Combining Dictionary- and Corpus-Based Concept Extraction

Joan Codina-Filbà[1] and **Leo Wanner**[2]

Abstract. Concept extraction is an increasingly popular topic in deep text analysis. Concepts are individual content elements. Their extraction offers thus an overview of the content of the material from which they were extracted. In the case of domain-specific material, concept extraction boils down to term identification. The most straightforward strategy for term identification is a look up in existing terminological resources. In recent research, this strategy has a poor reputation because it is prone to scaling limitations due to neologisms, lexical variation, synonymy, etc., which make the terminology to be submitted to a constant change. For this reason, many works developed statistical techniques to extract concepts. But the existence of a crowdsourced resource such as Wikipedia is changing the landscape. We present a hybrid approach that combines state-of-the-art statistical techniques with the use of the large scale term acquisition tool BabelFy to perform concept extraction. The combination of both allows us to boost the performance, compared to approaches that use these techniques separately.

1 Introduction

Concept extraction is an increasingly popular topic in deep text analysis. Concepts are individual content elements, such that their extraction from textual material offers an overview of the content of this material. In applications in which the material is domain-specific, concept extraction commonly boils down to the identification and extraction of *terms*, i.e., domain-specific (mono- or multiple-word) lexical items. Usually, these are nominal lexical items that denote concrete or abstract entities. The most straight-forward strategy for term identification is a look up in existing terminological dictionaries. In recent research, this strategy has a poor reputation because it is prone to scaling limitations due to neologisms, lexical variation, synonymy, etc., which make the terminology be submitted to a constant change [15]. As an alternative, a number of works cast syntactic and/or semantic criteria into rules to determine whether a given lexical item qualifies as a term [3, 4, 7], while others apply the statistical criterion of relative frequency of an item in a domain-specific corpus; see, for example, [1, 10, 22, 24, 25]. Most often, state-of-the-art statistical term identification is preceded by a rule-based stage in which the preselection of term candidates is done drawing upon linguistic criteria.

However, most of the state-of-the-art proposals neglect that a new generation of terminological (and thus conceptual) resources emerged and with them, instruments to keep these resources updated. Consider, for instance, BabelNet http://www.babelnet.org [21] and BabelFy http://www.babelfy.org [20]. BabelNet captures the terms from Wikipedia[3], WikiData[4], OmegaWiki[5], Wiktionary[6] and Wordnet [19] and disambiguates and structures them in terms of an ontology. Wikipedia is nowadays a crowd-sourced multilingual encyclopedia that is constantly being updated by more than 100,000 active editors only for the English version. There are studies, cf., e.g., [11], which show that observing edits in the Wikipedia, one can learn what is happening around the globe. BabelFy is a tool that scans a text in search of terms and named entities (NEs) that are present in BabelNet. Once the terms and NEs are detected, it uses the text as context in order to disambiguate them.

In the light of this significant change of the terminological dictionary landscape, it is time to assess whether dictionary-driven concept extraction cannot be factored in into linguistic and corpus-driven concept extraction to improve the performance of the overall task. The three techniques complement each other: while linguistic criteria filter term candidates, statistical measures help detect domain-specific terms from these candidates, and dictionaries provide terms from which we can assume that they are semantically meaningful.

In what follows, we present our work in which we incorporate BabelFy, and by extension BabelNet and Wikipedia, into the process of domain-specific linguistic and statistical term recognition. This work has been carried out in the context of the MULTISENSOR Project, which targets, among other objectives, concept extraction as a basis for content-oriented visual and textual summaries of multilingual online textual material.

The remainder of the paper is structured as follows. In Section 2, we introduce the basics of statistical and dictionary-based concept extraction. In Section 3, we then outline our approach. The set up of the experiments we carried out to evaluate our approach and the results we achieved are discussed in Sections 4 and 5. In Section 6, we discuss the achieved results, while Section 7, finally, draws some conclusions and points out some future work.

2 The Basics of statistical and dictionary-based concept extraction

Only a few proposals for concept extraction rely solely on linguistic analysis to do term extraction, always assuming that a term is a nominal phrase (NP). Bourigault [5], as one of the first addressing the task of concept extraction, uses for this purpose part-of-speech (PoS) tags. Manning and Schütze [16], and Kaur [14] draw upon regular expressions of PoS sequences.

