Entity Disambiguation for Knowledge Base Population

†**Mark Dredze** and †**Paul McNamee** and †**Delip Rao** and †**Adam Gerber** and ⋆**Tim Finin**
†Human Language Technology Center of Excellence, Center for Language and Speech Processing
Johns Hopkins University
⋆University of Maryland – Baltimore County
mdredze,mcnamee,delip,adam.gerber@jhu.edu, finin@umbc.edu

Abstract

The integration of facts derived from information extraction systems into existing knowledge bases requires a system to disambiguate entity mentions in the text. This is challenging due to issues such as non-uniform variations in entity names, mention ambiguity, and entities absent from a knowledge base. We present a state of the art system for entity disambiguation that not only addresses these challenges but also scales to knowledge bases with several million entries using very little resources. Further, our approach achieves performance of up to 95% on entities mentioned from newswire and 80% on a public test set that was designed to include challenging queries.

1 Introduction

The ability to identify entities like people, organizations and geographic locations (Tjong Kim Sang and De Meulder, 2003), extract their attributes (Pasca, 2008), and identify entity relations (Banko and Etzioni, 2008) is useful for several applications in natural language processing and knowledge acquisition tasks like populating structured knowledge bases (KB).

However, inserting extracted knowledge into a KB is fraught with challenges arising from natural language ambiguity, textual inconsistencies, and lack of world knowledge. To the discerning human eye, the "Bush" in "Mr. Bush left for the Zurich environment summit in Air Force One." is clearly the US president. Further context may reveal it to be the 43rd president, George W. Bush, and not the 41st president, George H. W. Bush. The ability to disambiguate a polysemous entity mention or infer that two orthographically different mentions are the same entity is crucial in updating an entity's KB record. This task has been variously called entity disambiguation, record linkage, or entity linking. When performed without a KB, entity disambiguation is called coreference resolution: entity mentions either within the same document or across multiple documents are clustered together, where each cluster corresponds to a single real world entity.

The emergence of large scale publicly available KBs like Wikipedia and DBPedia has spurred an interest in linking textual entity references to their entries in these public KBs. Bunescu and Pasca (2006) and Cucerzan (2007) presented important pioneering work in this area, but suffer from several limitations including Wikipedia specific dependencies, scale, and the assumption of a KB entry for each entity. In this work we introduce an entity disambiguation system for linking entities to corresponding Wikipedia pages designed for open domains, where a large percentage of entities will not be linkable. Further, our method and some of our features readily generalize to other curated KB. We adopt a supervised approach, where each of the possible entities contained within Wikipedia are scored for a match to the query entity. We also describe techniques to deal with large knowledge bases, like Wikipedia, which contain millions of entries. Furthermore, our system learns when to withhold a link when an entity has no matching KB entry, a task that has largely been neglected in prior research in cross-document entity coreference. Our system produces high quality predictions compared with recent work on this task.

2 Related Work

The information extraction oeuvre has a gamut of relation extraction methods for entities like persons, organizations, and locations, which can be classified as open- or closed-domain depending on the restrictions on extractable relations (Banko and Etzioni, 2008). Closed domain systems extract a fixed set of relations while in open-domain systems, the number and type of relations are unbounded. Extracted relations still require processing before they can populate a KB with facts: namely, entity linking and disambiguation.

Motivated by ambiguity in personal name search, Mann and Yarowsky (2003) disambiguate person names using biographic facts, like birth year, occupation and affiliation. When present in text, biographic facts extracted using regular expressions help disambiguation. More recently, the Web People Search Task (Artiles et al., 2008) clustered web pages for entity disambiguation.

The related task of cross document coreference resolution has been addressed by several researchers starting from Bagga and Baldwin (1998). Poesio et al. (2008) built a cross document coreference system using features from encyclopedic sources like Wikipedia. However, successful coreference resolution is insufficient for correct entity linking, as the coreference chain must still be correctly mapped to the proper KB entry.

