Collective Named Entity Disambiguation using Graph Ranking and Clique Partitioning Approaches

Ayman Alhelbawy[*†] and **Robert Gaizauskas** [*]
[*]The University of Sheffield, Regent Court, 211 Portobello Street, Sheffield, S1 4DP, U.K
[†]Faculty of Computers and Information, Fayoum University, Fayoum, Egypt
`ayman,R.Gaizauskas@dcs.shef.ac.uk`

Abstract

Disambiguating named entities (NE) in running text to their correct interpretations in a specific knowledge base (KB) is an important problem in NLP. This paper presents two collective disambiguation approaches using a graph representation where possible KB candidates for NE textual mentions are represented as nodes and the coherence relations between different NE candidates are represented by edges. Each node has a local confidence score and each edge has a weight. The first approach uses Page-Rank (PR) to rank all nodes and selects a candidate based on PR score combined with local confidence score. The second approach uses an adapted Clique Partitioning technique to find the most weighted clique and expands this clique until all NE textual mentions are disambiguated. Experiments on 27,819 NE textual mentions show the effectiveness of both approaches, outperforming both baseline and state-of-the-art approaches.

1 Introduction

Named entities (NEs) have received a lot of attention from the NLP community over the last two decades (see, e.g. Nadeau and Sekine (2007)). Most of this work has focussed on the task of recognizing the boundaries of NE mentions in text and classifying them into one of several classes, such as Person, Organization or Location. However, references to entities in the real world are often ambiguous: there is a many-to-many relation between NE mentions in text and the entities denoted by these mentions in the world. For example, the same NE mention "Norfolk" may refer to a person, "Peter Norfolk, a wheelchair tennis player", a place in the United Kingdom, "Norfolk County", or a place in the United States like "Norfolk, Massachusetts"; conversely, one entity many be known by many names, such as "Cat Stevens", "Yusuf Islam" and "Steven Georgiou". The task of named entity disambiguation (NED) is to establish a correct mapping between each NE mention in a document and the entity it denotes in the real world. Following most researchers in this area, we treat entries in a large knowledge base (KB) as surrogates for real world entities when carrying out NED and, in particular, use Wikipedia as the reference KB against which to disambiguate NE mentions. NED is important for tasks like KB population, where we want to extract new information from text about an entity and add it to a pre-existing entry for that entity in a KB, or for information retrieval where we may want to cluster or filter results for different entities with the same textual mentions.

The main hypothesis underlying this work is that different NEs in a document help to disambiguate each other. However, other textual mentions in the document are also ambiguous. So, what is needed is a *collective disambiguation* approach that jointly disambiguates all NE textual mentions.

In our approaches we model each possible candidate for every NE mention in a document as a distinct node in a graph and model candidate coherence by links between the nodes. Figure 1 shows an example of the disambiguation graph for three mentions "A", "B", and "C" found in a document, where the candidate entities for each mention are referred to using the lower case form of the mention's letter together with a distinguishing subscript. The goal of disambiguation is to find a set of nodes where only one candidate is selected from the set of entities associated with each mention, e.g. a_3, b_2, c_2.

We propose two different approaches to find the best disambiguation candidates in the graph. The first approach starts by finding the most confident and coherent set of disambiguation entities and iteratively expands this set until all NE textual mentions are disambiguated. The second approach ranks all nodes in the solution graph using the Page-Rank algorithm, then re-ranks all nodes by combining the initial confidence and graph ranking scores. We consider several different measures for computing the initial confidence assigned to each node and several measures for determining and weighting the graph edges. Node linking relies on the fact that the textual portion of KB entries typically contains mentions of other NEs. When these mentions are hyper-linked to KB entries, we can infer that there is some relation between the real world entities corresponding to the KB entries, i.e. that they should be linked in our solution graph. These links also allow us to build up statistical co-occurrence counts between entities that occur in the same context, which may be used to weight edges in our graph.

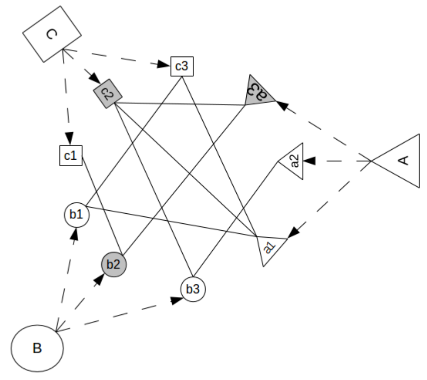

Figure 1: Example of solution graph

We evaluate our approaches on the AIDA dataset (Hoffart et al., 2011). Comparison with the baseline and some state-of-the-art approaches shows our proposed approaches offers substantial improvements in disambiguation accuracy.

The rest of the paper is organized as follows: Section 2 discusses related work. Section 3 discusses selection of NE candidate entities from the Wikipedia knowledge base and the assignment of confidence scores to each candidate. Formulation of the NE disambiguation problem in terms of a graph model is presented in section 4. Sections 5 and 6 describe the clique partitioning and ranking disambiguation approaches for collective NED. The experimental dataset and experimental results are presented in Section 7. Section 8 concludes the paper and presents some suggestions for future work to improve the results.

