segments in the corpus, in order to find a second and final lexicon. At stage 6 of our algorithm, we try to find anchor points on the DTW paths which divide the texts into multiple aligned segments for compiling the secondary lexicon. We believe these anchor points are more reliable than those obtained by tracing all the words in the texts.

For every word pair from this lexicon, we had obtained a DTW score and a DTW path. If we plot the points on the DTW paths of all word pairs from the lexicon, we get a graph as on the left hand side of Figure 6. Each point (i, j) on this graph is on the DTW path$\varphi(V1, V2)$ where $V1$ is from the source words in the lexicon and $V2$ is from the target words in the lexicon. The union effect of all these DTW paths shows a salient line approximating the diagonal. This line can be thought of the text alignment path. Its departure from the diagonal illustrates that the texts of this corpus are not identical nor linearly aligned. Note that the path for the AWK corpus is considerably more "crooked" than the one for the HKUST corpus, demonstrating that it is indeed a more noisy corpus.

There are superfluous points in this graph due to the rough lexical alignment in the first stage. Specifically, if a word w_s is part of one technical term and corresponds to w_{t1} in the target language, and it is part of another term in which it corresponds to w_{t2} in the target language. And if the frequency of w_{t1} is greater than that of w_{t2}, it is probable that w_s is translated to w_{t1} only by the DTW measure and positions in which w_s should correspond to w_{t2} would be mistaken by w_{t1}.

Previous alignment methods we used such as [Church1993, Fung & Church1994, Fung & McKeown1994] would bin the anchor points into continuous blocks for a rough alignment. This would have a smoothing effect. However, we later found that these blocks of anchor points are not precise enough for our Chinese/English corpus [Fung1995]. We found that it is more advantageous to increase the overall reliability of anchor points by keeping the highly reliable points and discarding the rest.

From all the points on the union of the DTW paths, we filter out the points by the following conditions: A point is considered reliable if it is not too far away from the diagonal with slope $= \frac{\text{length of source text}}{\text{length of target text}}$. If there is a noisy paragrah in one text which does not have any translation in the other text, words in this paragraph still have a chance to pass the DTW test. Erroneous corresponding word positions would be yielded in this case. However, the DTW paths of these erroneous word pairs would give points farther away from the diagonal line since the matched pairs were wrong. This slope constraint would eliminate these superfluous points. In addition, we impose the constraint that there should be no alignment in the text outside a window size of 500, i.e. a segment of the text cannot be the translation of another segment if they were 500 words apart. This constraint can be adjusted according to the noisiness of the corpus. If there can be a inserted paragraph of > 500 words, then the constraint can be relaxed. A continuity constraint is also imposed which means cross alignment is not allowed, i.e. given segments $i1, i2$ in text1, and $j1, j2$ in text2, where $i2 > i1, j2 > j1$, the alignment $i1 \rightarrow j2, i2 \rightarrow j1$ is not allowed. This is a more reasonable constraint for the HKUST English/Chinese corpus than for the AWK corpus since transcriptions of speech usually do not have cross-alignment whereas there is no guarantee that translated technical menus do not contain cross segment translation. This constraint can also be relaxed if cross-alignment is known to exist.

Finally, we only keep an anchor point (i, j) if it satisfies the following

(slope constraint)	$j/i <= N[1]/N[0] + \delta$
(continuity constraint)	$j - j_{previous} <= 0$
(window size constraint)	$i - i_{previous} <= 25$
(offset constraint)	$j - j_{previous} < 500$

Note that the above heuristic constraints are imposed on the *global* segment alignment path of the bilingual text, whereas those implicit constraints of DTW algorithm listed in the previous section are imposed on *local* pairs of word vectors. i and $i_{previous}$, j and $j_{previous}$ are neighboring points from the pre-filtered set of anchor points. So these constraints apply to all consecutive two points in the left graphs of Figure 6, whether any of the points are filtered out or not.

After filtering, we get points such as shown on the right hand side of Figure 6.

The constants in the above conditions are set roughly in proportion to the square root of the corpus size. As a result, the filtered picture looks close to a clean, diagonal line. This ensures that our development stage is still unsupervised. We would like to emphasize that if they were chosen by looking at the lexicon output as would be in a supervised training scenario, then one should evaluate the output on an independent test corpus.

9. Finding bilingual word pair matches

Stage 6 of our algorithm calls another procedure **Compile word translation** to obtain the secondary and final bilingual word lexicon. This section describes various steps in this procedure, including finding a non-linear K segment binary vector representation for each word, and using a similarity measure to compute word pair correlations.

9.1. Non-linear K segments

The anchor points $\langle (i_1, j_1), (i_2, j_2), \ldots, (i_k, j_k) \rangle$ divide a bilingual corpus into $k + 1$ non-linear segments where the n-th segment is between i_n, i_{n+1} in text1 and between j_n, j_{n+1} in text2, and where, in general, all three quantities $i_{n+1} - i_n$, $j_{n+1} - jn$, $i_n - i_{n-1}$ are different.

