Identifying Parallel Web Pages

Marcela Macedo Vieira and Viviane Pereira Moreira

Universidade Federal do Rio Grande do Sul, Brazil
{mmvieira,viviane}@inf.ufrgs.br

Abstract. Research on statistical machine translation and corpus-based approaches for cross-language information retrieval depend on the availability of multilingual data, particularly in the form of parallel corpora (collections of equivalent texts in two or more languages). However, the scarcity of parallel corpora limits the development of these applications. The Web is a vast repository of multilingual information, which has motivated research aimed at mining corpora from it. In this article, we present PPLocator an approach for locating parallel Web pages. PPLocator was designed to be effective while keeping a low processing cost, thus it avoids making exhaustive pairwise comparisons in order to identify the candidate pairs. In addition, it tries to minimize the number of pages that need to be downloaded during the intra-site crawl. An important characteristic of our approach is that it does not rely on resources such as dictionaries, translators, or language identifiers. PPLocator demands little effort from the human expert. Experiments using real Web data from over 284K pages attest for the viability of PPLocator. The results show superiority in relation to a baseline system in terms of both recall and precision, despite the fact that the baseline uses more resources.

Categories and Subject Descriptors: H.3 [**Information Storage and Retrieval**]: Miscellaneous; I.7 [**Document and Text Processing**]: Miscellaneous

Keywords: classification, parallel corpora, similarity functions

1. INTRODUCTION

Parallel Corpora are document collections composed of translation-equivalent texts in two or more languages. This is a fundamental resource for training statistical Machine Translation systems and corpus-based strategies for Cross-Language Information Retrieval. Machine Translation Systems typically require a parallel corpus to generate statistical translation models [Caseli et al. 2006; Li et al. 2007; Melamed 2000]. In Cross-Language Information Retrieval (in which the language of the query is different from the language of the documents), one of the most efficient approaches consists in analyzing a parallel corpus to automatically discover matching terms across languages [Kraaij et al. 2003].

However, this is a rare resource. Freely available parallel corpora are limited to the Proceedings of the European[1] and Canadian[2] parliaments and The Bible, which is available in dozens of languages [Resnik et al. 1999]. It is also possible to purchase parallel corpora from providers of linguistic resources such as ELRA[3] and LDC[4].

The fact that the Web is a massive repository of multilingual information has motivated research which aims at mining parallel corpora from it. While about 56% of the Web content is in English[5],

[1] http://www.statmt.org/europarl/
[2] http://www.isi.edu/natural-language/download/hansard/
[3] http://www.elra.info/
[4] http://www.ldc.upenn.edu/
[5] http://w3techs.com/technologies/overview/content_language/all

This work was partially funded by CNPq (project no. 479703/2010-8).

(a) English (b) French

Fig. 1. Example of parallel pages in English and French (http://www.rmc.ca/aca/cc-ccg/index-eng.asp and http://www.rmc.ca/aca/cc-ccg/index-fra.asp)

the participation of other languages is growing. Web documents are freely available, the challenge is to efficiently identify parallel corpora. Gathering this type of resource freely from the web, for a language-pair, is what motivates our work.

In this article, we propose a method called Parallel Page Locator (PPLocator) which takes as input a URL from a website and a pair of languages of interest and efficiently searches for pairs of parallel web pages within that site, thus creating a parallel corpus. Figure 1 shows an example of parallel web pages in the English-French language pair found by the method. PPLocator consists basically of two steps. First, candidate pairs are located based solely on URL similarity. Then, the candidate pairs are submitted to a classifier which will analyze the features of the candidate pair to decide whether the pair is indeed parallel.

Our approach is based on two assumptions. The first one is that the websites which contain parallel web pages follow some naming convention for the different language versions of a page. Thus, the URLs will exhibit evidences about the language of the page. This hypothesis has also been used by some important related work [Resnik 1998; Chen and Nie 2000; Chen et al. 2004] which report good results. The second assumption is that parallel pages belong to the same website (i.e., are under the same domain). We are aware that this assumption can potentially limit the number of parallel pairs identified, but it narrows down the search space to a tractable size.

The main contribution of this article is an efficient method for identifying parallel documents. For this purpose, we use evidences from different sources (URL, structure, and content). Unlike existing

approaches, we do not employ linguistic resources such as bilingual dictionaries, automatic translators, or language identifiers. This can be particularly important because it makes PPLocator independent from external tools and resources.

The task of finding parallel web pages involves a number of challenges. Even if we limit the search space for candidate pairs to a single website, the volume of data can be very high as some sites have thousands of pages. Thus, downloading the entire site and making pairwise comparisons with all possible pairs becomes too costly. To circumvent this problem, in PPLocator, we avoid downloading pages which do not display characteristics of being parallel. In addition, the URLs of the pages identified in the website are divided into lists according to their characteristics and this way we avoid exhaustive pairwise comparisons.

We carried out experiments with real web data from 361 websites containing over 284K pages and compiled a parallel corpus of English and French pages for the domain *university*. Our results confirm the hypothesis that the URL has helpful clues as to whether the pages can be considered parallel. We compared PPLocator against a baseline method which uses a language identifier and a bilingual term list. The superiority of our results attest for the quality of our proposed method.

