TBXTools: A Free, Fast and Flexible Tool for Automatic Terminology Extraction

Antoni Oliver
Universitat Oberta de Catalunya
aoliverg@uoc.edu

Mercè Vázquez
Universitat Oberta de Catalunya
mvazquezga@uoc.edu

Abstract

The manual identification of terminology from specialized corpora is a complex task that needs to be addressed by flexible tools, in order to facilitate the construction of multilingual terminologies which are the main resources for computer-assisted translation tools, machine translation or ontologies. The automatic terminology extraction tools developed so far either use a proprietary code or an open source code, that is limited to certain software functionalities. To automatically extract terms from specialized corpora for different purposes such as constructing dictionaries, thesauruses or translation memories, we need open source tools to easily integrate new functionalities to improve term selection. This paper presents TBXTools, a free automatic terminology extraction tool that implements linguistic and statistical methods for multiword term extraction. The tool allows the users to easily identify multiword terms from specialized corpora and also, if needed, translation candidates from parallel corpora. In this paper we present the main features of TBXTools along with evaluation results for term extraction, both using statistical and linguistic methodology, for several corpora.

1 Introduction

Automatic terminology extraction (ATE) is a relevant natural language processing task involving terminology which has been used to identify domain-relevant terms applying computational methods (Oliver et al., 2007a; Foo, 2012).

Automatic term extraction is a relevant task that can be useful for a wide range of tasks, such as ontology learning, machine translation, computer-assisted translation, thesaurus construction, classification, indexing, information retrieval, and also text mining and text summarisation (Heid and McNaught, 1991; Frantzi and Ananiadou, 1996; Vu et al., 2008).

The automatic terminology extraction tools developed in recent years allow easier manual term extraction from a specialized corpus, which is a long, tedious and repetitive task that has the risk of being unsystematic and subjective, very costly in economic terms and limited by the current available information. However, existing tools should be improved in order to get more consistent terminology and greater productivity (Gornostay, 2010).

In the last few years, several term extraction tools have been developed, but most of them are language-dependent: French and English –Fastr (Jacquemin, 1999) and Acabit (Daille, 2003); Portuguese –Extracterm (Costa et al., 2004) and ExATOlp (Lopes et al., 2009); Spanish-Basque –Elexbi (Hernaiz et al., 2006); Spanish-German –Autoterm (Haller, 2008); Arabic (Boulaknadel et al., 2008); Slovene and English –Luiz (Vintar, 2010); English and Italian –KX (Pianta and Tonelli, 2010); or English and German (Gojun et al., 2012).

Some tools are adapted to a specialized domain: TermExtractor (Sclano and Velardi, 2007), TermMine (Ananiadou et al., 2009) or BioYaTeA (Golik et al., 2013), for example. Specific tools have been developed to extract corpus-specific lexical items comparing technical and non-technical corpus: TermoStat (Drouin, 2003). And other tools are based on under-resourced language –TWSC (Pinnis et al., 2012)–, or use semantic and contextual information –Yate (Vivaldi and Rodríguez, 2001).

Furthermore, there was TermSuite, which was developed during the European project TTC (Terminology Extraction, Translation Tools and Com-
parable Corpora). This project focused on
the automatic or semi-automatic acquisition of
aligned bilingual terminologies for computer-
assisted translation and machine translation. To
this end, automatic terminology extraction is part
of the process of identifying terminologies from
comparable corpora (Blancafort et al., 2010).

This paper presents TBXTools, a free automatic
term extraction tool which allows multiword terms
from specialized corpora to be identified easily,
combining statistical and linguistic methods.

This paper is structured as follows: in the next
section we present the TBXTools implementation
and statistical and linguistic methods, as well as
the automatic finding of translation equivalents.
The experimental settings are described in detail
in section 3. The paper concludes with some final
remarks and ideas for future work.

2 TBXTools

2.1 Description

TBXTools is a Python class that implements a set
of methods for ATE along with other utilities re-
lated to terminology management. This tool has a
free software licence and can be downloaded from
SourceForge\(^1\). TBXTools is an evolution of pre-
vious tools developed by the authors (Oliver and
Vázquez, 2007; Oliver et al., 2007b). The tool is
still under development but it already implements
a set of methods that permit the following func-
tionalities:

- Statistical term extraction using n-grams and
  stop words and allowing some normalizations:
capital letter normalization, morphological nor-
malization and nested candidate detection.