[1] NLP Group, Pompeu Fabra University, Barcelona, email: joan.codina@upf.edu
[2] Catalan Institute for Research and Advanced Studies (ICREA) and NLP Group, Pompeu Fabra University, Barcelona, email: leo.wanner@upf.edu

[3] http://www.wikipedia.org
[4] wikidata.org
[5] omegaWiki.org
[6] wikitionary.org

More common is the extension of statistical term extraction by a preceding linguistic feature-driven term detection stage, such that we can speak of two core strategies for concept extraction: the statistical (or corpus-based) concept extraction and the dictionary-based concept extraction. As already pointed out, concept extraction means here "term extraction". Although resources such as BabelNet are considerably richer than traditional terminological dictionaries, they can be considered as the modern variant of the latter. Let us revise the basics of both of these two core strategies.

2.1 Statistical term extraction

Corpus-based terminology extraction started to attract attention in the 90s, with the increasing availability of large computerized textual corpora; see [13, 6] for a review of some early proposals. In general, corpus-based concept extraction relies on corpus statistics to score and select the terms among the term candidates. In the course of the years, a number of different statistics have been suggested to identify relevant terms and best word groupings; cf., e.g., [2].

As a rule, the extraction is done in a three-step procedure:

1. **Term candidate detection.** The objective of this first step is to find words and multiword sequences that could be terms. This first step has to offer a high recall, as the terms missed here will not be considered in the remainder of the procedure.
2. **Compute features for term candidates.** For each term candidate, a set of features is computed. Most of the features are statistical and measure how often the term is found as such in the corpus and in the document, as part of other terms, and also with respect to the words that compound it. These basic features are then combined to compute a global score.
3. **Select final terms from candidates** Term candidates that obtain higher scores are selected as terms. The cut-off strategy can be based on a threshold applied to the score (obtained from a training set, in order to optimize precision/recall) or on a fixed number of terms (in that case, the top N terms are selected).

In what follows, we discuss each of these steps in turn.

2.1.1 Term candidate detection

The most basic statistical term candidate detection strategies are based on n-gram extraction. Any n-gram in a text collection could be a term candidate. For instance, Foo and Merkel [9] use unigrams and bigrams as term candidates.

n-gram based concept extraction is straightforward to implement. However, it produces too many false positives, which add noise to the following stages. As already mentioned above, for this reason, most of the works use linguistic features such as part-of-speech patterns or NP markers [16, 10] for initial filtering. See [23] for an overview.

2.1.2 Feature Extraction

Once the term candidates have been selected, they need to be scored in order to be ranked with respect to the probability that they are actual terms.

Most of the proposed metrics are based on term frequency TF, as the number of occurrences of a term in a text collection. In Information Retrieval, TF is contrasted to IDF (Inverse Document Frequency), which penalizes the most common terms. For the task of term extraction, IDF of a term candidate can be computed drawing upon a reference corpus, while the frequency of the candidate term in the target domain corpus can be assumed to be TF, such that we get: $TF_{target} * IDF_{ref}$ [16].

Other measures have been developed specifically for term detection. The most common of them are:

- **C-Value** [10]. The objective of the C-Value score is to assign a *termhood* value to each candidate token sequence, considering also its occurrence inside other terms. The C-value expands each term candidate with all its possible nested multiword subterms that will become also term candidates. For instance, the term candidate *floating point routine* includes two nested terms: *floating point*, which is a term, and *point routine*, which is not a meaningful expression.
  The following formula fomarlizes the calculation of the C-Value measure:

$$\begin{cases} log_2 |t| \, TF(t), & t \text{ is not nested} \\ log_2 |t| \, \left(TF(t) - \frac{\sum_{b \in T_t} TF(b)}{P(T_t)} \right) & \text{otherwise} \end{cases} \quad (1)$$

  where t is the candidate token sequence, T_t the set of extracted candidate terms that contain t, and $P(T_t)$ the number of the candidate terms.
- **Lexical Cohesion** [22]. Lexical cohesion computes the cohesion of multiword terms, that is, at this stage, any arbitrary n-gram. This measure is a generalization of the Dice coefficient; it is proportional to the length of the term and the frequency:

$$LC(t) = \frac{|t| log_{10} \left(TF(t) \right) TF(t)}{\sum_{w \in t} TF(w)} \quad (2)$$

  where $|t|$ is the length of the term and w the number of words that compound it.
- **Domain Relevance** [25]. This measure compares frequencies of the term between the target and reference datasets:

$$DR(t) = \frac{TF_{target}(t)}{TF_{target}(t) + TF_{ref}(t)} \quad (3)$$

- **Relevance** [24]. This measure has been developed in an application that focuses on Spanish. The syntactic patterns used to detect term candidates are thus specific for Spanish, but the term scoring is language-independent. The formula aims to give less weight to terms with lower frequency in the target corpus and a higher value to very frequent terms, unless they are also very frequent in the reference corpus or are not evenly distributed in the target corpus:

$$Relevance(t) = 1 - \frac{1}{log_2 \left(\frac{TF_{target}(t) + DF_{target}(t)}{TF_{ref}(t)} \right)} \quad (4)$$

  where $TF(t)$ is the relative term frequency, while $DF(t)$ is the relative number of documents in which t appears. The document frequency tries to block those terms that appear many times in a single document.
- **Weirdness** [1]. Weirdness takes into account the relative sizes of the corpora when comparing frequencies:

$$Weirdness(t) = \frac{TF_{target}(t) \cdot |Corpus_{ref}|}{TF_{ref}(t) \cdot |Corpus_{target}|} \quad (5)$$

2.1.3 *Term selection*

Each of the metrics in the previous subsection produces a score for each term candidate.The final step is to use the scores produced by the chosen metric to filter out the terms under a given threshold.

Taking the terms sorted by their scores, we expect to have a decreasing precision as we move down to the list, while recall increases. The F-score reaches a maximum around the point where precision and recall cross. The list should be truncated at this point, defining the minimum threshold. But, of course, each dataset provides a different threshold that needs to be set after observing different training sets. Some authors (as, e.g., Frantzi et al. [10]) set an arbitrary threshold; others just measure precision and recall when truncating the list after some fixed number of terms [8].

When more than one metric is available, the different metrics can be combined to produce a single score. There are two main strategies to do it: The first one is to feed a machine learning model with the different metrics and let it learn how to combine these metrics [26]. The simplest procedure in this case is to calculate a weighted average tuned by linear regression; cf., e.g., [22]. The second strategy is to come up with a decision for each metric, trained with its own threshold, and then apply majority voting [27].

2.2 Use of terminological resources for terminology detection

The problem of the use of traditional terminological resources for concept (i.e., term) identification mentioned in Section 1 is reflected by the low recall usually achieved by dictionary-based concept extraction. For instance, studies on the medical domain with the Gene Ontology (GO) terms show a recall between 28% and 53% [17]. To overcome this limitation, different techniques have been developed in order to expand the quantity of matched terms. Thus, Jacquemin [12] uses a derivational morphological processor for analysis and generation of term variants. Other authors, like Medelyan [18], use a thesaurus to annotate a training set for the discovery of terms within similar contexts.

BabelNet is a new type of terminological resource. It reflects the state of the continuously updated large scale resources such as Wikipedia, WikiData, etc. At least in theory, BabelNet should thus not suffer from the coverage shortcoming of traditionally static terminological resources.[7]

BabelFy takes all the n-grams (with $n \leq 5$) of a given text that contain at least one noun, and checks whether they are substrings of any item in BabelNet. To perform the match, BabelFy uses lemmas.

We can thus hypothesize that an approach that draws upon BabelNet is likely to benefit from its large coverage and continuous update.

3 Our Approach

In the MULTISENSOR project, term recognition is realized as a hybrid module, which combines corpus-driven term identification with dictionary-based term identification that is based on BabelFy. Combining corpus-driven and dictionary-based term identification, we aim to enrich BabelFy's domain-neutral strategy with domain information in order to be able to identify domain-specific terms.

