Turkish Journal of Electrical Engineering and Computer Sciences

Volume 23 | Number 4 Article 10

1-1-2015

Matching named entities with the aid of Wikipedia

ABDULLAH BAWAKID

MOURAD OUSSALAH

NAVEED AFZAL

SEONG SHIM

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2015) 23: 1051 – 1068

doi:10.3906/elk-1304-237

Matching named entities with the aid of Wikipedia

Abdullah BAWAKID[1,*], Mourad OUSSALAH[2], Naveed AFZAL[1], Seong SHIM[1]
[1]Faculty of Computing and Information Technology – North Jeddah, King Abdulaziz University, Jeddah, Saudi Arabia
[2]Department of Electronic, Electrical, and Computer Engineering, School of Engineering, University of Birmingham, Birmingham, UK

Received: 25.04.2013 • **Accepted/Published Online:** 08.07.2013 • **Printed:** 10.06.2015

Abstract: In this paper we propose a novel framework using features extracted from Wikipedia for the task of matching named entities. We present how the employed features are extracted from Wikipedia and how its structure is utilized. We describe how a term-concepts table constructed from Wikipedia and the redirect links are integrated in the framework. In addition, the internal links within Wikipedia along with the categories structure are also used to compute the relatedness between concepts. We evaluate the built framework and report its performance in the applications of word sense disambiguation and named entities matching. The system performance is compared against other learning-based state-of-the-art systems and its reported results are found to be competitive. We also present a method in this paper for named entities matching that is context-independent and compare its results with those of other systems.

Key words: Matching named entities, Wikipedia, features extraction, word sense disambiguation, links analysis, strong links

1. Introduction

With the continuous expansion of the web, research in the information extraction (IE) field to extract structured information from unstructured or semistructured documents has continued to grow. Named entities linking (NEL) is among the most recently investigated areas in IE. The need for NEL is emphasized when examining text documents that often contain references to different types of entities such as people, locations, or organizations. These entities can be difficult to track by language processing systems. For illustration, consider the following sentences:

1. **David** was a king of Israel and is a major figure in the Bible and the Quran.
2. The youngest son of Malcolm III, **David**, spent most of his childhood in Scotland.
3. **David** played active character roles and was associated with the IPTA.

The bolded term, **David**, in the above sentences refers to multiple entities. In the first sentence, it refers to **King David**, while in the second it refers to a king of Scotland called **David I of Scotland**. The reference in the third sentence is made to an actor named **David Abraham Cheulkar**. From the given examples above, it can be noted that a background knowledge base is required to properly distinguish between the different

*Correspondence: abawakid@kau.edu.sa

entities. Without such distinction, all mentions of David would be assumed to refer to a single entity. In natural language processing (NLP) systems, the task of NEL is to resolve the mentions of named entities.

NEL is the task of linking the mentions of named entities within a document to the best representative candidate that was previously extracted from a knowledge base (KB), or NIL if no suitable representative candidate exists in the KB. Figure 1 provides an example for a document sentence containing the entity ***David***. Also shown in the same figure are multiple entities existing in a KB and sharing the same entity name, ***David***. In addition, some entities carry more than one name, as in the case of ***David Abraham Cheulkar*** being named ***David (actor)***. A potential reference is ***David (other)*** as well, which resembles the NIL entry described above when no matching entity is found in the KB.

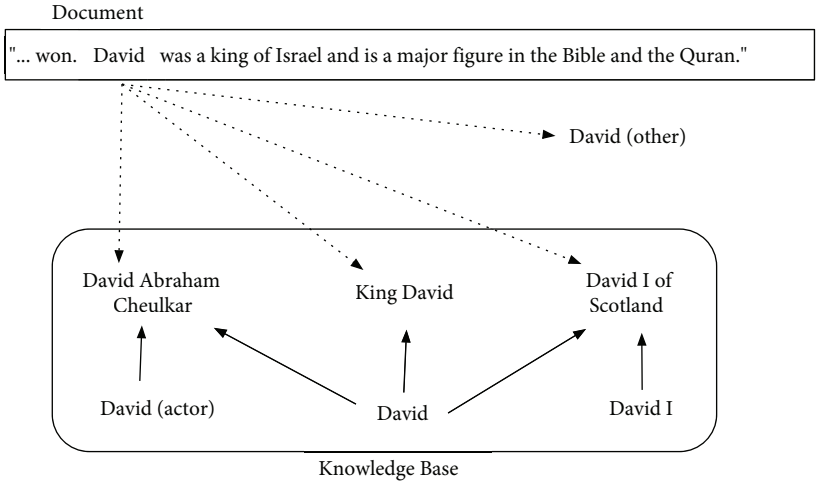

Figure 1. An example showing the task of NEL for the entity David.

The problem of NEL is similar to that of word sense disambiguation (WSD), which has been extensively studied in the NLP community [1–3]. However, it differs in two ways. First, WSD assumes that all entity mentions would have a match in the used reference dictionary, which is often WordNet or Wikipedia. This is not the case with NEL as systems have to consider the possibility of not having a matching entity in the used KB. The second difference is that NEL often relies on KBs containing a much larger number of entries than in WSD. This makes the task of identifying the candidate entity reference more difficult and requires a noisier candidate generation process to improve recall, as reported in [4].

Perhaps the most popular datasets used for evaluating NEL systems are the ones provided in the knowledge population tasks organized by the NIST for the Text Analysis Conference (TAC; http://www.nist.gov/tac/). For the main task, participants are asked to link a set of entities mentioned in text documents to a KB that was extracted from Wikipedia. Over the past 3 years, the tasks have encouraged the NLP research community to implement a wide range of novel NEL systems with different methods and strategies. Some methods rely on simple surface-level features extracted from the source documents to identify the candidate entities for each query [5]. Others perform full-document entity extraction to help rank the candidate entities [6]. A very limited number of systems implemented query-dependent supervised learning-to-rank (L2R) models. In [7], the authors dynamically created a ranking model for a given query by relying on the feature vectors extracted from the nearest query-document pairs to the given query. The results that they obtained indicated a significant improvement in their NEL system. A similar finding was concluded by Zhu et al. [8] in the domain of web searches.

In this paper, we employ a novel Wikipedia-based semantic relatedness measure utilizing the content and structure of Wikipedia for the task of NEL. The work described here differs from most of the previously implemented systems in the literature in that it involves the usage of unsupervised methods for the entities' *selection* and *ranking* stages of the NEL system. After building the system, an evaluation of the output of the system is performed to reflect the performance of the different runs we implemented.

