Parallel Texts Alignment

Luís Gomes, José Aires, and Gabriel Pereira Lopes

CITI, Departamento de Informática,
Faculdade de Ciências e Tecnologia,
Universidade Nova de Lisboa,
2829-516 Caparica, Portugal
luismsgomes@gmail.com, {aires,gpl}@di.fct.unl.pt

Abstract. Alignment of parallel texts (texts that are a translation of each other) is a step required by many applications that use parallel texts, including statistical machine translation, automatic extraction of translation equivalents, automatic creation of concordances, etc.
Most of existing methods for parallel texts alignment try to infer simultaneously a bilingual word lexicon and a set of correspondences between the occurrences of those words in the texts. Some authors suggest that an external lexicon can be used to complement the inferred one, but they tend to consider it secondary/optional. We defend that lexicon inference should not be embedded in the alignment process, and present LEXIC-AL, a new alignment method that relies exclusively on externally managed lexicons. In our experiments with the European Constitution corpus, LEXIC-AL achieves 78.7% precision and 79.2% recall.

Key words: parallel texts alignment, parallel corpora, extraction of translation equivalents

1 Introduction

Simard et al [17], Davis et al [5], Melamed [11, 12], Ribeiro et al [16, 8] and Ribeiro [15] use cognates as lexical cues for alignment. However, the number of cognates and loan words is highly dependent on the languages of the texts being aligned, as noted by Melamed [12] and confirmed by the results of the evaluation carried by Bilbao et al [2] on the impact of cognates on alignment. Melamed [12] suggests using an external bilingual lexicon in addition to cognate matching to increase the number of correspondences. However, further ahead we present several arguments supporting that we should go one step further and use *only* an external lexicon; the identification of cognate words shouldn't be done within the alignment process.

Kay and Röscheisen [9], Chen [3], Fung and Church [6] and Fung and McKeown [7] infer (by different methods) a bilingual lexicon as part of the alignment process and use that lexicon to establish correspondences between words in the two parallel texts. Like for the identification of cognates mentioned above, we defend that alignment should not be entangled with lexicon inference and instead

it should rely solely on an externally managed bilingual lexicon. This separation of alignment and lexicon extraction is supported by the following arguments:

1. Using exclusively an externally managed lexicon gives us *full control over the lexicon that is used to align*. By contrast, it is not possible to avoid eventual bogus entries in an inferred lexicon because the whole method is automatic.
2. Because the alignment depends on the size of the lexicon, an external lexicon has greater potential to obtain better alignments because we can enrich the lexicon over time. By contrast, the alignment methods that infer the lexicon automatically do not improve over time, because every time they align a pair of texts, they infer a new lexicon from scratch.
3. We can combine several sophisticated extraction methods to enrich the alignment lexicon. For example, we can use methods that specialize on identification of possible cognates, extraction of multi-word translations, extraction of single word translations, extraction of named entities, etc.

We have developed an extractor of phrase translations from aligned parallel corpora (Aires et al [1]) that has been used to periodically augment our English-Portuguese lexicon. The extracted equivalents are evaluated (marked as accepted or rejected) by a linguist before they are used for alignment.

2 Method outline

Our alignment method is divided in two stages. The first stage is performed independently for each text and is executed in parallel, taking advantage of multiple cores or processors in the computer. In this stage we obtain the list of occurrence offsets for each term of the lexicon occurring in the texts. This is implemented by a procedure named LOOKUP that takes as input a sorted list of terms and a text, and produces a list of occurrence offsets for each term found in the given text. This procedure is further discussed in section 3.

The second stage iterates over two steps: identify corresponding occurrences and select a list of non-crossing occurrences to be used as alignment anchors (see figure 1). Each iteration produces a more precise alignment (with more anchors) and the loop terminates when two consecutive iterations produce the same anchors.

English	Portuguese
the united kingdom	**o reino unido**
shall not be obliged	não ficará obrigado
or	**ou**
committed	comprometido
to adopt	**a adoptar**
the euro	**o euro**
without	**sem**
a	**uma**
separate	
decision	**decisão**
to do so by its	distinta em esse sentido de o seu
government	**governo**
and	**e**
	de o seu
parliament	**parlamento**

Fig. 1: Alignment of a passage of the European Constitution. The segments in bold are alignment anchors, i.e., non-crossing correspondences between occurrences of equivalent terms that are in the lexicon. The other segments contain whatever text exists between each two consecutive anchors.

