Automatic Parallel Corpora and Bilingual Terminology extraction from Parallel WebSites

José João Almeida[1], **Alberto Simões**[2]

[1] Departamento de Informática, Universidade do Minho, Portugal
[2] Escola Superior de Estudos Industriais e de Gestão, Instituto Politécnico do Porto, Portugal
jj@di.uminho.pt, alberto.simoes@eu.ipp.pt

Abstract

In our days, the notion, the importance and the significance of parallel corpora is so big that needs no special introduction. Unfortunately, public available parallel corpora is somewhat limited in range. There are big corpora about politics or legislation, about medicine and other specific areas, but we miss corpora for other different areas. Currently there is a huge investment on using the Web as a corpus.
This article uncovers GWB, a tool that aims automatic construction of parallel corpora from the web. We defend that it is possible to build high quality terminological corpora in an automatic fashion, just by specifying a sensible Internet domain and using an appropriate set of seed keywords. GWB is a web-spider that works in conjunction with a set of other Open-Source tools, defining a pipeline that includes the documents retrieval from the web, alignment at sentence level and its quality analysis, bilingual dictionaries and terminology extraction and construction of off-line dictionaries.

1. Introduction

As it is already well known, parallel corpora is relevant for different natural language studies, as translation studies, machine translation and other important tasks.

One of the many uses for parallel corpora is the extraction of bilingual resources, like bilingual dictionaries or bilingual terminology. Unfortunately not all parallel corpora are suitable for terminology extraction. In fact, first developed parallel corpora were mainly devoted to literary translation studies and did not include relevant quantities of terminology. Recently corpora started to include other types of texts, like juridical or law texts. Examples are EuroParl (Koehn, 2005) or JRC-Acquis (Steinberger et al., 2006).

If we focus on languages like the Portuguese, we notice that other than these big corpora there are not much more choices. There are a few technical corpora compiled in the OPUS project (Tiedemann and Nygaard, 2004) like OpenOffice or Apache documentation, or literary corpus like (Frankenberg-Garcia and Santos, 2003).

These corpora include some terminology and are relevant for terminology analysis and extraction. But they have problems. EuroParl is mainly oral, that results in a bad quality alignment. JRC-Acquis include some more interesting terminology and has good alignment quality. But the range of terminological terms found is quite limited. JRC-Acquis includes the basic norms for every country joining the European Union. These norms focuses mainly on social and economic behavior laws. Finally, the technical corpora from OPUS are mainly in the computer science area. There is also a medicine corpus and a subtitles corpus.

Therefore, methods to create automatically closed-domain corpora are relevant, especially if one can construct it fast and easily.

With that in mind we present a tool, GWB(GetWebBitext), to lookup for parallel documents in the web and create parallel corpora, from a closed-domain area of knowledge and rich on terminology. As main design principle, all this process should be completely automatic.

Our system is based on a set of seed keywords (normally, a couple of terminology term examples) and one or more Internet domains where the tool will search for the texts that will comprise the parallel corpus.

While we present the full pipeline of GWB, this article will focus essentially the parallel page candidates detection, their download and analysis. The remaining part of the pipeline uses a set of tools that where chosen for being open-source and freely available, but can be easily swapped by other similar tools.

1.1. GWB Design Principles

GWB was develop with the following design principles:

Control over the text sources: the user provides the set of Internet domains in which the search process will be performed;

Full pipeline for terminology extraction: GWB is not designed just to download the text that comprises the parallel corpus, but includes a complete pipeline of corpora processing that ends with the automatic extraction of bilingual resources;

Modularity: it is important to have a full pipeline of tools that work correctly as an unique tool. But is is also important that all the tools of the pipeline can be used as a standalone application[1]. Thus, it should be possible to make GWB perform just part of the pipeline, accordingly with the user needs. Also, some of the GWB modules depend on other specific languages, or use a specific tools. Being modular, GWB lets the user substitute any of the modules by any other tool.

Reuse: GWB does not try to reinvent the wheel, but instead, use already available Open-Source tools, like Open-Corpus-Workbench, Easy-Align, NATools, *Yahoo!* API or

[1] Following the Unix tradition: each command should do only one thing but do it well (in our case, we have a lot of space for improvement)

Proceedings of the 3rd Workshop on Building and Using Comparable Corpora, LREC 2010, pages 50–55
Malta, 22 May 2010

StarDict[2] (and others).