Previous work by Bunescu and Pasca (2006) and Cucerzan (2007) aims to link entity mentions to their corresponding topic pages in Wikipedia but the authors differ in their approaches. Cucerzan uses heuristic rules and Wikipedia disambiguation markup to derive mappings from surface forms of entities to their Wikipedia entries. For each entity in Wikipedia, a context vector is derived as a prototype for the entity and these vectors are compared (via dot-product) with the context vectors of unknown entity mentions. His work assumes that all entities have a corresponding Wikipedia entry, but this assumption fails for a significant number of entities in news articles and even more for other genres, like blogs. Bunescu and Pasca on the other hand suggest a simple method to handle entities not in Wikipedia by learning a threshold to decide if the entity is not in Wikipedia. Both works mentioned rely on Wikipedia-specific annotations, such as category hierarchies and disambiguation links.

We just recently became aware of a system fielded by Li *et al.* at the TAC-KBP 2009 evaluation (2009). Their approach bears a number of similarities to ours; both systems create candidate sets and then rank possibilities using differing learning methods, but the principal difference is in our approach to NIL prediction. Where we simply consider absence (*i.e.,* the NIL candidate) as another entry to rank, and select the top-ranked option, they use a separate binary classifier to decide whether their top prediction is correct, or whether NIL should be output. We believe relying on features that are designed to inform whether absence is correct is the better alternative.

3 Entity Linking

We define *entity linking* as matching a textual entity mention, possibly identified by a named entity recognizer, to a KB entry, such as a Wikipedia page that is a canonical entry for that entity. An entity linking *query* is a request to link a textual entity mention in a given document to an entry in a KB. The system can either return a matching entry or `NIL` to indicate there is no matching entry. In this work we focus on linking organizations, geo-political entities and persons to a Wikipedia derived KB.

3.1 Key Issues

There are 3 challenges to entity linking:

Name Variations. An entity often has multiple mention forms, including abbreviations (Boston Symphony Orchestra vs. BSO), shortened forms (Osama Bin Laden vs. Bin Laden), alternate spellings (Osama vs. Ussamah vs. Oussama), and aliases (Osama Bin Laden vs. Sheikh Al-Mujahid). Entity linking must find an entry despite changes in the mention string.

Entity Ambiguity. A single mention, like Springfield, can match multiple KB entries, as many entity names, like people and organizations, tend to be polysemous.

Absence. Processing large text collections virtually guarantees that many entities will not appear in the KB (`NIL`), even for large KBs.

The combination of these challenges makes entity linking especially challenging. Consider an example of "William Clinton." Most readers will immediately think of the 42nd US president. However, the only two William Clintons in Wikipedia are "William de Clinton" the 1st Earl of Huntingdon, and "William Henry Clinton" the British general. The page for the 42nd US president is actually "Bill Clinton". An entity linking system must decide if either of the William Clintons are correct, even though neither are exact matches. If the system determines neither matches, should it return NIL or the variant "Bill Clinton"? If variants are acceptable, then perhaps "Clinton, Iowa" or "DeWitt Clinton" should be acceptable answers?

3.2 Contributions

We address these entity linking challenges.
Robust Candidate Selection. Our system is flexible enough to find name variants but sufficiently restrictive to produce a manageable candidate list despite a large-scale KB.
Features for Entity Disambiguation. We developed a rich and extensible set of features based on the entity mention, the source document, and the KB entry. We use a machine learning ranker to score each candidate.
Learning NILs. We modify the ranker to learn NIL predictions, which obviates hand tuning and importantly, admits use of additional features that are indicative of NIL.

Our contributions differ from previous efforts (Bunescu and Pasca, 2006; Cucerzan, 2007) in several important ways. First, previous efforts depend on Wikipedia markup for significant performance gains. We make no such assumptions, although we show that optional Wikipedia features lead to a slight improvement. Second, Cucerzan does not handle NILs while Bunescu and Pasca address them by learning a threshold. Our approach *learns* to predict NIL in a more general and direct way. Third, we develop a rich feature set for entity linking that can work with any KB. Finally, we apply a novel finite state machine method for *learning* name variations. [1]

The remaining sections describe the candidate selection system, features and ranking, and our novel approach learning NILs, followed by an empirical evaluation.