The main alignment procedure is:

procedure ALIGN(TextX, TextY, ListOfTermsX, ListOfTermsY)
    (first stage)
  OccursX ← LOOKUP(ListOfTermsX, TextX)
  OccursY ← LOOKUP(ListOfTermsY, TextY)
    (second stage)
  Anchors ← {Origin, Terminus}
  repeat
    Correspondences ← CORRESPOND(OccursX, OccursY, Anchors)
    Anchors ← SELECT(Correspondences)
  until two consecutive iterations produce the same Anchors
  OUTPUT(Anchors)
end procedure

The CORRESPOND procedure tries to find out which occurrences of a given term (in one text) correspond to which occurrences of an equivalent term (in the other text). It takes three lists as input: two lists of term occurrences (one of each text) and the list of alignment anchors to be used as guide (which consists of a list of non-crossing correspondences). The criteria to decide which occurrences correspond to each other is presented in section 4.

The SELECT procedure selects a list of non-crossing correspondences to be used as alignment anchors according to the criteria given in section 5.

The CORRESPOND and SELECT procedures are executed one after the other in an iterative refinement loop. The output of CORRESPOND is passed to SELECT,

and the output of SELECT is passed to CORRESPOND in the next iteration, and so on. To bootstrap this cycle we give two alignment anchors, the origin and the terminus of the texts, to the first invocation of CORRESPOND, which is equivalent to say that we use the *golden diagonal* as an initial alignment approximation. As it turns out, the criteria used to find correspondences is very robust and the crudeness of the alignment given as guide impacts mostly the number of correspondences produced (not their precision), meaning that more iterations are needed for texts that are less parallel. In our experiments, further discussed in section 6, LEXIC-AL needs on average 3 iterations.

3 Lookup lexicon terms in the texts

The LOOKUP procedure is performed independently for each text and may be executed in parallel on multi-core/multiprocessor machines. It takes as input a sorted list of terms and a text, and produces a list of occurrence offsets for each term that occurs in the text. Figure 2 is an excerpt of the output of LOOKUP.

Id	Term	Occurrence offsets
...	...	...
13292	comercialização	26442, 118619
59480	cominação	36238
129400	cominação de multas	36238
129401	cominação de multas e sanções pecuniárias compulsórias	36238
20649	comissão	6938, 8237, 8462, ...
...	...	...

Fig. 2: Excerpt of the output of the LOOKUP procedure executed on the same Portuguese text that was used to create the suffix array in figure 3, where term "comercialização" occurs at offsets 26442 and 118619.

First we construct the suffix array (Manber and Myers [10]) for the text (see figure 3) and then we perform a simultaneous scan through the list of terms and the suffix array, checking *which terms are prefix of which suffixes*.

The pseudocode for LOOKUP is presented below. It produces the output presented in figure 2.

Offset	Suffix
...	...
223627	comercial comum é conduzida de acordo com os princípios e ...
26442	comercialização . ¶ o presente artigo é aplicável a ...
118619	comercialização de os diversos produtos ; medidas de ...
74580	cometidas a o sistema europeu de bancos centrais , o banco ...
75263	cometidas a o sistema europeu de bancos centrais a o abrigo ...
69327	cometidas a o sistema europeu de bancos centrais são : ¶ ...
247838	cometido falta grave . ¶ 3 . ¶ o provedor de justiça europeu ...
255647	cometido falta grave pode ser demitido por o tribunal de justiça ...
...	...