1.2. Other Tools

The idea of using the Web as a Corpus is not new (Bernardini et al., 2006) and there are a couple of well known applications for automatic corpora construction from the Web.

1.2.1. BootCat and WebBootCat

Probably, the most well known application for corpora construction is BootCat (Baroni and Bernardini, 2004), also available as a web application (Baroni et al., 2006).
BootCat was originally designed for automatic building of disposable corpora, using a set of seed terms. Web pages retrieved did not had to contain all the terms specified, but at least some combination.
BootCat is not just a retrieval tool. It includes a rich set of tools used for terminology extraction and statistical manipulation of terms, n-grams and others.
In order to connect all the small available tools in one single task some Unix expertise may be useful. This has some advantages and drawbacks. In one hand it makes the system flexible, in the other hand, it makes the system hard to use for less knowledge users.
GWB deals with a different problem (parallel corpora) but it try to reuse part of the BootCat principles, but adding an extra layer: a "work-flow" level command — a single command that can hide some of the typical tools combination. This tool defines a set of rules that specify how to run a pipeline of tools until the intended results are achieved.

1.2.2. STRAND

Another tool for Parallel Corpora retrieval and construction from the web is STRAND (Resnik and Smith, 2003). STRAND approach is completely different from GWB or BootCat. STRAND does not search for specific terms. It just searches for parallel pages from the Web (or a specific domain). The procedure is simple: after retrieval, each page is checked for one of the following two properties:

- an entry page, with links to different language websites (thus, links with language names);
- check pages that link for the respective translated page.

GWB parallel page detection system is faster as it does not need to parse the HTML files neither to download all the document from the web. Also, GWB detects non-HTML documents that would not be detected using the above mentioned heuristics.

2. Architecture

GWB main algorithm might be defined as the following steps:

1. from a set of user-provided keywords K, a pair of languages, L_1 and L_2 and a set of valid Internet domains D, retrieve the first N document URLs that contain all the keywords K and is cataloged by the search engine (for instance, *Yahoo!*) as being in language L_1.

$$Docs_{L_1} = yahoo(K, site : D, lang : L_1)$$

2. for each URL in $Docs_{L_1}$ try to guess the corresponding URL with the document in language L_2 (this process is similar to the described by Mohler and Mihalcea (2008)):

$$Docs_{L_2} = parguess(Docs_{L_1}, L_2)$$

3. retrieve all documents pointed by the obtained URL (if they exist) and convert them to a textual format (PML):

$$Bitexts = retrieve(Docs_{L_1}, Docs_{L_2})$$

4. build a parallel corpora PC aligning at the sentence level the retrieved documents. Note that this is done for each document pair.

$$PC = align(Bitexts)$$

5. filter the parallel corpora discarding translation units or documents with low alignment quality:

$$PC = filter(PC)$$

6. extract probabilistic translation dictionaries from the aligned corpora:

$$PTD = extractPTD(PC)$$

7. extract bilingual terminology using the probabilistic dictionaries and a set of alignment patterns:

$$Terms = terms(PC, PTD, Patterns)$$

8. create a StarDict dictionary for off-line usage based on the bilingual terminology and dictionaries extracted:

$$StarDict = mkSD(PTD, Terms)$$

The GWB modules work in pipeline as shown in figure 1.

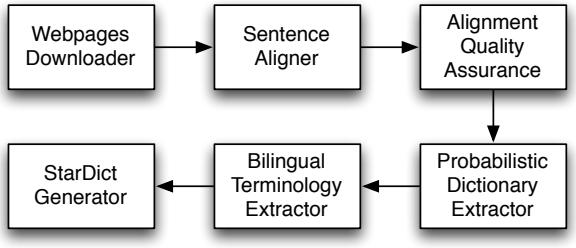

Figure 1: GWB Architecture

The next sections describe each one of these modules in greater detail.

[2] Available from `http://stardict.sourceforge.net/`

2.1. Web-pages downloader

The downloader module relies on an Applications Programmer Interface (API) to a web search engine, like *Google* or *Yahoo!*. Given the current limitations imposed by *Google* for its API, GWB uses *Yahoo!*.

