Proceedings of the 6th International Workshop on Computational Terminology (COMPUTERM 2020), pages 106–113
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

TermEval 2020: Using TSR Filtering Method to Improve Automatic Term Extraction

Antoni Oliver, Mercè Vàzquez
Universitat Oberta de Catalunya
Barcelona (Spain)
aoliverg, mvazquezga@uoc.edu

Abstract

The identification of terms from domain-specific corpora using computational methods is a highly time-consuming task because terms have to be validated by specialists. In order to improve term candidate selection, we have developed the Token Slot Recognition (TSR) method, a filtering strategy based on terminological tokens which is used to rank extracted term candidates from domain-specific corpora. We have implemented this filtering strategy in TBXTools. In this paper we present the system we have used in the TermEval 2020 shared task on monolingual term extraction. We also present the evaluation results for the system for English, French and Dutch and for two corpora: corruption and heart failure. For English and French we have used a linguistic methodology based on POS patterns, and for Dutch we have used a statistical methodology based on n-grams calculation and filtering with stop-words. For all languages, TSR (*Token Slot Recognition*) filtering method has been applied. We have obtained competitive results, but there is still room for improvement of the system.

Keywords: Automatic Terminology Extraction, TSR, Token Slot Recognition

1. Introduction

Automatic Term Extraction (ATE) has been considered a relevant Natural Language Processing task involving terminology since the early 1980s, due to its accurate terminology construction that can improve a wide range of tasks, such as ontology learning, computer-assisted translation or information retrieval. However, automatic term extraction methods implemented up to now usually involve extracting a large list of term candidates that has to be manually selected by specialists (Bourigault et al., 2001; Vivaldi and Rodríguez, 2001), a highly time-consuming activity and a repetitive task that poses the risk of being unsystematic, and very costly in economic terms (Loukachevitch, 2012; Conrado et al., 2013; Vasiljevs et al., 2014).

In order to achieve a more accurate term candidate selection, we implemented the Token Slot Recognition (TSR) method, a filtering strategy based on terminological tokens used to rank extracted term candidates from domain-specific corpora. The TSR filtering method has been implemented in TBXTools, a term extraction tool, and can be used both with statistical and linguistic term extraction (Oliver and Vàzquez, 2015).

The main goal of this paper is to determine whether the TSR filtering method could provide an accurate term candidate's selection from the Annotated Corpora for Term Extraction Research (ACTER) Dataset (Rigouts Terryn et al., 2019), provided by the organizers of the TermEval 2020 shared task on monolingual term extraction (Rigouts Terryn et al., 2020). The TSR filtering method is based on reference terms to provide a precise term candidate selection.

This paper is structured as follows: in Section 2, the background of automatic term extraction is described. In Sections 3 and 4, the TSR filtering method and the TBXTools are described. In Section 5, the experimental part is presented. In section 6 the discussion about the obtained results is presented. The paper is concluded with some final remarks and ideas for future research.

2. Automatic terminology extraction

Under the generic name of *Automatic Terminology Extraction* (ATE) we can find a series of techniques and algorithms for the detection of terms in corpora. ATE programs provide a list of term candidates, that is, a set of words or group of words with high probability of being terms. Results of the ATE programs should be revised by human specialists. The methods for ATE can be classified in two main groups: (Pazienza et al., 2005):

- Statistical methods: term extraction is performed based on statistical properties (Salton et al., 1975) and usually implies the calculation of n-grams of words and filtering them with a list of stop-words. Although the most common and easiest to implement statistical property is the term candidate frequency, a long set of statistical measures and other approaches have been developed for term candidate scoring and ranking (Evert and Krenn, 2001; Vàzquez and Oliver, 2013; Astrakhantsev et al., 2015).
- Linguistic methods (Bourigault, 1992): term extraction is performed based on linguistic properties. Most of the systems use a set of predefined morphosyntactic patterns (Evans and Zhai, 1996). After term candidates are extracted using the patterns, a set of statistical measures, the simplest of them being the frequency, are also used to rank the candidates (Daille et al., 1994).

