SMU-SIS at TAC 2010 - KBP Track
Entity Linking

Swapna Gottipati
School of Information Systems
Singapore Management University
Singapore
swapnag.2010@smu.edu.sg

Jing Jiang
School of Information Systems
Singapore Management University
Singapore
jingjiang@smu.edu.sg

Abstract

Entity linking task is a process of linking the named entity within the unstructured text, to the entity in the Knowledge Base. Entity liking to the relevant knowledge is useful in various information extraction and natural language processing applications that improve the user experiences such as search, summarization and so on. We propose the two way entity linking approach to reformulate query, disambiguate the entity and link to the relevant KB repository. This paper describes the details of our participation in TAC 2010 - Knowledge Base Population track. We provided an innovative approach to disambiguate the entity by query reformulation using the query context and Wikipedia knowledge through heuristic approach. We participated in Entity-Linking task using KB text and Entity-Linking task without using KB text tasks. We developed several entity linking engines to evaluate our solution. We compared our methods with a baseline approach and analyzed the experimental results. For both the tasks our system performance is competitive with 76% and 75% mean average scores respectively.

1 Introduction

Online readers often need to look up phrases, usually named entities (e.g. person, location, organization) and the details of these entities. In the case of news articles or web articles or business documents which are online, it would be useful if those named entities were linked to relevant encyclopedic information. The ability to link the named entities to the relevant knowledge is useful in various information extraction and natural language processing applications such as search, summarization and so on. In this paper we study the Entity Linking problem.

Entity Linking task as defined by the TAC KBP 2010 refers the process of linking the entity candidates in the documents to their representation in the knowledge base or correctly determine that the named entity does not have any entry in the KB.

The major challenges in this task are resolving the named entity ambiguity and handling textual variations. Named entity ambiguity refers to the cases where more than one entities share the same name (polynemous words), e.g. *George Bush* refers to the President of the United States as well as there are many individuals with the same name in other professions.

A textual variation challenge arises when the acronyms are used or when the stage names are used for an entity instead of the actual name. Many times a person is more popular with his stage name rather than the real name, e.g. *Vanilla Ice* is the stage name (alias) for Robert who is the rapper. Another example to illustrate the acronym is *MGM*, which refers to Metro-Goldwyn-Mayer Inc.

Another challenge that is faced by this task is the availability of encyclopedic information. Most of the times we are able to find the entity in the knowledge base. Encyclopedic information challenge rises when the entity information is not found in the knowledge repository. For example, when the entity is very new, the chances are almost negligible to find the relevant information in encyclopedia.

Bunescu et al. (BM, 2006) and Cucerzan (CS,

2007) explored the entity linking task and ranked the similarity using the the Vector Space Models. They took a classification approach together with the novel idea of exploiting Wikipedia knowledge. In their pioneering work, they used Wikipedia's category information for entity disambiguation. They show that using different background knowledge, we can find efficient approaches for disambiguation. In their work, they took an assumption that every entity has a KB entry and thus the NIL entries are not handled.

As many other researchers, Zhang et al. (ZSC, 2010) took an approach of classification and used a two-stage approach for entity liking. They proposed a supervised model with SVM ranking to filter out the candidates and deal with disambiguation effectively. For entity diambiguation they used the contextual comparisons between the Wikipedia article and the KB article. Although we used a similar approach, we used the named entity comparisons rather than the entire context. However, their work ignores the possibilities of acronyms in the entities. Also, the ambiguous geo-political names are not handled in their work.

Mark et al. (MPD, 2010) proposed an entity disambiguation method and with the approach that large number of entities will be unlinkable, as there is a probability that the relevant KB entry is unavailable. Their algorithm for learning NIL has shown very good results. But their proposal for handling the alias name or stage name via multiple lists is not scalable. Unlike their approach, we use the Wikipedia knowledge to handle the stage names as well and thus this gives an optimized solution to handle alias names. Similarly, for acronyms we used the Wikipedia for entity disambiguation.

Unlike other approaches, Zheng et al. (ZFM, 2010) took a learning to rank approach and compared list-wise rank model to the pair-wise rank model. They achieved good results on the list-wise ranking approach. They handled the acronyms and disambiguity through wiki redirect pages and the anchor texts. Unlike their machine learning approach, we use manually defined heuristic rules to link the entities to the knowledge base. Many researchers have proposed many efficient solutions in their works and we integrate them to achieve our objective.