Since sentence boundaries are not required for this algorithm, each segment can contain more than one sentence, including part of a sentence. Likewise, each segment can contain part of a paragraph.

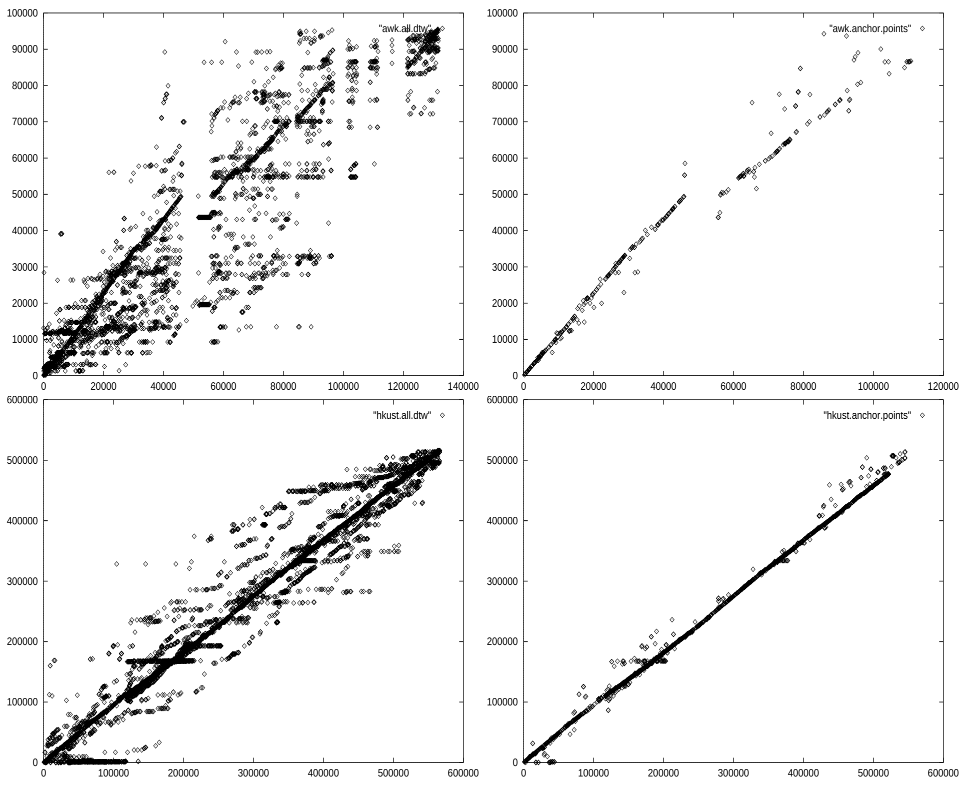

Figure 6. DTW path reconstruction output and the anchor points obtained after filtering

Note that if one chunk of noisy data appeared in text1 but not in text2, this part would be segmented between two anchor points (i, j) and (u, v). We know point i is matched to point j, and point u to point v, and thus the texts between these two points are matched but we do not make any assumption about how the texts of this segment are matched. In the extreme case where $i = u$, we would know that the text between j and v is noise. We have at this point a segment-aligned parallel corpus with noise elimination.

In the AWK corpus, there are 395 highly reliable anchor points. They divide the texts into 396 segments. The total length of the texts is around 100,000, so each segment has an average window size of 253 words which is considerably longer than a sentence length; thus this is a much rougher alignment than sentence alignment, but nonetheless we still get a bilingual lexicon out of it. There are 3647 anchor points in the HKUST corpus dividing the 550,000 word corpus into 150 words per segment on the average.

The algorithm then proceeds to obtain a secondary bilingual lexicon, considering words of both high and low frequency. To better evaluate our word translation algorithm, we consider words in all technical terms extracted from AWK and HKUST corpus, including those which occurred only once or twice. When translating terms, only English terms which occurred 7 times or more were considered for translation.

The key aspect of statistical bilingual lexicon compilation is to model the correlation between two events, namely the occurrences of a pair of translated words in a bilingual corpus. If the two appear together more often than they do with other words, then they are considered to be translations of each other.

Specifically, to compute the correlation between two words, represented by random variables w_s and w_t. We need to obtain the marginal probabilities $\Pr(w)$ of each occurring in the corpus and the joint probability of both of them $\Pr(w_s, w_t)$ occurring in the same place in the corpus. This same place is assumed to be the same aligned sentence in [Smadja *et al.*1996, van der Eijk1993, Kupiec1993, Wu & Xia1994]. In general it should be the same aligned *segment* in the corpus [Fung & McKeown1994, Dagan & Church1994, Shemtov1993].