Fig. 3: Part of a suffix array of a Portuguese text. The suffixes in the rightmost column are obtained by printing the text from the offset presented in the first column of the table; besides the text being aligned we only have the column on the left loaded into main memory.

```
procedure LOOKUP(ListOfTerms, Text)
    SuffixArray ← MAKE_SUFFIX_ARRAY(Text)
    Term ← first term in ListOfTerms
    Suffix ← first suffix in SuffixArray
    repeat
        if Term is prefix of Suffix then
            OUTPUT(id of Term)
            OUTPUT(Term)
            OUTPUT(offset of Suffix in the text)
            for all suffixes S next to Suffix having Term as prefix do
                OUTPUT(offset of S in the text)
            end for
            OUTPUT(newline)
            Term ← the next term from ListOfTerms
        else if Term is lexicographically lower than Suffix then
            Term ← the next term from ListOfTerms
        else
            Suffix ← the next suffix in SuffixArray
        end if
    until we have run through all terms or through all suffixes
end procedure
```

This algorithm runs in linear time and it does not impose any limitation to the length of the terms. The list of terms is read sequentially from disk, one term at a time, making this algorithm very economic in terms of memory usage. It is also very fast because data is read sequentially.

4 Finding corresponding occurrences of equivalent terms

This section describes a criteria to decide which occurrences $o_{x:i}$ of a term in text X correspond to which occurrences $o_{y:k}$ of an equivalent term in the other text Y. In Figure 4 we represent the parallel texts as two parallel line segments. The small black rectangles represent occurrences of the word "Commission" in the X (English) text and it's translation, "Comissão", in the Y (Portuguese) text. In the English text the word "Commission" was replaced by the pronoun "it" in one place but not so in the Portuguese text. Thus the number of occurrences in both texts is different. Looking at this representation, which occurrences in X and Y can we assume to correspond?

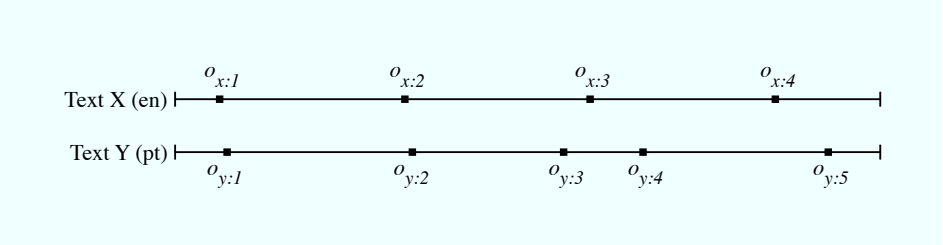


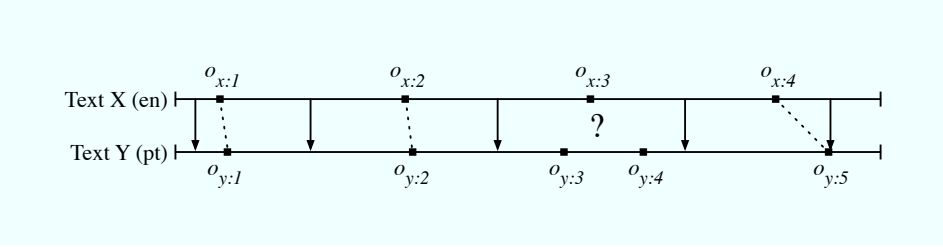
Fig. 4: Correspondences between occurrences of the word "Commission" in the English text (X) and occurrences of the word "Comissão" in the Portuguese text (Y).

We assume a correspondence between occurrences that are roughly at the same position in both texts. But, even if the occurrences are slightly apart, we can still match them if that pair is *isolated* from the other occurrences. In figure 4 the occurrences $o_{x:1}$ and $o_{y:1}$ are close to each other and distant from $o_{x:2}$ and $o_{y:2}$, thus we have no problem assuming that they correspond. The same holds for $o_{x:2}$ and $o_{y:2}$. The occurrences $o_{x:4}$ and $o_{y:5}$ are not so close to each other, but we can, arguably, match them as well because they are distant enough from other occurrences. However, we could not decide which occurrence of Y corresponds to $o_{x:3}$, even though $o_{x:3}$ is closer to $o_{y:3}$ than $o_{x:4}$ is to $o_{y:5}$.