For the TAC task, the input documents are the collection of news articles, web articles and so on. KB is a set of entities represented as nodes. The fine grained named entity types are Person, Organization and Location. The output is the corresponding entity node in the knowledge base. Given a phrase or named entity in the article, our objective is to link it to the relevant node in the KB and if the entity node is unavailable we return NIL.

In this work we introduce an entity linking model that links entities to corresponding knowledge base entry. Our proposed model is two stage approach: query reformulation and entity linking. Query Reformulation: reformulates the query and handles the entity ambiguities. Entity Linking: links the entities to the relevant knowledge base by handling entity ambiguities.

This paper is organized as follows. Section 2 presents the overview of our proposed entity linking model. We explain the details of the entity linking engine in section 3. We explain our data sets and some pre-processing techniques in section 4. In section 5, we explain our experimental settings, evaluation approaches, results and analysis. Finally, we conclude in the section 6.

2 Our Model Overview

In this section, we present the overview of our proposed entity linking method. The figure depicts our proposed model. Our method has two major steps. First, we perform query reformulation. The goal of this process is to generate a better representation of the query, thus minimizing the ambiguities. The query is re-formulated with two techniques: *Context-based Disambiguation* and *Knowledge-based Disambiguation*. The first technique uses the context of the entity to disambiguate entities (shown in the flow as 1.1) and second technique uses the external knowledge for disambiguation (shown in the flow as 1.2). This is a sequential process and results in the ambiguity minimized reformulated query.

The second major step is the entity linking with and without using the wiki text associated with the KB text. The entity is linked to KB with two techniques: *KB-Wiki mutual similarity* and *KB-Query mutual similarity*. If the wikipedia page is available

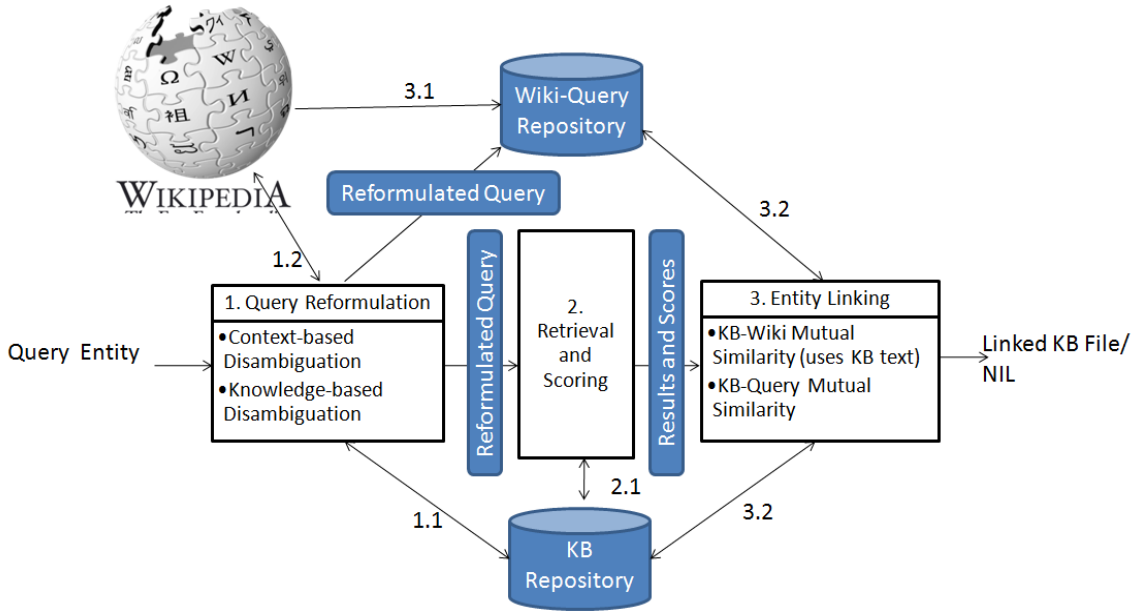

Figure 1: Proposed Entity Linking Framework

for the querying, we perform the KB-Wiki mutual similarity ranking and retrieve the highest relevant document. If the wikipedia page is not available, we perform the KB-Query mutual match and retrieve the relevant document (shown in the flow as 3.2). The wiki-query repository is the wikipedia content for the reformulated query. We built this repository using Wikipedia pages that are relevant to the query (shown in the flow as 3.1).