Most of the systems may be considered as hybrid, as they use both approaches in a higher or lesser extent (Earl, 1970). A recent study indicates that the hybrid approaches are the most relevant, and the strategies that use noun identification, compound terms and TF-IDF metrics are the most significant (Valaski and Malucelli, 2015).

In the last few years a semantic and contextual information is used to improve term extraction systems. The first one involves using lexical semantic categories from an external lexical source of the corpus, such as WordNet (Miller, 1995). The second one involves extracting the semantic categories of the words from the same corpus through contextual elements that refer to the syntactic-semantic combination of words (Velardi et al., 2001). Recently, external semantic resources are also used for building ontologies in the medical domain (Bouslimi et al., 2016).

As already mentioned, with any of these methods we are able to detect a set of term candidates, that is, units with a high chance of being real terms. After the automatic procedure, manual revision must be performed in order to select the real terms from the list of term candidates.

3. Token Slot Recognition filtering method

To get a more accurate term candidate selection from specialized corpora, we implemented the Token Slot Recognition (TSR) method (Vàzquez and Oliver, 2018), a filtering strategy which uses terminological units to rank extracted term candidates from domain-specific corpora.

The algorithm is based on the concept of *terminological token* (a token or word of a term) to filter out term candidates. Thus, an unigram term is formed by a token that can be the first token of a term (FT) or the last token of a term (LT) depending on the language, a bigram term is formed by FT LT, a trigram term is formed by FT MT LT (where MT is the middle token of a term), and a tetragram term is formed by FT MT1 MT2 LT. In general, an n-gram term is formed by FT MT1 [..] MTn-2 LT. For example: for English, a unigram term like ”rate” can be considered an LT unit as it can also be part of a bigram term like ”interest rate”. However, a term like ”interest” can be considered either an LT unit, such as ”vested interest”, or an FT, like ”interest rate”.

The algorithm reads the terminological tokens from a list of already known terms and stores them taking into account its position in the terminological unit (first, middle, last). As a list of already known terms a terminological database for the language and subject can be used. If no terminological database is available, a first terminology extraction without TSR filtering can be performed to create a small set of terms to use for TSR filtering. TSR filtering can be performed iteratively to enrich the set of already known terms to use in the next TSR filtering process.
Thus, the TSR method filters term candidates by taking into account their tokens. To do so, two filtering variants are designed: *strict* and *flexible* filtering. In strict TSR filtering, a term candidate will be kept only if all the tokens are present in the corresponding position. In flexible TSR filtering, a term candidate will be kept if any of the tokens is present in the corresponding position.

The algorithm performs this filtering process recursively, that is, by enlarging the list of terminological tokens with the new selected term candidates. In strict mode this is not possible, because all the validated candidates are formed with already known terminological tokens. With flexible filtering it is possible to extract new terminological units, as the candidates are validated if they have a terminological unit in any position. Furthermore, we designed a *combined TSR filtering* variant. In combined TSR filtering, strict filtering is first used and is then followed by flexible filtering.

Using flexible and combined TSR filtering variants the term candidates are processed in each iteration in descending order of frequency. If a term candidate is not filtered out, this is stored in the output stack following that order. Since the process is recursive in these filtering strategies, the term candidates filtered out in the previous iteration are processed again in descending order of frequency in the following iterations. The process is repeated until no new terminological tokens are detected.

4. TBXTools, a term extraction tool

TBXTools (Oliver and Vàzquez, 2015) is a Python class that provides methods to perform a set of terminology extraction and management tasks. Using this class, Python programs performing state-of-the art terminology extraction tasks can be written with few lines of code. A completely new version of TBXTools have been developed. The old version stored most of the data in memory and this provoked memory problems when working with large corpora. The new version of TBXTools uses a SQLite database to store all the data of a given terminology extraction project, allowing us to work with very big corpora in standard computers with no memory restrictions. Using this database we can open again a project, and we can continue to work in the project.

To use TBXTools a Python3 interpreter[1] should be installed on the computer. As the interpreter is available for most operating systems, TBXTools can be used in Linux, Windows and Mac.