The main questions addressed by this paper are as follows:

- How effective can an unsupervised semantic-based system be in the application of named entities matching when compared to other learning-based systems?
- In a semantic Wikipedia-aided framework, which extracted features lead to a more effective system in the applications of WSD and named entities matching?

The main contributions of this paper are as follows:

- New algorithms leveraging both the content and structure of Wikipedia are proposed for quantifying how semantically related two terms, or a composite of terms, are when compared with others. With the new proposed algorithms, it is also possible to aggregate terms, phrases, or even sentences based on the semantic meaning they carry.
- A framework is introduced for generating and embedding semantic features into text fragments through the use of Wikipedia. This allows for introducing human knowledge in the form of concepts in a new way to different applications in the domain of NLP. The relationship between the introduced concepts can also be quantified with the algorithms we propose.
- The developed framework is extended by adapting its main algorithms in the applications of WSD and named entities matching. An evaluation is performed to compute the performance of the system and the obtained results are reported. Even though the developed system does not make use of any training data, the obtained performance is competitive when compared to other state-of-the-art training-based systems.
- In the application of named entities matching, one of the runs presented in this paper is context-independent. Its performance also competes with the runs of other systems that require the context being present and sometimes are learning-based, too.

This paper is organized as follows: Section 2 gives an overview on related work. In Section 3, we describe the Wikipedia-based framework that we built and how features were extracted from Wikipedia to be used in our system. Section 4 presents the evaluation performed for two different datasets: one was built by us from Wikipedia, while the other was taken from the named entities matching task of the TAC. In the last section, we summarize our findings and conclude this paper.

2. Related work and discussion

Named entities ambiguity has been a common problem in many information retrieval areas such as question answering, automatic document summarization, and web search. Before the availability of large KB corpora to be used as references for named entities, the problem of named entities ambiguity was addressed by cross-document coreference resolution. Mentions of named entities within a document or across several documents were clustered together. Each cluster would then be assumed to reference one single real-world entity. The

seminal work on this was performed in [9], where the authors used the bag of words (BOW) model to represent the contexts of entities. They clustered all references in documents corresponding to the same entity by applying the vector cosine similarity measure. In [10], a similar but refined method was proposed by integrating context similarity into cross-document coreference heuristics previously developed in [11] that simply matches strings and checks entities types. The results obtained in that system were encouraging. In [12], an attempt was made to consider the usage of bigrams to represent the contexts of each ambiguous named entity through the creation of a cooccurrence matrix where the rows and columns represent the first and second words in bigrams. This work was later extended in [13] by adding a rich feature space of biographic facts for extracting personal names.

In all of the above-mentioned methods, similarity between contexts or named entities is computed by examining the statistical cooccurrences of terms. Semantic relations embedded between named entities such as social relatedness between the entities, semantic relatedness between concepts, acronyms, or synonyms are generally not captured with such systems. Therefore, it can be deduced that systems that primarily rely on the BOW-model cannot capture the actual similarity between named entities and their contexts. Background knowledge may be very useful for addressing such a problem.

Recent research has investigated the usage of KBs as a background to many systems for the task of disambiguation. In [14], disambiguation between people's names was performed by examining the content and link structure of the web pages referring to a set of socially related individuals. Subsequent work in [15] proposed a links analysis method based on random walks to address the ambiguity problem. Another graph-based method that relies on links analysis in a semisupervised fashion was presented in [16]. In [17], a similarity measure was developed by collecting named entities' cooccurrences through a web search.

After several KBs became available, such as WordNet [18] and the Open Directory Project (ODP), researchers developed several systems taking advantage of them. In [19], an unsupervised procedure for discovering, disambiguating, and linking named entities with the aid of WordNet was introduced. A method was proposed in [20] that exploits the semantic knowledge embedded in the ODP to compute the semantic similarity between named entities to be used for disambiguation. Another system in [21] proposed another measure for computing the semantic similarity between named entities by comparing the semantic similarity of their contexts after relying on WordNet.

WordNet and the ODP have long been criticized for their relatively limited coverage, while Wikipedia has the advantage of containing vast and highly organized human knowledge in a large number of domains. After the inception of Wikipedia, researchers began shifting their focus to it in many information extraction and data mining applications. For the application of named entities disambiguation, it also allowed for making a reference database to link detected named entities after being disambiguated. The work in [22] exploited a set of features derived from Wikipedia for discovering and disambiguating entities, employing the taxonomy of Wikipedia to model a word-category correlation and using it with a support vector machine (SVM) kernel for disambiguation. In [23], Cucerzan developed a process of maximizing the agreement between a document text and the contextual information extracted from Wikipedia as well as the agreement among category tags associated with the candidate entries. The results were promising. However, that system assumes that all detected entities in the source documents have matching entities in a reference KB, which is not always the case.

The Wikipedia-based methods mentioned above are able to exploit the semantic knowledge within Wikipedia only to a limited extent as they do not take the structural semantic knowledge within Wikipedia into consideration. They also do not fully address the problem of named entities matching as an assumption is

made that a match for a detected entity always exists in the reference KB. In addition, they aim to discover entities within the source documents in addition to disambiguating them. Hence, they have to deal with tradeoff between recall and precision for detecting entities and then disambiguating them. On the contrary, with named entities matching an attempt is made to only map the detected entity to the reference KB or NIL (if no match exists) to achieve a high accuracy. Furthermore, the level of ambiguity for entities in the NEL application is usually much higher than in other applications targeted by the systems mentioned above (as in WSD).

The work applied in this paper can be viewed as an extension of the work performed in [24]. In this paper, we describe in detail how the features were extracted from Wikipedia. We also apply the extracted features to the application of WSD, as was done in the previous paper. However, the extracted features from Wikipedia are applied differently in three more optimized methods, as detailed in Section 4.4. The results obtained here are also different from those reported in [24] and the obtained performance is overall better. In this paper, we also adapt our framework and use the extracted features to address the problem of named entities matching and linking.

3. Wikipedia-based framework

Due to the structure and openness of Wikipedia, it is not feasible to use it without its being preprocessed and analyzed first. In the system that we adopt for extracting semantics from Wikipedia, each Wikipedia article is viewed as a unique concept labeled with the article's title. We form a term-concepts vector that is constructed from the inner content of Wikipedia articles and their titles. Furthermore, we analyze the categories network within Wikipedia and its various links to help compute a relatedness score between any two concepts or text fragments. In the following subsections, we briefly describe the stages that we apply to the downloaded Wikipedia dump to create the required vectors and extract the desired features.

