IIIT Hyderabad at TAC 2012

Vasudeva Varma
IIIT, Hyderabad
vv@iiit.ac.in

Bhaskar Ghosh
IIIT, Hyderabad
bhaskarjyoti.ghosh@research.iiit.ac.in

Mohan Soundararajan IIIT, Hyderabad
mohan.s@research.iiit.ac.in

Deepti Aggarwal
IIIT, Hyderabad
deepti.aggarwal@research.iiit.ac.in

Priya Radhakrishnan
IIIT, Hyderabad
priya.r@students.iiit.ac.in

Abstract

In this paper, we report our participation in Knowledge Base Population at TAC 2012. We adopted an Information Retrieval based approach for the Entity Linking and Slot Filling tasks. In Entity Linking we identify potential nodes from the Knowledge Base and then identify the mapping node using tf-idf similarity. We achieved very good performance in the Entity Linking task. For Slot Filling task we identify documents from the document collection that might contain attribute information. We extract the attribute information using a rule based approach. Our rule based approach hasnt performed up to the mark.

1 Introduction

Knowledge Base Population (KBP)

The rise of web 2.0 technology has provided a platform for content generators on the web through blogs, forums etc. This has lead to information overload on the web and users face difculty in nding the information they are looking for. Structured Knowledge Bases (KB) like Wikipedia, act as a rich source of information. The problem with Knowledge Bases like Wikipedia is that they have to be created and maintained manually. Manual effort is not only time consuming but can also lead to erroneous values fed, information being outdated and other inconsistencies. Automatically updating Knowledge Bases from news source, where the latest information is available, is a possible solution. The Knowledge Base Population (KBP) track at TAC-2010 proposes the problem of automatically updating Knowledge Bases from textual content. The task has been broken down into 2 fundamental sub-problems:

Entity Linking

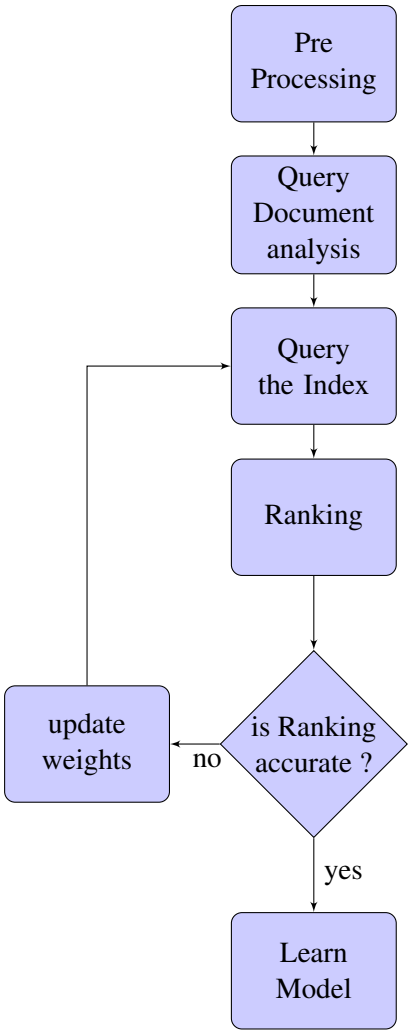
The task of Entity Linking is to determine for each query entity, which node is being referred to in the KB or if the query entity is not present in the KB. The query consists of a named entity and an associated document from the Document Collection (DC) using which we need to link the named entity to its corresponding node in the KB, if any. The purpose of the associated document is to provide context that might be useful in linking it. We need to return the entity node if the query entity is present in the KB else NIL.

Slot Filling

Slot Filling involves mining information about entities from textual content. Slot Filling shares similarities with traditional Information Extraction and Question Answering tasks. Slot Filling involves learning a predened set of relationships and attributes for target entities based on the documents in the Document Collection. A query in the Slot Filling task contain a name-string, doc-id, and an optional list of slots to ignore. As in the Entity Linking task the doc-id is intended to provide the context for the entity. We need to return the slot name and slot value along with the document in which it occurs.

2 Approach

We have broken down the Entity Linking task into four separate modules.



Pre Processing

During the prepossessing step, we build a Knowledge Index of the given resources by indexing. We have used Lucene (Bialecki et al,2012) to Index the Knowledge Base (KB), Knowledge repository (KR) (wikipedia) and document dataset. Lucene is a full-featured, open source text search engine.We also indexed the publicly available knowledge resource Google's text-entity map (Spitkovsky et al,2012).

Query document analysis

Processing the query word and query document d (get local context of w) In order to decrypt and understand the anaphora references we try to resolve them in the reference document. Co-reference resolution is done with the help Stanford NLP parser(Finkel et al,2005). Then we extract the local context of the query word w from document d by considering the proximal sentences and then filter the stop words (based on tf-idf). We proceed to apply Named Entity Recognizer(NER) on the filtered sentences to arrive at the relative weights of the query and context word or terms. The query word is analyzed to understand its Named Entity(NE) type, with the support of the document context, using the Stanford NER. The NE type of the query word is later used in Ranking the results. From the co-reference resolved and NE type identified text, we extract key phrases using key phrase extractor(Niraj Kumar,2010)

Query the Index

Query construction is based on the query word, the extracted context words w along with the NE type t, Key phrase weights k and proximity weights p A Lucene query Q is constructed such that Q=w, t, p, k This query is then fired on the resources in the order 1. Query KB index with Q, get all KB entries for w 2. Query KR, get top documents

Ranking

The retrieved documents are ranked using a ranking scheme. From the set of candidate nodes that are retrieved, we rank them using Tf-idf. The best ranked node is returned as the mapping node for our query entity. Tf-idf is the most popular weighting function used in the eld of Information Retrieval. Given a query Q=t1,t2...tn and a document D=w1,w2,...,wn, the similarity between them is given by

$$Similarity(D,Q) = \sum_{t_i inQ} t_f(t_i, D) idf(t_i)$$

where term frequency (tf) is simply the number of times the term ti appears in the document D. The

inverse document frequency (idf) is a measure of the general importance of the term. Along with these, the phrasal proximity is given priority with a specified slop value. Further selected fields in the KB, KR (say, infobox fields, category fields, and specific fields like name, country of residence, etc.,) are given more significance. The match from these fields are weighted higher to pop them up in the search results.

We further re-rank the ranked documents according to the following order with decreasing order of

their priority.

i) query word, w is mandatory for each document. More weightage is given to title words of query document whenever available. ii) increase weightage to words of same NE type as query words iii) weightage to contextual words

To link the word to KB node, we query the KB index and retrieve top m KB entries having highest tf from top n documents.

3 Experiment & Results

We have submitted four runs for the Entity Linking task.Summary of runs is in Table 1. The description of each run is provided below

Run-1

For this run we used the knowledge repository, Stanford NER for identifying the candidate list documents. Using these variations we did a phrase search on the titles of Wikipedia and KB nodes. All the nodes whose titles exactly match these phrases or match these phrases and have an extra token are considered as candidate nodes. We then rank these nodes using Tf-idf and re-rank using pseudo relevance feedback. We use title and do query expansion. The nal relevance score for each node is a linear combination of the rank and re-rank scores. This does not use the web.

Run-2

Without nil classification (better weights, lucene ranking + custom ranking) For this run we used the knowledge repository built and Stanford NER for identifying candidate list documents. Once these candidate documents are identied we rank them using Lucene ranking. We also re-rank these documents using query expansion. This is custom ranking. This does not use the web.

Run-3

This run is similar to above run, except that we do not do re-ranking(custom ranking). Here we rely purely on the Lucene ranking. This does not use the web.

Run-4

This is the base system. Here we search on the titles of Wikipedia and KB to identify candidate nodes. All those nodes whose titles contain the variation tokens or have token variation and an extra token are considered as candidate nodes. The ranking is Lucene default. This does not use the web.

4 Conclusion

Entity Linking can be viewed as an Retrieval problem. The basic system devised itself is capable of linking a large number of entities properly. The scores seem to be less, for the fact that NIL entities are not handled yet in our approach.

Future Work

We are in the process of using Google's cross wiki (text-entity map) to aid the ranking module. Further NIL entities are to be recognised to boost the precision scores.

References

Andrzej Biaecki, Robert Muir, Grant Ingersoll 2012. *Apache Lucene 4*, SIGIR 2012 Workshop on Open Source Information RetrievaSIGIR 2012 Workshop on Open Source Information Retrieval.

Valentin I. Spitkovsky and Angel X. Chang. 2012 A Cross-Lingual Dictionary for English Wikipedia Concepts,Proceedings of the Eighth International Conference on Language Resources and Evaluation.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning 1983. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics

Niraj kumar,Kannan Srinathan,Vasudeva Varma 1981. Alternation. Key Fact Extraction from Newswire Articles by Exploiting Local features based weighting and Interaction of sentences

Table 1: Runs' Comparison

Run Id	Run Description	Micro Average Score	Bˆ3+ F1 score
Run-1	base system with wiki text, better weights, lucene ranking and custom ranking	13.1	Bˆ3+ F1 (All – 2226 queries) : 0.131 Bˆ3+ F1 (in KB – 1177 queries) : 0.245 Bˆ3+ F1 (not in KB – 1049 queries) : 0.003 Bˆ3+ F1 (NW docs – 1471 queries) : 0.130 Bˆ3+ F1 (WB docs – 755 queries) : 0.133 Bˆ3+ F1 (PER – 918 queries) : 0.137 Bˆ3+ F1 (ORG – 706 queries) : 0.093 Bˆ3+ F1 (GPE – 602 queries) : 0.162
Run-2	base system with smarter weights, lucene ranking and custom ranking	13.0	Bˆ3+ F1 (All – 2226 queries) : 0.130 Bˆ3+ F1 (in KB – 1177 queries) : 0.243 Bˆ3+ F1 (not in KB – 1049 queries) : 0.003 Bˆ3+ F1 (NW docs – 1471 queries) : 0.130 Bˆ3+ F1 (WB docs – 755 queries) : 0.130 Bˆ3+ F1 (PER – 918 queries) : 0.139 Bˆ3+ F1 (ORG – 706 queries) : 0.089 Bˆ3+ F1 (GPE – 602 queries) : 0.160
Run-3	base system with smarter weights and lucene ranking	5.5	Bˆ3+ F1 (All – 2226 queries) : 0.055 Bˆ3+ F1 (in KB – 1177 queries) : 0.101 Bˆ3+ F1 (not in KB – 1049 queries) : 0.003 Bˆ3+ F1 (NW docs – 1471 queries) : 0.067 Bˆ3+ F1 (WB docs – 755 queries) : 0.031 Bˆ3+ F1 (PER – 918 queries) : 0.064 Bˆ3+ F1 (ORG – 706 queries) : 0.039 Bˆ3+ F1 (GPE – 602 queries) : 0.056
Run-4	base system	6.1	Bˆ3+ F1 (All – 2226 queries) : 0.061 Bˆ3+ F1 (in KB – 1177 queries) : 0.114 Bˆ3+ F1 (not in KB – 1049 queries) : 0.003 Bˆ3+ F1 (NW docs – 1471 queries) : 0.071 Bˆ3+ F1 (WB docs – 755 queries) : 0.043 Bˆ3+ F1 (PER – 918 queries) : 0.068 Bˆ3+ F1 (ORG – 706 queries) : 0.044 Bˆ3+ F1 (GPE – 602 queries) : 0.068