UNIVERSITY OF CALIFORNIA
Santa Barbara

Connecting Text with Knowledge

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

by

Yang Li

Committee in Charge:

Professor Xifeng Yan, Chair

Professor Ambuj Singh

Professor Linda Petzold

September 2015

The Dissertation of
Yang Li is approved:

Professor Ambuj Singh

Professor Linda Petzold

Professor Xifeng Yan, Committee Chairperson

June 2015

Connecting Text with Knowledge

by

Yang Li

To my admired and beloved parents, from whom I have got unconditional love and endless support, and inherited the great curiosity and passion for knowledge.

Acknowledgements

First of all, I express my sincerest gratitude and appreciation to my advisor Professor Xifeng Yan for his continuous guidance, tremendous patience and endless encouragement through the past five years. Not only did he teach me how to conduct solid research, but he also guided me to become a more mature person in life. The mindset and philosophy I developed while working with him will be definitely beneficial for a lifetime.

I would also like to thank the members of my PhD committee, Professor Ambuj Singh and Professor Linda Petzold. During my PhD study, they have provided me a lot of insightful comments and suggestions that guide me to always think bigger and deeper.

I owe my gratitude to many collaborators who contributed a lot to this thesis and provided me tremendous guidance. The exciting discussions with Dr. Chi Wang sparked the idea of mining evidences for entity linking. The collaboration with Dr. Bo Zhao, Dr. Ariel Fuxman and Fangbo Tao at Microsoft Research leads to the promising work on enterprise entity extraction and acronym disambiguation. And the joint work with Dr. Peter Clark at Allen Institute for Artificial Intelligence greatly inspired me to explore the exciting field of knowledge acquisition and machine intelligence.

I feel very fortunate to meet my adorable labmates in Professor Yan's group. Particularly, I would like to acknowledge Shulong Tan, Fangqiu Han, Huan Sun and Shengqi Yang for insightful discussions and collaborations.

I really appreciate the help and support from all my friends. I would like to thank Wen Chen, Yang Lin and many other friends here at UCSB for their company and assistance during the past five years.

Finally, I would like to dedicate this dissertation to my parents. Their selfless love and care helped me overcome all the difficulties and encouraged me to be my personal best.

Curriculum Vitæ

Yang Li

Education

2010 – 2015 **Ph.D.**, Computer Science, University of California, Santa Barbara

2006 – 2010 **B.E.**, Computer Science, Zhejiang University, China

Selected Publications

Yang Li, Peter Clark, "Answering Elementary Science Questions by Constructing Coherent Scenes using Background Knowledge", in *Proc. of the 2015 Conference on Empirical Methods on Natural Language Processing (EMNLP'15)*

Fangbo Tao, Bo Zhao, Ariel Fuxman, **Yang Li**, Jiawei Han, "Leveraging Pattern Semantics for Extracting Entities in Enterprises", in *Proc. of the 24th International World Wide Web Conference (WWW'15)*

Huan Sun, Mudhakar Srivatsa, Shulong Tan, **Yang Li**, Lance Kaplan, Shu Tao, Xifeng Yan, "Analyzing Expert Behaviors in Collaborative Networks" , in *Proc. of the 20th SIGKDD Conference on Knowledge Discovery and Data Mining (KDD2014)*

Shulong Tan, **Yang Li**, Huan Sun, Ziyu Guan, Xifeng Yan, Jiajun Bu, Chun Chen, Xiaofei He, "Interpreting the Public Sentiment Variations on Twitter", in *Transactions on Knowledge and Data Engineering (TKDE2013)*

Yang Li, Chi Wang, Fangqiu Han, Jiawei Han, Dan Roth, Xifeng Yan, "Mining Evidences for Named Entity Disambiguation", in *Proc. of the 19th SIGKDD Conference on Knowledge Discovery and Data Mining (KDD2013)*

Yang Li, Pegah Kamousi, Fangqiu Han, Shengqi Yang, Xifeng Yan, Subhash Suri, "Memory Efficient Minimum Substring Partitioning", in *Proc. of the 39th International Conference on Very Large Databases (VLDB2013)*

Under Review

Yang Li, Shulong Tan, Huan Sun, Jiawei Han, Dan Roth, Xifeng Yan, "Named Entity Disambiguation for Linkless Knowledge Bases"

Yang Li, Bo Zhao, Ariel Fuxman, Fangbo Tao, "Artificial Intelligence or Asset Intelligence? Acronym Disambiguation for Enterprises"

Research Experience

2010.9 – 2015.6 Research Assistant, University of California, Santa Barbara

2015.1 – 2015.4 Research Intern, Allen Institute for Artificial Intelligence

2014.6 – 2014.9 Research Intern, Microsoft Research

2013.6 – 2013.9 Research Intern, Apple Inc.

Professional Services

Program Committee	CIKM 2015
External Reviewer	ICDM 2013; WAIM 2013; CIKM 2012; ICDM 2012

Awards and Honors

2013 VLDB Travel Fellowship Award

2010 – 2012 Citrix Go-To Fellowship Award

2010 Outstanding Graduates of ZJU

2009 He Zhijun Scholarship

2009 Tencent Technology Scholarship for Excellence

2009 Cross-disciplinary Scholarship in Science and Technology

2007 – 2009 First Prize of Excellent Undergraduate Scholarship

2007 – 2008 National Scholarship

Abstract

Connecting Text with Knowledge

Yang Li

Access to well-organized, precise knowledge is critical for many practical applications, such as Semantic Search, Reasoning and Question Answering. Real-world knowledge is often unstructured, noisy and embedded in texts. This inspires us to connect text with knowledge. Text can be seen as both the source and the destination of knowledge. In one direction, knowledge can be distilled from text. While in the other direction, knowledge can be leveraged to understand text. Therefore, connecting text with knowledge can benefit both knowledge harvesting and text understanding, and ultimately facilitate many other tasks. Connecting text with knowledge is challenging for several reasons. First, knowledge is implicit in text. Various kinds of signals need to be leveraged to distill knowledge out of noises. Second, text is inherently ambiguous. The same textual mention can have different meanings, depending on the contexts of its appearance. Finally, there is a gap existing between knowledge representation and text understanding/reasoning.

In this thesis, we propose to tackle the challenges from two perspectives: (1) For linking text with knowledge, we study the entity linking problem. It links text mentions to the corresponding entries in a reference knowledge base, so that

semantic information can be transferred from knowledge base to text, and then new knowledge can be harvested from text to complete the knowledge base. In this work we target for alleviating the limitations of using Wikipedia as the reference knowledge base. We design innovative models to mine evidences scattered in text corpus and leverage them to compensate the missing information in the reference knowledge base. (2) For leveraging knowledge for text understanding, we develop the "coherent scene extraction" algorithm to utilize background knowledge for filling in the implicit yet critical information in text and show its effectiveness in answering elementary science questions. All the methods proposed in this thesis are comprehensively evaluated on real-life data to demonstrate the power of connecting text with knowledge.

Contents

List of Tables **xiv**

List of Figures **xv**

1 Introduction **1**
1.1 Motivations 1
1.2 Contributions of the Thesis 5

2 Literature Synopsis **9**
2.1 Linking Entities to Wikipedia 9
2.2 Linking Entities to More Knowledge Bases 12
2.3 Acronym Meaning Discovery/Disambiguation 14
2.4 Elementary Science Question Answering 15
2.5 Related Methodologies 17

3 Mining Evidences for Named Entity Disambiguation **19**
3.1 Background and Preliminary Material 20
3.1.1 Problem Statement 26
3.2 Model & Algorithm 27
3.2.1 Intuitions Behind the Model 28
3.2.2 Model Details 29
3.2.3 Incremental Evidence Mining Algorithm 34
3.2.4 Inference Algorithm 35
3.2.4.1 Likelihood Functions 35
3.2.4.2 A Blocked and Collapsed Gibbs Sampler 37
3.2.4.3 Estimating Document Label 40
3.2.4.4 Estimating Label-Word Association 41
3.3 Experiments 41

3.3.1 Datasets . . . 42
3.3.2 Experiments Setup . . . 43
3.3.3 NED Accuracy & Robustness . . . 44
3.3.4 Effectiveness of Evidence Mining . . . 47
3.3.5 Impact of Evidence Mining Iterations . . . 49
3.4 Summary . . . 50

4 Named Entity Disambiguation with Linkless Knowledge Bases 52
4.1 Background and Preliminary Material . . . 53
4.1.1 Problem Statement . . . 58
4.2 The Evidence Mining Approach . . . 59
4.2.1 Documents . . . 60
4.2.2 Word Evidences . . . 61
4.2.3 LNED via Evidence Mining . . . 65
4.3 Mining Evidences . . . 65
4.3.1 Model Intuitions . . . 66
4.3.2 Model Details . . . 67
4.3.3 Inference Algorithm . . . 73
4.3.3.1 Likelihood Function . . . 73
4.3.3.2 Approximate Inference via Gibbs Sampling . . . 74
4.3.3.3 Estimating Document-Label Association . . . 76
4.3.3.4 Estimating Label-Word Association . . . 77
4.4 Ranking Referent Candidates . . . 77
4.5 Experiments . . . 78
4.5.1 Datasets . . . 79
4.5.2 Experiments Setup . . . 80
4.5.3 Effectiveness of Evidence Mining . . . 81
4.5.4 Comparison of Evidence Mining Methods . . . 83
4.5.5 End-to-end NED Accuracy . . . 83
4.5.6 Quality of Mined Evidences . . . 86
4.5.7 Impact of Surrounding Window Size . . . 87
4.6 Summary . . . 88

5 Acronym Disambiguation for Enterprises 91
5.1 Background and Preliminary Material . . . 93
5.1.1 Problem Statement . . . 98
5.2 Framework . . . 100
5.3 Acronym Meaning Mining . . . 102
5.3.1 Candidates Generation . . . 102
5.3.2 Popularity Calculation . . . 104