A binary vector can be used to represent the presence of each word in the corpus. The binary vector has k dimensions where k is the number of segments in the corpus. A binary vector for a certain word has its i-th bit set to one if that word appears in the *ith* segment *at least once*, zero otherwise. For every given *pair* of source and target words to be matched, another co-occurrence binary vector is computed where the i-th bit is set to one if *both* words are found in the i-th segment.

The binary vector notation can be illustrated by the following contingency tables using the English word *information* and its Japanese translation as an example. In the table, (a) represents the number of segments where both English and Japanese words were found, (b) shows the number of segments where just the English word was found, (c) is the number of segments where just the Japanese word was found, and (d) is the number of segments where neither word was found:

	Japanese	
English	a	b
	c	d

Figure 7. A contingency matrix

In general, if the source and target words are good translations of one another, then a should be large, and b and c should be small. In contrast, if the two words are not good translations of one another, then a should be small, and b and c should be large.

The relationship between a binary segment vector and a binary sentence vector is as follows:

(1) $$\Pr(w_s = 1) = \frac{a+b}{a+b+c+d}$$

(2) $$\Pr(w_t = 1) = \frac{a+c}{a+b+c+d}$$

(3) $$\Pr(w_s = 1, w_t = 1) = \frac{a}{a+b+c+d}$$

where a=number of sentences where both words occur
b=number of sentences where only w_s occur
c=number of sentences where only w_t occur
d=number of sentences where neither words occur

(4)

In the above formula, note that corpus size is defined as the number of sentences in a text, and that number of sentences in text1 is equal to those in text2. Furthermore, segment boundaries are defined as sentence boundaries.

In a noisy parallel corpus, segment boundaries are not equivalent to sentence boundaries and the number of sentences in text1 is not equal to the number in text2. Second, there can be a different number of sentences on each side in a noisy parallel corpus because there are inserted or deleted paragraphs.

To resolve these problems, we use the following definitions. First, although we do not know the sentence boundaries, we can delimit segment boundaries by using anchor points obtained from step 5. Second, we have to redefine corpus size to be the number of segments in a corpus. Texts in both languages are divided into equal segments and we align the same number of segments in the noisy parallel corpus. There is a one-to-one alignment between the segments. Third, since we are interested in finding the correlation between a bilingual word pair, the fact that they co-occur in the same segments is the important part. We do not care as much how often they each occur in the particular segment where they co-occur. In the extreme case, if a word co-occur in 1,000 segments, even though one word appears 10 times in each of these segments, the other one occurs only once in each of the 1,000 segments, we still believe they have certain correlations if they are the only pair co-occurring in 1,000 segments. So, as is the case in algorithms using sentence aligned corpora, we propose using a *smoothed segment binary vector*, i.e. the i-th bit of this vector is one if the word it represents occurs *at least once* in that segment. Our experimental results show that this last assumption holds quite well if the size of segments are not too large.

The various probabilities are then modeled as in equations 1 to 3, substituting segments for sentences. The segment delimiters in our algorithm were set by the anchor points obtained from previous stage 5. The 395 anchor points $(223, 225), (436, 459), \ldots, (110721, 86737)$ divide the two texts into 396 non-linear segments. Text1 is segmented by the points $(223, 436, \ldots, 110401, 110721)$ and text2 is segmented by the points $(225, 459, \ldots, 86591, 86737)$.

For the words we are interested in finding the translations for, we again look at the position vectors. For example, the word *information* occurred 35 times in the English text. Its position vector is $\langle 1868, 29248, \ldots, 91990, 130999 \rangle$. We convert this position vector into a binary vector $V1$ of 395 dimensions where $V1[i] = 1$ if *information* occurred within the *ith* segment, $V1[i] = 0$ otherwise. For *information*, $V1[i] = 1$ where $i = 21, 205, 217, \ldots, 373, 395$. The Japanese translation for *information* is *joohoo*. Its position vector is $\langle 2081, 8186, \ldots, 93010, 94903 \rangle$. Its binary vector is $V2[i] = 1$ where $i = 21, 78, 205, \ldots, 395$. These two vectors share 15 segments in common. Their contingency table is as follows:

	joohoo	
information	15	9
	9	362

Figure 8. information in English and Japanese

We compute the segment vector for all words in the English word list whose frequency is above two. Words occurring only a few times are extremely hard to translate although our algorithm was able to find some rare word pairs.

9.3. Binary vector correlation measure

To measure the correlation between a word w_s in the source text and another w_t in the target text, a similarity measure needs to be formulated. Various similarity measures have been proposed for different applications, such as mutual information average mutual information, mutual information with t-scores (which we used in an earlier version of this algorithm) [Fung1995, Fung & McKeown1994, Fung & Church1994] and the Dice coefficient [Smadja *et al.*1996]. It was convincingly argued in [Smadja *et al.*1996] that average mutual information is not suitable for bilingual lexicon translation tasks since it grants the same significance to the case where neither words occur as to the case where they co-occur. For translation tasks, the information about how often two words do not co-occur is trivialized by the fact that there are a large number of words in the corpus and thus, this information should be disregarded. The Dice corrects for this by ignoring the d term (i.e., the number of segments/sentences where neither word occurs) and we would like to retain this feature in the correlation metric used here.