2.1.1. Find Documents in language 1

GWB receives a pair of language identifiers, a set of keywords (terminology examples) in the first language and a set of Internet domains to crawl. GWB lets the user to specify a closed set of keywords or to let the application generate more seeds based on morphological information. Therefore, any keyword followed by an asterisk will be lemmatized and its lemma will be used for generation. For the Portuguese and English language we have been using jSpell morphological analyzer and respective dictionaries (Simões and Almeida, 2001).

GWB will search for pages with all those keywords (generated words will be OR'ed automatically) under the specified Internet domains.

By default, GWB searches for PDF and HTML pages. Other document types can be defined, being just a matter of writing a plug-in to convert that document type to plain text.

The number of pages retrieved can be defined (as a command line option or by configuration). By default, GWB retrieves 100 pages.

2.1.2. Calculate URL candidates for language 2

The next step is to find the translation pages. The URL candidates are calculated using a technique described in (Almeida et al., 2002) that relies on systematic web-pages organization (as *good* web-masters usually do): it is natural that a page in Portuguese includes a substring in the URL that specifies that language, like `Portuguese`, `pt` or `port`[3].

Therefore, one can rewrite that substring in the URL for a set of possible equivalences in the target language and check if any of the pages exist. There is a list of common keywords for each language which makes it easy to any user to use the tool without further configuration. In any case, it is possible for the user to add new languages or new language keywords.

Note that some caution should be taken during this substitution, as the domain portion of the URL should not be adapted.

Each document pair successfully retrieved, becomes a bitext candidate, is converted to plain text and sent to the sentence aligner.

2.2. Sentence aligner

Each plain text pair needs to be processed before alignment. It is necessary to detect sentence boundaries (segmentation) and detect word boundaries (tokenization). In our experiments we are using `Lingua::PT::PLNbase`, a Perl modules written for segmentation and tokenization of Portuguese. While the module was written with Portuguese in mind it supports some constructs from English, French and Spanish. In any case, it is easy to plug-in any other tool to segment and tokenize the text.

The segmented and tokenized texts are stored in a specific XML-based format, named PML, where just texts, paragraphs and sentences are annotated. Words are separated from each other with a blank.

These PML files are then sentence-aligned using easy-align. This aligner is part of Corpus Workbench (Christ et al., 1999). It uses the usual sentence size information to perform the alignment, but it also supports external bilingual dictionaries to help the synchronization. As easy-align relies on CWB, the PML files are firstly encoded as two separate monolingual corpora and then aligned.

The alignment result is then exported in TMX (Translation Memory Exchange) format files. While we are aware that this is not the most usual format for parallel corpora we find it more usable than TEI (Text-Encoded Initiative) or XCES (XML Corpus Encoding Standard).

Note that at this moment we still have several different TMX files (one for each retrieved document).

2.3. Alignment quality assurance

Each TMX file is analyzed in terms of quality. This can make the full document to be rejected, or some specific translation units to be deleted.

For translation unit quality analysis GWB uses the following heuristics[4]:

- sentence length comparison: while the main algorithm for easy-align is based on sentence length, some times the algorithm results include alignments with big sentence length differences. More precisely, the system will discard any translation unit with more than 20 characters for both languages, and with one language length greater than two times the length of the other.
- non-words preservation: numbers are extracted and compared. While the sequence is not required to be the same, they must all be preserved.
- punctuation analysis: while it is natural that punctuation changes (sentences are split, some languages use more commas than other and so on), some specific punctuation should be preserved.
- word translation probabilities: GWB is also able to evaluate translation units quality using bilingual dictionaries (or probabilistic translation dictionaries). As this subject is not the main topic for this article details will not be presented.

Full translation memories will be discarded if:

- more than half of the translation units were discarded by the previous heuristics;
- the majority (80%) of the alignments are not one-to-one sentence alignments.

[3] We know not all web sites follow these convention. Also, there is the possibility of false positives. But the pipeline of tools take the needed care to check languages and alignment possibilities.

[4] All these values can be user configured. This is relevant as different language pairs will have different ratios.

The TMX files that are not discarded are then concatenated together in a single file using the `XML::TMX` Perl module. This TMX file is the final corpus that will be processed by the next modules to produce bilingual resources.

2.4. Probabilistic dictionary extractor

The resulting parallel corpus is processed by NATools toolkit (Simões and Almeida, 2007) for the extraction of probabilistic translation dictionaries (Simões and Almeida, 2003). As the NATools extractor handles TMX files directly, this step is nothing more than the NATools corpus creation application and the final treatment of probabilistic translation dictionaries.