5.3.3 Candidates Deduplication . . . 106
5.3.4 Candidates Filtering . . . 107
5.3.5 Context Harvesting . . . 109
5.4 Meaning Candidate Ranking . . . 110
5.4.1 Candidate Ranking . . . 110
5.4.1.1 Training Data Generation . . . 111
5.4.1.2 Training Algorithm . . . 111
5.4.1.3 Features . . . 112
5.4.2 Confidence Estimation . . . 115
5.4.2.1 Training Data Generation . . . 117
5.4.2.2 Training Algorithm . . . 117
5.4.2.3 Features . . . 117
5.4.3 Final Selection . . . 118
5.4.4 Iterative Popularity Calculation . . . 119
5.5 Experiments . . . 121
5.5.1 Data . . . 121
5.5.1.1 Mining and Training Corpus . . . 121
5.5.1.2 Evaluation Datasets . . . 122
5.5.2 Compared Methods . . . 125
5.5.2.1 Baseline . . . 125
5.5.2.2 Ablations of Our System . . . 125
5.5.2.3 State-of-the-art EL Systems . . . 127
5.5.3 Evaluation Measures . . . 128
5.5.4 Quality of Mined Acronym/Meaning Pairs . . . 129
5.5.5 Disambiguation Performance . . . 129
5.5.6 Comparison with EL Systems . . . 132
5.6 Summary . . . 133

6 Answering Elementary Questions via Coherent Scene Extraction 135
6.1 Background and Preliminary Material . . . 136
6.1.1 Problem Statement . . . 139
6.2 Approach . . . 140
6.2.1 Question Analyzer . . . 140
6.2.2 Elaborator . . . 142
6.2.3 Scene Extractor . . . 143
6.2.4 Ranking Function . . . 144
6.3 Experiments . . . 145
6.3.1 Evaluation Datasets . . . 145
6.3.2 Baselines . . . 145
6.3.3 Experiments Setup . . . 147

6.3.4 Performance Comparison 147
6.3.5 Error Analysis 148
6.4 Summary 150

7 Conclusions and Future Directions **151**
7.1 Conclusions 151
7.2 Future Directions 153

Bibliography **156**

List of Tables

3.1 Mined Evidences for Michael I. Jordan, Michael B. Jordan, Owen Bieber, General Aircraft Hotspur and David Young Cameron. 48
3.2 Notations used in proposed model for MENED 51

4.1 Basic statistics of test datasets for LNED 79
4.2 Notations used in proposed model for LNED 90

5.1 Candidate Ranking Features 114
5.2 Confidence Estimation Features 116
5.3 Basic statistics of the evaluation datasets for acronym disambiguation 124
5.4 Mined meanings for sample acronyms 130

6.1 Basic statistics of the evaluation datasets 146
6.2 QA Accuracy Comparison 148

List of Figures

1.1 Our Solution for Connecting Text with Knowledge 4

3.1 Named Entity Disambiguation Example 22
3.2 Entity, Word (Evidence), and Document 28
3.3 Proposed Model for MENED 30
3.4 MENED vs. Wikifier vs. AIDA 45
3.5 Varying Evidence Mining Iterations for MENED 49

4.1 Snapshot of hyperlinks in the Wikipedia page of *Mitsubishi Eclipse* 54
4.2 Effects of hyperlinks in NED features 57
4.3 Illustration of mention *Michael Jordan*'s candidate documents and mention documents 60
4.4 Mimicing the effects of hyperlinks 62
4.5 Documents, Entities and Words 64
4.6 Proposed Model for LNED 68
4.7 LENS vs. Labeled-LDA and MENED 82
4.8 LENS vs. Wikifier vs. AIDA 84
4.9 Quality of Evidences: LENS vs. MENED 86
4.10 Varying Surrounding Window Size for LENS 88

5.1 Acronyms in Enterprises 93
5.2 Acronym Disambiguation Example 95
5.3 Framework 101
5.4 Candidates Deduplication Example 107
5.5 Distant Supervision Example 113
5.6 Conditional vs. Iterative 120
5.7 Ranking Performance 127
5.8 Effectiveness of Confidence Estimator and Final Selector 127

5.9 Comparison with EL Systems 133

6.1 Entity-centric QA vs. Elementary QA 137
6.2 Coherent Scene Examples 138
6.3 Framework 141

Chapter 1

Introduction

1.1 Motivations

Information overload is critical in today's era of big data. The data people can access today are bigger and richer than ever. Most such real-world data is unstructured, noisy, and often expressed in the form of text. For example, there are millions of books published all over the world, and there are billions of Web pages in the Internet. Despite the fact that text is everywhere, what really valuable to us is the knowledge inside them. Just as the famous author John Naisbitt said, "we are drowning in information and starving for knowledge".

Access to well-organized, precise knowledge is critical for many real-world applications, such as Semantic Search, Intelligent Assistant and Question Answering.

For example, if I ask an intelligent assistant to book me a ticket to visit UCSB, it has to leverage the knowledge that UCSB is a university located at Santa Barbara, and then book me a ticket to Santa Barbara. Since text is the main carrier of knowledge, it is natural to extract knowledge from text. For instance, we can construct a structured knowledge graph from the large volume of text data, where nodes are entities and edges encode relationships among entities. Such formal and structural representation of knowledge has the advantage of being easy to manage and reason with, which can greatly facilitate many tasks. To achieve this goal, knowledge bases such as DBpedia [4], Freebase [10] were manually constructed. However, due to the laborious, time consuming, and costly extracting and labeling process, these knowledge bases are often restricted by a very limited coverage. Recently, automatically constructed knowledge networks including YAGO [97], NELL [17], Reverb [35], have emerged. But unfortunately, they suffer from the problems of low coverage [97, 17], or poor quality [35]. How to automatically distill high-quality knowledge from unstructured and noisy text data remains an open research problem.

As we mentioned, text is an important carrier of knowledge. Yet on the other hand, knowledge is also critical for understanding text. For example, in the sentence [106] shown in Example 1, if we just utilize the syntactic structure to interpret it, there are two possible readings: (dog isA cat) and (dog isA animal).

If we know nothing about cat, dog, and animal, we cannot decide which one is more probable. However, if we are given some background knowledge like dog is a kind of carnivore and carnivore is a subcategory of animal, then we can correctly choose between the two possible readings.

Example 1. *It is harmful to some animals other than cats such as dogs.*

To summarize, text can be seen as both the source and the destination of knowledge. In one direction, knowledge can be distilled from text. While in the other direction, knowledge can be leveraged to understand text. Therefore, connecting text with knowledge can benefit both knowledge harvesting and text understanding, and ultimately facilitate many other tasks.

Despite the big attraction, connecting text with knowledge is not trivial. There are several challenges inside. First, knowledge is implicit in text. Various kinds of signals need to be leveraged to distill knowledge out of noises. Second, text is inherently ambiguous. The same textual mention can have different meanings, depending on the contexts of its appearance. Finally, there is a big gap existing between knowledge representation and text understanding/reasoning.

In this thesis, we investigate the problem of connecting text with knowledge and tackle the aforementioned challenges from two perspectives: (1) Enriching text with knowledge. We study the entity linking [57] problem. It links text mentions to the corresponding entries in a reference knowledge base, so that se-

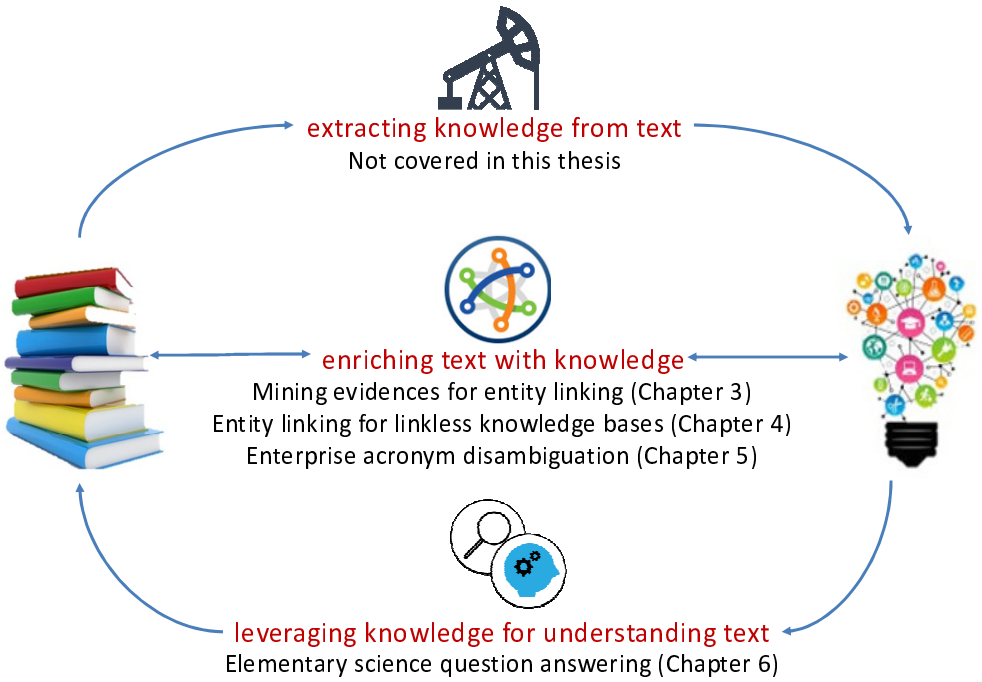


Figure 1.1: Our Solution for Connecting Text with Knowledge

mantic information can be transferred from knowledge base to text, and then new knowledge can be harvested from text to complete the knowledge base. In this work we target for alleviating the limitations of using Wikipedia as the reference knowledge base. We design innovative models to mine evidences scattered in text corpus and leverage them to compensate the missing information in the reference knowledge base. (2) Leveraging knowledge for understanding text. We study the elementary science question answering [23] problem. We develop the "coherent scene extraction" algorithm to utilize background knowledge for filling in the implicit yet critical information in question, and show how it helps for the ultimate

question answering. Figure 1.1 summarizes our solution for connecting text with knowledge.

1.2 Contributions of the Thesis

As aforementioned, we aim at connecting text with knowledge in a better and easier manner in this thesis. Towards this goal, we propose methods to mine hidden information scattered across noisy text data so that the requirements for the underlying reference knowledge bases in entity linking process can be minimized. We also investigated how to leverage background knowledge for filling the implicit information in elementary science question answering. We summarize the contributions of this thesis from the following perspectives.