Based on the insights from [8, 27], who compare different metrics, we decided to implement the C-Value measure and the Weirdness metric. The C-Value measure serves us to measure the termhood of a candidate term, while the Weirdness metric reveals to what extent a term candidate is domain specific.

However, the Weirdness metric requires some adaptation. The original Weirdness metric can namely range from 0 to infinite, which is not desirable. To keep the possible values within a limited range, we changed the quotient between probabilities to a quotient between IDF's. As a result, Equation 5 is transformed to:

$$DomWeight(t) = \frac{IDF_{ref}(t)}{IDF_{target}(t)} \quad (6)$$

BabelFy offers an API that annotates terms of a given text found in one of the resources it consults (WordNet, Wikipedia, WikiData, Wiktionary, etc.), distinguishing between named entities and concepts. Cf. Figure 1 for illustration. The figure shows the result of processing a sentence with BabelFy's web interface. As can be observed, BabelFy annotates nouns (including multiword nouns), adjectives and verbs (such as *working* or *examine*). In accordance with the goals of MULTISENSOR, we keep only nominal annotations and discard verbal and adjectival ones. Furthermore, BabelFy can be considered a general purpose thesaurus, which is not tailored to any specific domain. For this reason, during domain-specific term extraction as in MULTISENSOR, not all terms that have been annotated by BabelFy should be considered as part of the domain terminology.

To ensure the domain specificity, we index the documents for which the $IDF(t)$ is computed in a Solr index,[8] with a field that indicates the domain to which each of them belongs. This allows us an incremental set up in which new documents can always be indexed and the statistics can be continuously updated.

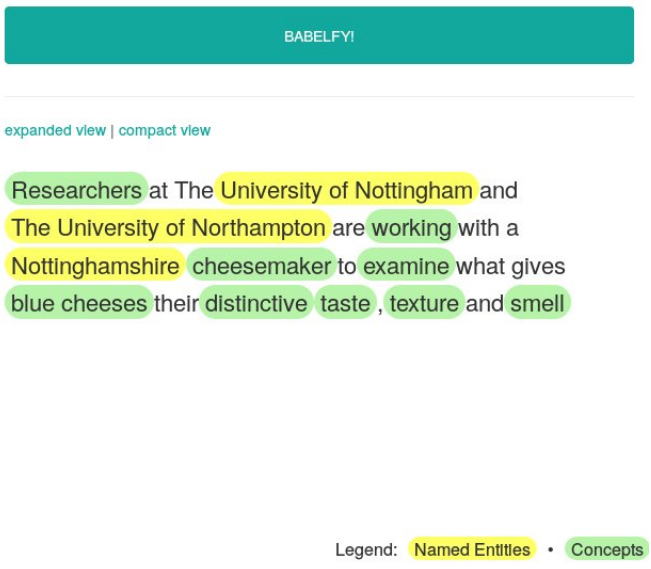

Figure 1. Concepts and named entities detected in a sentence using the BabelFy web interface

The documents indexed in Solr comprise the texts of these documents, together with all the term candidates in them. To index the term candidates, and in order to allow for queries that may match either a full term or parts of it (which can be, again, full terms), we use lemmas (instead of word forms) and underscores between the lemmas to indicate the beginning, middle, and end of the term. The first

[7] Note, however, that even if the Wikipedia is continuously updated, BabelNet is updated in a batch mode from time to time, producing a delay between the crowdsourced changes and their availability in BabelNet.

[8] http://lucene.apache.org/solr

lemma of the term is suffixed with an underscore, the middle lemmas are prefixed and suffixed with underscores, while the last lemma is prefixed with an underscore (for instance, the term candidate *real time clocks* would be indexed as *real_ _time_ _clock*).

At the beginning, the index is filled with the documents that conform the reference and domain corpora. When a new document arrives, we check in both corpora the frequencies of the term candidates as well as the frequencies of their parts as terms and as parts of other terms. To extract these frequencies, several partial matches are required, which can be specified taking advantage of the underscores within the term notation. For instance, to obtain the frequency of the expression *real time* as a term, without that it is part of a longer term, we must search for *real_ _time*. To obtain the frequency of the same sequence of lemmas as part of longer terms, the corresponding query would be *real_ _time_ OR _real_ _time_ OR _real_ _time*. In this last query, the first part would match terms starting with the sequence under consideration (as, e.g., *real time clock*); the second part will match terms that contain the sequence in the middle (as, e.g., *near real time system*); and the last part seeks terms ending with sequence (as, e.g., *near real time*).