4 Candidate Selection for Name Variants

The first system component addresses the challenge of name variants. As the KB contains a large number of entries (818,000 entities, of which 35% are PER, ORG or GPE), we require an efficient selection of the relevant candidates for a query.

Previous approaches used Wikipedia markup for filtering – only using the top-k page categories (Bunescu and Pasca, 2006) – which is limited to Wikipedia and does not work for general KBs. We consider a KB independent approach to selection that also allows for tuning candidate set size. This involves a linear pass over KB entry names (Wikipedia page titles): a naive implementation took two minutes per query. The following section reduces this to under two *seconds* per query.

For a given query, the system selects KB entries using the following approach:

- Titles that are exact matches for the mention.
- Titles that are wholly contained in or contain the mention (e.g., *Nationwide* and *Nationwide Insurance*).
- The first letters of the entity mention match the KB entry title (e.g., *OA* and *Olympic Airlines*).
- The title matches a known alias for the entity (aliases described in Section 5.2).
- The title has a strong string similarity score with the entity mention. We include several measures of string similarity, including: character Dice score > 0.9, skip bigram Dice score > 0.6, and Hamming distance $<= 2$.

We did not optimize the thresholds for string similarity, but these could obviously be tuned to minimize the candidate sets and maximize recall.

All of the above features are general for any KB. However, since our evaluation used a KB derived from Wikipedia, we included a few Wikipedia specific features. We added an entry if its Wikipedia page appeared in the top 20 Google results for a query.

On the training dataset (Section 7) the selection system attained a recall of 98.8% and produced candidate lists that were three to four orders of magnitude smaller than the KB. Some recall errors were due to inexact acronyms: ABC (Arab Banking; 'Corporation' is missing), ASG (Abu Sayyaf; 'Group' is missing), and PCF (French Communist Party; French reverses the order of the pre-nominal adjectives). We also missed International Police (Interpol) and Becks (David Beckham; Mr. Beckham and his wife are collectively referred to as 'Posh and Becks').

[1]http://www.clsp.jhu.edu/ markus/fstrain

4.1 Scaling Candidate Selection

Our previously described candidate selection relied on a linear pass over the KB, but we seek more efficient methods. We observed that the above non-string similarity filters can be precomputed and stored in an index, and that the skip bigram Dice score can be computed by indexing the skip bigrams for each KB title. We omitted the other string similarity scores, and collectively these changes enable us to avoid a linear pass over the KB. Finally we obtained speedups by serving the KB concurrently[2]. Recall was nearly identical to the full system described above: only two more queries failed. Additionally, more than 95% of the processing time was consumed by Dice score computation, which was only required to correctly retrieve less than 4% of the training queries. Omitting the Dice computation yielded results in a few milliseconds. A related approach is that of canopies for scaling clustering for large amounts of bibliographic citations (McCallum et al., 2000). In contrast, our setting focuses on alignment vs. clustering mentions, for which overlapping partitioning approaches like canopies are applicable.

5 Entity Linking as Ranking

We select a single correct candidate for a query using a supervised machine learning ranker. We represent each query by a D dimensional vector $\mathbf{x}$, where $\mathbf{x} \in \mathbb{R}^D$, and we aim to select a single KB entry y, where $y \in \mathcal{Y}$, a set of possible KB entries for this query produced by the selection system above, which ensures that $\mathcal{Y}$ is small. The ith query is given by the pair $\{\mathbf{x}_i, y_i\}$, where we assume at most one correct KB entry.

To evaluate each candidate KB entry in $\mathcal{Y}$ we create feature functions of the form $f(\mathbf{x}, y)$, dependent on both the example $\mathbf{x}$ (document and entity mention) and the KB entry y. The features address name variants and entity disambiguation.