A sample script to perform statistical terminology extraction over the corpus *corpus.txt*, using bigrams and trigrams, and filtering with stopwords (*stop-eng.txt*) is shown below. Term candidates are stored in *candidates.txt*.

[1]`https://www.python.org/`

```
from TBXTools import *
e=TBXTools()
e.create_project("project.sqlite","eng")
e.load_sl_corpus("corpus.txt")
e.ngram_calculation(nmin=2,nmax=3)
e.load_sl_stopwords("stop-eng.txt")
e.statistical_term_extraction()
e.save_term_candidates("candidates.txt")
```

The use of TBXTools is very easy but some minimal knowledge of Python is required. In the near future a graphical user interface providing the main functionalities will be developed.

TBXTools holds a free licence (GNU GPL) and can be downloaded from its Sourceforge page[2].

5. Experimental part

5.1. Methodology

We have participated in the TermEval 2020 shared task on monolingual term extraction in order to provide an accurate term candidate's selection in three languages (English, French and Dutch) and two domain-specific corpora (Corruption and Heart failure) using the ACTER Dataset.

We report in the sections below the results we have obtained for the Corruption corpora, a manually created corpora with the help of the Dutch DGT of the European Commission; and Heart failure corpora, a manually collected corpora based on a corpus of titles (Hoste et al., 2019). Both corpora are part of the ACTER Dataset.

Two different strategies have been used:

- For English and French corpora: linguistic strategy
- For Dutch corpora: statistical strategy

For all the strategies and language pairs a TSR filtering method has been performed. To use TSR filtering a reference terminological glossary should be used. The IATE[3] database has been used in the experiments. We have downloaded the TBX file and used the *IATExtract.jar* program provided to get a subset for the subjects LAW and HEALTH for the three working languages. Then, for each language we have selected the *full form* terms with a *confidence score* of 3 or higher. In Table 1 the number of terms for each reference glossary can be observed.

The linguistic strategy has been performed in the following steps. In Figure 2 the scripts used for each step are shown:

- Corpus tagging has been performed using Freeling (Padró and Stanilovsky, 2012) through its Python API.

[2]https://sourceforge.net/projects/tbxtools/

[3]https://iate.europa.eu/

Glossary	Terms
LAW eng	16,055
LAW fra	15,566
LAW nld	14,860
HEALTH eng	29,463
HEALTH fra	29,051
HEALTH nld	28,825

Table 1: Number of terms in the reference glossaries

```
228 |#|NN
112 |#|JJ |#|NN
40 |#|JJ #||NNS
36 |#|NN |#|IN |#|NN
32 |#|NN |#|NN
```

Figure 1: Example of automatically learnt patterns.

- Automatic learning of POS patterns: Using the tagged corpus and the list of reference terms, a set of POS patterns are automatically learnt. TBXTools can provide a list of learnt patterns along with its frequency, that is, the number of terms that can be detected with the given POS pattern. In Figure 1 an example of the learnt patterns is shown. These patterns are manually revised and some of them are dropped. To decide whether to accept or reject a pattern we take into account its frequency and the examples of extracted terms that can be retrieved using TBXTools. In Table 2 the number of automatically learnt and accepted patterns are shown.
- Linguistic terminology extraction and TSR filtering: the terminology extraction is performed using the tagged corpus and the accepted POS patterns. An additional step of filtering using stop-words and a step of nested terms detection are performed. For English a list of 399 stop-words is used and for French a list of 352 stop-words. As a last step, a combined TSR filtering using the IATE reference terms is performed. As a result, a list of term candidates is obtained.

The script for statistical automatic terminology extraction performed for Dutch can be observed in Figure 3:

- N-gram calculation (with *n* from 1 to 5) and filtering wit stop-words.
- Case normalization.
- Nested terms detection.
- Dropping some term candidates using a rejection regular expressions list. This list usually includes combinations of `.+` (any character) `\w+` (combinations of word characters, that is `[a-zA-Z0-9\_]`, `\W+` (combinations of non word characters) and `[0-9]+` (numbers). Each element of the regular expression will be matched against each component of the given n-gram. For example, the regular expression `.+ \W+`

would reject any bigram with the second element containing one or more non-word characters.