Chapter 3 - Mining Evidences for Named Entity Disambiguation. We explore the topic of *named entity disambiguation* in linking text with knowledge base entries, by studying the problem of mining additional evidences from external corpus to bridge the information gap between text and reference knowledge bases. The contributions of this chapter include: (1) We introduce the task of *mining evidences for named entity disambiguation*, which is critical for overcoming the incompleteness of reference knowledge bases. (2) We develop an innovative generative model and a novel incremental algorithm for mining additional evidences to help boost the disambiguation performance. (3) Experimental results

show that our proposed method can mine additional evidences to significantly improve knowledge base's disambiguation ability. (4) Our work is useful to the work on developing new disambiguation algorithms and the mined evidences can be beneficial to any such algorithms.

Chapter 4 - Named Entity Disambiguation with Linkless Knowledge Bases. We notice that existing work on *named entity disambiguation* heavily rely on the cross-document hyperlinks within the knowledge base. But unfortunately, such hyperlinks are rarely available in many closed domain knowledge bases and it is very expensive to manually add such links. Therefore we study the challenging *named entity disambiguation with linkless knowledge bases* problem and tackle it by leveraging the useful disambiguation evidences scattered across the reference knowledge base. The contributions of this chapter include: (1) We introduce the challenging *named entity disambiguation with linkless knowledge bases* problem, which is critical in connecting text with knowledge for closed domain data. (2) We develop an innovative generative model for mining evidences to mimic the role of cross-document hyperlinks. (3) Experimental results show that our proposed method can harvest evidences to boost the disambiguation performance for linkless reference knowledge bases.

Chapter 5 - Acronym Disambiguation for Enterprises. We further explore the *named entity disambiguation* problem where there are no reference

knowledge bases being available. This setting further looses the underlying requirements for connecting text with knowledge, as it can be potentially applied to any domains. But on the other hand, it is much more challenging since the knowledge has to be directly mined and organized from plain text. As a first step towards this goal, we study the acronym disambiguation for enterprises problem. The contributions of this chapter include: (1) We target at the important problem of acronym disambiguation for enterprises, and are the first to identify its unique challenge: how to accurately resolve acronyms to their internal and external meanings. (2) We develop novel and practical algorithms so that our system can understand both the enterprise world and the public world. (3) We conduct a thorough experimental study on Microsoft enterprise corpus to justify the effectiveness of our system. (4) Our proposed framework can be easily deployed to any domains without requiring any domain knowledge.

Chapter 6 - Answering Elementary Questions via Coherent Scene Extraction We study the elementary science question answering problem and tackle it by extracting coherent scene from knowledge graph. The contribution of this chapter include: (1) We develop an end-to-end framework for answering elementary science questions. (2) We propose to extract coherent scene from knowledge graph and leverage the scene as a proper interpretation of question/answer

pair. (3) We conduct comprehensive experiments on real elementary science questions to justify the effectiveness of our method.

Chapter 2

Literature Synopsis

In this section, we give an overview of the existing literature for the problems studied in this thesis.

2.1 Linking Entities to Wikipedia

Named entity disambiguation [33] (abbreviated as NED, also known as entity linking [57]) has received a lot of attentions in recent years. Approaches that disambiguate named entity mentions with respect to Wikipedia date back to Bunescu and Pasca's work [14]. They defined a similarity measure to compute the cosine similarity between the text around the entity mention and the referent entity candidate's Wikipedia page. The referent entity with the maximum context similarity score is selected as the disambiguation result. Hoffart et al [52]

proposed a method to mine salient phrases for each entity, and then compare the overlap of such phrases for NED. They utilized "internal and external links" to harvest phrases and relies on links among entity pages for measuring salience. Several subsequent work incorporated more information into similarity comparison: Gottipati and Jiang [43] explored query expansion, while Zhang and Sim [111] considered acronym expansion. To incorporate different types of disambiguation knowledge together, Han and Sun [46] proposed a generative model to include evidences from entity popularity, mention-entity association and context similarity in a holistic way. And to overcome the deficiency of the bag of words model, Sen [88] adopted a latent topic model to learn the context-entity association to help disambiguation. Cucerzan's work [30] is the first one to realize the effectiveness of using topical coherence to help named entity disambiguation. In that work, the topical coherence between the referent entity candidate and other entities within the same context is calculated based on their overlaps in categories and incoming links in Wikipedia. Milne and Witten [69] refined Cucerzan's work by defining topical coherence using Normalized Google Distance [22] and only using "unambiguous entities" in the context to calculate topical coherence. Several new measures of topical coherence were also proposed in recent years: Bhattacharya and Gatoor [7] modeled the topical coherence as the association of an entity and the latent topics of a document, and Sen [88] modeled the topical coherence using

the co-occurrence of entities. Recently, several methods [48, 51, 85, 89, 111] also tried to combine together "context similarity" and "topical coherence" using a hybrid strategy which could further improve disambiguation accuracy. A comprehensive survey of the main approaches for NED or Entity Linking can be found at [91].

Almost all these previous NED algorithms fall into the scope of "single document NED". Their disambiguation decisions depend on the comparison (both textual and topical) of the query document (named entity mention along with its context) and the referent entity candidates' Wikipedia pages. Therefore they cannot handle the cases where there are no enough overlaps between the compared documents. To solve this problem, Chen and Ji [20] proposed the "Collaborative Ranking" technique. In their work, they used document clustering to find several "query collaborators" (documents which are in the same cluster with the query document) and ran existing NED algorithms on each "query collaborator" separately. Their disambiguation results were then assembled together to make the final decision. If most of a query's collaborators exhibit enough overlap with referent entity candidates' Wikipedia pages, the final disambiguation decision will likely be reasonable. Han and Sun [47] tackled this problem in another way. They proposed a generative entity-topic model which can jointly model context compatibility, topical coherence and their correlations. Since their model was trained

on all Wikipedia pages, it made use of not only the contents of referent entity candidates' Wikipedia pages, but also the contents of the Wikipedia pages where those referent entity candidates appear. Compared with "single document NED" algorithms, their method utilized cross-document information. However, both [20] and [47] cannot work well in cases where a query's context information does not exist in the entire Wikipedia corpus. In this thesis, we investigate such cases and explores information both inside and outside Wikipedia to mine additional evidences for named entity disambiguation.

2.2 Linking Entities to More Knowledge Bases

Almost all these previous NED algorithms use Wikipedia as the reference knowledge base. However, most not well known or domain specific entities are not captured by Wikipedia. To solve this problem, Sil et. al [92] proposed the Open-DB NED problem, which is to resolve an entity to any relational database that meets mild conditions about data format. They investigated a distant supervision approach and a domain adaptation approach to leverage the structural information in the reference relational databases. Their experiments on the movie and sports domain demonstrated their method's effectiveness. Similarly, Zheng et. al [112] studied disambiguating entity mentions to Freebase, Jin et. al [58] investigated linking entity mentions to a people profile database and Pantel et.al [79]

addressed the task of associating Web search queries with entities from a product catalog. Recently, Shen et. al [90] proposed a probabilistic model to link entity mentions in Web text to DBLP bibliographic network. All these work used some schema-rich databases or networks as the reference knowledge bases, and made use of the structural information to help perform disambiguation. Unfortunately, none of these approaches can be applied to linkless reference knowledge bases which are comprised of a set of noisy, unstructured and isolated text documents. There are no schemas or structural links inside them. Actually most of the closed-domain knowledge bases are linkless and some domain specific mentions can only be resolved to them. In order to assure high performance for connecting text with knowledge in these closed-domains, we study the *named entity disambiguation with linkless knowledge bases* problem in this thesis. Instead of directly applying the information provided by the relational databases, we propose method to mine useful disambiguation evidences from the knowledge base and use them to bridge the information gap caused by the absence of cross-document links. Our focus is also different from Cai et. al [16]'s work on link enrichment for named entity disambiguation. In their work, the goal is to add more cross-document links to Wikipedia via using the co-occurrence of the existing links. While in our study, none of the existing links are available and we have to mine evidences completely out of the linkless documents.

2.3 Acronym Meaning Discovery/Disambiguation

Acronym meaning discovery has received a lot of attentions in vertical domains (mainly in biomedical). Most of the proposed approaches [2, 3, 80, 87, 105] attempted to use generic rules or text patterns (e.g. brackets, colons) to discover acronym meanings. For example, Schwartz et. al [87] developed a simple algorithm combining patterns and rules for acronym meaning identification in biomedical text and achieved good performances. These methods are usually based on the assumption that acronyms are co-mentioned with the corresponding meanings in the same document. However, in some closed-domains like enterprises, this assumption rarely holds. Enterprises themselves are closed ecosystems, so it is very common for people to define the acronyms somewhere and use them elsewhere. As a result, such methods cannot be used for acronym meaning discovery in enterprises.

Recently, there have been a few works [55, 62, 74, 101] on automatically mining acronym meanings by leveraging Web data. For example, Jain et. al [55] explored subsequent queries in query sessions, while Taneva et. al [101] proposed to utilize "co-clicks" in search engine query click log to mine acronym meanings. These approaches can successfully harvest huge amount of acronym/meaning pairs from Web with reasonable precision and recall. However, it is hard to apply them to

some closed-domains like enterprises, since most data in enterprises are raw text and the query sessions/logs are rarely available.

Most of the previous work [38, 77, 83, 96, 109] on acronym disambiguation heavily rely on context words and domain specific resources. For example, Pakhomov et. al [77] barely used words occurring in the same sentence with the acronym as disambiguation features, while Stevenson et. al [96] utilized the Medical Subject Headings feature, which is specific to the biomedical domain. Another particular limitation of all these previous work is that they do not distinguish internal and external meanings. They merely rely on the internal corpus to discover information about external meanings, which is quite ineffective.

In this thesis, we study the *acronym disambiguation for enterprises* problem. We propose a general framework to perform high-quality acronym meaning discovery and disambiguation, without requiring any domain knowledge. Our method also leverages public resources together with the internal corpus to better understand the acronym semantics and therefore is capable of resolving acronyms to both domain-specific meanings and public meanings.