- Linguistic term extraction using morpho-
syntactic pattern and a tagged corpus. Any ex-
ternal tagger and a connection with a server run-
ning Freeling (Padró and Stanilovsky, 2012) are
implemented. The tool uses an easy formal-
ism for the expression of patterns, allowing the
use of regular expressions and lemmatization of
some of the components, if required.

- Detection of translation candidates in parallel
corpora, using a statistical strategy.

- Automatic learning of morphological patterns
  from a list of reference terms.

Nowadays TBXTools does not have a user in-
terface, but it will be developed in the future. At
present the extraction is done by means of sim-
ple Python scripts calling the TBXTools class. In
this paper we will see the code of some of these
scripts. Several examples of scripts can be found
in the TBXTools distribution.

2.2 Statistical Terminology Extraction

The statistical strategy for terminology extraction
is based on the calculation of n-grams, that is, the
combination of \(n\) words appearing in the corpus.
After this calculation, filtering with stop words is
performed, eliminating all the candidates begin-
ning or ending with a word from a list. Some nor-
malizations, such as case normalization, nesting
detection and morphological normalization, can
be performed. Here we can see a complete code
for terminology extraction:

```python
from TBXTools import *
e=TBXTools()
e.load_sl_corpus("corpus.txt")
e.load_stop_l1("stop-eng.txt")
e.set_nmin(2)
e.set_nmax(3)
e.statistical_term_extraction()
e.case_normalization()
e.nesting_detection()
e.load_morphopatterns("morpho-eng.txt")
e.morpho_normalization()
e.save_term_candidates("candidates.txt")
```

The code, as can be seen, is very simple. First
of all, we import TBXTools and create a TBX-
Tools object, called \(e\) in the example. This code
calculates the term candidates from the corpus in
the `corpus.txt` file using the stop words in the
`stop.txt` file. Afterwards, we fix the minimum \(n\)
to 2 and the maximum to 3, in order to calculate
bigrams and trigrams term candidates. The next
step in the code performs the statistical term ex-
traction. After that, the following normalizations
are implemented:

- Case normalization: it tries to collapse the same
term appearing with a different case: for ex-
ample, “interest rate”, “Interest Rate” and “IN-
TEREST RATE” into “interest rate”.

- Nesting detection: sometimes shorter term can-
didates are not terms in and of themselves, but
are part of a longer term. For example, the bi-
gram term candidate “national central” is a part
of the trigram term candidate “national central
bank”.

---

\(^1\)http://sourceforge.net/projects/tbxtools/

Proceedings of Recent Advances in Natural Language Processing, pages 473-479,
Hissar, Bulgaria, Sep. 7-9 2015.
Morphological normalization: it tries to collapse several forms at the same time into a single form, for example, to collapse the plural term candidate “economic policies” into “economic policy”. To perform this normalization, a simple set of morphological patterns is used.

After all these normalizations, the term candidates are saved into the text file "candidates.txt". The candidates are stored in descending frequency order and the value of frequency is also stored, as in the following example:

53 euro banknotes
51 central bank
47 payment institution
23 payment instrument

2.3 Linguistic Terminology Extraction

To perform linguistic terminology extraction we need a POS-tagged corpus. The tagging can be performed with any tagger offering lemma and POS tags. TBXTools can be easily used with Freeling. In the following example we will perform linguistic extraction from a tagged corpus (ct.txt) using a set of patterns (p) and storing the term candidates into the file candidates.txt. The Python script would look like this:

```python
from TBXTools import *
e=TBXTools()
e.load_tagged_corpus("ct.txt")
e.load_ling_termextract_patterns("p.txt")
e.ling_term_extract()
e.save_term_candidates("candidates.txt")
```

If our tagged corpus uses the Penn Treebank POS tags, the patterns should be expressed with these same tags, for example NN NN or JJ NN. If we want to use the lemma instead of the word form in a pattern, we use square brackets, as in NN [NN.*]. Note that in this pattern we have also used regular expressions to make it more general. The formalism also allows for the inclusion of the lemmas and word forms in the patterns, as in [N.*]/of/[N.*], where the lemma of is used.

TBXTools is able to calculate the translation equivalent for a given term using a parallel corpus. If the given term appears several times in the corpus, TBXTools can use simple statistical calculations to try to select the translation equivalent in the target language. In the following code we can observe how this task can be performed:

```python
from TBXTools import *
import codecs
e=TBXTools()
e.load_tabtxt_corpus("corpus.txt")
e.load_stop_l2("stop.txt")
...
tr=e.get_statistical_translation_candidate(t, candidates=5)
print(t,tr)
...
```

With this code we load a parallel corpus and a list of stop words for the target language. Then we calculate the translation equivalent (tr) from the term (t) and ask to return 5 candidates. The output would as follows:

```
payment institution entidad de pago:
servicios de pago: dinero electrónico:
entidad de crédito: Estado miembro:
```

In this example we want to find the translation of “payment institution” and we get 5 candidates in Spanish. In this case the first one is the correct one (“entidad de pago”).