3.1. Preprocessing Wikipedia

The framework described in this paper used the English version of Wikipedia that was created on 16/03/2010. A series of operations were applied to it to make it ready for the following stage before it is analyzed. This includes the removal of unimportant fields or tags in each article, such the non-English characters and the edit history of the article. Some articles in Wikipedia contain summarized facts presented in the form of an Infobox. Although these facts may be easy to comprehend, we chose to ignore them due to the difficulty imposed from parsing and processing these types of summarized facts. In Wikipedia, Infoboxes were designed to provide facts on an article-by-article case. In other words, they were not designed for the widespread formality of showing specific data types in specific formats. For example, there are four different Infobox templates used in Wikipedia for countries, and the attributes used for describing the population statistics year are different in each template. Therefore, we chose to remove Infoboxes as well when processing the inner content of articles.

We also removed articles that were too short by being less than 100 terms in length and those containing fewer than five links. In addition, generic categories such as centuries and years were removed and the articles belonging to them were removed, too. After applying the mentioned processes, we were left with 1,504,748 articles and 126,709 categories.

3.2. Extracting the features from Wikipedia

After preprocessing the adopted snapshot of Wikipedia, we attempt to extract its main features by analyzing the text of each article, its title, links, and the categories structure of Wikipedia. Stop-words are first removed

from the content of articles. Afterwards, the remaining words are utilized to represent the different concepts of Wikipedia by assigning a weight to each word with the term frequency-inverse document frequency (TFIDF) measure [25].

3.2.1. Boosted term-concepts table

Essentially, we attempt to link all the words within Wikipedia to all concepts by forming a vector for each word. The vector simply contains the l2-normalized TFIDF weight of the word within each Wikipedia article. These weights give an indication as to how much each word contributes to the concepts it is mapped to. We form the term-concepts table by ranking all the concepts attached to a term in a decreasing order based on the computed word weight.

After forming the initial version of the term-concepts table, we attempt to take advantage of the many redirect links within Wikipedia by analyzing the terms of the redirect links and the title of the article that they point to. In many cases, redirect links contain a term or more relevant to the articles that they point to, and yet the term does not appear in the article title or its inner content. To address this issue, we perform a two-level boosting process that attempts to factor in the presence of a term or more in a redirect link by updating the weight of the concept that the redirect link points to in attempt to reflect this link.

The boosting algorithm that we apply is demonstrated in Figure 2 and was previously described in [25]. In the illustrated algorithm, W is the set of words for which we need the most related concepts. c_t is the concept title while c_s is its score. C represents the set $(c_t\ ,c_s)$ for the concepts that were generated after applying the term-concept table process to W. *allC* represents all of the concepts within Wikipedia while *allR* is for all the redirect links. After applying the boosting process to the word *Unhappy*, we get the results illustrated in Figure 3.

3.2.2. Wikipedia internal links and categories

In this section, an attempt is made to utilize the internal hyperlinks within the articles' text pointing to other articles in the English Wikipedia and also the categories' structure to extract other useful semantic features. As not all the links in an article are of the same importance level, we apply a filtering module that aims to focus on mainly the links of high quality, discarding those of less significance. This is achieved by classifying the internal links within Wikipedia articles into several levels reflecting their significance. The hierarchies of the category network in Wikipedia were also utilized. In Figure 4, we illustrate the link types adopted in our framework. The links are sorted in a decreasing order based on the weights they carry. In general, there are three main categories of links: first is mutual links, in which two articles point to each other. Second is a single link between two articles, where both share a parent (or a grandparent) category. Third is the 'See Also' links, which are manually appended to many articles in Wikipedia by its contributors. The weights that were assigned to the different link types are shown Table 1. The weights given to w_9 and w_{10} are for 'See Also' links and 'Inverse See Also' links, respectively.

It should be emphasized that the chosen weights for the different link types reflect their importance levels. To test the effectiveness of the assigned weights, we utilize the extracted features first in the application of WSD. In the evaluation performed, we show how the weight of each link type affects the overall performance of the system.

4. Word sense disambiguation

The main goal of WSD is to identify the correct sense for a specific term in the context it exists in. The system we employ utilizes the explicit features present in the context in addition to the features previously extracted

```
Boost_Concepts( W, C, FirstLevelBoost, SecondLevelBoost, allC, allR){
      HS  ← Max(c_s ∈ C) /* Highest Score in the C set */
      FL  ← FindFirstLevelConcepts(W, allC, allR)
      FLB ← ϕ
      ∀ (c_t , c_s) ∈ FL, FLB ← FLB ∪ (c_t, (HS * FirstLevelBoost))
      SL  ← FindSecondLevelConcepts(W,C,SecondLevelBoost)
      Res ← C
      Res ← Res ∪ FLB ∪ SL /* Overwriting repetitions with those in SL and FLB */
      return Res
}

FindFirstLevelConcepts(W, allC, allR){
     Res ← ϕ
     Loop over (c_t, c_s) ∈ (allC ∪ allR) {
       /* all elements of W are contained within c_t */
        If ((W ≡ c_t) and (len(W) = len(c_t)) Then Res ← Res ∪ (c_t, c_s)
     }
     return Res
}

FindSecondLevelConcepts(W,C) {
     Res ← ϕ
     Loop over (c_t, c_s) ∈ C {
      If ( (∀ w ∈ W) ∃ c' where ( c' ≡ w ) and (c' ∈ c_t) and (len(W) <> len(c_t) )) Then
        c_s = c_s * (((SecondLevelBoost-1)^(|w|/|c_t|))+1)
        Res ← Res ∪ (c_t, c_s)
      EndIf
     }
     return Res
}
```

Figure 2. The pseudocode for the boosting algorithm applied to the term-concepts table.

