TCAR at TAC-KBP-2010

Alan Goldschen, Brian Kapp, Tim Meyer, Boyan Onyshkevych, Pat Schone

U.S. Department of Defense

Ft. George G. Meade, MD, 20755-6513

1 ABSTRACT

The TCAR team developed multiple information retrieval based entity linking systems in a matter of weeks for the TAC-KBP evaluation task. We focused primarily on developing entity linking algorithms that do not require Wikipedia text and correctly detect when a given entity does not exist in Wikipedia (NIL). Without using Wikipedia text, the system achieves an overall TAC 2010 score of 67 percent. The system achieves a score of 86 percent in the correct detection of when Wikipedia does not contain the entity; moreover, this score improves to 97 percent for PER entities. We provide descriptions of our systems and their TAC 2010 performance.

2 INTRODUCTION

Automatic knowledge base population (KBP) is a challenging problem that advances the state of the art in language processing while fusing efforts from multiple communities. As presented by the TAC-KBP organizers, KBP draws largely upon techniques that have been studied and analyzed through previous NIST evaluations, such as automatic content extraction (ACE), automatic questions answering (QA), and information retrieval (IR). Given our interest in each of these areas and our desire to help foster KBP efforts, we elected to participate in this year's evaluation. Our goals for TAC 2010 are to develop systems that do not use Wikipedia article text and to improve upon our TAC 2009 system, especially in detecting NIL nodes. Specifically to achieve these goals, we developed a new preprocessing filtering algorithm to improve system recall and developed a graph-based entity linking approach.

As we did last year [6], we use only information provided for the TAC competition. We use only the provided reference documents and Wikipedia entries; we did not use other sources of information. We chose primarily to develop approaches that did not use the text of the Wikipedia page.

We describe systems that we prototyped in this effort, and discuss the merits and weaknesses of each of these systems. We provide the overall performance of each system, and mention limitations to each of our system algorithms. Lastly, we will indicate some of our future directions in these areas.

3 TASK RESOURCES

As in TAC 2009 [6], we made use of our tools for content extraction, topic tagging, information retrieval, and our auxiliary resources for specific tasks. These tools include:

- A hybrid statistical and rule-based entity extractor extended from the BBN Identifinder system [1].
- A coreference and relation detection system developed that uses handcrafted templates to identify lexical patterns to express relations.

- The BBN SERIF system [2] that identifies coreferences, relations, and constituency parses.
- A tool, using semantic forests, that produces topical lists from statistics and machine-readable dictionaries as described in TREC competitions [7].
- A tool that characterizes each document in terms of ontology concepts that extends from the OMEGA ontology [8].
- Development Evaluation Resources – We used an internally developed set of 600 entity links and the TAC 2009 result set [6], as well as the TAC 2010 development set.
- Name Aliases – A rudimentary name-equivalence list with speech pronunciation (such as "Jon" and "John") name variants. We augment this list with common nicknames.

4 ENTITY LINKING

Figure 1 depicts the general TCAR entity linking system architecture for TAC. The entity linking task is to develop a correlation algorithm that links a given entity from a given document to a specific Wikipedia page from a set of 860,000 Wikipedia entries.

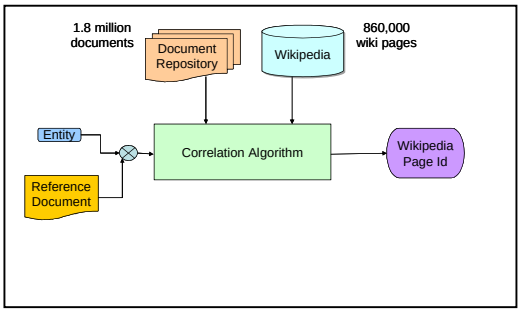

Figure 1: Overall entity linking TAC system that consists of correlation algorithms that link a given entity from a reference document to a Wikipedia page.

