PBML

The Prague Bulletin of Mathematical Linguistics

NUMBER 93 JANUARY 2010 77–86

Combining Content-Based and URL-Based Heuristics to Harvest Aligned Bitexts from Multilingual Sites with Bitextor

Miquel Esplà-Gomis[a], Mikel L Forcada[a b]

[a] Grup Transducens, Departament de Llenguatges i Sistemes Informàtics, Universitat d'Alacant, E-03071 Alacant, Spain
[b] Centre for Next Generation Localisation, Dublin City University, Dublin 9, Ireland

Abstract

Nowadays, many websites in the Internet are multilingual and may be considered sources of parallel corpora. In this paper we will describe the free/open-source tool Bitextor, created to harvest aligned bitexts from these multilingual websites, which may be used to train corpus-based machine translation systems. This tool uses the work developed in previous approaches with modifications and improvements in order to obtain a tool as adaptable as possible to make it easier to process any kind of websites and work with any pairs of languages. Content-based and URL-based heuristics and algorithms applied to identify and align the parallel web pages in a website will be described and, finally, some results will be presented to show the functionality of the application and set the future work lines for this project.

1. Introduction and background

Nowadays the biggest and most heterogeneous text corpus in the world is the World Wide Web. In fact, during the last years there have been many approaches to profit from the web as a corpus and, especially, as a text corpus. In our case, our approach is focused on using the web as a source of *bitexts* (parallel texts). It is known that many websites are, totally or partially, available in more than one language. This means that some of their web pages can be paired into bitexts.

Currently, bitexts have become a very important source of knowledge for the machine translation. It is in the area of corpus-based machine translation where the bitexts are more important. *Example-based* machine translation (EBMT) and *statistical* machine translation (SMT) need this kind of resources, for the process of training (Hutchins and Somers, 1992). In fact, there are corpora which have been obtained

Corresponding author: miquel.espla@ua.es
Cite as: Miquel Esplà-Gomis, Mikel L Forcada. Combining Content-Based and URL-Based Heuristics to Harvest Aligned Bitexts from Multilingual Sites with Bitextor. The Prague Bulletin of Mathematical Linguistics No. 93, 2010, pp. 77–86. ISBN 978-80-904175-4-0. doi: 10.2478/v10108-010-0003-9.

from the Internet with the aim of training SMT, such as the Europarl Corpus (Koehn, 2005). There are even approaches to extract translation rules from parallel corpora used to create *rule-based* machine translation (RBMT) systems (Caseli and Nunes, 2007; Sánchez-Martínez and Forcada, 2009).

Based on this idea, different systems have been developed to harvest bitexts from the Internet. One of the earliest approaches is the STRAND system (Resnik and Smith, 2003), which is designed to identify web pages which are candidates to be bitexts. This system uses the HTML structure and the text-block length to compare files between them through the application of different calculations and thresholds. Similar approaches have been developed with this kind of methods to harvest bitext from the web (Chen and Nie, 2000; Kit et al., 2005; Désilets et al., 2008). In these cases, the system used to obtain the preliminary candidates for each web page is the identification and substitution of language markers in the URLs (Nie et al., 1999) (this will be covered in the section 3). In our approach we have not used this system in order to create an application as independent as possible of the website structure and the language pair searched.

Taking all these ideas, Bitextor was created as a free/open-source tool with the aim of obtaining the maximum number of parallel texts from multilingual websites, aligning them and generating translation memories (TMs) in TMX format.[1] To do this, the content comparison techniques developed in the cited projects have been applied with some modifications, combining them with other heuristics which will be explained in next sections. To assist in this task, another free/open-source application has been used: the TagAligner tool (Sanchez-Villamil et al., 2006), which both uses the tag structure in XML files and the length of the sentences in a pair of documents to align them (Brown et al., 1991; Gale and Church, 1994).