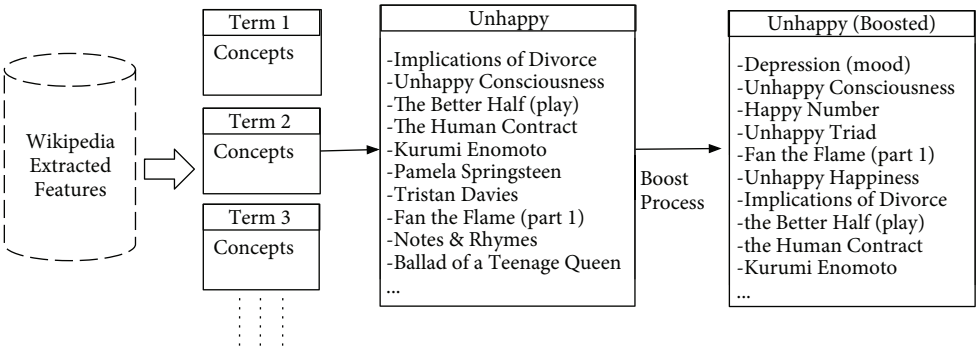

Figure 3. Using redirect links to boost the term-concepts table.

from Wikipedia. It works by mainly applying the term-concepts table to the target word and then giving a score to each concept according to the strong links analysis. Thus, we obtain a ranked list of concepts with the top concept as the chosen sense for the target word. The following subsections give a brief description on the main modules of WSD, which are illustrated in Figure 5.

4.1. Preprocessing and selecting the context

The text document containing the target term is first parsed and tokenized, and its stop words are removed. Then the target word is marked in the source document and a specific number $2n$ of its surrounding terms are extracted, with n words before the target term and another n after. The $2n$ extracted terms are called context

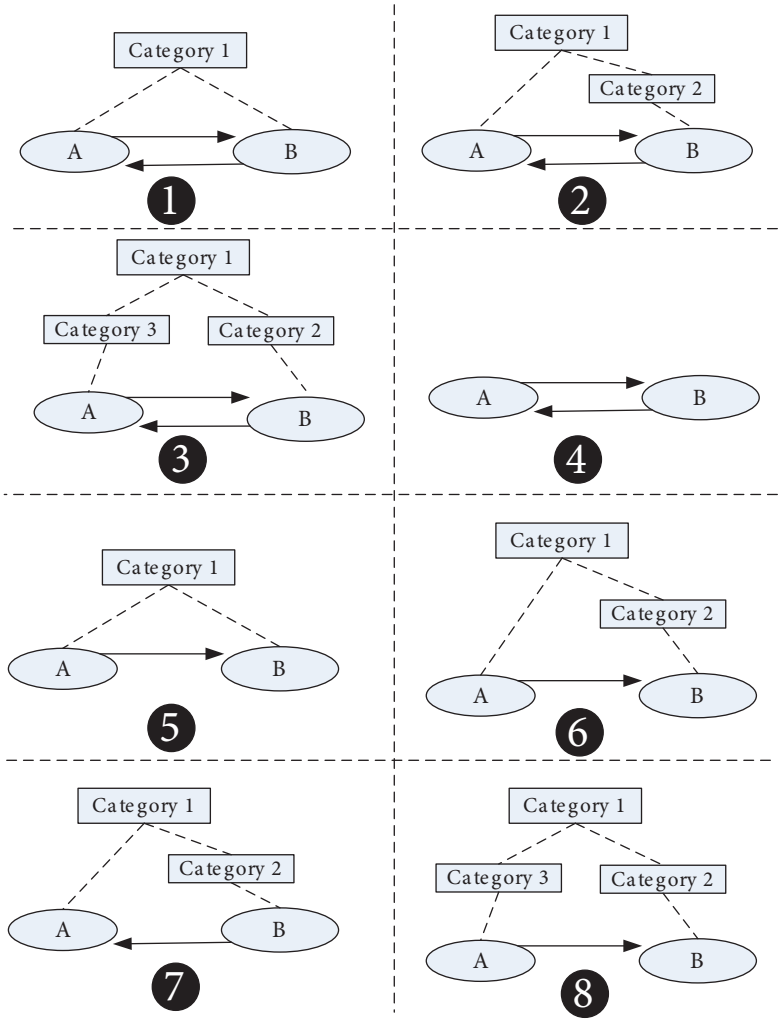

Figure 4. The defined link types ranked in a descending order based on the weights they carry.

Table 1. The assigned weights for the different links types.

Link type	Weight assigned	Link type	Weight assigned
1	3	6	1.5
2	2.75	7	1.5
3	2.5	8	1.25
4	2.25	9	3.75
5	1.75	10	3.25

terms and are abbreviated as CT. The elements of CT that are the individual context terms can thus be labeled as $ct_{i=1...|CT|}$.

4.2. Term-concepts expansion

After determining CT in the previous module, the term-concepts table is applied onto CT and the target term to generate a concept list C_i for each term, resulting in a total of |CT| + 1 concepts lists. Each concept list

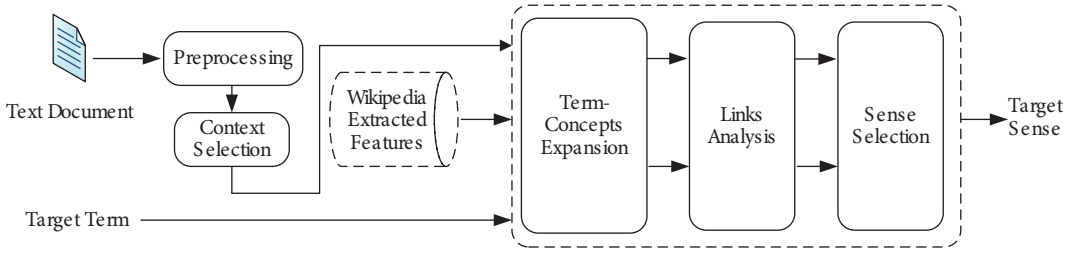

Figure 5. The main stages of the WSD process.

can be represented with the following formula:

$$C_i = \{c_{ij}\}_{j=\{1...V\},i=\{1....|CD|\}} \tag{1}$$

where i is the C_i concept list number, j is for identifying individual concepts within the concepts list, and V represents the total number of c_{ij} concepts in the list. As for the target word concepts list TW, we represent it with:

$$TW = \{g_k\}_{k=\{1...M\}} \tag{2}$$

Each concept in the list TW is labeled with g_k, where k is the concept number and M is the total number of concepts in TW.

4.3. Links analysis and sense selection

In this section, we demonstrate how the devised links from Wikipedia can be used to compute the relatedness between any two concepts and how this was utilized in the application of WSD. We describe different methodologies utilizing different aspects of the extracted features and report their performance in WSD. With the proposed methodologies, it is possible to quantify the relatedness between any two articles A and B. For this, we assign scores to the links present in A and B. We also create two sets of articles, S_A and S_B, in which the first set contains the most relevant articles to article A and the second set is for the most relevant articles to B. We find the similarity between S_A and S_B by using the cosine distance measure.

