Collective Approaches to Named Entity Disambiguation

Ayman A. Alhelbawy

A thesis presented for the degree of

Doctor of Philosophy

Department of Computer Science

University of Sheffield

United Kingdom

July, 2014

Dedication

I lovingly dedicate this thesis to my parents who have gave me every thing.

Acknowledgements

Thanks to Allah who is the source of all the knowledge in this world, and imparts as much as He wishes to any one He finds suitable. I would like to thank my supervisors, Robert Gaizuskas, who has supported me throughout my work on this thesis with his patience and knowledge. My special thanks to Mark and Rao Muhammad Adeel Nawab who helped me throughout my work and gave very useful suggestions regarding it.

I am thankful to my parents for their support, prayers, love and care throughout my life and they have played a vital role in achieving this milestone. My wife has always been a wonderful being to me and extended her whole-hearted support especially during my PhD studies, which I could not have completed without her. Finally, I am thankful to Fayoum University, Fayoum, Egypt, for funding this work.

Ayman Antar Alhelbawy 30 Jun, 2014

Abstract

Internet content has become one of the most important resources of information. Much of this information is in the form of natural language text and one of the important components of natural language text is named entities. So automatic recognition and classification of named entities has attracted researchers for many years. Named entities are mentioned in different textual forms in different documents. Also, the same textual mention may refer to different named entities. This problem is well known in NLP as a disambiguation problem. Named Entity Disambiguation (NED) refers to the task of mapping different named entity mentions in running text to their correct interpretations in a specific knowledge base (KB). NED is important for many applications like search engines and software agents that aim to aggregate information on real world entities from sources such as the Web. The main goal of this research is to develop new methods for named entity disambiguation, emphasising the importance of interdependency of named entity candidates of different textual mentions in the document.

The thesis focuses on two connected problems related to disambiguation. The first is Candidates Generation, the process of finding a small set of named entity candidate entries in the knowledge base for a specific textual mention, where this set contains the correct entry in the knowledge base. The second problem is Collective Disambiguation, where all named entity textual mentions in the document are disambiguated jointly, using interdependence and semantic relations between the different NE candidates of different textual mentions. Wikipedia is used as a reference knowledge base in this research.

An information retrieval framework is used to generate the named entity candidates for a textual mention. A novel document similarity function (NEBSim) based on NE co-occurrence

is introduced to calculate the similarity between two documents given a specific named entity textual mention. NEB-sim is also used in conjunction with the traditional cosine similarity measure to learn a model for ranking the named entity candidates. Naïve Bayes and SVM classifiers are used to re-rank the retrieved documents. Our experiments, carried out on TAC-KBP 2011 data, show NEBsim achieves significant improvement in accuracy as compared with a cosine similarity approach.

Two novel approaches to collectively disambiguate textual mentions of named entities against Wikipedia are developed and tested using the AIDA dataset. The first represents the conditional dependencies between different named entities across Wikipedia as a Markov network, where named entities are treated as hidden variables and textual mentions as observations. The number of states and observations is huge, and naïvely using the Viterbi algorithm to find the hidden state sequence which emits the query observation sequence is computationally infeasible given a state space of this size. Based on an observation that is specific to the disambiguation problem, we develop an approach that uses a tailored approximation to reduce the size of the state space, making the Viterbi algorithm feasible. Results show good improvement in disambiguation accuracy relative to the baseline approach, and to some state-of-the-art approaches. Our approach also shows how, with suitable approximations, HMMs can be used in such large-scale state space problems.

The second collective disambiguation approach uses a graph model, where all possible NE candidates are represented as nodes in the graph, and associations between different candidates are represented by edges between the nodes. Each node has an initial confidence score, e.g. entity popularity. Page-Rank is used to rank nodes, and the final rank is combined with the initial confidence for candidate selection. Experiments show the effectiveness of using Page-Rank in conjunction with initial confidence, achieving 87% accuracy, outperforming both baseline and state-of-the-art approaches.

Declaration

This thesis is an account of research undertaken between June 2010 and July 2014 at the department of Computer Science, University of Sheffield, Sheffield, United Kingdom.

Except where acknowledged in the customary manner, the material presented in this thesis is, to the best of my knowledge, original and has not been submitted in whole or part for a degree in any university.

Ayman A. Alhelbawy

July, 2014

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Over time, the Internet has became an increasing part of billions of people's daily life, and now massive amounts of new data are added to the Internet every day (e.g. news, research, blogs). The amount of data published on the internet increases exponentially, so it has become extremely difficult for users to find precisely the information they are looking for. Most web-pages are designed for human consumption and not for computer consumption. Even web search engines are only helpful in finding a good set of pages that may be related to the search query, but are unable to interpret the results, leaving the task of finding the precise piece of information to the user. Computers are only used to present the contents of web-pages, i.e. decoding different web script languages, and have no reliable way to process the semantics [Breitman et al., 2007].

In 2001, Berners-Lee et al. [2001] presented the concept of the Semantic Web . They defined the Semantic Web as an extension to the current web—in which informationin the form of natural language text is given a defined meaning that allows both humans and computers to work in co-operation. The task of giving a defined meaning to the information is a huge task and can be divided into many subtasks, including co-reference resolution, word sense disambiguation, named entity recognition and classification, and named entity disambiguation (NED).

In any document, particularly news stories, named entities (NE) are semantically richer than most vocabulary words [Petkova and Croft, 2007]. So, named entities are one of the main components of text on the web. A *textual named entity mention* is a pointer to a real world entity, such as a person, location, or organization. However, these pointers are ambiguous: one

named entity expression may refer to more than one real world entity. The relation between named entity textual mention and the real world named entity is many-to-many as the one real world entity may be referred to by more than one named entity textual mention.

A named entity may be a single word, such as "London" or "Microsoft", or a collection of words like "University of Sheffield" or "United Kingdom". Also, the named entity may be a dictionary word, meaning it may be found in the language dictionary, for example "Mark" is both a person name and an English verb, while the majority of named entities, like the name "David Cameron" or place "London", are not found in a language dictionary. There is no fixed dictionary for real named entities; new named entities arise every day, some of which are added to the knowledge bases. Textual mentions of named entities are also highly dynamic as many new textual mentions that refer to previously mentioned named entities are being added to online sources daily. The only available reference resources for real world entities are knowledge bases. Wikipedia is the best known such knowledge base. It is widely used as a reference knowledge base to disambiguate ambiguous named entity mentions by researchers working on this problem because of its breadth and free availability [Cucerzan, 2007, Han and Zhao, 2009a, Milne and Witten, 2008a, Ratinov et al., 2011]. It contains only references to relatively well known individuals, but is nevertheless suitable for research on this problem.

In general, named entities have a special importance in information extraction from text, or text mining. In the last two decades lots of research has been done on named entity recognition and classification, and good progress has been made [Nadeau and Sekine, 2007]. However, many of the recognized named entities are ambiguous, and it is very important for software agents—which aim to aggregate information on real world entities from sources such as the Web—to be able to identify which entities are the intended interpretation of different textual mentions. For example, the information associated with the basketball player "Michael Jordan" must be distinguished from the information associated with the football player "Michael Jordan". Also, it is important for search engines to correctly identify different names for the same named entity to get full coverage when searching for a named entity with different names. For example, the rapper MC Hammer's birth name is "Stanley Burrell" and correctly identifying documents in which "Stanley Burrell" is used to refer to him is important for completeness in information

gathering (of course Stanley Burrell, aka MC Hammer, needs to be distinguished from Stanley Burrell, the NBA basketball star). These examples illustrate the complexity of the many-to-many relation between NE textual mentions and the real world entity being named.

Named entity disambiguation approaches may be placed into two classes. The first class, which we refer to as *Individual entity disambiguation* approaches, addresses the problem of disambiguating an individual NE mention in a textual context (for example, in a search engine query or in a text document). In this case other NE mentions in the same context do not receive any particular attention, in particular they are not disambiguated themselves. Work addressing this problem includes [Bunescu and Pasca, 2006, Cucerzan, 2007, Han and Sun, 2011]. The second class of approaches may be termed *Collective entity disambiguation* approaches and in these all NE textual mentions in a context are jointly disambiguated.

In this work, we highlight both individual and collective disambiguation approaches. However, we hypothesis that when disambiguating a NE textual mention, the other NEs found in the same context will be a good source of information.

1.1 Motivation

In the last two decades a lot of attention has gone into analysing web text as a valuable source of continually updated information. Disambiguating named entities is very exciting task in NLP. It is a challenging task because of the domain diversity and dynamics of the knowledge bases and named entities. NED can help to improve performance in the following research domains:

- ***Information Retrieval***: Search engines needs named entity disambiguation to resolve the cross linking between different named entites with different textual mentions within web page text.

- ***Knowledge Base Builders***: As is well known, building a knowledge base is a very difficult task. First, it reacquires a lot of resources to navigate through a collection of data to build knowledge base nodes and fill each node with the required information. The second challenge is to update this knowledge base, as new data contains extra information. The research reported here may useful for this second task. For example, from new news wire

stories, entity linking may be used to locate the knowledge base nodes that are related to each entity,enabling KB contents to be updated.

- ***Information Seekers***: When someone reads an article, some terms may be unknown in the article or some terms may be of interest. The reader must then carry out another search to investigate the term of interest. This research will help to build new technologies to link named entities which are recognized automatically or manually by the reader to a knowledge base or other explanatory documents to get more information about that entity.

All in all, the main benefit to be gained from this research is providing an underpinning capability for use in other text mining applications and research.

1.2 Problem definition

The relation between names and the real world entities they denote is many-to-many: one entity may have several names, and the same name may be shared by multiple entities. Establishing which real world entity a name mention denotes in a particular textual context is the problem of named entity disambiguation (NED). We interpret this problem, more specifically, as the problem of disambiguating named entity textual mentions—that are annotated manually or by a Named Entity Recognition (NER) system in a document—against a set of named entities in a reference knowledge-base (KB). In addressing the problem we make two assumptions. First, we assume the correct interpretation for each NE textual mention has an entry in the knowledge base. Second, each NE textual mention is included in a text document which contains one or more other named entities.

1.3 Research Focus

The purpose of this research is to develop and evaluate new approaches to disambiguate the different textual mentions of different named entities within a document, linking them to the correct named entity reference in a specific knowledge base. The novel perspective in addressing the

NED problem is to use the mutual information gained from other co-occurring textual mentions of different named entities. overcoming the accuracy of existing approaches.

1.4 Thesis Contributions

The main contributions of this thesis are:

1. An NE based search framework for retrieving and scoring a reliable short list of NE candidates from a knowledge base.

2. An investigation of learn-to-rank approaches to re-rank the different candidates of a specific NE textual mention.

3. A collective NED approach based on Hidden Markov Models (HMMs).

4. Development of various approximations to use the Viterbi algorithm with a huge number of states, making it feasible for use in NED.

5. A graph based collective NED approach using Page-Rank in conjunction with local features and entity coherence to rank NE candidates.

6. A graph partitioning for collective NED, based on clique partitioning.

1.5 Research Findings

The main findings of this research are:

1. The named entity mentions in a document can help to disambiguate each other or, at least, be used to generate a short list of the disambiguation NE candidates. IR based approaches can use the other textual mentions successfully to find a short list of NE candidates, then use learn-to-rank approaches to re-rank these candidates for disambiguation.

2. The NED problem can be modelled as the problem of finding the NE sequence that emits a specific sequence of NE textual mentions. This approach is theoretically limited and can

not disambiguate all NE textual mentions properly because there may not be a sequential path of NE dependency between the candidates of different NE textual mentions.

3. Graph representation is the best for NE dependency or coherence as it can model all relations. The main problem with graph approaches is algorithm complexity. Effective employment of the NE candidate confidence scores may help to find the disambiguation candidates in the graph using linear time algorithms.

4. Using NE candidate confidence score as an initial node score in Page-Rank and recombining this score with the final PR score improves the results of NED.

5. Clique Partitioning can be used to find highly cohesive NE candidates in a graph. Starting with a seed clique and iteratively expanding it, or finding new seeds and expand those, improves the results of NED.

1.6 Thesis Outline

The rest of this thesis is organized into five chapters:

- Chapter 2: Literature Review and Existing Data Sets

  This chapter consists of two parts. The first explores work related to the named entity disambiguation task, like word sense disambiguation (WSD), and entity linking (EL). The main differences between these related tasks and named entity disambiguation are also highlighted. Following that a discussion of state-of-the-art approaches for NED is presented. The second part describes two different datasets which are widely used to evaluate state-of-the-art approaches, the TAC-dataset and AIDA-dataset. Finally, measures commonly used to evaluate NED system performance are presented.

- Chapter 3: Named Entity Based Document Similarity with SVM Re-ranking

  This chapter presents a named entity based document similarity approach, NEB-Sim, which retrieves a short NE candidate list for each named entity textual mention. This similarity function is based on NE textual mention co-occurrence in the knowledge base. Also,

re-ranking methods are applied using the NEB-Sim scores and other similarity scores to re-rank the candidate list and select the highest ranked candidate. Results show that using our NEB-Sim similarity score helps to find a short list of candidates that contains the correct candidate. Additionally, results show reliable relations between NEB-Sim and cosine-similarity scores that can be learned using the SVM^{rank} algorithm to re-rank the candidates and get the correct candidate in the first position.

- Chapter 4: Disambiguating Named Entities using HMMs

  This chapter presents a new formulation for the collective named entity disambiguation problem where it is framed as the problem of finding the best hidden state sequence using a Hidden Markov model (HMM). Three different approximations are presented to overcome problems with using the Viterbi algorithm when dealing with a huge number of states. Results show our approximations work well, passing the baseline and some state-of-the-art approaches.

- Chapter 5: Graph-Based Named Entity Disambiguation

  This chapter presents two collective disambiguation approaches based on a graph model. The first approach models the NED problem as one of ranking graph nodes using candidate confidence and coherence between different NE candidates. The second approach uses a clique partitioning algorithm to find the good cliques of candidates and iteratively expand these until all NE textual mentions are disambiguated. Results of both approaches show an improvement in the accuracy of NED, leading to scores that exceed the state-of-the-art as well as the baseline.

- Chapter 6: Conclusions and Future Work

  This chapter presents a summary of contributions and also discusses how these achieve our research goals. Some avenues for future work are also discussed.

1.7 Publications

The following publications have been produced during this research work:

1. Ayman Alhelbawy, Robert Gaizauskas."Collective Named Entity Disambiguation using Graph Ranking and Clique Partitioning Approaches" *Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014)*, 2014.

2. Ayman Alhelbawy, Robert Gaizauskas."Graph Ranking for Collective Named Entity Disambiguation." *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, 2014.

3. Ayman Alhelbawy, Robert Gaizauskas. "Named Entity Disambiguation Using HMMs." *In Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, vol. 3, pp. 159-162. IEEE, 2013.

4. Ayman Alhelbawy, Rob Gaizauskas. "Named Entity Based Document Similarity with SVM-Based Re-ranking for Entity Linking." *In Advanced Machine Learning Technologies and Applications,* pp. 379-388. Springer Berlin Heidelberg, 2012.

5. Amev Burman, Arun Jayapal, Sathish Kannan, Madhu Kavilikatta, Ayman Alhelbawy, Leon Derczynski, Robert Gaizauskas. "USFD at KBP 2011: Entity linking, slot filling and temporal bounding." *In Proceedings of Test Analysis Conference (TAC)*, 2012.

Chapter 2

Literature Review and Datasets

2.1 Introduction

Sense ambiguity is one of the old, well known, problems in NLP. It is easy for humans to disambiguate words, and names while it is not such a simple task for machines. Sense ambiguity has been a serious problem since the rise of machine translation in the early of 1940s, and has been a separate computational linguistics task since then. Until the twenty years ago, the sense ambiguity problem definition was limited to the ambiguity of common nouns, adjectives, and verbs. In the 1990s, the problem of named entity recognition and classification became a subtask of information extraction. Initially, little attention was payed to the ambiguity problem of named entities, and research focused mainly on recognizing and classifying named entities in text. After making good progress in recognizing and classifying named entities, research then moved to solving the next problem—that of NE ambiguity. In this chapter we present an overview of sense disambiguation problem tasks, and some detailed explanation of the NED problem.

The chapter is organized into four sections. Section 2.2 presents an overview of the disambiguation tasks related to the NED task, like word sense disambiguation (WSD), Linking with Wikipedia (Wikification), and entity linking (EL). A general framework for named entity disambiguation, and the state-of-the-art disambiguation approaches are explored and classified in section 2.3. Section 2.4 provides a listing and description of the most popular features used for NED. Section 2.5 presents a detailed description of the available datasets that have been

widely used in evaluating state-of-the-art approaches which we will also use to evaluate our proposed solutions. Finally, the evaluation measures used to evaluate the different approaches are described in section 2.7.

2.2 Overview of Sense Disambiguation Tasks

In this section, the sense ambiguity problem and its challenges are explored, and a short survey of some interesting disambiguation tasks is presented. In general, disambiguation is the process of selecting the proper candidate from a list of candidates, given an ambiguous entity within a specific context. Perhaps, the oldest disambiguation problem in NLP is that of word sense disambiguation (WSD). A brief description of the WSD task is presented in section 2.2.1. Furthermore, research into linking entities to a Wikipedia knowledge base is discussed in section 2.2.2.

2.2.1 Word Sense Disambiguation

Word sense disambiguation is one of the oldest defined computational linguistics tasks. It was formulated as a distinct task during the early days of machine translation. This task attracted the attention of researchers for many years and is still an open problem. WSD is an important task in natural language processing, which addresses the process of identifying which sense (meaning) of a word is used in a sentence, when the word has multiple meanings.

WSD approaches can be broadly classified into two main classes. The first, supervised approaches, use machine learning techniques to learn a classifier for each polysemous word using a labelled dataset, and classify a term (word) to its sense label [Agirre and Soroa, 2008, Boyd-Graber et al., 2007, Gutiérrez et al., 2012, Sinha and Mihalcea, 2007]. The second class of approach is the unsupervised methods, where no labelled datasets are provided. Graph based approaches are widely used for unsupervised disambiguation [Navigli and Lapata, 2007]. Knowledge resources like Wikipedia and WordNet are used to add different contextual features to the words or to enrich the graph [Ponzetto and Navigli, 2010]. In general, supervised approaches for WSD have achieved better results than unsupervised approaches [Navigli, 2009].

NED has some similarities with WSD, since both are concerned with meaning based on context. Some approaches are re-used to solve NED problems as shown in section 2.3. Nevertheless, there are still some differences, providing a new set challenges; some of these differences are listed below.

- WSD assumes a nearly static list of words with associated senses, e.g. the dictionary from an online resource such as WordNet. The challenge is to link the word in the context to the proper sense in this list. In contrast, in NED the challenge is to link the named entity textual mention to a *dynamic* list of knowledge base entries.

- A named entity textual mention may be an abbreviation or alias, e.g. "NY" or "the Big Apple" may be used to refer to New York city. In knowledge bases there is just one entry for each named entity. So, in our example, only one entry titled with "New York city" is found in the KB. This problem is not found in WSD, because all synonyms have entries in the dictionary as well.

- While WSD defines a word as a single token, a named entity may be referred to by a single token or series of tokens (i.e. "the Big Apple").

- WSD is only concerned with the dictionary term, which may be a noun, verb, or adjective, but a named entity may not be a dictionary term, and should be a person, organization, or location name.

2.2.2 Named Entity Linking

The problem of named entity linking is to identify and connect textual named entity mentions to a knowledge base entry that has some information about this mention. There are different tasks defined with this core definition. To Wikify, or link to a Wikipedia target, is to link any mention that has an entry in the Wikipedia knowledge base. So, the objective is to identify and link named entities (e.g. Microsoft, Barack Obama, Sheffield), events (e.g. The Second World War), theories (e.g. Pythagorean theorem), expression definitions (e.g. algorithm), etc., to the Wikipedia knowledge base [Mihalcea and Csomai, 2007].

In 2008, the US National Institute of Standards and Technology (NIST) initiated the Text Analysis Conference (TAC) to support research within the Natural Language Processing community by providing the infrastructure necessary for large-scale evaluation of NLP methodologies. One TAC track is the Knowledge Base Population (KBP) track which defined the task of entity linking (EL).

The entity linking task — as KBP defines it — is to determine, for each query, which knowledge base entity is being referred to, or if the entity is not present in the reference KB [McNamee and Dang, 2009]. A query in the KBP track consists of a named entity and the context for that entity to use in disambiguation. Disambiguation is one of the main challenges of this task because some entities will share confusable names (e.g. *George Washington* could refer to the president, the university, or the jazz musician; *Washington* could refer to a city, state, or person).

Many researchers have tried to tackle the problem from different points of view, depending on their experience in related NLP domains (Chang et al. [2010], Reddy et al. [2010], Varma et al. [2009]). Efforts have also been made to build standard resources for the evaluation of entity linking techniques by the Linguistic Data Consortium (LDC) at the University of Pennsylvania. All linguistic resources, including data, annotations, system assessment, tools, and specifications, have been created by LDC. These linguistic resources are distributed for NIST as part of the TAC KBP evaluations [Simpson et al., 2010]; this is discussed later, in section 2.5.1.

The entity linking task can be divided into the two subtasks of named entity disambiguation and identifying which mentions have no link in the knowledge base. In the following sections, we survey different techniques used to disambiguate named entities against the knowledge base.

2.3 Previous Work in NED

Before 2008, very few researchers were trying to find solutions for disambiguating different textual mentions of different named entities in Web pages. After NIST introduced the task of entity linking, and provided resources, many researchers started to pay attention to the NED problem and tackled it from different perspectives. This section presents a summary of the NED system architecture and different approaches used to disambiguate NE textual mentions.

2.3.1 General Definitions

Some expressions are frequently used by researchers working in named entity disambiguation; these are related to the Wikipedia KB. In this section, a list of these expressions are defined as follows:

- *Redirect Page:* A redirect page in Wikipedia is an aid to navigation; it contains no content itself, only a link to another page (target page) with a different name, and strongly relates to the concept of the redirect page name. It contains #REDIRECT [[target page]]. A redirect page may be created for a number of reasons:

  Misspellings: When Wikipedia entries with misspelled titles are corrected, the old titles are redirected to the new correct title. For example, "Barak Obama" is redirected to the entry with the correct spelling "Barack Obama";

  Alternative names: Some entities have different aliases. The different aliases are used to redirect to one name. For example, "44th President of the United States" is redirected to "Barack Obama", and informal name "Sheffield University" is redirected to "University of Sheffield";

  Short names: A short surface form may be used to refer to the full surface form of an entity. For example, the surface form "Obama" is redirected to the full surface form "Barack Obama";

  Abbreviations and initials: Abbreviation is frequently used in English as a short form that refers to an entity. In Wikipedia, if there is a primary topic to the abbreviation or the initial then a redirect page is created, and if there is no primary topic a redirect will be for a disambiguation page. For example, "USA" is the abbreviation which redirects to the primary topic "United States".

- *Disambiguation Page:* Disambiguation pages are specifically created for ambiguously named entities, and consist of links to articles defining the different interpretations of the named entity textual mention. The same string may have a redirect page to the primary topic and a disambiguation page to refer to different entities with the same name. For

example “USA” has a redirect page to “United States” and a disambiguation page that contains links to entities for which USA may be used as an acronym, such as “Union of South Africa”, “United Space Alliance”, “University of South Alabama”, “United States Army”, “United Spirit Arena”, and “Université’ Sainte-Anne”.

- *Wikipedia Hyper-links:* Wikipedia pages typically contain some links to external pages or other Wikipedia entries. These links are written as an anchor, which is the textual form that appears in the page and the entity to be linked with. External links are written using the html “href” tag, while internal links are formatted as [[mention—NE]]. *Mention* is the surface string form that is used to mention the NE, while *NE* refers to the page title. If the mention is equal to the NE, then the second parameter, NE, is ignored. Some different links to the “Barack Obama” page could be [[Obama—Barack Obama]], [[44th President of the United States—Barack Obama]], and [[Barack Obama]].