2.5. Bilingual terminology extractor

The same NATools toolkit includes an application for parallel terminology extraction (Simões and Almeida, 2008). This extraction is guided by a set of translation patterns, where the user can specify what kind of constructions he/she is searching. Therefore, this method can be used to extract terminology but also to extract specific linguistic constructions that are under analysis.

2.6. StarDict generator

It is our conviction that results extracted automatically should be made available to the end-user in a legible format. While to extract resources and have them available in textual format is useful when statistics are to be calculated, or the resources are to be integrated in other tools, for translation or linguistic studies it is easier to consult the resources as if they were a dictionary. With this in mind, GWB final module grabs the probabilistic translation dictionaries and the terminology extracted in the previous steps and constructs a StarDict dictionary for off-line viewing and querying.

3. Experiments

In this section we will discuss some experiments in order to give a better picture of what we can do with this tools (how we are using it) and show some simple metrics. The presented case-studies are:

- extract a translation memory from a small-size Web-site (a call-for-papers web-site);
- build a narrow domain parallel corpus (about alcoholic beverages) following the complete pipeline.

3.1. Small parallel corpus for a simple terminologically rich Web-Page

The first experiment corresponds to the following situation:

- we spotted a well written call for papers, with a good introduction to the area and a large set of *central topics*[5];
- we are in the presence of a small-size Web-site, with suitable translation quality;
- the web-site is available both in Portuguese and Spanish.
- we do not expect to obtain a real-size parallel corpus, but just a very small translation memory file (TMX) with a specific list of topics.

While this is a small text that will not be suitable for terminology extraction, our main purpose here is the extraction of a small translation memory that can be later used together with other to translate or create a bigger parallel corpus.
To create the translation memory we can use GWB as follows:

```
getwebbitext
 -s "ciawi-conf.org"   #  site-sources
 -l pt:es              # language pair
 -until tmx    # stop when TMX is done
 trabalhos
```

The keyword used — **trabalhos** (*call for papers* = *chamada de trabalhos*) — is present in the text and is valid in just one of the languages (Portuguese).
After near 10 seconds of network activity, we obtained 7 bitexts. GWB found 8 documents matching "*trabalhos*", but only 7 parallel documents. Follows an example of a retrieved URL (Portuguese) and the respective rewritten URL for the target language (Spanish).

`www.ciawi-conf.org/pt/cfp.asp`

⇓

`www.ciawi-conf.org/es/cfp.asp`

After bitexts extraction the alignment process takes place, aligning the documents and building a TMX file, with about 305 translation units (1 987 words for the source language and 2 067 for the target language).)
This experiment took less then 20s. In this case we decided not to generate dictionaries or terminology as the corpus size is too small.
In any case, the TMX file is still useful and can be used directly in common computer aided translation (CAT) tools like SDL-Trados, POedit or Omega-T.
The exercise is not complex – all the bitext candidate pairs passed in the quality control (100%) and the TMX had 4 alignment errors (98.6%)

3.2. Terminology on alcoholic beverages

In this second experiment we built a parallel corpus for a specific narrow domain (wine, spirit drinks and similar).
To start using GWB the user needs an Internet domain where there is texts on the chosen area. European laws include sections related to that subject, so we used `http://eur-lex.europa.eu/` as the source web-site.
In our experiments we concluded that EurLex web-site is both, one of the biggest multilingual quality sources and a good source for terminologically rich documents. That is why EurLex is the default source for GWB.
In order to select relevant information we used some terms in domain we are searching. In this case they keywords were *cerveja* (beer) and *vodka* (in Portuguese we use the same word as in English):

[5]For example, `http://www.ciawi-conf.org/`

```
getwebbitext
   -s eur-lex.europa.eu
   -l pt:en
   cerveja  vodka
```

By default GWB will get the first 100 documents[6] in the source language. GWB took 3m22s[7] to execute this task, and we obtained 37 MB of bitext candidates (12 bitexts in HTML format and 22 in PDF)[8].

3.2.1. Bitext to TMX

We made two align experiments, one excluding the PDF files and another one including all retrieved files.
Excluding the PDF documents the final TMX had 32 941 translation units (about 9MB). Including PDF documents (several of the PDF documents were rejected given format or alignment problems) the final TMX had 81 844 translation units (about 22MB of text — about 1 300 000 tokens per language).