This article is organized as follows: Section 2 covers the existing literature on parallel web page identification. In Section 3, we introduce PPLocator detailing its phases and algorithms. An experimental analysis is reported in Section 4. Finally, Section 5 concludes the article.

2. RELATED WORK

One of the first systems for mining the web for bilingual text was STRAND, proposed by Resnik [Resnik 1998]. The input to this system is a pair of languages of interest. The system then issues a query to Altavista to retrieve pages in which one language appears in the text or URL and another language appears on the anchor text[6]. All pairs of URLs appearing at most 10 lines away from each other were put in a list of candidate parallel pages. The candidate pairs go trough an evaluation phase that takes into consideration the HTML markup (which should be similar for parallel pages) and the correlation between the lengths of the texts. Experiments on 90 candidate pairs of English-Spanish pages achieved 88.2% precision and 62.5% recall. A further study by the same author [Resnik 1999] added a language identification module to STRAND. In an evaluation of 179 candidate pairs, the system achieved a precision of 92% and a recall of 47.3%. In [Resnik and Smith 2003], STRAND was applied to the Internet Archives and found 1,399 pairs of English-Arabic pages. A manual evaluation of 149 pairs yielded 95% precision and 98% recall.

Similarly to STRAND, PTMiner [Chen and Nie 2000] also uses a query sent to a search engine to retrieve candidate web pages. An initial filtering is done based on URL, size, and date similarity. A second filtering based on text length, language identification, and HTML structure is then performed. Experiments on English-French and English-Chinese have assembled parallel corpora of approximately 14K documents. An evaluation of 100 pairs yielded precision and recall of 95 and 90% respectively.

The BITS system [Ma and Liberman 1999] receives a language-pair as starting point. Then it crawls a domain which is likely to have parallel texts in the languages of interest, i.e., for English-French the .ca domain could be used. A list of all www servers is obtained. The next step is to identify if the website is indeed multilingual, which is done by applying a language identifier at pages from the top 3 or 4 levels of the website. If the site is found to be at least bilingual, wget is used to retrieve all pages in the website. Finding translation pairs is done in a two-stage process: (i) filename similarity, and (ii) content similarity using a translation lexicon and cognate analysis. To avoid exhaustive comparisons, the size of the files and the number of anchors are used to prune out impossible pairs. The pair-finding algorithm was tested on 300 pairs of German-English pages yielding 97.1% recall and 99.1% precision.

[6]Search within the anchor text is no longer supported by Altavista.

An experiment on 30,000 websites from the "de" domain identified 3,415 bilingual websites, from which 1,547 were found to contain parallel text.

Chen et al. [Chen et al. 2004] developed a parallel page identification system, known as Parallel Text Identification (PTI) system, that identifies and aligns parallel documents on the web. A spider crawls the web and passes the documents to a filename comparison module. This comparison relies on language-indicating substrings such as `en-` ↔ `ch-` for English and Chinese, respectively. When the filename for one page is identified, the system generates a list of possible filenames for the translated version of the page using the language-indicating substrings. Documents that did not get a positive identification on the filename comparison module go through a content analysis module which uses a bilingual word list. Experiments on 427 documents in English and Chinese fetched from the Hong Kong Government website (www.info.gov.hk) achieved 96% recall and 93% precision in the alignments.

Tomás et al. [Jesús Tomás and Casacuberta 2005] manually created a list of bilingual websites combining the query on anchor text used in [Chen and Nie 2000; Resnik 1998; 1999] and the crawl of domains that are likely to have parallel pages used in [Ma and Liberman 1999]. All pages in the websites from the list are downloaded and compared to all previously downloaded pages. To determine parallelism, three filters are used: (i) comparison of file size, number of HTML tags and number of paragraphs; (ii) language identifier to ensure that the pages are in different languages; and (iii) for pages that passed the previous filters, their HTML structures are compared. A further step involves content-based matching which relies on a statistical translation model that is trained on-the-fly. Experiments on 8 selected websites identified 652 parallel pages with a precision of 89% and a recall of 78%.

Fry [Fry 2005] developed a system that assembles parallel corpora of news feeds. The system collected Japanese news from four websites along with their English translations. A corpus of 17,277 parallel documents was gathered and made available online. This approach is simpler as it does not need to align the texts since the correspondences are implied by the cross-language links on the news articles. The main shortcoming is that it was designed specifically for the four news websites experimented on and would therefore need adaptations to work on other websites.

Shi et al. [Shi et al. 2006] propose comparing the DOM trees of the web pages to find out whether they are parallel. In their experiments, 300K URLs of Chinese websites were obtained from web directories (Yahoo!) in China, Hong Kong, and Taiwan. They discovered 11K bilingual websites from which they mined 63,214 parallel documents. An evaluation on a sample 3K of these documents shows and estimated precision of 97% against 93% achieved by URL comparison.