2.4 Elementary Science Question Answering

Question Answering (QA) has been extensively studied in the past few years. Existing methods for entity-centric factoid QA can be roughly divided into two

categories: KB-based methods and Web-based methods. KB-based methods [108, 113, 102, 36, 6] try to answer questions by directly query curated knowledge bases such as Freebase. While Web-based methods [13, 40, 61, 104] focus on mining answers from document collections or the rich web corpus, using different kinds of information retrieval techniques. All these methods are designed for entity-centric QA and therefore cannot work well for elementary science QA [23]. Compared with entity-centric QA, elementary science QA are mainly about general concepts and its answer lookup involves more implicit knowledge and logical reasoning. Clark et. al [26] analyzed the knowledge requirements for passing an elementary science test. Few methods [56, 24] were also proposed to tackle the more challenging elementary science QA problem. For example, Jansen et. al [56] combined language models (likelihood of answer option given question) and information retrieval scores (score of top retrieved sentence matching question plus option) using an SVM ranker, and Clark et. al [24] proposed to "prove" answer option from question by applying lexical inference rules automatically extracted from science texts. In this thesis, we specifically focus on the challenge that many elementary science questions require implicit knowledge to infer the answer, and propose a graph-based method for filling the knowledge gap.

2.5 Related Methodologies

The proposed evidence mining models in this thesis are inherited from the Latent Dirichlet Allocation (LDA) model. LDA was first proposed by Blei et. al [8] for finding the document-topic association and the topic-word association in text documents. Ramage et. al [84] extended LDA to Labeled-LDA so that each document can have multiple labels and the label-word correspondences can be inferred. Different from both LDA and Labeled-LDA, our models are particularly designed for the evidence mining purposes and they have different generative processes for different types of documents. To prevent the learnt topics being pervaded, most LDA-based models require a preprocessing step to remove stopwords. As many stopwords are domain-dependent and difficult to be pre-defined, several previous work [18, 44] introduced a special "background" topic and assumed all stopwords being generated by this background distribution. In our evidence mining procedures, there are far more types of noises than just stopwords. Therefore, our models incorporated several special topics to capture different types of noisy words which are less useful for our disambiguation purposes. With the help of these special topics, our models can successfully dig out the real statistically interesting evidences (e.g. word patterns) for each entity candidate, thus achieving good disambiguation performances.

The distant supervised learning technique used in this thesis is originally inspired by Craven et. al's work [29] on leveraging existing database records to automatically label training examples for constructing biological knowledge bases. Since distant supervision allows for cheap creation of large amounts of training data, it has recently been extensively studied for many applications. For example, in relation extraction [53, 71, 98, 100, 107, 86, 70, 37], researchers leverage the manually crowd-sourced relation tuples in Wikipedia's infobox and Freebase to automatically label large amount of training sentences. And in sentiment analysis, researchers utilize the manually created star ratings [9, 78], emoticons [42, 82] and hashtags [99] to distantly label plenty of training instances. In all these previous works, the training examples are distantly generated from manually created records. Different from them, in this thesis we distantly generate training examples from automatically extracted pairs. Since the automatic extractions are very likely to introduce noises or errors, we have to apply some constraints and filterings before matching the extracted records to plain text.

Chapter 3

Mining Evidences for Named Entity Disambiguation

In this chapter, we explore an interesting problem in connecting text with knowledge: named entity disambiguation. Named entity disambiguation is the task of disambiguating named entity mentions in natural language text to their corresponding entries in a knowledge base such as Wikipedia. Such disambiguation can help enhance readability and add semantics to plain text. It is also a central step in constructing high-quality information network or knowledge graph from unstructured text. Previous research has tackled this problem by making use of various textual and structural features from a knowledge base. Most of the proposed algorithms assume that a knowledge base can provide enough explicit

and useful information to help disambiguate a mention to the right entity. However, the existing knowledge bases are rarely complete (likely will never be), thus leading to poor performance on short queries with not well-known contexts. In such cases, we need to collect additional evidences scattered in internal and external corpus to augment the knowledge bases and enhance their disambiguation power. In this work, we propose a generative model and an incremental algorithm to automatically mine useful evidences across documents. With a specific modeling of "background topic" and "unknown entities", our model is able to harvest useful evidences out of noisy information. Experimental results show that our proposed method outperforms the state-of-the-art approaches significantly: boosting the disambiguation accuracy from 43% (baseline) to 86% on short queries derived from tweets. The work in this chapter is published in [65].

3.1 Background and Preliminary Material

An important component in connecting text with knowledge is *named entity disambiguation (NED)*. Given the named entity mentions in unstructured text data, the goal of NED is to map them to their corresponding real world entities in a knowledge base such as Wikipedia. Different from *entity resolution (ER)*, whose goal is to cluster entity mentions into several disjoint groups with each group representing a unique entity, NED requires explicitly identifying which un-

derlying entity a given named entity mention should refer to. The NED task is challenging due to the fact that many named entity mentions are ambiguous: the same mention can refer to various different real world entities when they appear in different contexts. For example, "Michael Jordan" can refer to the basketball star in NBA, the Machine Learning researcher in Berkeley or some other people. NED plays a critical role in high-quality information network construction. When new information extracted from text data is ready to be inserted into the network, it is necessary to know which real world entity this piece of information should be associated with. If the system makes a wrong decision here, the network will not only lose some information, but also introduce errors. For example, as shown in Figure 3.1, if the extracted information *"elected as AAAI fellow"* is wrongly associated with the basketball player *Michael Jordan*, the network will lose the information that *Michael Jordan (Machine Learning)* is an AAAI fellow, as well as wrongly including *Michael Jordan (Basketball Player)* as a fellow of AAAI.

In recent years, the NED task has received a lot of research interests. Many methods [30, 33, 47, 48, 51, 60, 85, 88, 89, 111] have been proposed to disambiguate named entity mentions in free text with respect to Wikipedia. Generally speaking, three kinds of features are explored by those methods. The first one is a statistical feature called *entity popularity*. It is based on the assumption that the most prominent entity for a given entity mention is the most probable underlying entity

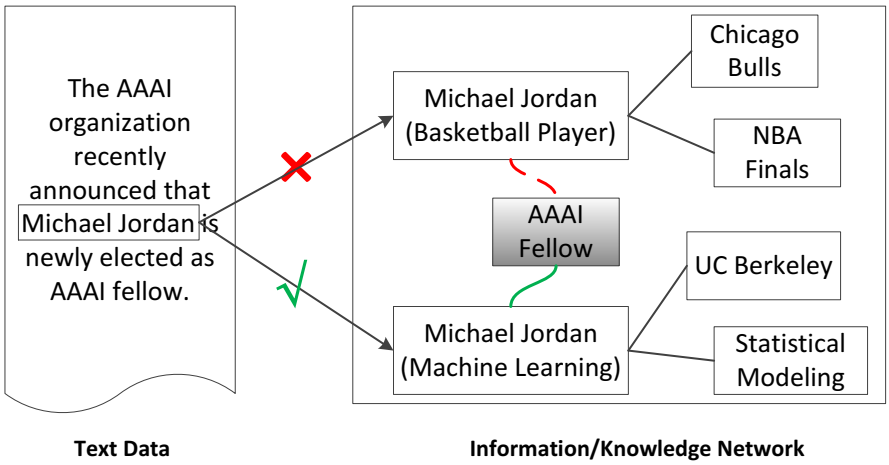


Figure 3.1: Named Entity Disambiguation Example

for that mention. Usually the "most prominent" entity is defined as the entity which uses the mention most frequently as a hyperlink anchor text in Wikipedia. Previous study [85] has shown that this simple heuristic is a very reliable indicator of the correct disambiguation. But obviously, methods merely depending on this feature are not robust as they will disambiguate all appearances of an entity mention to a fixed entity, regardless of the context along with it.

The second feature is a textual feature called *context similarity*. It takes the entity mention's context into consideration and defines similarity measures between the text around the entity mention and the document describing the referent entity in Wikipedia. This feature complements the *entity popularity* prior and is widely used in almost every method. One problem with *context similarity* is that it requires exact word overlap between the two compared texts, which

may become an over-strict constraint due to natural language's usage flexibility. To handle this problem, the third feature "topical coherence" is proposed. This feature is a structural feature making use of Wikipedia's cross-page links to define two entities' topical coherence. The intuition for using this feature is that the mention's referent entity should be topical coherent with other entities within the same context. Previous study [85] has proved the effectiveness of using this feature. Recently, several methods [48, 51, 85, 89, 111] also tried to combine together all these three features using a hybrid strategy which can further improve the accuracy.

Almost all of the previously proposed algorithms assume that the knowledge base can provide enough explicit and useful information to help disambiguate a mention to the right entity. However, in many situations, the information contained in the knowledge base is insufficient, thus leading to a connection gap between the keywords in a query (a named entity mention along with its context) and the knowledge base. Note that such situations are not rare since the reference knowledge base (e.g. Wikipedia) has a limited coverage and therefore cannot capture every aspect of a referent entity. In these cases, the existing state-of-the-art methods will fail to make correct disambiguation decisions because there is not enough information for them to utilize. The following two examples show the cases where key evidences (*"Eric Xing"* and *"paper"*, respectively) are not avail-

able (*Example 2*) in the knowledge base or overwhelmed (*Example 3*) by other evidences (*"won, best, award"*).

Example 2. *Eric Xing worked with Michael Jordan from 1999 to 2004.*

Example 3. *Michael Jordan won the best paper award.*

To solve the above problem, we need to collect additional evidences scattered in internal and external corpus to augment the knowledge base and enhance its disambiguation power. Mining additional evidences is an effective method for improving NED performance because it helps address at least two types of failures in existing approaches:

1. *No evidence failure*, i.e. the knowledge base does not cover the information contained in the query. Evidence mining helps by adding that information into the knowledge base. For instance, as shown in Example 2, the knowledge base contains no information about "Eric Xing," therefore the existing methods have no idea which entity the mention "Michael Jordan" should refer to. With the help of evidence mining, we can directly add "Eric Xing" as a supporting evidence of "Michael Jordan (Machine Learning)", via analyzing a large amount of documents outside the knowledge base, thus making the disambiguation an easy task.

2. *Insufficient evidence failure*, i.e. the important disambiguation evidences appear rarely in the knowledge base. Evidence mining again helps by increasing the weight of those evidences. For instance, as shown in Example 3, the most important disambiguation evidence here would be "paper". But since the occurrence of "paper" in "Michael Jordan (Machine Learning)" is not as frequent as the occurrences of "won", "best" and "award" in "Michael Jordan (Basketball Player)", the existing methods may wrongly disambiguate the mention "Michael Jordan" to the basketball player. With the help of evidence mining, we can give more weight to "paper" and thus avoiding such mistakes.