We continued to use the resources as we did in TAC 2009 [6]. Figure 2 illustrates the resources used by each of the entity linking systems. The document repository contains entities, relations, within-document co-references, and sentence parses that BBN Serif extraction engine identifies for each document. Additionally, the document repository contains document topics and semantic concepts. The Wikipedia repository contains similarly extracted information from the text associated with each Wikipedia page. An algorithm automatically determines Wikipedia page title, Wikipedia page facts, and Wikipedia page types (PER, ORG, or GPE). TAC relations map to the relation names from both the document and Wikipedia repositories. We developed lists and algorithms to process name variants. An algorithm transliterates name accents into unaccented names. A list contains each name represented as overlapping n-grams. Finally, we created an algorithm to represent each name as a nickname or acronym. We did not use an already developed acronym or nickname list. The nicknames or acronyms occur from within-document co-reference chains and various name parts.

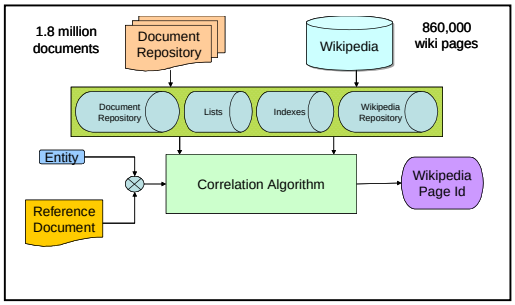

Figure 2: The entity linking system resources that consist of content information about documents and Wikipedia entries, lists, and indexes.

Our TAC 2010 system consists of a filtering phase and various entity linking systems that Figure 3 depicts. Unlike

our TAC 2009 system [6], we did not use search indexes to filter our initial set of documents nor did we use search indexes to detect when Wikipedia does not contain the reference entity. Instead, we relied on a filtering process based on n-grams to provide an initial set of Wikipedia entries. Our entity linking systems, depicted in Figure 3, uses an information-retrieval-based approach to determine which Wikipedia page corresponds to the given entity (NON-NIL) or that no Wikipedia page corresponds to the given entity (NIL).

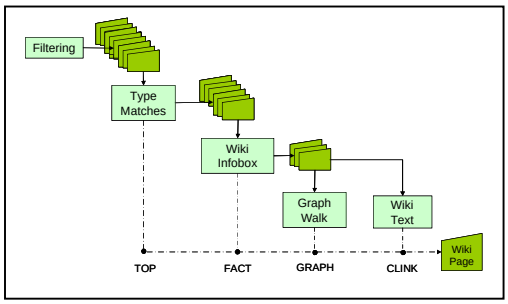

Figure 3: The overall system flow and systems (TOP, FACT, GRAPH, and CLINK). Each system outputs the correct wiki page based on the filtered set of Wikipedia pages from the filtering process.

The filtering phase consists of identifying and ranking a set of Wikipedia entries from an initial set of 860,000 Wikipedia entries. The first step in the filtering process is to retrieve an initial set of Wikipedia entries based on how likely the entity name and the content of the reference document matches the Wikipedia page title.

We developed four entity-linking systems, three of which do not use the Wikipedia text and one that uses only the links in the Wikipedia page text. We describe the components of these systems in further detail after describing filter set generation.[1]

4.1.2 Filtering Set Generation

Figure 4 illustrates the filtering steps, which produce our initial set of Wikipedia entries. The filtering process always adds NIL as a valid response to the set of Wikipedia entries. The approach uses a combination of an overlapping character n-gram matching algorithm and a set of filters designed to remove spurious candidates. This system does not consider any infobox material and focuses only on the titles of the Wikipedia entries to return a ranked set of candidates.