To bridge these two tasks, we need a query *retrieval and scoring* process. We submit the reformulated query to the search engine to retrieve the top k matches and the relevant scores to the query from the KB repository (shown in the flow as 2.1). The results and scores from the search tool are used for the first level similarity ranking. The top k results from the search engine are represented by result set, R={ *rq*, *kf*, *r*, *rs* } where *rq* is reformulated entity query, $kf \in KB$ is the kbFile name, *r* is the rank of the result and *rs* is the result score. The details of our approach are explained in the next section.

3 Proposed Entity Linking Model

In this section, we present the details of our proposed engine for entity linking task. We first explain the query reformulation model and then the entity linking process - with and without using the wiki text associated with the KB node.

3.1 Query Re-Formulation Model

Several cases need the query reformulation as described below to handle entity ambiguity. We use the samples from the experimental queries to explain them.

1. Case 1: Full name of the person may sometimes resolve ambiguities, e.g. For the given entity, Shabazz, the full name is Malik Zulu Shabazz and it is unambiguous. But given the entity Jordan and the full name, Micheal Jordan still remain ambiguous.

2. Case 2: The geo-political name may exist in various places. So, combining the location with its state/country may resolve ambiguities in location names, e.g. Cambridge is located in Massachusetts, in the United States as well as Ontario, in Canada.

3. Case 3: Acronyms are most ambiguous terms that can confuse any search engine if used directly, e.g. AZ has 14 articles in Wikipedia[1].

[1]http://en.wikipedia.org/wiki/AZ

Sno	Query	Query Context	Re-formulated Query
1	AZ	Shelley Farringer is back in AZ saturday	Arizona
2	Bucharest	which was filmed in Bucharest, Romania in the spring	Bucharest Romania
3	Laguna Beach	planning on driving down to Laguna Beach to hang out	Laguna Beach California
4	Rangoon	while gunfire echoed around Rangoon last	Yangon
5	Anton Johnson	POSTER "Anton Johnson"	Anton J Johnson
6	Chris Breezy	Chris Breezy popped the question to Rihanna	Chris Brown (American singer)
7	Eretz Yisrael	push for the division of Eretz Yisrael	Land of Israel
8	Jackman	Hugh Jackman is Jacked!!	Hugh Jackman
9	Shabazz	Malik Zulu Shabazz of the New Black Panther Party	Malik Zulu Shabazz
10	William Ayers	Weather Underground honcho William Ayers	Bill Ayers
11	AMPAS	The AMPAS will be having a meet the Oscars promo.	Academy of Motion Picture Arts and Sciences
12	NY Times	According to the NY Times,	The New York Times
13	Reserve Bank	In January, the Reserve Bank of India held interest rates	Reserve Bank of India
14	Cornell	traveled Ithaca NY to visit our friend at Cornell	Cornell University

Table 1: Sample reformulated queries based on context-based and knowledge-based disambiguation approaches.

But for the given article in the experiments, AZ refers to Arizona. Expanding the acronyms may resolve the entity ambiguities.

4. Case 4: Official name or alias names are more popular than the original names. The KB entities are sometimes represented by these names, e.g. For the given entity William Ayers, the alias name is Bill Ayers.

Case 1 and 2 are handled by *Context-based Disambiguation*. Case 3 and 4 are handled by *Knowledge-based Disambiguation* and the details are explained below.

3.1.1 Context-based Disambiguation

We use the context of the query, which is the sentence where the query is located. The context of the query is found in the corresponding news article or web article associated with the query. These lines are extracted from the query document and processed to reformulate the query. The context helps resolve some of the entity ambiguities as explained below.

Case 1 Resolution To resolve the case 1 ambiguities, we use the named entity tagging process. This process tags the name found in an given text. First, the sentence where the query entity is found, is extracted from the corresponding query document. Second, this line is tagged using the NER tagger to find the type of the entity and the complete name of the entity.

