Evaluating the Impact of Phrase Recognition on Concept Tagging

Pablo N. Mendes[1], **Joachim Daiber**[2], **Rohana Rajapakse**[3], **Felix Sasaki**[4], **Christian Bizer**[1]

[1] Web Based Systems Group, Freie Universität Berlin, Germany
[2] Institute of Formal and Applied Linguistics (ÚFAL), Charles University in Prague, Czech Republic
[3] Goss Interactive Limited, UK
[4] Language Technology Lab, German Research Center for Artificial Intelligence (DFKI), Germany
first.last@fu-berlin.de, last.first@gmail.com, first.last@gossinteractive.com, first.last@dfki.de

Abstract

We have developed DBpedia Spotlight, a flexible concept tagging system that is able to tag – *i.e.* annotate – entities, topics and other terms in natural language text. The system starts by recognizing phrases to annotate in the input text, and subsequently disambiguates them to a reference knowledge base extracted from Wikipedia. In this paper we evaluate the impact of the phrase recognition step on the ability of the system to correctly reproduce the annotations of a gold standard in an unsupervised setting. We argue that a combination of techniques is needed, and we evaluate a number of alternatives according to an existing evaluation set.

Keywords: term recognition, named entity recognition, keyphrase extraction, wikification

1. Introduction

Concept Tagging refers to the task of associating unambiguous 'sense' identifiers to chunks of natural language text. The objectives of concept tagging include, among others, better describing the meaning conveyed by the text and complementing the content with additional information from a Knowledge Base (KB). Concept Tagging can be used to enable a number of applications such as semantic search and faceted browsing, which are able to use factual knowledge to organize and better associate related content to one another.

The process of automatically producing concept taggings usually comprises at least two important tasks: recognizing phrases to annotate - herein called **phrase recognition** - and finding suitable unambiguous identifiers to describe the meaning of those phrases - herein called **disambiguation**. In its broader sense, concept tagging does not specify an annotation focus: which segments of text should be annotated: if all or only some. The "annotation focus" has been approached differently across computational linguistics tasks described in literature. All-words WSD (Word Sense Disambiguation) targets the association of a unique sense identifier to *each and every word* in the input text, while Targeted WSD uses the input text to determine the sense of *one* given ambiguous word (Navigli, 2009). Similarly, Entity Linking (Simpson et al., 2010) focuses on determining the sense of *one* given entity name. Automatic Term Recognition (ATR) (Kageura and Umino, 1996) focuses on extracting phrases that are significant for a given domain, usually with the aim of building a terminology. Similarly, Keyphrase Extraction (Witten et al., 1999) relies on notions of significance: as defined by the user, or as classified by how well phrases capture the contents of the input text. Named Entity Recognition (NER), on the other hand, focuses on recognizing only those phrases representing *entities* of pre-specified types (while classifying them in the correct type). Finally, Wikification (Mihalcea and Csomai, 2007) attempts to replicate the annotation style of Wikipedia, where only the first mention of notable entities in an article should be annotated.

Echoed by the different "annotation focus" observed both in practical applications and literature in computational linguistics, we argue that general purpose annotation systems must allow users to adapt the style of annotation according to their specific needs. In this paper we investigate the impact of phrase recognition on the overall quality of annotation as established by an existing annotated corpus (Kulkarni et al., 2009).

We describe approaches for phrase recognition and evaluate the precision and recall of each approach on that dataset. We investigate if it is possible to adapt to the annotation style of that dataset in an unsupervised setting – i.e. without using the evaluation dataset to train our system. The reason for this design choice is that it allows users to explicitly influence the system at run time, as a Web service, without having to first train the system to indicate an annotation style of choice. In future work we would like to compare unsupervised and supervised approaches for this problem.

The phrase recognition strategies described in this paper were incorporated into the DBpedia Spotlight system (Mendes et al., 2011), where users can customize the kinds of annotations generated by the system through a number of configuration parameters available through a Web interface. The target knowledge base of this system is DBpedia (http://dbpedia.org), a structured database of 3.5M entities/concepts and hundreds of millions of facts extracted from Wikipedia.

2. Methods

A naïve approach for phrase recognition is the enumeration of all possible token sub-sequences from length 1 to the

number of tokens in the input. This approach is, however, impractical as it generates an exponential number of false positives – i.e. phrases that should not be annotated. Executing the disambiguation step for these phrases would be unnecessary and computationally wasteful.
In this section we describe a number of practical approaches that exploit different characteristics of phrases that commonly constitute annotations in different use cases.

