

An Automatic Filter for Non-Parallel Texts

Chris Pike

Computer Science Department
New York University
715 Broadway, 7th Floor
New York, NY 10003 USA
{lastname}@cs.nyu.edu

I. Dan Melamed

Computer Science Department
New York University
715 Broadway, 7th Floor
New York, NY 10013 USA
{lastname}@cs.nyu.edu

Abstract

Numerous cross-lingual applications, including state-of-the-art machine translation systems, require parallel texts aligned at the sentence level. However, collections of such texts are often polluted by pairs of texts that are comparable but not parallel. Bitext maps can help to discriminate between parallel and comparable texts. Bitext mapping algorithms use a larger set of document features than competing approaches to this task, resulting in higher accuracy. In addition, good bitext mapping algorithms are not limited to documents with structural mark-up such as web pages. The task of filtering non-parallel text pairs represents a new application of bitext mapping algorithms.

1 Introduction

In June 2003, the U.S. government organized a “Surprise Language Exercise” for the NLP community. The goal was to build the best possible language technologies for a “surprise” language in just one month (Oard, 2003). One of the main technologies pursued was machine translation (MT). Statistical MT (SMT) systems were the most successful in this scenario, because their construction typically requires less time than other approaches. On the other hand, SMT systems require large quantities of parallel text as training data. A significant collection of parallel text was obtained for this purpose from multiple sources. SMT systems were built and tested; results were reported.

Much later we were surprised to discover that a significant portion of the training data was not parallel text! Some of the document pairs were on the same topic but not translations of each other. For today’s sentence-based SMT systems, this kind of data is noise. How much better would the results have been if the noisy training data were automatically filtered out? This question is becoming more important as SMT systems increase their reliance on automatically collected parallel texts.

There is abundant literature on aligning parallel texts at the sentence level. To the best of our knowledge, all published methods happily misalign non-parallel inputs, without so much as a warning. There is also some recent work on distinguishing parallel texts from pairs of unrelated texts (Resnik and Smith, 2003). In this paper, we propose a solution to the more difficult problem of distinguishing parallel texts from texts that are comparable but not parallel.

Definitions of “comparable texts” vary in the literature. Here we adopt a definition that is most suitable for filtering SMT training data: Two texts are “comparable” if they are not alignable at approximately the sentence level. This definition is also suitable for other applications of parallel texts, such as machine-assisted translation and computer-assisted foreign language learning.

Resnik and Smith (2003) suggested three approaches to filtering non-parallel texts: STRAND, tsim, and a combination of the two. STRAND relies on mark-up within a document to reveal the document’s structure. STRAND then predicts that documents with the same structure are parallel. Tsim uses a machine-readable bilingual dictionary to find word-to-word matches between two halves of a bitext. It then computes a similarity score based on the maximum cardinality bipartite matching between the two halves. We chose to compare our method with tsim because we were interested in an approach that works with both marked up and plain text documents.

2 A Modification to SIMR

Our work is based on a modification of the SIMR bitext mapping algorithm (Melamed, 1999). The SIMR algorithm attempts to construct a piecewise linear approximation to the True Bitext Map (TBM) of a bitext by greedily searching for small chains of points of correspondence. Each chain forms one section of the approximation. SIMR uses a two-phase approach to generating chains. First, it generates a set of potential points of correspondence within a search rectangle. Next, it searches the

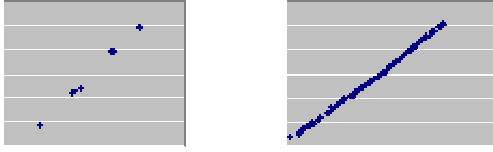


Figure 1: On the left is part of a bitext map generated by SIMR for non-parallel texts. On the right is part of a bitext map for parallel texts.

points of correspondence for chains whose points meet requirements for linearity, injectivity, and maximum angle deviation. If no such chain is found, the search rectangle is expanded and the search repeats.

Our method of detecting translations is based on the premise that SIMR will find fewer points of correspondence in comparable texts than it will in parallel texts. This is because points of correspondence are more likely to occur in closely corresponding locations in the two halves of a bitext than in two documents that are merely comparable. Therefore, the bitext map of parallel texts will usually be much denser than the bitext map of comparable texts. Figure 1 above contrasts the bitext maps output by SIMR for non-parallel and parallel texts.

To maximize the percentage of correctly classified document pairs, we need to maximize the difference between the map densities of parallel and comparable texts. SIMR’s built in restrictions on the chains it will accept severely limit the number of points of correspondence SIMR accepts from most non-parallel texts. Despite this SIMR still generated bitext maps for some non-parallel documents that had densities very close to the densities of parallel documents. Chains of spurious points tended to form over a longer section of the bitext than correct chains. Therefore we introduced an additional parameter that limited the length of chains that SIMR would accept. This modification of SIMR is called SIMR-cl.