Zhang et al. [Zhang et al. 2006] created the WPDE system which shares many similarities with the previously existing ones, adding slight improvements to each phase. During candidate selection, the image ALT text is also considered. In the alignment phase, they use a combination of pattern matching on the URL and edit distance measures. The candidate pairs are then filtered based on file size, structure, content translation, and a KNN classifier. Experiments took as starting point a snapshot containing 2 million web pages from Microsoft Research, from which they mined 1809 candidate websites. A random sample of 26 .hk sites and 35 .cn sites was used as seed for a crawl of 53,000 web pages. The authors report a precision of 95% and a recall of 97% on a sample of 6.5K candidate pairs.

More recently, Tsvetkov & Winter [Tsvetkov and Wintner 2010] devised a method for acquiring parallel news corpora. Candidate web-sites are manually identified and, in contrast to the methods previously discussed, it does not take into account the structure of the pages, the links to the translated page or URL similarity. Identification of pairs is done relying solely on content. News articles are crawled daily from a website that publishes news in Hebrew and their English versions (not all articles are translated and some are translated only partially). Prior to analyzing content similarity, the articles in Hebrew are translated into English using a bilingual dictionary (all possible translations

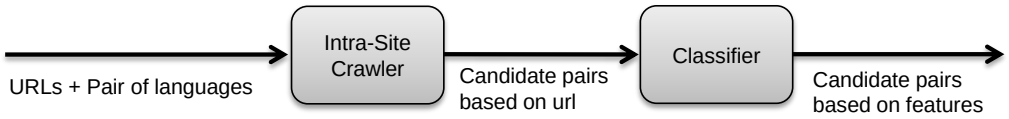


Fig. 2. Basic architecture of PPLocator

are kept). Similarly, the English articles are translated into Hebrew. Two texts are identified as mutual translations if the Hebrew and English versions share more words than a predefined threshold. This approach requires pair-wise comparisons with all documents (however they only compared articles that are published in close dates). An evaluation experiment gathered 6129 news for 3 months. Amongst those, 505 pairs were identified with 100% precision and 86.5% recall.

Existing approaches have a number of shortcomings. Some techniques [Chen et al. 2004; Ma and Liberman 1999; Tsvetkov and Wintner 2010] rely on some form of bilingual dictionaries, or even use automatic translators [Zhang et al. 2006]. This is an important limitation since if there are bilingual term lists for a pair of languages, the need for a parallel corpus to derive such lists becomes less critical. Another important aspect is about the domain (i.e., topic) of the website. It is unlikely that bilingual term lists will have a good coverage for specialized terminology (e.g. biology, medicine, physics, etc.). Many approaches [Resnik 1999; Chen and Nie 2000; Ma and Liberman 1999; Chen et al. 2004; Jesús Tomás and Casacuberta 2005] rely on language identification, which can make mistakes since languages can have similar words, or the text in the web page may be too short to allow for an accurate identification. The criteria for finding candidates used by STRAND [Resnik 1998] is too limiting and could only be successful if the URLs for both pages are found in a parent page. None of the parallel pages in the websites we analyzed for our experiments satisfy this criteria. On the other hand, trying to maximize recall by doing exhaustive pairwise comparisons, as done in [Ma and Liberman 1999], is wasteful. Finally, while structural similarities between pages like the one proposed in [Shi et al. 2006] are helpful, they cannot be used alone to determine parallelism.

As mentioned in the Introduction, unlike existing approaches, PPLocator does not employ bilingual dictionaries and automatic translators. It also does not rely on mutual links between pages. Further differences include the fact that it does not consider a number of features used by others to prune unlikely candidates. Such features include: date similarity [Resnik 1998], the number of anchors [Ma and Liberman 1999], the image ALT text [Zhang et al. 2006], the number of HTML tags, and the number of paragraphs [Jesús Tomás and Casacuberta 2005].

Similar to other approaches, PPLocator uses language-indicating substrings contained in the URL [Resnik 1998; Ma and Liberman 1999; Chen and Nie 2000; Chen et al. 2004; Zhang et al. 2006]. Other similarities include considering structural similarities [Resnik 1998; Chen and Nie 2000; Jesús Tomás and Casacuberta 2005; Shi et al. 2006] and a correlation on the lengths of the pages [Resnik 1998; Chen and Nie 2000; Jesús Tomás and Casacuberta 2005].

3. IDENTIFYING PARALLEL WEB PAGES

Our solution for identifying parallel web pages is composed of two modules: an intra-site crawler and a classifier. These modules, together with their inputs and outputs, are shown in Figure 2. The idea is that the intra-site crawler acts as an initial filter and should aim at high recall. Thus, it should minimize the number of positive pairs that are discarded. On the other hand, the classifier should aim for high precision since in order to be useful, the parallel corpus generated should indeed contain parallel pages.