We take a maximum margin approach to learning: the correct KB entry y should receive a higher score than all other possible KB entries $\hat{y} \in \mathcal{Y}, \hat{y} \neq y$ plus some margin γ. This learning constraint is equivalent to the ranking SVM algorithm of Joachims (2002), where we define an ordered pair constraint for each of the incorrect KB entries $\hat{y}$ and the correct entry y. Training sets parameters such that score$(y) \geq$ score$(\hat{y}) + \gamma$. We used the library SVM$^{\text{rank}}$ to solve this optimization problem.[3] We used a linear kernel, set the slack parameter C as 0.01 times the number of training examples, and take the loss function as the total number of swapped pairs summed over all training examples. While previous work used a custom kernel, we found a linear kernel just as effective with our features. This has the advantage of efficiency in both training and prediction [4] – important considerations in a system meant to scale to millions of KB entries.

5.1 Features for Entity Disambiguation

200 atomic features represent $\mathbf{x}$ based on each candidate query/KB pair. Since we used a linear kernel, we explicitly combined certain features (e.g., acroynym-match AND known-alias) to model correlations. This included combining each feature with the predicted type of the entity, allowing the algorithm to learn prediction functions specific to each entity type. With feature combinations, the total number of features grew to 26,569. The next sections provide an overview; for a detailed list see McNamee et al. (2009).

5.2 Features for Name Variants

Variation in entity name has long been recognized as a bane for information extraction systems. Poor handling of entity name variants results in low recall. We describe several features ranging from simple string match to finite state transducer matching.

String Equality. If the query name and KB entry name are identical, this is a strong indication of a match, and in our KB entry names are distinct. However, similar or identical entry names that refer to distinct entities are often qualified with parenthetical expressions or short clauses. As an example, "London, Kentucky" is distinguished

[2]Our Python implementation with indexing features and four threads achieved up to 80× speedup compared to naive implementation.

[3]`www.cs.cornell.edu/people/tj/svm_light/svm_rank.html`

[4]Bunescu and Pasca (2006) report learning tens of thousands of support vectors with their "taxonomy" kernel while a linear kernel represents all support vectors with a single weight vector, enabling faster training and prediction.

from "London, Ontario", "London, Arkansas", "London (novel)", and "London". Therefore, other string equality features were used, such as whether names are equivalent after some transformation. For example, "Baltimore" and "Baltimore City" are exact matches after removing a common GPE word like city; "University of Vermont" and "University of VT" match if VT is expanded.

Approximate String Matching. Many entity mentions will not match full names exactly. We added features for character Dice, skip bigram Dice, and left and right Hamming distance scores. Features were set based on quantized scores. These were useful for detecting minor spelling variations or mistakes. Features were also added if the query was wholly contained in the entry name, or vice-versa, which was useful for handling ellipsis (e.g., "United States Department of Agriculture" vs. "Department of Agriculture"). We also included the ratio of the recursive longest common subsequence (Christen, 2006) to the shorter of the mention or entry name, which is effective at handling some deletions or word reorderings (e.g., "Li Gong" and "Gong Li"). Finally, we checked whether all of the letters of the query are found in the same order in the entry name (e.g., "Univ Wisconsin" would match "University of Wisconsin").

Acronyms. Features for acronyms, using dictionaries and partial character matches, enable matches between "MIT" and "Madras Institute of Technology" or "Ministry of Industry and Trade."

Aliases. Many aliases or nicknames are non-trivial to guess. For example JAVA is the stock symbol for Sun Microsystems, and "Ginger Spice" is a stage name of Geri Halliwell. A reasonable way to do this is to employ a dictionary and alias lists that are commonly available for many domains[5].