However, most metrics require some form of frequency thresholding as they do not produce accurate scores for low frequency words. Thus with the Dice, all words with frequency below a given threshold, usually 5 to 10, are filtered and not translated. This is an approximation of a confidence threshold. A t-score in conjunction with mutual information performs a similar function. This is not practical for our experiments given that our corpora are relatively small compared to those used by others. In [Smadja *et al.*1996], one year of the Hansards was used. In our experiments, we only have nine days of transcription from the HKUST corpus and The AWK manual is even smaller. Thus, we cannot afford to filter out *too many* low frequency words. For our goal of compiling bilingual word pairs to aid translators to get the terms, we need to translate all words in the English word list, whether they are rare or not. As a result, we cannot use either the t-score or the Dice coeffecient with frequency thresholding at 5. Consequently, we have compromised by taking the threshold to be one in the AWK corpus, i.e. if there are technical words which occurred once, we do not consider them. This way, we preserve some rare words, but weigh the correlation score by their relative frequency to the number of segments in the corpus.

Thus, we propose another similarity measure, **weighted mutual information**:

$$\begin{aligned} W(w_s, w_t) &= \Pr(w_s = 1, w_t = 1) \log_2 \frac{\Pr(w_s = 1, w_t = 1)}{\Pr(w_s = 1)\Pr(w_t = 1)} \\ &= \frac{a}{a+b+c+d} \log_2 \frac{a \cdot (a+b+c+d)}{(a+b) \cdot (a+c)} \end{aligned}$$

Weighted mutual information is the first term in the average mutual information score, and also corresponds to the mutual information score multiplied by the joint probability of the co-occurring words. Unlike Dice, the d term is not completely ignored. However, when the marginal probabilities are low (i.e. when $a+b$ or $a+c$ are low which makes the mutual information high) weighted mutual information offsets the effect by having a low weight $\frac{a}{a+b+c+d}$.

To compare the various similarity measures for our task, we tested the Dice coefficient, average mutual information and the weighted mutual information, all with frequency threshold of one discarding words which occurr only once, on the AWK corpus. Of the 327 technical words in the English word list, some of them occur only once in the Japanese text. Others, such as the special text mark-up symbol *font* do not appear in the Japanese text at all. Our algorithm computes the correlation scores between the technical English words and all Japanese words in the corpus and outputs a best-candidate list ranked by the correlation scores. So each technical English word has one best Japanese word candidate. Only direct and obviously correct translations are judged to be a correct pair. Word pairs which could be part of the same term are not judged to be correct in this evaluation. For example, *information/joohoo* is a direct and correct translation, but *awk/programming* is not correct.

10. Word translation results

Using various similarity measures on the AWK corpus, we obtain word translation results as shown in Figure 9.

Here the results show a better performance of using the Dice coefficient over using the average mutual information, and an even better performance when the weighted mutual information is used. The low precision rate from all three measures is due to (1) low frequency words including English words without Japanese correspondence; (2) error in the English technical word extraction, (3) error in tokenization of the Japanese words, and (4) word pairs as part of a collocation are not considered.

If we divide the list into different regions according to similarity score ranking, we observe that the wrong translations occur at different places on

measure	correct	precision
weighted mutual information	181/327	55.35%
Dice coefficient	156/327	47.7%
average mutual information	109/327	33.3%

Figure 9. AWK bilingual word lexicon results

the ranked list when different similarity measures are used. Precision decreases as more word pairs with lower scores are included in the list. This is shown in Figure 10. The horizontal axis indicates the first m source words with their translations according to the similarity measure. The vertical axis shows the precision if only these m words are translated.

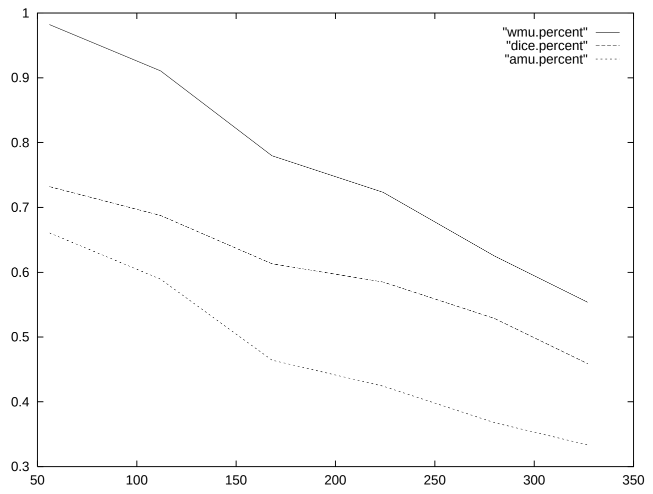

Figure 10. AWK results from different similarity measures

From Figure 10, the weighted mutual information shows an advantage over the other two measures. Using weighted mutual information, the precision range is from 98.2% to 55.3%. Both overall and local precisions on the AWK corpus are better than those of the other two. However, further experimentation is needed to compare the Dice coefficient and the weighted mutual information with larger corpora, and with a frequency threshold.