Queries in Solr provide the number of documents matching the query. This implies that a document with a multiple occurrence of a term will be counted only once. In some of the formulas of Section 2.1.2, document frequencies are considered, while in others it is the term frequency. In order to minimize this discrepancy, and weight evenly very long and very short documents, we split long documents into groups of about 20 sentences.

To generate term candidates for the statistical term extraction, all NPs in the text are detected. The module takes as input already tokenized sentences of a document. Tokens are lemmatized and annotated with POS and syntactic dependencies. To detect NPs, we go over all the nodes of the tree in pre-order, finding the head nouns and the dependent elements. A set of rules indicates which nouns and which dependants will form the NP. The system includes sets of rules for all the languages we work with: English, German, French and Spanish. Each term candidate is expanded with all the subterms (i.e., n-grams that compose them). The term candidates and all the substrings they contain are then scored using the $C - Value$ and $DomWeight$ metrics. Those with a $DomWeight$ below 0.8 and nested terms with a lower $C - Value$ than the term they belong to are filtered out. The remaining candidates are sorted by decreasing $C - Value$ and, when there is a tie, by $DomWeight$.

After processing the text with BabelFy, we obtain another list of term candidates, namely those that are found in BabelNet. Both lists are merged by intersection and again sorted according to their $C - Value$ and $DomWeight$ scores.

4 Experimental setup

The term extraction methodology described above has been tested for three different use cases. All three use cases are composed by a selection of 1,000 news articles, blogs and other web pages related to different domains. The reference corpus is a set of about 22,000 documents from different sources.

The first use case contains documents about household appliances, with information about both appliances as such and companies involved in the market of household appliances manufacturing and trading. The second use case is about energy policies; it includes news and web pages on green and renewable energy. The third use case covers yoghurt industry; it contains documents about yoghurt products, legal regulations concerning the production and trade with yoghurts, and diary industries.

Table 1. Number of documents and concepts annotated for each use case. The number of indexed chunks indicates in how many different text portions the documents have been split (at sentence boundaries)

Use Case	Name	Num. of documents	Num. of indexed chunks	annotated terms
0	Reference Corpus	21,994	43,808	—
1	Household Appliences	1,000	2,171	123
2	Energy Policies	1,000	1,565	80
3	Yoghurt Industry	1,000	2,096	118

The collection of documents for the three use cases has been extracted from controlled sources, which ensures that the texts within the collection are clean. The documents have been first processed with the goal to detect term candidates, i.e., tokenized, parsed and passed through the NP detector. Once processed, they have been indexed in a Solr index. In addition, all documents have been split into chunks of about 20 sentences to balance the length of the processed texts. In order to evaluate the performance of our hybrid term extraction, for each use case, a set of 20 sentences (from different documents) has been annotated as a ground truth by a team of three annotators.

Table 1 summarizes the information about the different use cases, the reference corpus, the number of original documents, the number of documents after indexing (with some of the documents split as mentioned above), and the number of manually annotated terms for each domain.

5 Evaluation

In order to evaluate the proposed approach to concept extraction, and to observe the impact of the merge of corpus-driven and dictionary-based extraction, we first measured the performance of both of them separately and then of the merge. Table 2 shows the precision and recall of the three runs.

Table 2. Results obtained by the different approaches and the hybrid system in the three use cases ('p' = precision; 'r'= recall)

Use Case	Corpus-driven		Dictionary-based		Hybrid	
	p	r	p	r	p	r
1	38.1%	93.5%	50.3%	76.4%	65.2%	71.54%
2	28.0%	97.3%	36.2%	74.68%	48.3%	70.9%
3	34.8%	79.5%	46.2%	68.4%	60.9%	57.3%
avg	33.6%	90.1%	44.2%	73.2%	58.1%	66.6%

It can be observed that the hybrid approach increases the precision by between 14% and 25% points and decreases the recall by between 7 and 24% . To assess whether the increase of precision compensates for the loss of coverage, we computed the F-score in Table 3.