3 Experimental Settings

3.1 Resources

We performed some experiments on terminology extraction using controlled corpora, that is, we knew in advance which terms are in these corpora. We used a subset of 1,000 segments from the ECB (European Central Bank) corpus and EMEA (European Medicines Agency documents corpus) corpus (Tiedemann, 2012) in English.

A manual selection of terms in these corpus subsets was performed. Terms in the corpus were manually annotated and those in plural form were lemmatized. This annotation task was performed independently by two terminologists, and those cases with no agreement were discussed and a common solution adopted. Having these annotated corpora, we extracted a list of all terms and their frequencies. Two different lists were extracted for each corpus: a list containing the terms as they appeared in the corpus (in plural or lemma form), and another list containing only the lemmatized terms. These lists of extracted terms from the manually annotated corpora were used to evaluate the extraction results.
3.2 Methodology

In our experiments we performed and evaluated 3 different tasks for both corpus subsets:

– Statistical terminology extraction for English
– Linguistic terminology extraction for English
– Automatic extraction of translation equivalents into Spanish

In all these experiments we used TBXTools. The programs used have been described in section 2.

3.3 Evaluation and Results

Since we have a list of all terms appearing in both corpus subsets, evaluation of the automatic terminology extraction experiments could be done automatically. We have evaluated precision for different values of frequency. TBXTools has a method that, given a set of translation candidates, a list of terms and a value of frequency, calculates the precision and recall values. Here we can see a piece of code for the evaluation task:

```python
... 
ed.load_evaluationterms("ref_terms.txt")
(p,r)=extractor.eval_prec_recall_byfreq(5)
... 
```

This code returns the value of precision \((p)\) and recall \((r)\) for all candidates with a frequency of 5 or higher.

The task of automatic extraction of translation equivalents has been evaluated manually by a terminologist.

Statistical Approach

In tables 1 to 4 we can see the evaluation results for the statistical approach. We have presented figures of precision \((P)\) and recall \((R)\) for bigrams and trigrams and for the ECB and EMEA subsets of 1,000 segments. As we can observe in all results, for high values of frequency we get very few term candidates and the values of precision are not significant, as recall is too low.

In Table 1 we can observe the results for the statistical approach using the ECB corpus for bigrams:

<table>
<thead>
<tr>
<th>Freq</th>
<th>Word forms</th>
<th>P.</th>
<th>R.</th>
<th>Lemmata</th>
<th>P.</th>
<th>R.</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>100.00</td>
<td>0.34</td>
<td>50.00</td>
<td>0.41</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>100.00</td>
<td>2.06</td>
<td>71.43</td>
<td>2.03</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>61.54</td>
<td>5.50</td>
<td>57.14</td>
<td>6.50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>59.09</td>
<td>13.40</td>
<td>41.27</td>
<td>10.57</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>43.21</td>
<td>41.58</td>
<td>27.37</td>
<td>30.49</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>29.58</td>
<td>73.20</td>
<td>17.10</td>
<td>48.37</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: Results for statistical approach using ECB corpus for bigrams

In Table 2 we can observe the results for trigrams. The total number of candidates for trigram word forms are 726, and for trigram lemmata 722. As we can see, the precision values for trigrams are worse than for bigrams (for frequency equal to 2, from 43.21% to 18.72% for word forms and from 27.37% to 5.47% for lemmata).

<table>
<thead>
<tr>
<th>Freq</th>
<th>Word forms</th>
<th>P.</th>
<th>R.</th>
<th>Lemmata</th>
<th>P.</th>
<th>R.</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>0</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>20</td>
<td>100.00</td>
<td>1.87</td>
<td>50.00</td>
<td>2.38</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>75.00</td>
<td>2.80</td>
<td>33.33</td>
<td>4.76</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>50.00</td>
<td>9.35</td>
<td>25.00</td>
<td>11.90</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>18.72</td>
<td>35.51</td>
<td>5.47</td>
<td>26.19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>10.06</td>
<td>68.22</td>
<td>2.08</td>
<td>35.71</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Results for statistical approach using ECB corpus for trigrams

In tables 3 and 4 the results for the EMEA sub-corpus are presented. The total number of candidates for bigram word forms is 432, and for bigram lemmata, 422, whereas for trigrams the total is 367 both for word forms and lemmata. The behaviour here is very similar to that of the ECB corpus, but here the number of bigram and trigram candidates is lower than for the ECB corpus.

Linguistic Approach

In tables 5 to 8 the results for the linguistic approach are presented.
Table 3: Results for statistical approach using EMEA corpus for bigrams

<table>
<thead>
<tr>
<th>Freq</th>
<th>Word forms</th>
<th>Lemmata</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>100.00</td>
<td>1.90</td>
</tr>
<tr>
<td>10</td>
<td>77.78</td>
<td>8.86</td>
</tr>
<tr>
<td>5</td>
<td>52.24</td>
<td>22.15</td>
</tr>
<tr>
<td>2</td>
<td>30.41</td>
<td>70.25</td>
</tr>
<tr>
<td>1</td>
<td>27.78</td>
<td>75.95</td>
</tr>
</tbody>
</table>

Table 4: Results for statistical approach using EMEA corpus for trigrams

<table>
<thead>
<tr>
<th>Freq</th>
<th>Word forms</th>
<th>Lemmata</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>100.00</td>
<td>2.33</td>
</tr>
<tr>
<td>10</td>
<td>28.57</td>
<td>4.65</td>
</tr>
<tr>
<td>5</td>
<td>13.89</td>
<td>11.63</td>
</tr>
<tr>
<td>2</td>
<td>9.70</td>
<td>67.44</td>
</tr>
<tr>
<td>1</td>
<td>8.45</td>
<td>72.09</td>
</tr>
</tbody>
</table>

Table 5: Results for linguistic approach using ECB corpus for bigrams

<table>
<thead>
<tr>
<th>Freq</th>
<th>Word forms</th>
<th>Lemmata</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>66.67</td>
<td>2.75</td>
</tr>
<tr>
<td>5</td>
<td>38.14</td>
<td>8.39</td>
</tr>
<tr>
<td>2</td>
<td>41.10</td>
<td>33.33</td>
</tr>
<tr>
<td>1</td>
<td>25.82</td>
<td>67.70</td>
</tr>
</tbody>
</table>

Table 6: Results for linguistic approach using ECB corpus for trigrams

<table>
<thead>
<tr>
<th>Freq</th>
<th>Word forms</th>
<th>Lemmata</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>33.33</td>
<td>0.93</td>
</tr>
<tr>
<td>5</td>
<td>30.77</td>
<td>3.74</td>
</tr>
<tr>
<td>2</td>
<td>13.95</td>
<td>16.82</td>
</tr>
<tr>
<td>1</td>
<td>9.36</td>
<td>49.53</td>
</tr>
</tbody>
</table>

Table 7: Results for linguistic approach using EMEA corpus for bigrams

<table>
<thead>
<tr>
<th>Freq</th>
<th>Word forms</th>
<th>Lemmata</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>66.67</td>
<td>2.75</td>
</tr>
<tr>
<td>5</td>
<td>38.14</td>
<td>8.39</td>
</tr>
<tr>
<td>2</td>
<td>41.10</td>
<td>33.33</td>
</tr>
<tr>
<td>1</td>
<td>25.82</td>
<td>67.70</td>
</tr>
</tbody>
</table>

Table 8: Results for linguistic approach using EMEA corpus for trigrams

For the extraction of bigrams candidates we have used a set of patterns that have been learnt with TBXTools. This feature uses the tagged corpus and a set of reference terms and returns a list of patterns. This list should be manually revised and modified in order to make the patterns more general.

In Table 7 the results for the linguistic approach using the ECB corpus for bigrams are presented. For frequency equal to 2, a precision of 41.10% for word forms and 36.48% for lemmata is achieved. If we now observe the difference between these values (a difference of 4.62 points instead of the 15.84 points for morphological normalization in the statistical approach), we can conclude that the linguistic approach performs much better in the task of normalizing the terms into their base form.

Automatic Extraction of Translation Equivalents in Parallel Corpora

In this section we present the results for the experiments with automatic extraction of translation equivalents in parallel corpora. The Spanish equivalents selection for the English terms (in lemma form) in ECB and EMEA subcorpora was done by two experts translators. As TBXTools is able to return several translation candidates for each corpora, we assessed if the first candidate was correct ($P_1$) and if any of the first five candidates were correct ($P_5$). As the algorithm did not produce Spanish translations for many English terms, we also presented a corrected precision ($P_1^*$ and $P_5^*$), taking only into account the English terms for which the algorithm returned some translation candidates. In some cases we failed to find the translation of a term because we searched using the lemma form and the term always appeared in plural in the corpus. Tables 9 and 10 shows the recall values.
<table>
<thead>
<tr>
<th>ECB 2g</th>
<th>P₁</th>
<th>P₂</th>
<th>P*₁</th>
<th>P*₂</th>
<th>R₁</th>
<th>R₂</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>12.60%</td>
<td>26.01%</td>
<td>27.93%</td>
<td>57.66%</td>
<td>12.60%</td>
<td>26.01%</td>
</tr>
<tr>
<td>ECB 3g</td>
<td>2.78%</td>
<td>12.96%</td>
<td>10%</td>
<td>46.67%</td>
<td>2.78%</td>
<td>12.96%</td>
</tr>
<tr>
<td>EMEA 2g</td>
<td>23.40%</td>
<td>43.97%</td>
<td>34.02%</td>
<td>63.92%</td>
<td>23.40%</td>
<td>43.97%</td>
</tr>
<tr>
<td>EMEA 3g</td>
<td>2.38%</td>
<td>35.70%</td>
<td>4.00%</td>
<td>60.00%</td>
<td>2.38%</td>
<td>35.71%</td>
</tr>
</tbody>
</table>

Table 9: Results for automatic extraction of translation equivalents for 1,000 segments subcorpora

<table>
<thead>
<tr>
<th>P₁</th>
<th>P₂</th>
<th>P*₁</th>
<th>P*₂</th>
<th>R₁</th>
<th>R₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECB 2g</td>
<td>30.89%</td>
<td>47.15%</td>
<td>46.63%</td>
<td>71.17%</td>
<td>30.89%</td>
</tr>
<tr>
<td>ECB 3g</td>
<td>11.11%</td>
<td>36.11%</td>
<td>21.05%</td>
<td>68.42%</td>
<td>11.11%</td>
</tr>
<tr>
<td>EMEA 2g</td>
<td>49.65%</td>
<td>68.79%</td>
<td>56.00%</td>
<td>77.60%</td>
<td>49.65%</td>
</tr>
<tr>
<td>EMEA 3g</td>
<td>16.67%</td>
<td>52.38%</td>
<td>22.58%</td>
<td>70.97%</td>
<td>16.67%</td>
</tr>
</tbody>
</table>

Table 10: Results for automatic extraction of translation equivalents for the full corpora

ments of the corpora (the same subset used for extracting the English term candidates). It is evident that precision for bigrams is much higher than precision for trigrams. This is mainly due to the fact that, in general, frequency for trigram terms is much lower than for bigram terms. This fact becomes less important when we correct the results excluding these terms with no translation candidates.

Table 10 shows the evaluation results using the full corpora for finding the translation candidates. As can be observed, precision and recall values are now much higher, as more English sentences can be found containing the desired term, and therefore there are more Spanish sentences with which to find the translation equivalent.

4 Conclusions and Future Work

This paper has presented a free automatic terminology extraction tool. This tool is written in Python and it can work under any popular operating system. The tool is designed to achieve the following:

- The tool is fast and efficient.
- The tool is flexible, allowing several techniques and normalizations to be used.
- It works in terminal and the user only needs to write simple Python scripts. No Python programming knowledge is required, as scripts are simple and readable. The user can make new scripts by copying and modifying example scripts.
- It is designed to work under Python 2.X and 3.X, without the need for external libraries, avoiding installation problems.

This tool is still under development but it can be used to build monolingual or bilingual terminology glossaries in a fast and efficient way.

In the near future we plan to add the following features:

- Statistical measures for term candidate reordering.
- Improved algorithm for automatic learning of patterns for linguistic terminology extraction.
- Implementation of an algorithm for learning morphological variants of term candidates.
- Development of a simple visual user interface, to make the use of TBXTools even more easy.

In this paper we have also presented the results of the experiments for statistical and linguistic monolingual terminology extraction and for the automatic detection of translation equivalents in parallel corpora.

References


Siham Boulaknadel, Beatrice Daille, and Driss Aboutajdine. 2008. A multi-word term extrac-


Ulrich Heid and John McNaught. 1991. EUROTRA-7 study: Feasibility and project definition study on the reusability of lexical and terminological resources in computerised applications. Final report. CEC-DG XIII.


Antoni Oliver and Mercè Vázquez. 2007. A free terminology extraction suite. Londres. Information Management.