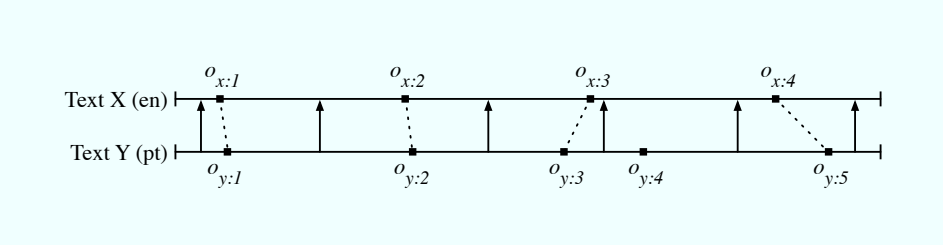
Our criteria to find pairs of occurrences that correspond is based on the distance between the pair and their isolation from other occurrences.

We compute the midpoints between consecutive occurrences in one text and we map them to positions in the other text as described further ahead. The midpoints and their mapped positions are used to divide both texts into a number of corresponding segments as shown in figure 5.

We assume that two occurrences correspond if they are within segments that correspond and if there are no other occurrences in those segments. According to this criteria we find that $o_{x:1}$ corresponds to $o_{y:1}$, $o_{x:2}$ corresponds to $o_{y:2}$ and $o_{x:4}$ corresponds to $o_{y:5}$. The occurrences $o_{x:3}$ and $o_{y:3}$ meet our correspondence criteria for the segments in figure 5b but not for the segments in figure 5a. In our



(a) Segmentation of both texts using midpoints between occurrences in X and their mapped positions in Y.



(b) Segmentation of both texts using midpoints between occurrences in Y and their mapped positions in X.

Fig. 5: Two occurrences are assumed to correspond if they are within segments that correspond and if there are no other occurrences in those segments. Correspondences are represented by dotted lines.

experiments we have considered only correspondences that are confirmed both ways. Therefore, we leave $o_{x:3}$, $o_{y:3}$ and $o_{y:4}$ without correspondence.

Figure 6 presents a polygonal chain that is obtained by connecting the origin of the texts to the lower bounds of the first alignment anchor, the upper bounds of that anchor, the lower bounds of the second anchor, and so on. The polygonal chain ends at the terminus the texts. The sequence of points of the polygonal chain are:

$$((0,0),(l_{x:1},l_{y:1}),(u_{x:1},u_{y:1}),(l_{x:2},l_{y:2}),\cdots,(l_{x:n},l_{y:n}),(u_{x:n},u_{y:n}),(L_x,L_y))$$

To map a position in text X to a position in Y we compute the ordinate of the point in the line having the given abscissa. Conversely, to map a position in Y to a position in X we compute the abscissa of the point in the line having the given ordinate.

5 Selecting a list of non-crossing correspondences

The Longest Sorted Sequence Algorithm (LSSA) described by Ildefonso and Lopes [8] selects a monotonic sequence of points from a set of corresponding

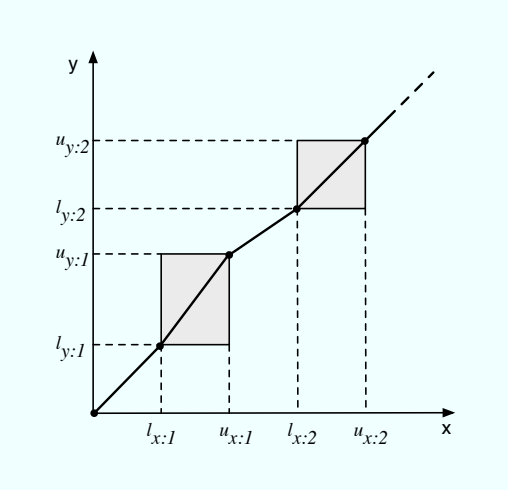


Fig. 6: A monotone polygonal chain obtained from alignment anchors (represented as shaded rectangles).

points under the assumption that the most reliable sequence is the one that includes more correspondence points.

We have observed that correspondences are not all equally reliable, and thus, they should not be all equally weighted when deciding the best alignment. The correspondences between small segments, like correspondences between occurrences of punctuation characters, are usually not as reliable as the occurrences between occurrences of large terms, as for example a correspondence between "rubber products manufacturing" and the Portuguese translation "a indústria transformadora de produtos de borracha". We hypothesize that correspondences between larger segments are more reliable. According to this hypothesis, the list of most reliable anchors is the one that maximizes the sum of the lengths of all anchor segments (for brevity we omit the procedure to obtain this list).