PPLocator works as follows: given a URL corresponding to the entry point of the website being

Algorithm 1 Intra-site Crawling Algorithm

```
1: function INTRA-SITE CRAWL(url, lang1, lang2)
2:     listOfUrls = ExhaustiveCrawl(url, n)
3:     normalize(listOfUrls)
4:     [listLang1, listLang2, listUnk] = guessLanguage(listOfUrls, lang1, lang2)
5:     if hasTwoPopulatedLists(listLang1, listLang2, listUnk) then
6:         [listLang1, listLang2] = populateLists(listLang1, listLang2, listUnk)
7:         candidatePairs += findPairs(listLang1, listLang2, t_sim)
8:     end if
9:     return candidatePairs
10: end function
```

investigated and a pair of languages of interest, the intra-site crawler examines the web pages from the site trying to find pairs of URLs which will likely have parallel content. This is done analyzing the URLs themselves looking for similarities and for the presence of language-identifying tokens (e.g. *pt* or *ptg* for Portuguese). If the intra-site crawling algorithm was able to identify any candidate pair, the pair will be submitted to a previously trained classifier which will decide whether the pair has parallel content. Parallel web pages should be similar in terms of structure, content, and some features such as the ratio between the number of words should be stable for a given language-pair. Thus, our classifier uses features that aim at capturing this notion.

3.1 Intra-site Crawler

The intra-site crawler requires the following inputs: a URL, a pair of languages of interest, and the number of levels to be crawled (n). Its output is a set of pairs of URLs for pages which likely contain parallel texts. Details are given in Algorithm 1. The first step is to do an exhaustive crawl in the first $n-1$ levels of the website. The exhaustive crawl is basically a BFS traversal of the site considering only the internal links. The behavior of the intra-site crawler is depicted in Figure 3. The URLs collected are added to a list (listOfUrls). Note that we only look at the URLs — the content of the page is not considered. The rationale is that making decisions based on the URL alone makes processing more efficient. The text of the URL is much shorter than the contents of the page and, in addition, it avoids the need for downloading the web page. Note that the pages on the n-th level are only downloaded if their URLs (which are collected from pages on the $n-1$ level) are listed as candidates to form a parallel pair.

Once the list of URLs is obtained, the URLs are normalized to remove unnecessary tokens which would lower their similarity score. Tokens like "www", "index", and "http", are discarded. For example, consider the two URLs below:
(a)`www.travelingtolisbon.com/promocoes.php`
(b)`http://www.travelingtolisbon.com/pt/promocoes.php`
The URLs in (a) and (b) probably contain parallel text, however, only (b) has the token "http". Normalization makes (a) and (b) more similar, which in turn increases the similarity score between them.

This list of URLs is then processed taking the languages of interest into consideration. The idea is to examine the text of the URL looking for clues which indicate the language of the page. In order to do that, we rely on lists of tokens which are typically used to identify a language. For example, the list of tokens for the English language would contain "en", "us", "eng", *etc.*, which could appear in URLs such as http://www.brasil-natal.com.br/**en**/, http://www.newhotel.com/**us**/Info.aspx, and http://**eng**.ondehospedar.com.br/ac/index.php. These lists were manually constructed, however, the effort in building such lists is small as they tend to be very short (between 1 and 9 tokens).

Using the two lists of tokens for the language-pair of interest, the algorithm tries to identify the

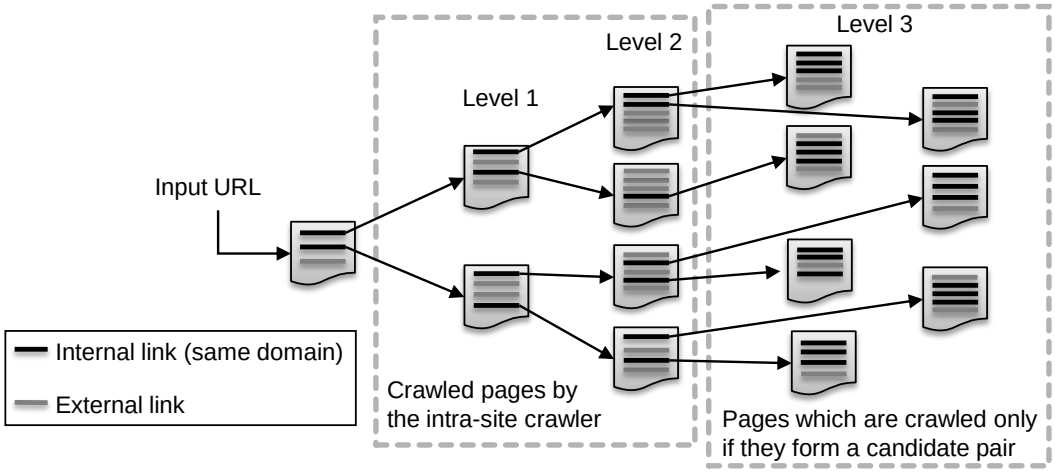


Fig. 3. Behavior of the intra-site crawler. Note that the pages on the third level are only downloaded if their URLs (which were collected from pages in the second level) have an indication that they can form a parallel pair.