2.1. Lexicon-based phrase recognition (L)

A simple approach for phrase recognition is the usage of a string matching algorithm that relies on a lexicon of name variations for the target terms in the knowledge base. For our Lexicon-based approach, we used the LingPipe Exact Dictionary-Based Chunker (Alias-i, 2011) which relies on the Aho-Corasick string matching algorithm (Aho and Corasick, 1975) with longest case-insensitive match. The lexicon used was obtained from the DBpedia Lexicalization Dataset (Mendes et al., 2011).
Because the lexicon-based phrase recognition does not select phrases with regard to their context but merely searches for any phrases known as possible DBpedia entities/concepts, this step still produces a high number of false positives. One example is the set of function words that have entries on Wikipedia, but whose annotation would be undesirable in use cases such as blog annotation, because it would confuse the reader with too many (arguably unnecessary) links. However, eliminating those phrases from the lexicon upfront is not an option, as they may have other significant meanings – e.g. the word '*up*' can be a function word in some contexts, but it can also refer to `Up(2009 film)`, a movie by Pixar.

2.2. Noun-phrase chunk heuristic (L_{NP*})

In many use cases, the objective of annotation is to mark the *things* being talked about in text. Therefore, a simple heuristic to eliminate false positives early in the process is to only annotate terms that are within noun phrases. We therefore extended the Lexicon-based phrase recognizer with a simple heuristic that only allows phrases that contain at least one noun. This annotation style would disregard general concepts such as `Running` and `Crying` when they appear as verbs, but would include concepts like `Perpetual war`. The suitability of this heuristic depends on the use case.

2.3. Noun-phrase chunking with probabilistic dictionary (NP_{L*})

Similar to the L_{NP*} implementation, NP_{L*} assumes that we are considering only noun-phrases. However, instead of using the heuristic above, we use NP chunks extracted by a NP chunker. For each NP chunk and its sub-expressions, we choose the longest expression contained in the set of acceptable phrases(in our case the lexicon). For efficient representation of the set of acceptable phrases, we use a Bloom filter (Bloom, 1970) with a false positive probability of 1%. This data structure allows a conscious trade-off of memory usage, speed and accuracy. Its properties guarantee that an acceptable phrase will be selected if it is in the set, but that there can also be false positive annotations, which we can still disregard in a later step in the annotation process. This implementation is slower than the lexicon-based phrase recognizer L, however, it reduces memory usage significantly. With the same dictionary consisting of 3.8M entries, L requires 2588MB of memory, while NP_{L*} needs 437MB.[1]

2.4. Detecting common words (CW)

In the case of Wikification, only notable entities should be annotated. The Wikipedia guidelines for link creation[2] focus on encouraging the annotation of non-obvious references that help with understanding an encyclopaedic article. The guidelines explicitly instruct users to "avoid linking plain English words".
Therefore, we developed a classifier to attempt the detection of common words at phrase recognition time. We defined two main classifiers: a classifier for single-token phrase candidates and a multi-token phrase candidate classifier. Both classifiers try to find words that carry common knowledge meaning, however the multi-token phrase candidate classifier focuses mainly on detecting phrases that appear in syntactically irregular positions.
In a manual classification of 4497 mentions in encyclopedic and newspaper texts, 50.05% of all mentions were classified as common word occurrences (including annotations of verbs, adjectives, idioms and fixed phrases). Based on this dataset we trained two Bayesian Network models. This choice was empirical – we chose the model that provided the best results in our preliminary tests. Features included neighboring tokens, POS and n-grams. A detailed description of the method and implementation are available online (Daiber, 2011).

2.5. Keyphrase Extraction (KE)

In other use cases (e.g. blogs) one would like to identify only important phrases in the context of the document or website. We use Kea (Frank et al., 1999), a supervised algorithm to identify candidate keyphrases in a document collection. It relies on the Naïve Bayes algorithm and features such as TF*IDF and token distance in order to learn its 'keyphraseness' from a training set of known keyphrases. Admittedly, the fairest evaluation for this spotter would have included a cross-validation experiment over the evaluation set. However, we are interested in testing its fitness for an unsupervised, general-purpose online annotation task. Therefore, in this paper, we use Kea with the default prediction model distributed by its authors.