In this chapter, we aim at developing a method to automatically mine helpful evidences from internal and external corpus to boost the NED performance. Mining external evidences is much harder than mining internal ones, since internal documents in a knowledge base are well labeled and linked. Mentions in external documents are not disambiguated; yet it is still possible to extract new evidences from them, through our model. Our method can incrementally enrich the useful evidence set, making use of information both inside and outside the reference knowledge base. With a specific modeling of "background topic" and "unknown entities", our method can harvest helpful evidences out of noisy information. Experimental results show that our proposed method can mine additional evidences

to significantly improve knowledge base's disambiguation ability. Our work is also useful to the work on developing new NED algorithms and the mined evidences can be beneficial to any such algorithms.

3.1.1 Problem Statement

We formalize the *Named Entity Disambiguation (NED)* problem and our *Mining Evidences for Named Entity Disambiguation (MENED)* task as follows.

Definition 1 (Named Entity Disambiguation)**.** *Na-med Entity Disambiguation (NED) is the process of associating an entity name mentioned in a text to an entry, representing that entity, in a knowledge base (e.g. Wikipedia). Given a textual named entity mention m along with the unstructured text t in which it appears, and a reference knowledge base K, the goal is to produce a mapping from the mention m to its referent real world entity e in K.*

Definition 2 (Mining Evidences for NED)**.** *Mining Evidences for Named Entity Disambiguation (MENED) is the task of finding additional evidences inside and outside the knowledge base to improve NED accuracy. Given a textual named entity mention m, a reference knowledge base K, and a document corpus C outside K, the task is to mine additional evidences from K and C which can further help the disambiguation of m with respect to K.*

The MENED task is independent of the query context. For each named entity mention m, MENED is performed only once, regardless of different query contexts for the same mention. In practice the set of solvable ambiguous mentions can be pre-calculated from the knowledge base K (e.g. the whole set of entities indexed by K). Therefore the MENED process shall run **offline as a preprocessing step**. After MENED, any NED algorithm can make use of the evidences harvested by MENED to disambiguate m. In this work, a component in our MENED model can be reused to perform NED directly (Section 3.2.4.3).

3.2 Model & Algorithm

In this section we formally introduce our proposed model and algorithm, for mining new evidences to help named entity disambiguation. We will first describe the intuitions behind our model, then provide details about how the model is constructed and how the incremental algorithm works, and finally we will discuss how to perform inference on our model to estimate the document-label(entity) association and the label(entity)-word association.

3.2.1 Intuitions Behind the Model

We first describe the intuitions behind our model, before detailing the model in the next section. The goal of named entity disambiguation is to find a named entity mention's referent entity by utilizing the context along with the mention. The reason why context can help disambiguation is that each referent entity candidate can be distinguished by a set of representative words. Those representative words can be seen as the disambiguation evidences for those entity candidates. Therefore it is natural to model each entity as a topic/label and imagine those representative words are generated from such topics. Since we are only interested in those representative words which are highly related to the underlying entities, we model each entity mention's limited size context as a document. Each document can be associated with only one topic/label corresponding to its entity mention's real referent entity.

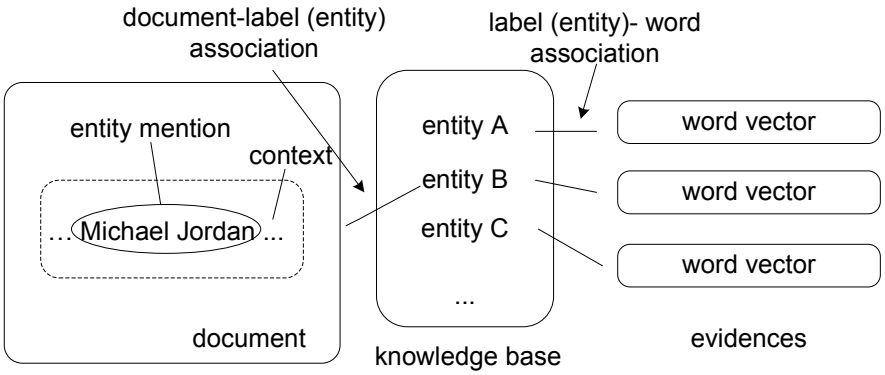


Figure 3.2: Entity, Word (Evidence), and Document

Though we have adopted the limited size context constraint to ensure topic centrality, some words within the context may still be general to some or all topics/labels. To specifically model this phenomenon, we introduce a special background topic to capture those non-representative words. On the other hand, we also notice that sometimes we may encounter documents whose underlying entities are not within the referent entity candidates. This is due to the fact that currently there is no perfect solution to generate complete referent entity candidates for a given entity mention. Obviously it is not appropriate to assign any topic/label to these documents. Therefore we introduce another special topic called "default" to capture words from the documents with unknown or unsure underlying entities. With all these intuitions, we are able to properly model the document-label association and the label-word association. Figure 3.2 shows the association among entities, words, and documents. Evidences are reflected by the words and their association strengths with entities, after discounting "background" and "default" topics. In the next section we will introduce our proposed generative model based on these intuitions.

3.2.2 Model Details

We now explain the details of our generative model. Figure 3.3 shows the graphical structure of dependencies of our model. Each node in the figure cor-

responds to a random variable or prior parameter. The shaded nodes represent observed variables while other nodes represent latent variables. A plate means the nodes within it are replicated for multiple times. A directed edge from node a to node b indicates that the variable represented by b is dependent on the the variable represented by a.

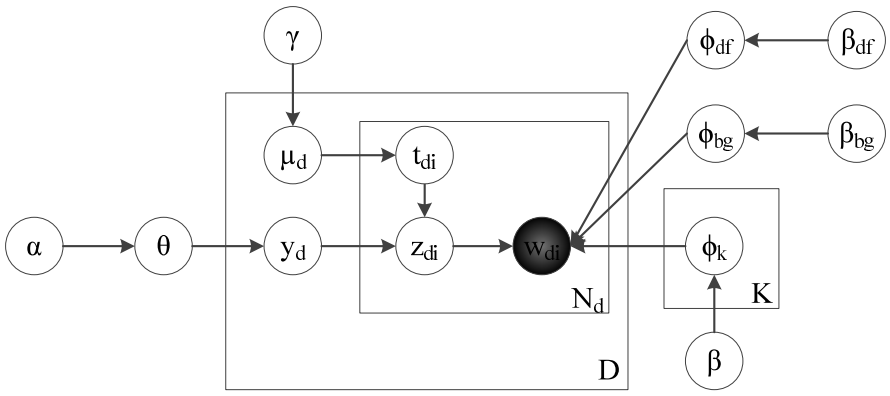


Figure 3.3: Proposed Model for MENED

Table 3.4 summarizes the notations used in our model. Given a named entity mention m, we will first generate all of its possible referent entity candidates. Following previous work [89, 85] on NED, we make use of the structural information of Wikipedia to find all the entities that m can be mapped to. Each referent entity candidate will be treated as a regular topic/label and the total number of them is K. We denote the set of regular topics/labels as S. For each occurrence of m, we model its limited size context (e.g. a width-W word window surrounding m) as a document. For a labeled document (e.g. the document in which the genuine

underlying entity for m is already identified), its document label y is fixed and within S. For an unlabeled document (e.g. the document in which the genuine underlying entity for m is not clear yet), its document label y is drawn from $S \cup$ "$default$", according to a multinomial distribution θ, which itself is drawn from a Dirichlet prior with α and α_{df} as the hyperparameters. As mentioned in the last section, "default" means the label for this document is unknown or unsure (in other words, not within S). The difference between α and α_{df} should reflect how conservatively we choose between regular topics/labels and the special "default" one.

Initially, all the documents inside the reference knowledge base (e.g. Wikipedia) are labeled documents, while all the documents in the external corpus are unlabeled documents. For each word w in both labeled and unlabeled documents, its label z is either the same as the label of the document in which it appears, or the special "background" label. The selection is controlled by an indicator variable t drawn from a Bernoulli distribution μ, which itself is drawn from a Beta prior with γ_1 and γ_2 as the hyperparameters. The difference between γ_1 and γ_2 should reflect the proportion of background topic. For each label in $S \cup$ "$default$" $\cup$ "$background$", it is associated with a multinomial distribution ϕ over words, which is drawn from the Dirichlet prior with β, β_{bg} and β_{df} as the hyperparameters. The difference among β, β_{bg} and β_{df} should reflect the content

difference among regular labels, the "default" label and the "background" label. Finally, each word w is drawn from the multinomial distribution ϕ_z, where z is the word label for w. **Our goal** is to infer the document-label association y and the label-word association ϕ from this model. The document-label association helps reveal the entity labels for unlabeled documents, and the label-word association helps demonstrate the disambiguation evidences for each referent entity candidate.

To summarize, the detailed generative process of our model is as follows:

1. Draw the multinomial distribution over words $\phi_k \sim \mathit{Dirichlet}(\beta)$ for each regular topic/label k.

2. Draw the multinomial distribution over words $\phi_{bg} \sim \mathit{Dirichlet}(\beta_{bg})$ for the background topic/label.

3. Draw the multinomial distribution over words $\phi_{df} \sim \mathit{Dirichlet}(\beta_{df})$ for the default topic/label.

4. Draw a topic/label distribution $\theta \sim \mathit{Dirichlet}(\boldsymbol{\alpha})$, where $\boldsymbol{\alpha} = (\alpha_{df}, \alpha_1, ..., \alpha_k)$ and $\alpha_1 = \ldots = \alpha_k = \alpha$.

5. For each document $d \in D$:

   (a) Choose a topic/label $y_d \sim \mathit{Multinomial}(\theta)$.

(b) Choose a background topic proportion $\mu_d \sim Beta(\gamma_1, \gamma_2)$.

(c) For each word position i in document d:

i. Choose a background indicator $t_{di} \sim Bernoulli(\mu_d)$.

ii. if $t_{di} = 0$:

A. Choose topic/label $z_{di} = bg$.

iii. else:

A. Choose topic/label $z_{di} = y_d$.

iv. Choose a word $w_{di} \sim Multinomial(\phi_{z_{di}})$.

Note that the above generative process is for unlabeled documents. For labeled documents, the document label y_d is known and fixed. Thus 5(a) becomes unnecessary and should be skipped. The other steps will remain the same.