Algorithm 2 Algorithm for finding candidate pairs

```
1: function FINDPAIRS(listLang_1, listLang_2, ≥ t_sim)
2:     for all urlLang_1 ∈ listLang_1 do
3:         urlLang_2 = findMostSimiliar(urlLang_1, listLang_2)
4:         if similarityScore(urlLang_1, urlLang_2) ≥ t_sim then
5:             candidatePairs += pair(urlLang_1, urlLang_2)
6:         end if
7:     end for
8:     return candidatePairs
9: end function
```

language of each web page contained in the listOfUrls. URLs containing tokens indicating a language which is not listed as language of interest are discarded. Thus, if any token from the list for language $lang_1$ is found in the URL, the URL is added to $listLang_1$. Note that many times, only one of the URLs in the parallel pair has a language-indicating token. For example, in the pair:

`http://portoalegre.8k.com/ccmq.htm`
`http://portoalegre.8k.com/ccmq_eng.htm`

only the second URL has a language-indicating token. URLs without such a token are added to $listUnk$. This way, the URLs collected from the website which have some indication that they are in the languages of interest or which do not have a language-indicating token, are added to one of three lists: $listLang_1$, $listLang_2$, or $listUnk$. If both $listLang_1$ and $listLang_2$ are not empty, then the algorithm calculates the similarity between each URL in $listLang_1$ and all URLs in $listLang_2$. If one of these lists is empty, then the URLs in the non-empty list is compared against each URL in $listUnk$. Finally, if both $listLang_1$ and $listLang_2$ are empty, the website is discarded as it does not seem to contain parallel texts in the languages of interest. The rationale behind separating the URLs of the website into lists is to avoid doing pairwise comparisons among all URLs. For example, in a website in which the crawler collects 100 URLs, a brute force algorithm would require 4950 pairwise comparisons, while our algorithm would require 2500 comparisons in the worst case.

Algorithm 2 shows how the candidate pairs are formed. Comparing URL similarity (line 3) is done using Levenshtein distance. In order to find candidate pairs, we iterate through one of the lists of URLs and find the URL in the other list with the highest similarity score which also surpasses a given threshold (t_{sim}).

3.2 Classifier

The output of the intra-site crawler are pairs of URLs which are candidates to having parallel texts. The classifier then takes these pairs and decides, based on a set of features, whether they are parallel. The features used in our classifier are:

– *Ratio between the number of words:* For parallel texts in two languages, it is likely that the ratio between their number of words remains stable. This feature is simply calculated by dividing the number of words in the web page in $lang_1$ by the number of words in the web page in $lang_2$.

– *Ratio between file sizes:* Similar to the previous feature, the ratio between file sizes of equivalent texts in two languages should be stable. Thus, we take the file sizes (in bytes) of both pages and divide the size of the web page in $lang_1$ by the size of the web page in $lang_2$.

– *Structural similarity:* To quantify structural similarity we extended Levenshtein edit distance. The original algorithm computes the number of edits (i.e., insertions, substitutions, and deletions) needed to transform one string into another, taking single characters as basic units. For our purposes, we use HTML tags as basic units by simply extracting them in the order they appear on the page. The idea is that parallel web pages will be similar in HTML structure. For example, consider the HTML tags of two web pages as follows:

```
<html>                          <html>
  <head> Page P1 </head>          <head> Page P2</head>
  <body>                          <body>
    <h1> ... </h1>                  <div> ... </div>
    <div> ... </div>                <div> ... </div>
  </body>                         </body>
</html>                         </html>
```

	1	2	3	4	5	6	7	8
P1	<html>	<body>	<h1>	</h1>			</body>	</html>
P2	<html>	<body>					</body>	</html>

In this case, cells 3 and 4 in page P1 (<h1> and </h1>) should be replaced by and in order to make the structure of both files identical. Since there are two operations needed, the edit distance between these pages is 2.

– *Word Overlap:* Even if they are in different languages, parallel web pages are likely to share some words such as proper nouns. This feature computes the proportion of identical words shared by both pages.

To train the decision tree classifier, we manually labeled a sample of the candidate pairs resulting from the intra-site crawler. In our experiments, a small sample of about 100 pairs was enough to obtain good results. Furthermore, once the classifier is trained for a given domain and pair of languages, it can be reused. This shows that the effort demanded from the human expert who labels the pairs is small, which makes our approach usable.

4. EVALUATION

In this section, we report the experiments carried out to evaluate our proposed approach. We start by detailing the methodology used and then present our results.

4.1 Experimental Setup

The domain of the websites used in the experiments was *universities* and English-French was the language-pair of choice. In order to run our algorithms we required URLs of websites in this domain. Thus we used two hub pages for Canadian and French universities (`http://uwaterloo.ca/canu/` and `http://www.french-at-a-touch.com/Schools_and_Universities/french_universities.htm` respectively). In total, there were URLs for 361 websites containing over 284K pages.

These URLs were fed into PPLocator and the baseline method. Because of copyrights, we are not allowed to distribute the corpus we obtained with our method. However, as done by Resnik [Resnik 1998], we can publish a list of pairs of URLs containing parallel texts which were discovered by PPLocator. This list can be downloaded from `http://www.inf.ufrgs.br/~mmvieira/PPLocator_output_pairs`.

PPLocator was implemented according to the description in Section 3. Only pages in the top 3 levels ($n = 3$) were analyzed. Based on some empirical observations, the threshold for URL similarity (t_{sim}) was set to 0.85. WEKA [Hall et al. 2009] was used to build the decision tree classifier using the J48 algorithm. The intra-site crawler returned 1109 candidate pairs. A sample containing 10% of those was randomly selected to train the classifier. The sample contained 110 pairs of which 25 were positive (i.e. truly parallel pairs) and 85 were negative. Note that the pairs of URLs used to train the classifier were excluded from the evaluation set. Thus the evaluation was made on the remaining 90% of pairs resulting from the intra-site crawler.