4.3.1. Terms vectors intersection

This is a basic method in which we devise a vector T that is formed by combining both CT and the target word. This new vector can be represented with:

$$T = \{w_y\}_{y=\{1...|CD|+1\}} \tag{3}$$

Assume that the possible senses for the target word are represented with S_{ye}, in which y is a unique identifier for the target word in list T and e is the possible sense number. The assumption being made in WSD is that every possible sense would have a corresponding article in Wikipedia. Therefore, the term-concepts table can be used to deduce the top R representative words with the highest weights for each sense. The list of the representative words for each sense can be represented with:

$$S_{ye} = \{a_{yef}\}_{f=\{1...D\}} \tag{4}$$

where a is a representative word in list S_{ye} and f is a unique number for each word in the list. For each possible sense (or concept) e, we compare its representative words in S_{ye} with T. The chosen sense is the one having the largest number of overlapping terms in S_{ye} and T.

The following example is meant to demonstrate the above methodology. Suppose that there is a simple document with the following T list:

T = {group, Foxtel, *Fox*, CBS, international, team}

Suppose that *Fox* is the word that needs to be disambiguated and that it has two possible senses: fox (the animal) and Fox Broadcasting Company. Each of these senses has a corresponding article (or concept) in Wikipedia which are represented with the following representative words, respectively:

S_{31} = {Mammals, tail, dog}

S_{32} = {Foxtel, entertainment, CBS}

Computing the intersection of S_{31} and S_{32} with T gives the following results:

$$S_{31} \cap T = 0$$

$$S_{32} \cap T = 2$$

From the above results, it can be deduced that the sense (Fox Broadcasting Company) is the most representative one for the target word in the given document.

4.3.2. Unweighted strong links

With this method, CT is also examined first and T is created. When compiling a list of the most related articles for each possible sense of the ambiguous word, we use the strong links analysis previously performed, but without considering the weights of the links. The list of the most relevant concepts for each possible sense with this method can be represented with:

$$P_{ye} = \{a_{yef}\}_{f=\{1...Q\}} \tag{5}$$

where Q is the number of representative concepts according to the strong links analysis method. Similar to the previous method, the most related articles for each sense are compared with T. The chosen sense is the one having the largest number of overlaps with T. The following example demonstrates this method. Suppose there is a document having T as follows:

T = {group, Foxtel, *Fox*, CBS, international, team}

The ambiguous word here is Fox and it has two possible senses, as in the previous example. Each sense has the following most representative concepts:

P_{31} = {Animal, species, tail, dog, popular culture}

P_{32} = {Foxtel, United States, News Corporation, CBS}

After computing the intersection of P_{31} and P_{32} with T, we get:

$$S_{31} \cap T = 0$$

$$S_{32} \cap T = 2$$

We can again deduce here that the second meaning is the most representative one for the ambiguous word.

4.3.3. Weighted strong links

For each concept C_i, an analysis is performed on all the links present in its corresponding article and compared against all TW concepts. The comparison between any two articles is performed by examining the concepts

list of each article against the concepts list of the other. This can be translated into the following:

$$eTW = \{G_k\}_{k=\{1...M\}} \tag{6}$$

where eTW refers to the expanded list created for TW. This list contains a list of related articles that we call G_k for each possible meaning g_k. The G_k list is formed as:

$$G_k = \{(gc_w, f(g_k, gc_w))\}_{w=\{1...V\}, f(g_k, gc_w)>0} \tag{7}$$

where gc_w is an article related to g_k, and $f(g_k, gc_w)$ is a function that measures the relatedness between g_k and gc_w through the strong links method and returns the weight. V is the number of available concepts after expansion for G_k. The same process is applied to C_i concepts, which are aggregated afterwards into a single set with the scores of repeated concepts being summed. Thus, the following is obtained after expanding C_i:

$$eC_i = \{(rc_{ijw}, f(c_{ij}, rc_{ijw}))\}_{j=\{1...V\}=i=\{1...|CT|\}, w=\{1...Q\}, f(c_{ij}, rc_{ijw})>0} \tag{8}$$

where rc_{ijw} is the concept that is related to c_{ij}, and Q is the number of related concepts. The score computed by the function $f(c_{ii}, rc_{ijw})$ can be rewritten as tw_{ijw}. We group all of eC_i into a single list that we call eC, which also factors in repeated concepts by aggregating their weights. This results in:

$$eC = \{(rc_v, tw_v)\}_{v=\{1...|D|\}} \tag{9}$$

$$tw_v = \sum_{w=1}^{Q} (twoijw), \text{ where } rw_v = rc_w \tag{10}$$

where each rc_v is a unique concept in eC_i, D is the number of unique concepts in the list eCj, and tw_v is the weight given (after the aggregation) for rc_v.

After using the context terms to derive the eC list and obtaining a list G_k for each sense of the target word, the distance between eC and each G_k is computed using the cosine distance measure. We find the most representative sense in the list by choosing the concepts numbered k in:

$$\max_k(dist(G_k, eC)). \tag{11}$$

We now apply this method to the same example used in the previously described methods.

Doc = {Foxtel, *Fox*, CBS}
Foxtel $\rightarrow$ C_1 = {Foxtel$_{11}$, Optus Television$_{12}$} = { c_{11}, c_{12}}
CBS $\rightarrow$ C_2 = {CBS$_{21}$, The Early Show$_{22}$} = {c_{21}, c_{22}}
Fox $\rightarrow$ TW = {Fox (animal)$_1$, Fox Broadcasting Company$_2$} = {g_1, g_2}
By applying this method to the above example, we expand C_1 and C_2 into the following:
eC_1 = {(Optus Television , 3) , (HDTV , 2.5)}
eC_2 = {(20th Century Fox , 6.75) , (NBC , 3)}
These are then grouped into:
eC = {(Optus Television, 3) , (HDTV, 2.5) , (20th Century Fox , 6.75) , (NBC, 3)}
TW, on the other hand, is converted into two weighted lists:

G_1 = {(silver fox, 1.75) , (Pierson v. Post,1.5)}

G_2 = {(20th Century Fox , 3), {NBC, 3)}

Thus, by computing the relatedness between eC and G_1 with the cosine measure, we get 0.0 while obtaining 0.4 for eC and G_2. Therefore, we conclude that G_2 is the best representative sense.