Chains are not perfectly linear. Therefore we cannot calculate chain length by simply taking the distance between the first and last points in the chain. Instead we find the smallest possible rectangle for which all points in the chain are interior points. We then calculate the length of the chain as the distance from the lower left corner to the upper right hand corner of the rectangle.

When SIMR finds an acceptable chain the search rectangle is moved so that the point on the lower left is no longer included in the search. As a result,

when SIMR is finding a large number of chains, the length of those chains will remain relatively short. Therefore, in parallel texts SIMR will find many chains and limiting the chain length will have a minimal effect on the number of chains SIMR will find. On a non-parallel text, however, SIMR will find fewer sets of points of correspondence meeting the criteria for a chain. The result is longer chains, which can be filtered by our new parameter. E.g., the non-parallel bitext map in Figure 1, which was created without the chain length parameter, has on average 630 characters between points. In contrast, running SIMR on the same pair of non-parallel documents with a maximum chain length of 700 yielded only 22 points of correspondence, or 3032 characters between points on average.

3 Training

Training SIMR-cl, much like SIMR, requires a state space search algorithm, and an objective function to evaluate the current state. We chose to use simulated annealing to perform our state space search. The first step in training is to generate a set of parameter values that make up the current state. SIMR-cl uses the standard SIMR parameters plus the additional chain length parameter discussed above. Once the current state is set SIMR-cl generates a bitext map and calculates the density of the map. The bitext map density is defined as the number of points in the bitext map divided by the length of the main diagonal of the bitext space. We call this the SIMR-cl score.

Our objective function seeks to drive the parameters to a state where we can select a single threshold value that will classify all candidate bitexts in the development set correctly. That is, all parallel texts should have a SIMR-cl score greater than the threshold, and all non-parallel texts should have a SIMR-cl score less than the threshold. We cannot achieve this by simply measuring the percentage of correctly classified candidate text pairs, because any given change to the parameters is not likely to change the classification of any candidate bitexts.

In order to measure the amount of error we borrowed the concept of margin slack from the support vector machines literature. For simplicity we used a margin of zero, which reduces the margin slack of a SIMR-cl score to the difference between the threshold density, and the density of a misclassified candidate pair. Any correctly classified candidate pair is defined to have a margin slack of zero. From there we defined our objective as minimizing the sum of the margin slack of all candidate pairs. All that is left at this point is to select an optimal threshold. We

performed a line search for the best possible threshold for each parameter set.

4 Experiments

In our first two experiments we limited the points of correspondence to orthographic cognates. We used the Longest Common Subsequence Ratio (LCSR) to measure similarity (Melamed, 1995). The LCSR ratio is the length of the longest common subsequence of two tokens, divided by the length of the longer token. In our English-Hindi experiments we used an English-Hindi dictionary because the languages are written in different character sets, limiting the effectiveness of orthographic cognates.

4.1 STRAND data

Before evaluating our approach on the more difficult task of discriminating parallel texts from comparable texts, we compared it to previous approaches on the easier task of discriminating parallel texts from unrelated texts. For this purpose, we used the STRAND corpus, which consists of 326 candidate bitexts in French and English¹ (Resnik and Smith, 2003). As a precursor to generating a bitext map of a candidate pair we tokenized the STRAND documents and generated the axis files required by SIMR-cl. We attempted several schemes on training data and found that generating one token per HTML tag gave us the best results.

While the end performance of the two approaches was comparable, we did find that *tsim* had an advantage over SIMR-cl in training. Resnik and Smith (2003) trained *tsim* using 32 of the 326 available STRAND candidate pairs to achieve their published result. We repeated their experiments using 1/4 of the available candidate pairs for training and found no improvement, indicating that *tsim* can be optimally trained using a small development set. By contrast, using 32 training instances, SIMR-cl achieved only 86% agreement with the human judges, compared to *tsim*'s 96%. When trained with 1/4 of the candidate pairs, SIMR-cl achieved 96% accuracy.

4.2 Filtering of Comparable Texts

We were unable to find a suitable corpus containing both parallel and comparable texts. Expert opinion suggests that no such corpora are publicly available². Therefore we proceeded by simulation. We constructed 3 sets of two corpora from the Romanian/English Multext-East 1984 corpus (Tufis,

¹We removed all document pairs which were not in French/English.

²Doug Oard, personal communication, 2004.

text length	164	820	1640
<i>tsim</i>	66%	66%	66%
SIMR-cl	90%	96.5%	98.5%

Table 1: Percentage of documents correctly classified by *tsim* and SIMR-cl on parallel and comparable corpora with texts of varying lengths, by average number of words in the English text.

1999). We constructed parallel texts by breaking the corpus into aligned chunks of 10, 50, and 100 segments. We then simulated comparable texts by pairing non-aligned, consecutive chunks of the same length. We chose to use consecutive chunks because there is a better chance for overlap between words in adjacent segments than in segments far apart. After breaking the corpus into chunks, 1/3 of the chunks were used as a training set and the remaining 2/3 were used as a test set. We had 63 training and 130 test pairs of size 100, 126 training and 259 test pairs of size 50, and 642 training and 1285 test pairs of size 10. On average each English segment was 16 words in length.

