HLTCOE Efforts in Entity Linking at TAC KBP 2010

Paul McNamee
Human Language Technology Center of Excellence
The Johns Hopkins University
Baltimore, MD 21218, USA
paul.mcnamee@jhuapl.edu

Abstract

This report documents the HLTCOE submission to the 2010 Text Analysis Conference Knowledge Base Population Track Entity Linking task. This year we incorporated a number of engineering changes to simplify our 2009 system, and as a results the new software runs approximately 20 times faster than the previous version. We also eschewed use of the Internet entirely. Details of the HLTCOE prototype system's design and implementation are given with a preliminary analysis of the results.

1 Introduction

The TAC Knowledge Base Population track aims to encourage research that supports automatic mining of information about entities from unstructured texts and insertion of that information into a knowledge base (KB). An important component that is required for augmenting information about entities is grounding the entity mentions that are observed in text. The *Entity Linking* task tries to match a name mention that occurs in a document to a specific KB entry, or to determine that no appropriate entry presently exists.

The track made available a collection of information derived from an October 2008 snapshot of Wikipedia. Wikipedia pages that contained semi-structured 'infoboxes,' tables of attributes about the page's subject, were extracted along with the set of *slots* contained in the table.

Applications for linkage of personal entities include gathering of census data, linking patient health records from separate hospitalizations, mail delivery, personal credit files and prevention of identity crimes, law enforcement (*e.g.,* serving arrest warrants), and national security (*e.g.,* border control and terrorist watch lists). Winkler (2006) provides an overview of entity linking based on his research at the U.S. Census Bureau.

Exact string matching is not a viable approach for matching entities. False positives occur because distinct entities share a name in common. False negatives occur because different names can refer to the same entity (*e.g.,* nicknames, aliases, legal name change) and because name variants can be non-trivial to match due to acronyms, abbreviations, omission of name fragments, and foreign translations and transliterations.

Our work in 2010 focused on improving our 2009 system, principally to ensure that our approach scales to large data sets. We continued to rely on supervised machine learning, however, we substantially reduced the size of our feature set and concentrated on efficiently computable features which do not rely on connection to the live Internet.

2 Technical Approach

The number of KB entries can be quite large, so we break down our processing into two phases. In the first phase we seek to identify reasonable candidate entities and hope to obtain high recall while reducing the number of possible entities by several orders of magnitude. Then, after the initial candidate identification phase, we rank each candidate according to the likelihood that it is the KB entry corresponding to the textual query mention. Each of these phases is described in detail below.

2.1 Candidate Identification

We relied almost exclusively on a variety of name string comparisons to select candidates for further consideration. The only exception is that we would "rewrite" the query name if it appeared to be an acronym and if there was a named-entity in the source document that matched the acroynym (*e.g., UAW* could be rewritten as *United Auto Workers*). This idea was used at KBP 2009 by the QUANTA team. We discovered that this heuristic is highly effective on the KBP 2009 queries, which include many organizational acronyms.

The set of KB entities was richly indexed so that it was possible to efficiently determine which KB nodes share a word in common with the query, or are a reasonable approximate match. The original form of the KB entry name is retained, but most comparisons are based on a normalized form that removes case, punctuation, and appositive constructions (*e.g.,*'London, Ontario' and 'London (film)' are normalized to *london*) Five different operations contribute to building the candidate list:

1. Checking whether the query name (Q_{name}) is an exact match with the KB entry name (KB_{name})
2. Is Q_{name} an acronym that could match KB_{name} (*e.g.,* MSF for Médecins Sans Frontières)
3. Is Q_{name} is a known alias for KB_{name}. We compiled a large list of aliases from sources including Wikipedia redirects and stock ticker symbols.
4. KB_{name} shares a word in common with Q_{name}. But if this operation would add many entries, the entries are ranked and only the top $k_1 = 20$ choices are added to the candidate list
5. KB_{name} is a good approximate match with Q_{name} based on possessing character 4-grams in common. As with words, only the top $k_2 = 20$ choices are added to the candidate list

Steps 1-3 above are easily and efficiently computed using hashtables. For Step 4, entries sharing common words are ranked by summing the IDF values of the words shared. Thus sharing a rarer word like *malkovich* would be more indicative than simply sharing the word *john*. Finally, in Step 5 entries sharing 4-grams are ranked by the number of common 4-grams. This is intended to allow close, but inexact matches to be considered (*e.g., George W. Bush* and *George Bush*).

k_1	k_2	*Other NEs*	*Recall*
10	10	No	3661 (93.77%)
25	25	No	3746 (95.95%)
50	50	No	3789 (97.05%)
50	50	Yes	3869 (99.10%)

Table 1: Parameter settings for improving recall in candidate generation based on 3904 queries from KBP 2009.