2 Related Work

Named Entity Disambiguation has received a lot attention in the past few years. Perhaps the best known related work is the Entity Linking (EL) shared task challenge first proposed by the National Institute of Standards and Technology (NIST) as part of the Knowledge Base Population (KBP) track within the Text Analysis Conference (TAC) in 2009 (McNamee and Dang, 2009). EL is a similar but broader task than NED: NED is concerned with disambiguating a textual NE mention where the correct entity is known to be one of the KB entries, while EL also requires systems to deal with the case where there is no entry for the NE in the reference KB. Ji et al. (2011) group and summarise the different approaches to EL taken by participating systems.

In general, there are two main lines of approach to the NED problem. The first, *single entity disambiguation approaches (SNED)*, disambiguates one entity at a time without considering the effect of other NEs. These approaches use local context textual features of the mention and compare them to the textual features of NE candidate documents in the KB, and link to the most similar. The first approach in this line was Bunescu and Pasca (2006), who measure similarity between the textual context of the NE mention and the Wikipedia categories of the candidate. More similarity features were added by Cucerzan (2007) who realized that topical coherence between a candidate entity and other entities in the context will improve NED accuracy by calculating the nodes' coherence based on the their incoming links in Wikipedia and the overlaps in Wikipedia categories. Milne and Witten (2008) improve Cucerzan's work by calculating the topical coherence using Normalized Google Distance and restrict the context entities to the unambiguous entities. Different query expansion approaches are incorporated into this framework, such as using context term expansion (Gottipati and Jiang, 2011) and acronym expansion (Zhang et al., 2011). Sen (2012) proposed a latent topic model to learn the context entity association. Machine learning is widely used in SNED as some approaches deal with the problem as a search result ranking problem. Supervised learn-to-rank models are used to re-rank the ambiguous candidate set (Zheng et al., 2010; Dredze et al., 2010; Alhelbawy and Gaizauskas, 2012; Nebhi, 2013).

The second line of approach is *collective named entity disambiguation (CNED)*, where all mentions of entities in the document are disambiguated jointly. These approaches try to model the interdependence between the different candidate entities for different NE mentions in the query document, and reformulate the problem of NED as a global optimization problem whose aim is to find the best set of entities. As this new formulation is NP-hard, many approximations have been proposed. Kulkarni et. al. (2009) presents a collective approach for entity linking that models the coherence between all pairs of entity candidates for different mentions as a probabilistic factor graph. They present two approximations to solve this optimization problem where the interdependence between decisions is modelled as the sum of the pairs' dependencies. Alhelbawy and Gaizauskas (2013) proposed a sequence dependency model using HMMs to model NE interdependency. Another approximation uses a mixture of local and global features to train the coefficients of a linear ranking SVM to rank different NE candidates (Ratinov et al., 2011). Shirakawa et al. (2011) cluster related textual mentions and assign a concept to each cluster using a probabilistic taxonomy. The concept associated with a mention is used in selecting the correct entity from the Freebase KB.

Graph models are widely used in collective disambiguation approaches. All these approaches model NE interdependencies, while different methods may be used for disambiguation. Han (2011) uses local dependency between NE mention and the candidate entity, and semantic relatedness between candidate entities to construct a referent graph, proposing a collective inference algorithm to infer the correct reference node in the graph. Hoffert (2011) poses the problem as one of finding a dense sub-graph, which is infeasible in a huge graph. So, an algorithm originally used to find strongly interconnected, size-limited groups in social media is adapted to prune the graph, and then a greedy algorithm is used to find the densest graph.

The word sense disambiguation (WSD) task has many similarities to NED, since in both cases the goal is to determine which of a set of predefined senses or reference entities is the correct interpretation of a surface string in context. Many researchers have used graph-based approaches successfully for the WSD task. Sinha and Michalecea (2007) proposed using four different graph centrality algorithms – Indegree, PageRank, Closeness and Betweenness for WSD. We propose to use a clique partitioning algorithm, originally proposed by Born et al. (1973), for NED. Clique algorithms have been successfully used for WSD problems. Gutiérrez et al. (2011; 2012), for example, use an N-cliques graph partitioning technique to identify sets of highly related senses. However, this approach has not been used for NED.

Our second proposed model uses the Page-Rank algorithm (PR), which to our knowledge has also not previously been applied to NED. PR was proposed by Page et al. (1999) to produce a global rank for web pages based on the hyperlink structure of the web. Xing and Ghorbani (2004) adapted PR to take into account the weights of links and the nodes' importance. PR and Personalized PR algorithms have been used successfully in WSD (e.g. Sinha and Mihalcea (2007), Agirre and Soroa (2008; 2009)).