2. Obtaining and preprocessing web pages

To start the process of obtaining TM from a multilingual website with Bitextor, the first step is to download the entire website. To do this, Bitextor uses the tool HTTrack,[2] which is able to filter and download only the HTML files in the website. All these files are saved locally and are tagged with their URL.

Once this is done, some normalisation tasks are performed on the files in order to convert them into a valid format for processing. Firstly, Bitextor uses the library LibEnca[3] to detect the original character set encoding. It then uses LibTidy[4] to normalize the HTML files into valid XHTML files and to convert the detected original encoding into UTF-8.

[1]`http://www.lisa.org/Translation-Memory-e.34.0.html` [Last visited: 26th November 2009]

[2]`http://www.httrack.com` [Last visited: 26th November 2009]

[3]`http://sourceforge.net/projects/freshmeat_enca/` [Last visited: 26th November 2009]

[4]`http://tidy.sourceforge.net` [Last visited: 26th November 2009]

3. Choosing the parameters for the comparison

To look for web page pairs that are bitexts, Bitextor needs to obtain and save some features from these pages. In this section, we will explain how this is done.

3.1. Surface features

The first features observed in a web page are those which we will consider, in this paper, *surface features*. These are the features that may not be used for an accurate comparison, but can be used as a indicator to discard very unlikely pairs of files. In our approach, these features are:

- *Text-language comparison*: It is obvious that if two files are written in the same language, one of them can not be a translation of the other one. The language in which each text has been written is detected and stored by using LibTextCat.[5]
- *File size ratio*: This parameter is relative and is used to filter pairs of files whose size is very different.
- *Total text length difference*: This parameter has the same function that the previous one, but compares the size of each file's plain text in characters.

3.2. Web page content

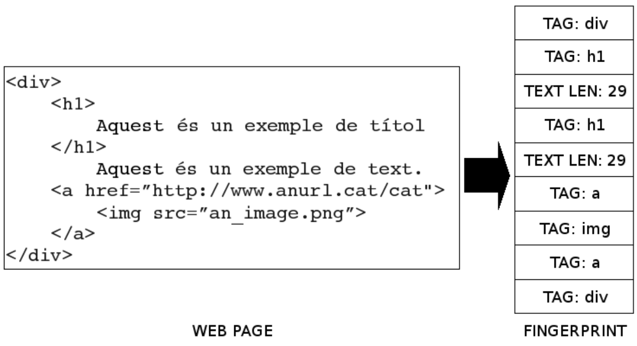
In order to obtain a more precise comparison Bitextor uses web page content in the comparison process. Web pages have an advantage over plain text: they are tagged with format and structure tags, which provide additional information that can be used to compare them. The idea is that two parallel web pages should have the same HTML tag structure (or, at least, a similar one).

Basically, two elements in the content of the web pages are considered in our approach: the HTML tag structure and the text block length. This is the same information used in the STRAND approach. In Bitextor, the extraction of this information is divided into two steps: the file cleaning and the encoding. In the first one, the objective is to remove all the irrelevant information, such as comments, the heading of the web page, the tag parameters, the irrelevant HTML tags and the extra spaces in the text blocks. In the second step Bitextor encodes the remaining information into a string in which two kinds of information can be represented: the tag names and the text block lengths (measured in characters[6]). This string acts as a *fingerprint* of the web page.[7] We can see an example of this kind of encoding in the Figure 1. This method of encoding provides the possibility of using the edit-distance algorithm to make the comparison.

[5] `http://software.wise-guys.nl/libtextcat/` [Last visited: 26th November 2009]

[6] An interesting study issue could be to analyse the differences in the results calculating the length of the text blocks in characters or in words

[7] To optimise calculations, this information is encoded with integers.



```
<div>
    <h1>
        Aquest és un exemple de títol
    </h1>
        Aquest és un exemple de text.
    <a href="http://www.anurl.cat/cat">
        <img src="an_image.png">
    </a>
</div>
```

Figure 1. Example of conversion from HTML to a fingerprint string.