- TSR filtering.

Lang.	Subject	Learnt	Accepted
eng	LAW	62	45
fra	LAW	76	58
eng	HEALTH	41	23
fra	HEALTH	88	77

Table 2: Number of learnt and accepted POS patterns.

5.2. Results and evaluation

The number of term candidates obtained for each language and corpus are shown in Table 3. The evaluation of the results has been performed using the term list provided by the organizers of the task. As no detection of named entities is done in our scripts, the sets of terms including named entities are used. In Table 4 the number of tokens of each corpus along with the number of terms are shown.

Corpus	eng	fra	nld
Corruption	1,001	740	358
Heart failure	1,066	900	744

Table 3: Number of term candidates

Corpus	lang	tokens	terms
Corruption	eng	45,218	1,174
Corruption	fra	50,403	1,217
Corruption	nld	47,288	1,295
Heart failure	eng	45,665	2,585
Heart failure	fra	46,626	2,423
Heart failure	nld	47,734	2,257

Table 4: Number of tokens and terms

As the TSR filtering method aims to filter and resort term candidates with a high likelihood to be terms in the top positions, for each corpus and language, we show the evaluation results for subsets of the list of candidates: the top 100, 200, 500 and 1,000 (when the number of candidates is higher than 1,000). The last row of the Table of results shows the overall values.

In Table 6 the evaluation values for the Corruption corpus for English are shown. As we can observe, best values of precision are achieved for the top positions: 37% of precision for the top 100 candidates, whereas we achieve 26.4% for the overall set (position 1001). But values of recall and F_1 show that top candidates results are very low, because we are getting fewer candidates than the current number of terms in the corpus. To illustrate this benefits of using TSR filtering, in Table 5 we offer results of term candidates extraction without filtering for the corruption English corpus.

Position	Precision	Recall	F1
100	0.23	0.02	0.036
200	0.205	0.035	0.06
300	0.207	0.053	0.084
400	0.21	0.072	0.107
500	0.21	0.089	0.125
600	0.22	0.112	0.149
700	0.22	0.131	0.164
800	0.212	0.145	0.172
1000	0.2	0.17	0.184
2395	**0.151**	**0.307**	**0.202**

Table 5: Evaluation results: Corruption English with no TSR filtering

Position	Precision	Recall	F1
100	0.37	0.032	0.058
200	0.36	0.061	0.105
500	0.336	0.143	0.201
1000	0.264	0.225	0.243
1001	**0.264**	**0.225**	**0.243**

Table 6: Evaluation results: Corruption English

Results for the Corruption corpus for French have a similar behaviour (see Table 7), but we tend to get lower precision but higher recall for all the evaluation positions. The overall results for French achieves lower precision but higher recall, yielding to almost exact F_1 value.

Position	Precision	Recall	F1
100	0.28	0.023	0.043
200	0.285	0.047	0.08
500	0.298	0.122	0.174
1000	0.252	0.207	0.227
1633	**0.214**	**0.287**	**0.245**

Table 7: Evaluation results: Corruption French

The situation is different for Corruption corpus in Dutch (see Table 8), where we achieve worse values both of precision (11.5%) and recall (3,2%), yielding to a very low value of F_1 (0.05). It may suggest that the statistical methodology doesn't work well for this language.

In Tables 9, 10 and 11 we can observe the values for the Heart failure corpus. These values are the one that have been compared with other participants in the shared task. In general, if we compare the results for the Corruption corpus and the Heart failure corpus we observe a higher

Corpus tagging:

```
from TBXTools import *
extractor=TBXTools() extractor.create_project(”ACTER-corruption-ling-eng.sqlite”,”eng”,overwrite=True)
extractor.load_sl_corpus(”corpus-en.txt”)
extractor.start_freeling_api(”en”)
extractor.tag_freeling_api()
extractor.save_sl_tagged_corpus(”corpus-tagged-en.txt”)
```