3 Named Entity Candidates Selection

Given an input document D containing a set of pre-tagged NE textual mentions $M = \{m_1, m_2, m_3 \dots m_k\}$, we need to select all possible candidate interpretations for each m_i from the knowledge base. I.e. for each NE textual mention $m_i \in M$ we select a set of candidates $E_i = \{e_{i,1}, e_{i,2}, e_{i,3} \dots e_{i,j}\}$ from the KB. The NE textual mention m_i is used to search the KB entry titles using Lucene[1] to find entries with titles that fully or partially contain the NE textual mention. The following example shows the possible candidates for the textual mention "Sheffield": "Sheffield, New Zealand,", "University of Sheffield", "Sheffield United F.C.", "Sheffield, Massachusetts", "Fred Sheffield", "Sheffield, Alabama", etc. The result of this search is quite large and this increases the likelihood of the correct entry occurring somewhere in the list, i.e. it improves recall. However, the challenge now moves to the disambiguation step. In this step, we need to assign a confidence score to each candidate, as shown in the following section.

[1]https://lucene.apache.org/

3.1 Candidate Confidence Score

For each candidate $e_{i,j}$, a set of initial confidence scores $IConf(e_{i,j})$ is assigned. These scores are calculated for each NE candidate independent of other candidates or the candidates for other NE textual mentions in the document. Three scores are calculated locally using the NE textual mention context. There is also one global confidence score, entity popularity (EP), which is calculated globally independent of the document or the textual mention context by using the Freebase KB (Bollacker et al., 2008). The four confidence scores to be calculated for each NE candidate as follows:

- Cos: The cosine similarity between the NE textual mention and the KB entry title.

- JwSim: While the cosine similarity between a textual mention in the document and the candidate NE title in the KB is widely used in NED, this similarity is a misleading feature. For example, the textual mention "Essex" may refer to either of the following candidates "Essex County Cricket Club" or "Danbury, Essex", both of which are returned by the candidate generation process. The cosine similarity between "Essex" and "Danbury, Essex" is higher than that between "Essex" and "Essex County Cricket Club", which is not helpful in the NED setting. We adopted a new mention-candidate similarity function, $jwSim$, which uses Jaro-Winkler similarity as a first estimate of the initial confidence value for each candidate. This function considers all terms found in the candidate entity KB entry title, but not in the textual mention as disambiguation terms. The percentage of disambiguation terms found in the query document is used to boost in the initial $jwSim$ value, in addition to an acronym check (whether the NE textual mention could be an acronym for a specific candidate entity title). Experiments show that $jwSim$ performs much better than the standard cosine similarity.

- Ctxt: The cosine similarity between the sentence containing the NE mention in the query document and the textual description of the candidate NE in the KB (we use the first section of the Wikipedia article as the candidate entity description).

- EP: Entity popularity refers to connectivity to this entity. It has been used successfully as a discriminative feature for NED (Nebhi, 2013). Freebase provides an API interface to get an entity's popularity score, which is computed during Freebase data indexing. This score is a function of the entity's inbound and outbound link counts in Freebase and Wikipedia[2].

Initial confidence scores are calculated independently for each candidate entity for an NE mention. However, after the initial calculation, initial confidence scores for all candidates for a single NE mention are normalized to sum to 1.

4 Disambiguation Graph Model

In this section we discuss the graph model we use for NED. All candidate entities for the different NE textual mentions in the document are represented as an undirected graph $G = (V, D)$ where V is the node set of all possible candidate entities for different NE textual mentions in the input document and D is the set of edges between nodes. Because the same entity may be found multiple times as a candidate for different textual mentions and each occurrence must be evaluated independently, each node is formed as an ordered pair of textual mention m_i and candidate entity $e_{i,j}$. So, the graph nodes are formulated as a set $V = \{(m_i, e_{i,j}) \mid \forall e_{i,j} \in E_i, \forall m_i \in M\}$.

A set of entities is coherent if real world relations hold between them. We use such relations to link candidate entities for different NE textual mentions in our graph model. Edges are not drawn between different nodes for the same mention. However, they are drawn between two entities when there is a relation between them. Different approaches to determine and weight entity coherence relations are presented in the following section.

[2]https://developers.google.com/freebase/v1/search

4.1 Entity Coherence

Entity coherence refers to the real world relatedness of different entities which are candidate interpretations of different textual mentions in the document. Such relatedness is not based on documentcontext, so the relatedness of two candidate entities is always the same regardless of the query document. Coherence is represented as an edge between nodes in the graph. We used two measures for coherence:

- Entity Reference Relation (Ref): This is a boolean relation between two entities e_1 and e_2. The Ref relation holds if the Wikipedia document for either entity has a link to the other. Since the Wikipedia hyperlinks are directed, this relation is implicitly directed. However, we assume an inverse relation also exists and represented the relation as undirected.

$$\text{Ref}(e_i, e_j) = \begin{cases} 1, & \text{if } e_i \text{ or } e_j \text{ refers to the other} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

- Entity Co-occurrence ($Jprob$): An estimate of the probability of both entities appearing in the same sentence. Wikipedia documents are used to estimate this probability, as shown in (2), where $S(e)$ is the set of all sentences that contain a hyperlink reference to the entity e and S is the set of sentences containing any such entity references.