Compared with regular topic models like LDA [8], our model is different in three aspects:

1. In our model, the regular topics, the "default" topic and the "background" topic may have multinomial distributions over words from different Dirichlet priors; while in LDA, the multinomial distributions over words are generated from the same Dirichlet prior.

2. In our model, each document has only one topic/label since we assume that the document is centered around the entity mention and the mention refers

to a single entity; while in LDA, each document is a mixture of different topics.

3. In our model, a word can only have two possible labels: foreground or background, and the foreground label is restricted by the document label; while in LDA, the word topic is generated directly from a multinomial distribution over topics.

3.2.3 Incremental Evidence Mining Algorithm

Now we will explain our incremental evidence mining algorithm based on the model introduced in the last section. As discussed in Sections 3.2.1 and 3.2.2, our model is able to infer both the document-label association and the label-word association (the inference details will be discussed in next section). So after a run of our model, each unlabeled document will be assigned a label with the maximum likelihood (Section 3.2.4.3), and the words associated with each label will change accordingly (Section 3.2.4.4). Each run of the model will bring in some new knowledge (e.g. more labeled documents and more comprehensive label-word correspondences) and those new knowledge can further help the model to find more additional evidences. So this is a typical incremental mining scenario. We thus introduce an incremental evidence mining algorithm, as described in Algorithm 1. Algorithm 1 will first do inference only for the labeled documents

of named entity mention m in reference knowledge base K (e.g. documents which contain mention m and a hyperlink to m's real referent entity). Then in each iteration, the algorithm will collect additional documents (D_{add}) from an external corpus C which have overlapped words with the current labeled documents (D_{i-1}). Inference will then be performed on current documents (D) and newly added documents (D_{add}) together, with a constraint that the labels of the current labeled documents (D_{i-1}) will remain unchanged. After inference, documents whose labels are found in the knowledge base will be added into the new labeled document set (D_i). The incremental process will continue until the iteration limit ($MaxIter$) is reached.

3.2.4 Inference Algorithm

3.2.4.1 Likelihood Functions

The joint likelihood is

$$p(\boldsymbol{w}, \boldsymbol{t}, \boldsymbol{y}, \boldsymbol{z} | \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \tag{3.1}$$
$$= \int_{\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\mu}} p(\boldsymbol{\theta}|\boldsymbol{\alpha}) p(\boldsymbol{\phi}|\boldsymbol{\beta}) p(\boldsymbol{\mu}|\boldsymbol{\gamma}) p(\boldsymbol{y}|\boldsymbol{\theta}) p(\boldsymbol{t}|\boldsymbol{\mu}) p(\boldsymbol{z}|\boldsymbol{y}, \boldsymbol{t}) p(\boldsymbol{w}|\boldsymbol{z}, \boldsymbol{\phi}) d\boldsymbol{\theta} d\boldsymbol{\phi} d\boldsymbol{\mu}$$

We use $\Gamma = \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}\}$ to denote the hyperparameters. We would like to calculate the posterior probability of $p(\boldsymbol{t}, \boldsymbol{y}, \boldsymbol{z} | \boldsymbol{w}, \Gamma)$ and use the maximal marginal probability to infer each topic assignment y_d and z_{di}. In typical topic models with conjugate

Algorithm 1 Incremental Evidence Mining

Input: Reference knowledge base K, external corpus C, named entity mention m, integer $MaxIter$.

$D_0 \leftarrow$ the set of labeled documents for mention m in K

$S \leftarrow$ the set of entity candidates' labels for mention m

$D \leftarrow D_0$

Do inference for $D = D_0$

for all i from 1 to $MaxIter$ **do**

$D_{add} \leftarrow$ the set of documents in $C - D$ which have overlapped words with D_{i-1}

$D \leftarrow D \cup D_{add}$

Do inference for D but fix labels for documents in D_{i-1}

$D_i \leftarrow$ the set of documents in D whose labels are in S

end for

prior such as LDA, one can apply collapsed Gibbs sampling to iteratively sample the variables z_{di}, t_{di}, y_d one by one, and estimate marginal probabilities with the samples. However, in our model, it is difficult to apply that sampling method due to the fact that every document has only one label y_d. In fact, when y_d and t_{di} are determined, z_{di} is uniquely decided as either bg or y_d. Therefore, if we sample y_d and z_{di} alternatively, once y_d is assigned some value fg, all the z_{di}'s associated with the corresponding document can only take values from $\{fg, bg\}$, and hence y_d will never be assigned any other value than fg because $p(y_d = l | z_{di} \in \{fg, bg\}) = 0$ for any $l \neq fg$. In other words, the Gibbs sampler will be trapped in a particular region $y_d = fg$ and never able to jump out of it.

To overcome that issue, we propose a blocked and collapsed Gibbs sampling algorithm with variational approximation.

3.2.4.2 A Blocked and Collapsed Gibbs Sampler

A blocked Gibbs sampler groups two or more variables together and samples from their joint distribution conditioned on all other variables, rather than sampling from each one individually. In our model, we group the variables $z_{d\cdot}, t_{d\cdot}$ and y_d for the same document together because of the aforementioned "deterministic trap" issue. In each blocked sampling stage, we sample the variables $z_{d\cdot}, t_{d\cdot}, y_d$ for one document d with all the other variables fixed as given.

Algorithm 2 Blocked Gibbs Sampling

for all $iter$ from 1 to $MaxIter$ **do**

for all $d \in D$ **do**

3: sample $\{z_{d\cdot}, t_{d\cdot}, y_d\}$ together according to $p(z_{d\cdot}, t_{d\cdot}, y_d|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}_{-d}, \Gamma)$

//call Algorithm 3

end for

end for

Now we explain how we sample $z_{d\cdot}, t_{d\cdot}, y_d$. First, we notice that z_{di} is determined by t_{di} and y_d, so we only need to sample $t_{d\cdot}$ and y_d according to $p(t_{d\cdot}, y_d|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}_{-d}, \Gamma)$. Second, based on chain rule of joint probability we have:

$$p(t_{d\cdot}, y_d|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}_{-d}, \Gamma) = p(y_d|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}_{-d}, \Gamma)$$

$$\prod_i p(t_{di}|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}, \Gamma, t_{d1} \dots t_{di-1})$$

So for a particular document d, we can first sample y_d and then sample t_{di} for each position i in d.

However, it is hard to compute $p(y_d|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}_{-d}, \Gamma)$ exactly because the parameters $\boldsymbol{\phi}$ and $\boldsymbol{\mu}$ are hard to be integrated out when marginalizing $p(w_{d\cdot}, t_{d\cdot}|\boldsymbol{w}_{-d}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}, \Gamma)$. To sample y_d with the advantage of collapsed sampler, we make a variational approximation

$$p(t_{d\cdot}|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}, \Gamma) = \prod_i \psi(t_{di}|w_{di}, \boldsymbol{w}_{-d}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}, \Gamma)$$

where $\psi(t_{di}|w_{di}, \boldsymbol{w}_{-d}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}, \Gamma)$ is a variational distribution of $p(t_{di}|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}, \Gamma)$ as if d has only one word w_{di}. In other words, we temporarily assume the labels $t_{d\cdot}$ in one document is conditionally independent given y_d and all variables in other documents. This approximation is reasonable because our documents are short but the number of documents is large. The conditional probability of $p(t_{d\cdot}|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}, \Gamma)$ changes little with this approximation but the calculation of $p(y_d|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}_{-d}, \Gamma)$ now becomes easy to accomplish.

$$
\begin{aligned}
p(y_d = l|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}_{-d}, \Gamma) \propto & \frac{\alpha_l + |\{y_{d'} = l, d \neq d'\}|}{\sum_{k=1}^{K} \alpha_k + \alpha_{df} + |D| - 1} \prod_{i=1}^{N_d} \\
& \left(\frac{\gamma_1 + |\{t_{d'j} = 0, d' \neq d\}|}{\gamma_1 + \gamma_2 + \sum_{d' \neq d} N_{d'}} \frac{\beta_{bg} + |\{t_{d'j} = 0, w_{d'j} = w_{di}, d' \neq d\}|}{|W|\beta_{bg} + |\{t_{d'j} = 0, d' \neq d\}|} \right. \\
& \left. + \frac{\gamma_2 + |\{t_{d'j} = 1, d' \neq d\}|}{\gamma_1 + \gamma_2 + \sum_{d' \neq d} N_{d'}} \frac{\beta_l + |\{z_{d'j} = l, w_{d'j} = w_{di}, d' \neq d\}|}{|W|\beta_l + |\{z_{d'j} = l, d' \neq d\}|} \right)
\end{aligned} \tag{3.2}
$$

After we sample y_d, we can sample t_{di} for each position i in d. If $y_d = default$,

$$
\begin{aligned}
\frac{p(t_{di} = 0|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}, \Gamma)}{p(t_{di} = 1|\boldsymbol{w}, \boldsymbol{z}_{-d}, \boldsymbol{t}_{-d}, \boldsymbol{y}, \Gamma)} = & \frac{\gamma_1 + |\{t_{d'j} = 0, d' \neq d\}|}{\gamma_2 + |\{t_{d'j} = 1, d' \neq d\}|} \\
& \cdot \frac{\beta_{bg} + |\{t_{d'j} = 0, w_{d'j} = w_{di}, d' \neq d\}|}{\beta_{df} + |\{z_{d'j} = y_d, w_{d'j} = w_{di}, d' \neq d\}|} \\
& \cdot \frac{|W|\beta_{df} + |\{z_{d'j} = y_d, d' \neq d\}|}{|W|\beta_{bg} + |\{t_{d'j} = 0, d' \neq d\}|}
\end{aligned} \tag{3.3}
$$

Otherwise replace the β_{df} in the formula with β. Finally, we have the following sampling steps for sampling one block.

Algorithm 3 Sampling for One Block (Line 3 in Algorithm 2)

Sample y_d according to Eq. (3.2)

for all i from 1 to N_d **do**

Sample t_{di} according to Eq. (3.3)

if $t_{di} = 0$ **then**

$z_{di} \leftarrow bg$

else

$z_{di} \leftarrow y_d$

end if

end for

3.2.4.3 Estimating Document Label

We infer the document label using maximal marginal probability with one exception: if the maximal marginal probability is smaller than a threshold η, we predict the label to be $default$. With this threshold we can control the noise by only labeling the documents on which our model has sufficiently high confidence.

$$y_d = \begin{cases} \arg\max_k p(y_d = k|\boldsymbol{w}, \Gamma) & \max_k p(y_d = k|\boldsymbol{w}, \Gamma) \geq \eta \\ default & \max_k p(y_d = k|\boldsymbol{w}, \Gamma) < \eta \end{cases} \tag{3.4}$$

Since our model can infer the document label for unlabeled document, it can also be directly used for named entity disambiguation if we treat the query as an unlabeled document.

