# A Minimally Supervised Approach for Detecting and Ranking Document Translation Pairs

**Kriste Krstovski**
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003, USA
kriste@cs.umass.edu

**David A. Smith**
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003, USA
dasmith@cs.umass.edu

## Abstract

We describe an approach for generating a ranked list of candidate document translation pairs without the use of bilingual dictionary or machine translation system. We developed this approach as an initial, filtering step, for extracting parallel text from large, multilingual—but non-parallel—corpora. We represent bilingual documents in a vector space whose basis vectors are the overlapping tokens found in both languages of the collection. Using this representation, weighted by tf·idf, we compute cosine document similarity to create a ranked list of candidate document translation pairs. Unlike cross-language information retrieval, where a ranked list in the target language is evaluated for each source query, we are interested in, and evaluate, the more difficult task of finding translated document pairs. We first perform a feasibility study of our approach on parallel collections in multiple languages, representing multiple language families and scripts. The approach is then applied to a large bilingual collection of around 800k books. To avoid the computational cost of $O(n^2)$ document pair comparisons, we employ locality sensitive hashing (LSH) approximation algorithm for cosine similarity, which reduces our time complexity to $O(n \log n)$.

## 1 Introduction

A dearth of parallel data has been, and still is, a major problem for developing highly reliable statistical machine translation systems in many languages and domains. There have been many proposed approaches for alleviating this problem by utilizing techniques for creating and extracting parallel documents, sentences or phrases from comparable bilingual data available on the open web (Resnik and Smith, 2003), such as Wikipedia articles (Smith et. al, 2010), to name a few, or through digitized archives from various sources (Zhao and Vogel, 2002), (Munteanu and Marcu, 2005).

In general, in the process of utilizing comparable corpora to obtain sentence-aligned bilingual text, the first step involves performing initial filtering where text entities from both language collections are compared to each other and based on comparison score they are matched and grouped as potential translation candidate pairs. After this initial step, text entity pairs or tuples are further analyzed in order to extract parallel sentence pairs. In this paper we only focus on this initial step. We present a novel exploration of approaches that retrieve actual document translation pairs without the use of any bilingual resources such as lexicons or sentence aligned bitext.

Rather than solving separate retrieval or translation problems for each source language document, we retrieve translation pairs from the space of all possible bilingual document pairs. Most machine

*Proceedings of the 6th Workshop on Statistical Machine Translation*, pages 207–216,
Edinburgh, Scotland, UK, July 30–31, 2011. ©2011 Association for Computational Linguistics

translation (MT) and information retrieval (IR) systems rely on conditional probabilities; in contrast, we require comparable scores or probabilities over all document pairs. To avoid directly computing the similarity of all pairs, we use a randomized approximation algorithm based on locality sensitive hashing (LSH).

For this joint approach, we represent each document in both languages using an n-dimensional feature vector template which consists of the set of intersecting words which are found across all documents in both language collections. For each dimension i.e. word, in the feature vector template we calculate tf·idf score for the given document. Unlike other approaches, where documents or their word representations are first translated from foreign language to English using bilingual dictionary (Fung and Cheung, 2004), (Munteanu and Marcu, 2005) and (Uszkoreit et. al., 2010) in our approach we don't utilize any existing MT type artifact. In other words, for a given language pair we don't use translation lexicon by training an existing statistical machine translation system using sentence aligned parallel bilingual data in the same language or existing translation lexicon. Earlier work done by Enright and Kondrak (2007) uses only hapax words to represent and rank (based on the overlap number) translation documents pair in a parallel bilingual collection which is an easier task to evaluation due to the presence of a one-to-one matching among the bilingual documents. Most recently, Patry and Langlais (2011) show an improvement over this method by using an IR system to first retrieve translation document candidates and then identify translation document pairs by training a classifier.

We start off by giving detailed explanation of the above mentioned data representation. We then test the feasibility of our approach using aligned parallel document data from three different bilingual collections in several languages and writing systems. Results from these tests are given in section 3. The goal of developing our approach was to utilize it as an initial filtering step in developing parallel corpora from large, multilingual collections, such as the collection of more than 800K English and German books we describe in section 4. Since we start with no information on the possible translation pairs in our large collection and in order to verify the potential of our method, we first show results on retrieving 17 known parallel book pairs embedded in a small randomly selected subset of 1K books (section 4.1). Since performing cosine similarity across all document pairs is computationally expensive with time complexity of $O(n^2)$ we utilize the LSH based approximation algorithm for the cosine similarity measurement based on the work by Ravichandran et. al (2005). A brief overview of this approach is given in Section 5, which is followed by our implementation results explained and analyzed in section 6. To conclude the paper, we give a brief outlook on future work.