2.6. Named Entity Recognition (NER)

In use cases such as the concept tagging in online newspapers, we commonly see the focus on entities of specific types (e.g. people and organizations), whereas keyphrases such as '*foreign policy*' are less commonly annotated [3]. For use cases that focus on a small set of common term types (such as People, Location, Organization) it is viable to apply named entity recognizers as a strategy for phrase

[1] The values were measured as the mean memory consumption during phrase recognition performed on 100 short texts.

[2] http://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style_(linking)

[3] http://nyti.ms/qsYAyt

spotter	P	R	time per spot
$L_{>3}$	4.89	68.20	0.0279
$L_{>10}$	5.05	66.53	0.0246
$L_{>75}$	5.06	58.00	0.0286
L_{NP*}	5.52	57.04	0.0331
$NP_{L*>3}$	6.12	45.40	1.1807
$NP_{L*>10}$	6.19	44.48	1.1408
$NP_{L*>75}$	6.17	38.65	1.2969
CW	6.15	42.53	0.2516
Kea	1.90	61.53	0.0505
NER	4.57	7.03	2.9239
NER $\cup$ NP	1.99	68.30	3.1701

Table 1: Evaluation results.

recognition. Therefore we also tested a NER-backed phrase recognizer based on the default models distributed with OpenNLP 1.5.1[4].

2.7. NER extended by noun phrase n-grams (NER $\cup$ NP)

We provide also a hybrid approach mixing named entities and more general terms within noun phrase chunks, inspired by previous work (Ratinov et al., 2011). We consider as phrases only the expressions marked as named entities by the NER phrase recognizer, the noun-phrase chunks extracted by a NP chunker, and all sub-expressions of up to 5 tokens of the noun-phrase chunks. This increases the coverage of the NE phrase recognizer, which tends to generate fewer phrases. This implementation is also based on standard models distributed with the OpenNLP project.

3. Results

We used the annotations produced in the CSAW project (Kulkarni et al., 2009) in which texts from random webpages were manually annotated with Wikipedia page ids.

We conducted an evaluation to assess the precision and recall of each phrase recognizer. We model each phrase as a tuple (phrase, phrase offset) in order to account for phrase repetition in the document. Let S be the set of spotted phrases generated by a given phrase recognizer. Let A be the set of annotated phrases in the gold standard. The phrase recognition **precision** is $P = |S \cap A|/|S|$, the proportion of recognized phrases that are correct. The phrase recognition **recall** is the proportion of phrases in the gold standard that were found by the spotter: $R = |S \cap A|/|A|$. The most important evaluation measure in the context of this paper is arguably the recall. As subsequent steps rely on input obtained from phrase recognition, the overall recall of an annotation system can only be at maximum equal to the recall of phrase recognition. Better precision in phrase recognition is also a desirable feature, as a large amount of false positives may degrade time performance, and errors in boundary detection can negatively influence the accuracy of the disambiguation step.

Table 1 shows the phrase recognition evaluation results. Note that the precision values of the phrase recognition strategies for this dataset are generally low since the CSAW dataset does not exhaustively annotate every potential entity mention.

The best recall was obtained by the NER $\cup$ NP phrase recognizer. However, it generates a large number of false positives that would be unnecessarily sent for disambiguation. The CW phrase recognizer was able to reduce the number of generated phrases in roughly half (from $\approx$168K to $\approx$83K), but at the cost of $\approx$26% loss in recall. The L_{NP*} phrase recognizer had a smaller loss in recall ($\approx$11%), but only removed $\approx$25% of the false positives. The keyphrase extractor had the third best recall, but it generated a higher percentage of false positives as compared to most approaches.

For the lexicon-based approaches, we tested lexica of 3 different sizes. For the lexicon $L_{>3}$, we included all name variations for entities or concepts that are the target of at least 3 links on Wikipedia. Similarly, we applied thresholds 10 and 75, yielding $L_{>10}$ and $L_{>75}$ respectively. We observed that the occurrences of page links in Wikipedia follow a power-law distribution – few entities have many links and many entities have very few links. Reducing the lexicon according to the number of links allows drastic savings in main-memory storage, while focusing on the most common concepts.

Although the disambiguation step is not the focus of this work, for the sake of completeness we also conducted a disambiguation experiment in a targeted WSD setting. In this experiment, each phrase in the gold standard is submitted to DBpedia Spotlight which assigns an identifier to that phrase. The system uses a Vector Space Model of terms scored by Inverse Candidate Frequency (Mendes et al., 2011). The disambiguation accuracy obtained for this dataset is 82.54%.

