University of Amsterdam at TAC 2011: English slot filling task

Roderik Lagerweij, Marc Bron, Christof Monz, Maarten de Rijke
ISLA, University of Amsterdam
Science Park 904, 1098 XH Amsterdam, The Netherlands
roderik.lagerweij@student.uva.nl, {m.m.bron, c.monz, derijke}@uva.nl

Abstract

We describe our participation in the Knowledge Base Population (KBP) English slot filling track of TAC. Our system is based on a distant supervision approach, where training instances are created by heuristically matching DBpedia relation tuples with sentences. Our official submission, which uses web snippets returned by search engines to collect positive instances, results in a performance comparable to the median. Based on an intial analysis we find a number of shortcomings with our official runs. After the TAC evaluation, we submit an unofficial run which matches relation tuples with sentences appearing in Wikipedia pages. Our proposed solutions implemented in our unofficial run did not further improve performance.

1 Introduction

In our first year at TAC we focus on the Knowledge Base Population (KBP) English slot filling task. Our approach is based on learning relation extractors using a distant supervision approach that is similar to those proposed by Mintz et al. [3] and Surdeanu et al. [5]. A distant supervision starts out by extracting relation triples from a knowledge base, then heuristically matches these triples with sentences to generate training instances, and finally trains a statistical model on these instances.

We divide the description of our system into two steps. First, we describe our distant supervision method for collecting training instances and the feature set used. Then, we describe how documents are retrieved during evaluation, slot candidates are extracted, classified and reranked. Based on an intitial analysis of our results we propose a number of adaptations to our system.

2 Training

This section describes our distant supervision approach to training relation extractors.

2.1 Extracting and Mapping Triples from DBpedia

To obtain training examples for our distant supervision approach, we use the DBpedia dump based on Wikipedia dumps generated in March 2010. Extractions take the form of a tuple $t = (e_i, r_{ij}, e_j)$, where e_i and e_j are strings that denote entities, and r_{ij} denotes a relationship between these entities. As many DBpedia relations do not have a one-to-one mapping with slots from the KBP task, a custom mapping is created which maps DBpedia infoboxes to either zero, one or more KBP slots. For some slots, additional processing of the values are required, e.g. for the slot *per:age* we compute the age based on the date of birth of an entity.

2.2 Collecting Positive and Negative Instances

Mintz et al. [3] heuristically match training instances with Wikipedia pages. Wikipedia pages are generally of high quality, yielding high quality training instances, but with limited variety. We explore the use of a search engine to collect training instances with possibly more variety. For each tuple t that results from the mapping, we query a search engine with the entities e_i and e_j and collect the snippets from the first 10 results. For instance, for a tuple *t = (Barack Obama, per:spouse, Michelle Obama)*, we query a search

engine with *"Barack Obama" AND "Michelle Obama"*, and collect the 10 first snippets. If there is an exact match of both the entity and the relation value in this snippet, the text in between these values is extracted as regarded as a positive training example. Negative training instances are accommodated for by randomly sampling positive training instances from other relations. The resulting dataset is balanced such that the number of positive instances equals the number of negative instances.

2.3 Training Slot Extractors

For each relation a binary Support Vector Machine [2] classifier is trained on the collected dataset. Only lexical features are used as features for the classifier. From each instance stemmed bi-grams are extracted, and treated as a bag-of-words, losing any sequential information. Feature selection is performed to compensate for an excessive number of features. As we wish to retain features that are characteristic for a particular slot, the selection of features is based on the TF $\cdot$ IDF measure for a feature f of slot s:

$$TF \cdot IDF(f, s) = TF(f, s) \log \frac{|D|}{DF(f)} \quad (1)$$

$|D|$ is the total number of training instances, $TF(f)$ is the number of training instances of s where f appears, and $DF(f)$ is the total number of training instances in which f appears. Features that have high frequency for a particular relation but low frequency for all relations reach high TF/IDF scores. For each relation, we retain the 500 highest scoring features.

3 Evaluation

During evaluation, three steps are performed. (a) documents and sentences that are relevant to the query entity are retrieved; (b) potential slot candidates are extracted and classified; and (c) fillers are heuristically reranked and filtered. The next section provides descriptions of each of these steps.

3.1 Query Processing and Document Retrieval

Documents that include a mention of the query entity are retrieved. To increase recall, variations on the original query entity are generated. Variations include abbreviations, filtered by stop words, and some special cases where words have multiple spelling variations, e.g., *organization* and *organisation*. As for some queries the resulting set of documents can be very large, e.g. *Samsung*, the number of documents is limited to 250. Documents are ranked by the number of occurrences of the query entity and its variations. From the training set we retrieve 76% of the relevant documents by this method.