3.3. URL

One difference between this work and other previous approaches is that Bitextor does not download the candidate files by using rules of detection and substitution of language markers in the URL (Nie et al., 1999). Bitextor downloads the whole website and, then, uses the URLs as one more parameter to discard pairs of files with low probability to be bitexts. In order to do this, Bitextor divides the URL of a file into three sections: thedirectory path, the filename and the variables. In this way, it can compare each section separately.

4. Web page comparison process

To compare the web pages, the features explained in the previous section are compared one by one. Firstly, the surface features are compared in order to discard the most obviously incorrect pairs of files. With the remaining files the following two methods are applied.

4.1. URLs comparison

For the URL comparison Bitextor applies a restriction: candidate pairs can have at most, one difference in their URL. In practice, this implies one of these three possibilities in which the difference can be:
The filename: This is the simplest difference. When both files are saved in the same directory but they have a different name:

`http://www.gnu.org/home.ca.html` ⇒ `http://www.gnu.org/home.en.html`

A directory: This means that the directory structure differs in the name of a directory. For example it happens when files are saved in a path with the same structure but that is forked in a particular level in the directory tree. This can also happen when one of the files is saved in a subdirectory of the directory where the other one is saved:

`http://www.ua.es/va/index.html` ⇒ `http://www.ua.es/en/index.html`

`http://www.ua.es/index.html` ⇒ `http://www.ua.es/en/index.html`

A variable: This difference consists in the fact that the same file is called using a variable with a different value in each of the cases. It can also happen when one file has one more variable:

`http://www.dlsi.ua.es/index.html?id=val` ⇒ `http://www.dlsi.ua.es/index.html?id=eng`

`http://www.dlsi.ua.es/index.html` ⇒ `http://www.dlsi.ua.es/index.html?id=eng`

With this system, Bitextor tries to take advantage of the information provided by the URL without having to manually generate the rules of pattern recognition and substitution of language markers in the web pages URL. It means that Bitextor can be used directly on any website without having to analyse its structure.

4.2. Web page content comparison

Finally, those pairs that have not been discarded in the previous step are compared through their fingerprint (see section 3.2). To do this, Bitextor uses the Levenshtein edit distance algorithm (Levenshtein, 1966). It is important to explain that, when Bitextor applies this algorithm on the obtained fingerprints, it has to process two kinds of elements (tags and text blocks). The comparison between XHTML tags is simple: they can be different or equal. However, the comparison between text blocks is not as easy. As in other methods (Gale and Church, 1994), the length is the parameter used to perform the comparison between text blocks, so the most reasonable option seems to be to use the following measure of divergence between the length two text block lengths b_1 and b_2:

$$D(b_1, b_2) = \frac{|b_1 - b_2|}{\max(b_1, b_2)} \tag{1}$$

In fact, in our approach we implement two ways to use this information in order to obtain two different values as a result of the edit distance calculation. The first way is to set a threshold for $D(b_1, b_2)$ for each pair of languages. In this way, we can evaluate if two texts blocks could be parallel or not. The value obtained from applying the edit-distance with this method is used to discard improbable pairs by defining a maximum number of absolute differences between both fingerprints. The other way to compare the text block lengths is to directly use $D(b_1, b_2)$ as a cost. The value obtained from the edit-distance calculation with this method is used to know which is the most probable candidate for a given file from the group of files that may have passed all the other heuristics (the one having the lowest value).