In a subsequent experiment with the HKUST corpus, we chose the weighted mutual information score as the similarity measure. The word translation result for the technical words in HKUST corpus is 89.93%. Its precision by ranking is shown in Figure 11. The precision rate averages at 95% with different score thresholding.

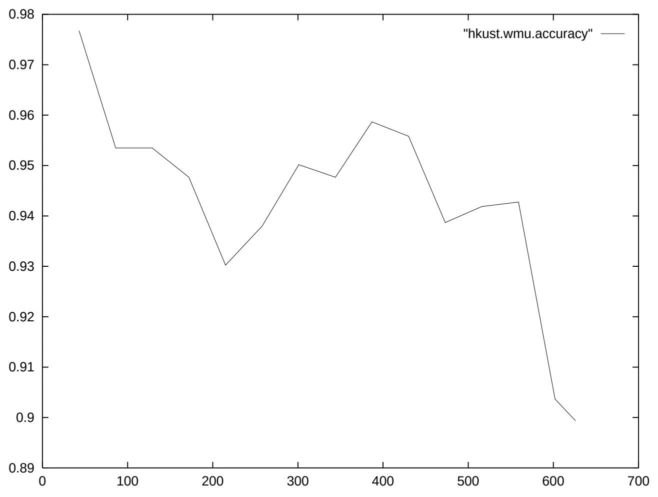

Figure 11. HKUST bilingual word lexicon results

11. Term translations from word groups

In our word translation algorithm, only the best candidate is kept and only direct translation is judged correct. However, in devising the algorithm for helping translators to translate technical terms, there are cases when an incorrect translation of a word is a part of a correct translation of the term containing the word. We describe some of the cases where the translated word pair suggests a potential element of term translation and thus, we actually should keep the candidate translation words:

– **finding Chinese words**: Chinese texts do not have word boundaries such as space in English. Therefore, our text was tokenized into words by a statistical Chinese tokenizer [Fung & Wu1994]. Tokenizer errors caused some Chinese characters not to be grouped together as one word. Our program located some of these words. For example, *Green* was aligned to 綠, 皮 and 書 which suggests that 綠皮書 could be a single Chinese word. It indeed is the name for Green Paper—a government document.

– **compound noun translations**: *carbon* could be translated as 碳, and *monoxide* as 一氧化. If *carbon monoxide* were translated using the individual word translation, we would get 碳 一氧化 . However, our algorithm found both *carbon* and *monoxide* to be most likely translated to the single Chinese word 一氧化碳 which is the correct translation for *carbon monoxide*.

  The words *Legislative* and *Council* were both matched to 立法 and similarly we can deduce that Legislative Council is a compound noun/collocation. The interesting fact here is, *Council* is also matched to 局. So we can deduce that 立法局 should be a single Chinese word corresponding to *Legislative Council*.

– **collocations**: Some word pairs such as *collage* and *electoral* are not direct translations. However, they are both constituent words of the collocation *Grand Electoral College*. Both *Cross* and *Harbour* are translated to 海底(*sea* *bottom*), and then to 隧道(*tunnel*), not a very literal translation. Yet, the correct translation for 海底隧道 is indeed *the Cross Harbor Tunnel* and not *the Sea Bottom Tunnel. Basic* and *Law* are both matched to 基本法, so we know the correct translation for 基本法 is *Basic Law* which is a compound noun.

–**proper names** Many proper names and titles are translated correctly with multiple word mapping. However, these are not considered for evaluation since the evaluators need to be familiar with both the English and Chinese names of the Legislative Council members. For example *Lau* has the English name *Emily*, but her Chinese name is not a transliteration of *Emily* at all.

Given a technical term S in English, our algorithm outputs the top five candidate words for each source word in the term into a set T. If there are n words in S, there would be $n \times 5$ words in T. There is a correlation score between the $n \times 5$ target words with the source words. This forms a list of $n \times 5$ entries. This list is then sorted according to the scores. Here, the top candidates are the best words for the whole term S. Since many of the $n \times 5$ target words are repeated, we discard the repeated words. Now the list can contain anywhere between five words (in the case the same five words are the top candidates for each of the n target words), to $n \times 5$ words (in the case where all top five candidates are different words). The output from the HKUST corpus actually yields candidate word lists of length five to thirteen.

Examples of set T ranked to the correlation scores are shown in Figure 12. Examples of candidate word list are shown in Figure 13. The # sign shows the correct translation word.