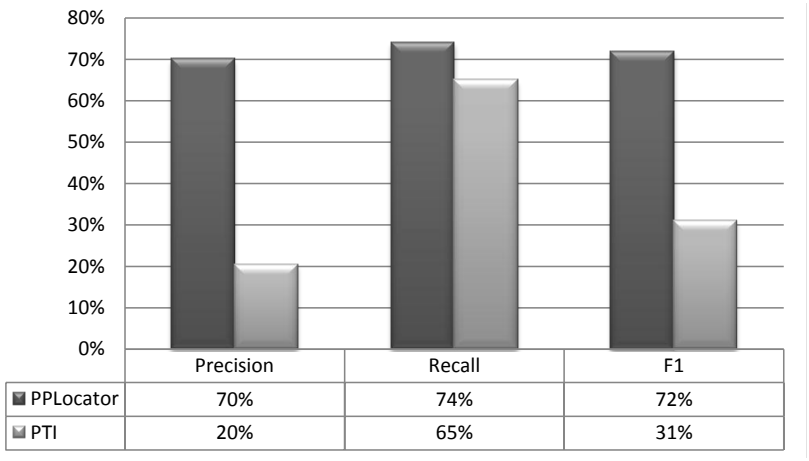
The method implemented as baseline was PTI [Chen et al. 2004], which we described in Section 2. PTI requires a language identifier – we used TextCat [Cavnar and Trenkle 1994], which has been extensively used in the literature. It also needs a bilingual term list. We built a list combining four bilingual dictionaries available freely from `http://www.dicts.info/uddl.php`. There were a number of reasons for choosing PTI as our baseline: (i) it achieved good results in the experiments reported [Chen et al. 2004]; (ii) its description in the paper was clear enough to allow its implementation; (iii) it was not designed to accept only specific inputs (unlike [Fry 2005; Tsvetkov and Wintner 2010] which were designed for news articles, or [Resnik 1998; 1999; Chen and Nie 2000; Jesús Tomás and Casacuberta 2005] which require as input URLs returned by a search engine in response to a query made on the anchor text); and finally (iv) it uses language identifiers and bilingual wordlists which are not employed by PPLocator and thus allows for an interesting comparison.

The output from both methods (i.e. pairs of URLs considered parallel) was examined by a human assessor. The assessor marked each of the 814 pairs as *correct* if both pages had the same content in different languages, or *incorrect*, otherwise. Partial matches (i.e., when a page contains a sub or superset of the other) were considered incorrect. Based on this assessment, we computed precision (Eq. 1), recall (Eq. 2), and F-measure (Eq. 3). Note that it was impractical to manually analyze every page from all 361 websites in order to establish the total number of parallel pairs that the methods should find. Thus, we only examined the output for each method and our reported recall is actually the *relative recall* between PTI and PPLocator. Since our aim was to compare the two, the relative recall is a valid measure. The union of the correct pairs found by the methods contains 211 pairs.

$$Precision(P) = \frac{\#ParallelPairs \bigcap \#IdentifiedPairs}{\#IdentifiedPairs} \tag{1}$$

$$Recall(R) = \frac{\#ParallelPairs \bigcap \#IdentifiedPairs}{\#ParallelPairs} \tag{2}$$

$$F - measure = \frac{2 \times P \times R}{P + R} \tag{3}$$



	Precision	Recall	F1
PPLocator	70%	74%	72%
PTI	20%	65%	31%

Fig. 4. Precision, Recall, and F-Measure for PPLocator and PTI

Another useful statistic is the ratio between the number of correctly identified pairs and the number of pages downloaded by the method. The idea is to penalize methods which unnecessarily download web pages that are not parallel. The higher the ratio, the better, as this means that fewer resources were used.

4.2 Results

Comparing against the baseline: The results of the comparison between PPLocator and PTI are shown in Figure 4. PPLocator achieved better results in terms of precision, recall, and F-measure. PTI returned 673 pairs, of those only 137 were indeed parallel pages. PPLocator identified 223 pairs, and 156 of those were correct. These results attest for the effectiveness of our method even though it uses fewer resources. The main reasons for PTI's worse performance are: (i) its filename comparison module applies very strict criteria, i.e., if the URL of an English page has *help* and the URL of its French counterpart has *aide*, this pair would be discarded as this was not one of the expected filenames; (ii) it relies on a language identifier, and thus is susceptible to the errors made by these tools; (iii) its matching criteria requires a similarity score greater than 0.9, which is too high for the pages in our dataset. PTI was better than PPLocator in cases where the language-indicating substring in the URL is missing or misleading. For example, our approach has identified the pair
`http://www.univ-nancy2.fr/formations/f_continue/fc_nu.html?depuis_id=1246&lang=EN`
`http://www.univ-nancy2.fr/formations/f_continue/fc_nu.html?depuis_id=1246&lang=FR`
as being parallel. However, the texts in both pages are in French, despite the first URL having the token "lang=EN".