## 2 Document Representation

In Figure 1, we depict the process that we use to represent documents from bilingual collections in vector space and perform similarity measurements. We start by computing a word frequency count for each of the documents in our collection and creating a word frequency list. For each language, we take a union of the words in each document's frequency list to construct a global word list for the given language. The two global word lists are then intersected, and a list of overlapping words is created. From the initial list of overlapping words in both languages, we remove stop words by using stop word lists (words with high document frequency). The space-separated tokens extracted in this process are not necessarily words in the linguistic sense; therefore, we further refine the overlapping word list by removing tokens that contain non-alphanumeric characters. We make one exception for tokens (such as might appear in a time/date format) that contain hyphens, backslashes, apostrophes, and periods so long as these characters do not occur at the beginning or at the end of the token.

We call this list of overlapping tokens a feature vector template, where each token in the list is one feature. Using this feature vector template we go back and represent each document in the bilingual collection using the template vector by computing the tf·idf value for each token in the template vector over each particular document. Now that we have the original documents from both languages represented in a language-independent space, we compute vector similarity across all document pairs in order to come up with a single ranked list. We talk more in detail about the similarity metrics that we have considered and decided to use in the following section.

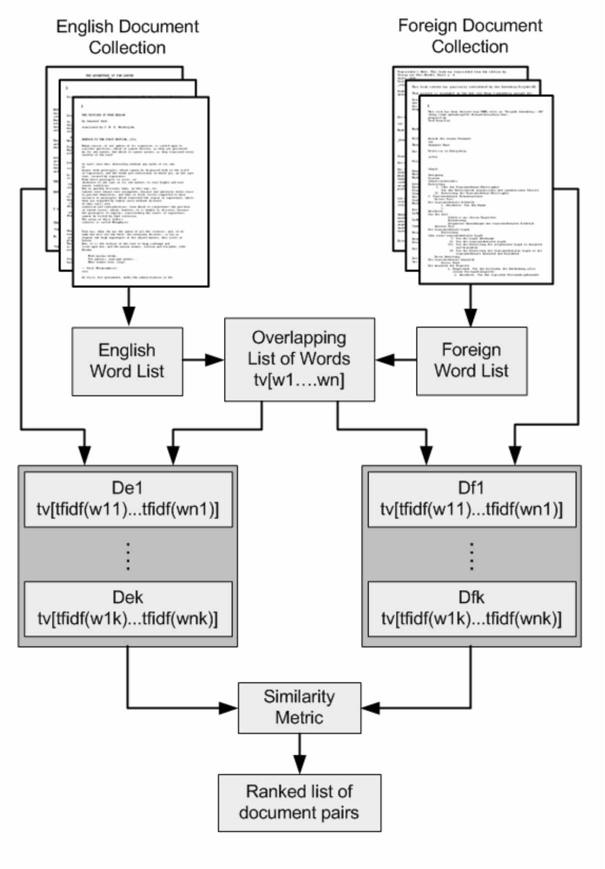
English Document Collection
Foreign Document Collection
English Word List
Overlapping List of Words tv[w1….wn]
Foreign Word List
De1 tv[tfidf(w11)...tfidf(wn1)]
Dek tv[tfidf(w1k)...tfidf(wnk)]
Df1 tv[tfidf(w11)...tfidf(wn1)]
Dfk tv[tfidf(w1k)...tfidf(wnk)]
Similarity Metric
Ranked list of document pairs

Figure 1. Process of creating and representing each document of a bilingual collection in an independent vector space.

## 3 Motivational Experiments

### 3.1 Evaluation Collections

We start off by evaluating the above proposed approach of determining candidate document translation pairs using three different parallel collections: Europarl, created by Koehn (2005), UN Arabic English Parallel Text (LDC2004E13) and the Arabic News Translation Part 1 (LDC2004T17). The purposes of first testing our approach using the Europarl corpus were twofold: This collection contains parallel documents (sessions of the European Parliament) that are further aligned at the speech and sentence level, which allows us to test alignment accuracy at several levels of granularity. Second, this collection contains parallel data from different groups of languages (Germanic, Romance, Slavic, Hellenic, etc.) and therefore is useful to observe the performance of our approach across different language families, which in turn are important to observe the difference in the cognate rates and the size of the overlapping words. In addition to the Europarl corpus we use the two English-Arabic parallel collections to test our approach across various alphabets (Arabic in addition to the Latin, Greek and Cyrillic found in the Europarl collection). Shown in Table 1 are basic statistics for all 3 corpora on the language pairs considered. We give min, max and median values over the number of words in each document.

