Towards Concept Identification using a Knowledge-Intensive Approach

Óscar Muñoz-García[1], Andrés García-Silva[2], and Oscar Corcho[2]

[1] Havas Media Group, Madrid, Spain,
oscar.munoz@havasmg.com
http://www.havasmg.com

[2] Ontology Engineering Group, Departamento de Inteligencia Artificial
Universidad Politécnica de Madrid, Madrid, Spain,
hgarcia@fi.upm.es, ocorcho@fi.upm.es
http://www.oeg-upm.net

Abstract. This paper presents a method for identifying concepts in microposts and classifying them into a predefined set of categories. The method relies on the DBpedia knowledge base to identify the types of the concepts detected in the messages. For those concepts that are not classified in the ontology we infer their types via the ontology properties which characterise the type.

Keywords: concept identification, microposts, dbpedia

1 Introduction

In this paper we present an approach to identify concepts and their types in micro posts relying on the DBpedia knowledge base and ontology. Our approach consist first in carrying out a preprocessing task where messages are normalised. Then we attempt to identify candidate concepts leveraging part-of-speech tags and Wikipedia article titles. Next we associate the candidate concepts with DBpedia resources and tap into the ontology hierarchy of classes and resource properties to classify the resource in one of the following types: Person, Organization, Location, and Miscellaneous, which covers films, sport events, software, awards and television shows.

2 Spotting concepts

The concept spotting stage analyses the micropost for extracting the keywords that are candidates for being concepts, or can serve as context for disambiguating the concepts. The stage is executed in three steps, namely:

1. Text normalisation.
2. Part-of-speech tagging.
3. Keyword selection.

Next, each of the steps is described.

2.1 Text Normalisation

The text normalisation step converts the text of the micropost, that often includes metalanguage elements, to a syntax more similar to the usual natural language. Previous results demonstrate that this normalisation step improves the accuracy of the part-of-speech tagger [3]. Specifically, we have implemented several rules for syntactic normalization of Twitter messages (some of them have been described in [6]). The rules executed are the following:

- Transform to lower-case the text completely written with upper-case characters.
- Delete the sequence of characters "RT" followed by a mention to a Twitter user (marked by the symbol "@") and, optionally, by a colon punctuation mark.
- Delete mentions to users that are not preceded by a coordinating or subordinating conjunction, a preposition, or a verb.
- Delete the word "via" followed by a mention to a user at the end of the tweet.
- Delete the hashtags found at the end of the tweet.
- Delete the "#" symbol from the hasthtags that are maintained.
- Delete the hyperlinks contained within the tweet.
- Delete ellipses that are at the end of the tweet, followed by a hyperlink.
- Delete characters that are repeated more than twice (e.g., "yeeeeeessss" is transformed to "yes").
- Transform underscores to blank spaces.
- Divide camel-cased words in multiple words (e.g., "AnalyticsTools" is converted to "Analytics Tools").

2.2 Part-of-speech Tagging

After normalising the micropost text, we execute the part-of-speech analysis of the normalised text. For doing so, we make use of Freeling [7].

2.3 Keyword Selection

Once the part-of-speech tagging is obtained, the keyword selection step is executed. For each sentence within the micropost text we extract all the possible n-grams. In this case a gram is a word in the sentence. After that we select only the n-grams that satisfy the following criteria:

- The n-gram contains at least one noun.
- The n-gram is not contained in a set of stop words.
- If the number of words included in the n-gram is greater than one, the n-gram is included in the set of Wikipedia article titles.
- The n-gram is not contained in another n-gram that has been added to the keyword set (longer n-grams prevail).

To speed-up the process of querying the millions of Wikipedia article titles we have uploaded the list of titles (available at [9]) to a Redis store [8].

3 Semantics of the Concepts

To identify the semantics of the keywords we tap into the DBpedia knowledge base [2] to elicit the types of the concepts to which the keywords correspond. DBpedia contains knowledge from Wikipedia for close to 3.5 million resources; 1.6 million resources are classified in a cross domain ontology containing 272 classes. DBpedia strengths include its large coverage and the fact that its data are exposed in RDF allowing to query them using SPARQL queries through the available endpoint [4].

Our process starts by identifying for each keyword the DBpedia resource which represents its intended meaning. Once we have the corresponding resource we query in DBpedia its classes, whenever they are available, or infer them through the identification of specific resource properties, so that we can identify the types defined in the challenge.

3.1 From keywords to DBpedia resources

First we query DBpedia for a resource with a label matching the keyword. We use exact string matching between the resource label and the keyword which has been previously modified to fit the style of article titles in Wikipedia. The output resource of this query represents the most frequent meaning of the keyword defined by Wikipedia editors. We call this resource default sense of a keyword. If the resource is not related to a disambiguation resource, we consider that the term is not ambiguous and therefore we use the default sense as the one representing the keyword meaning.