The results obtained by PTI in this experiment are considerably lower than the results reported in [Chen et al. 2004]. We believe this happened because using bilingual wordlists to compare the contents of the pages leads to many false positives when the pages are on similar subjects — that is the case for our dataset, in which all pages were from university websites. PTI's authors experimented with news articles, which probably were not similar among themselves. In addition, the experiments reported in [Chen et al. 2004] were made over a very small dataset containing only 427 documents from a single website. Probably, results obtained from a single website do not generalize to others.

As for the ratio between the number of parallel pages identified and the number of pages downloaded by each method, we observed that PPLocator is much more efficient. Both approaches downloaded all 20,713 pages in the first two levels of the websites. The big difference is on the third level, in which PTI downloaded 263,982 pages, while PPLocator only downloaded the 1,376 which had some

Table I. Number of parallel pairs identified by each method versus the number of pages downloaded

	Correct Pairs	Downloaded Pages	Ratio
PPLocator	156	22089	0.7062%
PTI	137	284695	0.0481%

Table II. Contributions of the Different Features

	Precision	Recall	F1
PPLocator	84%	87%	86%
PPLocator-ratio_num_words	81%	81%	81%
PPLocator-ratio_file_size	81%	75%	78%
PPLocator-struct_similarity	76%	81%	78%
PPLocator-word_overlap	75%	78%	76%

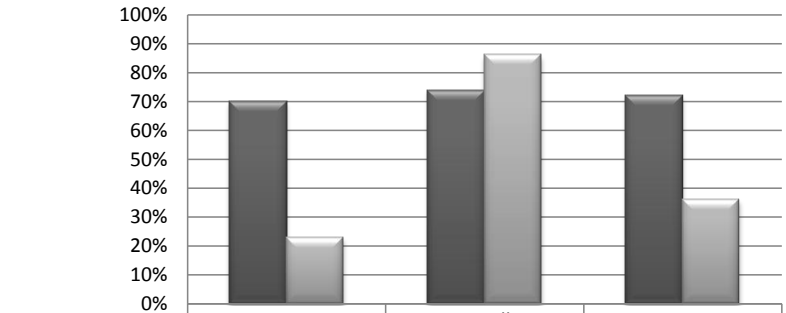
indication that they could be part of a candidate pair. The results are displayed in Table I.

Contribution of each Feature: To analyze the contribution of each feature, we ran our classifier multiple times, and each time we removed one of the features. Table II show the results for 10-fold cross validation. Note that these results are different from the ones reported in Figure 4, since in those we used different training and test sets. In terms of F-measure and precision, the feature with the highest contribution was the word overlap. This feature, however, can be misleading since a high word overlap may indicate that both pages are in the same language. Because we do not use language identifiers, a high score on this feature could indicate a false positive. However, in terms of recall, the most important feature was the ratio between file sizes. But, most importantly, the numbers show that all features contribute to the overall performance.

We were also interested in finding out how much our method could be improved if we used an automatic translator. Thus, we translated the French pages into English and computed two additional features: word overlap after translation and the stem overlap after translation (using the Porter Stemmer). The results of this test have shown no improvement in the quality of the classification. We believe the reasons for this are: (i) the automatic translator retains only one translation for each word. So, if the translated word does not match the original word in English word overlap would not increase; and (ii) word overlap without translation is able to identify a good portion of identical words (due to proper nouns which are not translatable).

Contribution of Each Module: In order to have an idea about the performance of each of the two components of PPLocator, we analyzed the results of the intra-site crawler separately. We examined the output of the intra-site crawler and observed out of the 1109 candidate pairs, 252 were positive, yielding a precision of 22%. The intra-site crawler aims for high recall, keeping as many positive pairs as possible. To validate whether it was satisfying this goal, we looked at a random sample 100 websites which were discarded by the intra-site crawler. Only four sites had parallel pages (i.e., were false negatives). In three of those, the parallel pages were further than the 3 levels considered by the intra-site crawler and, in one of them, numbers were used to designate languages. For example, in `http://www.univ-tlse3.fr/09895179/0/fiche___pagelibre/&RH=rub08&RF=1237282723130` and `http://www.univ-tlse3.fr/09895179/1/fiche___pagelibre/&RH=rub08&RF=1237282723130` the numbers 0 and 1 indicate that the pages are in French and English, respectively.

Comparing the results of PPLocator with and without the classifier, we observe that precision improves significantly (from 23 to 70%). This represents a proportional gain of over 200%. Such improvement compensates the small loss in recall incurred by the classifier, since some parallel pairs end up being discarded. Figure 5 shows the results for this comparison.



	Precision	Recall	F1
PPLocator	70%	74%	72%
Intra-site Crawler	23%	86%	36%

Fig. 5. Precision, Recall, and F-Measure for the intra-site crawler on its own and PPLocator

The biggest cause behind *false positives*, in both the intra-site crawler and the classifier, was a misleading language-indicating substring. Pages which had some parallel content but were not completely parallel are also a cause for false positives. *False negatives* were mainly cases in which both English and French versions were in the same file, and then Javascript controls which version is rendered by the browser.