| Collection | # doc. Pairs | Lang. | Min | Max | Median |
|---|---|---|---|---|---|
| Europarl en-de | 654 | En | 92 | 109030 | 46800.5 |
| | | De | 95 | 99753 | 43161.0 |
| Europarl en-bg | 430 | En | 4872 | 59284 | 10706.5 |
| | | Bg | 4771 | 56907 | 10167.0 |
| Europarl en-es | 642 | En | 92 | 109793 | 46790.5 |
| | | Es | 104 | 114770 | 48989.0 |
| Europarl en-gr | 412 | En | 92 | 93886 | 21290.0 |
| | | Gr | 103 | 93304 | 21122.0 |
| Newswire en-ar | 230 | En | 66 | 47784 | 691.5 |
| | | Ar | 62 | 34272 | 560.0 |
| UN en-ar | 430 | En | 17672 | 71594 | 23027.0 |
| | | Ar | 15478 | 62448 | 19682.0 |

Table 1. Document length statistics over 6 Parallel Collections.

From the Europarl collection we sentence aligned sessions in the following four language pairs where the English language is the source language: English-German, English-Spanish, English-Bulgarian and English-Greek. The foreign language in all four language pairs is selected from a different language group (Germanic, Romanic, Slavic), with Greek being a more isolated branch. For the Arabic language we used two parallel document collections in different domains – newswire and documents published by the United Nations. The Newswire parallel collection consisted of 1526 news stories which we combined based on the news story publication date and obtained 230 parallel documents. The purpose of combining the news articles is to increase the number of words present in each document since the original size of the news articles was not at a level to be treated as a document as in the case of the remaining two collections. The UN parallel collection consists of 34,575 document pairs.

### 3.2 Similarity Metrics

We considered five similarity metrics proposed at one time or another for vector space models in IR: Cosine (shown below), Dice, Product, Jaccard and Euclidean.

$$\frac{\sum x_i y_i}{\sqrt{\sum x_i^2 \sum y_i^2}} \tag{1}$$

Document similarity using the cosine metric relies on the angle between the vector representations and it is length invariant. The Dice metric relies on the number of common tokens between the two documents. Euclidean computes the similarity as a point distance between the two vector representations and is not normalized by the vector length which does not make it vector invariant. Jaccard distance is the ratio of the intersection and the union of the two vector representations while the product coefficient is simply the inner product of the two vectors. While there is no clear evidence across the literature whether one similarity metric is more useful across a range of tasks compared to another, the cosine similarity metric is mostly preferred. Shown in Figure 2 are the precision vs. recall plots of the above similarity measurements when used with our method. Tests were done on our set of 654 English-German sessions from the Europarl collections. To test the impact of the document length on the performance of the metric we performed two types of tests across all 5 metrics. In the first type we performed similarity analysis on the full document length (marked as 100%) and on the final 10% of each document (marked as 10%). We deliberately omitted the top part of the document to avoid any inadvertent inclusion of session date, topic, title, etc. (As it turned out, this was not a problem in our data.) We perform similarity measurements across all document pairs, and we generate a single ranked list. As can be seen from the plot, all five metrics yield better performance when all words in documents are considered compared to only considering 10%. The performance ranking of all five metrics was identical on both versions of the document set. Even though depicted in the above plot, the Jaccard distance performed pretty much the same as the Dice distance and therefore there is no visible difference between the two. While on the 10% version of the collection, the Euclidean distance has the worst precision, it could still be explored as a metric to obtain document translation pairs with the original collection with a modest to moderate recall range for P=1. The Jaccard distance along with the Dice distance yield the highest precision values across all recall values but they achieve the same recall range for P=1 as the Cosine metric. Since we are only interested in top-N document pairs that have P=1 and furthermore there are approximate algorithms for the Cosine similarity metrics we decided to further utilize this metric. The same metric has been previously used in determining potential translation candidates on sentence level by Munteanu and Marcu (2005) and in our case we are extending it to perform pair-wise document similarity.