The tagger outputs two elements: *the full name of the entity* and *its entity type*. Moreover, the tagger can even extract the full name $\langle fn, mn, ln \rangle$ of the person if a partial name is used in the query, i.e. if either $\langle ln \rangle$ or $\langle fn \rangle$ is used for the query entity.
E.g. given the query - *Hughman*, and the corresponding query line in the associated article - *Hugh Jackman is Jacked!!*.
NER tagger tags the line to:
$\langle PERSON \rangle HughJackman \langle /PERSON \rangle$.

Hence the reformulated query entity is *Hugh Jackman*.

To achieve the above, we used Stanford NER Classifier from Krishnan et al (VC, 2000). It is a named entity recognizer that can label the following types of entities: PERSON (PER), ORGANIZA-

TION (ORG), and LOCATION (GPE). It uses the Conditional Random Field (CRF) sequential model to identify named entity tokens in sentences.

Case 2 Resolution Geo-political ambiguities occur when the same location name exists in various countries or states. Most of the times, query articles consist of location followed by the country or state to which it belongs to. So this information is used for the query reformulation.

E.g. given the query - *Cambridge* and the corresponding query line in the query file - *We visited Cambridge, Massachusetts last month*. The second location within the line is extracted which is Massachusetts. And hence the reformulated query entity is *Cambridge, Massachusetts*.

3.1.2 Knowledge-based Disambiguation

Knowledge-based model uses an external repository to disambiguate the entities. In our case, we use *Wikipedia (Wiki)* as an external repository. This can be always replaced by any such knowledge sources like Uncyclopedia, DBpedia etc,. *Wiki* provides a standard representation for the ambiguous entities that has been used by our model extensively. Wikipedia is a huge repository with various properties like search pages, acronyms list, disambiguation pages etc., used by Mihalcea et al (RM, 1972). The knowledge-based disambiguation method used by Montoyo et al (MSR, 2005), disambiguates nouns by matching context with information from a prescribed knowledge source.

Case 3 Resolution Case 3 ambiguities of acronyms can be resolved by expanding them and the expanded acronyms can improve the accuracy of the search engine. Wiki provides ample resources to expand the acronyms. Creating a repository for such acronyms will aid the reformulating process. The state abbreviations for various locations are extracted from Wiki[2] and stored in an acronym repository. Given the query, the line with the query entity is extracted and tagged with NER to find the entity type. If the entity type is GPE, the location name associated with the acronym is used for the query formulation.

E.g. given the query - *AZ* and the corresponding query line in the query file is *Shelley Farringer is back in AZ saturday*. AZ is the acronym for the location Arizona. Hence the reformulated query entity is *Arizona*.

To expand the organizational acronym, we use Wikipedia. Mark et al (MPD, 2010) proposed the entity type specific lists, whereas, we used the Wikipedia acronym knowledge. Instead of creating an acronym repository, the acronym is directly searched in Wiki. The query entity is searched in Wiki using the URL "http://en.wikipedia.org/wiki/*acronym*".

E.g. Given the query *q = AMPAS*, the corresponding Wikipedia page *http://en.wikipedia.org/wiki/AMPAS* exists. Hence the reformulated query *rq = Academy of Motion Picture Arts and Sciences*.

Whereas, given the query *q = ICC*,
the page *http://en.wikipedia.org/wiki/ICC* doesn't exist. Hence the reformulated query *rq= ICC*.

Case 4 Resolution To find the stage name or the alias name of the person, we use Wikipedia pages. The query entity is searched in Wiki using the URL "http://en.wikipedia.org/wiki/*name*". If the page is found then the reformulated query is set to the Wiki title of the page. If the page is not found, the reformulated query is the same as the actual entity. The query is reformulated as shown below.

E.g. Given the query *q = William Ayers*, the corresponding Wikipedia page *http://en.wikipedia.org/wiki/William_Ayers* exists. The Wiki title is Bill Ayers. Hence the reformulated query *rq = Bill Ayers*.

Another example is given the query *q = Chris Breezy*, the corresponding Wikipedia page *http://en.wikipedia.org/wiki/Chris_Breezy* exists. The Wiki title is Chris Brown (American singer). Hence the reformulated query *rq = Chris Brown (American singer)*.

$$rq = \begin{cases} wt & \text{if wiki page exists} \\ q & \text{if wiki page does not exist} \end{cases}$$

where *q* - query entity, *wt* - Wiki title from the Wiki page

Table 1 shows some reformulated queries by applying the context-based and knowledge-based reformulation techniques.