3.2.4.4 Estimating Label-Word Association

We infer the label of each word in each document with maximal marginal probability:

$$t_{di} = \arg\max_{l \in \{0,1\}} p(t_{di} = l | \boldsymbol{w}, \Gamma) \tag{3.5}$$

$$z_{di} = \begin{cases} y_d & t_{di} = 1 \\ default & t_{di} = 0 \end{cases} \tag{3.6}$$

And the label-word distribution can be estimated by *maximum a posteriori* (MAP) inference:

$$\phi_k^{(v)} = \frac{\beta_k + |\{z_{di} = k, w_{di} = v\}|}{|W|\beta_k + |\{z_{di} = k\}|} \tag{3.7}$$

3.3 Experiments

In this section, we evaluate the effectiveness of our proposed method for the MENED task on two real-life datasets: one from news, and the other from Twitter. We will: (1) compare the disambiguation accuracy and robustness of our method, to two state-of-the-art NED methods that utilize various kinds of features; (2) analyze the effectiveness of the additional evidences mined by our method; (3) show how the performance of our method changes with respect to the number of the incremental evidence mining iterations. All the experiments, if not specifically

mentioned, are conducted on a server with 2.40GHz Intel Xeon CPU and 48GB RAM.

3.3.1 Datasets

Since our work is the first one to tackle the MENED problem, there is no established publicly available benchmark for us to test. As mentioned before, the goal of MENED is to bridge the potential information gap between a query and the reference knowledge base. Here we use two real-world datasets where such information gap indeed exists, to test the performance of our algorithm.

The first one is derived from the TAC-KBP2009 dataset, which is created for the Entity Linking task [57] in the Knowledge Base Population track at the Text Analysis Conference. The TAC-KBP2009 dataset consists of 3,904 queries (again, each query is an entity mention along with its context) and entity mentions in 1,675 of them can be linked to their corresponding entries in the knowledge base (Wikipedia). The fact that more than half of the entity mentions cannot find their underlying entities in the knowledge base also proves that the reference knowledge base is usually a limited information source and therefore it may lack some important information to help disambiguate named entity mentions. The original dataset contains many long news articles. In order to test the abilities of different algorithms in a challenging scenario where information gap is large, we modify

this dataset to keep only a fixed-size word window surrounding the query mention as its "context" (in this work, we choose the word window size as 60). By adding this constraint the information gap is enlarged and the disambiguation difficulty is increased. Among the 1,675 resolvable queries, we choose the queries whose named entity mentions have a corresponding Disambiguation Page in Wikipedia as our first test dataset. This dataset contains 424 queries.

Our second dataset is generated from Twitter. Since tweets have the 140-character constraint and the words used in them are often irregular, the probability of seeing information gap between tweets and the reference knowledge base is relatively high. Therefore running NED on tweets is much harder than on news. We randomly picked 25 ambiguous entities from the Wikipedia's Disambiguation Page Category and crawled 500 tweets containing these mentions as queries. After filtering out the queries which are unsolvable (e.g. even human beings cannot specify which entity the mention refers to), 340 queries are left and we treat them as our second test dataset.

3.3.2 Experiments Setup

In this work, we use Wikipedia as the reference knowledge base and the web-pages indexed by Google as the external corpus. For each reference entity candidate, we generate its labeled data (D_0 in Section 3.2.3) by utilizing its Wikipedia

page and all Wikipedia pages which have hyperlinks to its Wikipedia page. For fetching related documents (D_{add} in Section 3.2.3) from the external corpus, we make use of the Google Search API and collect the top 20 webpages for each referent entity candidate.

3.3.3 NED Accuracy & Robustness

We first conduct experiments to compare our method with two NED methods utilizing various kinds of features: **Wikifier** [85], a state-of-the-art NED system using a machine learning based hybrid strategy to combine popularity prior, context similarity and topical coherence features together, and **AIDA** [51], a robust NED system making use of weighted mention-entity graph to find the best joint mention-entity mapping. As explained in Section 3.2.4.3, our model for MENED can be directly used for NED if we treat the query as an unlabeled document. We test our model under two settings: (1) using the evidences mined from Wikipedia only; (2) using the evidences mined from both Wikipedia and the external corpus. We denote the first setting as **MENED(Wiki)** and the second setting as **MENED(All)**. For both MENED(Wiki) and MENED(All), we use the following parameter settings: $\alpha = 0.001, \alpha_{df} = 0.01, \beta = 0.001, \beta_{df} = 0.01, \beta_{bg} = 0.1, \gamma_1 = 0.0003, \gamma_2 = 0.001$. These parameters are tuned on a small test dataset containing 15 queries and then reused in all the experiments without any fur-

ther tuning. The threshold for predicting document label is chosen as $\eta = 0.9$. For MENED (All), we incrementally mine evidences from external corpus for 5 rounds. For Wikifier and AIDA, we use the parameter settings suggested by their authors. The same parameter settings were applied to both datasets. Wikifier used a Wikipedia repository of 2009[1]. Originally the Wikipedia repository used by AIDA is of 2010, later the authors kindly provided us an updated version which used a Wikipedia repository of late 2012. We denote the original one and the updated one as AIDA(2010) and AIDA(2012), respectively. Both MENED(Wiki) and MENED(All) rely on a Wikipedia repository of late 2012.

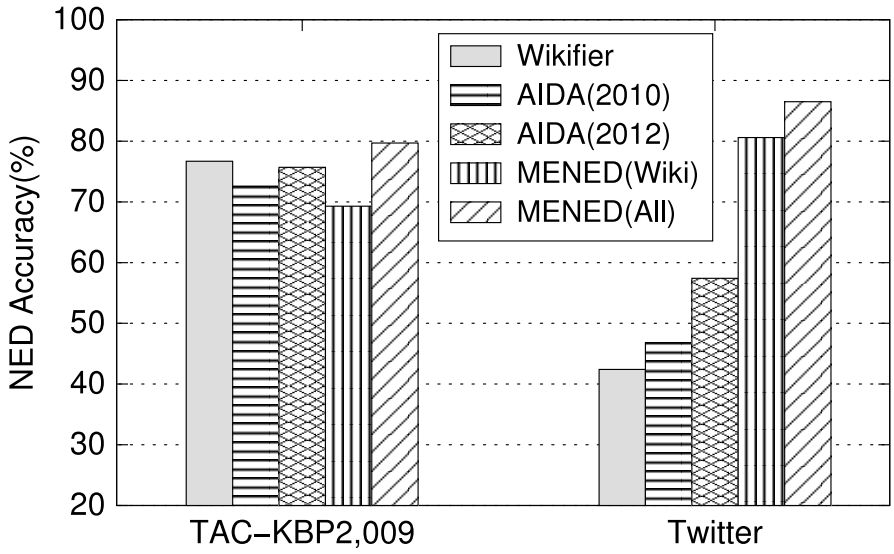


Figure 3.4: MENED vs. Wikifier vs. AIDA

[1]We were unable to configure Wikifier to utilize recent Wikipedia repository due to the hidden database schema. We were also unable to obtain a Wikipedia repository of 2009 to run our method with.

Figure 3.4 shows that MENED(All) slightly outperforms Wikifier and AIDA on TAC-KBP2009 dataset. Compared with Wikifier and AIDA, MENED(All) does not utilize any complicated features (e.g. topical coherence). On the Twitter dataset, MENED(All) performs remarkably better than Wikifier and AIDA. Both Wikifier and AIDA get very poor NED accuracy on short and noisy texts like tweets. MENED(All) retains high accuracy on tweets, indicating a much more robust performance. We notice that MENED(Wiki) also greatly outperforms Wikifier and AIDA on the Twitter dataset. This is due to two reasons. First, MENED(Wiki) mines new evidences from the Wikipedia pages that hyperlink to the entity candidates' own Wikipedia pages. Second, the topical coherence feature utilized by both Wikifier and AIDA is less helpful on very short texts like tweets since there are very few entities within the short context.

We also tried to compare our method with **TAGME** [39], an NED system specifically designed for very short texts like tweets. Different from Wikifier and AIDA, the TAGME API does not allow us to specify the named entity mentions to disambiguate. As a result, many queries in our test datasets are not properly responded. Without considering the queries which cannot be handled, TAGME obtains the NED accuracy of 78.3% and 61.1% on TAC-KBP2009 data and Twitter data respectively. Our method MENED(All) outperforms TAGME on both datasets.

3.3.4 Effectiveness of Evidence Mining

We then conduct experiments to demonstrate the effectiveness of mining evidences from external corpus. As can be seen from Figure 3.4, MENED(All) outperforms MENED(Wiki) in terms of NED accuracy on both datasets. The accuracy gain illustrates that our method for MENED is effective and the mined evidences from external corpus are indeed very helpful for boosting the NED performance.

Table 3.1 shows the mined evidences from external corpus for several entities. We can see that the mined evidences can provide complementary knowledge for disambiguating the entities, especially for those entities that are not very popular and therefore do not have many context information in Wikipedia. For example, in "Michael I. Jordan" case, the evidences "layers, nonparametric, nonlinear" correspond to his research work, "pehong, chen, distinguished" indicate the fact that he is a Pehong Chen Distinguished Professor at UC Berkeley, and "david, heckerman, kearns, marina, meila" describe his collaborators. All these evidences are not captured in Wikipedia but available in external sources (e.g. his homepage and DBLP page). Our model and algorithm can successfully dig out these useful evidences scattered across multiple documents in the external corpus.

Entity	Mined Additional Evidences
Michael I. Jordan (Michael Jordan)	layers, nonparametric, nonlinear, pehong, chen, distinguished, david, heckerman, kearns, marina, meila ...
Michael B. Jordan (Michael Jordan)	wood, oscar, role, peters, gilliard, detmer, larry, freamon, true-frost, pryzbylewski, octavia, spencer, troubled, ...
Owen Bieber (Bieber)	jobs, automobile, corporation, approved, presidential, lofton, support, vote, organizer, worley, conventions, worker ...
General Aircraft Hotspur (Hotspur)	operating, ground, states, cargo, aviation, capacity, built, fighter, targets, spitfire, flight, eben, paratroops ...
David Young Cameron (David Cameron)	engravers, technique, sculpture, printmaking, reproduced, scotch, lorne, muirhead, walton, french, nature, lovely ...