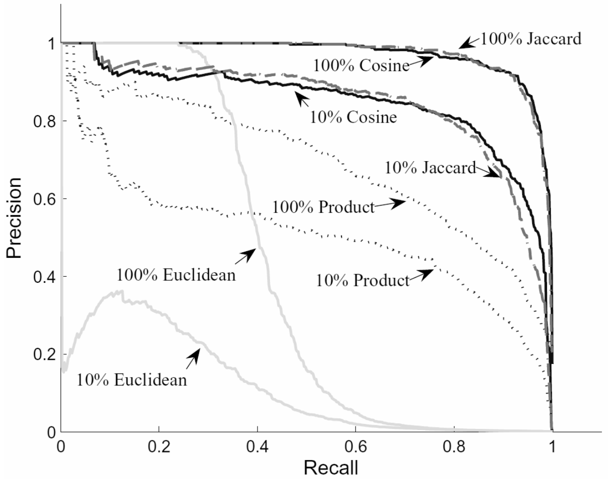

Figure 2. Precision vs. recall plot using various similarity measurements on the Europarl English-German collection.

When run on the same English-German collection, Enright's and Kondrak's (2007) approach achieves mean reciprocal rank (MRR) of 0.989 when using document specific hapax words and MRR=0.795 when using collection specific hapax words. With the above explained approach we obtain MRR=0.995.

### 3.3 Post Filtering Approaches

To further improve the precision of our approach we tested out two types of filtering the initial results. Since we threat documents as “bag of words” and since the Cosine metric uses the angle between the vector representations and is length invariant there may be instances of source documents that would yield high cosine coefficients over all target documents. In these instances, multiple document pairs with the same source document may be ranked high. To alleviate this problem, we consider two types of filtering the initial results. We go over the single ranked list and we only keep the top five document pairs for a given source document, thus introducing “diversity” in the ranked list. The second filter is motivated by the basic assumption used in the machine translation field that the length of the target sentence is in a given length range of the source sentence. We extend this assumption on a document level and we filter out all document pairs from the ranked list that are not in the ±20% range of the source document length. Both of the above values were selected based on empirical evidence without detailed explanation. Shown in Figure 3 are the effects of these two simple filtering techniques.

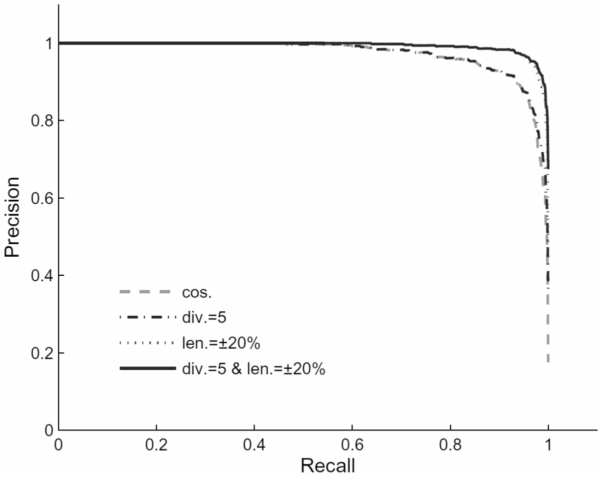

Figure 3. Diversity and length based filtering effects on the English-German Europarl collection.

Compared to the diversity filter, the length based filter yields better gain in precision while a combination of both methods achieves the highest recall range for P=1.

### 3.4 Target Languages and Writing Systems

Shown in Figure 4 are the precision/recall results on all six collections explained in Section 3.1. Post-filtering steps explained in the previous section were not utilized on these results. Our approach yields best precision on the Arabic News Translation Part 1 collection while the worst performance is on the UN Arabic English Parallel Text. While the performance on the English-German and English-Spanish collections is somewhat the same, out of all 4 Europarl collections we achieve best results on the Greek collection and worst results on the Bulgarian target language.

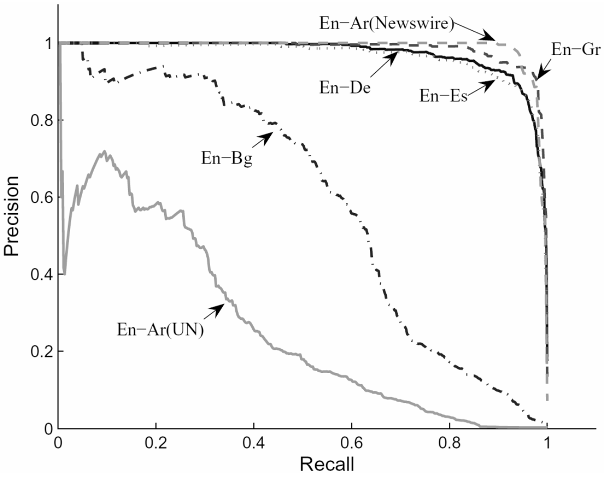

Figure 4. Precision vs. recall on 5 different language pairs using cosine similarity distance metric.