6 Evaluation of alignment results

We conducted our experiments upon the European Constitution corpus, consisting of 47 English texts and the respective Portuguese translations, extracted from pages available at `http://europa.eu/scadplus/constitution/`. The average size of the texts is 21KB (4012 tokens) and the largest text is 331KB (59071 tokens).

We used a lexicon with 62025 English-Portuguese pairs of terms that had been automatically extracted from another parallel corpus (European Legisla-

tion) using the method described in [1] and manually verified.

LEXIC-AL took on average 0.3 seconds to align each pair of texts on an Intel Core 2 Duo 2.66GHz. The average number of iterations was 3 and the maximum was 6. The largest pair of texts, 323KB (English) and 331KB (Portuguese), is responsible for both the maximum number of iterations and the longest time, 7.7 seconds.

There are two mainstream methodologies for evaluating the quality of an alignment: (a) comparing the alignment against a golden standard created by hand *apriori* (Melamed [13], Véronis and Langlais [18], Och and Ney [14] and Chiao et al [4]), and (b) manual verification of the correctness of an alignment (Bilbao et al [2]) *aposteriori*. Method (a) is suitable for comparing alignments of the same corpus produced by different programs. However, the creation of the golden standard is very expensive in terms of human resources. Furthermore, because the segmentation produced by our method has variable granularity that depends on the quantity and the nature of the entries present in the lexicon, the comparison of the alignment produced against the golden standard would be very difficult because the segmentations would not match. We opted by a *aposteriori* evaluation of the precision and recall of each segment using the formulas below (proposed by Veronis and Langlais):

$$Precision = \frac{\text{Number of correct words in Portuguese segment}}{\text{Total number of words in Portuguese segment}}$$

$$Recall = \frac{\text{Number of correct words in Portuguese segment}}{\text{Total number of words in Portuguese translation of English segment}}$$

Then we computed the average of precision and recall on all segments evaluated. Below we present five example segments and the respective precision and recall.

EN Seg.	PT Seg.	PT Translation	Precision	Recall
particular situations	situações especiais ,	situações especiais	2/3	2/2
at the time	em o momento em que	em o momento	3/5	3/3
, any	,	, alguns	1/1	1/2
separate		distinta	0	0
decision	decisão	decisão	1	1

We selected 2424 aligned segments from the largest pair of texts, 316KB (English) and 323KB (Portuguese), which we consider to be the most challenging pair of texts in this corpus due to their size and because it contains some passages that were not translated to Portuguese.

According to the measures above, LEXIC-AL obtains a precision of 78.69% and a recall of 79.17%.

A second experiment was done after adding to the lexicon mentioned above 4479 new entries, extracted from the previously aligned European Constitution corpus (Aires et al [1]). This enabled to raise alignment precision and recall to 84.45% and 84.55% on the same pair of texts.

Bilbao et al [2] report a maximum[1] alignment precision of 75.46% for the alignment method described by Ildefonso and Lopes [8]. Although the evaluation method is similar, the results of the two evaluations cannot be meaningfully compared because there are too many variables that may have an impact on the result — evaluations were conducted on different corpora, the alignment granularity was different, etc. Nevertheless, the precision of the alignments in these experiments allow us to say that LEXIC-AL can produce better alignments than those obtained by the method of Ildefonso and Lopes [8].

7 Conclusions and Future Work

The results obtained in the evaluation are encouraging and we feel motivated to continue improving the method. One problem that is not yet addressed is how to handle crossed correspondences.

The method is language independent and we plan to do experiments with other languages to compare the performance on several language pairs.

From the two experiments described in previous section we conclude that alignment quality can be improved by iterating on term translation extraction, validation of extracted term translations and realignment using the extended lexicon.

A comparative evaluation of the alignments produced by LEXIC-AL and GIZA++ (Och and Ney [14]) is being prepared.