In this way, for the operations defined for the Levenshtein edit-distance (insertion, deletion and substitution) we can define the following cost functions: for insertions $C_i(x)$ and deletions $C_d(x)$, the cost is the same for tags and a text block lengths, independently of its length x:

$$C_i(x) = 1 \qquad\qquad C_d(x) = 1 \tag{2}$$

For substitutions of tags (t) we will have the cost function $C_s(t_1, t_2)$:

$$C_s(t_1, t_2) = \begin{cases} 0 & \text{if } t_1 = t_2 \\ 1 & \text{if } t_1 \neq t_2 \end{cases} \tag{3}$$

In the case of text block lengths b_1 and b_2, as we have said, we have two functions: the direct cost function without using the threshold $C_s(b_1, b_2)$ and the cost function using the threshold $C'_s(b_1, b_2)$:

$$C_s(b_1, b_2) = D(b_1, b_2) \qquad\qquad C'_s(b_1, b_2) = \begin{cases} 1 & \text{if } D(b_1, b_2) > t_b \\ 0 & \text{if } D(b_1, b_2) \leq t_b \end{cases} \tag{4}$$

Substitutions between tags and text block lengths are not allowed:[8]

$$C_s(t_1, b_1) \to \infty \qquad\qquad C_s(b_1, t_1) \to \infty \tag{5}$$

5. Aligning the obtained websites

The last task performed by Bitextor is the alignment of the candidates. In order to align a pair of XHTML files, Bitextor uses the LibTagAligner to perform the alignment. The method used by TagAligner to align files is similar to the one used by Bitextor to compare them. TagAligner encodes the file with a fingerprint (as Bitextor does), but it uses a more detailed weight structure with the edit-distance algorithm. In contrast to Bitextor, when this algorithm is performed, not all tags are compared in the same way. This tool allows the user to group the tags in categories. For these categories, the user can define weights for the operations defined in the edit-distance algorithms.

Thus, for a tag t in a category k, the cost of an insertion $C_i(t)$ or deletion $C_d(t)$ operation is expressed by the functions:

$$C_i(k) = W_i(t) \qquad\qquad C_d(t) = W_d(k) \tag{6}$$

[8]In our approach, we assign the C++ `MAXDOUBLE` value to these cost functions.

where $W_i(k)$ and $W_d(k)$ are the functions that return the weights assigned by the user for the insertion and deletion operations.

In the case of substitution, the cost function for two tags (t_1 and t_2) in two categories (k_1 and k_2) is:

$$C_s(t_1, t_2) = \begin{cases} 0 & \text{if } t_1 = t_2 \\ W_s(k_1, k_2) & \text{if } t_1 \neq t_2 \end{cases} \tag{7}$$

where $W_s(k_1, k_2)$ is the function that determines the cost of a substitution on a pair of different tags belonging to the same category or two different categories.[9]

Weights are also assigned to text block length operations, and they are relative to the length of the blocks operated. But, in contrast to the fingerprint comparison used by Bitextor, in LibTagAligner the user can choose whether the length of the text block is measured in characters or words. So, in our case, the set of cost functions for text blocks b is:

$$C_i(b) = W_i(b) \cdot b \qquad C_d(b) = W_d(b) \cdot b \tag{8}$$

$$C_s(b_1, b_2) = W_s(b_1, b_2) \cdot |b_1 - b_2| \tag{9}$$

Again, tag–text block substitutions are not allowed, so, the cost of the operation will be implemented as an infinite (as has been explained in section 4.2).

6. Results

This section presents results from the system. These tests have been performed by using version 3.2.0 of Bitextor.[10] What we are going to analyse is the capacity of Bitextor to find the parallel web pages in a given website. In terms of alignment quality, there is a complete study (Sanchez-Villamil et al., 2006) with results about this issue. The metrics used to evaluate Bitextor have been precision and recall. We define precision (P) as the number of correct pairs obtained (N_C) over the total number of pairs obtained (N_T). The recall (R) is then the number of correct pairs obtained (N_C) over the total number of possible pairs in the website (N):

$$P = \frac{N_C}{N_T} \qquad R = \frac{N_C}{N} \tag{10}$$

It is obvious that it would be a huge work to check all the pairs of files generated by Bitextor, or find the total number of possible pairs of files in a website composed of thousands of web pages. To obtain an approximate estimation of the precision, we have randomly obtained a sample of 100 pairs generated by Bitextor and have checked them by hand. In the same way, we have obtained a list of 300 web pages

[9] An optimal set of weights can be found in (Sanchez-Villamil et al., 2006).