- *Wikipedia Categories:* A classification of the topics associated with each page. These are normally found at the end of the page. All pages under the same topic have some semantic relation. For example, “Barack Obama” is classified under the following topics: “African-American United States Senators”, “Illinois Lawyers”, “Politicians from Chicago, Illinois”, “Presidents of the United States”, and “Presidents of the United Nations Security Council”. It is formatted in the Wikipedia dump as [[Category:xyz]].

2.3.2 NED General Framework

In this section, a generic framework for NED systems is discussed. Figure 2.1 shows a general architecture which the majority of developed systems follow. Some approaches ignore certain steps that they do not consider important. A detailed description of the components in the architecture is presented in the following subsections.

2.3.3 Query Expansion

The target of query expansion is to find different forms of the NE textual mention that may refer to the same NE in the knowledge base. This phase may include different steps, like correcting

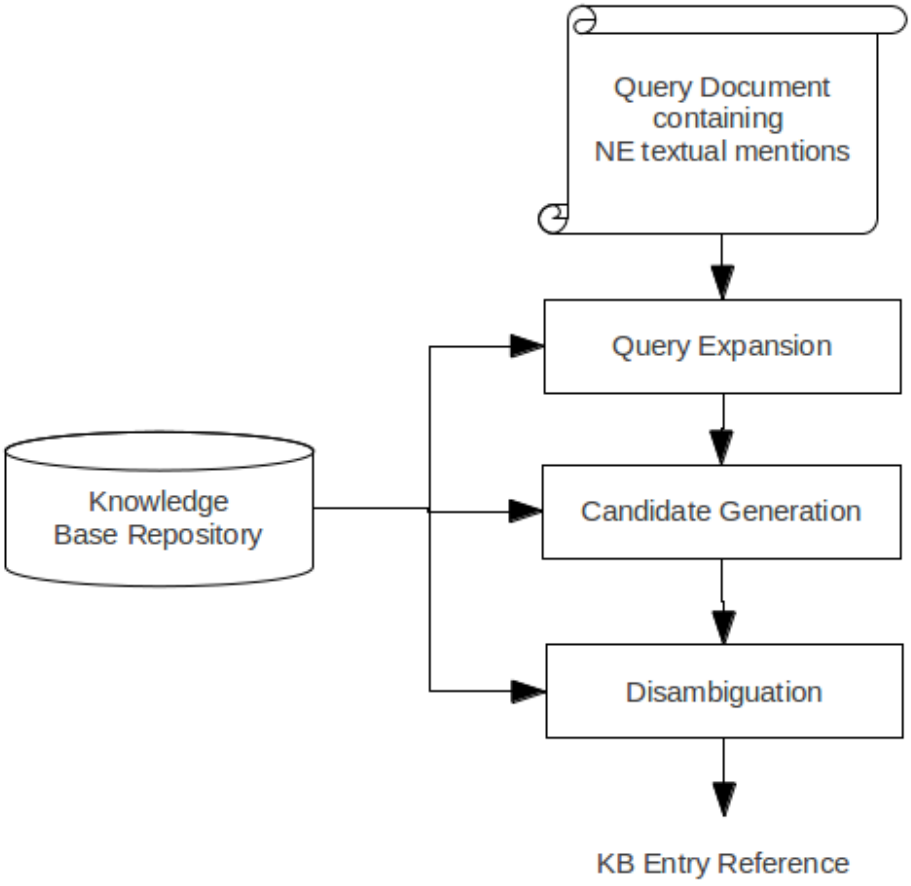


Figure 2.1: General Architecture of NED systems

spelling errors and expanding a short textual mention to its longer names that appear in the query document [Zheng et al., 2010]. Chang et al. [2010] used the Stanford NER system and Stanford deterministic co-reference system to find all textual mentions that are co-referent with the query textual mention. Generally, expanding the NE textual mention query improves both accuracy and recall, while ignoring it makes the disambiguation phase more difficult. We did not use any query expansion techniques, and focused on the candidate generation and disambiguation approach because using expansion techniques will affect the overall results. In other words, using expansion techniques (as in Chang et al. [2010]) will affect the total results by the Stanford deterministic co-reference system accuracy.

2.3.4 Candidate Generation

The process of evaluating the likelihood for each entry in the knowledge base to be the correct disambiguation entity for a specific NE textual mention in a specific context is excessively time consuming. So identifying an appropriate sub-set of candidate entries, i.e. candidate generation, becomes one of the main tasks of NE textual mention disambiguation. Candidate generation is defined as the task of reducing the set of KB entries to be examined to a feasible number of entries to allow the calculation within a feasible time. Methods used to generate the candidate list are classified into two classes, static and dynamic approaches. Static approaches use a predefined list and knowledge base repository to generate a candidate list of NE textual mentions. Dynamic approaches use search techniques to find all possible NE candidates. There is no independent evaluation of both candidate generation approaches, as researchers only select one of them to generate a candidate list for each textual mention, and the final evaluation is for the overall NED solution.

Static Candidate Generation

In this approach, a knowledge base repository is used to find all candidates of a specific NE textual mention. Wikipedia is used to build a knowledge base repository containing a huge map of real NE entries in the KB and the different textual forms using hyper-links, redirect pages, and disambiguation pages. Therefore, NE candidates for a specific NE textual mention are selected by finding the mention's corresponding entries in the knowledge base repository. This approach is widely used to find candidates for a specific NE textual mention (Han and Sun [2011], Han and Zhao [2009b], Reddy et al. [2010], Shen et al. [2012], Varma et al. [2010]). However, there are drawbacks and limitations to this approach as shown below:

1. Wikipedia is not guaranteed to contain all textual forms for different named entities. So some NEs may have different surface forms that are not found in the Wikipedia hyper-links, disambiguation pages, or redirect pages.

2. The NE textual mention may be found in Wikipedia, but without being used to refer to the true target NE, i.e. only mentioned in the article as plain text. One example is the

Wikipedia NE entry of “Cardiff International Arena”, which may be referred to as “CIA”, while there is no anchor for “CIA” which refers to that entity. The disambiguation page for “CIA” also has no mention of “Cardiff International Arena” as one of its possibilities for the acronym. There is no automatic method to generate these cases. Consequently, if “CIA” is mentioned in a query document, “Cardiff International Arena” never appears in the candidate list using this approach. This problem is not limited to acronyms, and also appears for full names. A frequent occurrence in the dataset is a short name which could be used to refer to the real NE in the context, like using “Austria” to refer to “Austria national football team” or “Manchester City” to refer to “Manchester City Football Club”. None of these short names are used in Wikipedia to refer to the real NE. Consequently, using this approach, the candidate lists generated for many NE textual mentions do not include the correct NE candidate.

We used this approach indirectly while using an HMM for disambiguation, as the conditional probability will only have a value greater than zero if it appears in the training data (details in chapter 4). Due to the limitations of this approach, recall falls very low, which affects the overall accuracy. As an alternative, dynamic candidate generation is used in our proposed graph-based solution (see chapter 5).

Dynamic Candidate Generation

This approach uses more advanced methods to select the NE candidates from the knowledge base. The basic dynamic NE candidate generation method is to use information retrieval techniques to search over the set of all titles of entries in the KB to find all NE candidates that are similar, or partially similar, to the NE textual mention. This search may use different similarity metrics like Dice score, skip bigram Dice score, or Hamming distance [Rao et al., 2013]. Another approach expands the query by adding some selected terms from the query document and extending the search scope by including Wikipedia documents [Reddy et al., 2010]. More advanced approaches use context modelling and document semantic analysis to create the candidate list [Ponzetto and Navigli, 2010] or to filter huge candidate lists [Nguyen and Cao, 2012].

Dynamic approaches may employ the static approach as an initial list and enrich it with

	Individual Disambiguation	Collective Disambiguation
Information Retrieval	YES	NO
Supervised Approaches	YES	NO
Graph based	YES	YES

Table 2.1: NED Approach classification Matrix

search results. The main advantage of this approach is of increasing the probability that the true NE may be included in the candidate list. The disadvantage is that a huge number of candidates must be evaluated. In our work, dynamic candidate list generation is always used. Additionally, an intelligent candidate generation approach using named entity based search is proposed in chapter 3.

2.3.5 Named Entity Disambiguation Approaches

Many disambiguation approaches have been proposed since 2009. These can be classified using two different perspectives. The first considers whether entities are disambiguated jointly, i.e. individual versus collective disambiguation approaches. The second is concerned with the disambiguation techniques used, like machine learning, information retrieval, semantic, and graph-based approaches. There is some overlap between different approaches, while others do not work together, for example, collective approaches do not intersect with information retrieval or machine learning approaches. Table 2.1 shows a summary of the intersection between these different approaches from the different classification perspectives.

In this section different approaches for disambiguating named entities are presented. Some research is concerned only with person name disambiguation [Ono et al., 2008], but the majority is concerned with disambiguating three major types of named entities (locations, persons, and organizations). A few researchers include all types of named entities, including a Misc NE class [Hoffart et al., 2011]. The difference between approaches is in the features used, as some are based on a specific class of named entities.

Dependency Perspective

Individual Disambiguation Individual named entity disambiguation approaches are used to

disambiguate a single textual named entity mention in a document without considering any other named entities in the document, or by considering other textual mentions but without disambiguation [Bunescu and Pasca, 2006, Dredze et al., 2010, Mihalcea and Csomai, 2007, Zhou et al., 2010].

Collective Disambiguation Collective approaches include those where the different mentions to different named entities in a document must be disambiguated jointly. The reference of a named entity for each textual mention will affect the references of the other named entity mentions [Han et al., 2011, Hoffart et al., 2011, Kleb and Abecker, 2010, Kulkarni et al., 2009].

Methodology Perspective This classification perspective is based on the disambiguation methodology. These approaches can be classified into three main categories, though some are mixes of them. The first class contains Information Retrieval (IR) based approaches, where a named entity mention is disambiguated by enhancing the search strategy and weighting criteria for the different candidates. The second class contains supervised approaches, where training data is required to train a learning model and learn a ranking function or weighting parameters. Finally, graph-based approaches model the problem as a solution graph that includes all possible solutions, and different approaches are developed to find the best candidate for every textual mention.

IR-Based Approaches

This set of approaches formulates the NED problem as an information retrieval problem, so the textual mention and its context form the query, and the knowledge base forms the document collection.

Cucerzan [2007] proposed a collective disambiguation approach that models the interdependence between the disambiguation decisions. He used a named entity recognizer to identify all named entity textual mentions in a query document. Alo, all Wikipedia categories are extracted and each entity in Wikipedia is assigned a set of categories. KB entities are represented as two vectors of the categories and the context entities, i.e. the named entities found in the en-

tity Wikipedia page. NE candidate lists for any NE textual mention in the query document are formed by finding all NEs mapped to the same textual mention, then the query document is represented as two vectors of the context entities, i.e. all NE candidates of all NE textual mentions in the document, and categories (union of categories of all NE candidates). The disambiguation process is defined as a maximization of the agreement between the KB entity context vector and the document context vector, as well as the agreement between the document categories vector and the KB entries category vector.

Gottipati and Jiang [2011] proposed an unsupervised approach to disambiguate individual NEs in a document by adopting a KL-divergence retrieval model (Lafferty and Zhai [2001]) to rank all candidates. The query language model is expanded by considering the local context of the NE textual mention in the query document and global knowledge obtained from the most likely NE in Wikipedia. The candidate selection process is based on the highest rank above a threshold, and the NE class agreement, i.e. the NE textual mention class identified by the NER tagger must be the same as the NE candidate class.

Supervised Disambiguation Approaches

These methods use machine learning algorithms to disambiguate an individual named entity's textual mentions. Machine learning is used in two contexts, to learn weights for the importance of different features for combination [Shen et al., 2012], and in learn-to-rank where the disambiguation problem is treated as a ranking problem. For learn-to-rank, training samples are used to learn a ranking function that minimizes a loss function; researchers propose three approaches.

Point-wise: Ranking is tackled as a classification of candidates. This approach does not select the best candidate for the textual mention, instead it considers the different candidates independently and assigns to each candidate the probability that it is the correct one, allowing the most probable candidate to be selected as the disambiguation candidate. Different classifiers may be used to find such probabilities. Zhang et al. [2010] employed a SVM binary classifier to learn context compatibility for disambiguation candidates. Milne and Witten [2008a] proposed using different classifiers, such as C4.5, Naïve Bayes, and SVM, to train a model with different

features to link with Wikipedia.

The point-wise approach is widely used in information retrieval to rank the resulting documents, but is rarely used in NED. Possible uses for a point-wise approach in NED may be to reduce the candidate list, or to decide whether the correct KB entry is included in the candidate list regardless of the reason (which may be due to omissions by the candidate list generation, or that the entry has not yet been added to the actual KB). Using this approach, each candidate is given a score independently of the other candidates. The main disadvantage of this approach is that it ignores the relationship between candidates — in other words, the preference between different candidates is ignored.

Pair-wise: Ranking is tackled as a classification of candidate pairs; the objective function is to minimize the number of misclassified pairs. Each pair of instances (a, b) is labelled with a being more relevant than b, or b being more relevant than a. Evaluating the preferences between candidate pairs overcomes the disadvantage of the point-wise ranking [Rao et al., 2013].

Different algorithms have been developed to minimize the number of misclassified pairs, like RankBoost [Freund et al., 2003], Ranking Perceptron [Zheng et al., 2010], and RankNet [Burges et al., 2005]. The first use of the pair-wise learn-to-rank approach in NE disambiguation was by Bunescu and Pasca [2006], where a SVM kernel is used to compare the context around the NE textual mention and the context of the candidate entity, in combination with the estimated contextual words and the NE candidate Wikipedia categories. Joachims [2002] proposed SVM^{rank} which is an adaptation of the SVM algorithm using the maximum margin approach to learning the preference between a pair of objects [Joachims, 2002, 2006]. Maximum margin approaches assume the correct NE candidate y should receive a higher score than all other candidates, $\hat{y} \in Y, \hat{y} \neq y$ plus a margin γ. SVM^{rank} is widely used in NED to learn the preference between different candidates [Dredze et al., 2010, Rao et al., 2013]—providing us with confidence in its use for ranking in our NE based document similarity approach(see chapter 3).

List-wise: Ranking is used to learn from lists of candidates. This approach tries to optimize the value of the evaluation measure, averaged over all queries in the training data. Different

algorithms like SVM^{map} [Yue et al., 2007], and ListNet [Cao et al., 2007, Zheng et al., 2010] has been developed to learn the ranking, using mean average precision (MAP) to calculate the list-wise loss function.

Graph-Based Approaches

Graph models has been widely used in word sense disambiguation (WSD), which has many similarities with NED (Gutiérrez et al. [2011, 2012]). Graph-based approaches have also been used successfully for the NED problem, usually with collective approaches. Guo et al. [2011] used a directed graph, with both textual mentions and NE candidates as graph nodes; edges connect textual mentions to candidates, or vice versa, but there are no links between mentions or between candidates. The rank of each candidate is calculated based on the out-degree and in-degree links. The key point in this approach is the links found between some textual mentions and NE candidates of other textual mentions. However, the interdependency between different candidates of different textual mentions is not represented. Hachey et al. [2011] proposed a graph-based approach which initialises a graph of unambiguous named entities, and assumes that textual mentions with only one NE candidate are unambiguous. Other candidates are added to the graph if they have a link to one or more graph nodes with a specific length through the link pages or categories. This assumption is not accurate enough to rely on, as it makes further decisions of the textual mentions depend on the accuracy of the candidate generation phase, which already has a trade off between recall and the candidate list size.

Other researchers pay more attention to the NE candidates interdependence. Han et al. [2011] use local dependency between the 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. Random graph walk models are used, but this does not give significant improvement; Gentile et al. [2009] reported 0.06% improvement over Cucerzan's approach, and Liu [2009] reported 92.9% as the highest accuracy with random walks for a transition count of one, while the exact match approach achieves 98.4% accuracy. The exact match approach tries to find the exact match string first and if no entity with exact string match exists, then it selects the entity which is the most frequent attached to the NE

textual mention. Hoffart et al. [2011] poses the problem as one of finding a dense sub-graph, which is infeasible for graphs of any large size, as finding the densest sub-graph is a NP-hard problem. 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 finds the densest graph. We compare the graph-based model results presented in chapter 5 to Hoffart's results, as the same dataset is used for evaluation. Our graph-based approach differs from Hoffart's work in evaluating all graph nodes using the Page-Rank algorithm, without resorting to reduction or greedy algorithms.

2.4 NED Popular Features

Many features have been used for NED. Features may be separated into two classes: *local features* are extracted from the query document without any external sources, while *global features* are calculated independently of the query document and are based solely on external sources. A summary of the most well known local features are listed in section 2.4.1, and section 2.4.2 lists some of the popular global features.

2.4.1 Local Features

Local features are popular and many researchers have proven their importance. This type of feature takes into account the context of the ambiguous NE textual mention and the document concepts. The following list shows some frequently used local features.

Named Entity Type Matching

A binary feature that indicates NE class agreement between the annotated NE textual mention in the query document and the candidate NE class in the KB [Dredze et al., 2010, Zhang et al., 2010].

Cosine Similarity

A lexical feature based on the vector-space model, presented by Salton et al. [1975]. The

cosine similarity between a document d and query q is defined as shown in equation 2.1.

$$sim(d,q) = \frac{dq}{\|d\| \, \|q\|} = \frac{\sum_{i=1}^{N} w_{i,d} w_{i,q}}{\sqrt{\sum_{i=1}^{N} w_{i,d}^2} \sqrt{\sum_{i=1}^{N} w_{i,q}^2}} \tag{2.1}$$

Where $V_d = [w_{1,d}, w_{2,d}, \ldots, w_{N,d}]^T$ is the weight vector for document d;

and:

$$w_{t,d} = tf_{t,d} log \frac{|D|}{|\{d' \in D | t \in d'\}|} \tag{2.2}$$

where:

$tf_{t,d}$ is the term frequency of term t in document d;

$|D|$ is the total number of documents in the document set;

$|\{d' \in D | t \in d'\}|$ is the number of documents containing term t;

and $log \frac{|D|}{|\{d' \in D | t \in d'\}|}$ is the inverse document frequency.

Different researchers use cosine similarity for different features in different contexts. Let $Ctxt(e)$ denote the context of the NE e, i.e. the top 200 token TF-IDF summary of the text within which the entity is hyperlinked in Wikipedia; $text(e)$ denotes the top 200 token TF-IDF summary of the entity page in Wikipedia; $text(m)$ refers to the textual mention tokens; $Ctxt(m)$ the context of NE textual mention m, i.e. N-token window around m. Then, four variations of similarity measure may be calculated as follows:

1. cos-Sim$(text(e), text(m))$
2. cos-Sim$(Ctxt(e), text(m))$
3. cos-Sim$(text(e), Ctxt(m))$
4. cos-Sim$(Ctxt(e), Ctxt(m))$

This feature is widely used in NED [Dredze et al., 2010, Fader et al., 2009, Ratinov et al., 2011, Zheng et al., 2010]. Cosine similarity is not a reliable feature for NED, especially when trying to find the similarity between the textual mention itself and KB entry title, as shown in detail in chapter 5.

Entity Context Probability

Encodes the context of the named entities, i.e. $P(c|e)$, where c is context of the named entity e. For a specific context, a higher probability will be assigned to the named entity which frequently appears with that context. Han and Sun [2011] proposed an entity context model to estimate the distribution $P(c|e)$ by encoding the context of an entity e in a unigram language model. They define the context as the surrounding window of 50 terms, and used formula 2.3 to find the entity context probability.

$$P(c|e) = P_e(t_1)P_e(t_2)P_e(t_3)\dots P_e(t_n) \tag{2.3}$$

where $P_e(t) = \frac{Count_e(t)}{\sum_t Count_e(t)}$, and $Count_e(t)$ is the frequency of occurrence of term t in the context of the named entity e.

2.4.2 Global Features

Global features refer to context independent features, i.e. calculated independently of the query document or the local context of the NE textual mention. It is used widely and successfully in the NED task, as it is easy to calculate and can be calculated once offline. The following is a shortlist of the most popular global features used for the NED task:

Word Category Pair

This feature was first proposed by Bunescu and Pasca [2006] for NED and was used in much research, including Zhang et al. [2010], where they used word–category pairs extracted from the Wikipedia article as a good signal for disambiguation. Each Wikipedia article has been assigned at least one category, so all words in the article can be assigned the article categories. For each word in the query document, or words in the NE textual mention context, a list of word–category pairs can be generated. This feature was used by Bunescu and Pasca [2006] in conjunction with the cosine similarity to rank a specific NE candidate. Zhang et al. [2010] formalised this as the probability of a word appearing with different categories in the Wikipedia KB.

Entity Popularity

The probability of an entity occurring in the KB. This feature has been implemented in

different formulas with slightly different meanings; Han and Sun [2011] defined it as in equation 2.4, while Ratinov et al. [2011] defined it as the fraction of Wikipedia pages that have a link to the NE e.

$$P(e) = \frac{Count(e) + 1}{|M| + N} \tag{2.4}$$

where $Count(e)$ is the number of references to the entity e in the KB, M is the set of textual mentions referring to e, and N is the total number of entities in the KB.

Mention–Entity Popularity

The fraction of times that the NE candidate e is linked to the textual mention m. To calculate this probability, all Wikipedia hyper-links are used as the source of mentions (anchor text) and NE (the KB entry URL) association to build a dictionary of mention–entity frequency. This feature is widely used, and is also known as *commonness* [Han and Zhao, 2009a, Milne and Witten, 2008a, Ratinov et al., 2011, Shen et al., 2012]. Mention–Entity popularity score can be calculated using equation 2.5.

$$P(e|m) = \frac{Count_m(e)}{\sum_{e_i \in E_m} Count_m(e_i)} \tag{2.5}$$

where $Count_m(e)$ is the number of associations between the NE candidate e and the textual mention m, and E_m is the set of entities to which the textual mention m refers. Some researches like [Nguyen and Poesio, 2012]. normalize it over all mentions by dividing by the probability of the textual mention $p(m)$ as show in equation 2.6.

$$commonness(e, m) = \frac{p(e|m)}{p(m)} \tag{2.6}$$

Entity–Mention Popularity

The probability of textual mention m, given a specific NE e. Like mention–entity popularity, all Wikipedia hyper-links are used as a source for the mention–entity association

dictionary. Entity–Mention popularity score is calculated using equation 2.7.

$$P(m|e) = \frac{Count_e(m)}{\sum_{m_i \in M_e} Count_e(m_i)} \tag{2.7}$$

where $Count_e(m)$ is the number of associations between the textual mention m and the NE candidate e, and M_e is the set of mentions that refer to the entity e [Han and Sun, 2011].