FST Name Matching. Another measure of surface similarity between a query and a candidate was computed by training finite-state transducers similar to those described in Dreyer et al. (2008). These transducers assign a score to any string pair by summing over all alignments and scoring all contained character n-grams; we used n-grams of length 3 and less. The scores are combined using a global log-linear model. Since different spellings of a name may vary considerably in length (e.g., *J Miller* vs. *Jennifer Miller*) we eliminated the limit on consecutive insertions used in previous applications.[6]

[5]We used multiple lists, including class-specific lists (i.e., for PER, ORG, and GPE) lists extracted from Freebase (Bollacker et al., 2008) and Wikipedia redirects. PER, ORG, and GPE are the commonly used terms for entity types for people, organizations and geo-political regions respectively.

5.3 Wikipedia Features

Most of our features do not depend on Wikipedia markup, but it is reasonable to include features from KB properties. Our feature ablation study shows that dropping these features causes a small but statistically significant performance drop.

WikiGraph statistics. We added features derived from the Wikipedia graph structure for an entry, like indegree of a node, outdegree of a node, and Wikipedia page length in bytes. These statistics favor common entity mentions over rare ones.

Wikitology. KB entries can be indexed with human or machine generated metadata consisting of keywords or categories in a domain-appropriate taxonomy. Using a system called *Wikitology*, Syed et al. (2008) investigated use of ontology terms obtained from the explicit category system in Wikipedia as well as relationships induced from the hyperlink graph between related Wikipedia pages. Following this approach we computed top-ranked categories for the query documents and used this information as features. If none of the candidate KB entries had corresponding highly-ranked Wikitology pages, we used this as a `NIL` feature (Section 6.1).

5.4 Popularity

Although it may be an unsafe bias to give preference to common entities, we find it helpful to provide estimates of entity popularity to our ranker as others have done (Fader et al., 2009). Apart from the graph-theoretic features derived from the Wikipedia graph, we used Google's PageRank to by adding features indicating the rank of the KB entry's corresponding Wikipedia page in a Google query for the target entity mention.

[6]Without such a limit, the objective function may diverge for certain parameters of the model; we detect such cases and learn to avoid them during training.

5.5 Document Features

The mention document and text associated with a KB entry contain context for resolving ambiguity.

Entity Mentions. Some features were based on presence of names in the text: whether the query appeared in the KB text and the entry name in the document. Additionally, we used a named-entity tagger and relation finder, SERIF (Boschee et al., 2005), identified name and nominal mentions that were deemed co-referent with the entity mention in the document, and tested whether these nouns were present in the KB text. Without the NE analysis, accuracy on non-NIL entities dropped 4.5%.

KB Facts. KB nodes contain infobox attributes (or facts); we tested whether the fact text was present in the query document, both locally to a mention, or anywhere in the text. Although these facts were derived from Wikipedia infoboxes, they could be obtained from other sources as well.

Document Similarity We measured similarity between the query document and the KB text in two ways: 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).

Entity Types. Since the KB contained types for entries, we used these as features as well as the predicted NE type for the entity mention in the document text. Additionally, since only a small number of KB entries had PER, ORG, or GPE types, we also inferred types from Infobox class information to attain 87% coverage in the KB. This was helpful for discouraging selection of eponymous entries named after famous entities (e.g., the former U.S. president vs. "John F. Kennedy International Airport").

5.6 Feature Combinations

To take into account feature dependencies we created combination features by taking the cross-product of a small set of diverse features. The attributes used as combination features included entity type; a popularity based on Google's rankings; document comparison using TF/IDF; coverage of co-referential nouns in the KB node text; and name similarity. The combinations were cascaded to allow arbitrary feature conjunctions. Thus it is possible to end up with a feature *kbtype-is-ORG AND high-TFIDF-score AND low-name-similarity*. The combined features increased the number of features from roughly 200 to 26,000.

6 Predicting NIL Mentions

So far we have assumed that each example has a correct KB entry; however, when run over a large corpus, such as news articles, we expect a significant number of entities will not appear in the KB. Hence it will be useful to predict `NIL`s.