12. Term translation aid result

We asked three people to evaluate the candidate word list for each of the 95 English technical terms. These evaluators are all native speakers of mandarin with bilingual knowledge in English and Chinese. They have some knowledge of the political and social issues in Hong Kong which is the content of these technical terms although none of them know the Chinese translations of these terms as well as a native resident from Hong Kong would. They are asked to be the users of the translation output. First, they are asked to translate all English terms by themselves and output a translation list SET A. Secondly, they are asked to compile the term translation from the suggested word lists, and output a list SET B. Both SET A and B are compared to the actual translation obtained from the corpus. If the translation is the same as in the corpus, then it is judged as correct. Figure 14 shows the output for SET A and B. It shows that evaluators on average are able to translate 30 terms out of 95 by themselves, whereas they can translate 75 terms on average with aid. This evaluation shows that translation precision is improved on an average by 47%.

Obviously, the evaluators' knowledge of the language and familiarity with the domain influenced the results. For example, evaluator A had some exposure to the domain terms previously and scored higher in SET A, but evaluator B got the most help from the suggested list. In general, the evaluators know the correct translation for terms like *Basic Law* because it is frequently mentioned in the press and the news. On the other hand, the evaluators could not

English term	score	English word	Chinese word
Basic Law	0.0638997	Basic	基本法#
	0.0498881	Law	基本法#
	0.0174354	Law	選舉
	0.0155637	Law	九
	0.0149477	Law	中
	0.0146798	Basic	選舉
	0.0136949	Basic	九
	0.0128208	Law	年
	0.0120095	Basic	中
	0.0109621	Basic	年
Joint Declaration	0.0274651	Joint	聯合#
	0.0271423	Declaration	聲明#
	0.0266034	Joint	聲明
	0.0260613	Declaration	聯合
	0.01276	Joint	英
	0.0123645	Declaration	英
	0.00964212	Joint	中
	0.00956897	Declaration	中
	0.00856835	Declaration	基本法
	0.00827709	Joint	基本法
Secretary for Transport	0.0698721	SECRETARY	司#
	0.0605514	SECRETARY	文
	0.0582695	SECRETARY	譯
	0.0451182	SECRETARY	)
	0.0445229	SECRETARY	(
	0.0110008	TRANSPORT	運輸#
	0.00976618	TRANSPORT	隧道
	0.00743703	TRANSPORT	空氣
	0.00589024	TRANSPORT	答
	0.00575569	TRANSPORT	司

Figure 12. Example candidate word list (# shows correct translation)

come up with a consistent translation for *Joint Declaration* by themselves but invariably chose the correct translation from the suggested list. Translation of these domain specific terms are greatly improved by our system. As another example, *Education Department*, although not a rare term, has different offi-

English term	Chinese word
Basic Law	基本法#
	選舉
	九
	中
	年
Joint Declaration	聲明#
	聯合#
	英
	中
	基本法
Secretary for Transport	司#
	文
	譯
	(
	)
	運輸#
	隧道
	空氣
	答

Figure 13. Example candidate word list (# shows correct translation)

Evaluator	SET A	without aid	SET B	with aid	improvement
A	38/95	40%	76/95	80%	40%
B	22/95	23%	75/95	79%	56%
C	29/95	31%	73/95	77%	46%

Figure 14. Translator improvement on term translation

cial names in China and Hong Kong. The evaluators again chose the correct translation from the suggested list although none gave the right translation in SET A.

We believe this evaluation scheme is a realistic one for our translator-aid program. Most translators, like our evaluators, would be able to translate some relatively familiar terms but would have difficulties with other more regional and domain-dependent terms.

13. Future work

We can extend our system to be a fully automatic technical term translation system without human intervention. As seen from the output of our algorithm, most constituents words of the target term are found among the top five candidate words for each of the constituent words in the source term. In [Smadja *et al.*1996], individual constituent words of the target term are found by an iterative process which discards unlikely candidate words at each step. Word order inside a term is determined from rigid collocations in the parallel corpus. A similar process can be applied to our candidate word list to obtain full term translation.

We have discovered that in a word translation scheme without frequency thresholding, weighted mutual information seems to give better performance over other similarity measures. However, we need a better understanding of their behavior in different circumstances. We plan to compare these measures when a frequency threshold is applied to filter out rare words. We also need to demonstrate a theoretical rationale for using weighted mutual information as similarity measure.

Part of our algorithm can also be used as a text-based alignment step. The initial anchor points and primary bilingual lexicon can be used to align large bilingual texts can by segments. Other word-alignment or term translation methods can be used on this bootstrapping step to obtain a larger bilingual lexicon.