We tested the recall of the above process using the 3904 queries from the KBP 2009 evaluation. In particular we examined different values of k_1 and k_2 (see Table 1). We also considered an aggressive method which combined the candidates produced from the query name with candidates produced from all other named-entities found in the provided query document. This method was highly effective as shown in the final row of Table 1, and enabled matching for difficult queries such as the metaphorical usage of 'Iron Lady' for Yulia Tymoshenko (KBP 2009 query EL1687). However, while this increases recall, it had the effect of lowering end-to-end accuracy in our system, so we did not apply this method for our submissions this year.

We found queries involving sports teams (*e.g., The Lions*) to be the most difficult to resolve as the reference is highly ambiguous. We did not try to customize the candidate identification phase based on the predicted entity type of the query; however, this seems like a reasonable technique to consider.

In 2009 we observed that submitting query names to Google and asking for search results from `en.wikipedia.org`, and then including as candidates those results which were present in the KBP knowledge base, reduced our residual recall error by about half. This year we did not exploit this technique.

2.2 Ranking Candidates

The second phase in our system is to score each viable candidate using supervised machine learning.

As in KBP 2009, we used a learning-to-rank framework and utilized the SVM^{rank} tool [1] to train a model for ranking candidates (Joachims, 2002). We use a linear kernel and set the slack parameter C to be 0.01 times the number of training examples. The cost function we used to optimize the algorithm is based on the number of steps required to elevate the correct candidate to rank 1.

Unique to our system is that we consider absence from the knowledge base to be a distinct candidate, the so-called `NIL` candidate. We integrate `NIL` prediction into the process by including features which are indicative of no other candidate being correct (see Section 2.3.6 below). Considering absence as a separate candidate simplifies the process of making predictions and also prevents the need to select a threshold or similar device to select between the top KB entry and the possibility that the queried entity is missing from the KB.

2.3 Features

We considered approximately 200 basic features in our 2009 system, and also considered combinations of features which increased the number of features to over 26,000. In 2010 we reduced the number of atomic features to 116 and did not combine features. In post-evaluation experiments on the KBP 2009 dataset we found that the combination features slightly lowered accuracy.

This year we avoided use of Internet resources (*i.e.,* those resources that required live connections to the Internet) in all of our run submissions. The various types of features that were used are described below.

2.3.1 String Features

A variety of string similarity features are incorporated to account for misspellings, name variants, or partially specified names. Particularly useful is the use of the Dice coefficient for the sets of character bigrams from the query name, Q_{name}, and the title name of a candidate KB entry, KB_{name}.

The ratio of the recursive longest common substring (Christen, 2006) to the shorter of Q_{name} or KB_{name} is effective at handling some deletions or word reorderings (*e.g.,* "John Adams" and "John Quincy Adams", or "Li Gong" and "Gong Li"). This method works by finding the longest common substring (*e.g.,* "Adams" in the first example) and removing it from each string, then recursively identifying the next longest common substring from the residual pieces and stopping the recursion when the length of the common substring found is less than some constant (we used a length of 2).

Checking whether all of the letters of Q_{name} are found in the same order in KB_{name} can also be indicative (*e.g.,* "Univ Maryland" would match "University of Maryland").

The set of string features is described in Table 2.

2.3.2 Document Features

Features based on analysis of the query document (Q_{doc}) or KB document text (KB_{doc}) are described in Table 3. We measured document similarity between Q_{doc} and the KB text in two ways: using cosine similarity with TF/IDF weighting (Salton and McGill, 1983); and using the Dice coefficient over bags of words. IDF values were approximated using counts from the Google 5-gram dataset as by Klein and Nelson (2008)

2.3.3 Entity Type Features

A few features based on entity type are described in Table 5. The type of the query entity mention is determined based running the named-entity recognizer by Ratinov and Roth (2009).

The reference knowledge base provided a type for each entity; however we found it to be rather incomplete. Table 4 shows that only 35% of KB nodes were assigned a tag of PER, ORG, or GPE; the remainder were typed UKN (unknown). Accordingly we used the *class* information (*i.e.,* the original Wikipedia Infobox class) which can be mapped to PER, ORG, GPE, or *other* with high accuracy.

Working through classes by frequency of occurrence we created an independent list that assigned types to 87% of nodes. In addition to PER, ORG, and GPE, we labelled many classes as BAD, meaning they could not be a PER, ORG, or GPE. This was helpful for discouraging selection of eponymous nodes named after famous entities (*e.g.,* the former U.S. president vs. E0194013 "John F. Kennedy International Airport"). We marked the remaining uncategorized classes as RARE.