4.4. WSD evaluation

Choosing a dataset for evaluating a WSD system is greatly dependent upon the variations of the developed system. For example, the SemEval and Senseval-1/2/3 test collections are based on WordNet. This makes them difficult to use, as senses defined in WordNet need to be matched with concepts of Wikipedia. This has proven to be a challenging task [26] and requires its own evaluation. Therefore, we adopted our own benchmark, which is most similar to the what was adopted in [2,27,28]. We utilized the links created manually by the contributors within the articles of Wikipedia to construct our dataset. Links in Wikipedia take the form of [[PartA | PartB]], where PartA is the title of the linked page and PartB is the text displayed to the reader of the article. We constructed a dataset of 1000 Wikipedia links along with the paragraphs containing them. In our evaluations, 20 words were chosen as the context terms. This choice was based on the experiment performed on the third methodology we adopted. The findings of this experiment are reported in Figure 6.

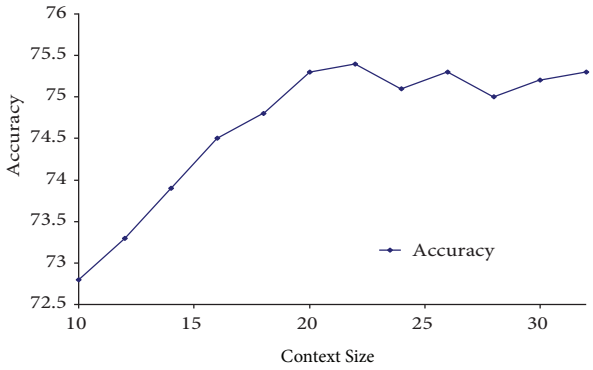

Figure 6. Effect of changing context size on the accuracy in WSD using the simple links analysis method.

We also performed another experiment that aimed to choose the best optimal weights for the different strong links. In the experiment, the weight of each link type was changed while keeping the weights for the rest of the links fixed. The results obtained and their effects on WSD are reported in Figure 7 for weights w_1 to w_5 and in Figure 8 for w_6 to w_{10}. From the reported results, it can be seen that the 'See Also' and 'Inverse See Also' link types have the strongest effect on the system. The overall accuracy of the system decreases to 65.91% and 66.41% respectively when setting their weights to zero. The weakest link type is the single link from an article to another article in which both articles share a grandparent category, as the accuracy of the system decreased to 75.31% when w_8 was set to zero. Increasing w_8 to any value above 1.25 affects the system performance negatively, too. It was also found that the effects of link types 6 and 7 are very similar, as their chart curves in Figure 8 almost overlap. When the link weights for the mutual links of types 1–4 are set above 2.5, the overall performance of the system is best. On the other hand, the performance of the system degrades when setting the weights above 1.75 for all single link types.

Several experiments were performed on the constructed dataset to test the described methodologies of this paper. In particular, we tested term vectors intersection, unweighted strong links, and weighted strong links.

The weighted strong links method showed the best performance by giving an accuracy of 75.41%. The term vectors intersection method gave an accuracy of 69.17%. The results obtained for all methods are illustrated in Table 2. We also computed the chances of having the correct sense being one of the top 2 and 3 senses of those in the ranked senses list produced with our methods. The accuracy of our best performing method increases to 91.82% when the top 3 senses are considered.

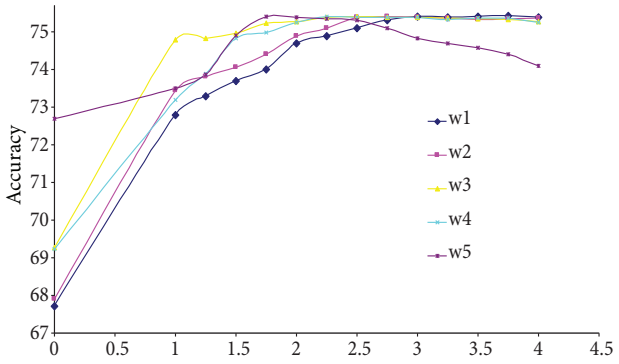

Figure 7. Effects of strong links weight change for w_1 to w_5.

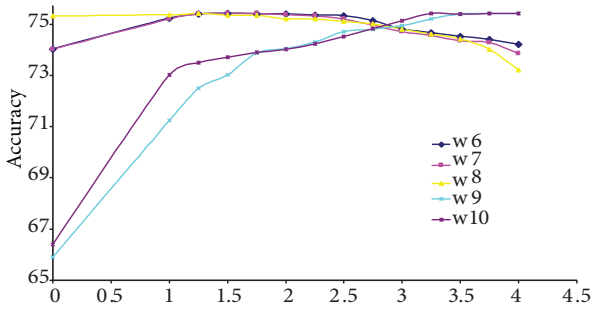

Figure 8. Effects of strong links weight change for w_6 to w_{10}.

Table 2. The accuracy of all implemented methods, including term vectors intersection, unweighted strong links, and weighted strong links methods.

	Top	Top 2	Top 3
Term vectors intersection	69.17	75.8	82.71
Unweighted strong links	71.84	84.08	87.29
Weighted strong links	75.41	87.19	91.82

4.5. Matching named entities

In addition to the above evaluation, we also adapted our system and applied it to the application of named entities matching. We used the data provided for the first task of the knowledge base population (KBP) track at TAC 2010. The task is to determine for each query which KB entity it refers to, or NIL if no match is found in the provided KB. A query consists of a name-string and a source document containing it. The aim of the source document associated with each query is to provide a context that may be useful in identifying a match for the name-string if one exists.

The test collection comprises source documents drawn from newswire and blog collections. The KB is derived from Wikipedia and contains 818,741 entities in total. Among these entities are 114,523 person (PER) entities, 55,813 organization (ORG) entities, 116,498 geopolitical entities (GPE), and 531,907 unknown (UKN) entities.

4.6. Methods applied

We handle this task by first forming a ranked list of candidate entities using the features that we extracted from Wikipedia with our methods. Afterwards, we match our top ranked entity with its counterpart in the reference KB if it exists, or NIL if it does not. The methods that we applied are as follows.

4.6.1. Term-concepts table

The method applied for this run is similar to the one applied for the application of WSD. We consider the combined tokens of the name-string of the query as the target terms while the provided context terms in the reference document are used according to the method described above in Section 4.3.1. To detect NIL matches, we simply compare the top 10 concepts corresponding to the query entity against the entities provided in the reference KB. If no matching entity exists, we assume that NIL is the answer. We call this run *TC*.