[10] The configuration file used to perform the tests can be looked up in the trunk of the SVN server of Bitextor for its revision 146: `https://bitextor.svn.sourceforge.net/svnroot/bitextor/trunk/`

from the downloaded website and have tried to find them in the list of pairs generated by Bitextor. Then, have checked if these pairs were correct or not.

For a first test, we have tried with a very simple case: the website of the Parliament of Canada.[11] This country has two official languages (English and French), and this website must have all its pages in the both languages, so, in theory, all the pages must have a bitext candidate. The website was downloaded by using HTTrack and 56,173 HTML files were obtained. Bitextor was applied with a threshold of 10 maximum differences between fingerprints. From the website, 24,717 pairs of web pages were found. The results in this case were very satisfactory: P=99% and R=85,33%.

These are very promising results, but, probably, the quality of the extraction is probably due to the fact that this is a very well structured website, with uniform language markers in the URL and highly parallel contents. Because of this, we wanted to try with a more complex case. The next results were obtained from a website from the Universitat d'Alacant,[12] which is written in three languages: English, Catalan and Spanish. This website is heterogeneous, with some pages without any translation and with multiple systems to mark the language in the URL. In this way, these were the obtained results: $P = 86\%$ and $R = 61\%$. Obviously, these results are worse than the obtained in the previous test. There are some reasons to explain what has happened in this case with the precision: the noise caused by web pages without any translation, the fact that some pages have no language marker in the URL, the presence of pages with a very similar content and the same structure (for example, the staff section, which uses a template for all the web pages and only changes a few lines of text).

Analysing the results one by one, we have noticed that, in many occasions it is better to have a lower recall because in many of the discarded pairs, the information contained in the pages was minimal (only some words), with mixed languages, only numeric data, etc. So, it is important to understand that the recall can be more related with the quality of the website as a parallel corpus than with the performance of the application.

7. Conclusions

After this study, we can extract some conclusions. Firstly, it is clear that this system gives promising results for webpages with a high number of parallel pages and with not much noise. Certainly, it is probable that many of the multilingual webpages in the Internet do not fit this profile. Thus, it seems that one of the most important future lines of work in this project will be to develop new heuristics to clean all the possibly noisy files.

Regarding a comparison of previous works in the area and Bitextor, we have obtained some good results, comparable to those obtained with other similar approaches

[11]`http://www.parl.gc.ca`

[12]That of the Department of Computer Languages and Systems, `http://www.dlsi.ua.es`

(although it is difficult to quantify without applying them on the same websites in a controlled work environment). In addition, the system of comparison of URLs of Bitextor has been designed to be more adaptable and, as consequence, obtain better results for any website without studying its structure.

One important point in our approach is the fact that it is a free/open-source tool. We think that free/open-source is very important in this kind of applications, since we are working with a very heterogeneous material: websites are very different between them, different corpora with different languages can present very different problems (for example, the alignment), etc. With a free application we are allowing people to try our application and to add new features to face all the possible problems.

8. Where to find Bitextor and TagAligner

Bitextor and LibTagAligner are under the GNU General Public License (GPL) version 2.0[13] and they are available for UNIX-like platforms. Its code and releases can be found at `http://sourceforge.net/projects/bitextor` and `http://sourceforge.net/projects/tag-aligner`.

9. Future work

Currently, there are various tasks pending for the Bitextor project. We are exploring ways to increase the precision of our system in order to obtain better results on noisy websites. Another important task planned is the integration of Bitextor with other free tools, like Bitext2TMX[14] to create a more powerful work environment for the creation and editing of translation memories.