[2] http://en.wikipedia.org/wiki/List_of_U.S._state_abbreviations

$$aScore = \begin{cases} 1 & \text{unambiguous entity} \\ between\ 0\ and\ 1 & \text{ambiguous entity} \\ -1 & \text{NIL entity} \end{cases}$$

Figure 2: Entity ambiguity score.

3.2 Entity Linking using KB text

For certain entities, the retrieval tool may return empty results. Such entities are the NIL entities. For the remaining entities, the linking is handled in two ways, first using the KB text and the second without using the KB text. In this section we explain the entity linking using KB text task.

We implemented a two phase entity linking approach: *KB-Wiki Mutual Similarity* followed by *KB-Query Mutual Similarity*. The first technique is used for non-ambiguous entities and the second technique is for the ambiguous entities. Finally, the entity linking, *e* is derived at the end which links the entities to KB or to NIL. Wiki disambiguation pages are used to find the ambiguity status of the query. And to determine the entity status, we need the ambiguity scores for the queries which are calculated using *Entity Ambiguity Scoring* technique as shown below.

Parameter	Description
q	entity query
rq	reformulated query
rqtype	entity type of query
kf	name of the file in KB
kft	text in the file *kf*
rs	similarity score
wf	name of the file in Wiki
wft	text in the file *wf*
e	entity linking
aScore	ambiguity score
kname	name of KB entity
ktype	entity type of KB entity

Table 2: Notation

Entity Ambiguity Score The URL for the reformulated query *rq* is generated using "http://en.wikipedia.org/wiki/*rq*". The HTML content is extracted and stored in WikiQueryRepository, *WF* where the filename is the query entity name. For ambiguous entities, wiki returns the disambiguation pages. The ambiguity score of the query entity is calculated from its corresponding wiki page in WF as follows,

1. Case 1: For the given query *rq*, if the wiki page is disambiguation page, $ae = \{e_1, e_2, ..e_i\}$, is the set of the entities listed in the page. Ambiguity score $aScore = 1 - \frac{1}{|ae|}$.

2. Case 2: For the given *rq*, if the wiki page is found and it is not disambiguous page, *aScore*=1.

3. Case 3: For the given *rq*, if the wiki page is not found page, *aScore*=-1.

The entity ambiguity score is derived as shown in the Figure 2.

3.2.1 KB-Wiki Mutual Similarity

The technique is applicable for the queries, which are unambiguous i.e, the corresponding *aScore*=1. The idea here is to first extract entity sets from both wiki file text - (*wft*) and KB file text - (*kft*) associated with the *rq*, using the tagger. Second, the similarity between these two files is calculated and the best matched KB file linked to the given query entity. To extract the entity sets, we should tag the text using the NER tagger. We explain the details of this idea below.

KB and Wiki files tagging For the given query *rq*, if *aScore*=1, the wiki file, *wf*, the file whose name is equal to *rq* is extracted from the WikiQueryRepository, *WF* where $wf \epsilon WF$. The wiki text is now tagged using NER tagger. The ouput is the tagged wiki text, $wft = \{te_1, te_2, ..., te_i\}$ where $te_i = (en, et)$ is a tagged entity, *en* is entity name and *et* is entity type. Similarly, for each KB file from the search results, *kfi* where ($0<i<k$) is extracted from the KBFileRepository, KF and tagged using NER tagger i.e

$$e = \begin{cases} \underset{kf_i}{argmax}\ sim(wft, kft) & \text{sim(wft,kft)}>0\ and\ aScore = 1 \\ NIL & \text{sim(wtf,kft)=0 or } rs \leq t \text{ or } aScore = -1 \\ kf & \text{kname=rq and ktype=rqtype and } 0 < aScore < 1 \end{cases}$$

Figure 3: Entity Linking Derivation.

kft $= \{te_1, te_2, .., te_i\}$. *k* is the number of files that match the *rq*.

Grep the PER, GPE and ORG entities from *wft* and *kft* separately. i.e. $gW = \{gpe_1, gpe_2, .., gpe_i\}$, gpe_i is the entity name of the GPE type from the tagset.

