WebSAIL Wikifier: English Entity Linking at TAC 2013

Thanapon Noraset **Chandra Bhagavatula** **Yi Yang** **Doug Downey**
Department of Electrical Engineering
Computer Sciene
Northwestern University
Evanston, IL 60208
{nor|csbhagav|yiyang}@u.northwestern.edu, ddowney@eecs.northwestern.edu

Abstract

In this paper, we report on our participation in the English Entity Linking task at TAC 2013.

We present the WebSAIL Wikifier system, an entity disambiguation system that links textual mentions to their referent entities in Wikipedia. The system uses a supervised machine learning approach and a string-matching clustering method, and scores 58.1% B^3+ F1 on the TAC 2013 test set.

1 Introduction

Entity linking is the task of identifying and linking mentions in text to entities in a reference knowledge base. In Text Analysis Conference (TAC), a set of documents and mentions are given as input, and the goal is to map each mention to its referant entity in the given KB (a subset of English Wikipedia 2009). In the relatively common case that the KB does not contain the appropriate entity, the system should output 'NIL.' Further, as a substantial additional challenge the system must cluster together all 'NIL' mentions that refer to the same entity.

In this paper we report on our first attempt participating in the English Entity Linking track at TAC. Our System, the WebSAIL Wikifier, is designed for Wikification task, a similar task to the TAC task that does not including clustering the NIL mentions. In this paper, we describe the system, and provide performance measurements and preliminary error analysis.

2 Data Preprocessing

We preprocessed several resources to generate *candidates*, i.e. potential target entities for each mention, and features for use in our machine learner. This section describes the resources and the processing steps, and our techniques to scalably store and retrieve the required data.

2.1 Wikipedia Resources

First we preprocessed a Wikipedia dump from May, 2012[1] including articles, links, and redirects. When processing Wikipedia, we used JWPL Wikipedia Parser (Zesch et al., 2008) to parse all English and Simple English Wikipedia articles. We stored the plain text of the articles in an Apache Lucene index, and the links in MySQL.

2.2 Knowledgebase Entities

We extracted hyperlinks to Wikipedia from two resources: a copy of English Wikipedia parse as described above, and the Google Cross-Lingual Dictionary for English Wikipedia Concepts (Spitkovsky and Chang, 2012). We used the extracted hyperlinks to construct a trie of names that maps anchor text strings to their referent Wikipedia entities. We also resolved any redirects, and mapped all the entities to titles from English Wikipedia of May, 2012. Since TAC has its own entity set (with IDs distinct from Wikipedia), we created a map from TAC entity IDs to our link trie entities using simple string matching. This process was unable to find a correspondence in

[1] http://dumps.wikimedia.org

our knowledge base for a total of 12,562 TAC entities.

2.3 Sketches for Disambiguation

In addition to the trie, we extracted semantic information from Wikipedia to use during disambiguation. Specifically, we extracted *article summaries* of each entity page, in the form of a bag-of-words model. We adopt a count-min sketch (Matusevych et al., 2012) for the model to enable scalable querying and retrieval of approximate frequencies for all terms. We also computed *context summary* sketches of the text surrounding hyperlinks to each entity in the Wikipedia resource. In this task, the context window was set to be 25 tokens to the left and right of the hyperlinks, and the stop words and symbols were removed.

3 Methods

Given a query (a string mention and its containing document), our system generates candidate entity disambiguations for the query, extracts features for the candidates, and then ranks the candidates to select the correct entity.

3.1 Candidate Generation

When processing a query, we first attempt to locate the mention in our trie, and identify which entities the mention string referred to in our resources. These entities become our candidate entities for disambiguation. Our trie may return many candidate entities for common query mentions, so we retain only the top 100 candidate entities based on the maximum reference probability between the two resources.