14. Conclusion

We have shown techniques to align noisy parallel corpora by segments, and to extract a bilingual word lexicon from it. The bilingual lexicon is used to help translators to compile technical term translations which they would not be able to translate otherwise. Our algorithm yields a 55.35% precision in technical word translation for the AWK corpus from English to Japanese, and a 89.93% precision for the HKUST corpus from English to Chinese. Our algorithm produces a suggested word list for each technical term in the HKUST corpus and improved human translator performance by an average of 47%.

Our algorithm substitutes the sentence alignment step with a rough segment alignment to find a bilingual lexicon of technical words. It shows promise for compilation of domain-specific, technical and regional terms. It has shown effectiveness in computing such a lexicon from texts with no sentence boundary information and with noise. Instead, it uses highly-reliable anchor points that are derived using DTW to serve as segment delimiters. Compared to other word alignment algorithms, it does not need *a priori* information such as pre-compiled lexicon, specific punctuations in the text or sentence boundary markers. It is independent of character sets and sentence structure and therefore is applicable to different language pairs. It has also shown promise for machine translation of technical terms into terms in the target language.

Our algorithms eliminate the need to manually clean up large parallel corpora by deleting unparallel paragraphs and segments, as well as the need for precise, clear sentence boundary markers. This saves large amounts of manual labor and time. Many OCR texts can therefore be used as input, increasing the amount of useful bilingual corpora for our application.

Word feature representations such as the recency vector and smoothed segment binary vector used in this algorithm capture the frequency, position and recency information of words in noisy parallel texts. These features are more robust than conventional sentence-based word position vectors in that they can be obtained despite many mismatching sentences and segments in noisy parallel corpora.

The effectiveness of our translator aid is rather remarkable given that by simply using a 5-best candidate list for each word in the source term, the translators are able to improve by 47% their translation of technical terms they are not familiar with.

15. Acknowledgment

This research was partially supported by a joint grant from the Office of Naval Research and the Defense Advanced Research Projects Agency under contract # N000 14-95-1-0745 and by National Science Foundation Grant GER-90-2406. The authors would like to thank Min-Yen Kan for implementing the term extractor, and the evaluators Pan Shimei, Jing Hong-Yan and Michelle Zhou. We would like to thank Ken Church, Ido Dagan, Vasileios Hatzivassiloglou and Dekai Wu for useful discussions. We would also like to thank Ken Church for making available some of his software tools, Dekai Wu for the HKUST bilingual corpus, and Brian Kernighan and Lucent Bell Laboratories for providing the AWK bilingual corpus. Many thanks to the anonymous reviewers who have given us very valuable comments.

References

Aho *et al.*1980. AHO, A., B. KERNIGHAN, & P. WEINBERGER. 1980. *The AWK programming language.* Addison-Wesley, Reading, Massachusetts, USA.

Brown *et al.*1991. BROWN, P., J. LAI, & R. MERCER. 1991. Aligning sentences in parallel corpora. In *Proceedings of the 29th Annual Conference of the Association for Computational Linguistics.*

Brown *et al.*1990. BROWN, P.F., J. COCKE, S.A. DELLA PIETRA, V.J. DELLA PIETRA, F. JELINEK, J.D. LAFFERTY, R.L. MERCER, & P. ROOSIN. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16:79–85.

Brown *et al.*1993. BROWN, P.F., S.A. DELLA PIETRA, V.J. DELLA PIETRA, & R.L. MERCER. 1993. The mathematics of machine translation: Parameter estimation. *Computational Linguistics*, 19(2):263–311.

Chen1993. CHEN, STANLEY. 1993. Aligning sentences in bilingual corpora using lexical information. In *Proceedings of the 31st Annual Conference of the Association for Computational Linguistics*, 9–16, Columbus, Ohio.

Church *et al.*1993. CHURCH, K., I. DAGAN, W. GALE, P. FUNG, J. HELFMAN, & B. SATISH. 1993. Aligning parallel texts: Do methods developed for English-French generalize to Asian languages? In *Proceedings of Pacific Asia Conference on Formal and Computational Linguistics.*

Church1988. CHURCH, KENNETH. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, 136–143, Austin, Texas.

Church1993. CHURCH, KENNETH. 1993. Char_align: A program for aligning parallel texts at the character level. In *Proceedings of the 31st Annual Conference of the Association for Computational Linguistics*, 1–8, Columbus, Ohio.

Dagan & Church1994. DAGAN, IDO & KENNETH W. CHURCH. 1994. Termight: Identifying and translating technical terminology. In *Proceedings of the 4th Conference on Applied Natural Language Processing*, 34–40, Stuttgart, Germany.

Dagan *et al.*1993. DAGAN, IDO, KENNETH W. CHURCH, & WILLIAM A. GALE. 1993. Robust bilingual word alignment for machine aided translation. In *Proceedings of the Workshop on Very Large Corpora: Academic and Industrial Perspectives*, 1–8, Columbus, Ohio.

Fung1995. FUNG, PASCALE. 1995. A pattern matching method for finding noun and proper noun translations from noisy parallel corpora. In *Proceedings of the 33rd Annual Conference of the Association for Computational Linguistics*, 236–233, Boston, Massachusettes.