Similarity Score Calculation To calculate the similarity between the *wft* and *kft* the following equation is used.

$$sim(wft, kft) = \frac{|gW \bigcap gK|}{|gW|} + \frac{|pW \bigcap pK|}{|pW|} + \frac{|oW \bigcap oK|}{|oW|} \quad (1)$$

where *pW*, *gW*, *oW* are the PER, GPE, ORG entity sets in *wft* respectively and *pK*, *gK*, *oK* are the PER, GPE, ORG entity sets in *kft* respectively. The query is mapped to the best suitable KB file which has the maximum *sim(wft, kft)* score of all the kfiles (kf_i) retrieved from the results of the given query, *rq*.

3.2.2 KB-Query Mutual Similarity

This technique is applicable for the queries, which are ambiguous i.e, the corresponding *aScore*<1. For those queries, if *aScore*=-1, i.e. when there is no corresponding *wf*, the queries with the retrieval score less than the threshold *t* are considered to be the **NIL**. If the score is greater than the threshold and for those queries where 0<=*aScore*<1, i.e., wiki returns disambiguation pages. Similar to Mark et al, the KB entity types are used in linking. But we further applied some heuristics to resolve the ambiguity with the disambiguation technique as explained below.

Disambiguate Technique The result set R for top k results, { kf_1,.... kf_i } from search tool is used in entity linking algorithm. From each KBfile of the results, $kf_i \in KF$, the name, *kname* and type, *ktype* are extracted. If *ktype* is UKN the NER tagger is used for the tagging the *kname*.

1. Case Exact-Match: If the *kname* directly matches the reformulated query *rq*, the kf_i is the link file.

2. Case Relatedness-Match: If the *kname* doesn't match, best match of *kname* and *ktype* to the reformulated query entity name and its entity type respectively is extracted and the corresponding kf_i is the link file.

If the *kname* directly matches the reformulated query *rq*, the kf_i is returned else the best match of *kname* and *ktype* to the reformulated query name, *rq* and its entity type, *rqtype* respectively is used.

3.2.3 Entity Linking

The results from both the above techniques are used to link the query entity to the KB file. Entity linking, *e* is derived as shown in the Figure 3.

3.3 Entity Linking without using KB text

This is an additional task from TAC 2010 where the objective is to link query entities to the relevant KB files without accessing the KB text. The idea here is very similar to KB-Query mutual similarity. Additionally, we re-calculate the similarity scores for 3 cases to link the entities as shown below.

1. Case NIL: If the result score, *rs* from R is less than threshold *t* then the result is skipped and NIL is returned. Hence, similarity score *sim(kf,rq)*=0.

2. Case Exact-Match: If the *rq* matches with the *kbname*, then the corresponding *kfile* is linked to the entity. Hence, similarity score *sim(kf,rq)*=1. This is very similar to Zhang et

al and Mark et al idea on the exact title match to the query name.

3. Case Relatedness-Match: If *rq* partially matches the *kname* and *rqtype* matches/differs the *ktype* the corresponding similarity score is calculated as shown below. This logic is implemented until the results list is exhaustive. If no match is found, NIL is returned where *sim(kf, rq)*=0.

$$sim(kf, rq) = rs + \alpha \cdot nS + \beta \cdot tS$$

where *nS* is the score given for the name-match,

$$nS = \frac{\vec{rq} \cap \vec{kname}}{\vec{kname}} \quad (2)$$

and *tS* is the score for the type-match,

$$tS = \begin{cases} 1 & \text{ktype matches the NER TAG} \\ 0 & \text{else} \end{cases}$$

α and β are the experiment control factors.

Entity Linking Finally, the entity linking, *e* is derived from the above similarity scores as shown

$$e = \begin{cases} kf_i & \text{if } sim(kf,rq)=1 \text{ or } sim(kf,rq)>t \\ NIL & \text{if } sim(kf,rq)<t \end{cases}$$

The last case, relatedness-match handles the disambiguation problem using the type of the entity i.e disambiguation of same name for various entities is handled with the qtype-ktype mutual match (map the entity types of the query and the KB node).

4 Data Source

In this section, we shall explain the techniques for pre-processing the KB for the entity linking process. KB from TAC is represented in XML where each XML consists of hundreds and thousands of nodes. This representation may delay the search and hence we processed the large XML into the fine granular structures. Finally, the resulted KB is indexed for the searching the queries. The details are shown below.