Limitations: Because PPLocator relies on the text of the URL, it will fail in cases where the website does not follow a naming convention (e.g. some pages in a language have a language-indicating substring and others do not) or the language-indicating substring is misleading (e.g. the URL has `lang=FR` but the text of the page is not in French). In addition, because we crawl only the first n levels, parallel pages are deeper in the hierarchy of the website will not be identified. Finally, cases in which a page has a subset or a superset of the contents of the other page also pose a problem to PPLocator as it may incur in a false positive.

5. CONCLUSION

This article presented PPLocator, a method for locating parallel web pages. The proposed approach takes into consideration evidences from the URLs, structure, and the content of the pages. It is composed of two modules. First, an intra-site crawler examines the website and extracts pairs of pages which, based on their URLs, are likely parallel. These pairs are then submitted to the second module, a decision tree classifier. The classifier further refines the candidate pairs based on structure and content. PPLocator aims at being efficient, so it does not perform exhaustive pairwise comparisons with all pages of the website, nor does it download the entire website. Furthermore, it does not rely on external resources such as bilingual dictionaries or automatic translators.

An experimental evaluation with over 284K pages from 361 websites attest for the viability of PPLocator. The results are superior compared to baseline method which uses more expensive resources.

Future work will include the aggregation of an *inter*-site crawler which will feed the intra-site crawler with pages from the selected domain (i.e., topic of interest). The domain plays an important role in a parallel corpus since the translation scheme extracted for a specific domain cannot be reused in other domains. In addition, we would like to explore other features to include in the classifier (such as DOM tree similarity as in [Shi et al. 2006]) and other metrics to compute URL similarity. Finally, we plan to asses the validity of the corpora assembled corpora with PPLocator to train machine translation systems and corpus-based techniques for CLIR.

REFERENCES

Caseli, H. M., Nunes, M. D., and Forcada, M. L. Automatic induction of bilingual resources from aligned parallel corpora: application to shallow-transfer machine translation. *Machine Translation* 20 (4): 227–245, Dec., 2006.

Cavnar, W. B. and Trenkle, J. M. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*. Las Vegas, Nevada, U.S.A., pp. 161–175, 1994.

Chen, J., Chau, R., and Yeh, C.-H. Discovering parallel text from the world wide web. In *Proceedings of the second workshop on Australasian information security, Data Mining and Web Intelligence, and Software Internationalisation - Volume 32*. Dunedin, New Zealand, pp. 157–161, 2004.

Chen, J. and Nie, J.-Y. Parallel web text mining for cross-language ir. In *Proceedings of Recherche d'Informations Assistee par Ordinateur 2000*. Paris, France, pp. 62–77, 2000.

Fry, J. Assembling a parallel corpus from rss news feeds. In *Proceedings of the Workshop on Example-Based Machine Translation, MT Summit X*. Phuket, Thailand, 2005.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. The WEKA data mining software: an update. *SIGKDD Explor. Newsl.* 11 (1): 10–18, Nov., 2009.

Jesús Tomás, Enrique Sánchez-Villamil, J. L. and Casacuberta, F. Webmining: An unsupervised parallel corpora web retrieval system. *Proceedings from the Corpus Linguistics Conference Series* 1 (1), 2005.

Kraaij, W., Nie, J.-Y., and Simard, M. Embedding web-based statistical translation models in Cross-Language Information Retrieval. *Computational Linguistics* 29 (3): 381–419, 2003.

Li, B., Liu, J., and Shi, W. Web-based parallel corpora for statistical machine translation. In *Proceedings of the Sixth International Conference on Machine Learning and Applications*. Cincinnati, Ohio, USA, pp. 444–449, 2007.

Ma, X. and Liberman, M. Y. Bits: A method for bilingual text search over the web. In *Proceedings of the Machine Translation Summit VII*. Singapore, 1999.

Melamed, I. Models of translational equivalence among words. *Computational Linguistics* 26 (2): 221–49, 2000.

Resnik, P. Parallel strands: A preliminary investigation into mining the web for bilingual text. In *Third Conference of the Association for Machine Translation in the Americas*. London, UK, pp. 72–82, 1998.

Resnik, P. Mining the web for bilingual text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*. Stroudsburg, PA, USA, pp. 527–534, 1999.

Resnik, P., Olsen, M., and Diab, M. The bible as a parallel corpus: Annotating the 'Book of 2000 Tongues'. *Computers and the Humanities* 33 (1): 129–153, 1999.

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

Shi, L., Niu, C., Zhou, M., and Gao, J. A dom tree alignment model for mining parallel data from the web. In *Proceedings of the 21st International Conference on Computational Linguistics*. Sydney, Australia, pp. 489–496, 2006.

Tsvetkov, Y. and Wintner, S. Automatic acquisition of parallel corpora from websites with dynamic content. In *Proceedings of The seventh international conference on Language Resources and Evaluation (LREC-2010)*. Valletta, Malta, 2010.

Zhang, Y., Wu, K., Gao, J., and Vines, P. Automatic acquisition of chinese-english parallel corpus from the web. In *Proceedings of 28th European Conference on Information Retrieval*. London, UK, pp. 420–431, 2006.