Entity Semantic Relatedness

An adaptation of Normalized Google Distance (NGD) [Cilibrasi and Vitanyi, 2007] to use Wikipedia links rather than Google's search results, also called the Wikipedia Link-based Measure (WLM) proposed by Milne and Witten [2008b]. This feature is widely used in a number of different approaches ([Han and Zhao, 2009a, Milne and Witten, 2008a, Nguyen and Poesio, 2012, Ratinov et al., 2011, Zhou et al., 2010]). Equation 2.8 calculates the semantic relatedness between two entities e_1 and e_2, where e_1 and e_2 are two Wikipedia pages of interest, E_1 and E_2 are the sets of all pages that links to e_1 and e_2 respectively, and W is the set of all Wikipedia pages.

$$SR(e_1, e_2) = \frac{\log(\max(|E_1|, |E_2|)) - \log(|E_1 \bigcap E_2|)}{\log(|W|) - \log(\min(|E_1|, |E_2|))} \tag{2.8}$$

2.5 Evaluation Datasets

Despite the different datasets that have been manually developed to evaluate the linking to Wikipedia tasks, there are no established benchmarks for NED. The only available related benchmark is the TAC KBP entity linking task. NIST has released a dataset for use in the TAC KBP entity linking task (EL), referred to as the TAC-dataset; a detailed description is presented in section 2.5.1. The TAC-dataset is suitable for single entity disambiguation approaches. There are hand-annotated datasets for NED, such as the one reported in Kulkarni et al. [2009], though this is quite small and uses an out of date snapshot of Wikipedia. Another dataset called AIDA was prepared by Hoffart et al. [2011] for the NED task. The AIDA-dataset is based on the CoNLL-

2003 data for NER tagging, with the majority of the tagged NE mentions disambiguated against Wikipedia. It is described in section 2.5.2.

An important difference between the TAC and AIDA datasets is the entity type scope. All annotated mentions in the TAC-dataset were selected to be references to a specific named entity that was classified as Person, Organization, or Place; annotated mentions in the AIDA-dataset refer to these named entity types but in addition also include a Miscellaneous type (which includes events, eras in time, languages, religions, film titles, book titles, etc.).

2.5.1 TAC Dataset

Efforts have been made to build standard resources for the evaluation of entity linking techniques by the Linguistic Data Consortium (LDC), at the University of Pennsylvania. These have resulted in a set of linguistic resources, including data, annotations, system assessment, tools, and specifications, distributed for NIST as part of the TAC KBP evaluations [Simpson et al., 2010].

The KBP dataset consists of a reference knowledge base (KB) and a collection of documents that contain potential mentions of, and information about, the target entities for the KBP evaluation tasks. All datasets are prepared by LDC, ensuring they are well-formed XML and can thus be parsed using a standard XML parser. In 2009, the LDC released the first version of this data set, containing 1,289,649 data files collected from various genres. The following year, 63,943 new documents from a new web collection, and 424,296 documents from the existing GALE web collection, were added, resulting in the 2010 dataset of 1,777,888 documents for use in linking to a knowledge base. Table 2.2 summarizes the data genres and number of documents; the numbers are the same for 2009 and 2010, with the exception of the web collection, which was added in 2010.

The second part of this dataset is the knowledge base. LDC used the October 2008 snapshot of Wikipedia to construct a reference KB of 818,741 entities to support TAC-KBP. As presented in figure 2.2, each entity has the following items:

- **Entity ID** A unique identifier for each entity in the knowledge base.

Genre		Documents
bc	broadcast conversation transcripts	17
bn	broadcast news transcripts	665
cts	conversational telephone speech transcripts	1
ng	newsgroup text	562
nw	newswire text	1,286,609
wl	weblog text	1,795
wb	web collection (*2010 only*)	488,239

Table 2.2: Source Collection Corpus Released by LDC in 2009 & 2010

```
<?xml version='1.0' encoding='UTF-8'?>
<knowledge_base>
.
.
.
.
<entity wiki_title="Barepot" type="GPE" id="E0009429" name="Barepot">
<facts class="infobox UK place">
<fact name="country">England</fact>
<fact name="latitude">54.64</fact>
<fact name="longitude">-03.53</fact>
<fact name="official_name">Barepot</fact>
<fact name="shire_county"><link>Cumbria</link></fact>
<fact name="region">North West England</fact>
<fact name="os_grid_reference">NY0129</fact>
</facts>
<wiki_text><![CDATA[Barepot

Coordinates:

Barepot used to be a village in Cumbria, England. As Workington and Seaton grew,
Barepot and also Seaton became districts of Workington. Both Seaton and Barepot
share a Workington post code (CA14). Barepot has about 70 houses and is situated
on the River Derwent. There are no transport links (e.g. Workington Circulars),
but Barepot is only a 5-10 minute walk into the centre of Workington.
]]></wiki_text>
</entity>
.
.
.
.
</knowledge_base>
```

Figure 2.2: Example KB Entry

- **Wikipedia page title** A canonical name for the Wikipedia page.

- **Wikipedia page name** The title for the Wikipedia page.

- **Entity type** The type assigned to the entity; PER (person), ORG (organization), GPE (geo-political), or UKN (unknown). Types were assigned by the LDC in a processing phase after assigning UKN as a default type for all entities. The assignment process

depends on the type of an article's Infobox, if any, so this mapping was determined by the type most likely associated with a given Infobox name (e.g. entity id = E0009430 Infobox_ School is ORG). The assigned types are not 100% accurate (e.g. entity id = E0009382 Infobox_ Company is UKN). A breakdown of entity types and their frequencies in the KB is presented in table 2.3.

Type	Entities
GPE	116,498
ORG	55,813
PER	114,523
UKN	531,907

Table 2.3: A Break Down of KB Entity Types

- **Infobox** A table containing a list of attributes about the page's subject. Some types of Infobox are discarded, since they contain no key-value pairs; so not all entities have an Infobox. Some attributes (e.g. picture captions) are ignored, and other features may be parsed incorrectly. When a Wikipedia article contains more than one Infobox, only the first is included in the KB.

- **Wikipedia page text** A stripped version of the text of the Wikipedia article. This item may be helpful in disambiguating the target mention.

Queries and Golden Standard

All queries are formatted as XML files, where each node contains an entity mention and the document ID that contains this mention. Since this document provides a context for the entity mention, it may be helpful for mention disambiguation. Most target entities were selected to include ambiguous names; Simpson et al. [2010] describe the selection process performed by the LDC. A breakdown of the given queries used in TAC 2009 and TAC 2010 is presented in table 2.4, and a sample of XML entries in the query file is presented in figure 2.3

Since the queries are designed specially for evaluation purposes, queries should cover the following problems:

Year	Queries	NIL/KB	ORG	PER	GPE	Total
2010	2250	NIL	446	538	246	1230
		in KB	304	213	503	1020
2009	3904	NIL	1697	272	160	2129
		in KB	1013	255	407	1675

Table 2.4: Number of Queries, By Type and KB Presence

```
<?xml version='1.0' encoding='utf8'?>
<kbpentlink>
  .
  .
  <query id="EL1914">
    <name>Mahdi</name>
    <docid>AFP_ENG_20070406.0397.LDC2009T13</docid>
  </query>
  <query id="EL1915">
    <name>Mahdi</name>
    <docid>AFP_ENG_20070417.0098.LDC2009T13</docid>
  </query>
  <query id="EL1916">
    <name>Mahdi</name>
    <docid>AFP_ENG_20070427.0442.LDC2009T13</docid>
  </query>
  <query id="EL1917">
    <name>Mahmood Shah</name>
    <docid>APW_ENG_20070730.0206.LDC2009T13</docid>
  </query>
  <query id="EL1918">
    <name>Mahmood Shah</name>
    <docid>APW_ENG_20080922.0278.LDC2009T13</docid>
  </query>
  <query id="EL1919">
    <name>Mahmood Shah</name>
    <docid>APW_ENG_20080922.0397.LDC2009T13</docid>
  </query>
  <query id="EL1920">
    <name>Mahmood Shah</name>
    <docid>LTW_ENG_20080128.0022.LDC2009T13</docid>
  </query>
  <query id="EL1921">
    <name>Mahmood Shah</name>
    <docid>LTW_ENG_20080910.0094.LDC2009T13</docid>
  </query>
  <query id="EL1922">
    <name>Mahmood Shah</name>
    <docid>NYT_ENG_20070629.0167.LDC2009T13</docid>
  </query>
  .
  .
</kbpentlink>
```

Figure 2.3: Example Query XML File

- *Name Variations*: Since an entity often has multiple mention forms, such as aliases (Rommel vs "The Desert Fox").

- *Alternative Spellings*: like “Ossama”, “Ussamah”, or “Oussama”.

- *Abbreviations*: like NIST for “National Institute of Standards and Technology”.

Examples for these cases are presented by McNamee et al. [2010] as part of his breakdown of the queries used in the TAC 2010 evaluation. Query #1213 is a good example of the ambiguous acronyms problem, since the “DRC” mention refers to “the Democratic Republic of Congo”, however, both ‘DCR’ and ‘DRC’ appear as acronyms in the provided document. Another example of entity name alias is query #1717, where “Iron Lady” refers to Ukrainian Prime Minister “Yulia Tymoshenko”.

The TAC dataset was prepared for two tasks, “Entity Linking” and “Slot filling”, and this explains the big difference between the number of documents and the number of annotated mentions. This dataset is suitable for the EL task, and for single named entity disambiguation approaches. However it is not suitable for evaluating collective disambiguation approaches because not all named entity textual mentions are annotated in the query document.

2.5.2 AIDA Dataset

Hoffart et al. [2011] proposed the AIDA-dataset[1] to test their system, *Accurate Online Disambiguation of Named Entities*. This dataset is based on the CoNLL-2003 data for NER tagging. So, the dataset is pre-annotated and used to be a gold standard in NER tagging task before. Most tagged NE mentions have been manually disambiguated against multiple knowledge bases, such as Wikipedia, YAGO [Suchanek et al., 2007], and Freebase[2] [Bollacker et al., 2008]; details are summarized in table (2.5). There are some annotated NE textual mentions that do not have a proper entry in the KB, so we refer to them as ”Mentions Not Linked to Wikipedia”. There are some key differences between the AIDA-dataset and the TAC-dataset:

- The number of named entity mentions in the AIDA-dataset is significantly greater than in the TAC-dataset;

[1] The AIDA-dataset is available to download from `www.mpi-inf.mpg.de/yago-naga/aida` (last visited 9-April-2014)

[2] `www.freebase.com` (last visited 9-April-2014)

Property	Count
Documents	1,393
Annotated Mentions	34,956
Mentions Not Linked to Wikipedia	7,136

Table 2.5: AIDA Dataset Properties

- in the AIDA-dataset, the majority of named entities in each document are annotated, while only one or two named entities are annotated in each TAC-dataset query document;

- annotated named entity mentions in the TAC-dataset are always one of three standard classes (person, location, organization), while the AIDA-dataset includes the additional Misc class.

The number of NE textual mentions in dataset documents varies from one to 157 NE textual mention. Table 2.4 shows that around 85% of documents contain less than 30 NE textual mentions while 15% contains more than 30. Only one document contains 157 NE textual mentions and all other documents contains less than 100.

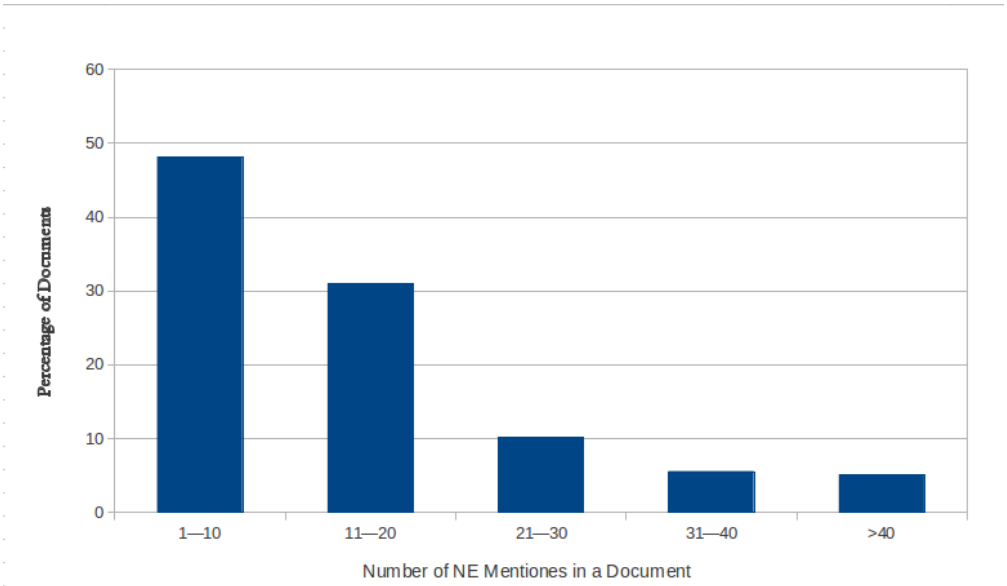


Figure 2.4: AIDA dataset documents analysis

2.6 Knowledge Base

There are a number of available knowledge bases for linking. We used the most popular, a snapshot of Wikipedia; most state-of-the-art approaches use this as a reference knowledge base, making our results comparable. However, different dumps of Wikipedia are used in different approaches. We used two different Wikipedia dumps in our experiments. The first is the Wikipedia 2008 snapshot provided by NIST for the Entity Linking task in TAC 2009, 2010, and 2011, which is used in evaluating the document similarity using NEs approach (see chapter 3). The second is the Wikipedia 2012 snapshot, which is used to evaluate HMMs, and graph model approaches (see chapters 4 and 5).

2.7 Evaluation Measures

Different evaluation metrics have been used to evaluate different approaches to NED. The key metric used by researchers in NED task is accuracy, though some researchers treat NED as an IR problem and use precision and recall to measure system performance.

2.7.1 Accuracy

We use accuracy as one principal evaluation metric. There are two types of accuracy measures: micro-averaged accuracy and macro-averaged accuracy.

Micro-averaged accuracy corresponds to the percentage of correctly disambiguated textual mentions taken across all entities and all documents. It is calculated as shown in equation 2.9, and is used in most state-of-the-art evaluations, and by the majority of researchers, to evaluate NED system performance [Chen and Ji, 2011, Du et al., 2013, Gentile et al., 2009, Gottipati and Jiang, 2011, Han and Sun, 2011, Jačala and Tvarožek, 2012, Liu, 2009]. Micro-accuracy was also the primary evaluation metric in TAC 2009.

$$A_{micro} = \frac{\text{Correctly Disambiguated Mentions}}{\text{Total NE Mentions}} \tag{2.9}$$

Macro-averaged accuracy has two different interpretations. The first is the average per-

centage of correctly disambiguated textual mentions for each unique named entity, as shown in equation 2.10. In other words, it is the average of micro-accuracy for each named entity $E_i \in E$. We follow this interpretation in our experiments on graph model approaches presented in chapter 5. This is the formal interpretation used in TAC for the KBP entity linking task, and also in other research [Chang et al., 2010, Dredze et al., 2010, Han and Sun, 2011].

$$A_{macro} = \frac{\sum_{i}^{|E|} \frac{\text{Num of Mentiones Correctly Linked to } E_i}{\text{Num of menstions must be linked to } E_i}}{|E|} \tag{2.10}$$

where E is the set of all NE entries in the KB that should be linked, and $|E|$ is the number unique NEs in the dataset.

The second interpretation is that macro-averaged accuracy is the average percentage of correctly disambiguated textual mentions for each document $D_i \in D$, as shown in equation 2.11.

$$A_{macro} = \frac{\sum_{i}^{|D|} \frac{\text{Num of Correctly disambiguated mentions in } D_i}{\text{Num of mentions in } D_i}}{|D|} \tag{2.11}$$

where D is the set of all documents in the dataset, and $|D|$ is the number of documents. We used this interpretation in experiments used to test the HMMs approach presented in chapter 4. This interpretation is also used by some researchers, such as [Hoffart et al., 2011, Shirakawa et al., 2011]

2.7.2 Precision and Recall

Precision and recall are well-known metrics in information retreival tasks. However, they are not widely used in evaluating NED. Some NED approaches include the named entity recognition task as part of the NED system, i.e. identify the NE textual mentions in the document which refer to an entry in the KB, for use in collective disambiguation [Han et al., 2011, Hassell et al., 2006]. One of the reasons is that not all NE textual mentions in the dataset are annotated, e.g. in the TAC dataset only one or two NE textual mentions are annotated; while in another dataset, many NE textual mentions may be annotated, but there is no guarantee of 100% coverage. The intuition is that annotating the missed NE textual mentions may help disambiguation.

Consequently, it is the proportion of annotated NE textual mentions which is evaluated. Precision is defined as the proportion of mention–entity assignments that match the ground truth assignments, while recall is the proportion of ground truth assignments that could be assigned by the proposed method. Equations 2.12 and 2.13 show formulae for the precision and recall.

$$Precision = \frac{|A \bigcap B|}{|B|} \tag{2.12}$$

$$Recall = \frac{|A \bigcap B|}{|A|} \tag{2.13}$$

where A represents the set of mention–entity assignments achieved by the proposed system, and B represent the ground truth mention–entity assignments.

As this metric evaluates the results of two different tasks, NE recognition and disambiguation, it is not valuable for evaluating the performance of the disambiguation task alone.

Hoffart et al. [2011] used precision at a specific recall level, $P@n$, and defined the recall level n as the n-most confident candidates of every specific textual mention. Thus, calculating the precision at recall level one (precision@1.0) is equal to accuracy, since it is defined as the overall correctness of the textual mentions assigned to a KB entity in the ground truth. This allows us to compare our results with Hoffart et al.'s even though they appear to be using different evaluation metrics.

Chapter 3

Named Entity Based Document Similarity with SVM Re-ranking*

As mentioned in chapter 2, candidate list generation is an important step in named entity disambiguation. This step is concerned with reducing the search space of candidate entities for an ambiguous NE textual mention to a limited number. So, we have a trade-off between the size of candidate list and the recall. The main objective in this chapter is to present a new document similarity function (NEB-sim) based on named entity mentions co-occurrence and show how it can contribute in NED.

In this chapter, the problem of named entity disambiguation is tackled as an information retrieval problem, i.e. the KB is treated as a document collection and the NED task as that of returning the correct document given the query mention and query document. A document similarity function, NEB-Sim, has been developed to calculate the similarity between two documents given a specific NE mention in one of them (see section 3.2). NEB-Sim is used in conjunction with cosine similarity to learn a model for ranking the reference knowledge base candidate documents. Naïve Bayes and SVM classifiers are used to re-rank the retrieved documents (see section 3.3). Our experiments, are carried out on the TAC-dataset (see section 2.5.1). Results show NEB-Sim achieves significant improvement in recall as compared with a cosine similarity

*Parts of this chapter has been published in *Advanced Machine Learning Technologies and Applications*, pp. 379-388. Springer Berlin Heidelberg, 2012

approach. Also, using a machine learning technique significantly improves the recall compared to each similarity function separately (see section 3.4). Finally, some conclusions and discussion of the presented approach and its relevance to the research problem are presented in section 3.5

3.1 Introduction

The problem of Information Retrieval (IR) has a special importance in NLP. General IR techniques do not always give the best solution for different tasks. So, many adaptations have been made for different tasks to obtain better accuracy in terms of precision and recall. Many approaches to NED adopt a Vector Space Model (VSM) using cosine similarity with a TF-IDF weighting scheme. Furthermore, they explore various query formulation strategies including using the query mention, the sentence containing the query mention, or a window of words surrounding the query mention, and a selected set of words [Reddy et al., 2010]. Traditional search methods consider the NE mention terms plus some selected terms from the query document according to the query formulation scheme, but not all of these terms are useful in search. Some research has modelled the dependency between named entities and other terms in the KB document to weight the different terms, [Gottipati and Jiang, 2011, Han and Sun, 2011, Petkova and Croft, 2007].

Named entity based search has improved retrieval performance in different tasks like cross-language retrieval [Mandl and Womser-Hacker, 2005] and event detection [Kumaran and Allan, 2004]. Such an approach has not been investigated for named entity disambiguation. As in a formulation of the NED problem as an IR problem, it is natural to explore a vector space model approach as a baseline. We consider several query formulation strategies. However, analysis of the poor results using this approach leads to the observation that it is the other NEs co-occurring with the query mention that are most helpful in disambiguating the query mention. Building on this insight, we have developed a novel similarity function to search the KB documents based on the statistical co-occurrence between NEs.

In this chapter, we present a tailored technique to retrieve the most probable knowledge base entries referring to a named entity mentioned in a document. We hypothesize that the similarity

between the different mentions for different named entities in the query document and the KB candidate documents (which are all KB documents that contain the query mention) is a good indicator for the correct KB entry. Also, such a similarity score could be used to learn a model for re-ranking the KB candidates. We present a document similarity measure (NEB-Sim) that uses the different mentions for different named entities in both documents (query document and KB candidate document) and show how it can contribute in named entity disambiguation. Different re-ranking techniques are tested to compare the performance of the NEB-Sim measure against the normal cosine similarity measure.

The proposed approach is evaluated on the TAC-dataset (see section 2.5.1). Comparison with a vector space model baseline approach based on cosine similarity with TF-IDF weighting shows our NE based search can indeed improve performance significantly. The effect of using only different mentions for different named entities in the document with the cosine similarity and NEB-Sim similarity function are also studied. The experiments show that using different mentions for named entities with cosine similarity does not significantly improve the performance, while using the NEB-Sim similarity function improves the performance significantly.

3.2 Named Entity Based Search

Our hypothesis is that textual mentions for different named entities in a document are the most valuable resource for disambiguating any of the named entity mentions. So, NE mentions extracted from the query document (other than the query mention) appear more useful than other query document terms. The query mention is defined as the ambiguous named entity mention to be disambiguated, while document terms refers to all terms that are not a part of any named entity, like nouns, verbs, adverbs,etc.

The following example is taken from the dataset “eng-NG-31-143446-10242486.sgm”, with some omissions for the sake of brevity (indicated by ellipses.) The example illustrates the difficulty of disambiguating the query mention “Barak”. It is very ambiguous, and could be any of “Barack Obama”, “Ehud Barak”, “Barak Moshe”, or “Barak Valley”.

```
<DOC>
<DOCID> eng-NG-31-143446-10242486 </DOCID>
<DOCTYPE SOURCE="usenet"> USENET TEXT </DOCTYPE>
<DATETIME> 2008-04-06T06:27:59 </DATETIME>
<BODY>
<HEADLINE>
swi news: Saturday, April 05, 2008, 29 Adar Bet, 5768
</HEADLINE>
<TEXT>
<POST>
<POSTER> Heidi <lilo97...@yahoo.com> </POSTER>
<POSTDATE> 2008-04-06T06:27:59 </POSTDATE>

Barak tries to calm Syrian nerves over Israeli drill 'Israel has no
intentions of launching any such operation," says defense minister in
public bid to allay Damascus concerns  scheduled nationwide exercise
foretelling of aggressive Israeli intent

Roni Sofer Published:  04.05.08, 23:33 / Israel News

"The home front drill commencing tomorrow is an exercise that has been
in the works for several months. Israel has no intentions to launch
any such operation or any others. Messages of reassurance have been
communicated across to all relevant parties," so reiterated officials
in the Prime Minister's Office on Saturday night after several days of
exacerbated tensions with Damascus.
.
....
.
Barak's deputy, Matan Vilnai, will brief the government on Sunday on
the course of the exercise. All State offices are also expected to
take part in the drill.

Hanan Greenberg contributed to this report
.....
</POST>
</TEXT>
</BODY>
</DOC>
```

It is clear from the example that most of the other non-NE words in the query document are unlikely to help in identifying the correct "Barak"; however, "Matan Vilnai" is very useful indeed. Thus, the joint relation between different mentions of named entities appears to be a promising factor for NE mention disambiguation.

NE based document retrieval is based on the assumption that NEs co-occurring with a specific NE in the same context will help in ranking the documents that contain information about this NE.