4.1 Pre-processing KB

Due to the lengthy KB file structure (thousands of node), which will have poor retrieval system performance, the indexing granularity is decided on the entity nodes of the XML file. Each document node is prepared by converting the KB nodes into an individual HTML file. The file name (kbFile name) is same as the DOCNO (node name in KB) and the HTML content is converted to text format using HTML2Txt [3]. The title, name and type are stored along with the text in the file. The document collection is represented by *KF*, KBFileRepository.

4.2 Indexing KB

Lemur - Indri [4] is a search engine that provides state-of-the-art text search for a large scale text collection. This tool supports the HTML text representation. IndriBuildIndex [5] application is used for indexing KB. The stop words [6] were removed. For each entity, its name, title and type are indexed as fields along with the main indexing.

4.3 Data Normalization

For better query retrieval, the reformulated query is normalized using the following techniques before querying the search tool.

1. Capitalization: Since all the 3 entity types - PER, GPE and ORG are proper nouns, the queries are converted to capitalized entities. For the organizations, which are represented by acronyms such as *CCC, ITL* etc., this rule is not applicable.

2. Deaccent and Diacritics: Queries that consists of diacritics on the characters are normalized by removing the diacritics. For eg: Cancùn is converted to Cancun.

3. Other issues: Special characters like ., &, (,), _ etc are removed from the query.

5 Experiments

In this section, we describe our experiments and compare the results with our baseline approach.

[3] http://kt.ijs.si/Dunja/textgarden/
[4] http://www.lemurproject.org/indri.php
[5] http://www.lemurproject.org/lemur/indexing.php#IndriBuildIndex
[6] www.ranks.nl/resources/stopwords.html

Queries	Base Model	W/ KB Text	W/O KB text
2250 queries	0.6502	0.7649	0.7547
1020 non-NIL	0.5588	0.6559	0.6588
1230 NIL	0.726	0.8553	0.8341

Table 3: Overall Micro-average scores

Queries	Base Model	W/ KB Text	W/O KB Text
750 ORG	0.6013	0.72	0.7187
304 non-NIL ORG	0.4572	0.4605	0.4605
446 NIL ORG	0.6996	0.8969	0.8946
749 GPE	0.5728	0.6742	0.6449
503 non-NIL GPE	0.666	0.7555	0.7614
246 NIL GPE	0.3821	0.5081	0.4065
751 PER	0.7763	0.9001	0.9001
213 non-NIL PER	0.4507	0.6995	0.6995
538 NIL PER	0.9052	0.9796	0.9796

Table 4: Micro-average scores split by entity types.

TAC-KBP query set contains 2250 queries of which 750 are organizations, 751 are persons and 749 are locations. For the entity linking experiments, we have prepared the baseline model to compare our proposed models for both the TAC tasks - entity linking with and without KB text. We developed 3 systems- Baseline system, Entity Linking using KB text system and Entity Linking without using KB text system. Below we explain the details of our query retrieval and scoring model followed by the three systems.

5.1 Retrieval and Scoring

Indri retrieval model by Strohman et al from Lemur is used for the query search. For each query entity, q, the tool returns result set, R={ q, kf, r, rs }, where q is entity query, $kf \in KF$ is the kbFile name, r is the result rank and rs is the result score.

5.2 Baseline system

For baseline model, we used the queries directly without any reformulation and and fetch the relevant KB files from the search tool. Result set R from search tool with query q is retrieved. We used a threshold t to determine the best match for linking the entity to its representation in the KB collection. After several experiments, t is set to -5.8. Finally, the entity linking, e is retrieved as shown below,

$$e = \begin{cases} kf & \text{if } rs > t \\ NIL & \text{if } rs \leq t \text{ or R is NULL} \end{cases}$$

The indri scores from Lemur are used for the first level ranking. The top k results from the indri are represented by R={ rq, kf, r, rs }. In our case, we set k=5 i.e. $0<r<6$.

5.3 Entity Linking using KB text system

The heart of this technique is extraction of entity sets of wiki-query file text and KB file text. The wiki file is a HTML content which needs to be converted into plain text for tagging process. The wiki file is converted to text format using HTML2Txt. The NER tagger is used to tag this text. For the entity linking formula shown in Figure 3, we set t to -5.8 after studying several experiments.