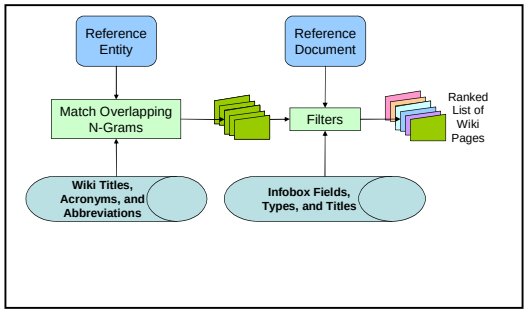

Figure 4: The filter process uses the reference entity for overlapping n-grams and the reference document for other filter algorithms

To aid in finding the overlapping n-grams, a substantial amount of analysis checks Wikipedia redirect links and considers information at the beginning of Wikipedia entries that appear to be variations of the title. Thus, the overlapping n-grams did not necessarily have to be a match with the title of a Wikipedia page, but rather could be a good match for a variant and still appear as a good candidate. Additionally, we

[1] Herein TOP, FACT, GRAPH, and CLINK refer to the following systems: TCAR-3, TCAR-2, TCAR-1, and TCAR-2 (using Wikipedia context), respectively.

add a list of acronyms and abbreviations, which identifies capital letters and overlapping n-grams, to this list of titles.

Once the system completes the check on the n-grams, it filters the results based on various criteria from the content of the document. As mentioned above, these filters remove spurious candidates. This process was especially focused on entities of type PER, and this aspect of the design is evidenced by the results (see Figure 4). The filtering process returns a ranked list of Wikipedia entries by using only the Wikipedia entry titles.

In addition to returning a list of candidates, the system also made an effort to determine infobox types by using processes from wiki mining such as category information, infobox fields, and disambiguation found within titles. This step was particularly useful in categorizing Wikipedia entries that had a type field of UKN and checking for type mismatches in other phases. [2]

4.1.3 System TOP

Using the filtered set of Wikipedia entries, TOP returns the most likely Wikipedia entry with the same entity type as the given entity in the reference document. The entity extractor determines the type (PER, ORG, GPE, or UKN) of the reference entity using the reference document. We use the type tag in the Wikipedia infoboxes for these candidates and any parenthetical information from the candidates' infoboxes to prune the list. If the entity extractor cannot determine a type for the entity mentioned in the document, no further pruning is done; however, if the entity extractor finds that an entity type is not a PER, ORG, or GPE, then it is removed.

[2] We use UKN to denote unknown, meaning that we did not know the entity type; that is, it is not a PER, ORG, or GPE entity.

4.1.4 System FACT

FACT prunes the Wikipedia candidate entries as based on the reference entity type as specified in TOP. After entity type classification, FACT uses Wikipedia page slots (facts) to detect Wikipedia entries that do not correspond to the query term. A name in the Wikipedia page name must first match the query term, where a match includes name parts and abbreviations. After a successful match, the algorithm counts different facts that have values that occur in the document. This process simply uses exact string matches between a slot value from any one of the remaining candidates and anything in the reference document. If there are more than two fact values in the documents, then the algorithm returns the Wikipedia page. If only a single fact value occurs in the document and the Wikipedia title and page type are the same the query entity, the algorithm returns the Wikipedia page. FACT, based on our TAC 2009 [6] system, uses Wikipedia titles and slots, not the text of the Wikipedia article. These constraints, along with the new filtering processing, greatly enhance our ability to detect when the Wikipedia page does not exist (i.e., the NIL case).

4.1.5 System GRAPH

Overview: GRAPH uses a graph-based approach that attempts to resolve identities of several other named entities in the reference document in addition to the one named in the query. Specifically, GRAPH uses the filtering algorithm described above to generate a set of Wikipedia candidate entries for the entity mentioned in the query and a

much stricter algorithm to generate potential matches for up to five other named entities within the reference document. GRAPH first completes a simple pruning process for the main candidates. If more then one Wikipedia entry remains, GRAPH then creates a graph and proceeds with a walk on this graph to determine which candidate Wikipedia entry, if any, is the appropriate Wikipedia entry. GRAPH uses only Wikipedia infobox information as FACT does.

Preprocessing: Much of the preprocessing that goes into this algorithm coincides with the preprocessing done for the filtering algorithm, so we will mention only the different portions here. To allow for a more strict system for generating potential matching Wikipedia entries to certain named entities in the document, GRAPH uses a Lemur [4] index based only on the titles in Wikipedia and another Lemur index based on the infobox text and the titles. GRAPH also makes use of pre-generated lists of incoming and outgoing links to and from each candidate Wikipedia entry. These lists only make use of the links in the infoboxes, so any links that occur within the text of a Wikipedia article are not contained in them. Additionally, GRAPH uses the entity extractor to tag all of the entities in the document, which then is part of the pruning process for the list of candidates in place.