The table shows that the F-score of the hybrid approach is 7% over the score of the BabelFy (i.e., dictionary-based) approach and 13% above the corpus-driven approach.

The results shown in Tables 2 and 3 have been calculated with all terms provided by corpus-driven and dictionary-based term extraction; only terms with a $DomWeight$ under 0.8 and nested terms

Table 3. F-scores obtained by the different approaches and the hybrid system in the 3 use cases

Use Case	Corpus-driven	Dictionary-driven	Hybrid
1	54.1%	60.7%	68.2%
2	43.5%	48.8%	57.4%
3	48.4%	55.1%	59.1%
avg	49.0%	55.1%	62.1%

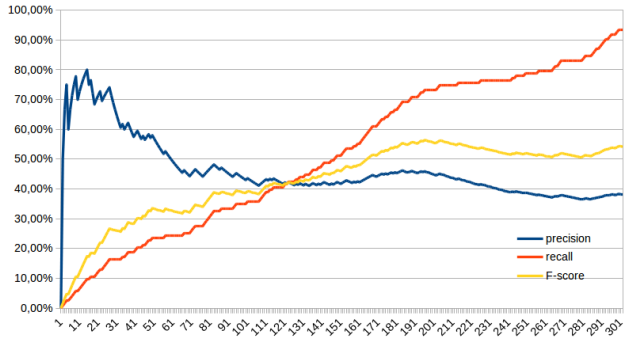

Figure 2. Evolution of precision, recall and F-score as we move down the list of terms generated by the corpus-driven term extraction and sorted by their score

with a $C - Value$ lower than the one of the term they belong to have been filtered out without any further threshold adjustment. In other words, the ordering of the terms according to their $C - Value$ and $DomWeight$ scores has not been considered. If we use only the top N terms with the highest scores, the precision of corpus-based term identification increases. In our current implementation, we do not implement a threshold to cut off the list because the users request the top N terms (with $N = 20$) as a concept profile of a document.

Figure 2 shows how precision, recall and F-score evolve as we move down the list of terms sorted by the score obtained with corpus-driven term extraction (recall that BabelFy does not provide any confidence score).

The score places the most relevant terms at the top of the list, increasing the precision by more than 25 points over the average (as can be observed in the precision/recall/F-score graph, the first 30 terms maintain a precision over 70%).

Figure 3 shows the evolution of precision, recall and F-score for the hybrid term extraction, keeping the ranking provided by the corpus-driven approach. In this case, hybrid term extraction maintains a 100% precision for the first 17 terms and ends with 95% of precision after the first 20 (a single term is wrong among them); 80% precision are maintained for the first 35 terms.

A baseline term identification that does not use scores would obtain a precision of 33%, or 44% using BabelFy and selecting 20 terms at random. When scores are used, the precision of the corpus-driven approach increases up to 47.7%. When both approaches are combined, the average precision for the three use cases increases to 73.6%, resulting in an overall increase of 26% compared to the individual techniques.

6 Discussion

The performance figures displayed in the previous section show that a combination of corpus-driven and dictionary-based term identification achieves better results than in separation, especially when the corpus-driven approach is preceded by a linguistic filtering stage.

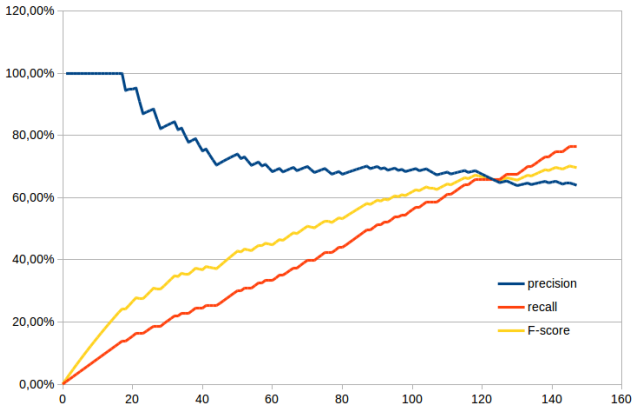

Figure 3. Evolution of precision, recall and F-score as we move down to the list of terms generated by the hybrid system, sorted by the score obtained by statistical metrics

Approaches that are based exclusively on linguistic features serve well to find very rare terms, but they tend to be language- and domain-dependent, which reduces their scalability and coverage. The same applies to approaches that use gazetteers.