[1] `http://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html`

Type	*Feature*
Real	The Dice coefficient for sets of characters from Q_{name} and KB_{name}
Real	The Dice coefficient for sets of characters bigrams from Q_{name} and KB_{name}
Real	The Dice coefficient for sets of characters bigrams from Q_{name} and KB_{name} multiplied by length(KB_{name})
Real	The Dice coefficient for sets of words from Q_{name} and KB_{name}
Bool	Whether Q_{name} and KB_{name} are an exact match
Bool	If Q_{name} is only one word in length
Real	Ratio of recursive longest common substring (Q_{name}, KB_{name}) to length of longer name
Bool	Do all letters from Q_{name} appear in order within KB_{name}, and similarly for KB_{name} and Q_{name}
Bool	Is Q_{name} wholly contained within KB_{name}, and similarly for KB_{name} and Q_{name}
Bool	Is Q_{name} a viable acroynm for KB_{name}?
Bool	Is Q_{name} an alias for KB_{name}?
Bool	Is Q_{name} either an acroynym or a known alias for KB_{name}?
Bool	Is the left/right Hamming distance for Q_{name} and KB_{name} $<= 1$?
Bool	Do Q_{name} and KB_{name} share more than 1 word in common?
Real	Ratio of shared name words to total number of words
Bool	Is the last word in KB_{name} present in Q_{name}?
Bool	Is the first word in KB_{name} present in Q_{name}?
Bool	Are both the first and last words in KB_{name} present in Q_{name}, and do they share more than 1 name word?
Real	A score reflective of being highly ranked compared to other candidate entries based on the Dice/bigram similarity score

Table 2: Features based on name similarity

Type	*Feature*
Real	The Dice coefficient for sets of words in Q_{doc} and KB_{doc}
Real	A TF/IDF weighted cosine score for Q_{doc} and KB_{doc}
Real	The percentage of KB_{name} words present in Q_{doc}
Real	The percentage of Q_{name} words present in KB_{doc}
Bool	If all words in Q_{name} are present early (*e.g.,* in the first 40 words) in KB_{doc}
Bool	If KB_{name} contains an appositive expression
Bool	If a KB_{name} appositive is found in Q_{doc}
Bool	If > 1 word in KB_{name} is present in Q_{doc}
Real	A score reflective of being highly ranked compared to other candidate entries based on document similarity

Table 3: Features based on document analysis

Type	*Feature*
Bool	If the labelled entity type for the KB entry is respectively UKN, PER, ORG, or GPE
Bool	If the Infobox class-derived (*i.e.,* calculated) entity type is respectively PER, ORG, GPE, BAD, or RARE
Bool	If the calculated query type is the same as the KB entry type, or recalculated KB entity type

Table 5: Features based on entity types

Type	*Reference KB*	*Calculated types*
PER	14.0%	21.8%
ORG	6.8%	7.2 %
GPE	14.2%	19.4%
UKN	64.9%	-
BAD	-	38.3%
RARE	-	13.3%

Table 4: Entity classes in the reference KB.

	Best	*Median*	*txt30*	*txt20*	*txt10*
All	0.8680	0.6836	0.8138	0.8147	0.8102
PER	0.9601	0.8449	0.9281	0.9241	0.9161
ORG	0.8520	0.6767	0.7773	0.7840	0.7733
GPE	0.7957	0.5975	0.7356	0.7356	0.7410

Table 9: Micro-averaged accuracy for our submitted runs (no restrictions) compared to best and median performance and differentiated by entity type.

2.3.4 Relation Features

The reference KB contains a set of facts, or slots, which indicate attributes about the KB entity or a relationship (*e.g.,* employment, spousal, etc...). While one could (and probably should) run a relation extractor over the query document and look for relational equivalences, or contradictions, we choose a simpler, more pragmatic approach. We simply treated the words from all facts as a surrogate 'document' and performed document similarity calculations against the query document. The specific features are described in Table 6.

2.3.5 Named-Entity Features

We applied the named-entity tagger by Ratinov and Roth (2009) to query documents and used a variety of features from the tagger output which are summarized in Table 7.

2.3.6 NIL Features

Some features can indicate whether it is likely or unlikely that there is a matching KB node for a query. For example, if many candidates have good name matches, it is likely that one of them is correct. Conversely, if no node has high node-text/article similarity, or overlap between facts and the article text, it becomes more reasonable to believe that the entity is missing from the KB. The list of features related to predicting absence is given in Table 8.

2.4 Training Data

Since our approach to entity linking is based on supervised learning we rely on the availability of training data. We used three sources of training examples: (1) the 3904 queries from the KBP 2009 evaluation; 1615 examples which we had annotated in 2009; and the 747 examples given as training for KBP 2010. In total we trained our models using 6266 examples, of which 1348 (21.5%) were PER, 3517 (56.1%) were ORG, and 1401 (22.4%) were GPE. 3510 of the examples (56.0%) were found in the KB and 2756 examples (44.0%) were not present in the KB.