Candidate Generation and Pruning: GRAPH initially uses the filtering algorithm mentioned above. Rather than choosing only the top candidate according to the algorithm, GRAPH produces several Wikipedia candidates for the entity in the query. In principle, this could return many candidates, but in practice, GRAPH never saw an overwhelming number and, on quite a few occasions, there were no candidates returned. In the event that there are no candidates during this stage, or if no candidates remain after any other stage, GRAPH returns a NIL response.

GRAPH next prunes the candidate list using the entity extractor as specified in TOP. After reducing the candidate list based on entity type classification, GRAPH then considers all of the slot values in the Wikipedia infoboxes and compares these to the document to see if overlap occurs (as in FACT). In the case where one candidate has more overlap with the document than all others, GRAPH returns this candidate as the response. However, in the case of ties in this overlap score we proceeded to the graph walk.

Graph Walk: When no clear winner emerges from the pruning processes, GRAPH next determines the best candidate by performing a walk on a graph. To do this, GRAPH constructs a graph for traversal. GRAPH first adds a node for each candidate, including NIL; for each candidate node, GRAPH places an acceptor node that has no outgoing links and whose only incoming link comes from its corresponding Wikipedia candidate. Next, GRAPH generates a list of up to five auxiliary entities which are other named entities from the reference document, For each of these auxiliary entities, GRAPH generates a set of Wikipedia entries that corresponds to the entity. These sets are generally smaller than the initial candidate set since a more stringent algorithm generates them. Each of these Wikipedia entries becomes a node in the graph.

To perform a walk on this graph, GRAPH links these nodes using the connectivity properties between entries in Wikipedia and other features from the Wikipedia entries' infoboxes. For instance, if two Wikipedia entries are either one or two clicks away in Wikipedia, GRAPH places a link between them. As an example of how GRAPH forms other links, consider the following: if an infobox contains the spouse of a person, GRAPH attempts to determine which node in the graph corresponds to the spouse (or if the node is even present at all). If a link is present in Wikipedia between the person and their spouse, GRAPH places an additional link in the graph to denote this relationship. When this relationship could not be exactly determined (i.e., no link was present), GRAPH resorts to checking to see if the name was a good match for the spouse and creates a link accordingly. These links contain a nominal value based on how good a name match for the spouse the node title is, so, in principle, there could be multiple links in the graph from a single person labeled “spouse” even if the person has only one spouse. GRAPH adds many other links in a similar manner. Additionally, GRAPH places a loop (a link from a node to itself) on every candidate and auxiliary node. This link, generally, denotes how strong a fit GRAPH initially believes the node is that corresponds to the document entity.

Once the links are in place, GRAPH places an initial mass on every NON-NIL candidate node and auxiliary node. In the case that at least one auxiliary node occurs, NON-NIL candidates contain 75% of the total mass on the graph and 25% on the auxiliaries; otherwise, all of the mass was initially placed on the NON-NIL candidates. The NIL candidate always starts with a mass of zero. GRAPH returns a result from the graph walk by simply considering the acceptor node with the most mass on it at the end of the walk.

To allow for NIL responses to queries, there must also be some links that connect the NIL node to the rest of the graph. Every node that was not an acceptor has a link to the NIL nodes based on the proportion of its links that were considered dead ends, that is, links that connect to nodes that do not link back to any candidate or auxiliary node in the graph. To make sure that mass can escape from the NIL node, it also links back into the graph in a way that redistributes a certain portion of its mass after each step in a proportion identical to that of the initial masses on the candidates and auxiliary nodes.

Another important aspect in the graph walk is the edge weights, which an error backpropagation algorithm [3] produces. The exceptions to this come with the links between candidates and their acceptor nodes, where GRAPH simply forces the issue: at each step, the candidate would release 30% of its current mass into its acceptor. With these weights and the exceptions in mind, it is easy to compute how mass flows into and out of nodes at each step of the graph walk.