4.6.2. Links analysis

For the string of each query, we attempt to find its exact match in a dictionary created mainly from the internal Wikipedia links extracted for preparing the weighted strong links. We used the structure of the internal links within Wikipedia to create a dictionary that comprises mainly two elements: the surface string and a list of concepts pointed to by the surface string. The concepts list is sorted according to the number of occurrences for each surface string in a decreasing order. In essence, the dictionary would give a reference to the popularity of each concept within Wikipedia given the surface string. For example, the surface string having the term "Battery" would be best represented with the concept "Battery (electricity)" as it is used most often in Wikipedia to refer to that concept. The second most popular concept for the same term is "Artillery Battery", then followed by the concept "Battery (Baseball)" and so on. We call this ranked list of concepts attached to each surface string *SC*. We created a run for this method during the evaluation and called it *SC* run.

In a separate run that we call *SC-WL*, we extend the *SC* run by comparing the candidate concepts for each surface string against the detected concepts within the reference documents containing the query. For the comparison, we use the weighted strong links method. The best representative concept for the query is the one found most related to the context concepts.

4.7. Evaluation results and discussions

We evaluated our runs using the scorer and dataset provided for the TAC-KBP 2010 entity linking task (http://nlp.cs.qc.cuny.edu/kbp/2010/). There were 2250 queries in total, with each query belonging to one of four types: PER, ORG, GPE, or NIL. Accuracy was used to measure the performance of the different runs for each system. Formally, it is defined as the number of entities correctly matched to a reference KB entity (or NIL) divided by the total number of queries. The number of runs that we evaluated was three and their results are illustrated in Table 3. As can be shown from the results, the SC-WL run gave the best overall performance when compared against the other two runs. It can also be noted that the run SC, which does not rely on the context terms, performed relatively well and obtained a score above the median.

Table 3. Accuracy of our runs compared to best and median runs and differentiated by entity types.

	All (2250)	ORG (750)	GPE (749)	PER (751)
TC	0.686	0.602	0.683	0.792
SC	0.802	0.761	0.723	0.941
SC-WL	0.810	0.787	0.774	0.925
Best	0.868	0.852	0.796	0.960
Median	0.684	0.677	0.598	0.845

In Table 4, we show the results obtained with our runs on the In-KB entities vs. NIL entities. The runs SC and SC-WL performed better than TC for all entity types. Overall, the SC-WL run performed best across all queries. For PER entity types, the SC run performed best.

Table 4. Accuracy of our runs in the TAC-KBP 2010 entity linking task for In-KB and NIL entities.

	All (2250)		ORG (750)		GPE (749)		PER (751)	
	In-KB	NIL	In-KB	NIL	In-KB	NIL	In-KB	NIL
TC	0.546	0.801	0.488	0.843	0.608	0.743	0.479	0.835
SC	0.642	0.918	0.519	0.913	0.673	0.820	0.808	0.972
SC-WL	0.688	0.906	0.563	0.884	0.670	0.851	0.791	0.917

We also compared the performance of our system against several other state-of-the-art systems. The system described in [22], labeled in this paper as Bunescu, relies on supervised learning. It uses the SVM ranking model and exploits Wikipedia by computing the cosine similarity between its article contents and the context terms of the ambiguous word. It also utilizes the categories structure of Wikipedia by mapping each of the context terms to the categories, parent categories, and grandparent categories of each of the candidate senses for the ambiguous word. The training data provided by TAC for this task are used with this system and the results of the implementation of this system in [29] are reported in Table 5.

Table 5. Comparison of the performance of our runs against the performance of other systems.

System	All	In-KB	NIL
TC	0.686	0.546	0.801
SC	0.802	0.642	0.918
SC-WL	0.810	0.688	0.906
Bunescu	0.808	0.684	0.911
PRIS	0.733	0.601	0.842
NUS-I2R	0.794	0.635	0.925
Title	0.689	0.373	0.973
NIL	0.547	0.000	1.000

Another system that we examined uses different information retrieval approaches and SVM classification. We label this system in this paper as NUS-I2R [30]. In their work, Zhang et al. developed a combined system that relies on a three-class classifier to judge which of its subsystems is to be trusted. They used SVM as their machine learning algorithm. The three subsystems that they employed are as follows. First is one that employs Lucene for indexing the Wikipedia text of the candidate entities and essentially choosing the right sense. Second is the ranking first system, which basically uses the ranking SVM model for ranking entities. Third is the classification first system, which employs the SVM to decide whether each (query, entity) pair is positive.

We also report the results of another system [31] that we call PRIS in this paper. The system combines machine learning methods with rule-based methods for matching named entities. Different rules were applied to different entity types and the learner used various features extracted mainly from the context terms of the ambiguous words such as the part of speech, token order, and existence of other named entities in the context. The task is thus regarded as a classification problem with the conditional random fields model being adopted.

Additionally, we implemented two more methods, which we used as baselines. One is called the *Title* method and it attempts to directly find an exact match for the query string on the titles of Wikipedia. The other is called *NIL* and it simply gives NIL as the answer to every encountered query string.

The results reported in Table 5 indicate that our best run SC-WL reported a relatively better overall performance than the other learning-based systems, even though none of our runs were learning-based. It can also be noted that the NUS-I2R system performed best in identifying the NIL matches.

5. Conclusion

In this paper, we presented several novel algorithms leveraging both the content and structure of Wikipedia for quantifying how semantically related two terms, phrases, or sentences are. We also described how to apply the Wikipedia-extracted features, namely the term-concepts table and strong links, to the application of WSD. We also illustrated how they can be adapted for use with named entities matching. We used the links created manually in Wikipedia as the basis for one of our evaluation datasets. We also evaluated the adapted runs derived from our system in the application of named entities matching using the dataset provided by the TAC for the entity linking task of the KBP track. In addition, we reported the effects of changing the strong links' weights on the performance of the system and highlighted the reasoning behind choosing the weights. The results that we obtained from evaluating the Wikipedia-extracted features in the applications of WSD and named entities matching suggest that the strong links method can give better performance than the term-concepts table. In comparison with other systems, the system described in this paper, which is unsupervised, obtained generally better results than the median scores of the other systems from the TAC KBP. We compared the best run of our system against other learning-based state-of-the-art systems in the application of named entities matching and reported the results, which were found to be in favor of our system.

Acknowledgment

This work was funded by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, under Grant No. (968-004-d1433).The authors, therefore, acknowledge with thanks the DSR's technical and financial support.