From the resulting documents individual sentences are recognized by using the Natural Language Toolkit (NLTK)[1]. Query entity mentions are detected by the simple heuristic that if one or more terms match the original query entity q_e, we assume it to refer to the query entity. Sentences that include an entity mention are assumed to be relevant.

3.2 Candidate Extraction and Classification

From relevant sentences candidate slot fillers are extracted. Candidates referring to entities are extracted by the Stanford Named Entity Recognition system [1]. For other slots where the target is not an entity, either manually compiled lists or rules are used, e.g. for the *per:religion* and *org:alternate_names* slots.

To reduce the number of extraction candidates, simple trigger-word detection is performed. As trigger-words, the 50 highest scoring features based on Equation 1 are used. Only sentences in which at least one of these features appears are considered.

All extraction candidates are classified by the binary SVM classifiers as either being a correct slot filler or an incorrect slot filler. Candidates yielding a score higher than 0 are retained. For two slots, *per:alternate_names* and *org:alternate_names*, the classification system is bypassed and all candidates are given as answers. The slot *per:title* we skip in its entirety, as it generates many false positives.

3.3 Candidate Reranking and Filtering

Classification yields a large pool of candidate entities. We boost scores of candidates that are more likely to refer to the query entity by the following weighing method:

[1] www.nltk.org

$$S_{c-weighed} = \frac{1}{d(e,c)} S_c \quad (2)$$

Here, S_c is the score of candidate slot filler and $d(c, e)$ is the distance in words between candidate c and query entity mention e.

To prune over generated candidates, the following three heuristics are used:

1. For *single* slots, the candidate receiving the highest score is given as answer.

2. For *list* slots, the set of answers is ordered by descending score, and if the drop in score is 30% or more, no more answers are generated.

3. For *list* slots, the maximum of candidates is limited to 3.

3.4 Geographic Disambiguation

Of the 42 slots, 12 slots take only geographic entities as slot fillers. Given only lexical clues, it is often difficult to disambiguate a geographic entity to the correct geographic type, which can either be a $city$, a $stateorprovince$, or a $country$. We investigate the use of Bing Maps service to disambiguate entities. During training, the service is queried with all candidates that are recognized as geographic entities by the NER system. Given a query, Bing Maps returns potential locations the query may refer to. Each potential location includes a level of confidence and the type of geographic location it refers to. All low or $medium$ confidence locations are ignored. From the $high$ confidence locations a list of geographic types is build containing zero, one or multiple types (e.g. New York can refer to the city or to the state). Extracted candidates are only considered if the list contains a type that matches the required geographical type of a slot.

During training of our system all mappings of geographic candidates to types are cached. During evaluation, if the type of candidate is available in this cache, only those geographic types are considered. At the point of the official evaluation, the database contains about 200k mappings.

Note that we use the same disambiguation method for mapping DBpedia properties to those KBP slots that require a geographic entity as slot filler. For instance, the DBpedia property *location* contains cities, states and countries where headquarters of organizations are located. To disambiguate instances of these properties to one of the correct KBP slots *org:city_of_headquarters*, *org:stateorprovince_of_headquarters* or *org:country_of_headquarters*, we use the same Bing querying method.

4 Post Submission System

The performance of our official submission was disappointing. In this section we analyse shortcomings of the system and propose a number of adaptations.

4.1 Document and Sentence Recall

A shortcoming of our original document retrieval module is its inability to correctly rank documents. We switch to Indri [4], which assigns a probability value to each document based on a language modeling approach. The original query entity is used as search query. The 75 highest scored documents are collected. Document recall is 74%, which is similar to the original retrieval method, but with less documents used.

We apply a simple heuristic as an attempt to resolve co references and thereby increase recall of relevant sentences. If a sentence either starts with *He* or *She*, and the previous sentence contains only a single entity of type *person* which is a query entity mention, we replace this word by this previous entity mention.

4.2 Training

Manual inspection of the training instances produced by the distant supervision method shows that this dataset is very noisy, and many positive examples are actually not positive instances of that particular relation. One explanation is that the method of collecting instances by querying a search engine is slow, averaging only around 300 positive instances for each individual slot when entering the official evaluation. Although many positive examples are included, the variety of instances is high. We suspect that the combination of high variety and limited number of training instances results in a low accuracy classifier.

Instead of searching Web snippets, for the post-submission system a Wikipedia page dump from August 2011 is searched for occurrences of entity-relation pairs. For each entity appearing in the training dataset generated from the DBpedia dump, we search the corresponding Wikipedia

page for mentions of its slot fillers. We apply the following heuristics to improve the quality of positive instances:

- From the set of slot fillers for an entity, we filter out fillers that appear more than once, e.g., if both the *country_of_birth* and *country_of_death* slots have the filler *USA*, no positive instances are generated for these slots.
- When a slot filler appears more than once in a document, it is ignored.
- Entities for which the number of slot fillers violate the KBP constraints, e.g., having more than one *per:city_of_birth*, are ignored.
- Sentences are only considered when a word unique to the query entity appears in it. For instance, a sentence with a match for *Michelle Obama* is only considered a positive instance for *per:spouse* when it also contains *Barack*.
- If sentences start with *He* or *She*, this word is replaced by the title of the page.