We *learn* when to predict `NIL` using the SVM ranker by augmenting $\mathcal{Y}$ to include `NIL`, which then has a single feature unique to `NIL` answers. It can be shown that (modulo slack variables) this is equivalent to learning a single threshold τ for `NIL` predictions as in Bunescu and Pasca (2006).

Incorporating `NIL` into the ranker has several advantages. First, the ranker can set the threshold optimally without hand tuning. Second, since the SVM scores are relative within a single example and cannot be compared across examples, setting a single threshold is difficult. Third, a threshold sets a uniform standard across all examples, whereas in practice we may have reasons to favor a `NIL` prediction in a given example. We design features for `NIL` prediction that cannot be captured in a single parameter.

6.1 NIL Features

Integrating `NIL` prediction into learning means we can define arbitrary features indicative of `NIL` predictions in the feature vector corresponding to `NIL`. For example, if many candidates have good name matches, it is likely that one of them is correct. Conversely, if no candidate has high entry-text/article similarity, or overlap between facts and the article text, it is likely that the entity is absent from the KB. We included several features, such as a) the max, mean, and difference between max and mean for 7 atomic features for all KB candidates considered, b) whether any of the candidate entries have matching names (exact and fuzzy string matching), c) whether any KB entry was a top Wikitology match, and d) if the top Google match was not a candidate.

	Micro-Averaged				Macro-Averaged			
	Best	*Median*	*All Features*	*Best Features*	*Best*	*Median*	*All Features*	*Best Features*
All	0.8217	0.7108	0.7984	0.7941	0.7704	0.6861	0.7695	**0.7704**
non-NIL	0.7725	0.6352	0.7063	0.6639	0.6696	0.5335	0.6097	0.5593
NIL	0.8919	0.7891	0.8677	**0.8919**	0.8789	0.7446	0.8464	0.8721

Table 1: Micro and macro-averaged accuracy for TAC-KBP data compared to best and median reported performance. Results are shown for all features as well as removing a small number of features using feature selection on development data.

7 Evaluation

We evaluated our system on two datasets: the Text Analysis Conference (TAC) track on Knowledge Base Population (TAC-KBP) (McNamee and Dang, 2009) and the newswire data used by Cucerzan (2007) (Microsoft News Data).

Since our approach relies on supervised learning, we begin by constructing our own training corpus.[7] We highlighted 1496 named entity mentions in news documents (from the TAC-KBP document collection) and linked these to entries in a KB derived from Wikipedia infoboxes. [8] We added to this collection 119 sample queries from the TAC-KBP data. The total of 1615 training examples included 539 (33.4%) PER, 618 (38.3%) ORG, and 458 (28.4%) GPE entity mentions. Of the training examples, 80.5% were found in the KB, matching 300 unique entities. This set has a higher number of NIL entities than did Bunescu and Pasca (2006) (10%) but lower than the TAC-KBP test set (43%).

All system development was done using a train (908 examples) and development (707 examples) split. The TAC-KBP and Microsoft News data sets were held out for final tests. A model trained on all 1615 examples was used for experiments.

7.1 TAC-KBP 2009 Experiments

The KB is derived from English Wikipedia pages that contained an infobox. Entries contain basic descriptions (article text) and attributes. The TAC-KBP query set contains 3904 entity mentions for 560 distinct entities; entity type was only provided for evaluation. The majority of queries were for organizations (69%). Most queries were missing from the KB (57%). 77% of the distinct GPEs in the queries were present in the KB, but for PERs and ORGs these percentages were significantly lower, 19% and 30% respectively.

Table 1 shows results on TAC-KBP data using all of our features as well a subset of features based on feature selection experiments on development data. We include scores for both micro-averaged accuracy – averaged over all queries – and macro-averaged accuracy – averaged over each unique entity – as well as the best and median reported results for these data (McNamee and Dang, 2009). We obtained the best reported results for macro-averaged accuracy, as well as the best results for NIL detection with micro-averaged accuracy, which shows the advantage of our approach to learning NIL. See McNamee et al. (2009) for additional experiments.