4.1.6 System CLINK

CLINK examines the candidate set of filtered Wikipedia entries from the filtering process where the Wikipedia page must of the same entity type as the reference entity in the document (just as described in TOP). CLINK returns a

Wikipedia page if the title of the Wikipedia page matches the query term entity, where a match includes name parts and abbreviations. The algorithm returns a Wikipedia page if the query term occurs in the text of the Wikipedia article and there is an html link nearby. CLINK is essentially the system from TAC 2009 [6]. The major difference is in the filtering process, which provides CLINK with a different set of candidate Wikipedia entries.

5 PERFORMANCE

5.1 Filter Performance

Figure 5 summarizes the overall result of when the correct Wikipedia exists in the filtering set; this is our system recall. The figure illustrates the results for all documents as well as for blog and newswire documents.

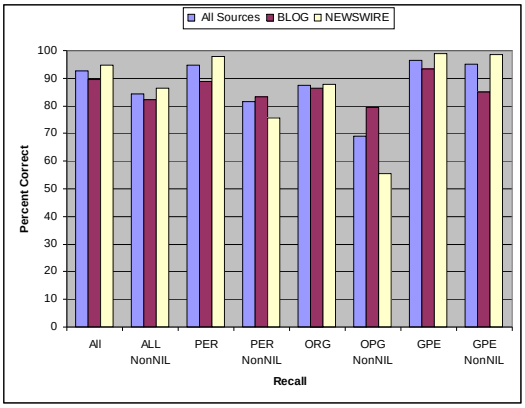

Figure 5: The percent correct of whether the correct Wikipedia page was in the filtered set produced by the filtering process for the all of the data as well as for newswire and blog data, and for each of the entity types.

The filtered set from the filtering process contains no more than 15 Wikipedia entries about 80 percent of the time, and contains no more than 25 entries about 90 percent of the time. The entity linking algorithms process this small filtered set rather than the 860,000 Wikipedia entries. The new filtering algorithm, when compared to our TAC 2009 system [6], is extremely accurate as the correct answer occurs in the filtered set pages about ninety-two percent of the time. The filtering algorithm had difficulty with ORG entities, and this led to reduced performance for the other algorithms since the correct answer is not in the filtered Wikipedia entries.

Figure 5 further illustrates the results when the correct answer is a Wikipedia entry (NON-NIL) or not a Wikipedia entry (NIL). The percent correct for NIL entries is always 100 percent since the filtered set always contains NIL, therefore is not in the figure. As depicted earlier in Figure 3 all of our algorithms use the results of filtering to determine the final Wikipedia page. Figure 5 further illustrates that the filtering process does better on newswire than blog documents.

5.2 Linking Performance

Figure 6 depicts the results from the TAC 2010 evaluation for the different entity linking algorithms. TOP, FACT, and GRAPH do not use information in the Wikipedia article, while CLINK uses link information within Wikipedia text. The vertical axis is the percent correct. The horizontal axis denotes the different experiments, which consist of the results when using all of the data and for each of the three major entity types (PER, ORG, and GPE). Furthermore, Figure 6 illustrates the results for when the correct answer was an existing Wikipedia page (NON-NIL) or not an existing Wikipedia page (NIL).

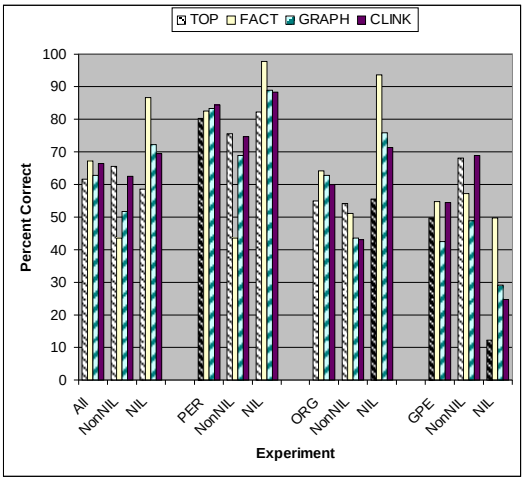

Figure 6: Official results in percent correct for the four different entity linking systems (TOP, FACT, GRAPH, CLINK). Also contains scores for each system on all data and for each entity type as well as whether the answer was NIL or NON-NIL.