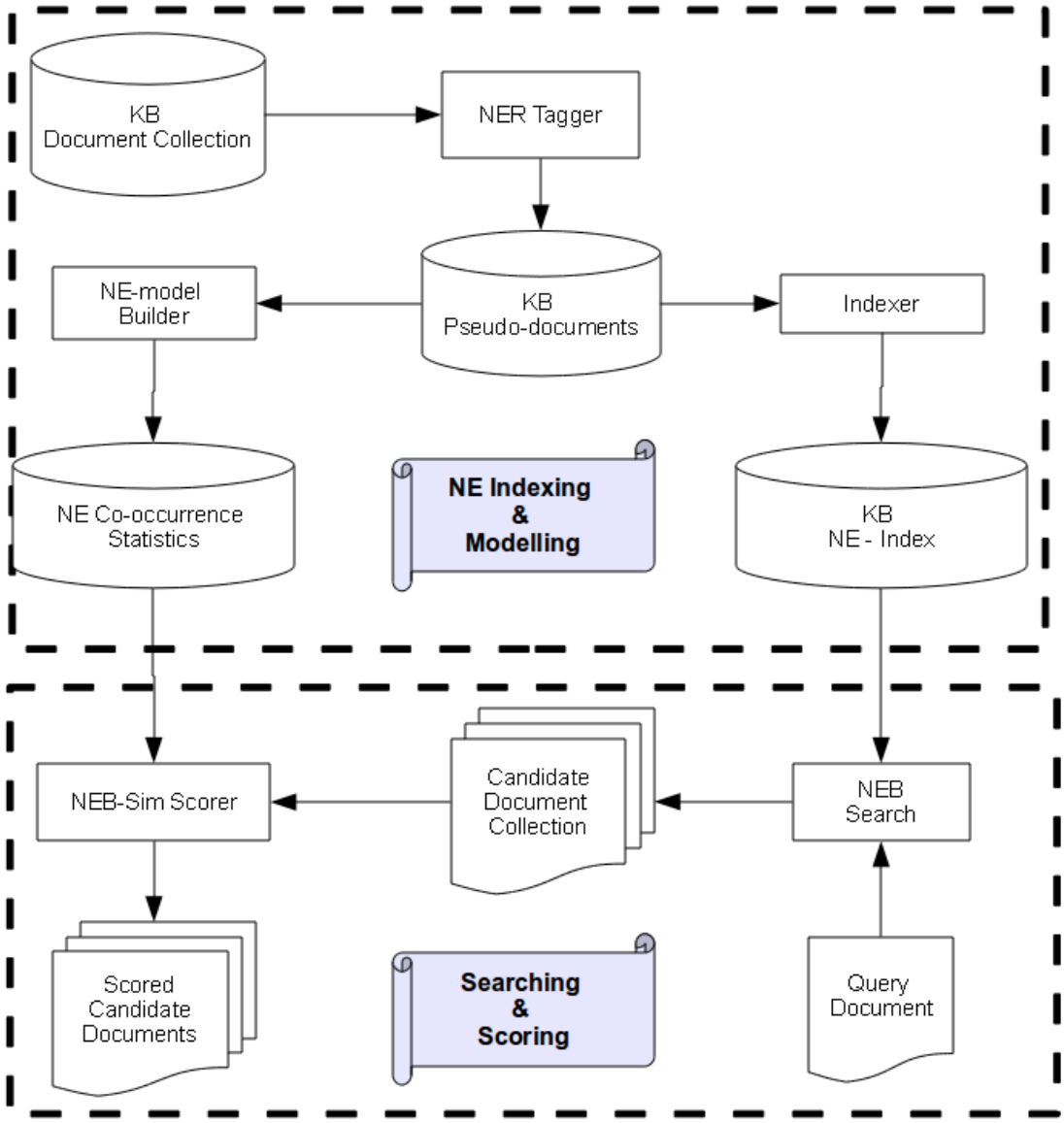


Figure 3.1: General Architecture of Named Entity Based Search

The architecture diagram of our proposed system is illustrated in figure 3.1. As shown in the architecture digram, the proposed system has two phases. The first is an offline phase which indexes all KB documents (see section 3.2.1). Additionally, the statistical co-occurrence relation between all NE mentions recognized in the KB documents is explored and a co-occurrence model is built (see section 3.2.2). The online phase is second, where the NE index is used to find candidate documents for the query mention in a given query document, and the NE

co-occurrence model is used to score each candidate document with the NEB-Sim similarity function (see section 3.2.3).

3.2.1 Document Collection Indexing

Referring to our hypothesis, not all document terms are useful. Clearly NE mentions are useful and we will see how far we can get with just them. So, just the named entity mentions are used to index. In this first phase, all KB documents are converted into a KB pseudo-document representation. A KB pseudo-document is a knowledge base document represented in terms of the different mentions for named entities only.

To convert a KB document into a KB pseudo-document, the Stanford NER tagger is used to extract named entity mentions in the knowledge base document text. The Stanford NER tagger implements a linear chain Conditional Random Field (CRF) sequence model [Lafferty et al., 2001]. It also incorporates some non-local structure models into the local model (which is the trained CRF) and uses Gibbs sampling to find the correct state sequence [Finkel et al., 2005]. We used the three class model (Location, Person, Organization) trained on the CoNLL2003, MUC-6, and MUC-7 datasets.

The KB pseudo-documents have to be indexed in order to speed-up the search process. KB pseudo-documents are represented in a vector space model using the Appache Lucene Indexer.

3.2.2 Modelling KB Named Entities

In the second phase, the named entity model θ is built given a set of KB pseudo-documents D and a set E of recognized named entities mentioned in the documents in D where $\forall\, d \in D \,\exists\, E$ where $E = \{e_1, e_2, e_3, ..., e_n\}$. The conditional probability between any two distinct named entity mentions $e_i, e_j \in E$ is estimated using the following formulae

$$p(e_i) = \frac{\sum_{d \in D} in(d, e_i))}{\|D\|} \tag{3.1}$$

$$p(e_i, e_j) = \frac{\sum_{d \in D} in(d, e_i \wedge e_j)}{\|D\|} \tag{3.2}$$

$$p(e_i|e_j) = \frac{p(e_i, e_j)}{p(e_j)} \tag{3.3}$$

where the function $in(d, e)$ returns 1 if the NE mention e occurs in d and 0 otherwise and $in(d, e_i \wedge e_j)$ returns 1 if $in(d, e_i) = in(d, e_j) = 1$, 0 otherwise.

3.2.3 Searching and Scoring

In this phase, all knowledge base pseudo-documents are searched for the query mention e_m given the query document. The query document is converted into a standard peseudo-document which contains all named entities extracted by the NER tagger. All KB peseudo-documents that contain the query mention named entity e_m will be retrieved as candidate documents that describe the query named entity mention e_m. As a rule of thumb, the document that describes a named entity must contain a mention of the named entity while not all documents mentioning a named entity describe it. This candidate set is huge and highly ambiguous. For each document, a numerical score is assigned to the candidate document using NEB-Sim. The basic concept is to use the relative information gained from the NE mentions in the query document and the document collection. Two similarity functions are proposed and their performance compared.

The first similarity function is based on the information theoretic definition of similarity proposed by [Lin, 1998]:

$$IT\text{-}Sim(A, B) = \frac{I(Common(A, B))}{I(Description(A, B))} \tag{3.4}$$

where $I(Common(A, B))$ is the information content of the statement describing what A and B have in common and $I(Description(A, B))$ is the information content of the statement describing what A and B are. In the proposed model, the elementary units of a document are taken to be the NE mentions recognized in it.

We define the NEB-Sim document similarity function as follows:

- Let A be the query document containing a set of NE mentions E_a;
- let B be a KB document containing a set of NE mentions E_b;

- let e_m be the query NE mention;
- let E_{ab} be the set of NEs common to A and B, i.e. $E_{ab} = E_a \bigcap E_b$;

$$NEB\text{-}Sim1(A,B) = \frac{\sum_{e \in E_{ab}} p(e \mid e_m)}{\sum_{e \in E_a} p(e \mid e_m) + \sum_{e \in E_b} p(e \mid e_m)} \tag{3.5}$$

However, this similarity function has a problem since it is affected by the relative weight of the candidate document NEs which are not in common with the query document, so the denominator will change for each candidate. As a kind of normalization, the relative weight of the KB document is removed in the second similarity function, based on the assumption that the weight of all related named entities found in the KB document ($Description(B)$) is not important in scoring the candidate. Following this assumption, we obtain the following formula:

$$NEB\text{-}Sim2(A,B) = \frac{\sum_{e \in E_{ab}} p(e \mid e_m)}{\sum_{e \in E_a} p(e \mid e_m)} \tag{3.6}$$

There is an analogy between the similarity function NEB-Sim1 and the Jaccard similarity coefficient. The Jaccard coefficient is defined as the size of the intersection divided by the size of the union of the sample sets A and B, as shown in equation 3.7. Both functions consider the proportion of shared items in both documents relative to the total items found in both documents. However, our NEB-Sim1 function does not consider the number of shared items in both documents, like Jaccard similarity, and calculates the sum of conditional probabilities of different named entities in the document, given the query NE mention.

$$J(A,B) = \frac{A \bigcap B}{A \bigcup B} \tag{3.7}$$

3.3 Learning to Rank Documents

As discussed in the literature review, learning-to-rank is a popular topic in document retrieval [Liu, 2011]. For the task of named entity disambiguation, re-ranking is one of the supervised approaches to get the correct NE reference document in the KB at the top of the candidate list. Learn-to-rank approaches are categorized into three classes, point-wise, pair-wise, and list-wise

(see section 2.3.5). The following learn-to-rank approaches are tested to re-rank the candidate documents and their results are compared.

1. Point-wise approach: This is the simplest learn-to-rank approach. A Naïve Bayes classifier is used to classify each instance into relevant or non-relevant. We used the point-wise approach to discard some of the non-relevant candidate documents while the rank score is still the same.

2. Pair-wise approach: This approach focuses on the relative order between two instances. It is a classification on instance pairs where the objective function is to minimize the number of misclassified pairs. SVM^{rank} [Joachims, 2002, 2006] is used to build a pair-wise ranking model and re-rank the candidate documents.

3. List-wise approach: The SVM^{map} algorithm is used to learn a model to optimize the value of the evaluation measure, averaged over all queries in the training data [Yue et al., 2007]. So, it is used to learn the ranking, using Mean Average Precision (MAP) to calculate the list-wise loss function.

3.4 Experimental Results and Discussion

The TAC-KBP 2011 data set is used to carry out different experiments. The dataset contains 2231 query documents containing 2250 query mentions (for more details see section 2.5.1). As this research focuses on the problem of disambiguation, we used only the set of queries which have an entry in the Wikipedia KB. Exactly 1124 out of the 2250 query mentions have an entry in the Wikipedia KB snapshot used in TAC 2011. For each query mention, the highest 100 scored documents are retrieved and the performance checked at different ranks, i.e. at rank 1, 5, 10, 30, 50, and 100. Accuracy is used as the performance measure. We explored three different query formulation schemes described as follows:

QM: Query mention terms alone are used.

QS: The sentences containing the query mention in the query document are used as a query, in

addition to the query mention which is the mandatory part in the query. The result documents must therefore contain the query mention and may contain some sentence tokens.

QD: This scheme extends the QS scheme and includes all document terms instead of the query sentence. The full query document is used in addition to the query mention, i.e. all query document terms are used in the query as optional terms, while the query mention tokens are mandatory.

QNES: This scheme uses the collection of different named entity mentions in the query document in addition to the query mention as a query.

We also considered three different search scopes, defined as follows:

Title: the search scope is the Wikipedia page title.

Text: the search scope is the Wikipedia page document.

NES: the search scope is the pseudo-document, i.e. the KB document which consists of only NEs.

As a baseline, cosine similarity with TF-IDF weighting is used to retrieve candidate entries from the Wikipedia KB using the different query formulation schemes and different search scopes with results as shown in section 3.4.1. Then, the results using named entities and a comparison between using named entities in search using cosine similarity and using our similarity function are shown in section 3.4.2. Re-ranking using different learn-to-rank approaches results are shown in section 3.4.3. Finally, we present a discussion of the results to show how NEs are used to improve the accuracy and the effect of using re-ranking in section 3.4.4.

3.4.1 Baseline Results

Cosine similarity with TF-IDF weighting is used with the different query formulation schemes as a baseline approach to search the Wikipedia knowledge base document. Apache Lucene[2] is used to index all Wikipedia KB documents and titles (which is referred as "Title" and "Text" scope

[2] `http://lucene.apache.org/java/docs/` (last visited 30-Jun-2014)

in our experiments). Also, Lucene is used to search the scope and score the results by cosine similarity using vector space model and TF-IDF term weighting scheme. Table 3.1 summarises the results of searching the Wikipedia KB using QM,QS, and QD query schemes. It is not meaningful to use the QS and QD query schemes with the title scope. So, we used only the QM scheme with the title scope.

Result Graphs: We used a graphical representation for all results. The x-axis represents the top-N scored candidate and y-axis represents the accuracy at recall level specified by top-N. Finally, legend is coded as <Query Scheme – Search Scope>, eg. QD-Text means using query document ("QD") query scheme and "Text" search scope.

Figure 3.2 is a graphical presentation to the results shown in table 3.1. The figure shows the accuracy of the baseline approach at different ranks. These results shows using the QD query scheme to search within the KB text is better than other query schemes i.e. QS, and QM. Also using query mention text (QM) as a search query to search in KB titles is better than using the same scheme to search within the KB text considering small number of candidates, i.e. less than 25 candidate, while considering large number of candidates the QM scheme performs better with text scope rather than title scope. From figure 3.2 we can conclude limiting the scope to title reduces the recall. Also, as the query becomes larger, adding more information, the recall is increased which is expected with the vector space model.

3.4.2 NEB-Sim Results

To explore the effect of using document similarity measures based on NE mentions, two sets of experiments were carried out. In the first set of experiments, we used the same query schemes that were used in the baseline experiments (QM, QS, and QD) to search the KB pseudo-

Query Scheme	Scope	A@1	A@5	A@10	A@20	A@30	A@50	A@100
QM	Title	0.23	0.37	0.48	0.57	0.60	0.63	0.65
QM	Text	0.10	0.20	0.31	0.44	0.53	0.62	0.72
QS	Text	0.15	0.30	0.42	0.54	0.62	0.71	0.78
QD	Text	0.22	0.45	0.56	0.64	0.69	0.74	0.80

Table 3.1: Baseline Approach Results

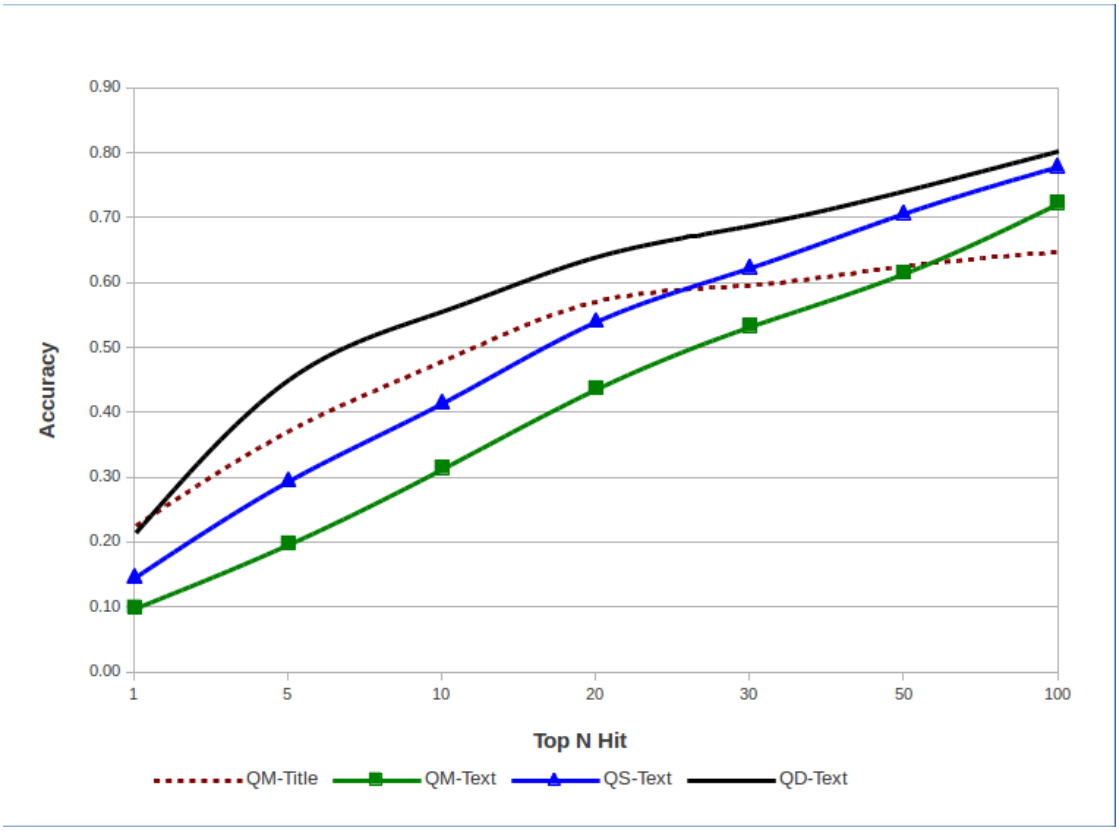


Figure 3.2: Baseline Approach Results

documents, i.e. NES scope. Similarity is measured with cosine similarity. The results of this set of experiments are shown in table 3.2.

Query Scheme	Scope	A@1	A@5	A@10	A@20	A@30	A@50	A@100
QM	NES	0.09	0.23	0.34	0.46	0.55	0.63	0.73
QS	NES	0.11	0.27	0.38	0.51	0.59	0.67	0.75
QD	NES	0.27	0.54	0.63	0.71	0.75	0.80	0.84
QNES	NES	0.27	0.56	0.64	0.72	0.77	0.81	0.85

Table 3.2: Results using Named Entities with Cosine Similarity

To make it easier to analyse these results, they are also presented graphically in figure 3.3. As shown in the figure, using the QM scheme achieves the lowest performance because it just uses one named entity mention and does not get the benefits of the occurrence of other named entities. So, as the number of named entity mentions increases in the query i.e. QS and QD, the accuracy gets better. Using QNES to search the KB text performs better than any searching over the NEs.

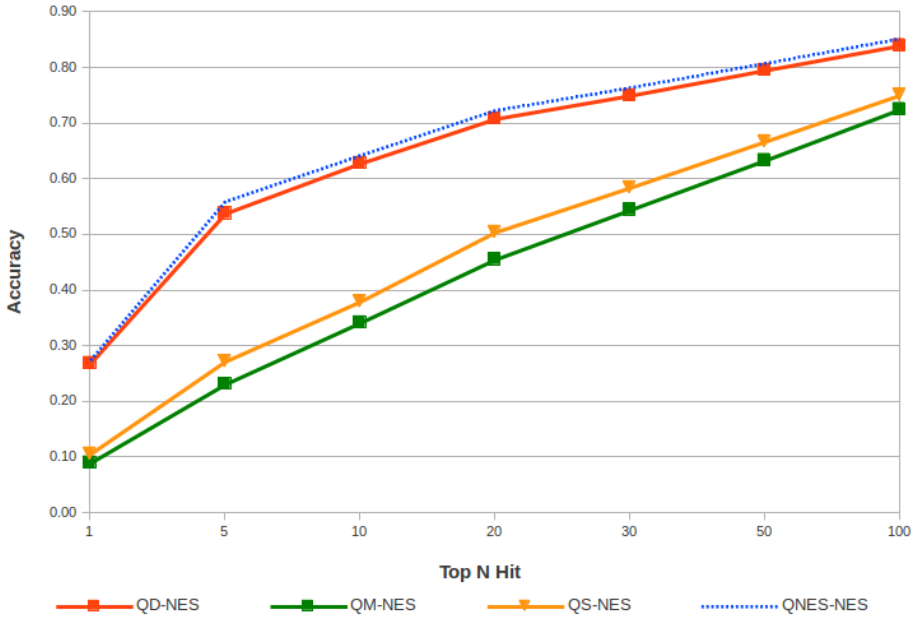


Figure 3.3: Results using Named Entities with Cosine Similarity

In the second set of experiments the NEB-Sim similarity function is used to score the dif-

ferent candidates. The scope in this experiments is always the KB pseudo-documents, i.e. NES, and the query scheme is QNES. The results of this set of experiments are shown in table 3.3. It is clear from the table that NEB-Sim2 does better than the NEB-Sim1.

Approach	A@1	A@5	A@10	A@20	A@30	A@50	A@100
NEB-Sim1	0.18	0.32	0.36	0.39	0.41	0.44	0.46
NEB-Sim2	0.28	0.57	0.68	0.77	0.81	0.84	0.90

Table 3.3: Results using NEB-Sim scoring functions

To conclude the investigation of using NEs on the query side and NEs in the search scope i.e. pseudo-document, we combined the results in one figure. Figure 3.4 shows the difference between matching NEs in the query and KB document using the cosine similarity function and when using the NEB-Sim named entity based document similarity measures. The figure shows NEB-Sim2 outperforms cosine similarity scoring. Also, comparing the results of QM-in-NES to other results, we see how using other named entity mentions in the query document is useful in creating a good candidate list with high recall.

3.4.3 Re-Ranking Results

Different experiments have been carried out for re-ranking. As described in section 3.3, three learn-to-rank approaches are tested to re-rank the candidate list for the query NE textual mention. Tables 3.4, 3.5, and 3.6 list the results of using the different techniques. The query scheme column shows which query scheme is used to retrieve the candidate list while the similarity column shows which similarity function is used as a feature for re-ranking. We tested re-ranking using only the cosine similarity function with the different query schemes. Then, we used the cosine similarity in conjunction with the NEB-Sim similarity function for re-ranking the candidates. In some cases the NEB-Sim similarity score is the same for all candidates because there is only one NE mention annotated in the document, so there are no other NE mentions to be used in NEB-Sim similarity. In such cases, re-ranking will not affect this result. Hence, we used the cosine similarity in addition to the NEB-Sim for re-ranking. For all experiments, accuracy is measured at recall levels 1, 5, 10, 20, 50, and 100. In this set of experiments, the whole candidate list are considered for re-ranking. That explains why around 100% accuracy is obtained at some

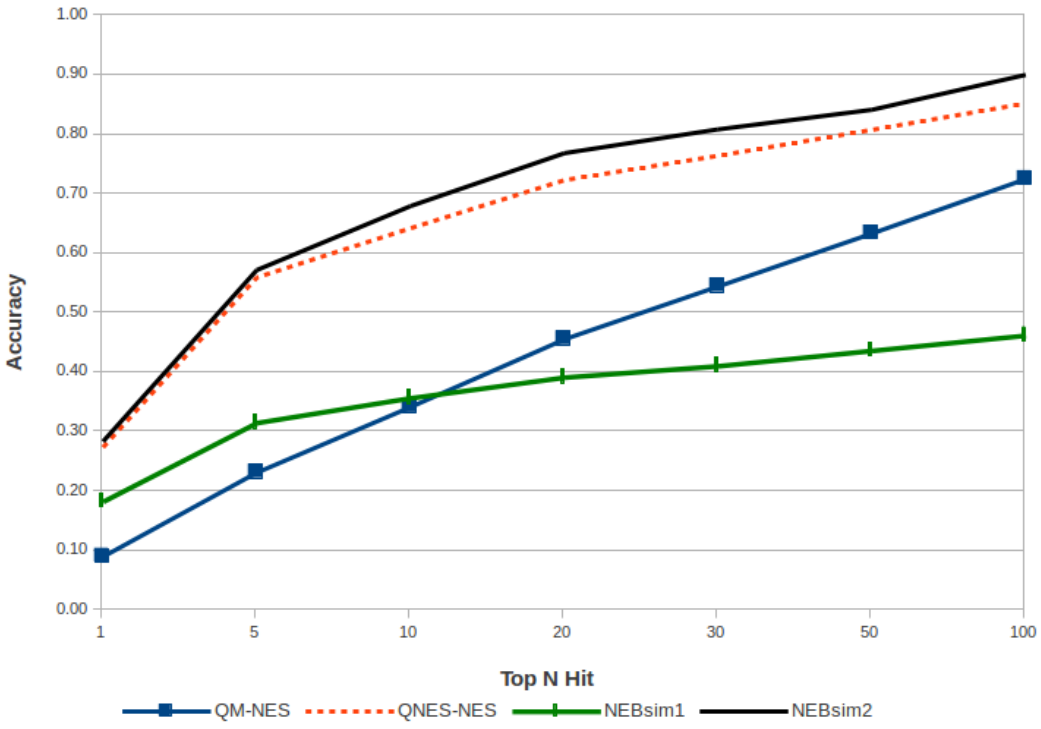


Figure 3.4: Results using NEB-Sim scoring functions

recall levels while the maximum achieved accuracy without re-ranking is not more than 90%.