Fung & Church1994. FUNG, PASCALE & KENNETH CHURCH. 1994. Kvec: A new approach for aligning parallel texts. In *Proceedings of COLING 94*, 1096–1102, Kyoto, Japan.

Fung *et al.*1996. FUNG, PASCALE, MIN-YEN KAN, & YURIE HORITA. 1996. Extracting Japanese domain and technical terms is relatively easy. In *Proceedings of the Second International Conference in New Methods for Language Processing*, Bilkent, Turkey. To appear.

Fung & McKeown1994. FUNG, PASCALE & KATHLEEN MCKEOWN. 1994. Aligning noisy parallel corpora across language groups: Word pair feature matching by dynamic time warping. In *Proceedings of the First Conference of the Association for Machine Translation in the Americas*, 81–88, Columbia, Maryland.

Fung & Wu1994. FUNG, PASCALE & DEKAI WU. 1994. Statistical augmentation of a Chinese machine-readable dictionary. In *Proceedings of the Second Annual Workshop on Very Large Corpora*, 69–85, Kyoto, Japan.

Gale & Church1991. GALE, WILLIAM & KENNETH CHURCH. 1991. Identifying word correspondences in parallel text. In *Proceedings of the Fourth Darpa Workshop on Speech and Natural Language*, Asilomar.

Gale & Church1993. GALE, WILLIAM A. & KENNETH W. CHURCH. 1993. A program for aligning sentences in bilingual corpora. *Computational Linguistics*, 19(1):75–102.

Justeson & Katz1995. JUSTESON, JOHN S. & SLAVA M. KATZ. 1995. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1:9–27.

Kay & Röscheisen1993. KAY, MARTIN & MARTIN RÖSCHEISEN. 1993. Text-Translation alignment. *Computational Linguistics*, 19(1):121–142.

Kupiec1993. KUPIEC, JULIAN. 1993. An algorithm for finding noun phrase correspondences in bilingual corpora. In *Proceedings of the 31st Annual Conference of the Association for Computational Linguistics*, 17–22, Columbus, Ohio.

Matsumoto & Nagao1994. MATSUMOTO, YUJI & MAKOTO NAGAO. 1994. Improvements of japanese morphological analyzer JUMAN. In *Proceedings of the International Workshop on Sharable Natural Language Resources*, 22–28.

Rabiner & Juang1993. RABINER, LAWRENCE & BING-HWANG JUANG. 1993. *Fundamentals of speech recognition*. Signal Processing Series. Englewood Cliffs, NJ: Prentice-Hall.

Shemtov1993. SHEMTOV, H. 1993. Text alignment in a tool for translating revised documents. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, 449–453, Utrecht, The Netherlands. Association for Computational Linguistics.

Simard *et al.*1992. SIMARD, M., G FOSTER, & P. ISABELLE. 1992. Using cognates to align sentences in bilingual corpora. In *Proceedings of the Forth International Conference on Theoretical and Methodological Issues in Machine Translation*, Montreal, Canada.

Smadja & McKeown1993. SMADJA, FRANK & KATHLEEN MCKEOWN. 1993. Translating collocations for use in bilingual lexicons. In *Proceedings of the ARPA Human Language Technology Workshop 94*, Plainsboro, New Jersey.

Smadja *et al.*1996. SMADJA, FRANK, KATHLEEN MCKEOWN, & VASILEIOS HATZ-SIVASSILOGLOU. 1996. Translating collocations for bilingual lexicons: A statistical approach. *Computational Linguistics*, 21(4). To appear.

van der Eijk1993. VAN DER EIJK, P. 1993. Automating the acquisition of bilingual terminology. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics*, Utrecht, The Netherlands.

Warwick-Armstrong & Russell1990. WARWICK-ARMSTRONG, S. & G. RUSSELL. 1990. Bilingual concordancing and bilingual lexicography. *Euralex*.

Wu1994. WU, DEKAI. 1994. Aligning a parallel English-Chinese corpus statistically with lexical criteria. In *Proceedings of the 32nd Annual Conference of the Association for Computational Linguistics*, 80–87, Las Cruces, New Mexico.

Wu & Fung1994. WU, DEKAI & PASCALE FUNG. 1994. Improving Chinese tokenization with linguistic filters on statistical lexical acquisition. In *Proceedings of the 4th Conference on Applied Natural Language Processing*, 180–181, Stuttgart, Germany.

Wu & Xia1994. WU, DEKAI & XUANYIN XIA. 1994. Learning an English-Chinese lexicon from a parallel corpus. In *Proceedings of the First Conference of the Association for Machine Translation in the Americas*, 206–213, Columbia, Maryland.

Address for correspondence:
Pascale Fung
Computer Science Department
Columbia University
New York, NY 10027
pascale@cs.columbia.edu