References

1. José Aires, José Gabriel Pereira Lopes, and Luís Gomes. Phrase translation extraction from aligned parallel corpora using suffix arrays and related structures. In *Progress in Artificial Intelligence: 14th Portuguese Conference on AI, EPIA'09*, Lecture Notes in Artificial Intelligence. Springer-Verlag, October 2009.
2. Víctor Bilbao, Gabriel Pereira Lopes, and Tiago Ildefonso. Measuring the impact of cognates in parallel text alignment. In Amilcar Cardoso Carlos Bento and Gael Dias, editors, *2005: Portuguese Conference on Artificial Intelligence, Proceedings*, pages 338–343. IEEE Computer Society, 12 2005.
3. F. Chen, Stanley. Aligning sentences in bilingual corpora using lexical information. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 9–16, Morristown, NJ, USA, 1993. Association for Computational Linguistics.

[1] the alignment method takes a treshold parameter that must be adjusted for each pair of languages; depending on the threshold parameter the alignment precision varies between 53.37% and 75.46%

4. Yun-Chuang Chiao, Olivier Kraif, Dominique Laurent, Thi Minh Huyen Nguyen, Nasredine Semmar, François Stuck, Jean Véronis, and Wajdi Zaghouani. Evaluation of multilingual text alignment systems: the ARCADE II project. In *5th international Conference on Language Resources and Evaluation - LREC'06*, Genoa/Italy, 05 2006.
5. Mark W. Davis, Ted E. Dunning, and William C. Ogden. Text alignment in the real world: improving alignments of noisy translations using common lexical features, string matching strategies and n-gram comparisons. In *Proceedings of the seventh conference on European chapter of the Association for Computational Linguistics*, pages 67–74, Dublin, Ireland, 1995. Morgan Kaufmann Publishers Inc.
6. Pascale Fung and Ward Church, Kenneth. K-vec: a new approach for aligning parallel texts. In *Proceedings of the 15th conference on Computational linguistics*, pages 1096–1102, Morristown, NJ, USA, 1994. Association for Computational Linguistics.
7. Pascale Fung and Kathleen Mckeown. Aligning noisy parallel corpora across language groups: Word pair feature matching by dynamic time warping. In *In Proceedings of the First Conference of the Association for Machine Translation in the Americas, 81–88*, pages 81–88, 1994.
8. Tiago Ildefonso and Gabriel Lopes. Longest sorted sequence algorithm for parallel text alignment. *Computer Aided Systems Theory –EUROCAST 2005*, pages 81–90, 2005.
9. Martin Kay and Martin Röscheisen. Text-translation alignment. *Computational Linguistics*, 19(1):121–142, 1993.
10. Udi Manber and Gene Myers. Suffix arrays: a new method for on-line string searches. *SIAM Journal on Computing*, 22(5):935–948, 1993.
11. Dan Melamed, I. Bitext maps and alignment via pattern recognition. *Comput. Linguist.*, 25(1):107–130, 1999.
12. I. Dan Melamed. A portable algorithm for mapping bitext correspondence. In *ACL-35: Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 305–312, Morristown, NJ, USA, 1997. Association for Computational Linguistics.
13. I. Dan Melamed. Manual annotation of translational equivalence: The blinker project. Technical Report IRCS-98-07, Institute for Research in Cognitive Science, 1998.
14. Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.
15. António Ribeiro. *Parallel Texts Alignment for Extraction of Translation Equivalents.* PhD thesis, Universidade Nova de Lisboa, Lisboa, 2002.
16. António Ribeiro, Gael Dias, G. P. Lopes, and João Tiago Mexia. Cognates alignment. In Bente Maegaard, editor, *Proceedings of the Machine Translation Summit VIII (MT Summit VIII), Santiago de Compostela, Spain, September 18-22, 2001*, pages 287–292. European Association of Machine Translation, 09 2001.
17. M Simard, G Foster, and P Isabelle. Using cognates to align sentences in parallel corpora. In *In Proceedings of the Fourth International Conference on Theoretical and Methodological Issues in Machine Translation*, pages 67–81, 1992.
18. Jean Véronis and Philippe Langlais. Evaluation of parallel text alignment systems. In Jean Véronis, editor, *Parallel Text Processing: Alignment and Use of Translation Corpora*, volume 13 of *Text, Speech and Language Technology*, chapter 19, pages 369–388. Kluwer, Dordrecht, 2001.