$$\text{Jprob}(e_i, e_j) = \frac{|S(e_i) \bigcap S(e_j)|}{|S|} \tag{2}$$

5 Cliques Partitioning Disambiguation

The clique model originated in social network studies when Luce and Perry (1949) defined a clique as a set of two or more people who are mutual friends. In graph theory, this pattern is known as a complete sub-graph. Assuming that NEs that appear in the same document can be split into groups of highly cohesive entities, we adopt the clique partitioning technique to find the biggest clique in terms of size and weight. Given an undirected graph $G(V, D)$ where V is the set of all nodes and D is the set of all edges, $G_s = (V_s, D_s)$ is a sub-graph of G where $V_s \subseteq V$ and $D_s \subseteq D$. G_s is called complete sub-graph or clique if and only if each node in V_s has a link in D_s to all other nodes in V_s. The clique partitioning algorithm aims to find all possible complete sub-graphs G_s in an undirected graph G. Our approach iteratively finds the 'best' clique, deletes all 'wrong' candidate entities for textual mentions that are disambiguated by the selected clique and converts the selected clique to a node in the graph to be used in the next iteration. The details are shown in algorithm 1. Figure 2 shows an exampler of the clique partitioning disambiguation algorithm given a graph of candidate entities for six NE textual mentions, 'A','B','C','D','E','F'. Candidate entities are coded with the lower case letter of the NE textual mention plus an index subscript, e.g., 'a1', 'a2', 'a3', etc. Cliques are shown with bold links in different colours.

As described in section 4, one of the properties of the disambiguation graph is that there are no links between candidates of the same NE textual mention. Because of this property, we can guarantee that there is no more than one candidate for each textual mention in any clique.

```
Data: Undirected graph $G(V, E)$ and for each node $v \in V$ an associated $IConf$ score
Result: Solution sub-graph
while not all textual mentions are disambiguated do
    1- clique-List = find cliques(G);
    2- weight each clique by summing the $IConf$ scores of all nodes in the clique;
    3- select the highest scoring clique and use its nodes as disambiguation candidates;
    4- remove all wrong candidates for any mention disambiguated in step 3;
    5- merge all nodes in the selected clique into one node with $IConf$ score of the
       new node = sum of the $IConf$ scores of the merged nodes;
end
```

Algorithm 1: Clique Disambiguation Algorithm

This approach does not use an entity coherence weighting (e.g. $Jprob$). Rather it just uses the entity

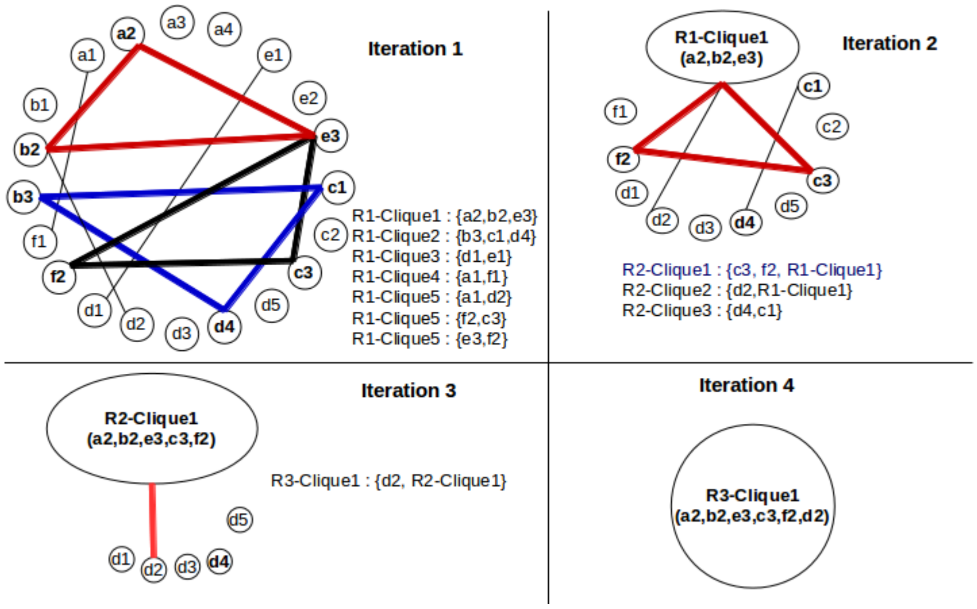

Figure 2: Example of Clique Partitioning Disambiguation

links to find the cliques regardless the relation strength. Because of the huge number of nodes, the clique finder algorithm is not fast. To speed-up the disambiguation, we filtered the nodes with low confidence in the graph, keeping only the top confidence scored 50 NE candidates for each NE textual mention.

6 Graph Ranking Disambiguation

The clique approach disambiguates different NE textual mentions iteratively, where in each iteration one or more NE mentions are disambiguated taking into account the disambiguated mentions from the previous iteration. The graph Ranking approach iteratively ranks all graph nodes depending on the links. So, all NE candidates of all NE textual mentions in the text are ranked together without ignoring any of them. Hence, a selection algorithm is used to combine the initial confidence and the graph rank score, and select the most appropriate NE candidate.