Since a Romanian/English bilingual dictionary was not readily available, we created a dictionary for *tsim* by searching all aligned segments for cognates. We then performed the same optimization process for *tsim* and SIMR-cl using documents containing 10, 50, and 100 segments. After performing our optimizations, we found that the LCSR parameters optimized for *tsim* generated a dictionary containing 3380 pairs.

Using this parameter set, *tsim* correctly classified 66% of the documents in the 1984 corpus. The accuracy was the same for all bitext lengths. Much like *tsim*, we found that for SIMR-cl the optimal parameter set was independent of the length of the bitexts being compared. SIMR-cl did however perform better on longer texts. Regardless, SIMR-cl outperformed *tsim* on all text lengths, as shown in table 1.

4.3 The Surprise Language Data

Encouraged by our success on French/English and on Romanian/English, we applied our method to the Hindi/English data used during the surprise language exercise. We did not have Hindi/English bitexts that were reliably classified as parallel or not, so we could not optimize SIMR-cl's parameters specifically for this language pair. However, we were interested in determining how sensitive the parameters were to changes in the input language pair and text genre. So we simply reused the param-

eters that were found to be optimal on the Romanian/English 1984 corpus.

With these parameters, we ran SIMR-cl on just over half of the Hindi/English collection, the part that was collected from Indian government web pages. Our method classified 6 of the document pairs as non-parallel. Some of these 6 document pairs were relatively long, together they accounted for 7% of the English word count in this part of the collection.

We asked a Hindi speaker to compare the Hindi and English text in each of these 6 document pairs. For each text pair, we asked our informant:

1. Do the texts express the same ideas?
2. If yes, was one of the texts probably written as a translation of the other?
3. If yes, was the translation done roughly at the sentence level?

The informant decided that in all 6 cases, the pair of texts expressed the same ideas. However in 4 of the pairs, the two texts were probably written independently, rather than one as a translation of the other. In the remaining two texts, the informant found large omissions on the English side, larger than what typical alignment algorithms can handle.

In these latter two documents, our Hindi informant also discovered an interesting phenomenon that we were not expecting — the sections that were translated were summarized to some degree. I.e., even in sections where the order of ideas was largely the same in the two languages, the English wording was much more terse (the informant said "compressed"), and omitted many details.

In summary, our method achieved 100% precision in filtering out document pairs that were comparable but not parallel. We then asked our informant to examine 3 document pairs that our method accepted as parallel. After a cursory inspection, the informant answered yes to all 3 questions above for each of these pairs. Unfortunately, it would have been very time-consuming to evaluate recall rigorously, because it would entail exhaustive reading of pairs of documents in parallel, to ensure that there were no non-parallel segments.

5 Conclusions

We have shown that SIMR-cl, a modified version of the SIMR bitext mapping algorithm, can reliably discriminate between parallel and comparable texts. We have demonstrated that SIMR-cl is effective on three language pairs, including two where no bilingual dictionary was available. In addition, we

have presented tentative evidence that the parameters of SIMR-cl are not very sensitive to particular language pairs or text genres on this task.

Our results suggest several new avenues for future research. First, it would be useful to combine our method for filtering out non-parallel texts with methods for detecting omissions in translations (Melamed, 1996). Some of the translations found on the web today might be made more literal by deleting the untranslated parts. Second, we seem to have discovered the existence of training data for a machine learning approach to translation with summarization. Third, our results suggest that the density of a bitext map is highly correlated with its accuracy, and that this correlation is largely invariant across language pairs and text genres. If this is true, then it should be possible to train bitext mapping algorithms without any hand-aligned training data, by using map density as the objective function instead of RMS error.

Acknowledgements

Thanks to Philip Resnik and Noah Smith for sharing STRAND data, human judgements, and tsim scores. Thanks also to Noah Smith for providing a tsim implementation. This research was sponsored by the DARPA TIDES program, by an NSF CAREER award, and by an equipment gift from Sun Microsystems.

References

- I. Dan Melamed. 1995. Automatic evaluation and uniform filter cascades for inducing n-best translation lexicons. In *Proceedings of the 3rd ACL Workshop on Very Large Corpora (WVLC)*, Cambridge, Massachusetts.
- I. Dan Melamed. 1996. Automatic detection of omissions in translations. In *Proceedings of the International Conference on Computational Linguistics (COLING) 1996*, pages 764–769, Copenhagen, Denmark, August.
- I. Dan Melamed. 1999. Bitext maps and alignment via pattern recognition. *Computational Linguistics*, 25(1):107–139, March.
- D. Oard. 2003. The surprise language exercises. In *ACM Transactions on Asian Language Information Processing (TALIP)*, pages 79–84, New York, NY, June.
- P. Resnik and N. A. Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, pages 349–380, September.
- D. Tufis. 1999. Multext-east 1984 corpus. <http://nl.ijs.si/ME/>.