5.4 Entity Linking without using KB text system

Kname and *ktype* are extracted from kbfile, *kf*. We normalized the *kname* using the data normalization techniques described in Section 4. If the *ktype* is UKN, NER tagger is used for tagging. Due to NER tagger limitation on tagging the GPE and ORG, which has been observed through our experiments, the following heuristic is used to distinguish the GPE and ORG.

If *kname* is greater than 2 words and has been tagged more than twice as GPE or GPE/ORG, the kfile is skipped. The reason is that, this may be entity of type other than PER or GPE or ORG. For e.g., *George Washington in the American Revolution* is tagged twice because it is a history article and not a named entity of type person or location or organization.

5.5 Results

Table 3 shows the micro-averaged accuracy (averaged over all queries) comparison of three models we developed. Base model represents baseline system. Overall accuracy of our model with and without using KB have more than average performance of 76.5% and 75.4% accuracy approximately. This is more than 10% improvement than the basic model and the median scores published by TAC. Table 4 shows the micro-averaged results split by the entity type.

The baseline system with original query entity linking resulted in 65% of accuracy where as, reformulated query helped improving the accuracy by 11% approximately. Queries like, Rangoon, Chris Breezy William Ayers etc whose representation in the KB are Yangon, Chris Brown and Bill Ayers respectively, have shown a significant results with the query reformulation techniques.

The overall score can be improved if one can resolve the geo-political issues. As one can notice the location entities did not perform well in the entity linking task. Particularly, the NIL has a very low performance. This is due to the fact that location names are highly ambiguous and and to disambiguate them we need some more information like the country or state it belongs to. Usually news articles do provide such information but the web articles which are informal content like tweaks lack such information. Similarly, the ORG entities linking has lower accuracy. This is due to the fact that the acronyms are not resolved during our experiments. We concentrated on the GPE acronyms with a wiki acronym list but we did'nt use any such for the ORG. We also observed that the NER tagger has its own limitation in mapping the entity types. This impacted our accuracy as our KB-Query mutual similarity algorithm. PER entities has the highest performance of all and demonstrates best performance as published by TAC 2010 results.

6 Conclusion

We demonstrated competitive results with innovative techniques and algorithms. Our query reformulation technique handled the ambiguity issues and the resulted in the smart query creation. Query re-formulation plays an important role in the success of this approach. We tried to reduce the noise in the query with the query re-formulation context and knowledge techniques and disambiguation techniques. Although, cases of disambiguation like various people with the same name, same location name at various places etc are not resolved completely, out model is still very optimized. We would like to deal with this ambiguity as our future task. We still achieved a macro-average of 76.49% and 75.4% for entity linking with and without KB text respectively. In future, we would like to improve on the disambiguating techniques and still provide a competitive resource optimized solution.

References

T.Strohman and D.Metzler and H.Turtle and W. B.Croft. 2005. *Indri:A language model-based search engine for complex queries*.

A.Montoyo and A.Suarez and G.Rigau and M.Palomar. 2005. *Combining knowledge- and corpus-basedword sense disambiguation methods*.

R.Mihalcea. 2007. *Using Wikipedia for automatic word sense disambiguation*. Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics, Rochester, New York.

Wikipedia Disambiguation. *http://en.wikipedia.org/wiki/Wikipedia:Disambiguation*

Wei Zhang and Jian Su and Chew Lim Tan. 2010. *Entity Linking Leveraging Automatically Generated Annotation*. The 23rd International Conference on Computational Linguistics (COLING)

Mark Dredze and Paul Mcnamee and Delip Rao and Adam Gerber and Tim Finin 2010. *Entity Disambiguation for Knowledge Base Population.* Conference on Computational Linguistics (COLING).

Zhicheng Zheng and Fangtao Li and Minlie Huang and Xiaoyan Zhu. 2010. *Learning to link entities with knowledge base*. Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics

Bunescu, R.C. and M.Pasca . 2006. *Using encyclopedic knowledge for named-entity disambiguation.* Proceedings of the 11th Conference of the European Chapter of the Association for Computational

Cucerzan, S. 2007. *Large-scale named-entity disambiguation based on Wikipedia data.* Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL).

Vijay Krishnan and Christopher D. Manning. 2000. *An Effective Two-Stage Model for Exploiting Non-Local Dependencies in Named Entity Recognition.* 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics.