Multilingual alignments by monolingual string differences

Adrien Lardilleux and **Yves Lepage**
GREYC, University of Caen Basse-Normandie,
BP 5186, Caen Cedex, France
Firstname.Lastname@info.unicaen.fr

Abstract

We propose a method to obtain subsentential alignments from several languages simultaneously. The method handles several languages at once, and avoids the complexity explosion due to the usual pair-by-pair processing. It can be used for different units (characters, morphemes, words, chunks). An evaluation of word alignments with a trilingual machine translation corpus has been conducted. A comparison of the results with those obtained by state of the art alignment software is reported.

1 Introduction

Several tools are available nowadays for alignment of pairs of languages. Among them, the bilingual word aligner GIZA++ (Och and Ney, 2003) can perform high quality alignments based on words statistics and is considered the most efficient tool. Three main criticisms may be addressed to this kind of tool.

Firstly, as denoted by Moore (2005), one needs to tune numerous parameters in order to optimize the results for a particular alignment task, which can be very time consuming. This is all the more important when multilingual alignment is concerned, since every language pair will require its own set of parameter values.

Secondly, the best methods available nowadays can only work on language pairs, which results in a complexity explosion when multilingual alignments are needed. Simard (1999) showed how to adapt a bilingual method to align more than two versions of a text at the sentence level, but languages have to be processed by pairs to identify which ones are the most similar.

Thirdly, these approaches are also designed to align specific units of texts (almost always words). The same method cannot be applied indifferently on different units. Languages which do not separate words by spaces require to be first segmented into words, while a character-based approach could be a worthy alternative.

To deal with all these issues, we propose a method which is primarily intended to align many languages simultaneously from sentence-aligned parallel corpora, whatever the segmentation unit.

2 Alignment by string differences

2.1 String differences

In order to introduce the operation we use, let us start with a well-known similar technique, the Longest Common Subsequence (LCS) (Hirschberg, 1975). Given two strings A and B, it is always possible to find their longest subsequence. Such a subsequence is a sequence of non necessarily contiguous characters.

For instance, assume we have the following short English sentences (space characters are marked by an underscore and have the same status as any other character):

$$A = \text{I␣would␣like␣a␣donut,␣please.}$$
$$B = \text{Regular␣size,␣please.}$$

In this case, the LCS for A and B would be:[1]

$$\text{LCS}(A, B) = \text{ul␣ie,␣please.}$$

which is 14 characters long. It is then possible to form their string difference:

[1]For purpose of simplicity, we do not mention that the LCS operation, as well as the difference operation, may deliver a plurality of strings.

$$\begin{aligned} A \ominus B &= \text{I␣wodlk␣a␣donut} \\ B \ominus A &= \text{Regarsz} \end{aligned}$$

where we define $A \ominus B = A - LCS(A, B)$.

Because several isolated characters are discarded, this results in a malformed string that is of limited interest. To avoid misinformed strings, we rather resort to Longest Common Substrings (LCSubstr), that are contiguous.

On the previous example, the LCSubstr is:

$$\mathrm{LCSubstr}(A, B) = \text{,␣please.}$$

which is 9 characters long and by far more meaningful. By removing this substring from A and B, we obtain:[2]

$$\begin{aligned} A \ominus B &= \text{I␣would␣like␣a␣donut} \\ B \ominus A &= \text{Regular␣size} \end{aligned}$$

2.2 A monolingual operation for multilingual alignment

String differences are monolingual. They serve as a starting point to compute alignments. It suffices to apply them in parallel on the source text and all aligned target texts.

Let us consider anew the previous English sentences and their translations into Japanese ($\widehat{A}$ stands for "A's translation"):

$$\begin{aligned} \widehat{A} &= \text{ドーナツを下さい。} \\ &\qquad \textit{/dônatu wo kudasai./} \\ \widehat{B} &= \text{普通サイズを下さい。} \\ &\qquad \textit{/hutuu saizu wo kudasai./} \\ \mathrm{LCSubstr}(\widehat{A}, \widehat{B}) &= \text{を下さい。} \\ \widehat{A} \ominus \widehat{B} &= \text{ドーナツ} \\ \widehat{B} \ominus \widehat{A} &= \text{普通サイズ} \end{aligned}$$

By performing simultaneously the difference between A and B and between $\widehat{A}$ and $\widehat{B}$, their aligned translations, we obtain:

$$\begin{aligned} \text{,␣please.} &\leftrightarrow \text{を下さい。} \\ \text{I␣would␣like␣a␣donut} &\leftrightarrow \text{ドーナツ} \\ \text{Regular␣size} &\leftrightarrow \text{普通サイズ} \end{aligned}$$

The method assumes that the three strings computed in the source language are translations of their corresponding strings in the target language That is:

$$\begin{aligned} \widehat{\mathrm{LCSubstr}(A, B)} &= \mathrm{LCSubstr}(\widehat{A}, \widehat{B}) \\ \widehat{A \ominus B} &= \widehat{A} \ominus \widehat{B} \\ \widehat{B \ominus A} &= \widehat{B} \ominus \widehat{A} \end{aligned}$$

[2]Idem. Experiments show that they are not as numerous as for LCS. For the sake of simplicity, we shall assume that LCSubstr's are unique.