In Table 2, we give the vector template length for each collection.

| Collection | # of overlapping tokens |
|---|---|
| Europarl en-de | 37785 |
| Europarl en-es | 36476 |
| Europarl en-bg | 29360 |
| Europarl en-gr | 17220 |
| UN en-ar | 3945 |
| Newswire en-ar | 1262 |

Table 2. Number of overlapping words (vector template length) in the six parallel collections.

Unsurprisingly, due to the difference in script and language family, the feature vector templates for the English-Arabic collections have the smallest lengths.

Shown in Figure 5 are effects of the trivial diversity and length based filtering on the above precision vs. recall results. Bulgarian has improve substantially and so has the UN Arabic, but recall on the Arabic newswire is truncated on reaching P=0.4.

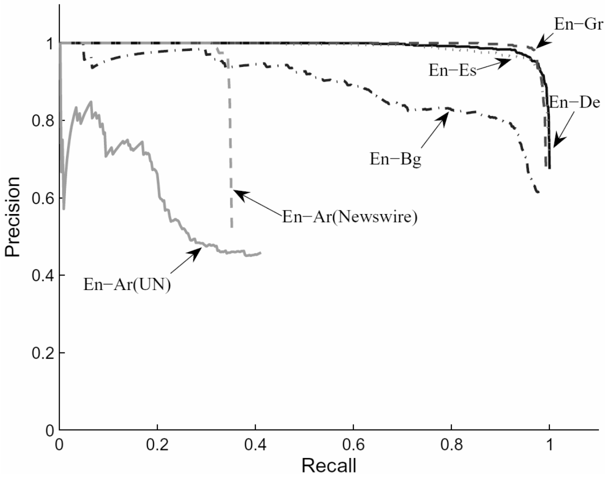

Figure 5. Precision vs. recall on 6 collections using div=5 and length filtering with ±20%.

### 3.5 Randomly Selected Documents

While useful to evaluate the feasibility of our approach, the previous parallel bilingual collections are unrealistic because there is, by the corpus' design, a translation for each document. To observe the performance on a bilingual document collection where there is no a priori information on translation pairs we created ten random subsets from the Europarl English-German collection. These subsets were created by randomly selecting 50% (328 documents) of the English and 5% (33 documents) of the German documents for each subset collection. Shown in  is interpolated average precision over the ten subsets. The Mean Average Precision (MAP) obtained was 0.986.

## 4 Multilingual Book Collection

Our multilingual book collection consists of around 800k books in German and English languages. It is a subset of a larger Internet Archive[1] collection of books in over 200 languages. The whole collection consists of OCRed books incorporating a small number of human transcribed books from Project Gutenberg[2]. The collection was initially annotated with author and language information using the existing database obtained from the Internet Archive. This database originally contained incorrect language metadata. Using the freely available language identifier TextCat (Cavnar and Trenkle, 2005) we tagged the whole book collection and extracted 705692 English and 96752 German books. This process had the additional benefit of cleaning the German book collection of books written in the Fraktur script due to the bad OCR output. (Incredibly noisy OCR was simply recognized as "not German" by the character n-gram models.) Shown in Table 3 are word length statistics over the books in the collection.

| Language | # of books | # of uniq. words | Min | Max | Median |
|---|---|---|---|---|---|
| German | 96752 | 5030095 | 33 | 2372278 | 109820 |
| English | 705692 | 20001702 | 37 | 5155032 | 75016 |

Table 3. Bilingual book collection statistics.

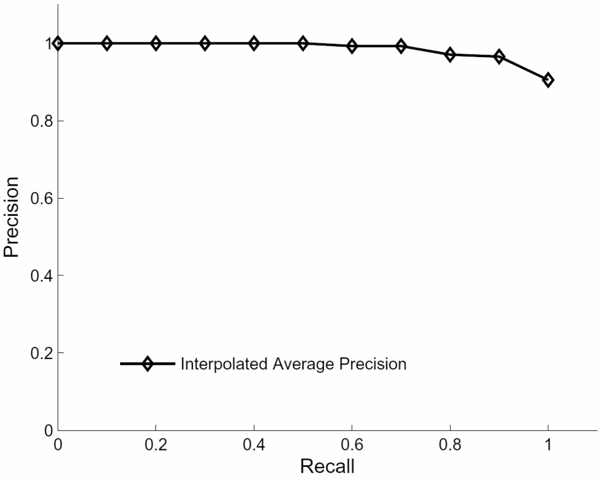

Figure 6. Average precision interpolated at 11 points over ten randomly created subsets consisting of 50% English and 10% German documents from the English-German Europarl collection.