3.2.2. Probabilistic translation dictionary extraction

Follows some (hand-selected) example entries from the extracted probabilistic translation dictionary. Each entry includes the term, its number of occurrences in the source language corpus and a set of probable translations.

$$cervejas\ (29)\ \begin{cases} beer & \rightarrow & 98\% \\ actual & \rightarrow & 2\% \end{cases}$$

$$cerveja\ (53)\ \begin{cases} beer & \rightarrow & 62\% \\ brewing & \rightarrow & 24\% \\ distilling & \rightarrow & 3\% \\ coloured & \rightarrow & 1\% \end{cases}$$

$$vodka\ (139)\ \begin{cases} vodka & \rightarrow & 94\% \\ flavoured & \rightarrow & 2\% \\ vodkas & \rightarrow & 1\% \end{cases}$$

$$licor\ (73)\ \begin{cases} liqueur & \rightarrow & 95\% \\ licor & \rightarrow & 2\% \\ liqueurs & \rightarrow & 1\% \end{cases}$$

$$rum\ (99)\ \begin{cases} rum & \rightarrow & 96\% \\ produced & \rightarrow & 1\% \\ word & \rightarrow & 1\% \\ solbaerrom & \rightarrow & 1\% \end{cases}$$

$$vinho\ (271)\ \begin{cases} wine & \rightarrow & 81\% \\ vinho & \rightarrow & 7\% \\ aromatised & \rightarrow & 2\% \\ wines & \rightarrow & 2\% \\ wine-based & \rightarrow & 1\% \end{cases}$$

$$vinagres\ (38)\ \{ vinegar \rightarrow 96\%$$

$$malte\ (208)\ \{ malt \rightarrow 95\%$$

$$aguardente\ (226)\ \begin{cases} spirit & \rightarrow & 70\% \\ aguardente & \rightarrow & 14\% \\ spirits & \rightarrow & 13\% \\ diluted & \rightarrow & 1\% \\ distilled & \rightarrow & 1\% \end{cases}$$

$$porto\ (42)\ \begin{cases} porto & \rightarrow & 89\% \\ port & \rightarrow & 6\% \\ reserva & \rightarrow & 3\% \\ doce & \rightarrow & 2\% \end{cases}$$

The full process took near 30 minutes but was completely automatic. In the end we obtained:

- a 81K translation unit TMX file (22MB);
- a pair of probabilistic translation dictionaries;
- a StarDict dictionary (check figure 2 for an example);

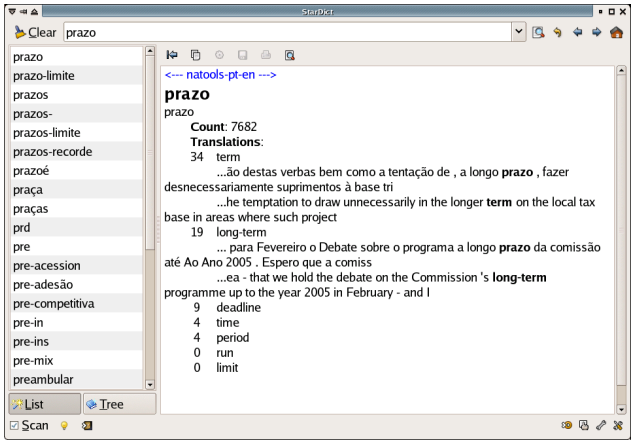

Figure 2: StarDict screenshot for an automatically built dictionary.

3.3. Some practical tips

In order to make useful things with GWB, some practical tips may help:

1. whenever possible choose sources that you know.
2. parallel corpora do not need to be built all at once. You can build small translation memories with good translation quality and join them later to create bigger resources.
3. when selecting the seed terms, try to use words that just belong to one of the languages (if possible the most less represented one).
4. use websites where you suspect there are annexed/linked documents like legislative texts;
5. chose a set of domain specific seed terms (eg. “ácido sulfúrico” and “nitrato de prata”);
6. always save the set of seed terms and domain sources, in order to be easy to continue the parallel corpora extraction task in the future.
7. sometimes is difficult to find good site with bitexts about some domains. Techniques like the ones presented in (Resnik and Smith, 2003) prove to be useful in finding sites of bitext sources.

[6] There is a command line option to redefine this value.
[7] real 3m21.913s
[8] It is possible to select the type of documents to be retrieved.