Query scheme	Similarity	A@1	A@5	A@10	A@20	A@30	A@50	A@100
QM		0.181	0.28	0.38	0.535	0.628	0.714	0.843
QS	Cosine	0.254	0.361	0.465	0.63	0.706	0.797	0.881
QD		0.367	0.572	0.678	0.752	0.809	0.869	0.936
QM		0.587	0.817	0.907	0.951	0.969	0.975	0.99
QS	Cosine,NEB-Sim	0.589	0.81	0.904	0.947	0.97	0.975	0.99
QD		0.654	0.858	0.941	0.962	0.977	0.977	0.985

Table 3.4: Point-wiseRe-Ranking using Naïve Bayes Classifier

Query scheme	Similarity	A@1	A@5	A@10	A@20	A@30	A@50	A@100
QM		0.137	0.207	0.274	0.344	0.38	0.429	0.525
QS	Cosine	0.171	0.224	0.28	0.348	0.376	0.422	0.523
QD		0.189	0.256	0.29	0.342	0.373	0.398	0.528
QM		0.783	0.938	0.962	0.988	0.995	0.998	0.998
QS	Cosine,NEB-Sim	0.79	0.935	0.968	0.983	0.99	0.998	0.998
QD		0.752	0.92	0.972	0.985	0.99	0.995	1

Table 3.5: Pair-wise Re-Ranking using SVM^{rank}

Query scheme	Similarity	A@1	A@5	A@10	A@20	A@30	A@50	A@100
QM		0.168	0.272	0.308	0.363	0.396	0.446	0.563
QS	Cosine	0.165	0.269	0.305	0.363	0.395	0.444	0.559
QD		0.155	0.253	0.293	0.349	0.386	0.431	0.563
QM		0.535	0.765	0.869	0.944	0.957	0.969	0.982
QS	Cosine,NEB-Sim	0.544	0.759	0.861	0.94	0.952	0.968	0.983
QD		0.623	0.812	0.918	0.967	0.975	0.98	0.985

Table 3.6: List-wise Re-Ranking using SVM^{map}

3.4.4 Discussion

In this section, we present some notes and comments about the results shown in previous sections. Initially, in the baseline approach we can notice that the QD scheme always does better than other schemes when searching in the Text scope (table 3.1). This result supports our hypnothesis because QD uses the whole query document, and the most effective parts are the NE mentions in the document. Furthermore, we can note the same conclusion for the QNES scheme, which is used to search NES scope, even with cosine similarity (table 3.2). Comparing tables

3.1 and 3.2, we note that when searching the NES instead of Text scope, and use QNES instead of QD scheme, improves the results a little, which indicates using other terms that are not part of NE mentions is misleading. The results shown in table 3.2 shows the performance of QD and QNES schemes are very close when we used NES scope. This result makes sense, as the NES search scope restricts the similarity to the named entity mentions in the query document. However, the difference in the results is because of the different size of the query when using QD and QNES, which is a factor in cosine similarity calculations. Our NEB-Sim scoring function does better than using the cosine similarity to score the search result documents based on NEs.

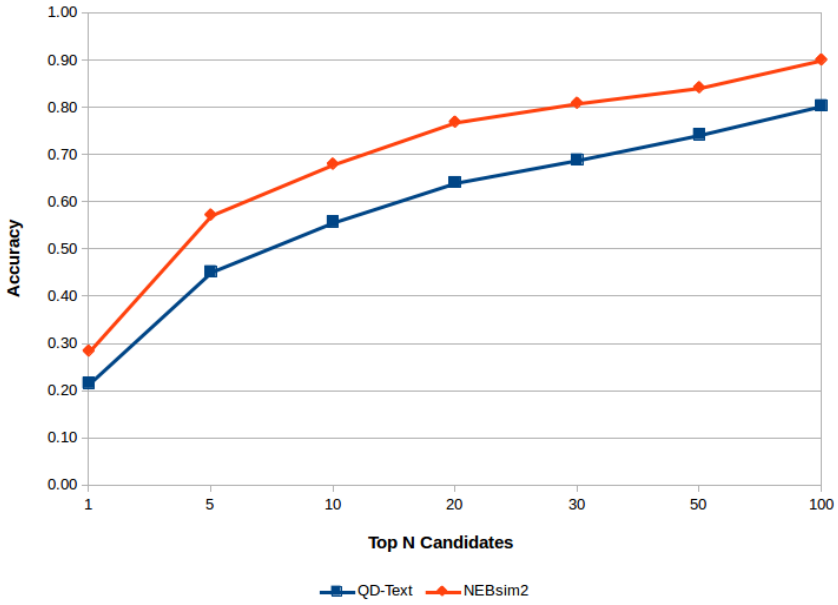


Figure 3.5: Comparison between BaseLine and Named Entity Based Search

Figure 3.5 shows a comparison between the baseline approach (QD-Text) and named entity based search approach (NEB-Sim2). The first is set up with the QD scheme and Text search scope, and cosine similarity is used to score the candidates. The second approach is set up with the QNES scheme and NES search scope, using our similarity function NEB-Sim2. NEB-Sim2 achieves a significant improvement over the best query scheme against full documents (QD), where $p < 0.05$. More analysis has been done to compare NEB-Sim1 and NEB-Sim2, and we conclude that ignoring the relative weight of the candidate document in similarity function NEB-

Method	A@1	A@5	A@10	A@20	A@30	A@50	A@100
Query scheme: QM							
SVM^{rank}	0.783	0.938	0.961	0.987	0.995	0.997	0.997
Naïve Bayes	0.586	0.816	0.907	0.951	0.969	0.974	0.990
SVM^{map}	0.535	0.765	0.868	0.943	0.956	0.969	0.982
Query scheme: QS							
SVM^{rank}	0.789	0.934	0.967	0.982	0.990	0.997	0.997
Naïve Bayes	0.589	0.810	0.903	0.947	0.969	0.975	0.990
SVM^{map}	0.543	0.759	0.860	0.939	0.952	96.70	98.22
Query scheme: QD							
SVM^{rank}	0.752	0.919	0.971	0.985	0.989	0.995	1.000
Naïve Bayes	0.654	0.858	0.940	0.961	0.977	0.977	0.984
SVM^{map}	0.623	0.811	0.917	0.966	0.974	0.979	0.984

Table 3.7: The accuracy after re-ranking

Sim2 improves performance significantly over NEB-Sim1, where $p < 0.0001$. In our analysis, we used unpaired (independent) two-sample student t-tests with two tails and unequal variances.

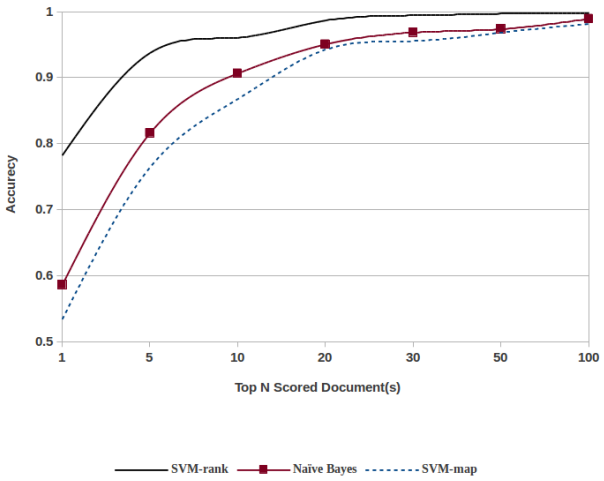
To study the effect of using different learn-to-rank approaches, one algorithm is used for each of these approaches and tested using the different query schemes (see section 3.4.3). The best results were achieved when cosine similarity scores and NEB-Sim score are used as features with the different query schemes. Table 3.7 summarizes the results of using different learn-to-rank approaches with different features generated using different query schemes. The same results were presented in section 3.4.3 in a different format. To study the contribution of each learn-to-rank approach, the results are grouped by the query scheme used to generate the cosine similarity score.

For visual analysis, the results are shown in figures 3.6a, 3.6b, and 3.6c. The first observation from these graphs is that SVM^{rank}, i.e. pair-wise re-ranking, outperforms both the Naïve Bayes and SVM^{map} approaches, and Naïve Bayes outperforms SVM^{map} for all query schemes. Re-ranking improves the short listing of the NE candidate list, as we can observe the accuracy dramatically increased at recall 1, 5, and 10 candidates. Then, using the top scoring candidate for disambiguation, we achieved 79% accuracy. Also, we can claim that the correct candidate is in the top 20 candidates. Comparing these results to either the base line or NEB-Sim approach results, we can see how re-ranking improves the accuracy at all recall points.

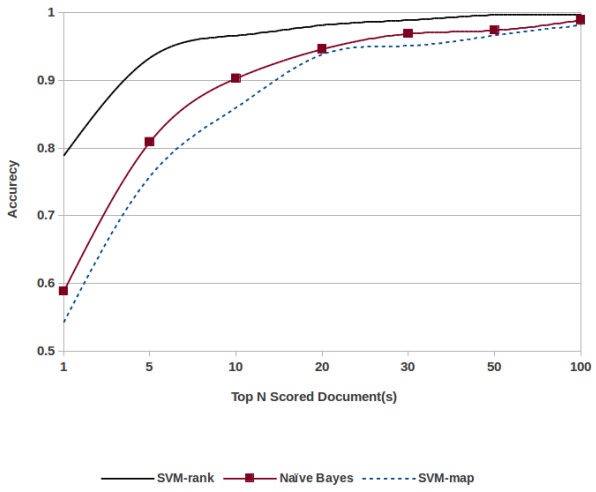
3.5 Conclusions

In this chapter we claimed that different mentions for different named entities occurring with the query mention in the same context are helpful for improving the search results in the candidate generation phase of an NED system. We presented a document similarity function based on NEs in both the query document and the KB document. The results show the correctness of the conjecture that the NEs co-occurring with a specific NE can disambiguate it in a useful way, or at least help shorten the list of NE candidates. One of our similarity functions, NEB-Sim2, achieves a significant improvement over the cosine similarity measure. Also, we proposed using our NEB-Sim similarity with the cosine similarity score to re-rank NE candidates. We explored different Learn-to-rank approaches and conclude this is a reliable method for re-ranking NE candidates, especially the pair-wise approach.

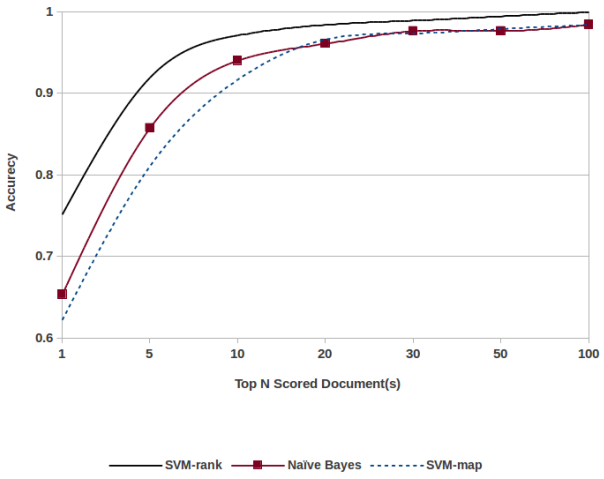
The approach presented in this chapter uses different named entity mentions in the same query document to disambiguate a specific textual mention. The other NE mentions in the document are still ambiguous and that is the main drawback of the individual disambiguation approaches. In the following chapters, two different collective disambiguation approaches are presented.



(a) Learning To Rank Results using Query Scheme: QM



(b) Learning To Rank Results using Query Scheme: QS



(c) Learning To Rank Results using Query Scheme: QD

Chapter 4

Disambiguating Named Entities using HMMs*

In this chapter we present a novel approach to disambiguate textual mentions of named entities against the Wikipedia knowledge base. The conditional dependencies between different named entities across Wikipedia are represented as a Markov network. In our approach, named entities are treated as hidden variables and textual mentions as observations. The number of states and observations is huge and naïvely using the Viterbi algorithm to find the hidden state sequence that emits the query observation sequence is computationally infeasible, given a state space of this size. Based on an observation that is specific to the disambiguation problem, we propose an approach that uses a tailored approximation to reduce the size of the state space, making the Viterbi algorithm feasible. Results show good improvement in disambiguation accuracy relative to the baseline approach and to some state-of-the-art approaches. Also, our approach shows how, with suitable approximations, HMMs can be used in such large-scale state space problems.

The rest of this chapter is organized as follows: section 4.1 presents a broad view of HMM and the problems of using such technique in disambiguation problems. The proposed NED framework and a detailed description for the HMM modelling and the decoding is discussed in section 4.2. A description of the evaluation dataset is given in section 4.3. Experimental setup

*Part of this work has been published in *In Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, vol. 3, pp. 159-162. IEEE, 2013.

and experimental results with comparison against the baseline approach and some state-of-the-art approaches are presented in section 4.4. Finally, some conclusions and a discussion of how HMMs can be used in such problems are presented in section 4.6.

4.1 Introduction

A Hidden Markov Model (HMM) is a tool to model the probability distribution of a state sequence that generates a sequence of observations [Ghahramani, 2001]. So, it is a simple dynamic Bayesian network. The main difference between simple Markov models and Hidden Markov models is that in the former the state is directly visible to the observer and the parameters are the transition probabilities while in a hidden Markov model the state is not visible but the output is visible. For each state, there is a probability distribution over all possible outputs. So, the sequence of observations gives some indications about the sequence of states. The hidden Markov model has the following assumptions:

1. The state of the hidden process satisfies the Markov property. This is a memoryless property of a stochastic process where the probability distribution of the future states of the process is based on the current state regardless the previous states. So, the value of state S_t at time t is dependent only on a finite number of immediately proceeding states, i.e. $S_{t-n}, \ldots, S_{t-1}$. When the current state is dependent only on the previous state, the model is known as a first order Markov model. In an n order Markov model, the current states value depends on n previous states.

2. The observation generated at time t is generated by a process whose state S_t is unknown or hidden.

3. The hidden state variable has a discrete value.

The joint probability distribution of a sequence of states $S_{1:T}$ and observations $O_{1:T}$ where T is the sequence length is given as shown in equation 4.1.

$$P(S_{1:T}, O_{1:T}) = P(S_1)P(O_1|S_1)\prod_{t=2}^{T} P(S_t|S_{t-1})P(O_t|S_t) \tag{4.1}$$

To define the probability distribution of a sequence of observations, we need to find the probability of the initial state $P(S_1)$, the transition probability between different states $P(S_t|S_{t-1})$ which is represented in a $K \times K$ matrix where K is the number of states, and the observation emission probability $P(O_t|S_t)$ which is represented in a $K \times L$ matrix where L is the number of observations.

The approach we present in this chapter is a collective approach, where all NE textual mentions in a context are jointly disambiguated. We assume that when disambiguating a NE mention, the other NE mentions found in the same context will be a good source of information.

We treat KB entries in Wikipedia as surrogates for real world entities. The textual portion of these entries typically contains mentions of multiple other named entities. When these mentions are hyper-linked to other KB entries we can infer that there is some relation between the real world entities corresponding to the KB entries. These links allow us to build up statistical co-occurrence counts both between entities and between mentions and entities that occur in the same context (i.e. in the same document, though other we explore other interpretations of "context" too).

Thus, all the named entities in the knowledge base can be represented in a Markov Network, where the nodes represent entities and the edges represent conditional dependencies computed between "grounded" (i.e. hyper-linked to a Wikipedia entry) mentions of the two connected entities, where the later occurrence is assumed to be dependent on the former. A Markov network is similar to Bayesian network in its representation of dependencies but differs in that Bayesian networks are directed and acyclic [Koller and Friedman, 2009], while Markov Networks need not be. For any text document containing a set of NE mentions, each NE mention may be mapped to a set of named entities in the Markov network, i.e. those entities which are possible interpretations of the name.

We treat the task of NED as finding the best sequence of Wikipedia KB named entities given a set of different mentions for different named entities in the same context. The KB named entity entry (state) is not directly visible and we have just the NE mention (observation). Thus, we employ an HMM approach and use the Viterbi decoding algorithm to find the best NE entry (state) sequence that generates the mention (observation) sequence.

HMMs have been successfully used for various sequence labelling tasks in natural language processing, including part of speech tagging [Ekbal et al., 2007, Hasan et al., 2007, Van Gael et al., 2009], and NE recognition for different languages [Chopra et al., 2012]. they have also been successfully used in Information Extraction [Elliott et al., 1995]. But, to the best of our knowledge, such an approach has not been investigated for the NED task.

The huge number of states/entities ($\approx$ 1 million) would appear to make it infeasible to use an HMM for this problem. However, we observe that by taking advantage of particular characteristics of the NE disambiguation task (e.g. considering only the set union of disambiguation sets for all NE textual mentions in a specific context of interest) we can reduce the state space, making the HMM approach feasible without affecting results.

4.2 Framework

In this section, a detailed description of the NED problem formulation and approximation are presented. Our proposed framework consists of two main modules. The first one is an offline process to model all states and observations found in the Wikipedia KB. The second module is the disambiguation module which uses the generated model to decode an input observation sequence into the most probable sequence of states that emits such an observation sequence. Figure 4.1 shows the general architecture of our framework. We begin by clarifying what we mean by state and observation:

State: The Wikipedia knowledge base entry for a NE.

Observation: A NE textual mention or surface form that appears in a document.

4.2.1 Wikipedia HMM Modelling

We address only the most popular types of named entities: locations, organizations and persons. As a first step, the DBpedia ontology[2] is used to build up a list of named entities of types of

[2]We used version 3.8 which is available to download from `http://wiki.dbpedia.org/Downloads38` (Last visited 30-Jun-1014)

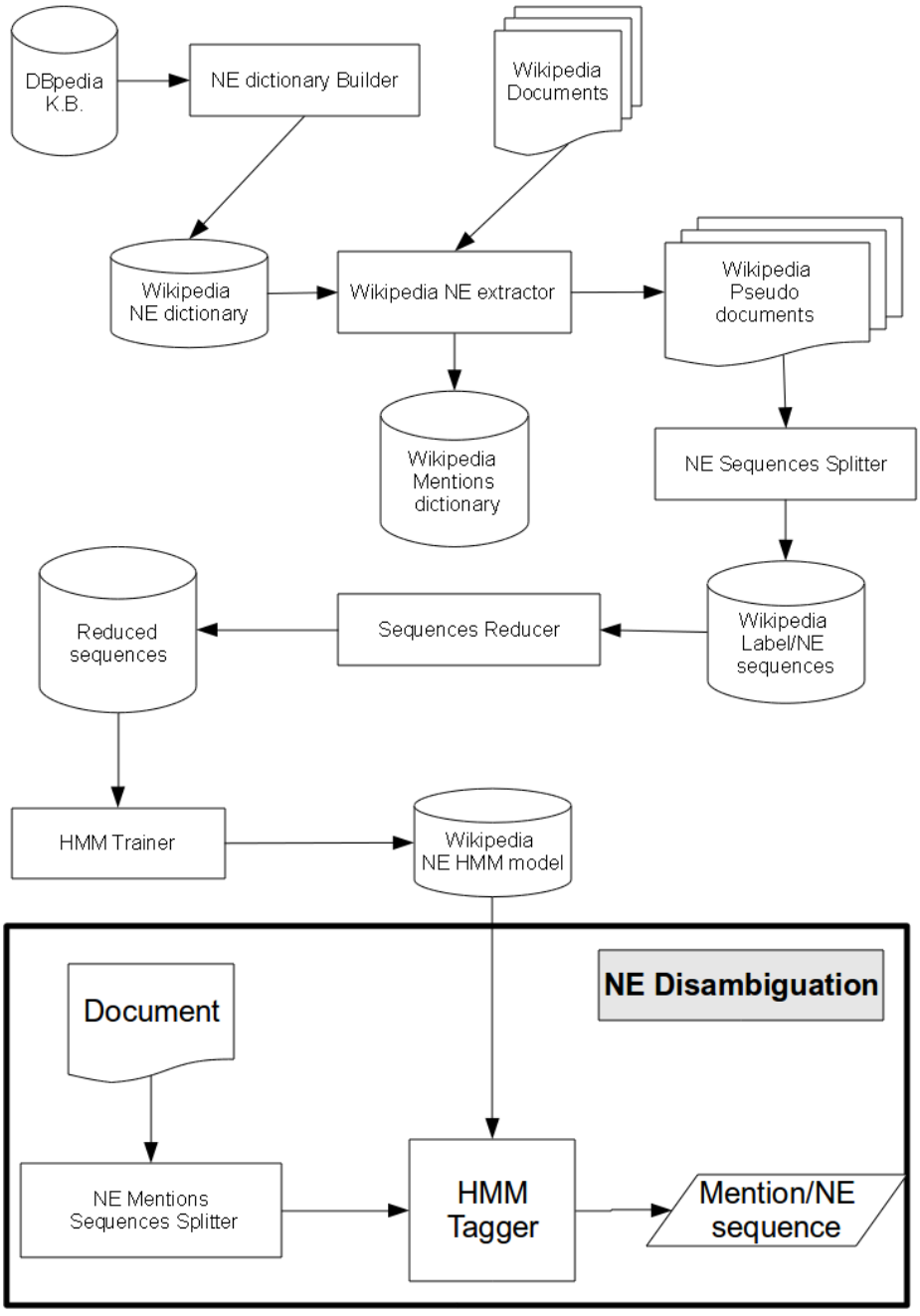


Figure 4.1: HMM-Based NED System Architecture

interest, resulting in a list of 1,244,682 unique NE entries in the KB that are taken to be the KB reference states.

sequence set	raw sequences	reduced sequences	States	Observations
sentence	17,700,551	5,492,638	475,569	592,503
paragraph	15,444,803	5,021,551	545,388	699,730
segment	6,801,829	3,587,421	786,772	1,080,807
document	3,541,181	2,468,442	996,836	1,414,560

Table 4.1: Properties of Extracted Wikipedia Sequences

Next, Wikipedia[3] is parsed to identify all sequences containing one or more NE mentions (anchor text) linked, via an inter-wiki link, to one of the reference states. This yields a list of 1,629,961 NE textual mentions extracted from the Wikipedia anchors that is used to form the observation dictionary. The dependencies between different named entities in the Wikipedia knowledge base and the emission probabilities for different textual mentions are modelled using the HMM trainer[4].

To model NE dependencies we explore four different assumptions about the textual scope, or "context" of NE dependencies, defining a segmentation scheme for each:

sent: NE context is a single sentence.

par: NE context is a single paragraph. Here the context is interpreted as a collection of sentences that have been aggregated into one paragraph and are likely on the same topic.

seg: NE context is a Wikipedia article subsection. Here the context consists of multiple paragraphs under the same subtitle, i.e. all paragraphs under a specific title in the page table of contents.

doc: NE context is the entire document. Here all named entities found in a document are considered as one sequence.

HMM models were trained using NE sequences extracted from the Wikipedia KB according to the different segmentation schemes. Thus, different models are trained with the same named entities but using different sequences.

[3]We use a Wikipedia dump that was taken in February 2012.

[4]We used the HMM trainer provided in the NLTK (version: nltk-2.0.1rc1). Also, we modified the Viterbi decoder provided in the same package

Because of the huge number of extracted sequences, training is slow[5] and the resulting models are very large. To alleviate these issues we reduced the number of sequences used in training by considering only sequences that have at least one of the test mentions or test named entities (of course, many named entities or states other than those used in test data will still appear in these sequences). Table 4.1 shows a summary of the number of sequences, reduced sequences, number of states and the number of observations across every context. Overall this reduced the number of sequences used in training by 30–70% and the number of states in the resulting models by 17–60%, depending on the sequence type. For each context scheme a model μ is generated, yielding four models: $\mu_{sent},\ \mu_{par},\ \mu_{seg}$, and μ_{doc}.

4.2.2 HMM disambiguation

The proposed formulation for NED takes the form of a hidden Markov model (HMM), where the states correspond to the Wikipedia named entities E and mentions M are stochastically emitted each time a state is visited. HMM NED training involves estimating the different mentions emission probabilities $p(m_i|e_j)$ and the transition probabilities between different states $p(e_i|e_{i-1})$ Given an observation sequence $M_{test} = m_1, \ldots, m_n$ where $n \geq 1$ the goal of NED is to find a stochastic optimal tag sequence $E_{test} = e_1, \ldots, e_n$ that maximizes the joint probability of a sequence of states and observations $Pr(e_{1:n}, m_{1:n})$, where $x_{1:n}$ abbreviates $x_1, \ldots, x_n$. According to the first order HMM assumption, this joint probability is given by:

$$Pr(e_{1:n}, m_{1:n}) \approx p(e_1)p(m_1|e_1)\prod_{i=2}^{n} p(e_i|e_{i-1})p(m_i|e_i) \tag{4.2}$$

The Viterbi algorithm is a well known algorithm used to find the most probable state sequence that emits an unlabelled observation sequence [Manning and Schütze, 1999]. We used the Viterbi algorithm to find the best NE sequence that emits the unlabelled sequence of mentions. However, using the Viterbi algorithm with all states can be computationally infeasible when working with a very large-scale state space. Suppose N is the number of states/named entities found in Wikipedia and M is the number of observations/textual mentions found in

[5]For the reduced datasets described below, training took about 5 days per model.

Wikipedia. $N \approx 10^6$ and $M \approx 1.5 \times 10^6$. Then, the size of transition matrix A that keeps the transition probability between all states is $N^2 \approx 10^{12}$ and the size of the emission probability matrix B that stores for each state the probability of that state emitting one of the observations is $N \times M \approx 1.5 \times 10^{12}$. Despite the sparseness of both matrices, all states must be visited to calculate the best path according to the Viterbi algorithm. The complexity of the Viterbi algorithm is $O(T \times N^2)$ where T is the sequence length. Thus the algorithm would appear to be intractable for our problem.

However, for the NED problem, where for each mention the correct entity is assumed to be a known entry in the KB, the set of candidate disambiguation states for each observation is known or can be constructed using simple techniques. Therefore it is a waste of time to try to find the best sequence from amongst *all* states when the correct one is partially known given the disambiguation sets. Thus, we make the following observation:

Observation: For any ambiguous observation M_i there is a set of candidate states $E_i = \{e_i^1, \ldots, e_i^j\}$, where j is the number of disambiguation candidates. Typically, $|E_i| \ll |E|$.

Assuming the correct NE falls into the NE candidate list of each mention/observation, a new state space of the disambiguation set of named entities for all mentions in the sequence can be used instead of using the original state space that includes all states. This approximation makes it computationally feasible to find the best sequence using the Viterbi algorithm. We propose three variant approximations for reducing the state space to decode a textual NE mention sequence into a sequence of Wikipedia knowledge base named entities. We have adapted the Viterbi algorithm to handle these approximations. To explain the following approximations, we use simple figures to show the state space representation for each approximation. The example used is for a NE textual mention sequence a, b, c and d. Their instances in their disambiguation candidate list are represented by squares, circles, triangles and hexagons respectively.

Approximation 1 Suppose we have a dependency model μ that models the dependency network of the states E. In this approximation, the state space is reduced to include only the set of states that emits any of the observations in the input sequence M, where $M = m_1, m_2, ...m_n$.

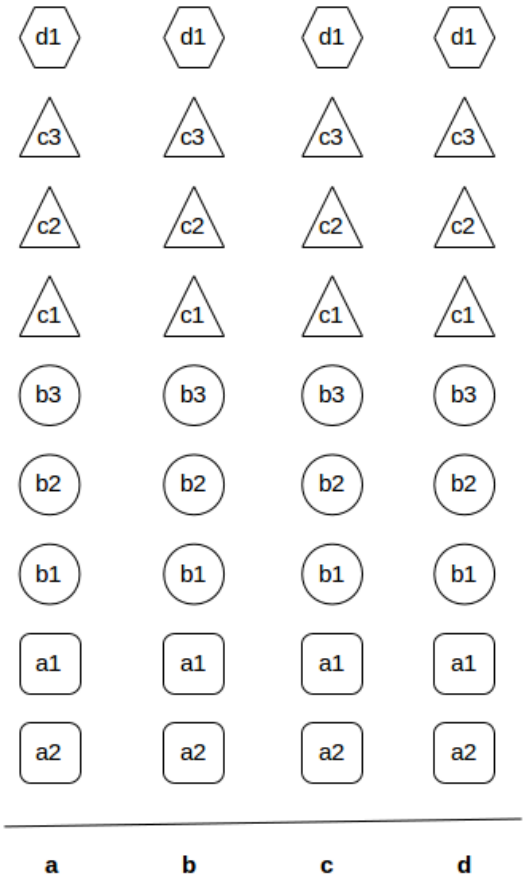


Figure 4.2: State space representation in Approximation 1

Hence a new state space E_M is defined where $E_M \subset E$ and $E_M = \{E_1 \bigcup E_2 \bigcup ... \bigcup E_n\}$ with $E_i = \{e_i^1, e_i^2, e_i^3, ..., e_i^k\}$, k being the number of NE candidates for a mention m_i and $1 \leq i \leq n$. Then, $E_M = \{e_M^1, e_M^2, e_M^3, ..., e_M^c\}$, where c is the total number of all candidates in all candidate lists. Figure 4.2 shows a simple diagram of the state space for an observation sequence a, b, c and d where a candidates are presented in rectangles, b candidates in circles and c candidates in triangles. Each column presents all possible states for all observations. The HMM decoding steps, as adapted from Manning and Schütze [1999], are as follows:

1. Initialization:

$$\delta(j) = p(E_M^j) \times p(m_1|e_M^j) \ , \ 1 \leq |E_M|$$

2. Recursion:

$$\delta_t(j) = max_{1 \leq i \leq c} [\delta_{t-1}(i) \times a_{ij}] \times p(m_t | e_M^j)$$

where $1 \leq j \leq c$; $2 \leq t \leq k$ *and* $c = |E_M|$

Note here that states considered are just those in E_M rather than in E.

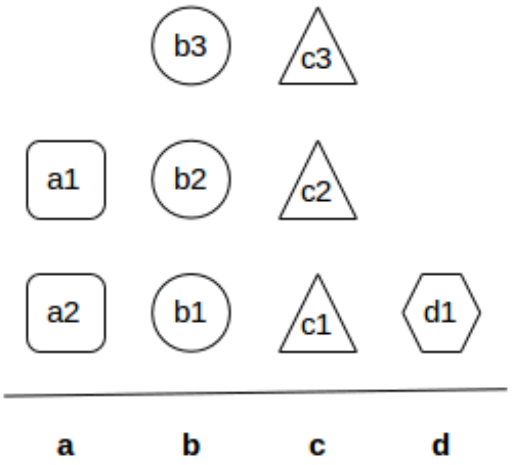


Figure 4.3: State space representation in Approximation 2

Approximation 2 In approximation 1 the state space is reduced to include only those states, i.e. entities, which are deemed potential states to emit the observations, i.e. mentions, in the test sequence to be labelled. However, a further observation is that not all states in this restricted state space are possible states for all observations. If approximation 1 works perfectly, then every state will emit the correct observed textual mention and the state should be in the mention candidate list. However, there is nothing to guarantee this as the relation between state and observation is many-to-many.

A more precise application of our observation is presented in this approximation. Not only is the state space customized for each test sequence to be labelled, the state space is determined for each NE textual mention. In this approximation, we reduce the state space for each observation separately. So, for every observation there is a specific state list, i.e. state list as shown in figure 4.3 where a, b, c, and d candidates are presented as rectangles, circles, triangles, and hexagons,

respectively. Hence a new E_M will be defined where $E_M \subset E$ and $E_M = \{E_1, E_2, \ldots, E_n\}$ and $E_i = \{e_i^1, e_i^2, e_i^3, \ldots, e_i^k\}$ where k is the number of NE candidates for a mention m_i. The HMM decoding steps in this case are:

1. Initialization:

$$\delta(j) = p(e_1^j) \times p(m_1|e_1^j) \ , \ 1 \leq j \leq |E_1|$$

2. Recursion

$$\delta_t(j) = max_{1 \leq i \leq |E_t|} \left[\delta_{t-1}(i) \times a_{ij}\right] \times p(m_t|e_j^t)$$

where $1 \leq j \leq |E_t| \, ; 2 \leq t \leq k$

Note that in this case a separate set of states is considered for each mention.

Approximation 3 In collective NED approaches a basic intuition is that named entities mutually inform the disambiguation of each other. This mutual information is inherently order free, i.e. independent of the order in which the named entities appear in the training or test sequences. However, the HMM technique we have adopted is order dependent, both at the model building stage and in decoding. One consequence is that the order in which observations occur in the query/test sequence may be one for which the model contains sparse or inaccurate information and that presenting the same observations in a different order would result in a more accurate tagging. One might consider computing all possible orderings of NE mentions in the query sequence, using the Viterbi algorithm to find the optimal labelling for each ordering and then picking the labelling with the highest probability across all orderings. However, this is not computationally feasible. Nevertheless, one simple heuristic is to reorder the observation sequence according to the *ambiguity degree* of the observations.

Ambiguity Degree We define the ambiguity degree of a NE textual mention as a measure of how ambiguous the NE textual mention is. It is defined as the number of candidate named entities for the NE textual mention. For a specific NE textual mention, the lower the number of candidates the less ambiguity is.

Thus, in this variant of our approach, the state space is reduced for each textual mention separately as in approximation 2, but with one difference which is reordering the mention sequence by ambiguity degree, as shown in figure (4.4). HMM decoding steps are the same as in approximation 2. However, we now compute the best state sequence probability for the query mentions twice: once in the order they occur in the query document and once after reordering the query mentions according to ambiguity degree. The state sequence with the highest probability is considered the solution for the query sequence.

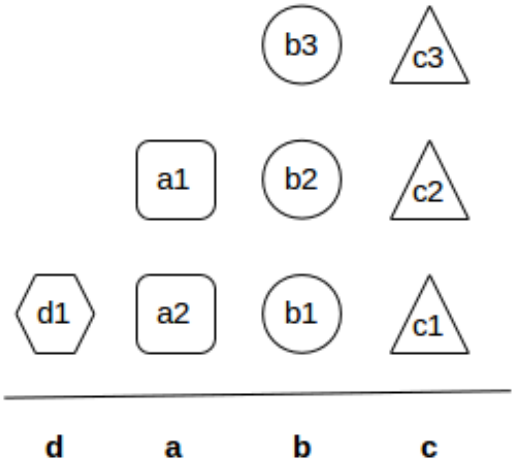


Figure 4.4: State space representation in Approximation 3

4.3 Dataset

For our experiments we use the AIDA-dataset, which is based on the CoNLL 2003 data for NER tagging, in which most tagged NE mentions have been disambiguated against Wikipedia [Hoffart et al., 2011]. The dataset contains 1,393 documents with 34,956 NE textual mentions of which 27,817 have been disambiguated against Wikipedia. Details of this dataset are presented in section 2.5.2.

As discussed in section 4.2.1 on HMM training, DBpedia is used to get a list of all named entities found in Wikipedia, of types Person, Location, and Organization. However, not all named entities in Wikipedia are classified in DBpedia. In particular, some named entities that

occur in AIDA are not included in DBpedia. Since this results in no training sequences that mention these named entities being extracted from Wikipedia, our system could not possibly disambiguate mentions of their NEs in the test set. To overcome this problem, all NEs found in the AIDA dataset which are not classified in DBpedia are added to our list of Person, Location, and Organization entities.

4.4 Experimental Results

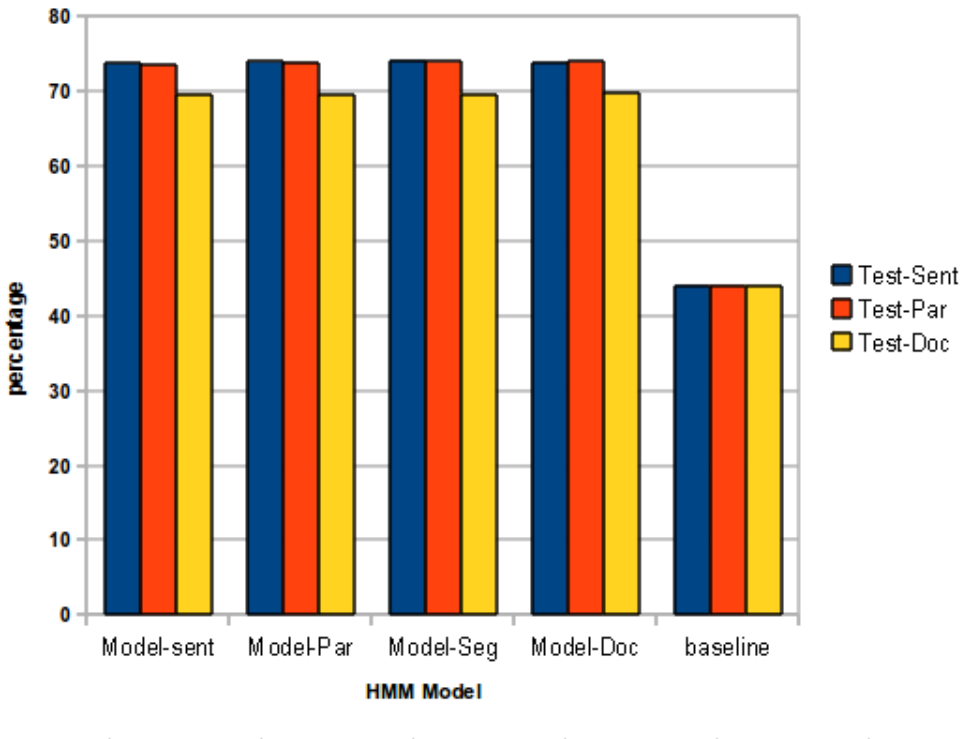


Figure 4.5: Macro Accuracy of NED using HMM with Approximation 3

Our baseline is a simple statistical approach that uses co-occurrence counts of NE mentions and the named entities to which they are linked in Wikipedia. An NE mention is always disambiguated to the NE with which it is most frequently associated in the Wikipedia pages.

During testing, the test data is segmented using three context schemes: sent, par and doc. While the test collection is not segmented into paragraphs, a heuristic is used to build up artificial paragraphs by taking consecutive sentences until the number of NE mentions is 3 or more. For each context scheme, the NE textual mention sequence is extracted and Viterbi is used to decode

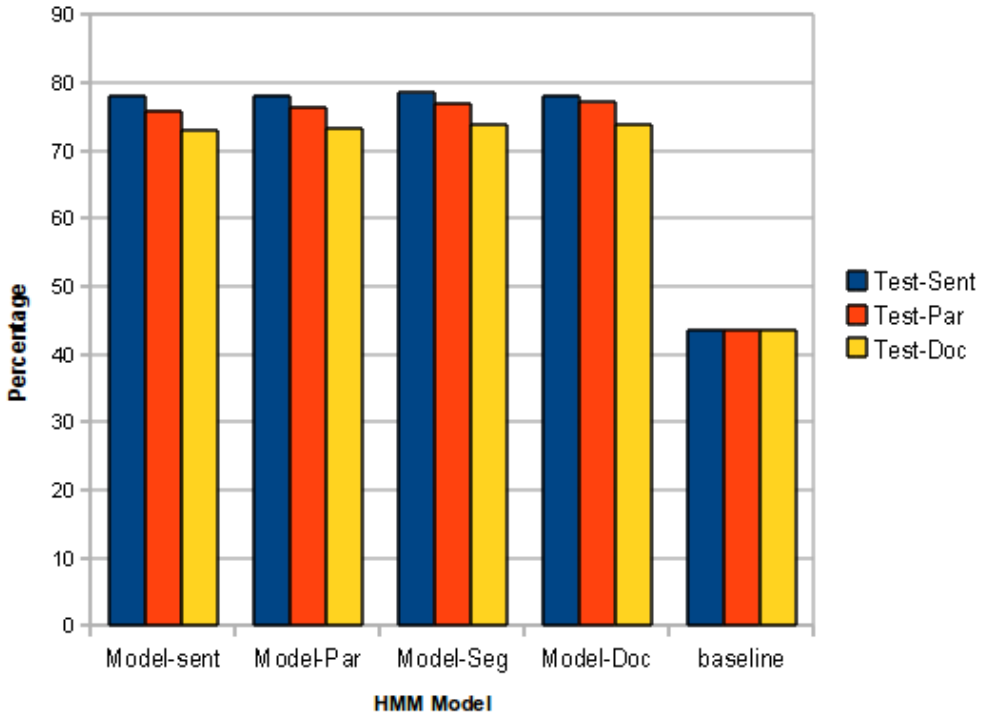


Figure 4.6: Micro Accuracy of NED using HMM with Approximation 3

the mention sequence using one of the pre-prepared models (see section 4.2.1).

A set of experiments was carried out to test the accuracy of different approximations using the different models and different contexts. Micro and macro accuracy are used as the evaluation measures. Recall that, micro accuracy is accuracy over all NE mentions in the test set, while macro accuracy is the accuracy per document averaged over all test documents (see section 2.7). Results are presented in table 4.2 and show best accuracy occurs when using approximation 3, sentence context in the query and the model μ_{seg}. For each approximation, the query context segmentation scheme has an impact on the results, while the effect of changing models is very slight. This observation is shown in an alternative presentation in figures 4.5 and 4.6, which show accuracy figures for Approximation 3.Theoretically, using shorter model contexts may divide the longer sequences into a smaller sequences. Then, transition probabilities of states at sequence boundaries are affected.

Test Context	μ	Approx. 1		Approx. 2		Approx. 3	
		A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}
Sent	μ_{sent}	62.09	69.46	69.33	74.65	73.76	78.09
	μ_{par}	62.44	69.76	69.72	74.97	73.96	78.18
	μ_{seg}	62.10	69.91	70.24	**75.55**	**74.18**	**78.49**
	μ_{doc}	60.06	68.74	70.03	75.50	73.90	78.13
Par	μ_{sent}	55.12	61.07	68.98	72.06	73.54	75.89
	μ_{par}	55.87	61.87	69.17	72.42	73.90	76.39
	μ_{seg}	56.79	63.09	70.14	73.51	73.99	76.94
	μ_{doc}	56.10	62.80	**70.28**	73.76	74.00	77.09
Doc	μ_{sent}	50.32	57.01	68.69	71.27	69.64	72.86
	μ_{par}	50.90	57.78	68.60	71.50	69.55	73.27
	μ_{seg}	51.82	59.12	68.92	72.15	69.66	73.73
	μ_{doc}	52.40	59.47	69.08	72.34	69.74	73.84

Table 4.2: Results of using different approximations for HMM disambiguation

4.5 Discussion

4.5.1 Comparison with the state-of-the-art

It is difficult to compare our results to the state-of-the-art results because there is no standard benchmark. Hoffart et al. [2011] re-implemented the methods of Cucerzan [2007] and Kulkarni [2009] and evaluated them using the AIDA dataset. Table (4.3) shows a comparison between the results of our approach, the baseline approach, and some state-of-the-art results. However, our approach is very simple and direct to apply, it exceeds the results of Cucerzan and Kulkarni, but not Hoffart's, which is more complex than ours.

	Approx. 1	Approx. 2	Approx. 3	Baseline	Cucerzan	Kulkarni	Hoffart
$Accuracy_{macro}$	62.10	70.24	74.18	44.06	43.74	76.74	81.91
$Accuracy_{micro}$	69.91	75.55	78.49	43.55	51.03	72.87	81.82

Table 4.3: HMM and State-of-the-art Results

Hoffart et al. proposed a graph model representation of all candidates for all NE textual mentions and treated the NED problem as finding the dense sub-graph. They used popularity prior feature which is equivalent to the emission probability in our proposed model. Moreover, they used more advanced features to measure the similarity between the NE textual mention and the NE candidate, such as keyphrase-based and syntax-based similarity. Graph links represent

the coherence between different entities, or between the prior probability of the NE textual mention and the NE candidate. The strength of Hoffart's work is using the graph model to represent the interdependency and coherence relations, while we used a sequence model. The graph model is more general than the sequence model.

Cucerzan proposed a collective disambiguation approach that models the interdependence between the disambiguation decisions. Each entity in the Wikipedia KB is represented as two vectors, one is the context vector which represents all NE links in the entity page, and the other vector is a vector of category tags of this entity page. NE textual mentions in the query document are mapped to NE candidates using a static repository. So the query document is also represented as two vectors, one vector contains all category tags associated with all NE candidates of all NE textual mentions, and the other is the context vector which contains all candidates of all NE textual mentions. The disambiguation process aims to maximize the agreement between the KB entity and the document vectors. The disadvantage of Cucerzan's work is using static candidates generation. Cucerzan uses a map of named entities and NE textual mentions which are extracted from Wikipedia hyperlinks, redirects, and disambiguation pages. So the correct disambiguation entity is less likely to be in the candidate list. Moreover, Cucerzan uses the context entities, i.e. the entities found in the entity page in Wikipedia, but does not consider their importance or dependency.

4.5.2 Analysis

We analysed the knowledge base sequences, test data sequences and results. We make the following observations.

How does context affect the results? Accuracy goes down when we used the document context scheme in testing. This is likely because different paragraphs in a long document may have different topics and the named entities mentioned in them may be weakly related. At the same time, the results are improved when we used the segment context when extracting the training sequences. This is likely because of high coherence and dependency between named entities found in Wikipedia page segments.

Topology problem According to our assumption, the dependencies between all named entities are represented as a Markov Network. However, Markov network is not fully connected, i.e. each node is not connected to all other nodes. For each mention sequence, we tried to find the best sequence of named entities/states from the set of NE candidates for the different mentions in the context. There are many solutions represented in different patterns in the NE dependency network. Figure 4.7 shows the NE candidate dependency network for four mentions a, b, c, and d. The shaded nodes represent the disambiguation candidates and the dashed links represent the connections between the correct candidates. The correct candidates are in the shaded shapes $a2, b2, d1$, and $c1$. However, an HMM model cannot decode the correct pattern here because three of the entities are dependent on only one.