The candidate selection phase obtained a recall of 98.6%, similar to that of development data. Missed candidates included *Iron Lady*, which refers metaphorically to Yulia Tymoshenko, *PCC*, the Spanish-origin acronym for the Cuban Communist Party, and *Queen City*, a former nickname for the city of Seattle, Washington. The system returned a mean of 76 candidates per query, but the median was 15 and the maximum 2772 (*Texas*). In about 10% of cases there were four or fewer candidates and in 10% of cases there were more than 100 candidate KB nodes. We observed that ORGs were more difficult, due to the greater variation and complexity in their naming, and that they can be named after persons or locations.

7.2 Feature Effectiveness

We performed two feature analyses on the TAC-KBP data: an additive study – starting from a small baseline feature set used in candidate selection we add feature groups and measure performance changes (omitting feature combinations), and an ablative study – starting from all features, remove a feature group and measure performance.

[7]Data available from www.dredze.com

[8]http://en.wikipedia.org/wiki/Help:Infobox

Class	*All*	*non-NIL*	*NIL*
Baseline	0.7264	0.4621	0.9251
Acronyms	0.7316	0.4860	0.9161
NE Analysis	0.7661	0.7181	0.8022
Google	0.7597	0.7421	0.7730
Doc/KB Text Similarity	0.7313	0.6699	0.7775
Wikitology	0.7318	0.4549	0.9399
All	0.7984	0.7063	0.8677

Table 2: Additive analysis: micro-averaged accuracy.

	Num. Queries		Accuracy		
	Total	*Nil*	*All*	*non-NIL*	*NIL*
NIL	452	187	0.4137	0.0	1.0
GPE	132	20	0.9696	1.00	0.8000
ORG	115	45	0.8348	0.7286	1.00
PER	205	122	0.9951	0.9880	1.00
All	452	187	**0.9469**	0.9245	0.9786
Cucerzan (2007)			0.914	-	-

Table 3: Micro-average results for Microsoft data.

Table 2 shows the most significant features in the feature addition experiments. The baseline includes only features based on string similarity or aliases and is not effective at finding correct entries and strongly favors NIL predictions. Inclusion of features based on analysis of named-entities, popularity measures (e.g., Google rankings), and text comparisons provided the largest gains. The overall changes are fairly small, roughly $\pm 1\%$; however changes in non-NIL precision are larger.

The ablation study showed considerable redundancy across feature groupings. In several cases, performance could have been slightly improved by removing features. Removing all feature combinations would have improved overall performance to 81.05% by gaining on non-NIL for a small decline on NIL detection.

7.3 Experiments on Microsoft News Data

We downloaded the evaluation data used in Cucerzan (2007)[9]: 20 news stories from MSNBC with 642 entity mentions manually linked to Wikipedia and another 113 mentions not having any corresponding link to Wikipedia.[10] A significant percentage of queries were not of type PER, ORG, or GPE (e.g., “Christmas”). SERIF assigned entity types and we removed 297 queries not recognized as entities (counts in Table 3).

We learned a new model on the training data above using a reduced feature set to increase speed.[11] Using our fast candidate selection system, we resolved each query in 1.98 seconds (median). Query processing time was proportional to the number of candidates considered. We selected a median of 13 candidates for PER, 12 for ORG and 102 for GPE. Accuracy results are in Table 3. The high results reported for this dataset over TAC-KBP is primarily because we perform very well in predicting popular and rare entries – both of which are common in newswire text.

One issue with our KB was that it was derived from infoboxes in Wikipedia's Oct 2008 version which has both new entities, [12] and is missing entities.[13] Therefore, we manually confirmed NIL answers and new answers for queries marked as NIL in the data. While an exact comparison is not possible (as described above), our results (94.7%) appear to be at least on par with Cucerzan's system (91.4% overall accuracy).With the strong results on TAC-KBP, we believe that this is strong confirmation of the effectiveness of our approach.