Automatic learning of POS patterns

```
from TBXTools import *
extractor=TBXTools()
extractor.create_project(”learnpatterns.sqlite”,”eng”,overwrite=True)
extractor.load_sl_tagged_corpus(”corpus-tagged-en.txt”)
extractor.load_evaluation_terms(”IATE-LAW-eng.txt”,nmin=1,nmax=5)
extractor.tagged_ngram_calculation(nmin=1,nmax=5,minfreq=1)
extractor.learn_linguistic_patterns(”learnt-patterns-eng.txt”,representativity=100)
```

Linguistic terminology extraction and TSR filtering:

```
from TBXTools import *
extractor=TBXTools()
extractor.create_project(”linguistic-tsr.sqlite”,”eng”,overwrite=True)
extractor.load_sl_tagged_corpus(”corpus-tagged-en.txt”)
extractor.load_linguistic_patterns(”clean-patterns-eng.txt”)
extractor.tagged_ngram_calculation(nmin=1,nmax=5,minfreq=2)
extractor.load_sl_stopwords(”stop-eng.txt”)
extractor.linguistic_term_extraction(minfreq=2)
extractor.nest_normalization(verbose=False)
extractor.tsr(”IATE-LAW-eng.txt”,type=”combined”,max_iterations=100)
extractor.save_term_candidates(”candidates-linguistic-tsr-eng.txt”,minfreq=2,show_measure=True)
```

Figure 2: Steps and scripts for linguistic terminology extraction

```
from TBXTools import *
extractor=TBXTools() extractor.create_project(”statistical-tsr-nld.sqlite”,”nld”,overwrite=True)
extractor.load_sl_corpus(”corpus-nl.txt”)
extractor.ngram_calculation(nmin=1,nmax=5,minfreq=2)
extractor.load_sl_stopwords(”stop-nld.txt”)
extractor.load_sl_exclusion_regexps(”regexps.txt”)
extractor.statistical_term_extraction(minfreq=2)
extractor.case_normalization(verbose=True)
extractor.nest_normalization(verbose=True)
extractor.regexp_exclusion()
extractor.tsr(”IATE-HEALTH-nld.txt”,type=”combined”,max_iterations=100)
extractor.save_term_candidates(”candidates-tsr-nld.txt”,minfreq=2,show_measure=True)
```

Figure 3: Script for statistical terminology extraction

precision value for Heart failure (for example 34.3% vs. 26.4% for English), but lower values of recall (for example 14.2% vs. 22.5% for English).

As regards Heart failure corpus the best values of precision are obtained for French, but the best values for recall are obtained for English. The values of F_1 for English and French are again almost identical.

With regard to Heart failure corpus the worse results

Position	Precision	Recall	F1
100	0.08	0.006	0.011
200	0.15	0.023	0.04
300	0.12	0.028	0.045
358	**0.115**	**0.032**	**0.05**

Table 8: Evaluation results: Corruption Dutch

are obtained again for Dutch, but results are much better than results obtained from Corruption corpus (29% vs. 11.5% of precision and 9.6% vs. 3.2% of recall).

Position	Precision	Recall	F1
100	0.35	0.014	0.026
200	0.435	0.034	0.062
500	0.43	0.083	0.139
1000	0.347	0.134	0.194
1066	**0.343**	**0.142**	**0.2**

Table 9: Evaluation results: Heart failure English

Position	Precision	Recall	F1
100	0.37	0.015	0.029
200	0.375	0.031	0.057
500	0.384	0.079	0.131
900	**0.363**	**0.135**	**0.197**

Table 10: Evaluation results: Heart failure French

Position	Precision	Recall	F1
100	0.44	0.019	0.037
200	0.385	0.034	0.063
500	0.352	0.078	0.128
744	**0.29**	**0.096**	**0.144**

Table 11: Evaluation results: Heart failure Dutch

The difference in the results between languages can be explained by the different strategies used. For English and French corpora we have used linguistic terminology extraction obtaining better results. Results for English and French are comparable, and the differences between them can be produced by different factors: the precision of the tagger for each language, the number of POS tags in the tagset for each language, French having a higher number of tags. This fact can make the revision of the automatically learnt patterns more difficult.