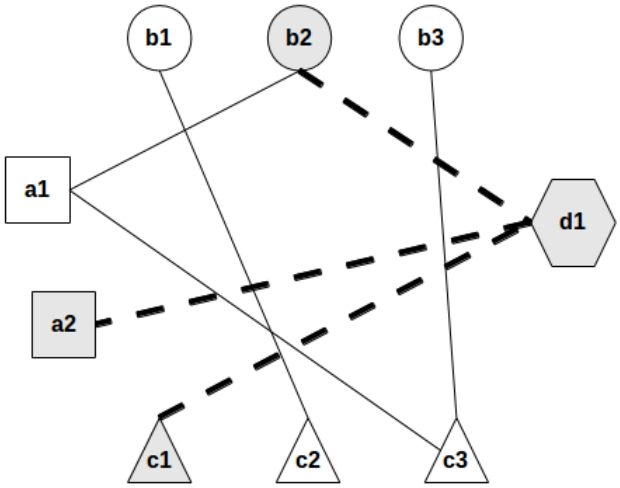


Figure 4.7: NE disambiguation network

As we found the sentence context the most robust and reliable context, we analysed the results when using the sentence context in disambiguation. We found 50% of the sequences are of length one (i.e. consist of just one NE mention) and that these sequences contain 26% of the named entities in the test set. For such singular sequences, the decision is taken by considering only the prior and emission probabilities which is not sufficiently reliable. So, collective disambiguation is effective in sequences of length more than one.

4.6 Conclusions

In this chapter we presented a novel approach for NED using HMMs. The proposed approach modelled the dependency between different named entities and tackled the NED problem as finding the best state sequence of candidates given a sequence of NE textual mentions (observations) in the query document. The Viterbi algorithm is used to decode the observation sequence into the hidden state sequence that emits this observation sequence. It is infeasible to use Viterbi to decode an observation sequence in a huge state space, so we proposed three different approximations to overcome this problem.

Our results show that HMMs can be used as an effective approach to collectively disambiguate different textual mentions of different named entities in a document. Our proposed approximations make using HMMs feasible for the NED task, where both numbers of states and observations are huge. Using the information contained in the joint presence of named entities may not be sufficient to solve the NED problem on its own, but it goes a surprising way towards doing so.

A new sequence of observations which have never been seen before in the training data can be decoded using a HMM model. However, in some cases the correct disambiguation candidate entities are not presented in a sequence. So, improving the features and increasing training for the model will not improve the results of this approach. We conclude that graph modelling is more suitable for modelling NE dependency rather than sequence modelling. In the next chapter we present two novel approaches for NED based on graph modelling for NE dependency.

Chapter 5

Graph-Based Named Entity Disambiguation*

5.1 Introduction

In chapter 4 we presented an HMM based approach for NED. These problem was tackled as a time series stochastic process where NE entries in the KB are treated as a hidden states and the textual mentions as the emitted observations. The main problem with this formulation is the statistical dependency topology, i.e. the dependency between one named entity candidate and the different NE candidates of other textual mentions may not be decoded, as Viterbi always chooses the best candidate at time t that maximizes the overall sequence probability. This chapter presents a graph based approach for collective named entity disambiguation where disambiguation of each textual mention considers the possible candidates of other NEs textual mentions. The main hypothesis in this approach is that different named entities in a document help to disambiguate each other and one NE may help to disambiguate more than one other textual mention.

In our approach, all possible NE candidates are represented as nodes in the graph and associations between different candidates are represented by edges between the nodes. Each node

*Parts of this chapter have been published in *Proceedings of the 25th International Conference on Computational Linguistics (COLING 2014)*, 2014.
Also in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, 2014.

has associated with it an initial confidence scores, e.g. entity popularity. Such graphs are called *solution graphs*. Figure 5.1 shows an example of the solution 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. The nodes in this set should be highly coherent and have high confidence scores.

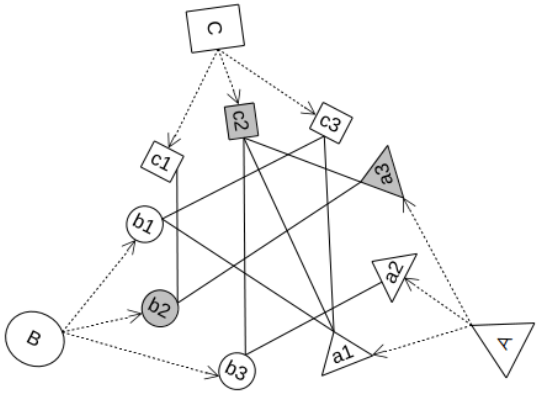


Figure 5.1: Example of solution graph

Our approach first ranks all nodes in the solution graph using the Page-Rank algorithm [Page et al., 1999], 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 links in our graph. The proposed graph approach is compared with a baseline and state-of-the-art approaches [Cucerzan, 2007, Hoffart et al., 2011, Kulkarni et al., 2009, Shirakawa et al., 2011]. Experiments are carried out using AIDA dataset (see section 2.5.1) and the results show the effectiveness of

using Page-Rank in conjunction with initial confidence.

The rest of this chapter is structured as follows: The following section (5.2) presents a quick review on the Page-Rank algorithm. Section 5.3 describes the process of selecting a candidate list for each NE textual mention from the Wikipedia KB and assigning a confidence score for each NE candidate. Section 5.4 describes the solution graph and the entity coherence representation. Section 5.5 presents our clique disambiguation approach for collective named entity disambiguation. Section 5.6 presents the graph ranking approach for collective named entity disambiguation including graph ranking, score combination schemes, and candidate selection techniques. Experimental results, comparison to the state-of-the-art and analysis of the results are presented in section 5.7. Finally, section 5.8 concludes the presented graph-based approaches.

5.2 Page-Rank Algorithm

The Page-Rank (PR) algorithm was developed by Larry Page and Sergey Brin in order to rank Web search results [Page et al., 1999]. It is a general algorithm to compute the rank of each node in a graph based on the links between nodes. The main intuition underlying this algorithm is that, pages that have more incoming links from important nodes should receive a higher rank than pages that have fewer incoming links from less important nodes. So, a node has a high rank if the sum of the ranks of incoming link nodes is high. In this section, a simple brief explanation to PR algorithm is presented.

A simplified version of Page-Rank is presented in formula 5.1 [Brin and Page, 2000] where

y: The target node to be ranked;

d: A damping factor, which is a fraction from 0 to 1;

$in(y)$: The set of nodes that have links pointing to node y;

$out(x)$: The set of nodes that have links from node x;

$$PR(y) = (1 - d) + d \sum_{x \in in(y)} \frac{PR(x)}{|out(x)|} \tag{5.1}$$

Example: To make this clearer, let us use the graph example shown in figure 5.2. Let us ignore the damping factor ($d = 1$) and let the initial rank for all pages be the uniform distribution. So, $PR(e1) = PR(e2) = PR(e3) = PR(e4) = 0.25$. Then,

$$PR(e1) = \frac{PR(e2)}{2} + \frac{PR(e3)}{1} + \frac{PR(e4)}{3} = \frac{0.25}{2} + \frac{0.25}{1} + \frac{0.25}{3} = 0.4583$$
$$PR(e2) = \frac{PR(e4)}{3} = \frac{0.25}{3} = 0.0833$$
$$PR(e3) = \frac{PR(e2)}{2} + \frac{PR(e4)}{3} = \frac{0.25}{2} + \frac{0.25}{3} = 0.2083$$

Page-Rank models the user behaviour, where a surfer clicks on links at random. This random surfer visits a particular web page with a certain probability. This is why the page-rank of a certain page receives the rank of incoming nodes divided by the number of outgoing links. The probability of this surfer visiting a given page is the sum of probabilities for the surfer clicking any link pointing to this page. For *sink nodes*, where there are no outgoing links, the surfer needs to change to another random node in the graph. Also, the surfer may getting bored and suddenly change to another node in the graph without following any links. So, the probability is reduced by the damping factor, d, the probability that the surfer does not stop clicking on links. The higher d is, the more likely the random surfer will keep clicking links; so $1 - d$ is the probability the surfer stops clicking links and jumps directly to the page. Therefore, all nodes always have a minimum rank, which is the probability of the surfer jumping to this page. The second version

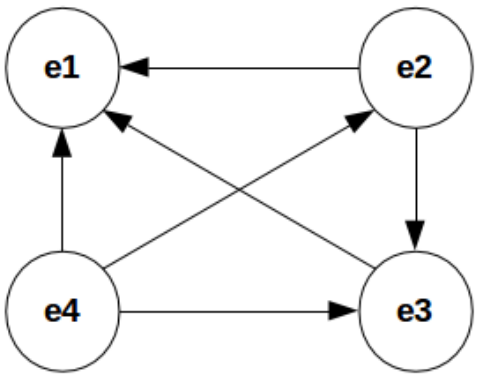


Figure 5.2: Example of solution graph

of the Page-Rank algorithm normalized this factor by dividing by the number of nodes in the graph, as shown in equation 5.2 where N is the number of nodes in the graph.

$$PR(y) = \frac{(1-d)}{N} + d \sum_{x \in in(y)} \frac{PR(x)}{|out(x)|} \tag{5.2}$$

In our example all edges are considered to be equally weighted. So, the rank of the source node is equally distributed over the destination nodes. When the graph edges are weighted, the source node rank is proportionally distributed over the target nodes according to the edge weight. Let us suppose node x has initial rank 100 and four links to four nodes with weights $0.2, 0.15, 0.35, 0.30$. Then, the initial rank will be distributed over four nodes and the nodes will receive the following values $20, 15, 35, 30$. In the normal case where all edges are equally weighted, each destination node will receive 25. So, if weighted edges are to be used PR is calculated as shown in equation 5.3, where $W_{x,y}$ is the weight of the edge from node x to node y.

$$PR(y) = \frac{(1-d)}{N} + d \sum_{x \in in(y)} PR(x) \times W_{x,y} \tag{5.3}$$

In our proposed solution, we use the Page-Rank implementation provided in the NetworkX package[2] which implements an eigenvector method for Page-Rank [Langville and Meyer, 2005].

5.3 Named Entity Candidate Generation

In this section, we present process of selecting NE candidates from the KB. 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[3] to find entries with titles that fully or partially contain the NE textual mention. The following

[2] `http://networkx.lanl.gov/reference/generated/networkx.algorithms.link_analysis.pagerank_alg.pagerank.html` (last visited 30-Jun-2014)

[3] `https://lucene.apache.org/` (last visited 30-Jun-2014)

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.

5.3.1 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[4].

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.

5.4 Solution Graph

In this section we discuss the construction of a graph representation that we call the *solution graph*. 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 (see figure 5.1 for example). Different approaches

[4] `https://developers.google.com/freebase/v1/search` (last visited 30-Jun-2014)

to determine and weight entity coherence relations are presented in the following section.

5.4.1 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 document context, 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.

$$\text{Ref}(e_i, e_j) = \begin{cases} 1, & \text{if } e_i \text{ or } e_j \text{ refers to the other} \\ 0, & \text{otherwise} \end{cases} \tag{5.4}$$

- 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 (5.5), 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.

$$\text{Jprob}(e_i, e_j) = \frac{|S(e_i) \bigcap S(e_j)|}{|S|} \tag{5.5}$$

5.5 Cliques Partitioning Disambiguation

The clique model originated in social network studies when Luce and Perry 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 which appear in the same document can be split into groups of highly cohesive entities, we adopt the clique partitioning

technique to find the largest 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 the 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, then 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 5.3 shows an example of the clique partitioning disambiguation algorithm with a graph of candidate entities for six NE textual mentions, A, B, C, D, E, and 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.

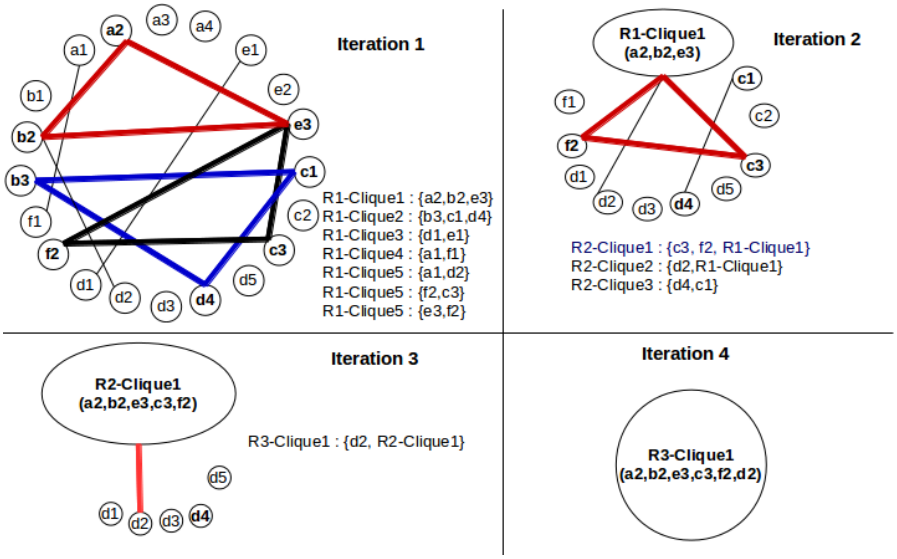


Figure 5.3: Example of Clique Partitioning Disambiguation

As described in section 5.4, one of the properties of the disambiguation graph is that there are no links between candidates of the same NE textual mention. As a result of this, 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 uses the entity links to find the cliques, regardless of the link weight. We used Algorithm 457 Bron and Kerbosch [1973] to find all maximal cliques in a graph. The algorithm is quite slow due to being recursive and the huge number of nodes. To speed up the disambiguation, we filtered the nodes with low confidence from the graph, keeping only the top 50 NE candidates for each NE textual mention. The NetworkX package also provides an implementation for this algorithm, which is used in this work.

5.6 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.

5.6.1 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 5.6, 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.

$$PR(e_i) = \frac{(1-d)}{N} + \frac{d}{F(e_i)} \sum_{e_j \in coh(e_i)} PR(e_j) \times W(e_i, e_j) \tag{5.6}$$

$$F(e_i) = \sum_{e_j \in coh(e_i)} W(e_i, e_j) \tag{5.7}$$

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.

5.6.2 Candidate Re-Ranking

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 5.9 and 5.8.

$$R_m(e_{i,j}) = IConf(e_{i,j}) \times PR(e_{i,j}) \tag{5.8}$$

$$R_s(e_{i,j}) = ICONf(e_{i,j}) + PR(e_{i,j}) \tag{5.9}$$

5.6.3 Decision Making

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

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 5.10 or 5.11, which correspond to the rank combining schemes expressed in equations 5.8 and 5.9. 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.

$$\hat{e}_i^m = \underset{e_{i,j}}{\operatorname{argmax}}\ R_m(e_{i,j}) \tag{5.10}$$

$$\hat{e}_i^s = \underset{e_{i,j}}{\operatorname{argmax}}\ R_s(e_{i,j}) \tag{5.11}$$

5.7 Experiments and Results

5.7.1 Experimental setup

The AIDA dataset is used in testing the proposed Graph-Based approach (see section 2.5.2 for more details). This dataset contains 1393 documents, and 34,965 annotated mentions, where 7136 mention are not linked to Wikipedia. For fair comparison to Hoffart's work, we only considered NE mentions with an entry in the Wikipedia KB, ignoring 20% of query mentions without a link to the KB, as Hoffart did.

Micro- and macro-averaged accuracy are the measures used to evaluate the proposed approach. Details of these measures are shown in section 2.7.

The Python package "NetworkX" is used to build the solution graph. This package provides an implementation of page-rank algorithm which is used in the experiments. This implementation uses the power method solver where the maximum number of iterations is set to 1000 and the convergence tolerance is set to 1.0e-8. The final setting is the damping factor used for page-rank: alpha = 0.85.

5.7.2 Baseline Results

Initially, we evaluated the performance of two baselines. One is a setup where a ranking based solely on different initial confidence scores is used for candidate selection, i.e. without using Page-Rank. In this setup a ranking based on Freebase popularity does best, with micro- and macro-averaged accuracy scores of 80.55% and 78.09%, respectively. This is a high baseline,

close to the state-of-the-art. Our second baseline is the basic PR algorithm, where weights of nodes and edges are uniform (i.e. initial node and edge weights are set to 1, with edges being created wherever REF or JProb are non-zero). Micro- and macro-averaged accuracy scores of 70.60% and 60.91% were obtained with this baseline. Detailed results is shown in table 5.1.

Ranking Score	A_{micro}	A_{macro}
Cosine Similarity (Cos)	38.44	45.68
Jaro-Winkler (JwSim)	61.01	58.81
Context Simillarity (Ctxt)	24.58	21.44
JwSim+Ctxt	62.38	56.58
Entity Popularity (EP)	80.55	78.09
Page-Rank score (PR)	70.60	60.91

Table 5.1: NED using Initial Confidence Score or PR

5.7.3 Cliques Approach Results

The clique partitioning disambiguation algorithm experiments are set up so that a link between nodes is created whenever a non-zero coherence relation is found between nodes regardless of 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. 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, the clique partitioning algorithm takes a long time because of the huge graph size. 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 5.2 shows the results of using different initial confidence scores in clique partitioning disambiguation.

IConf	Cos	JwSim	Ctxt	EP
A_{micro}	71.59	72.26	58.06	86.10
A_{macro}	64.83	69.53	57.37	81.79

Table 5.2: NED using Clique Partitioning Approach

$IConf$	PR		e^s		e^m		e^g	
	A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}
Cos	70.60	60.83	79.19	73.56	59.73	56.75	78.41	72.35
JwSim	70.61	60.94	82.61	77.66	79.34	73.89	83.16	78.28
Ctxt	70.61	60.83	74.18	62.95	70.09	58.93	75.45	65.22
JwSim+Ctxt	70.63	60.86	82.86	77.59	80.38	74.40	83.37	78.35
EP	71.78	61.23	86.93	83.90	82.03	81.07	87.59	84.19

Table 5.3: Results using initial confidence to initialize node rank before using Page-Rank (PR_I)

5.7.4 Graph Ranking Results

To study the graph ranking using PR, and the contributions of the initial confidence and entity coherence, experiments were carried out using PR in different modes and with different selection techniques. In the first experiment, referred to as PR_I, initial confidence is used as an initial node rank for PR and edge weights are uniform, with edges being created wherever REF or JProb are non-zero, as in the PR baseline. Table 5.3 shows the results both before re-ranking, i.e. using only the PR score for ranking, and after re-ranking using different selection schemes (indicated by $\hat{e}^s$, $\hat{e}^m$, and $\hat{e}^g$). When comparing these results to the PR baseline we notice a small positive effect in using the initial confidence as an initial rank, instead of uniform ranking. The major improvement comes from re-ranking nodes by combining the initial confidence with the PR score. All combination methods and selection schemes improve the results over the baseline results when using the same confidence score, while the dynamic selection scheme overcomes the static one.

In our second experiment, PR_C, entity coherence features are tested by setting the edge weights to the coherence score and using uniform initial node weights. We compared JProb and REF edge weighting approaches and a variant in which the REF edge weights are normalized to sum to one over the whole graph, and are then added to the JProb edge weights (Jprob+Ref). Results in table 5.4 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 initial confidence score, i.e. Freebase score, for re-ranking. Again, combining the initial confidence with the PR score improves the results.

Finally, table 5.5 shows the accuracy when using different combinations of initial confidence

Edge Weight	PR		e^s		e^m		e^g	
	A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}
$Jprob$	66.52	55.83	83.16	80.50	80.92	79.63	83.31	80.38
Ref	67.48	59.76	82.06	79.10	80.18	79.03	81.80	78.53
$Jprob + Ref$	72.69	65.71	83.59	80.98	81.33	80.53	83.46	80.69

Table 5.4: Results using edge weights for Page-Rank (PR_C)

and entity coherence scores just in the case when re-ranking is applied. Here, the $Jprob + Refs$ combination does not add any value over $JProb$ alone. Interestingly, using initial confidence with differentially weighted edges does not show any benefit over using initial confidence and uniformly weighted edges (table 5.3).

5.7.5 Comparison To the State-of-the-art

To compare our results with the state-of-the-art, we report Hoffart et al.'s results [2011], as they re-implemented two other systems and ran them over the AIDA dataset, which we also used to evaluate our approach. We also compare with Shirakawa et al. [2011] who carried out their experiments using the same dataset. Table 5.6 shows a comparison between the results of our approach and the state-of-the-art. Our approach exceeds the results of the state-of-the-art. However our approach is very simple and direct to apply, unlike Hoffart et al.'s and Shirakawa et al.'s which are considerably more complex. Also, our approach does not require any kind of training, like the HMM-based approach presented previously (in chapter 4.)

Hoffart et al. proposed a graph model representation of all candidates for all NE textual mentions and treated the NED problem as finding the dense sub-graph. Moreover, they used more advanced features to measure the similarity between the NE textual mention and the NE candidate, such as keyphrase-based and syntax-based similarity. Graph links represent the coherence between different entities, or between the prior probability of the NE textual mention and the NE candidate. Our approach differs from Hoffart's work in evaluating all graph nodes, without resorting to reduction or greedy algorithms. Also, the features we used are simpler and more direct to calculate.

Shirakawa et al. did not use candidate generation, as they used a probabilistic taxonomy to conceptualize the NE textual mentions and use Freebase to find the entities that have the same

$IConf$	Edge Weight	PR		e^s		e^m		e^g	
		A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}	A_{micro}	A_{macro}
Cos	$Jprob$	64.18	52.25	77.22	70.04	57.16	56.71	77.24	70.42
	Ref	62.7	49.04	72.14	63.18	57.94	52.95	71.27	61.45
	$Jprob + Ref$	**68.66**	**56.71**	77.33	70.71	58.31	53.6	77.82	70.83
JwSim	$Jprob$	64.18	52.25	82.17	75.27	78.67	72.42	82.56	76.16
	Ref	62.67	48.92	78.49	71.34	75.18	68.27	78.61	71.12
	$Jprob + Ref$	**68.66**	**56.71**	81.59	75.63	78.73	72.28	81.97	75.63
Ctxt	$Jprob$	64.15	51.83	70.12	55.26	67.74	55.99	70.93	57.9
	Ref	62.7	49.03	67.88	54.67	64.41	51.56	68.82	56.22
	$Jprob + Ref$	**68.66**	**56.71**	72.69	60.1	68.21	56.76	74.27	63.18
JwSim+Ctxt	$Jprob$	64.18	52.06	82.38	75.1	80.09	73.53	82.76	76.21
	Ref	62.67	49.01	78.79	71.76	76.09	68.64	78.78	71.65
	$Jprob + Ref$	**68.66**	**56.71**	81.82	75.72	79.95	72.91	82.29	76.26
EP	$Jprob$	64.17	51.23	85.92	82.73	80.21	78.95	**86.29**	**82.77**
	Ref	62.70	50.99	82.93	80.05	79.54	78.29	83.16	80.01
	$Jprob + Ref$	**68.66**	**57.71**	**85.56**	**82.42**	**80.76**	**79.50**	**86.10**	**82.80**

Table 5.5: Page-Rank with Weighted edges and Initial confidence Scores

	A_{micro}	A_{macro}
$IConf$	80.55	78.09
PR_C	83.59	80.98
PR_I	87.59	84.19
PR_{CI}	86.10	82.80
$Clique$	86.10	81.79
Cucerzan	51.03	43.74
Kulkarni	72.87	76.74
Hoffart	81.82	81.91
Shirakawa	81.40	83.57

Table 5.6: Comparison Between Proposed Approaches and State-of-the-art

concepts. When more than one entity is found, the Freebase rank is used to select the highest ranked entity. Looking at the Shirakawa results analysis, a high accuracy is achieved with the short text documents; this is because their approach is individual disambiguation, i.e. not a collective approach, so they do not gain any benefits from the co-occurring named entities. Also, for larger documents, more terms contribute in defining the concepts, which may be more confusing. Our collective approaches presented in this chapter work better with the huge documents that contain more NE textual mentions. Shirakawa's approach may be better than collective approaches when there is only one NE textual mention in the document. In such cases, the individual approaches work better than collective.