### 4.1 Development Set

Moving onto our book collection, we start off by evaluating the method on a smaller randomly selected subset of 1000 books in both languages. Since it is not feasible to perform a full recall

[1] http://www.archive.org/details/texts/

[2] http://www.gutenberg.org

evaluation on the whole book set we include 17 known book translation pairs in the 1000 random bilingual book collection. The 17 book translation pairs were constructed by hand by running a prevision version of our full algorithm and indentifying translation pairs. Shown in Figure 7 is the precision vs. recall plot on the 17 book pairs. As in the case of the 10 randomly selected Europarl subsets, we also performed diversity and length based filtering of the initial results prior to computing precision vs. recall.

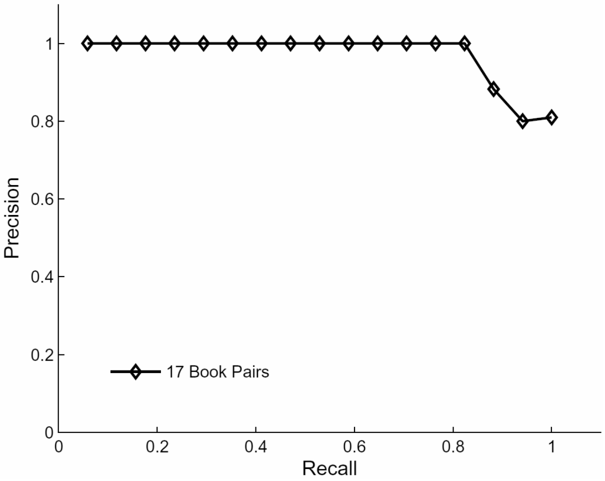

Figure 7. Precision vs. recall running our method on a 1000 randomly selected bilingual book subset with 17 book translation pairs inserted.

## 5 LSH Based Approximate Algorithm for Cosine Similarity

Due to the collection size and length of each book it is infeasible to perform cosine similarity over all possible book pairs, i.e. approximately 68.2B comparisons. This brute force approach has time complexity of $O(n^2k)$ where n is the number of books in the collection and k is the vector template length. We therefore employ a fast cosine similarity calculation approach developed by Charikar (2002) and utilized by Ravichandran et. al (2005) for creating similarity lists of nouns in large collection. In this section we give a summary of this approach and explain how it was applied for our task.

Locality Sensitive Hashing (LSH), initially introduced by Idyik and Motwani (1998), is used for finding approximate nearest neighbors in high dimensional spaces. In general, their approach hashes query vectors into bins where the probability of collision is higher due to the fact that vectors in the same bin share the same locality. Their approach reduces the approximate nearest neighbor problem on the Hamming space.

Charikar expanded this approach and showed that the probability of collision of hashed vectors for appropriately chosen hash function *h* is related to the angle between the vectors as:

$$\Pr[h(x) = h(y)] = 1 - \frac{\theta(x, y)}{\pi} \qquad (2)$$

This is closely related to the cosine function. From the above equation we thus have:

$$\cos(\theta(x, y)) = \cos\{(1 - \Pr[h(x) = h(y)])\pi\} \qquad (3)$$

Charikar uses a hash function based on random hyperplanes and creates a fingerprint for each original vector using the following approach:

Generate *d*, *k*-dimensional random vectors from a standard normal (Gaussian) distribution: $\{r_1, r_2, \ldots.. r_d\}$. For each original vector x use the following hash function to generate a fingerprint of *d* bits:

$$h_r(x) = \begin{cases} 0 & if \sum x_i r_i < 0 \\ 1 & if \sum x_i r_i \geq 0 \end{cases} \qquad (4)$$

By doing this we represent each vector in our original vector set into a bit stream that reduces our vector space representation from *k* to *d* dimensions, where *d* << *k*. Having bit stream as our data representation, the probability of hash collision, i.e. the probability of two vectors being equal $\Pr[h(x) = h(y)]$, is equivalent to the Hamming distance between the two bit streams:

$$\Pr[h(x) = h(y)] = \frac{HD}{d} \qquad (5)$$

Therefore, performing fast cosine similarity boils down to finding the Hamming distance between the two bit streams.

Now that we have an approximate method of finding the cosine similarity between two vectors, we use Ravichandran's (2005) formulation of the fast search algorithm developed by Charikar, which in turn used Indyk and Motwani's orginal PLEB (Point Location in Equal Balls) algorithm as a starting point. The steps of this algorithm are outlined in the next subsection. For more detailed explanation of this algorithm the reader is referred to Section 5 of Charikar's work (2002).