Graph Ranking: The links between different candidates in the graph represent real world relations. These relations are used to reliably boost relevant candidates. In some setups, the weight of these links are set to 1 and in some others they are set to the entities' coherence score. All nodes in the graph are ranked according to these relations using Page-Rank. We adapted a version of the PR algorithm with normalization term to rank the different NE candidates according to entity coherence as shown in equation 3, where N is the number of nodes in the graph, $coh(e_i)$ is the set of nodes that cohere with node e_i and $W(e_i, e_j)$ is the weight of the edge between e_i and e_j nodes. The original PR uses a directed graph while our graph is an undirected graph; so all links are treated as bidirectional.

$$PR(e_i) = \frac{(1-d)}{N} + \frac{d}{F(e_i)} \sum_{e_j \in coh(e_i)} PR(e_j) \times W(e_i, e_j) \tag{3}$$

$$F(e_i) = \sum_{e_j \in coh(e_i)} W(e_i, e_j) \tag{4}$$

The standard PR algorithm assumes the initial rank of all nodes is uniformly equal, while in our approach we used the initial confidence as an initial weight for the candidate nodes. A problem with Page-Rank for our purposes is the dissipation of initial node weight (confidence) over all linked nodes. The final rank of a node is based solely on the importance of linked nodes and the initial confidence plays no further role. In our case this is not appropriate, so the final rank for each mention is calculated after graph ranking, by combining the graph rank with the initial confidence score. Let us refer to the graph rank of a candidate as $PR(e_i)$. We used two different combination schemes R_s and R_m as described in equations 6 and 5.

$$R_m(e_{i,j}) = IConf(e_{i,j}) \times PR(e_{i,j}) \quad (5) \qquad\qquad R_s(e_{i,j}) = IConf(e_{i,j}) + PR(e_{i,j}) \quad (6)$$

Decision Making: Selecting the proper candidate is the final phase in the disambiguation process. The simplest approach is to select the highest ranked entity in the list for each mention m_i according to equation 7 or 8, which correpond to the rank combining schemes expressed in equations 5 and 6. Experiments show that overall using the R_m combining scheme is better than the R_s scheme. However, the highest rank, after combining graph rank score and initial confidence score, is not always correct. So we developed a dynamic selection algorithm which uses both combination schemes to pick the best disambiguation candidate. We found that a dynamic choice between the re-ranking schemes, based on the difference between the top two candidates, as described in Algorithm 2, works best. The selected candidate entity is referred to as $\hat{e}$ with the superscript showing the selection scheme.

```
Data: $E_i$ is a candidate list of one NE textual
      mention $m_i$
Result: The best disambiguation NE candidate $\hat{e}_i^g$
$R1 = \{(R_m(e_{i,j}), e_{i,j}) \mid \forall e_{i,j} \in E_i\}$;
$R2 = \{(R_s(e_{i,j}), e_{i,j}) \mid \forall e_{i,j} \in E_i\}$;
Sort $R1$ in descending order ;
Sort $R2$ in descending order ;
$R1diff$ = $R1$[0]-$R1$[1];
$R2diff$ = $R2$[0]-$R2$[1];
if R1diff > R2diff then
    return highest rank scored entity of $R1$, ($R1$[0])
else
    return highest rank scored entity of $R2$, ($R2$[0])
end
```

Algorithm 2: Selection Algorithm

$$\hat{e}_i^m = \underset{e_{i,j}}{\operatorname{argmax}}\, R_m(e_{i,j}) \quad (7) \qquad\qquad \hat{e}_i^s = \underset{e_{i,j}}{\operatorname{argmax}}\, R_s(e_{i,j}) \quad (8)$$

7 Experiments and Results

7.1 Dataset

NIST has released a dataset for use in the TAC KBP entity linking task (EL). But, the task of named entity disambiguation is different from entity linking task, as noted above in Section 2. Also, the NIST dataset is not suitable for evaluating the collective NE disambiguation task because only one NE mention is annotated and disambiguated per query document while we need all mentions of NEs in the document to be annotated and disambiguated to evaluate the performance of the collective named entity disambiguation technique. Another dataset manually annotated for NED is reported in (Kulkarni et al., 2009), but it uses an old version of Wikipedia and it is quite small. We have used another dataset, the AIDA dataset, which is based on the CoNLL 2003 data for NER tagging and in which most tagged NE mentions have been manually disambiguated against Wikipedia (Hoffart et al., 2011). This dataset contains 1393 documents, and 34,965 annotated mentions, where 7136 mention are not linked to Wikipedia[3].

We compare our results to Hoffart's work – Accurate Online Disambiguation of Named Entities (AIDA). For fair comparison, we only considered NE mentions with an entry in the Wikipedia KB, ignoring the 20% of query mentions without a link to the KB, as Hoffart did.

7.2 Evaluation Metric

We use accuracy as the evaluation metric. Micro-averaged accuracy is used as the official metric for the disambiguation task and has been used in much previous and related work. Micro-averaged accuracy

[3]AIDA dataset is available on the web to download http://www.mpi-inf.mpg.de/yago-naga/aida/

corresponds to the percentage of the correctly disambiguated textual mentions and it is calculated as shown in equation 9.

$$A_{micro} = \frac{\text{\#correctly disambiguated mentions}}{\text{Number of NE Mentions}} \tag{9}$$

Macro-averaged accuracy is used to calculate the average percentage of correctly identified named entities. Macro-averaged accuracy is calculated as shown in equation 10.

$$A_{macro} = \frac{\sum_{i}^{num} \frac{\text{Num Correct}(E_i)}{\text{Num Queries}(E_i)}}{\text{\# of unique entities}} \tag{10}$$

7.3 Results

In addition to the state-of-the-art, we used two strong baselines to evaluate the performance of the proposed approaches. The first baseline is a setup where the $IConf$ scores only are used to disambiguate the NE textual mention. In this setup a ranking based on Entity Popularity (EP) does best, with micro- and macro-averaged accuracy scores of 80.55% and 78.09% respectively. This high baseline is close to the state-of-the-art. A summary of the first baseline is shown in Table 1. The second baseline is the basic PR algorithm, where both $IConf$ scores and link weights are ignored. Links between nodes are created wherever any non-zero entity coherence relation, REF or JProb, is found. Micro- and macro-averaged accuracy scores of 70.60% and 60.91% respectively were obtained with this baseline.

	Baseline1		Cliques		PR_I		$\hat{e}^g$	
$IConf$	A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}
cos	38.44	45.68	71.59	64.83	70.6	60.83	78.41	72.35
jwSim	61.01	58.81	72.26	69.53	70.61	60.94	83.16	78.28
ctxt	24.58	21.44	58.06	57.37	70.61	60.83	75.45	65.22
EP	80.55	78.09	86.10	81.79	71.78	81.07	87.59	84.19

Table 1: Results using different $IConf$ scores with different approaches

The clique partitioning disambiguation algorithm experiments are setup so a link between nodes is created whenever a non-zero coherence relation is found between nodes regardless its weight. We used different settings for the candidates filter. In the case where no candidates filter is applied, all nodes are considered to find the best initial clique. So, bigger cliques with nodes that have lower confidence may be selected in the first iteration. This approach is very sensitive to the results of the first iteration. Consequently, the accuracy goes down. Also, because of the huge graph size, the clique partitioning algorithm takes a long time. At the other extreme, if we use only a small number of candidates with the highest confidence scores, then the accuracy also goes down because in most cases the correct disambiguation entity is filtered out of the graph. We used the highest 50 candidates in the graph and all other nodes are deleted. Table 1 shows the results of using different initial confidence scores in clique partitioning disambiguation.

Graph ranking disambiguation experiments were setup in three different settings in order to evaluate the contribution of different features like initial confidence and link weights. For all setups, we used different decision making approaches $\hat{e}^m$, $\hat{e}^s$ and $\hat{e}^g$. The results when using $\hat{e}^g$ are better than $\hat{e}^m$ and $\hat{e}^s$ for all setups. So, we report the results of $\hat{e}^g$ only. Different setups are as follows:

- PR_I: In this setup, the $IConf$ scores are used to be the initial rank for Page-Rank while the links between nodes are uniformly weighted to one. As in the PR baseline, links are created wherever Ref or $Jprob$ are not zero. Table 1 shows the results both without $IConf$ combination, i.e. using only the PR score for ranking, and after combining the initial confidence score using dynamic decision making (indicated by $\hat{e}^g$) When comparing these results to the PR baseline, we notice a slight positive effect of using the initial confidence as an initial rank instead of uniform ranking. The major improvement comes by combining the initial confidence with the PR score. All combining methods improve the results over the baseline results when using the the same confidence score while the dynamic selection algorithm overcomes other basic methods, i.e. $\hat{e}^m$ and $\hat{e}^s$.

- PR_C: In the second setup, entity coherence features are tested by setting the edge weights to the coherence score and the initial node rank is set to be uniform when running the PR algorithm. So, initial confidence scores are not considered in graph ranking but just considered in disambiguation decision making. This setting is intended to evaluate the contribution of different coherence relations. We compared $Jprob$ and Ref edge weighting approaches, where for each approach edges were created only where the coherence score according to the approach was non-zero. We also investigated a variant, called $Jprob + Ref$, in which the Ref edge weights are normalized to sum to 1 over the whole graph and then added to the JProb edge weights (here edges result wherever $Jprob$ and Ref scores are non-zero). Results in Table 2 show the $JProb$ feature seems to be more discriminative than the Ref feature but the combined $Jprob + Ref$ feature performs better than each separately, just outperforming the baseline. We used the best $IConf$ score, i.e. EP, for re-ranking. Again, combining the $IConf$ with the PR score improves the results.

- PR_{IC}: This setup uses different combinations of $IConf$ and entity coherence scores in PR. Table 3 shows the accuracy when using different combinations of all entity coherence scores and some selected (i.e. the best) $IConf$ scores. Here the $Jprob + Ref$ combination does not add any value over $Jprob$ alone. Interestingly using $IConf$ score with differentially weighted edges does not show any benefit over using $IConf$ score and uniformly weighted edges (Table 1).

	PR		$\hat{e}^g$	
Edge Weight	A_{micro}	A_{macro}	A_{micro}	A_{macro}
$Jprob$	66.52	55.83	83.31	80.38
Ref	67.48	59.76	81.80	78.53
$Jprob + ref$	72.69	65.71	83.46	80.69

Table 2: Results using weighted edges (PR_C)

		$\hat{e}^g$	
$IConf$	Edge Weight	A_{micro}	A_{macro}
jwSim	$Jprob$	82.56	76.16
jwSim	Ref	78.61	71.12
jwSim	$Jprob + Ref$	81.97	75.63
EP	$Jprob$	86.29	82.77
EP	Ref	83.16	80.01
EP	$Jprob + Ref$	86.10	82.80

Table 3: Results using initial confidence and weighted edges (PR_{IC})

To compare our results with the state-of-the-art, we report Hoffart et al.'s (2011) results as they re-implemented two other systems and ran them over the AIDA dataset which we used to evaluate our approach. We also compare with Alhelbawy and Gaizauskas (2013) and Shirakawa et al. (2011) who carried out their experiments using the same dataset. Table 4 shows a comparison between the results of our proposed approaches and the state-of-the-art. Both proposed approaches exceed the results of the state-of-the-art. However our approaches are very simple and direct to apply, unlike Hoffart et al.'s and Shirakawa et al.'s which are considerably more complex. Also, our approaches do not need any kind of training, unlike the Alhelbawy approach.

7.4 Discussion

The Page-Rank algorithm was originally designed for directed graphs while our coherence features are undirected. So, the node rank depends on both incoming and outgoing links (when converting the undirected graph to a directed graph). That explains the little improvement over basic PR when using the

	B1	B2	Cliques	PR_C	PR_I	PR_{CI}	Cucerzan	Kulkarni	Hoffart	Shirakawa	Alhelbawy
A_{macro}	78.09	60.91	81.79	80.98	84.19	82.80	43.74	76.74	81.91	83.02	74.18
A_{micro}	80.55	70.60	86.11	83.59	87.59	86.10	51.03	72.87	81.82	82.29	78.49

Table 4: Summary of Presented Approaches and State-of-the-art Results. B1 and B2 are baselines.

initial confidence as an initial rank before using PR (see Table 1). However, when comparing PR results in Tables 2 and 1, we can see that the PR algorithm is more sensitive to the links than to initial ranks. The combined coherence approach ($Jprob + Ref$) actually has a value other than the different weighting it supplies; the approach results in more edges than either of the combined approaches do alone. In all PR results wherever edge weights are applied, the result of using the combined coherence measures outperforms either of them singly.

Informal failure analysis was carried out to determine reasons for disambiguation failure. Reasons identified include:

1. The correct NE candidate does not exist in the graph. In such cases the disambiguation approach selected is irrelevant and what is needed is improved candidate selection.
2. Lack of edges. When there are no edges between any of the query NE mention candidate entities and other mentions' candidates. In this case the decision depends only on the $IConf$ score.
3. Where the Freebase popularity score (EP) is used, whenever this score for the correct NE candidate is 0, which means the selection process is based on the PR score.

Table 5 shows an example of the highest three NE candidates for three NE mentions taken from a document (overall the document contains textual mentions for ten different NEs). The first one is "Ford" and is disambiguated correctly to "Ford Motor Company", where the PR and popularity scores are higher than any of the other candidates. The second one is ,"Magna", disambiguated correctly, where the first two NE candidates have the same PR score but the popularity score discriminates between them. The third, "Markham", is disambiguated to "Clements Markham" while it should be disambiguated to "Markham, Ontario". The problem in this case is that all NE candidates for the mention "Markham" are not linked to any entity candidates for any other NE mentions in the document (problem 2 above). Therefore, the popularity score dominates the final rank score.

NE Candidate	PR score $\times 10^{-3}$	FB Rank $\times 10^{-3}$	our Rank $\times 10^{-3}$
Ford			
Ford Motor Company	21.37	62.12	1.32
Ford Galaxie	4.59	10.94	0.05
Ford GT	2.83	11.43	0.03
Magna			
Magna International	2.65	4.78	0.013
Magna Powertrain	2.65	2.18	0.005
Germania	0.83	3.46	0.003
Markham			
Clements Markham	0.83	4.42	0.004
Markham Waxers	0.83	3.67	0.003
Edwin Markham	0.83	2.89	0.002

Table 5: Example show the first three NE candidates for three NE mentions with scores

8 Conclusion

Our results show that graph ranking and cliques partitioning approaches in conjunction with the candidate confidence scores and entity coherence across a disambiguation graph can be used as an effective approach to collectively disambiguate named entity textual mentions in a document. Our proposed features are very simple and easy to extract, and work well when employed in PR or clique partitioning algorithms. Also, entity coherence is a discriminative feature when using graph models for NED. In future work we plan to explore enriching the edges between nodes, by incorporating semantic relations extracted from an ontology, and extending the scope of entity co-occurrence to be the document instead of the sentence. Also, it is worth investigating whether using the entity coherence score can help when evaluating clique weight in the clique partitioning algorithm.