Corpus-driven term identification provides term candidates that are domain-specific and common enough to be considered terms, but may be semantically meaningless.

Both corpus-driven and dictionary-based approaches offer a high recall at the expense of low precision because each of them adds its own noise. When combining the two techniques, we increase the precision but lose some recall. However, the decrease of recall is over-compensated by a sufficient increase of precision that leads to the improvement of the F-score. This increase is more evident when we concentrate on terms with a higher score.

The use of an index like Solr to maintain the corpus data allows for the creation of an incremental system that can be updated with upcoming news, making the response dynamic when new concepts appear in a domain.

7 Conclusions and Future Work

We presented a hybrid approach to concept (i.e., term) identification and extraction. The approach combines a state-of-the-art corpus-driven approach with a dictionary lookup based on BabelFy. The combination of both increases the overall performance as it takes the best of both. While statistics are very good in detecting domain-specific terms, dictionaries provide terms which are semantically meaningful.

The use of BabelFy (and thus of BabelNet) allows us to avoid the typical limitation of dictionary-based term identification of coverage. As already argued above, BabelNet, which has been generated automatically from Wikipedia and other resources, is a crowdsourced terminological resource that can be considered to contain a critical mass of terms needed for our task.

Crowdsourced and continuously updated dictionaries ensure the availability of up-to-date resources, but there is still a time offset between the emergence of a new term and its inclusion in the Wikipedia. In the future, it can be insightful to observe the first occurrences of a term and assess its potential status of an emerging concept that cannot be expected to be already in the Wikipedia. This would allow us to give those terms an appropriate score and thus

avoid that they are filtered out.

A relevant topic that we did not look at yet in our current work is the detection of the synonymy of terms, which would further increase the accuracy of the retrieved concept profiles of the documents.

ACKNOWLEDGEMENTS

This work was partially supported by the European Commission under the contract number FP7-ICT-610411 (MULTISENSOR).

REFERENCES

[1] Khurshid Ahmad, Lee Gillam, Lena Tostevin, et al., 'University of surrey participation in TREC8: Weirdness indexing for logical document extrapolation and retrieval (WILDER)', in *Procedings of TREC*, (1999).

[2] Lars Ahrenberg. Term extraction: A review draft version 091221, http://www.ida.liu.se/larah03/publications/tereview_v2.pdf, 2009.

[3] Hassan Al-Haj and Shuly Wintner, 'Identifying multi-word expressions by leveraging morphological and syntactic idiosyncrasy', in *Proceedings of the 23rd International conference on Computational Linguistics*, pp. 10–18. Association for Computational Linguistics, (2010).

[4] Colin Bannard, 'A measure of syntactic flexibility for automatically identifying multiword expressions in corpora', in *Proceedings of the Workshop on a Broader Perspective on Multiword Expressions*, pp. 1–8. Association for Computational Linguistics, (2007).

[5] Didier Bourigault, 'Surface grammatical analysis for the extraction of terminological noun phrases', in *Proceedings of the 14th conference on Computational linguistics-Volume 3*, pp. 977–981. Association for Computational Linguistics, (1992).

[6] M Teresa Cabré Castellví, Rosa Estopa Bagot, and Jordi Vivaldi Palatresi, 'Automatic term detection: A review of current systems', *Recent advances in computational terminology*, **2**, 53–88, (2001).

[7] Paul Cook, Afsaneh Fazly, and Suzanne Stevenson, 'Pulling their weight: Exploiting syntactic forms for the automatic identification of idiomatic expressions in context', in *Proceedings of the workshop on a broader perspective on multiword expressions*, pp. 41–48. Association for Computational Linguistics, (2007).

[8] Denis Fedorenko, Nikita Astrakhantsev, and Denis Turdakov, 'Automatic recognition of domain-specific terms: an experimental evaluation.', in *SYRCoDIS*, pp. 15–23, (2013).