4. Conclusions

Current web-life makes it easy to find interesting pages, rich in terminology. Main problem would be how to retrieve those pages and create some kind of lexicon from it. We defend that GWB is a suitable tool to attack this kind of web-sites and construct bilingual resources automatically.

Our main objective is not quantity but quality. That explains why GWB requires a specific domain for the documents to be searched and why it also requires a full set of terms (and not just a subset).

GWB is mainly designed for small knowledge areas. A well defined set of seed terms is the key for the quality of the obtained corpora.

Main future issue for GWB is the creation of a distribution package, and put the tool available in CPAN for easy dissemination and installation.

5. Acknowledgments

This work was partially funded by project *Português em paralelo com seis línguas (Português, Español, Russian, Français, Italiano, Deutsch, English)* grant PTDC/CLE-LLI/108948/2008 from *Fundação para a Ciência e a Tecnologia*.

6. References

José João Almeida, Alberto Manuel Simões, and José Alves Castro. 2002. Grabbing parallel corpora from the web. *Procesamiento del Lenguaje Natural*, 29:13–20, September.

Marco Baroni and Silvia Bernardini. 2004. BootCaT: Bootstrapping corpora and terms from the web. In *In Proceedings of LREC 2004*, pages 1313–1316.

Marco Baroni, Adam Kilgarriff, Jan Pomikálek, and Pavel Rychlý. 2006. WebBootCaT: instant domain-specific corpora to support human translators. In *Proceedings of EAMT 2006 - 11th Annual Conference of the European Association for Machine Translation*, pages 247–252, Oslo. The Norwegian National LOGON Consortium and The Deparments of Computer Science and Linguistics and Nordic Studies at Oslo University.

Silvia Bernardini, Marco Baroni, and Stefan Evert. 2006. A wacky introduction. In Marco Baroni and Silvia Bernardini, editors, *WaCky! Working Papers on the Web as Corpus*, pages 9–40. Gedit Edizioni, September.

Oliver Christ, Bruno M. Schulze, Anja Hofmann, and Esther König, 1999. *The IMS Corpus Workbench: Corpus Query Processor (CQP): User's Manual*. Institute for Natural Language Processing, University of Stuttgart, March.

Ana Frankenberg-Garcia and Diana Santos. 2003. Introducing COMPARA, the portuguese-english parallel translation corpus. In Silvia Bernardini Federico Zanettin and Dominic Stewart, editors, *Corpora in Translation Education*, pages 71–87. Manchester: St. Jerome Publishing.

Philipp Koehn. 2005. EuroParl: A parallel corpus for statistical machine translation. In *Proceedings of MT-Summit*, pages 79–86.

Michael Mohler and Rada Mihalcea. 2008. Babylon parallel text builder: Gathering parallel texts for low-density languages. In Nicoletta Calzolari et at., editor, *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco, may. European Language Resources Association (ELRA). http://www.lrec-conf.org/proceedings/lrec2008/.

Philip Resnik and Noah A. Smith. 2003. The web as a parallel corpus. *Computational Linguistics*, 29:349–380.

Alberto Manuel Simões and José João Almeida. 2001. `jspell.pm` — um módulo de análise morfológica para uso em processamento de linguagem natural. In *Actas da Associação Portuguesa de Linguística*, pages 485–495.

Alberto M. Simões and J. João Almeida. 2003. NATools – a statistical word aligner workbench. *Procesamiento del Lenguaje Natural*, 31:217–224, September.

Alberto Simões and José João Almeida. 2007. Parallel corpora based translation resources extraction. *Procesamiento del Lenguaje Natural*, 39:265–272, September.

Alberto Simões and José João Almeida. 2008. Bilingual terminology extraction based on translation patterns. *Procesamiento del Lenguaje Natural*, 41:281–288, September.

Ralf Steinberger, Bruno Pouliquen, Anna Widiger, Camelia Ignat, Tomaž Erjavec, Dan Tufiş, and Dániel Varga. 2006. The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *5th International Conference on Language Resources and Evaluation (LREC'2006)*, Genoa, Italy, 24–26 May.

Jörg Tiedemann and Lars Nygaard. 2004. The opus corpus - parallel & free. In *Fourth International Conference on Language Resources and Evaluation (LREC'04)*, Lisbon, Portugal, May 26–28.