Although the filtering algorithm did fairly well in having the correct answer in the filtered set of Wikipedia entries, the most likely Wikipedia entry from the filtered set was not the correct answer as TOP illustrates. FACT did fairly well at detecting NIL Wikipedia answers across the different entity types, as designed. Moreover, FACT tends do to best for entity types other than GPE's. GRAPH performs pretty well on the PER entities, and especially has trouble detecting GPE NIL entities. This result illustrates, arguably, that the overall connectivity of the Wikipedia links among the entities. The PER entities, most likely, have links that form a unique circuit among the associated entities. The GPE entities, however, cannot be discriminated by their neighboring links. CLINK, our TAC 2009 entry [6], uses only Wikipedia article link information, and does best with PER and GPE NON-NIL entries.

To further gain insight into our algorithms and identify weaknesses, we further analyze the TAC 2010 with newswire (Figure 7) and blog (Figure 8) documents. The entity linking systems did much better on newswire than blog documents, especially for NIL answers.

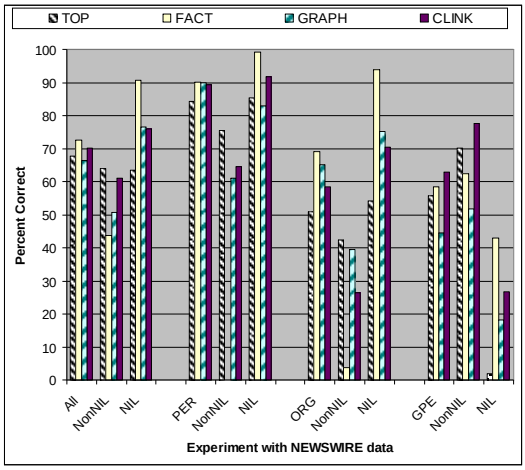

Figure 7: Newswire results in percent correct for the four different entity linking systems (TOP, FACT, GRAPH, CLINK). Also contains scores for each system on all data and for each entity type as well as whether the answer was NIL or NON-NIL.

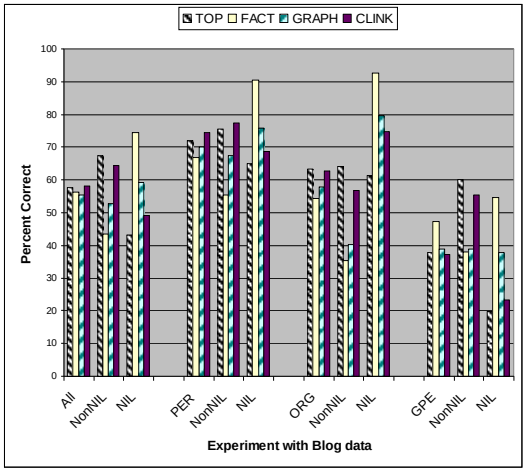

Figure 8: Blog results in percent correct for the four different entity linking systems (TOP, FACT, GRAPH, CLINK). Also contains scores for each system on all data and for each entity type as well as whether the answer was NIL or NON-NIL.

For newswire documents, Figure 7 illustrates that TOP did fairly well when compared to the other algorithms on NON-NIL answers, which indicates that the other algorithms had trouble picking the correct answer from the filtered set. The FACT algorithm does fairly well on NIL answers, except on GPE entities.

Figure 8 illustrates the TAC 2010 entity linking results for the blog documents. The algorithms generally perform worse on blogs than newswire documents since the filtering algorithm did not contain the correct answer in the filtered set of Wikipedia entries as well as it did for the newswire documents. Regardless, TOP continued to perform best on the NON-NIL answers, and FACT did best on the NIL answers. When compared to the newswire documents, FACT did much better detecting GPE NIL answers.