[9] Jody Foo and Magnus Merkel, 'Using machine learning to perform automatic term recognition', in *Proceedings of the LREC 2010 Workshop on Methods for automatic acquisition of Language Resources and their evaluation methods, 23 May 2010, Valletta, Malta*, pp. 49–54, (2010).

[10] Katerina T Frantzi, Sophia Ananiadou, and Junichi Tsujii, 'The c-value/nc-value method of automatic recognition for multi-word terms', in *Research and advanced technology for digital libraries*, 585–604, Springer, (1998).

[11] Martin Rudi Holaker and Eirik Emanuelsen, 'Event detection using wikipedia', Technical report, Institutt for datateknikk og informasjonsvitenskap, (2013).

[12] Christian Jacquemin, Judith L. Klavans, and Evelyne Tzoukermann, 'Expansion of multi-word terms for indexing and retrieval using morphology and syntax', in *Proceedings of the Eighth Conference on European Chapter of the Association for Computational Linguistics*, EACL '97, pp. 24–31, Stroudsburg, PA, USA, (1997). Association for Computational Linguistics.

[13] Kyo Kageura and Bin Umino, 'Methods of automatic term recognition: A review', *Terminology*, **3**(2), 259–289, (1996).

[14] Gagandeep Kaur, SK Jain, Saurabh Parmar, and Anand Kumar, 'Extraction of domain-specific concepts to create expertise profiles', in *Global Trends in Computing and Communication Systems*, 763–771, Springer, (2012).

[15] Michael Krauthammer and Goran Nenadic, 'Term identification in the biomedical literature', *Journal of biomedical informatics*, **37**(6), 512–526, (2004).

[16] Christopher D Manning and Hinrich Schütze, *Foundations of statistical natural language processing*, volume 999, MIT Press, 1999.

[17] Alexa T McCray, Allen C Browne, and Olivier Bodenreider, 'The lexical properties of the gene ontology', in *Proceedings of the AMIA Symposium*, p. 504. American Medical Informatics Association, (2002).

[18] Olena Medelyan and Ian H. Witten, 'Thesaurus based automatic keyphrase indexing', in *Proceedings of the 6th ACM/IEEE-CS Joint Conference on Digital Libraries*, JCDL '06, pp. 296–297, New York, NY, USA, (2006). ACM.

[19] George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller, 'Introduction to wordnet: An on-line lexical database*', *International journal of lexicography*, **3**(4), 235–244, (1990).

[20] Andrea Moro, Alessandro Raganato, and Roberto Navigli, 'Entity linking meets word sense disambiguation: a unified approach', *Transactions of the Association for Computational Linguistics*, **2**, 231–244, (2014).

[21] Roberto Navigli and Simone Paolo Ponzetto, 'Babelnet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network', *Artif. Intell.*, **193**, 217–250, (December 2012).

[22] Youngja Park, Roy J Byrd, and Branimir K Boguraev, 'Automatic glossary extraction: beyond terminology identification', in *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pp. 1–7. Association for Computational Linguistics, (2002).

[23] Maria Teresa Pazienza, Marco Pennacchiotti, and Fabio Massimo Zanzotto, 'Terminology extraction: an analysis of linguistic and statistical approaches', in *Knowledge mining*, 255–279, Springer, (2005).

[24] Anselmo Peñas, Felisa Verdejo, Julio Gonzalo, et al., 'Corpus-based terminology extraction applied to information access', in *Proceedings of Corpus Linguistics*, volume 2001. Citeseer, (2001).

[25] Francesco Sclano and Paola Velardi, 'Termextractor: a web application to learn the shared terminology of emergent web communities', in *Enterprise Interoperability II*, 287–290, Springer, (2007).

[26] Jordi Vivaldi, Horacio Rodríguez, et al., 'Improving term extraction by system combination using boosting', in *Machine Learning: ECML 2001*, 515–526, Springer, (2001).

[27] Ziqi Zhang, José Iria, Christopher Brewster, and Fabio Ciravegna, 'A comparative evaluation of term recognition algorithms.', in *Proceedings of LREC*, (2008).