8 Conclusion

We presented a state of the art system to disambiguate entity mentions in text and link them to a knowledge base. Unlike previous approaches, our approach readily ports to KBs other than Wikipedia. We described several important challenges in the entity linking task including handling variations in entity names, ambiguity in entity mentions, and missing entities in the KB, and we showed how to each of these can be addressed. We described a comprehensive feature set to accomplish this task in a supervised setting. Importantly, our method discriminately learns when not to link with high accuracy. To spur further research in these areas we are releasing our entity linking system.

[9] http://research.microsoft.com/en-us/um/people/silviu/WebAssistant/TestData/

[10] One of the MSNBC news articles is no longer available so we used 759 total entities.

[11] We removed Google, FST and conjunction features which reduced system accuracy but increased performance.

[12] 2008 vs. 2006 version used in Cucerzan (2007) We could not get the 2006 version from the author or the Internet.

[13] Since our KB was derived from infoboxes, entities not having an infobox were left out.

References

Javier Artiles, Satoshi Sekine, and Julio Gonzalo. 2008. Web people search: results of the first evaluation and the plan for the second. In *WWW*.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Conference on Computational Linguistics (COLING)*.

Michele Banko and Oren Etzioni. 2008. The tradeoffs between open and traditional relation extraction. In *Association for Computational Linguistics*.

K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD Management of Data*.

E. Boschee, R. Weischedel, and A. Zamanian. 2005. Automatic information extraction. In *Conference on Intelligence Analysis*.

Razvan C. Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *European Chapter of the Assocation for Computational Linguistics (EACL)*.

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

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Markus Dreyer, Jason Smith, and Jason Eisner. 2008. Latent-variable modeling of string transductions with finite-state methods. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2009. Scaling Wikipedia-based named entity disambiguation to arbitrary web text. In *WikiAI09 Workshop at IJCAI 2009*.

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 *Workshop on Web Information and Data Management (WIDM)*.

Fangtao Li, Zhicheng Zhang, Fan Bu, Yang Tang, Xiaoyan Zhu, and Minlie Huang. 2009. THU QUANTA at TAC 2009 KBP and RTE track. In *Text Analysis Conference (TAC)*.

Gideon S. Mann and David Yarowsky. 2003. Unsupervised personal name disambiguation. In *Conference on Natural Language Learning (CONLL)*.

Andrew McCallum, Kamal Nigam, and Lyle Ungar. 2000. Efficient clustering of high-dimensional data sets with application to reference matching. In *Knowledge Discovery and Data Mining (KDD)*.

Paul McNamee and Hoa Trang Dang. 2009. Overview of the TAC 2009 knowledge base population track. In *Text Analysis Conference (TAC)*.

Paul McNamee, Mark Dredze, Adam Gerber, Nikesh Garera, Tim Finin, James Mayfield, Christine Piatko, Delip Rao, David Yarowsky, and Markus Dreyer. 2009. HLTCOE approaches to knowledge base population at TAC 2009. In *Text Analysis Conference (TAC)*.

Marius Pasca. 2008. Turning web text and search queries into factual knowledge: hierarchical class attribute extraction. In *National Conference on Artificial Intelligence (AAAI)*.

Massimo Poesio, David Day, Ron Artstein, Jason Duncan, Vladimir Eidelman, Claudio Giuliano, Rob Hall, Janet Hitzeman, Alan Jern, Mijail Kabadjov, Stanley Yong, Wai Keong, Gideon Mann, Alessandro Moschitti, Simone Ponzetto, Jason Smith, Josef Steinberger, Michael Strube, Jian Su, Yannick Versley, Xiaofeng Yang, and Michael Wick. 2008. Exploiting lexical and encyclopedic resources for entity disambiguation: Final report. Technical report, JHU CLSP 2007 Summer Workshop.

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

Erik Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Conference on Natural Language Learning (CONLL)*.

Zareen Syed, Tim Finin, and Anupam Joshi. 2008. Wikipedia as an ontology for describing documents. In *Proceedings of the Second International Conference on Weblogs and Social Media*. AAAI Press.