6 ERROR ANALYSIS

We describe some of the more common mistakes that the entity linking algorithms encountered for TAC 2010. Although the new filtering process is much better than our TAC 2009 submission, it does not provide the correct answer in the filtered set 166 out of 2250 times. This, of course, only presents a problem in the NON-NIL case. Nevertheless, we achieved our goal of improving our system's ability to detect when an entity does not exist in Wikipedia (NIL). Next, we found that it is sometimes the case that the entity extractor incorrectly classifies entity types within the document. Such errors cause the algorithm to throw away a correct response, thus making any later processing ineffective. All of our linking algorithms use the reference entity type to further filter Wikipedia entries from the filtered set. Among the queries for a NON-NIL Wikipedia entry, the entity extractor tags incorrectly 85 times, and 66 of these mistakes were for queries where the entity in question was a GPE.

In GRAPH, there is an additional pruning step prior to the walk where overlap between the infobox and the document was checked. This step causes quite a few problems such as there were 261 incorrect responses returned based on the notion that more overlap implied a better match. Among these errors, there are 188 GPE's, 50 ORG's, and 23 PER's, which is another type of errors that GPE entities dominate.

We continue to notice, but find somewhat hard to quantify, that the linking algorithms continue to have difficulty is with acronyms. The acronyms do not appear to have problems with recall from our filtered set. Regardless, the linking algorithms did not determine the correct answer from the filtered set.

7 FINAL COMMENT AND DIRECTIONS

The filtering algorithm that was used for all systems this year performed remarkably well while considering only the titles of the Wikipedia entries. In addition to returning a set with the correct answer in many cases, the filtered set passed to the other algorithms was generally small. Overall, this portion of the system has a strong performance, and allows entity linking algorithms to examine a small filtered set of Wikipedia entries.

As one of the goals going into this year's evaluation, we wanted to improve our performance over last year's on the NIL

cases. Moreover, we wanted to develop algorithms where the Wikipedia article text is not used. We did improve the NIL and NON-NIL results as compared to TAC 2009, especially for PER queries. As many of our strides came in the PER area, we need to improve our system for GPE queries. As we mentioned above, there is also a fair amount of work to do in handling acronyms, and this would be most applicable to the GPE and ORG queries, but a more thorough analysis would certainly help PER queries as well. Additionally, there could still be some work done in the area of contextual matching and tuning graph parameters to achieve better results in the areas of ORG's and GPE's.

8 REFERENCES

[1] Bikel, D, Miller, S., Schwartz, R., Weischedel, R. (1997) "NYMBLE: An High-Performance Learning Name-finder." In Proceedings of the Fifth Conference on Applied Natural Language Processing, ACL, pp.194-201.

[2] Miller, S., Ramshaw, L. Fox, H., and Weischedel, R. 2000, A Novel use of Statistical Parsing to Extract Information from Text. In NAACL, Seattle, WA pp.226-233.

[3] E. Minkov, "Adaptive Graph Walk Based Similarity Measures in Entity-Relation Graphs", Doctoral Thesis, Carnegie Mellon University.

[4] LEMUR URL: http://www.lemurproject.org.

[5] NIST ACE Evaluation, URL: http://www.itl.nist.gov/iad/mig/tests/ace/

[6] P. Schone, A. Goldschen, C. Langley, S. Lewis, B. Onyshkevych, R. Cutts, B. Dawson, J. MacBride, G. Matrangola, C. McDonough, C. Pfeifer, M. Ursiak, "TCAR at TAC-KBP 2009", TAC 2009 Workshop , Gaithersburg, MD, pp. 339-350.

[7] P. Schone, J. Townsend, C. Olano, T.H. Crystal. "Text Retrieval via Semantic Forests," TREC-6, Gaithersburg, MD. NIST Special Publication 500-240, pp.761-773, 1997.

[8] Philpot, Andrew, Eduard Hovy, and Patrick Pantel, "The Omega Ontology." Proceedings of IJCNLP 2005.