References

Eneko Agirre and Aitor Soroa. 2008. Using the multilingual central repository for graph-based word sense disambiguation. In *LREC*.

Eneko Agirre and Aitor Soroa. 2009. Personalizing pagerank for word sense disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–41. Association for Computational Linguistics.

Ayman Alhelbawy and Rob Gaizauskas. 2012. Named entity based document similarity with svm-based re-ranking for entity linking. In *Advanced Machine Learning Technologies and Applications*, volume 322 of *Communications in Computer and Information Science*, pages 379–388. Springer Berlin Heidelberg.

Ayman Alhelbawy and Robert Gaizauskas. 2013. Named entity disambiguation using hmms. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, volume 3, pages 159–162.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Coen Bron and Joep Kerbosch. 1973. Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM*, 16(9):575–577.

Razvan C. Bunescu and Marius Pasca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *EACL*. The Association for Computer Linguistics.

S. Cucerzan. 2007. Large-scale named entity disambiguation based on wikipedia data. In *Proceedings of EMNLP-CoNLL*, volume 6, pages 708–716.

M. Dredze, P. McNamee, D. Rao, A. Gerber, and T. Finin. 2010. Entity disambiguation for knowledge base population. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 277–285. Association for Computational Linguistics.

S. Gottipati and J. Jiang. 2011. Linking entities to a knowledge base with query expansion. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 804–813. Association for Computational Linguistics.

Yoan Gutiérrez, Sonia Vázquez, and Andrés Montoyo. 2011. Word sense disambiguation: a graph-based approach using n-cliques partitioning technique. In *Natural Language Processing and Information Systems*, pages 112–124. Springer.

Yoan Gutiérrez, Sonia Vázquez, and Andrés Montoyo. 2012. A graph-based approach to wsd using relevant semantic trees and n-cliques model. In *Computational Linguistics and Intelligent Text Processing*, pages 225–237. Springer.

X. Han, L. Sun, and J. Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 765–774. ACM.

J. Hoffart, M.A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics.

H. Ji and R. Grishman. 2011. Knowledge base population: successful approaches and challenges. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1148–1158. Association for Computational Linguistics.

S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–466. ACM.

R.Duncan Luce and AlbertD. Perry. 1949. A method of matrix analysis of group structure. *Psychometrika*, 14(2):95–116.

P. McNamee and H.T. Dang. 2009. Overview of the tac 2009 knowledge base population track. In *Text Analysis Conference (TAC)*.

D. Milne and I.H. Witten. 2008. Learning to link with wikipedia. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 509–518. ACM.

D. Nadeau and S. Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Kamel Nebhi. 2013. Named entity disambiguation using freebase and syntactic parsing. In CEUR-WS.org, editor, *Proceedings of the First International Workshop on Linked Data for Information Extraction (LD4IE 2013) co-located with the 12th International Semantic Web Conference (ISWC 2013).* Gentile, A.L. ; Zhang, Z. ; d'Amato, C. & Paulheim, H.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, November.

L. Ratinov, D. Roth, D. Downey, and M. Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the Annual Meeting of the Association of Computational Linguistics (ACL).*

P. Sen. 2012. Collective context-aware topic models for entity disambiguation. In *Proceedings of the 21st international conference on World Wide Web*, pages 729–738. ACM.

Masumi Shirakawa, Haixun Wang, Yangqiu Song, Zhongyuan Wang, Kotaro Nakayama, Takahiro Hara, and Shojiro Nishio. 2011. Entity disambiguation based on a. Technical report, Technical report, Technical Report MSR-TR-2011-125, Microsoft Research.

Ravi Sinha and Rada Mihalcea. 2007. Unsupervised graph-basedword sense disambiguation using measures of word semantic similarity. In *Semantic Computing, 2007. ICSC 2007. International Conference on*, pages 363–369. IEEE.

Wenpu Xing and Ali Ghorbani. 2004. Weighted pagerank algorithm. In *Communication Networks and Services Research, 2004. Proceedings. Second Annual Conference on*, pages 305–314. IEEE.

Wei Zhang, Yan Chuan Sim, Jian Su, and Chew Lim Tan. 2011. Entity linking with effective acronym expansion, instance selection and topic modeling. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence-Volume Volume Three*, pages 1909–1914. AAAI Press.

Z. Zheng, F. Li, M. Huang, and X. Zhu. 2010. Learning to link entities with knowledge base. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 483–491. Association for Computational Linguistics.