The different results obtained for the two corpora, Corruption and Heart failure, can be due to several factors. Although the size of the corpora for every subject and every language is almost equal, the number of different terms in Heart failure is higher. For example, for English the Corruption corpus has 45,218 tokens and 1,174 terms, whereas the Heart failure corpus has almost the same number of tokens (45,665) but more than twice number of terms (2,585). The IATE reference terms used for the Token Slot Recognition filtering for Heart failure is almost twice the number of terms used for Corruption (see Table 1).

6. Discussion

The experimental results confirm that the combined TSR filtering method we have implemented to identify terms from Corruption and Heart failure domain-specific corpora is productive in terms of precision than recall for all three languages. As for Corruption domain the best results are obtained for English and as for Heart failure the best results are obtained for French. To apply the TSR filtering strategy we have use IATE glossaries for law and health. These glossaries are domain-specific, but for broader domains than the corpora. Results obtained could be enhanced using more specific reference glossaries.

The low results obtained for Dutch may be explained by the statistical methodology used. We decided to use statistical terminology extraction because the tagger we use, Freeling, is not available for Dutch. In further experiments we plan to use any available Dutch tagger, as for example TreeTagger[4] (Schmid, 1994) or Frog[5] (Bosch et al., 2007). We will adapt the output of these taggers to the TBXTools format for tagged corpora and perform a linguistic terminology extraction.

7. Conclusions and future work

In the TermEval 2020 shared task on monolingual term extraction we have implemented the combined TSR filtering method using TBXTools in order to extract the highest number of terms from Corruption and Heart failure corpora from the ACTER Dataset. This methodology uses tokens from already known terms, in this case from IATE glossaries, to search term candidates containing some tokens related to the subject of the corpora. The process is iterative and the list of terminological tokens can be enriched in each iteration, allowing the discovery of completely new terms.

The results obtained from the shared task can confirm that the combined TSR filtering method is suitable for term candidates extraction in any domain-specific corpora. Moreover, the TSR filtering method results would have been better if the reference terms had been more closely associated with the subject corpora.

[4]`https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/`

[5]`https://languagemachines.github.io/frog/`

As a future work, we plan to test the TSR filtering method with larger corpora and in other languages and domains.

8. References

Astrakhantsev, N. A., Fedorenko, D. G., and Turdakov, D. Y. (2015). Methods for automatic term recognition in domain-specific text collections: A survey. *Programming and Computer Software*, 41(6):336–349.

Bosch, A. v. d., Busser, B., Canisius, S., and Daelemans, W. (2007). An efficient memory-based morphosyntactic tagger and parser for dutch. *LOT Occasional Series*, 7:191–206.

Bourigault, D., Jacquemin, C., and L'Homme, M.-C. (2001). Introduction. In *Recent Advances in Computational Terminology*, page iix–xviii, Amsterdam, The Netherlands. John Benjamins.

Bourigault, D. (1992). Surface grammatical analysis for the extraction of terminological noun phrases. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 3*, COLING '92, pages 977–981, Stroudsburg, PA, USA. Association for Computational Linguistics.

Bouslimi, R., Akaichi, J., Gaith Ayadi, M., and Hedhli, H. (2016). A medical collaboration network for medical image analysis. In *Network Modeling Analysis in Health Informatics and Bioinformatics*, 5, pages 1–11.

Conrado, M. S., Pardo, T., and Rezende, S. O. (2013). Exploration of a rich feature set for automatic term extraction. In *Advances in Artificial Intelligence and Its Applications*, Lecture Notes in Computer Science, page 342–354, Berlin, Heidelberg. Springer.

Daille, B., Gaussier, E., and Langé, J.-M. (1994). Towards automatic extraction of monolingual and bilingual terminology. In *Proceedings of the 15th Conference on Computational Linguistics - Volume 1*, COLING '94, pages 515–521, Stroudsburg, PA, USA. Association for Computational Linguistics.