In case we do not find a match between the keyword and a resource label we use an spelling service that suggests similar titles of Wikipedia articles. This spelling service[3] compares the n-grams based on characters of both keywords and article titles, and takes into account the popularity of the articles in Wikipedia (*i.e.*, the times that an article has been linked from other articles) when producing the final ranking of suggestions. We use the most similar suggestion, above a given threshold, for searching for the DBpedia resource.

If the resource is related to a disambiguation resource, then we have to select the proper sense among the candidates. To do so we leverage the correspondence of DBpedia resources with Wikipedia articles to obtain textual descriptions of each resource. Thus, we calculate similarity between each resource and the term by comparing the resource textual description with the term context. The most similar resource is selected as the resource representing the term meaning. By context we mean the set of keywords identified in the same sentence.

To calculate similarity between the keyword context and the resource description we use a vector space model. The components of the vectors are the most frequent terms of the Wikipedia articles related to each candidate DBpedia resource. To populate the vectors representing resources we use term frequency

[3] The spelling service was built upon Lucene [1] spell checker.

and inverse document frequency (TF-IDF) as term weighting scheme. IDF is calculated using only the set of textual descriptions corresponding to the candidate resources. We calculate the cosine of the angle between the vector representing the keyword context with each of the vectors representing candidate resources. The candidate with the highest cosine is selected as the resource to represent the keywords. Details of this procedure can be found in [5].

In short for ambiguous keywords if there is not context and there is a default sense we select the DBpedia resource corresponding to the default sense. If there is context and default sense, and the context do not overlap with any of the candidate vectors we use the DBpedia resource corresponding to the default sense too. If there is overlap between the context and candidate vectors we use the most similar candidate.

3.2 Identifying concept types

We manually select the classes from DBpedia and linked ontologies that allow us to identify the types of the concepts defined in the challenge. For instance,

- *dbpedia-owl:Person* and *foaf:Person* are the classes for People;
- *dbpedia-owl:Place* is the class for Location;
- *dbpedia-owl:Organisation*, *dbpedia-owl:Company*, and *umbel:Organization* are the classes for Organizations;
- *dbpedia-owl:ProgrammingLanguage*, *umbel:SoftwareObject*, *dbpedia-owl:Film*, *dbpedia-owl:TelevisionShow*, *dbpedia-owl:Award*, and *dbpedia-owl:SportsEvent*, are the classes for the Miscellaneous type.

Therefore, for each DBpedia resource we obtain its class from the ontology and classify it according to the challenge types.

However, many DBpedia resources are not classified in the ontology. For those resources we infer its type from certain properties which are characteristic of the type. For instance, from a triple

```
<subject> dbpedia-owl:birthPlace <object>
```

we can infer that object is a location given that it is the birth place of the subject described in the triple. The same rationale can be used with predicates such as *dbpedia-owl:hometown* and *dbpedia-owl:location*. Similarly, from a triple

```
<subject> dbpprop:mvp <object>
```

we can infer that the subject is an sport event since it has a most valuable player. Other predicates used for identifying sport events include *dbpprop:menDraw*, *dbpprop:teams*, *dbpprop:sport*, and *dbpprop:referee*.

Finally, in case we cannot identify the concept type using DBpedia, we use a list of concepts and their types which have been collected from the training data set. From this list we take the first type associated with that concept.

We have not included evaluation results in this extended abstract since the only available source of annotated data for the evaluation in this challenge was the the training data set (the test data set was not annotated). Given that our approach uses a list of concepts gathered from the training set is not fair to report evaluation results on this data set.

Acknowledgements

This research is partially supported by the Spanish Centre for the Development of Industrial Technology under the CENIT program, project CEN-20101037, "Social Media" (`http://www.cenitsocialmedia.es`).

References

1. Apache Software Foundation: Apache Lucene. `http://lucene.apache.org` (2013), [Online; accessed 23-May-2013]
2. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the Web of Data. Journal of Web Semantic 7(3), 154–165 (2009)
3. Codina, J., Atserias, J.: What is the text of a tweet? In: Proceedings of @NLP can u tag #user_generated_content?! via lrec-conf.org. ELRA, Istanbul, Turkey (May 2012)
4. DBpedia: DBpedia SPARQL endpoint. `http://dbpedia.org/sparql` (2013), [Online; accessed 23-May-2013]
5. García-Silva, A., Cantador, I., Corcho, O.: Enabling folksonomies for knowledge extraction: A semantic grounding approach. International Journal on Semantic Web and Information Systems (IJSWIS) 8(3), 24–41 (2012)
6. Kaufmann, M., Jugal, K.: Syntactic normalization of twitter messages. In: Proceedings of the International Conference on Natural Language Processing (ICON-2010) (2010)
7. Padró, L., Stanilovsky, E.: Freeling 3.0: Towards wider multilinguality. In: Proceedings of the Language Resources and Evaluation Conference (LREC 2012). ELRA, Istanbul, Turkey (May 2012)
8. Sanfilippo, S.: Redis. `http://redis.io` (2009), [Online; accessed 23-May-2013]
9. Wikipedia: Wikipedia:Database download. `http://en.wikipedia.org/wiki/Wikipedia:Database_download` (2013), [Online; accessed 23-May-2013]