2.3 Iterative application

Assume we want to extract the translation equivalent of "Chicago" in Japanese from the following pairs of sentences:

$$\begin{aligned} A_0 &= \text{Is␣this␣a␣train␣for␣Chicago?} \\ \widehat{A_0} &= \text{この列車はシカゴ行きですか。} \\ &\qquad \textit{/kono ressya ha sikago yuki desu ka./} \\ B &= \text{Is␣this␣price␣correct?} \\ \widehat{B} &= \text{この値段で正しいですか。} \\ &\qquad \textit{/kono nedan de tadasii desu ka./} \\ C &= \text{What␣track␣does␣the␣train␣for␣Boston␣start␣from?} \\ \widehat{C} &= \text{ボストン行きの列車は何番から出ますか。} \\ &\qquad \textit{/bosuton yuki no ressya ha nani ban kara syutu masu ka./} \end{aligned}$$

A direct application of the method described above does not ensure that "Chicago" will correspond to a string difference. A way to solve the problem is to apply the method iteratively. String differences are applied on the sentence where "Chicago" appears (*i.e.*, A), in order to gradually cut it down to "Chicago" only. By applying the same process in parallel on all target sentences, strings are expected to reduce to the translation of "Chicago" as well. Also, we add the constraint that the string to be aligned must not be altered during the iterative process, *i.e.*, it should not be included in any LCSubstr.

Thus, starting with A_0 containing "Chicago", we perform at each step:

$$\begin{aligned} A_{n+1} &= A_n \ominus S_n \\ \widehat{A_{n+1}} &= \widehat{A_n} \ominus \widehat{S_n} \end{aligned}$$

where S_n is the first sentence S in the list of all source sentences sorted by the length of $\mathrm{LCSubstr}(A_n, S)$. In other words, amongst the available English sentences S, select the one that shares the longest substring with A_n, and remove this substring from A_n. The corresponding differences are applied in the target languages simultaneously.

Table 1 gives the details of an execution of the iterative process. On a large amount of data, the method may yield a plurality results; each of them may be obtained a certain number of times. We shall judge the quality of alignments based on these frequencies.

2.4 Best alignments selection

In practice, it is not possible to perform all the differences between sentences that would lead to the alignment of a particular string. This complexity explosion, where most of the LCSubstr's would be very short, would result in non reliable alignments.

n	A_n	S_n	LCSubstr(A_n, S_n)	$\widehat{A}_n$	LCSubstr$(\widehat{A}_n, \widehat{S_n})$
0	Is␣this␣a␣train␣for␣**Chicago**?	C	␣train␣for␣	この列車はシカゴ行きですか。	の列車は
1	Is␣this␣a**Chicago**?	B	Is␣this␣	こシカゴ行きですか。	ですか。
2	a**Chicago**?	B	?	こシカゴ行き	こ
3	a**Chicago**	C	a	シカゴ行き	行き
4	**Chicago**			シカゴ	

Table 1: Details of the steps necessary to extract one alignment for "Chicago" in Japanese. "Chicago" may not be modified during the iterative process and is not used to compute the LCSubstr's. The resulting alignment is $\widehat{A}_4 =$ シカゴ */sikago/*, which is correct.

We cut down the complexity by examining only the c first longest LCSubstr's longer than a predefined threshold. The threshold was set to half of the longest LCSubstr. Different values of c were tested for in the experiments reported in Section 3.3. This parameter is used in the source language only.

Well-formedness of strings is also tested by checking the presence of all their n-sequences of characters in the initial data. This is performed in the target languages.

Eventually, each alignment is scored by its frequency divided by the number of sentences that were required to obtain it. The reason for doing so is that, in practice, the less sentences required, the longer and the safer the LCSubstr's used.

3 Evaluation

3.1 Data used

We used the English, Japanese and Arabic training parts of the IWSLT 2007 machine translation campaign corpus (Fordyce, 2007) to conduct our experiments. It is made of nearly 20,000 triples of aligned sentences from the BTEC corpus (Takezawa et al., 2002).

3.2 Result samples

As mentioned earlier, one advantage of our method is that it can align any string of text, providing the data is sufficient. Table 2 shows a sample of alignments obtained using English as the source language. The strings requested to be aligned can be anything, from one character (see the first lines of the table) to entire sentences (see last line). Most alignments, if not perfect, differ from the expected meaning by slight differences only, even in Arabic.

3.3 Comparison against GIZA++

We compared our system to the state of the art device, GIZA++, in the particular case of bilingual word alignments on two pairs of languages: English to Arabic and English to Japanese. Our system aligned the three languages simultaneously, using English words as input. For each target language, the target unit with the best score (see Section 2.4) was kept. Different values of c were tested.

GIZA++ was used to compute IBM model 4. The default set of parameter values, which typically produces good results, was used. For each source word, the most probable pair of words (source, target) was kept. Note that the output of IBM model 4 produces word to word alignments, while there is no guaranty that our system would output a single target word as the unit of processing is the character.

For an objective assessment, we resort to two bilingual dictionaries: English-Japanese, and English-Arabic.[3] As for English-Japanese, the best results were obtained for $c = 40$, and are as good as those of GIZA++ (628 alignments found in the reference dictionary vs. 629, see Table 3). As for English-Arabic, the best results were obtained for $c = 20$, but only 37% of GIZA++'s results could be achieved (63 alignments found in the reference dictionary vs. 170).

Those alignments output by the two systems that do not belong to the reference dictionaries are not necessarily erroneous, since we relied on exact matching. Specifically, for our method, one extra character only may be responsible for an alignment to be considered wrong.

[3]We used an English-Arabic dictionary from sdict (87,000 entries): `http://sdict.com` and the EDICT English-Japanese dictionary (115,000 entries): `http://www.csse.monash.edu.au/~jwb/j_edict.html`. The Arabic part of the English-Arabic dictionary being lemmatized, we had to preprocess the Arabic part of our corpus so that it be lemmatized too (Debili and Achour, 1998).

English	Arabic	Japanese
.	. */./* ‘.’	。 */./* ‘.’
?	؟ */?/* ‘?’	か 。 */ka ./* ‘?’
Wh	أين */ʾayn/* ‘where’	何 */nani/*, */nan/* ‘what’, ‘wh…’
here	هنا */hnā/* ‘here’	ここ */koko/* ‘here’
I ’d like	أريد */ʾaryd/* ‘I’d like’	下さい */kudasai/* ‘please’
Thank you	شكرا */škrā/* ‘thank you’	ありがとう */arigatou/* ‘thank you’
Ice	آيس كريم */ʾāys krym/* ‘ice cream’	氷 を */koori wo/* ‘ice’
I have to get	عليصلقبل ال ساعة */ʿlyṣlqbl āl sāʿh/* * *malformed string* *	入手 し なければなりません */nyuusyu si nakerebanarimasen/* ‘I have to get’
At seven o’clock	في ال ساعة ال سابعة */fy āl sāʿh āl sābʿh/* ‘at seven o’clock’	七時 に */sitizi ni/* ‘at seven o’clock’

Table 2: A sample of alignments obtained using English as the source language. The Arabic and Japanese strings were generated in parallel by aligning the three languages at once. Parameter c was set to 20 in this experiment.

	GIZA++	Our system					
		$c = 1$	$c = 10$	$c = 20$	$c = 30$	$c = 40$	$c = 50$
English-Japanese	**629** / 4,268	369 / 1,569	573 / 2,629	603 / 3,038	598 / 3,165	**628** / 3,219	615 / 3,248
English-Arabic	**170** / 1,569	36 / 540	57 / 863	**63** / 982	57 / 1,017	51 / 1,025	51 / 1,029

Table 3: Comparison of our system against GIZA++. Each cell gives the number of alignments found in the reference dictionary over the number of alignments obtained.

4 Conclusion

We introduced a simple method to obtain subsentential alignments from several languages simultaneously. Its focus is on contexts rather than on units to be aligned. It avoids the complexity explosion due to the usual pair-by-pair processing by relying on the simultaneous application of a monolingual operation. The method comes close to GIZA++'s results in some word alignment task, while being by far much simpler.

References

Debili, Fathi and Hadhemi Achour. 1998. Voyellation automatique de l’arabe. In *Proceedings of the Workshop on Computational Approaches to Semitic Languages (COLING-ACL’98)*, pages 42–49, Montreal, Quebec, Canada, August.

Fordyce, Cameron Shaw. 2007. Overview of the IWSLT 2007 evaluation campaign. In *Proceedings of the 4th International Workshop on Spoken Language Translation (IWSLT 2007)*, pages 1–12, Trento, Italy, October.

Hirschberg, Dan. 1975. A linear space algorithm for computing maximal common subsequences. *Communications of the ACM*, 18:341–353, June.

Moore, Robert. 2005. A discriminative framework for bilingual word alignment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 81–88, Vancouver, British Columbia, October.

Och, Franz Josef and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29:19–51, March.

Simard, Michel. 1999. Text-translation alignment: Three languages are better than two. In *Proceedings of the Joint SIGDAT Conference of Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC)*, College Park, Maryland, USA.

Takezawa, Toshiyuki, Eiichiro Sumita, Fumiaki Sugaya, Hirofumi Yamamoto, and Seiichi Yamamoto. 2002. Toward a broad-coverage bilingual corpus for speech translation of travel conversation in the real world. In *Proceedings of the third International Conference on Language Resources and Evaluation (LREC 2002)*, pages 147–152, Las Palmas de Gran Canaria, Spain.