Earl, L. L. (1970). Experiments in automatic extracting and indexing. *Information Storage and Retrieval*, 6(4):313 – 330.

Evans, D. A. and Zhai, C. (1996). Noun-phrase analysis in unrestricted text for information retrieval. In *Proceedings of the 34th Annual Meeting on Association for Computational Linguistics*, ACL '96, pages 17–24, Stroudsburg, PA, USA. Association for Computational Linguistics.

Evert, S. and Krenn, B. (2001). Methods for the qualitative evaluation of lexical association measures. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, page 188–195. AWERProcedia Information Technology Computer.

Hoste, V., Vanopstal, K., Rigouts Terryn, A., and Lefever, E. (2019). The trade-off between quantity and quality. comparing a large web corpus and a small focused corpus for medical terminology extraction. In *Across Languages and Cultures*, pages 197–211.

Loukachevitch, N. V. (2012). Automatic term recognition needs multiple evidence. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*, page 2401–2407.

Miller, G. A. (1995). Wordnet: a lexical database for english. In *Communications of the ACM*, 38, page 39–41.

Oliver, A. and Vàzquez, M. (2015). Tbxtools: a free, fast and flexible tool for automatic terminology extraction. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 473–479.

Oliver, A. and Vàzquez, M. (2015). TBXTools: A free, fast and flexible tool for automatic terminology extraction. In *Proceedings of Recent Advances in Natural Language Processing (RANLP-2015)*, pages 473–479.

Padró, L. and Stanilovsky, E. (2012). Freeling 3.0: Towards wider multilinguality. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey, May. ELRA.

Pazienza, M. T., Pennacchiotti, M., and Zanzotto, F. M. (2005). Terminology extraction: an analysis of linguistic and statistical approaches. In *Knowledge mining*, pages 255–279. Springer.

Rigouts Terryn, A., Hoste, V., and Lefever, E. (2019). No uncertain terms: A dataset for monolingual and multilingual automatic term extraction from comparable corpora. *Language Resources and Evaluation*.

Rigouts Terryn, A., Drouin, P., Hoste, V., and Lefever, E. (2020). Termeval 2020: Shared task on automatic term extraction using the annotated corpora for term extraction research (acter) dataset. In *Proceedings of Computational Terminology CompuTerm 2020*, COMPUTERM 2020, pages 1–4, Paris, France. European Language Resources Association.

Salton, G., Yang, C.-S., and Yu, C. T. (1975). A theory of term importance in automatic text analysis. *Journal of the American society for Information Science*, 26(1):33–44.

Schmid, H. (1994). Probabilistic part-of-speech tagging using decision trees. In *Proceedings of International Conference on New Methods in Language Processing, Manchester, UK.*

Valaski, J., R. S. and Malucelli, A. (2015). Approaches and strategies to extract relevant terms: How are they being applied? In *Proceedings of the International Conference on Artificial Intelligence (ICAI 2015)*, page 478–484, San Diego, USA. The Steering Committee of the World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

Vasiljevs, A., Pinnis, M., and Gornostay, T. (2014). Service model for semi-automatic generation of multilingual terminology resources. In *Proceedings of the Terminology and Knowledge Engineering Conference*, page 67–76.

Velardi, P., Missikoff, M., and Basili, R. (2001). Identification of relevant terms to support the construction of domain ontologies. In *Proceedings of the Workshop on Human Language Technology and Knowledge Management*, pages 1–8, Morristown, USA. Association for

Computational Linguistics.

Vivaldi, J. and Rodríguez, H. (2001). Improving term extraction by combining different techniques. In *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, page 31–48, Amsterdam, The Netherlands. John Benjamins.

Vàzquez, M. and Oliver, A. (2013). Improving term candidate validation using ranking metrics. In *Proceedings of the 3rd World Conference on Information Technology (WCIT-2012)*, page 1348–1359. AWERProcedia Information Technology Computer.

Vàzquez, M. and Oliver, A. (2018). Improving term candidates selection using terminological tokens. In *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, pages 122–147, Amsterdam, The Netherlands. John Benjamins.