3 Experimental Results

We submitted six runs for the entity linking task; three for the unrestricted condition which allowed use of KB article text, and three that did not use the KB text. None of our submitted runs made use of a live Internet connection, and avoidance of Internet-based features could have impacted the performance of our runs.

3.1 All features - text allowed

The three runs which we submitted for the unrestricted condition varied only in the flexibility in generating candidates. The three runs used different choices of k_1 and k_2 (see Section 2.1). Run `txt30` used $k_1 = k_2 = 30$; `txt20` used $k_1 = k_2 = 20$; `txt10` used $k_1 = k_2 = 10$. Summary statistics for these runs are given in Table 9.

The three submitted runs did not differ appreciably in performance. Each had accuracies that were healthily above the median, and we were pleased to see that run *txt20* was ranked 4th among the 16 systems which submitted results for this task.

3.2 KB article text not used

For this condition we submitted one run using our supervised machine learning system (*ntxta*) and we also submitted two other runs intended only as baselines (*ntxtb* and *ntxtc*). The run *ntxta* had no access to the KB article text (KB_{doc}), but in other respects was similar to run *txt30*. The later two runs were based solely on name matching and ignored both the query document (Q_{doc}) and the KB text (KB_{doc}). Run *ntxtb* compared normalized KB ti-

Type	*Feature*
Real	Dice coefficient for facts 'document' and Q_{doc}
Real	Cosine similarity using TF/IDf weights for facts 'document' and Q_{doc}
Real	Percentage of facts words present in Q_{doc}

Table 6: Features based on KB slots

Type	*Feature*
Real	Percentage of named-entites present in the associated KB text
Real	Percentage of words from all named-entites that are present in the associated KB text
Real	Dice coefficient and TF/IDF cosine similarity scores between NE words and the associated KB text
Real	Percentages of words, and named-entity strings present in the set of slots associated with KB entries
Bool	If Q_{name} is found in an longer recognized NE from Q_{doc}
Real	The character bigram Dice score between Q_{name} and the longest NE containing Q_{name}
Bool	Whether two or more co-occurring entities are found between Q_{doc} and KB_{doc}
Bool	If no supporting NEs are found between Q_{doc} and KB_{doc}
Real	A score reflective of being highly ranked compared to other candidate entries based on the percentage of co-occuring NEs

Table 7: Features based on named entities

Type	*Feature*
Bool	Whether a given 'candidate' is the NIL candidate or an actual KB entity
Real	Minimum, Maximum, and Average scores for several other features (*i.e.,* Document Similarity scores, similarity between Q_{name} and KB_{name}, and the percentage of common NE)
Bool	If any candidate's KB_{name} is an exact match for Q_{name}
Bool	If no candidate KB entry had supporting (*i.e.,* co-occuring) named-entities
Bool	Whether the set of Wikipedia page titles from a June 2010 snapshot contained an exact match for Q_{name}, but no candidate from the KBP KB was an exact match

Table 8: Features designed to indicate absence from the KB

	Best	*Median*	*ntxta*	*ntxtb*	*ntxtc*
All	0.7791	0.6347	**0.7791**	0.6076	0.6058
PER	0.9001	0.8202	0.8961	0.8322	0.8322
ORG	0.7333	0.6293	**0.7333**	0.6173	0.6120
GPE	0.7076	0.4920	**0.7076**	0.3725	0.3725

Table 10: Micro-averaged accuracy for our submitted runs (no wiki text condition) compared to best and median performance and differentiated by entity type.

tles against the query string and returned the ID of an exactly matching KB entry, if any, or returned `NIL` . Run *ntxtc* allowed for slightly fuzzier string matching compared to *ntxtb*.

Performance for these three runs is given in Table 10. According to the summary statistics released by NIST, our run *ntxta* had the highest overall accuracy of the 20 runs submitted for this condition. Our runs based only on name matching achieved performance near, but slightly below the median.

4 Conclusions

Our machine learning approach to entity linking appears to have proven successful for the TAC KBP 2010 entity linking task, despite the reduction in complexity in our feature set compared to the 2009 evalation. We were pleased to see our approach achieved some top marks for the condition where no use of the knowledge base article text was allowed.

References

Peter Christen. 2006. A comparison of personal name matching: Techniques and practical issues. Technical Report TR-CS-06-02, Australian National University.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Knowledge Discovery and Data Mining (KDD)*.

Martin Klein and Michael L. Nelson. 2008. A comparison of techniques for estimating IDF values to generate lexical signatures for the web. In *WIDM '08: Proceeding of the 10th ACM workshop on Web information and data management*, pages 39–46, New York, NY, USA. ACM.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado, June. Association for Computational Linguistics.

Gerard Salton and Michael McGill. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company.

William E. Winkler. 2006. Overview of record linkage and current research directions. Technical Report 2006-02, Statistical Research Division, U.S. Bureau of the Census.