References

[1] Lesk M. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In: Proceedings of the 5th Annual International Conference on Systems Documentation; 1986; Toronto, ON, Canada: ACM. pp. 24–26.

[2] Turdakov D, Velikhov P. Semantic relatedness metric for Wikipedia concepts based on link analysis and its application to word sense disambiguation. In: Colloquium on Databases and Information Systems (SYRCoDIS); 2008; CEUR-WS.

[3] Patwardhan S, Banerjee S, Pedersen T. UMND1: Unsupervised word sense disambiguation using contextual semantic relatedness. In: Proceedings of the 4th International Workshop on Semantic Evaluations; 2007; Prague, Czech Republic: Association for Computational Linguistics. pp. 390–393.

[4] Lehmann J, Monahan S, Nezda L, Jung A, Shi Y. LCC approaches to knowledge base population at TAC 2010. In: Proceedings of the Third Text Analysis Conference (TAC 2010); 2010; Gaithersburg, MD, USA.

[5] Guo Y, Tang G, Che W, Liu T, Li S. HIT Approaches to entity linking at TAC 2011. In: Proceedings of the Fourth Text Analysis Conference (TAC 2011); 2011.

[6] Cucerzan S. TAC entity linking by performing full-document entity extraction. In: Proceedings of the Fourth Text Analysis Conference (TAC 2011); 2011.

[7] Geng X, Liu TY, Qin T, Arnold A, Li H, Shum HY. Query dependent ranking using k-nearest neighbor. In: Proceedings of the 31st Annual International Conference on Research and Development in Information Retrieval; 2008; ACM SIGIR. pp. 115–122.

[8] Zhu ZA, Chen W, Wan T, Zhu C, Wang G, Chen Z. To divide and conquer search ranking by learning query difficulty. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management; 2009; New York, NY, USA: ACM. pp. 1883–1886.

[9] Bagga A, Baldwin B. Entity-based cross-document coreferencing using the vector space model. In: Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics; 1998; Stroudsburg, PA, USA: Association for Computational Linguistics. pp. 79–85.

[10] Ravin Y, Kazi Z. Is Hillary Rodham Clinton the President? Disambiguating names across documents. In: Proceedings of the ACL '99 Workshop on Coreference and Its Applications; 1999; College Park, MD, USA. pp. 9–16.

[11] Wacholder N, Ravin Y, Choi M. Disambiguation of proper names in text. In: Proceedings of the Fifth Conference on Applied Natural Language Processing; 1997; Stroudsburg, PA, USA: Association for Computational Linguistics. pp. 202–208.

[12] Pedersen T, Purandare A, Kulkarni A. Name discrimination by clustering similar contexts. In: Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing; 2005; Berlin, Germany: Springer-Verlag. pp. 226–237.

[13] Mann GS, Yarowsky D. Unsupervised personal name disambiguation. In: Proceedings of the 7th Conference on Natural Language Learning at HLT-NAACL 2003; 2003; Stroudsburg, PA, USA: Association for Computational Linguistics. pp. 33–40.

[14] Bekkerman R, McCallum A. Disambiguating Web appearances of people in a social network. In: Proceedings of the 14th International Conference on World Wide Web; 2005; New York, NY, USA: ACM. pp. 463–470.

[15] Malin B, Airoldi E, Carley KM. A network analysis model for disambiguation of names in lists. Comput Math Organ Th 2005; 11: 119–139.

[16] Hassan H, Hassan A, Noeman S. Graph based semi-supervised approach for information extraction. In: Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing; 2006; Stroudsburg, PA, USA: Association for Computational Linguistics. pp. 9–16.

[17] Kalashnikov DV, Nuray-Turan R, Mehrotra S. Towards breaking the quality curse. A web-querying approach to web people search. In: The 31st Annual International Conference on Research and Development in Information Retrieval; 2008; ACM SIGIR. pp. 27–34.

[18] Miller GA. WordNet: A lexical database for English. Commun ACM 1995; 38: 39–41.

[19] Alfonseca E, Manandhar S. An unsupervised method for general named entity recognition and automated concept discovery. In: Proceedings of the 1st International Conference on General WordNet; 2002.

[20] Liu J, Birnbaum L. Measuring semantic similarity between named entities by searching the web directory. In: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence; 2007; Washington, DC, USA: IEEE Computer Society. pp. 461–465.

[21] Wan S, Angryk RA. Measuring semantic similarity using WordNet-based context vectors. In: 2007 IEEE International Conference on Systems, Man and Cybernetics; 2007; Montreal, QC, Canada. pp. 908–913.

[22] Bunescu R, Pasca M. Using encyclopedic knowledge for named entity disambiguation. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-06); 2006; Trento, Italy. pp. 9–16.

[23] Cucerzan S. Large-scale named entity disambiguation based on Wikipedia data. In: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL); 2007; Prague, Czech Republic: Association for Computational Linguistics. pp. 708–716.

[24] Bawakid A, Oussalah M. Using features extracted from Wikipedia for the task of word sense disambiguation. In: 9th IEEE International Conference on Cybernetic Intelligent Systems 2010; 2010; Reading, UK: IEEE.

[25] Manning CD, Schuetze H. Foundations of Statistical Natural Language Processing. 1st ed. Cambridge, MA, USA: MIT Press; 1999.

[26] Mihalcea R. Using Wikipedia for automatic word sense disambiguation. In: Proceedings of the Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; 2007; Rochester, NY, USA.

[27] Mihalcea R, Csomai A. Wikify!: linking documents to encyclopedic knowledge. In: CIKM '07: Proceedings of the 16th ACM Conference on Information and Knowledge Management; 2007; Lisbon, Portugal: ACM. pp. 242–233.

[28] Fogarolli A. Word sense disambiguation based on Wikipedia link structure. In: International Conference on Semantic Computing; 2009; IEEE. pp. 77–82.

[29] Hachey B, Radford W, Nothman J, Honnibal M, Curran JR. Evaluating entity linking with Wikipedia. Artif Intell 2013; 194: 130–150.

[30] Zhang W, Sim YC, Su J, Tan CL. NUS-I2R: Learning a combined system for entity linking. In: Proceedings of the Third Text Analysis Conference (TAC 2010); 2010; Gaithersburg, MD, USA.

[31] Gao S, Cai Y, Li S, Zhang Z, Guan J, Li Y, Zhang H, Xu W, Guo J. PRIS at TAC2010 KBP track. In: Proceedings of the Third Text Analysis Conference (TAC 2010); 2010; Gaithersburg, MD, USA.