## 5.1 Nearest Neighbor Search Algorithm

We now outline the steps of the fast search algorithm. For more detailed explanation of the algorithmic implementation users are referred to Section 3 of Ravichandran's work (2005):

- For all *m* documents represented in the vector space using the template vector, compute LSH *d*-bit signature using the formula given in (4).
- Generate *q* permutations of length *d*.
- For each of the *q* permutations, generate *m* permuted LSH signatures.
- For each of the *q* permutation bins, lexicographically sort the *m* permutated bit vectors.
- For each lexicographically sorted bin, go over the *m* bit streams and compute the Hamming distance between the current bit stream and the subsequent *b* bit streams in the sorted list starting from the top.
- If the Hamming distance is above a previously set threshold, output the book pair along with the Hamming distance result.

Compared to Ravichandran's algorithm for creating noun similarity lists, in our approach we deal with two distinct groups of documents: those in each language. We start off by creating a single list of documents and we represent each document in this list using the LSH based fingerprint. We then generate *q* permutation vector bins, and we lexicographically sort each bin. In our beam search approach, since we have documents in two different languages, we only consider documents that have a different language. The results of the beam search for each bin are then combined. Since in each beam the same permutation is performed over all fingerprints, the Hamming distance across all bins for a given document pair would be the same. Therefore after combining the results we remove duplicate document pairs and sort by the Hamming distance to obtain the final ranked list. The running time of this algorithm is dominated by the $O(qn\log n)$ step of sorting the permuted bit vectors in each of the bins.

## 6 Detecting and Ranking Book Translation Pairs in a Large Book Collection

Using the previously explained method we processed the large book collection by first computing the vector template. For the large book collection, the vector template size *k*, i.e. the number of overlapping tokens obtained, was 638,005. After removing stop words and unwanted tokens (explained in Section 2) the template vector length was reduced to 563,053. Shown in Table 4 are statistics over the number of vector template tokens whose tf·idf values are greater than zero across the two languages.

| Language | Min | Max | Median |
|---|---|---|---|
| German | 7 | 7212 | 229 |
| English | 11 | 6637 | 585 |

Table 4. Statistics over the number of tokens in the vector representation of each book whose tf·idf are greater than zero.

Once processed and represented in vector space, we proceed with computing the approximate cosine similarity across the bilingual collection. We precompute the Hamming distance based on a cosine similarity threshold of 0.18 which is equivalent to different Hamming distance values depending on the length of the LSH based fingerprint. For the book collection we experimented with 4 different sets of values for the number of hyperplane based hash functions, the number of permutations and the length of the beam search. For each of these parameters in our setup we created ranked lists as explained in Section 5.1. We then went over the top 300 book pairs in each list and annotated the correct book translations. Based on the human annotation we then computed average precision over the ranked list. Shown in Table 5 are the results for LSH based fingerprint of size *d*=500. Due to the randomness introduced by the permutations, there is not a monotonic increase in accuracy, but in general more permutations and wider beams show substantial improvements.

| q\b | | AP | Time [hrs] |
|---|---|---|---|
| **q=25** | **b=25** | 0.307 | 24.9 |
| | **b=50** | 0.213 | 41.1 |
| | **b=100** | 0.280 | 67.2 |
| **q=100** | **b=25** | 0.488 | 99.6 |
| | **b=50** | 0.388 | 164.4 |
| | **b=100** | 0.461 | 269.1 |
| **q=200** | **b=25** | 0.357 | 199.2 |
| | **b=50** | 0.412 | 328.8 |
| | **b=100** | 0.455 | 538.2 |
| **q=500** | **b=25** | 0.489 | 498.1 |
| | **b=50** | 0.490 | 822.0 |
| | **b=100** | 0.493 | 1345.5 |

Table 5. Average precision on the large English-German book collection across various parameters of the LSH based search algorithm.

For the above given results for *d*=500, we calculated an estimated time that it would take to perform the fast cosine similarity if the algorithm were to be run in serial fashion. Shown in Figure 8 is a scatter plot of the time vs. the average precision obtained.

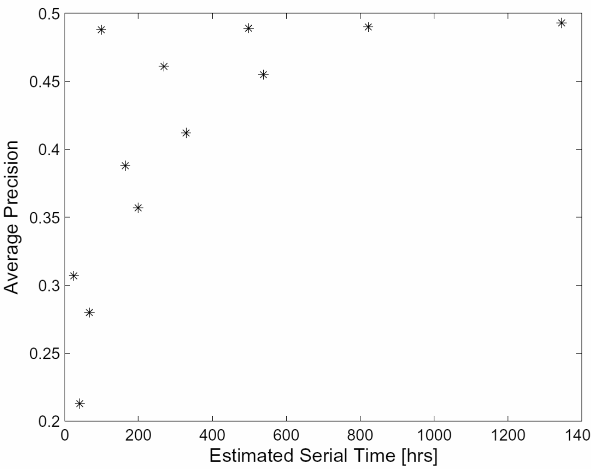

Figure 8. Estimated serial time vs. average precision with *d*=500 dimensional LSH based fingerprints.