4. Error Analysis

In this section we discuss some of the most common mistakes made by the phrase recognizer implementations tested.

Stopwords. While completely ignoring stopwords in phrase recognizers is not feasible due to relevant entities that can be confused with stopwords, a better strategy for

[4]http://opennlp.sourceforge.net/models-1.5/

detecting stopword occurrences is needed. Namely, case sensitive treatment and using POS tags are within our plans. While the common word detection (L − CW) managed to detect many of these stopwords, it also removed other common words that were deemed "annotation worthy" by the CSAW dataset, such as: soul, eating, specific, used and neat.

HTML Clutter. Some phrase recognizers consistently recognized text placed outside of what can be considered the main section of the HTML articles used as input. Phrases such as AP (American Press), October, News, etc. were not annotated in the CSAW dataset, but were identified by our methods since we analyzed the full text. Performing Boilerplate Detection (Kohlschütter et al., 2010) could help in this respect.

Phrase boundaries. In phrases such as "the Internet", "the government" and "the embryo" the CSAW dataset includes the determiner as part of the phrase. In our lexicon, the determiners are not included in the lexicon. Roughly 2% of all phrases in CSAW start with determiners.

Notability. Since Wikipedia enforces notability guidelines, phrase recognizer approaches such as NER are bound to make mistakes by correctly identifying people that are not on Wikipedia – e.g. Tawana Lebale, who was a participant in Big Brother Africa but does not have a Wikipedia page dedicated to her. For these cases, a combination of NER and keyphraseness is planned.

5. Conclusion

The evaluation of concept tagging is complicated by the lack of agreement on the question of what can be considered a valid annotation. Some use cases consider only named entities, other focus on keyphrases, some seem to require mixing more than one of these strategies. Some use cases require the annotation of only the first mention of every term, while other use cases require the annotation of each and every occurrence of phrases belonging to a terminology. We argue that general purpose concept tagging systems – such as DBpedia Spotlight and similar Web services – must be adaptable to each user's definition of what constitutes an "annotation-worthy" term.

In this paper we described a number of phrase recognition approaches alongside example use cases that motivate the use of each approach. In order to assess the impact of each approach on the quality of concept tagging, we measured their precision and recall with regard to an existing evaluation dataset that aims at wide annotation coverage. The results show that concept tagging can be very sensitive to the phrase recognition strategy used.

We integrated all evaluated approaches with DBpedia Spotlight, where they can be selected by the user at runtime in order to specify an "annotation policy". A demonstration of the system is available from `http://spotlight.dbpedia.org/demo/lrec2012/`.

Acknowledgements

The authors would like to thank Miloš Stanojević for the discussions that lead to the idea of applying Bloom filters in the NP_{L*} implementation. This work was partially funded by the European Commission through the FP7 grant LOD2 - Creating Knowledge out of Interlinked Data (Grant No. 257943).

6. References

A.V. Aho and M.J. Corasick. 1975. Efficient String Matching: An Aid to Bibliographic Search. *Communications of the ACM*, 18(6):333–340.

Alias-i. 2011. LingPipe 4.0.1. `http://alias-i.com/lingpipe` (accessed 15/04/2011).

Burton H. Bloom. 1970. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM*, 13(7):422–426, July.

Joachim Daiber. 2011. Candidate selection and evaluation in the DBpedia Spotlight entity extraction system. Bachelor's thesis. http://jodaiber.de/BScThesis.pdf.

Eibe Frank, Gordon W. Paynter, Ian H. Witten, Carl Gutwin, and Craig G. Nevill-Manning. 1999. Domain-specific keyphrase extraction. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, IJCAI '99, pages 668–673, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

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

Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining*, WSDM '10, pages 441–450, New York, NY, USA. ACM.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *KDD*, pages 457–466.

Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. DBpedia Spotlight: Shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems (I-Semantics)*.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242, New York, NY, USA. ACM.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41:10:1–10:69, February.

Lev-Arie Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *ACL*, pages 1375–1384.

Heather Simpson, Stephanie Strassel, Robert Parker, and Paul McNamee. 2010. Wikipedia and the web of confusable entities: Experience from entity linking query creation for tac 2009 knowledge base population. In *LREC*.

Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. 1999. Kea: practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries*, DL '99, pages 254–255, New York, NY, USA. ACM.