Another important improvement would be to add a new module to allow Bitextor to acquire by itself candidate websites to be parallel (for a given pair of languages) (Leturia et al., 2009).

Acknowledgements: The original development of Bitextor was funded by the Ministerio de Ciencia y Tecnología (Spanish Government) between 2004 and 2006 through grant (TIC2003-08681-C02). Later, it was funded by the Universitat d'Alacant. Enrique Sánchez Villamil was the author of the 1.0 version of Bitextor and the 1.0 version of TagAligner (on which the initial version of LibTagAligner, used by Bitextor was based). Miquel Simón i Martínez was the author of the second version (v2.0) of TagAligner, with an improvement in configurability through the incorporation of an XML configuration file. MLF's stay at Dublin City University is funded by Science Foundation Ireland through an ETS Walton Award. We thank the support of the

[13] `http://www.gnu.org/licenses/gpl-2.0.html` [Last visited: 26th November 2009]

[14] `http://www.sf.net/projects/bitext2tmx`

Spanish Ministry of Science and Innovation through project TIN2009-14009-C02-01. We thank Francis M. Tyers for comments on the manuscript.

Bibliography

Brown, P.F., J.C. Lai, and R.L. Mercer. Aligning sentences in parallel corpora. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pages 169–176. Association for Computational Linguistics Morristown, NJ, USA, 1991.

Caseli, HM and MGV Nunes. Automatic induction of bilingual lexicons for machine translation. *Int J Transl*, 19:29–43, 2007.

Chen, J. and J.Y. Nie. Parallel web text mining for cross-language IR. In *Proceedings of RIAO 2000: Content-Based Multimedia Information Access*, volume 1, pages 62–78, 2000.

Désilets, A., B. Farley, M. Stojanovic, and G. Patenaude. WeBiText: Building Large Heterogeneous Translation Memories from Parallel Web Content. *Proc. of Translating and the Computer*, 30:27–28, 2008.

Gale, W.A. and K.W. Church. A program for aligning sentences in bilingual corpora. *Computational linguistics*, 19(1):75–102, 1994.

Hutchins, W.J. and H.L. Somers. *An introduction to machine translation*. Academic Press New York, 1992.

Kit, Chunyu, Xiaoyue Liu, KingKui Sin, and Jonathan J. Webster. Harvesting the bitexts of the laws of Hong Kong from the Web. 2005.

Koehn, P. Europarl: A parallel corpus for statistical machine translation. In *MT summit*, volume 5. Citeseer, 2005.

Leturia, I., I. San Vicente, and X. Saralegi. Search engine based approaches for collecting domain-specific Basque-English comparable corpora from the Internet. In *Proceedings of the 5th Web As a Corpus, Sociedad Española de Procesamiento del Lenguaje Natural Conference*, 2009.

Levenshtein, V.I. Binary codes capable of correcting deletions, insertions and reversals. In *Soviet Physics Doklady*, volume 10, pages 707–710, 1966.

Nie, J.Y., M. Simard, P. Isabelle, and R. Durand. Cross-Language Information Retrieval based on Parallel Texts and Automatic Mining of Parallel Texts from the Web. In *Proceedings of SIGIR'99: 22nd International Conference on Research and Development in Information Retrieval: University of California, Berkeley, August 1999*, page 74. Association for Computing Machinery (ACM), 1999.

Resnik, P. and N.A. Smith. The web as a parallel corpus. *Computational Linguistics*, 29(3): 349–380, 2003.

Sánchez-Martínez, Felipe and Mikel L. Forcada. Inferring shallow-transfer machine translation rules from small parallel corpora. *Journal of Artificial Intelligence Research*, 34:605–635, 2009.

Sanchez-Villamil, E., S. Santos-Anton, S. Ortiz-Rojas, and M.L. Forcada. Evaluation of alignment methods for HTML parallel text. *Lecture Notes in Computer Science*, 4139:280, 2006.