Sentences that meet these requirements and contain a correct slot filler are used as positive examples. This results in a total of 230k positive instances. Negative instances are generated equally to our original method, by sampling positive instances from other slots. In contrast with our original method, the dataset is balanced such that there are twice as many negative instances as positive instances.

4.3 Candidate Classification

We augment our feature set inspired by Mintz et al. [3]. Given two entities e_1 and e_2 appearing in a sentence, where one is the query entity and the other is a candidate slot filler, and $e_1 < e_2$, the following features are used:

- The words and part of speech tags of the 2 words appearing before e_1 and 2 words appearing after e_2.
- The words and part of speech tags, the number of words and the number of entities appearing between e_1 and e_2, with a maximum of 4 tokens after e_1 and 4 tokens before e_2.
- The words, part of speech tags, named entity label and length of the candidate slot filler.

Table 1: Results of our official and unofficial distant supervision systems on the 2011 evaluation queries

System	Precision	Recall	F-measure
UvA_{50}	12.7	8.8	10.4
UvA_{250}	12.3	9.0	10.4
UvA_{post}	10.3	9.5	9.9
Human	86.1	72.6	78.8
Top System	35.0	25.5	29.5
Median	10.3	16.5	12.7

- The first word of the sentence.
- Words in the sentence of which the part of speech tag starts with VB, e.g. VBD, VGB, etc.

We drop the reranking and filtering method, and output a static number of candidates for *list* slots. Following Surdeanu et al. [5] we only output the single best candidate based on the score produced by Equation 2.

5 Slot Filling Experiments and Results

We submitted two official runs to the slot filling task. UvA_{250} considers 250 documents while UvA_{50} considers only 50 documents. Additionally, we submit an unofficial run, which is the post-submission system described in Section 4. We report our scores in Table 1, and those of the top performing system, a time limited run by humans and the median performance.

Our official submission achieves a performance that is comparable to the median. Our post submission system did not further improve performance. Based on an analysis we suspect the following issues to be the most important:

- For certain slots, such as *org:top_members/employees*, our mapping of DBpedia infobox attributes to KBP slots is incorrect, resulting in noisy training instances and an inaccurate classifier.
- The query entity mentions that are extracted include many false positives, where in fact a different entity is referred to. We belief that by explicitly addressing the entity linking task, slot filling scores can be substantially improved.
- Document recall is relatively low, which we suspect to be a result of insufficient query processing.

- Our heuristic for co reference resolution has low recall, resulting in relevant sentences being excluded when not containing the query entity.

6 Conclusion

In this years participation of the Knowledge Base Population task we experimented with a distant supervision approach, which creates training instances by heuristically matching DBpedia relation tuples with sentences. Our official submission, which uses web snippets returned by search engines to collect positive instances, results in a performance comparable to the median. Based on an intial analysis we find a number of shortcomings with our official runs. After the TAC evaluation, we submit an unofficial run which matches relation tuples with sentences appearing in Wikipedia pages. Our proposed solutions implemented in our unofficial run did not further improve performance.

7 Acknowledgements

This research was supported by the European Union's ICT Policy Support Programme as part of the Competitiveness and Innovation Framework Programme, CIP ICT-PSP under grant agreement nr 250430 (GALATEAS), by the PROMISE Network of Excellence co-funded by the 7th Framework Programme of the European Commission, grant agreement no. 258191, by the DuOMAn project carried out within the STEVIN programme which is funded by the Dutch and Flemish Governments under project nr STE-09-12, by the Netherlands Organisation for Scientific Research (NWO) under project nrs 612.066.512, 612.061.814, 612.061.815, 640.004.802, 380-70-011 and by the Center for Creation, Content and Technology (CCCT).

References

[1] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. ACL '05, pages 363–370, 2005.

[2] Thorsten Joachims. *Making large-scale support vector machine learning practical*. 1999.

[3] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. ACL '09, 2009.

[4] Trevor Strohman, Donald Metzler, Howard Turtle, and W. Bruce Croft. Indri: a language-model based search engine for complex queries. Technical report, in Proceedings of the International Conference on Intelligent Analysis, 2005.

[5] Mihai Surdeanu, David McClosky, Julie Tibshirani, John Bauer, Angel X. Chang, Valentin I. Spitkovsky, and Christopher D. Manning. A simple distant supervision approach for the tac-kbp slot filling task. In *TAC'10*, 2010.