We also tried a simple "query expansion" technique in some runs. We implemented a simple pattern matching heuristic to find words whose first character was capitalized. This query expansion is used to identify mentions where only last names are used to refer to a person. If the system can find a match, the name will be expanded to its full name. For example, a query name "Williams" could be expanded into "Serena Williams" if we find the full name in the document. This heuristic improved candidate generation precision, but it could also expand queries that were not a person name and sometimes gave us undesired query string. For example, "Swedin" was expanded to "Not Swedin".

3.2 Candidate Ranking

The resulting set of candidates from candidate generation was ranked using a trained machine learning model, described below. We selected the top-ranked candidate as the answer to the query (and assigned it a confidence value, as required by TAC, of 1.0). Our ranker utilized the features listed in Table 1. Prior probability features are a popular baseline used by many systems. Ratinov et al.(2011) showed that the bag of words cosine similarity between the query document/context and a candidate article/context can be useful for disambiguation. We applied this using term frequencies from the sketches built in our preprocessing step.

To rank the candidates, we experimented with many supervised machine learning approaches. One approach treated the ranking problem as a regression problem (Fuhr, 1989), where a correct candidate has higher output value. We used linear regression implemented in Weka (Hall et al., 2009). On the other hand, we also experimented with an existing ranker (Coordinate Ascent (Metzler and Croft, 2007), implemented in RankLib[2]) to rank candidates. In later experiments (reported in this paper but not submitted to the TAC competition), we used the LibSVM classifier (Chang and Lin, 2011) to rank the candidates. The instances for the SVM classifier are formed by taking differences of the feature values between each incorrect candidate and the correct candidate. We then use the learned model to compare two candidates (Joachims, 2002).

3.3 NIL Clustering

We implemented a simple method for clustering NIL entities. There were two cases for which the system would return an output of NIL for a query: when its top-ranked entity was not in TAC KB, and when the candidate generation failed to return candidates. Since our system contained more entities than there were in the TAC KB, it was possible that the selected candidate was not in the TAC KB. In this case, the system returns NIL clustered by the missing (top ranked) entity. For a query for which the system can-

[2] http://www.cs.umass.edu/~vdang/ranklib.html

Feature	Explanation
Prior	
internalPrior	$P(Entity \mid String)$ inside Wikipedia
externalPrior	$P(Entity \mid String)$ outside Wikipedia
internalPriorNC	internalPrior not normalized letter case
externalPriorNC	externalPrior not normalized letter case
probabilityRank	A ranked order of candidate concept by a combination of the first 4 features
Context (using Sketches)	
text2TextSim	Cosine similarity between article concept and input document
text2ContextSim	Cosine similarity between article concept and query context
context2TextSim	Cosine similarity between context concept and input document
context2ContextSim	Cosine similarity between context concept and query context
Top-Context (using Sketches)	
topTermText2TextSim	Same as text2TextSim, using a fixed set of top TF-IDF terms
topTermText2ContextSim	Same as text2ContextSim, using a fixed set of top TF-IDF terms
topTermContext2TextSim	Same as context2TextSim, using a fixed set of top TF-IDF terms
topTermContext2ContextSim	Same as context2ContextSim, using a fixed set of top TF-IDF terms
Top-Context (using Sketches)	
titleMatch	A boolean value whether a surface form matches a concept title
lastnameMatch	A boolean value whether a surface form matches a last name from dbpedia
exactMatch	A boolean value whether a surface form matches a surface in the Trie

Table 1: List of features used by rankers.

not generate a candidate, NILs are simply clustered using the query name.

4 Experiments

We performed experiments on TAC English evaluating queries from 2010 and 2011 to select machine learning models to use for our 2013 competition submissions. Each submitted run differed in terms of the selected model, and whether query expansion was employed. After receiving the results of the competition submissions, we also performed additional experiments on TAC 2013 queries as described below.

4.1 Training Data

For training, we used randomly selected existing links from English Wikipedia and Simple English Wikipedia, as of May 2012, and queries from TAC Training data from 2010 and 2011. The links are excluded from ones that we used to build the context summary. In total, we used around 200,000 queries. We transformed links and queries into a set of machine learning instances (the number of instances depends on the matching candidate concepts and machine learning model).

4.2 Results

We tested the system on TAC Evaluating Queries 2010 and 2011. We selected 5 results from different ranker models to submit, and our best model was the linear regression ranker using adaptation-2 (Church, 2000) for term weighting instead of IDF, with last name query expansion turned on. the results (B^3+) from the model are shown in Table 2. We excluded queries from TAC Evaluating Queries 2010 and 2011 to train a system for "Evaluating 2010" and "Evaluating 2011"

Query Set	Performance		
	Precision	Recall	F1
Evaluating 2010	0.77	0.82	0.79
Evaluating 2011	0.7	0.72	0.71
Evaluating 2013	0.571	0.362	0.443
Evaluating 2013*	0.718	0.540	0.617

Table 2: System performance.

After the competition, we performed additional experiments using the Logistic Regression classifier with IDF for term weighting and last name query expansion on Evaluating 2013 queries (denoted as "Evaluating 2013*" in Table 2. The new classifier was trained on 50,000 random selected links from English Wikipedia as of April 2013, and improved performance significantly.

4.3 Preliminary Analysis

We provide our preliminary analysis of the performance in this section. Since our system didn't have a proper way to cluster NIL, we divided the analysis into two parts: 1183 Non-NIL queries and 1007 NIL queries. For Non-NIL queries, the system performed relatively well. The candidate generation returned a set of candidates with correct entity for 875 queries from 1183 queries (74)%), and the ranker linked 851 (97%) of the mentions correctly (72% in total).

For NIL queries, the candidate generation could not find any candidates for 211 queries (21%), so the system grouped the queries based on the mention strings. Furthermore, the ranker selected a Wikipedia entity that was not in TAC KB for 639 queries (63%), and grouped these based on the missing entities. While the accuracy of NIL queries was around 84%, our system suffered from this naive clustering of NILs. An evidence for this is that there are only 399 NIL clusters in TAC 2013, but our system generated 742 NIL clusters.

5 Conclusion

We presented the application of the WebSAIL Wikifier to the TAC 2013 English entity linking task. The system used multiple resources to generate candidate entities, and a supervised ranker to select the correct entity. While the system performed relatively well on TAC 2010 and 2011 evaluation queries, it performed poorly on TAC 2013 data.

Our preliminary analysis shows that both candidate generating and ranking needs to be improved. Possibilities include using a fuzzy matching mechanism for candidate entity retrieval from the trie instead of exact string matching, and introducing additional features for ranking. Most importantly, we require a more sophisticated method for recognizing and clustering the NIL mentions.

Acknowledgments

This work was supported in part by NSF Grants IIS-1016754 and IIS-1065397, and DARPA contract D11AP00268.

References

Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Kenneth W Church. 2000. Empirical estimates of adaptation: the chance of two noriegas is closer to p/2 than p 2. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pages 180–186. Association for Computational Linguistics.

Norbert Fuhr. 1989. Optimum polynomial retrieval functions based on the probability ranking principle. *ACM Trans. Inf. Syst.*, 7(3):183–204, July.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, pages 133–142, New York, NY, USA. ACM.

Sergiy Matusevych, Alex J. Smola, and Amr Ahmed. 2012. Hokusai - sketching streams in real time. *CoRR*, abs/1210.4891.

Donald Metzler and W Bruce Croft. 2007. Linear feature-based models for information retrieval. *Information Retrieval*, 10(3):257–274.

L. Ratinov, D. Roth, D. Downey, and M. Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *ACL*.

Valentin I. Spitkovsky and Angel X. Chang. 2012. A cross-lingual dictionary for english wikipedia concepts. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uur Doan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may. European Language Resources Association (ELRA).

Torsten Zesch, Christof Müller, and Iryna Gurevych. 2008. Extracting lexical semantic knowledge from wikipedia and wiktionary. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco, May. electronic proceedings.