In summary, while increasing the number of permutations and the beam search over different values increases the average precision the time cost required is significantly larger especially for increasing the number of permutations.

## 7 Future Work

In the future we plan on experimenting with larger dimensionality *d* for the LSH fingerprint, the number of random permutations q i.e. bins and the beam search parameter b. In order to further improve the average precision we would also like to experiment with different longest common subsequence (LCS) based approaches for re-ranking the cosine based ranked lists. Furthermore, we plan on exploring more accurate joint models of translation. It would also be interesting to observe the performance of our system on other language pairs, such as English-Chinese and languages with resource-poor bilingual collections.

## 8 Conclusion

This paper presents and evaluates a new approach to detecting and ranking document translation pairs. We showed that this simple method achieves high precision vs. recall on parallel bilingual collections where there is one document translation for each source document. We also showed that the method is capable of detecting document translations in random subsets where no known document translation information is available. Using an approximation algorithm for cosine similarity, we showed that this method is useful for detecting and ranking document translation pairs in a large bilingual collection with hundreds of thousands of books and billions of possible book pairs. This method is conceivable to be used for other languages and collection genres and also on other types of translation methods such as transliteration. While in some instances other simple methods of aligning the dictionaries might be needed, as in the case of the Chinese language.

## Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by NSF grant #IIS-0910884. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

## References

Alexandre Patry and Philippe Langlais, 2011. Identifying Parallel Documents from a Large Bilingual Collection of Texts: Application to Parallel Article

Extraction in Wikipedia. Proceedings of the 4th Workshop on Building and Using Comparable Corpora, pages 87-95, Portland, OR.

Bing Zhao and Stephan Vogel. 2002. Adaptive Parallel Sentences Mining from Web Bilingual News Collection. Proceedings of IEEE International Conference on Data Mining, pages 745-750. Maebashi City, Japan.

Deepak Ravichandran, Patrick Pantel, and Eduard Hovy. 2005. Randomized Algorithms and NLP: Using Locality Sensitive Hash Function for High Speed Noun Clustering. Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pages 622–629, Morristown, NJ.

Dragos Stefan Munteanu and Daniel Marcu. 2005. Improving Machine Translation Performance by Exploiting Non-Parallel Corpora. Computational Linguistics, 31(4): 477-504.

Jacob Uszkoreit, Jay Ponte, Ashok Popat and Moshe Dubiner, 2010. Large Scale Parallel Document Mining for Machine Translation. Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pp. 1101-1109. Beijing, China.

Jason R. Smith, Chris Quirk, and Kristina Toutanova, 2010. Extracting Parallel Sentences from Comparable Corpora using Document Level Alignment, Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL (HLT NAACL'10), Los Angeles, California.

Jessica Enright and Grzegorz Kondrak 2007. A Fast Method for Parallel Document Identification, Proceedings of Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'07) companion volume, pages 29-32, Rochester, NY.

Matthew Snover, Bonnie Dorr, and Richard Schwartz. 2008. Language and Translation Model Adaptation using Comparable Corpora. Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP'08), pages 856–865, Honolulu, HI.

Moses S. Charikar. 2002. Similarity estimation techniques from rounding algorithms. In Proceedings of the thiry-fourth annual ACM symposium on Theory of computing (STOC'02), pages 380–388, New York, NY.

Pascale Fung and Percy Cheung. 2004. Mining Very-Non-Parallel Corpora: Parallel Sentence and Lexicon Extraction via Bootstrapping and EM. In Proceedings of Conference on Empirical Methods in Natural Language Processing (EMNLP'04), Barcelona, Spain.

Philip Resnik and Noah Smith. 2003. The Web as a Parallel Corpus. Computational Linguistics, 29(3): 349-380.

Philipp Koehn, 2005. Europarl: A Parallel Corpus for Statistical Machine Translation. MT Summit 2005. Phuket, Thailand.

Piotr Indyk and Rajeev Motwani. 1998. Approximate nearest neighbors: towards removing the curse of dimensionality. In Proceedings of the thirtieth annual ACM symposium on Theory of computing (STOC '98), pages 604–613, New York, NY.

William B. Cavnar and John M. Trenkle. 1994. N-Gram-Based Text Categorization. Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval, pages 161-175, Las Vegas, NV.