Han et al. [2011] presents a graph-based method which looks similar to a graph ranking approach. They proposed to use the compatibility between an NE candidate e and an NE textual mention m. This compatibility measures the overlapping words between the NE textual mention context (a window of size 50) and the NE article in the KB using the *bag-of-words* model. Compatibility measure is represented as an edge between the textual mention node and the NE candidate node in the *referent graph*. Also, semantic relations between entities are used to represent the links between different NE candidate nodes (for more details on the semantic relation feature, see section 2.4.2). The NE textual mention is used as the evidence for its NE candidates. Evidence importance is calculated by the proportion of this evidence in the document relative to the other evidences. The initial evidence reinforced by propagating them according to the graph links. Both compatibility and relatedness are used as link weights in the graph.

An iterative algorithm is used to propagate the evidence importance on the graph using the link weights. The disambiguation is done by selecting the node that maximizes the evidence and compatibility scores.

Unfortunately, we are unable to compare our results to Han et al.'s approach for two reasons. The first is that they used a dataset which is different from the AIDA dataset. Secondly, they used another evaluation metric to measure the method performance, i.e. precision and recall. However, we highlight the differences between our graph-based approaches and Han's approach, and comment on their published results.

- The graph representation is different. We refer to Han's graph as a referent graph and our graph as a solution graph. In the referent graph both NE textual mentions and NE candidates are represented as nodes, while in the solution graph only NE candidates are presented as nodes. The solution graph is also undirected, while the referent graph has directed links between candidate nodes and NE textual mention nodes, and undirected links between NE candidate nodes.

- There are some similarities between the Page-Rank and the collective inference, as both are a random walk process in the graph. However, Han propagates the initial evidence importance within the graph and falls to the same dissipation of evidence problem we previously highlighted. In our graph ranking approach we recombine the NE candidate confidence with the PR score and dynamically choose between two methods of combination, which implies a considerable improvement.

- Using the context similarity as a compatibility measure is not a good measure, as our results show.

- Both Han's and our graph-ranking approach aim to find a globally optimized set of coherent entities, which includes only one NE candidate for each NE textual mention. The clique disambiguation approach aims to find a set of locally optimised coherent NE candidates that disambiguate a subset of the NE textual mentions, and iteratively expand this set, or find new sets, until all NE textual mentions are assigned an NE candidate in these

selected sets.

- In Han's experiments, textual mentions which have an entry in the KB for evaluation are used; at the same time, they reported 0.76 recall. As they did not discuss the candidate generation method, we can not explore the reasons for this low recall. Also, they reported 0.87 precision with 0.20 recall, which indicates a weak approach.

5.7.6 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 initial confidence as an initial rank before using PR (see table 5.3). However, when comparing PR results in tables 5.3 and 5.4, 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 initial confidence 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 5.7 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	21.367	62.119	1.327
Ford Galaxie	4.593	10.937	0.050
Ford GT	2.835	11.433	0.032
Ford Zephyr	0.831	13.705	0.011
Ford Scorpio	0.831	9.514	0.008
Magna			
Magna International	2.647	4.779	0.013
Magna Powertrain	2.647	2.178	0.006
Germania	0.831	3.466	0.003
Chew Magna	0.831	2.988	0.002
Fascioloides magna	0.831	2.902	0.002
New York Stock Exchange			
New York Stock Exchange	4.405	42.339	0.186
Silver v. New York Stock Exchange	0.831	51.050	0.042
Stock exchange	0.831	22.422	0.019
New York Sack Exchange	0.831	9.189	0.008
Zimbabwe Stock Exchange	0.831	0.000	0.000
North American			
North America	16.546	52.688	0.872
North American Plate	0.831	18.023	0.015
North American Review	0.831	15.711	0.013
North American Union	0.831	13.398	0.011
North American GAA	0.831	12.970	0.011
Johnson Controls Inc			
Johnson Controls	4.603	125.000	0.575
Who Controls the Internet?	0.831	0.000	0.000
Vickie Johnson	5.537	0.000	0.000
Vantage Controls	0.831	0.000	0.000
Universal controls	0.831	0.000	0.000
FARMINGTON HILLS [Farmington Hills, Michigan]			
Farmington Public Schools	0.831	111.915	0.093
North Farmington High School	0.831	13.085	0.011
Wiscasset, Waterville and Farmington Railway	0.831	0.000	0.000
West Farmington, Ohio	0.831	0.000	0.000
University of Maine at Farmington	0.831	0.000	0.000
Markham [Markham, Ontario]			
Clements Markham	0.831	4.415	0.004
Markham Waxers	0.831	3.669	0.003
Edwin Markham	0.831	2.894	0.002
Monte Markham	0.831	2.803	0.002
E. A. Markham	0.831	2.749	0.002

Table 5.7: Example of NE Candidate Scores

5.8 Conclusion

In this chapter, we presented two collective approaches for named entity disambiguation based on graph models. Both approaches present all candidates for all NE textual mentions as nodes in a graph and link these nodes using the entity coherence relation. The first approach uses the Page-Rank algorithm to rank all graph nodes in different settings and then combine the candidate confidence score with the Page-Rank score to select the disambiguation candidate for each NE textual mention. The second approach evaluates all possible cliques in the graph using the candidate confidence score and selects the best clique; the algorithm iteratively expands the selected clique or finds new good cliques until all NE textual mentions are disambiguated.

Our results show that graph ranking 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 algorithms. Also, entity coherence is a discriminative feature when using graph models for NED.

Clique partitioning approaches can be used to find a cohesive set of candidates, and considering the best clique can help to delete wrong candidates from the graph as a kind of pruning. This pruning helps to expand the selected clique or find other good cliques in the next iterations. This iterative process improves the disambiguation accuracy because some of the wrong candidates will not affect the disambiguation of the other NE textual mentions.

The main conclusion from this work is that graph models are suitable for modelling the candidate dependency and gives the availability to decode different dependency patterns. However, it is very sensitive to the node links, such that missing links (undiscovered or absent relations) affect the accuracy of these approaches.

Chapter 6

Conclusions and Suggestions for Future Work

6.1 Introduction

This study has highlighted different approaches for disambiguating named entity textual mentions in a document. The study set out to find reliable methods to disambiguate different textual mentions of different named entities in a document. The study has also sought to determine whether the statistical co-occurrence between named entities in the Wikipedia knowledge base can result in accurate disambiguation for both individual and collective disambiguation approaches. The study sought to answer the following questions:

- Can document similarity search based on named entities help to find good candidates for named entity disambiguation?
- Can NE dependency relations help in selecting good candidates and in disambiguation?
- What is the best NE candidates dependency model?
- Are collective disambiguation approaches better than individual approaches?

To answer these questions, we first presented a document similarity function, NEB-Sim, based on a document's named entities. Our function results are compared to the traditional

cosine similarity scores using different context schemes. The evaluation metric is the accuracy over the top N candidates. This comparison shows the appropriateness of our NEB-Sim function to select good NE candidates from the knowledge base entries. Also, we explored using different learn-to-rank approaches to rank the retrieved NE candidates for selecting the highest ranked candidates as the disambiguation NE. This approach is an individual disambiguation approach and, while ranking improves the results, it does not match the state-of-the-art results. Therefore, we presented three different collective disambiguation approaches.

The first collective disambiguation approach treats the task of NED as finding the best sequence of named entities, given a set of different mentions for different named entities in the same context. We used an HMM based approach developed different approximations to deal with the huge number of NEs. The approach works successfully to decode the proper sequence of named entities, and works better than some individual disambiguation approaches. We realize there is a theoretical limitation to this approach because some dependency patterns can not be recognized using this approach. We got better results when we used graph approaches, where this limitation is not present. The second and third collective disambiguation approaches used a graph model to represent the candidateś coherence relations.

In the second collective NED approach, we used Page-Rank to rank all graph nodes based on the coherence relations. This rank score is combined with the local feature score using a novel algorithm to find a new rank for different candidates. The third collective disambiguation approach uses the cliques partitioning algorithm to iteratively find one or more cliques of highly coherent candidates for different NE textual mentions. The results of both the second and third approaches show good improvement in disambiguation accuracy and exceed the current state-of-the-art results.

The next section lists the findings of this research in addition to our initial research questions.

6.2 Empirical Findings

The main findings of this research are summarized in the following list:

1. Using a document similarity measure based on named entities improves the performance

of named entity candidate selection from a KB for the NED task (see section 3.4.2).

2. NE textual mention co-occurrence relations may help to select initial candidates for a specific NE textual mention (see section 3.4.2).

3. Sentences containing NE textual mentions are the richest contexts from which to extract contextual information or features. Using these contexts improves candidate quality in the candidate generation phase.

4. Sequence modelling is not the best solution for NED. It is limited because multiple NEs may be dependent on only one other NE, and the Viterbi algorithm can not decode such a non-sequential dependency pattern (see section 4.5.2).

5. Graph modelling is the best representation of the NE candidate dependencies; it can handle cases which HMMs can not (see section 4.5.2).

6. Entity coherence or co-occurrence relations are reliable relation for use in graph models to find the most cohesive set of NE candidates for NE textual mentions in a document (see section 5.7.3).

7. NED can be improved by finding the minimum number of coherent sets of candidates of different NE textual mentions in a document. The clique partitioning algorithm is reliable for finding subsets of cohesive NE candidates, given a good coherent set measure (see section 5.7.3).

8. Using the initial confidence score to initialize candidate nodes before using Page-Ranking is helpful to rank the candidate nodes based on the link structure, though it is not enough for NED as the initial confidence scores are distributed over the linked nodes. Combining the Page-Rank score with the confidence score improves the NED accuracy (see section 5.7.4).

6.3 Thesis Contributions

The main contributions of this thesis are:

1. Development of an NE based search framework for retrieving and scoring a reliable short list of NE candidates from a knowledge base (see section 3.2).

2. Demonstrating how using learn-to-rank approaches to re-rank the different candidates of a specific NE textual mention can improve candidate generation for NED (see section 3.3).

3. Design and development of a collective NED approach based on Hidden Markov Models (HMMs) (see section 4).

4. Developing different approximations to be used with the Viterbi algorithm with a huge number of states, making it feasible for use in NED (see section 4.2.2).

5. Developing a graph partitioning algorithm for collective NED (see section 5.5).

6. Developing a graph based collective NED approach using Page-Rank in conjunction with local features and entity coherence to rank NE candidates (see section 5.6).

6.4 Future Work

The previous section describes our contributions towards the problem of named entity disambiguation. There are some problems which still need to be addressed to improve NED accuracy. Future avenues for this work include:

- **Missing Cases Analysis:**

  The first extension to this work is to analyse the cases that are missing, especially in the graph base approaches which achieves good results, and from these cases try to work out why they are missed. This failure analysis may help to figure out better settings for the proposed approaches.

- **Coherent Set Measure:**

  We presented a clique partitioning based approach to NED. We used NE candidate confidence scores to weight the different clique coherences by summing all NE candidate confidence scores. A possible extension is to improve this coherency measure by considering the link weights, i.e. Jprob, in the weighting function.

- **Co-reference Resolution:**

  In the current work, the different textual mentions for the same named entity in the document are disambiguated separately and may have different features. A possible extension is to collect the co-referenced NE textual mentions together, combine their features, and use one candidate list for all.

- **Multiple Graph Ranking:**

  In the presented graph model approach (see chapter 5), we used Page-Rank to rank all possible candidates for all NE textual mentions in the document. The power of the graph model is considering the candidates of the other textual mentions when evaluating a specific NE candidate. The drawback of that approach is the sensitivity to wrong decisions. In other words, the wrong decision for one NE textual mention will affect the candidate selection for other NE textual mentions. It would be interesting to explore ways to identify the most accurate disambiguation for one of the NE textual mentions and then remove the other candidates for that NE textual mention and re-rank again, but using that accurate disambiguated entity. This will affect the other candidates' rankings.

- **Incorporating Entity Semantic Relations with the Graph Model:**

  The current work did not address the semantic relations between candidates of different textual mentions. Another possible avenue is to enrich the solution graph with semantic relations between different candidates. Different semantic relations between NE candidates of different textual mentions could be extracted from an ontology. These relations may be classified into direct and indirect relations. A direct relation like *birth place* may connect two NE candidates A and B, while another relation like *mother* connects NE candidates A and D; from this example we can note the differential importance of distinct relations. More research is needed to discover and evaluate this importance. Indirect relations between two entity candidates, A and B, occur when A has a relation to an intermediate NE candidate C, which in turn has a relation to NE candidate B. Machine learning techniques could be used to weight the different relations.

- **Combine Individual and Collective Approaches:**

  An individual disambiguation approach is presented in chapter 3 using NE document similarity based search. Two collective approaches are presented in chapters 4 and 5. Another avenue of future work is to combine both approaches by using our individual disambiguation approach to give some scores (like NEB-Sim and SVM-rank) to the candidates and use these as local features in the graph model.

- **Disambiguation by Finding the Densest Sub-graph:**

  We used Page-Rank to rank all graph nodes based on an initial confidence and link weighting. It could be useful to use the initial confidence and the link weights to find the densest sub-graph which contains one candidate from every NE textual mention candidate list.

Bibliography

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

T. Berners-Lee, J. Hendler, O. Lassila, et al. The semantic web. *Scientific american*, 284(5): 28–37, 2001.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 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, 2008. ACM.

Jordan L Boyd-Graber, David M Blei, and Xiaojin Zhu. A topic model for word sense disambiguation. In *EMNLP-CoNLL*, pages 1024–1033, 2007.

K.K.K. Breitman, M.A. Casanova, W. Truszkowski, United States. National Aeronautics, and Space Administration. *Semantic Web: Concepts*. NASA Monographs in Systems and Software Engineering Series. Springer-Verlag London Limited, 2007. ISBN 9781846287107.

Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1):107–117, 2000.

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

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

C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96. ACM, 2005.

Z. Cao, T. Qin, T.Y. Liu, M.F. Tsai, and H. Li. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th international conference on Machine learning*, pages 129–136. ACM, 2007.

A.X. Chang, V.I. Spitkovsky, E. Yeh, E. Agirre, and C.D. Manning. Stanford-UBC Entity Linking at TAC-KBP. In *Proc. TAC 2010 Workshop*, 2010.

Z. Chen and H. Ji. Collaborative ranking: A case study on entity linking. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 771–781. Association for Computational Linguistics, 2011.

D. Chopra, N. Jahan, and S. Morwal. Hindi named entity recognition by aggregating rule based heuristics and hidden markov model. *International Journal of Information*, 2(6), 2012.

Rudi L. Cilibrasi and Paul M.B. Vitanyi. The google similarity distance. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):370–383, 2007. ISSN 1041-4347.

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

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

Fang Du, Yueguo Chen, and Xiaoyong Du. Linking entities in unstructured texts with rdf knowledge bases. In Yoshiharu Ishikawa, Jianzhong Li, Wei Wang, Rui Zhang, and Wenjie Zhang, editors, *Web Technologies and Applications*, volume 7808 of *Lecture Notes in Computer Science*, pages 240–251. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-37400-5.

A. Ekbal, S. Mondal, and S. Bandyopadhyay. Pos tagging using hmm and rulebased chunking. *Proceedings of SPSAL-2007, IJCAI*, 7:25–28, 2007.

Robert J Elliott, Lakhdar Aggoun, and John B Moore. *Hidden Markov Models*. Springer, 1995.

Anthony Fader, Stephen Soderland, Oren Etzioni, and Turing Center. Scaling wikipedia-based named entity disambiguation to arbitrary web text. In *Proceedings of the IJCAI Workshop on User-contributed Knowledge and Artificial Intelligence: An Evolving Synergy, Pasadena, CA, USA*, pages 21–26, 2009.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

Yoav Freund, Raj Iyer, Robert E Schapire, and Yoram Singer. An efficient boosting algorithm for combining preferences. *The Journal of machine learning research*, 4:933–969, 2003.

A.L. Gentile, Z. Zhang, L. Xia, and J. Iria. Graph-based semantic relatedness for named entity disambiguation. 2009.

Zoubin Ghahramani. An introduction to hidden markov models and bayesian networks. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(01):9–42, 2001.

S. Gottipati and J. Jiang. 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, 2011.

Yuhang Guo, Wanxiang Che, Ting Liu, and Sheng Li. A graph-based method for entity linking. In *IJCNLP*, pages 1010–1018, 2011.

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

Yoan Gutiérrez, Sonia Vázquez, and Andrés Montoyo. 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, 2012.

Ben Hachey, Will Radford, and James R Curran. Graph-based named entity linking with wikipedia. In *Web Information System Engineering–WISE 2011*, pages 213–226. Springer, 2011.

X. Han and J. Zhao. Named entity disambiguation by leveraging wikipedia semantic knowledge. In *Proceeding of the 18th ACM conference on Information and knowledge management*, pages 215–224. ACM, 2009a.

Xianpei Han and Le Sun. A generative entity-mention model for linking entities with knowledge base. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 945–954. Association for Computational Linguistics, 2011.

Xianpei Han and Jun Zhao. Nlpr_kbp in tac 2009 kbp track: A two-stage method to entity linking. In *Proceedings of Test Analysis Conference 2009 (TAC 09)*. Test Analysis Conference (TAC), 2009b.

Xianpei Han, Le Sun, and Jun Zhao. 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, 2011.

F. Hasan, N. UzZaman, and M. Khan. Comparison of different pos tagging techniques (n-gram, hmm and brills tagger) for bangla. *Advances and Innovations in Systems, Computing Sciences and Software Engineering*, pages 121–126, 2007.

J. Hassell, B. Aleman-Meza, and I. Arpinar. Ontology-driven automatic entity disambiguation in unstructured text. *The Semantic Web-ISWC 2006*, pages 44–57, 2006.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 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, 2011.

Martin Jačala and Jozef Tvarožek. Named entity disambiguation based on explicit semantics. In *SOFSEM 2012: Theory and Practice of Computer Science*, pages 456–466. Springer, 2012.

T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142. ACM, 2002.

T. Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 217–226. ACM, 2006.

Joachim Kleb and Andreas Abecker. Entity reference resolution via spreading activation on rdf-graphs. In *The Semantic Web: Research and Applications*, pages 152–166. Springer, 2010.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. Adaptive Computation and Machine Learning Series. Mit Press, 2009.

S. Kulkarni, A. Singh, G. Ramakrishnan, and S. Chakrabarti. 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, 2009.

G. Kumaran and J. Allan. Text classification and named entities for new event detection. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 297–304. ACM, 2004.

John Lafferty and Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 111–119. ACM, 2001.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-778-1.

Amy N Langville and Carl D Meyer. A survey of eigenvector methods for web information retrieval. *SIAM review*, 47(1):135–161, 2005.

D. Lin. An information-theoretic definition of similarity. In *Proceedings of the 15th international conference on Machine Learning*, volume 1, pages 296–304. San Francisco, 1998.

C. Liu. Entity linking through neighborhood comparison and random walks. Technical report, Technical report, Johns Hopking University, 2009.

T.Y. Liu. *Learning to rank for information retrieval*, volume 13. Springer, 2011.

R.Duncan Luce and AlbertD. Perry. A method of matrix analysis of group structure. *Psychometrika*, 14(2):95–116, 1949. ISSN 0033-3123.

T. Mandl and C. Womser-Hacker. The effect of named entities on effectiveness in cross-language information retrieval evaluation. In *Proceedings of the 2005 ACM symposium on Applied computing*, pages 1059–1064. ACM, 2005.

Christopher D Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

P. McNamee, H.T. Dang, H. Simpson, P. Schone, and S. Strassel. An Evaluation of Technologies for Knowledge Base Population. In *Proceedings of the Seventh International Language Resources and Evaluation Conference*, 2010.

Paul McNamee and Hoa Trang Dang. Overview of the tac 2009 knowledge base population track. In *Text Analysis Conference (TAC)*, volume 17, pages 111–113, 2009.

Rada Mihalcea and Andras Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242. ACM, 2007.

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

David Milne and Ian H Witten. An effective, low-cost measure of semantic relatedness obtained from wikipedia links. In *Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence: an Evolving Synergy, AAAI Press, Chicago, USA*, pages 25–30, 2008b.

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

Roberto Navigli. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):10:1–10:69, 2009. ISSN 0360-0300.

Roberto Navigli and Mirella Lapata. Graph connectivity measures for unsupervised word sense disambiguation. In *IJCAI*, pages 1683–1688, 2007.

Kamel Nebhi. 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)*, 2013.

Hien T Nguyen and Tru H Cao. Named entity disambiguation: A hybrid approach. *International Journal of Computational Intelligence Systems*, 5(6):1052–1067, 2012.

Truc-Vien T Nguyen and Massimo Poesio. Entity disambiguation and linking over queries using encyclopedic knowledge. In *Proceedings of the 6th Workshop on Analytics for Noisy Unstructured Text Data. AND*, 2012.

S. Ono, I. Sato, M. Yoshida, and H. Nakagawa. Person name disambiguation in web pages using social network, compound words and latent topics. *Advances in Knowledge Discovery and Data Mining*, pages 260–271, 2008.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999. Previous number = SIDL-WP-1999-0120.

D. Petkova and W.B. Croft. Proximity-based document representation for named entity retrieval. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 731–740. ACM, 2007.

S.P. Ponzetto and R. Navigli. Knowledge-rich Word Sense Disambiguation rivaling supervised systems. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1522–1531. Association for Computational Linguistics, 2010.

Delip Rao, Paul McNamee, and Mark Dredze. Entity linking: Finding extracted entities in a knowledge base. In *Multi-source, Multilingual Information Extraction and Summarization*, Theory and Applications of Natural Language Processing, pages 93–115. Springer Berlin Heidelberg, 2013.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1375–1384. Association for Computational Linguistics, 2011.

Kranthi Reddy, Karuna Kumar, Sai Krishna, Prasad Pingali, Vasudeva Varma Location Iai, and Romania. Linking Named Entities to a Structured Knowledge Base. In *Proceedings of 11th International Conference on Intelligent Text Processing and Computational Linguistics*, 2010.

G. Salton, A. Wong, and C.S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

W. Shen, J. Wang, P. Luo, and M. Wang. Linden: linking named entities with knowledge base via semantic knowledge. In *Proceedings of the 21st international conference on World Wide Web*, pages 449–458. ACM, 2012.

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

H. Simpson, S. Strassel, R. Parker, and P. McNamee. Wikipedia and the Web of Confusable Entities: Experience from Entity Linking Query Creation for TAC 2009 Knowledge Base Population. In *7th Conference on Language Resources and Evaluation (LREC 2010), Malta*, 2010.

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

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A Core of Semantic Knowledge. In *16th international World Wide Web conference (WWW 2007)*, New York, NY, USA, 2007. ACM Press.

J. Van Gael, A. Vlachos, and Z. Ghahramani. The infinite hmm for unsupervised pos tagging. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*, pages 678–687. Association for Computational Linguistics, 2009.

V. Varma, V. Bharat, S. Kovelamudi, P. Bysani, GSK Santosh, K. Kumar, and N. Maganti. IIIT Hyderabad at TAC 2009. In *Proceedings of Text Analysis Conference 2009 (TAC 09)*, 2009.

V. Varma, P. Bysani, K. Reddy, V.B. Reddy, S. Kovelamudi, S.R. Vaddepally, R. Nanduri, N. Kiran Kumar, S. Gsk, and P. Pingali. IIIT Hyderabad in Guided Summarization and Knowledge Base Population. In *Proceedings of Text Analysis Conference TAC 2010 Workshop*, 2010.

Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 271–278. ACM, 2007.

W. Zhang, J. Su, C.L. Tan, and W.T. Wang. Entity Linking Leveraging Automatically Generated Annotation. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling2010)*, pages 1290–1298, 2010.

Z. Zheng, F. Li, M. Huang, and X. Zhu. 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, 2010.

Yiping Zhou, Lan Nie, Omid Rouhani-Kalleh, Flavian Vasile, and Scott Gaffney. Resolving surface forms to wikipedia topics. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1335–1343. Association for Computational Linguistics, 2010.