Table 3.1: Mined Evidences for Michael I. Jordan, Michael B. Jordan, Owen Bieber, General Aircraft Hotspur and David Young Cameron.

3.3.5 Impact of Evidence Mining Iterations

Next we conduct experiments to illustrate how the performance of our MENED method changes with respect to the number of incremental evidence mining iterations. Here the parameter settings are the same as those described in Section 3.3.3. Figure 3.5 shows that the NED accuracy increases as the number of iterations increases. But the increasing speed slows down as more evidences are collected.

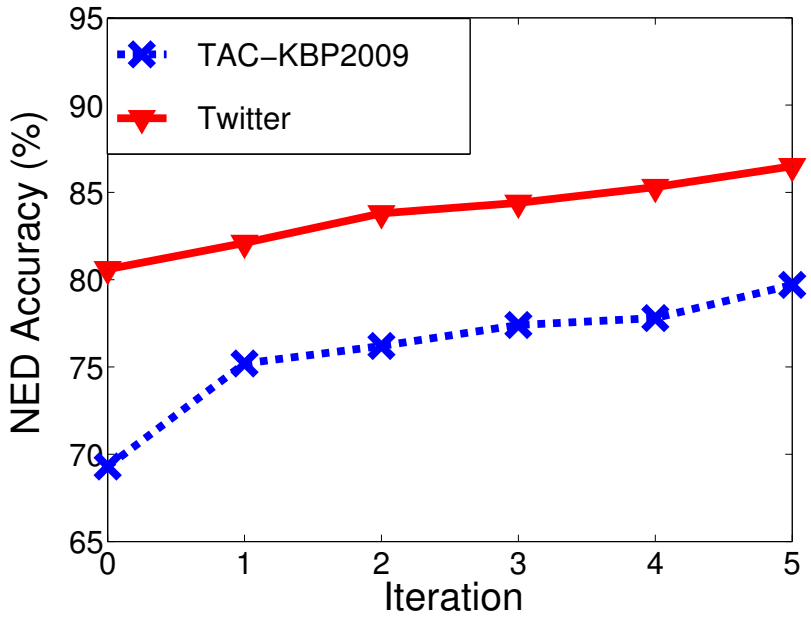


Figure 3.5: Varying Evidence Mining Iterations for MENED

3.4 Summary

In this chapter, we studied the problem of mining evidences for named entity disambiguation. We proposed a generative model and an incremental algorithm to automatically mine useful evidences across documents. With a specific modeling of "background topic" and "unknown entities", our model is able to harvest useful evidences from noisy text. To evaluate the effectiveness of our model and algorithm, a thorough experimental study was conducted. The experimental results demonstrated that our proposed method can mine additional evidences to significantly boost the disambiguation performance.

Symbols	Descriptions
D	the set of documents (e.g. named entity mention's limited size context)
K	the number of referent entity candidates
S	the set of regular entity labels
N_d	the number of words in document d
w_{di}	the i-th word of document d
z_{di}	the label associated with the i-th word of document d
y_d	the label associated with document d
t_{di}	the background indicator for the i-th word of document d
μ_d	the background topic proportion for document d
θ	the topic/label distribution
ϕ_{bg}	the word distribution for the background topic/label
ϕ_{df}	the word distribution for the default topic/label
ϕ_k	the word distribution for the k-th regular topic/label ($1 \leq k \leq K$)
α_{df}, α	the hyperparameters for Dirichlet prior of θ
β_{bg}	the hyperparameter for Dirichlet prior of ϕ_{bg}
β_{df}	the hyperparameter for Dirichlet prior of ϕ_{df}
β	the hyperparameters for Dirichlet prior of ϕ_k ($1 \leq k \leq K$)
γ	the hyperparameters for Beta prior of μ

Table 3.2: Notations used in proposed model for MENED

Chapter 4

Named Entity Disambiguation with Linkless Knowledge Bases

In this chapter, we explore a more challenging version of named entity disambiguation: disambiguating entity mentions to linkless knowledge bases. Previous research on named entity disambiguation (mainly respect to Wikipedia) has tackled the problem by making use of two types of context-aware features derived from the reference knowledge base, namely, the context similarity and the semantic relatedness. Both features heavily rely on the cross-document hyperlinks within the knowledge base: the semantic relatedness feature is directly measured via those hyperlinks, while the context similarity feature implicitly makes use of those hyperlinks to expand entity candidates' descriptions and then compares

them against the query context. Unfortunately, cross-document hyperlinks are rarely available in many closed domain knowledge bases and it is very expensive to manually add such links. Therefore few algorithms can work well on linkless knowledge bases. In this chapter, we propose the challenging Named Entity Disambiguation with Linkless Knowledge Bases (LNED) problem and tackle it by leveraging the useful disambiguation evidences scattered across the reference knowledge base. We propose a generative model to automatically mine such evidences out of noisy information. The mined evidences can mimic the role of the missing links and help boost the LNED performance. Experimental results show that our proposed method substantially improves the disambiguation accuracy over the baseline approaches.

4.1 Background and Preliminary Material

An important component in constructing information networks is *named entity disambiguation* (NED). Given the named entity mentions extracted from unstructured text data, the goal of NED is to map them to their corresponding real world entities in a reference knowledge base such as Wikipedia. NED has many key applications in text analysis and understanding, e.g., tweet tagging, text classification, and ad placement. Due to natural language's inherent ambiguities, the NED task is quite challenging. The same textual mention can represent multiple

different real world entities depending on the context of its appearance. For example, "Eclipse" can refer to the Java development platform, the car designed by Mitsubishi or even a breath freshener brand.

NED requires a reference knowledge base to serve as the real world entity collections to which the named entity mentions will be resolved. Usually the reference knowledge base is comprised of a set of documents with each document describing one specific entity. Almost all previous research on NED uses Wikipedia as the reference knowledge base. Despite the fact that Wikipedia covers millions of entities, most not well known or domain specific entities are not captured by Wikipedia. Therefore in closed domains (e.g. biomedicine, entertainment, enterprise, etc.) [32, 110], in order to achieve good NED performance, we have to rely on some domain specific knowledge bases (e.g. product catalog [79], restaurant directory [31], gene descriptions [73], etc.) as the reference knowledge bases.

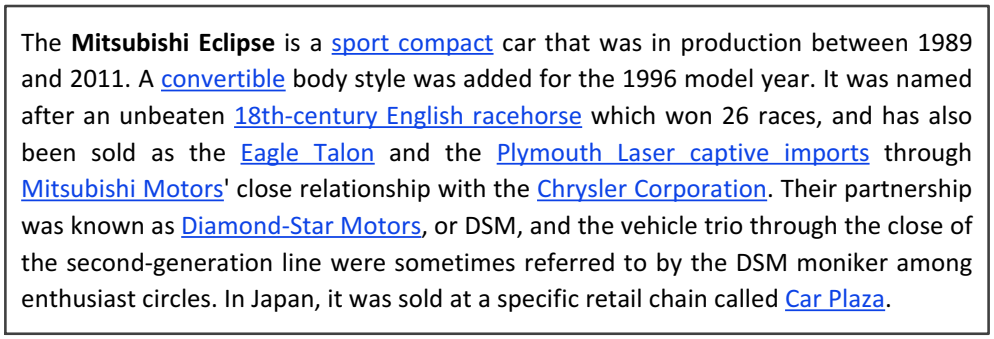


Figure 4.1: Snapshot of hyperlinks in the Wikipedia page of *Mitsubishi Eclipse*

One particular difference between Wikipedia and closed domain knowledge bases is the existence of cross-document hyperlinks. As the largest publicly available encyclopedia in the world, Wikipedia contains not only the description pages for millions of entities, but also the huge amount of cross-document hyperlinks connecting those entities. Those links are created by 20,694,972 [1] contributors from all around the world. Figure 4.1 shows a snapshot of such links in the Wikipedia page of entity *Mitsubishi Eclipse*. As we can see, 9 hyperlinks (denoted with underlines) are created to connect *Mitsubishi Eclipse* with 9 different entities in Wikipedia. On the contrary, we notice that most closed domain knowledge bases contain few cross-document hyperlinks. For example, the knowledge base used inside Microsoft covering all the projects/products/tools is linkless. The repository of crime cases in California Police Departments is also linkless. This is due to the fact that closed domain knowledge bases are usually created by a limited number of domain experts. It is very costly to manually cross link all the entity mentions.

Previous research [30, 33, 47, 48, 51, 60, 85, 88, 89, 111] has tackled the NED problem (with respect to Wikipedia) by making use of various textual and structural information from Wikipedia. Generally speaking, two kinds of context-aware features are explored by those methods. The first one is *context similarity*. It defines similarity measures (e.g. cosine similarity in word vector space) between the

context around the entity mention and the referent entity candidate's Wikipedia page. The referent entity with larger context similarity score is more likely to be the genuine disambiguation result. The second feature is *semantic relatedness*. It defines correlation measures between a mention's candidate entity and the unambiguous entities within the same context. The referent entity with a larger correlation score should get more preference.

We notice that both features used by previous research heavily rely on the cross-document hyperlinks in Wikipedia. Semantic relatedness is usually measured in terms of the common incoming hyperlinks to the entities' Wikipedia pages. For example, as shown in Figure 4.2(a), the referent entity *Michael I. Jordan* has a high semantic relatedness score with the entity *Andrew Ng*, since they share a lot of common incoming hyperlinks. Meanwhile the context similarity feature implicitly makes use of the hyperlinks to expand entity candidates' descriptions and then compare them against the mention's context. For instance, as shown in Figure 4.2(b), the referent entity *Michael I. Jordan*'s description is expanded by the surrounding context words of its anchor texts. Therefore if we want to perform NED with respect to linkless closed domain knowledge bases, none of these existing algorithms can work well.

In this chapter, we aim at solving the *Named Entity Disambiguation with Linkless Knowledge Bases* (LNED) problem. We develop a method to automatically

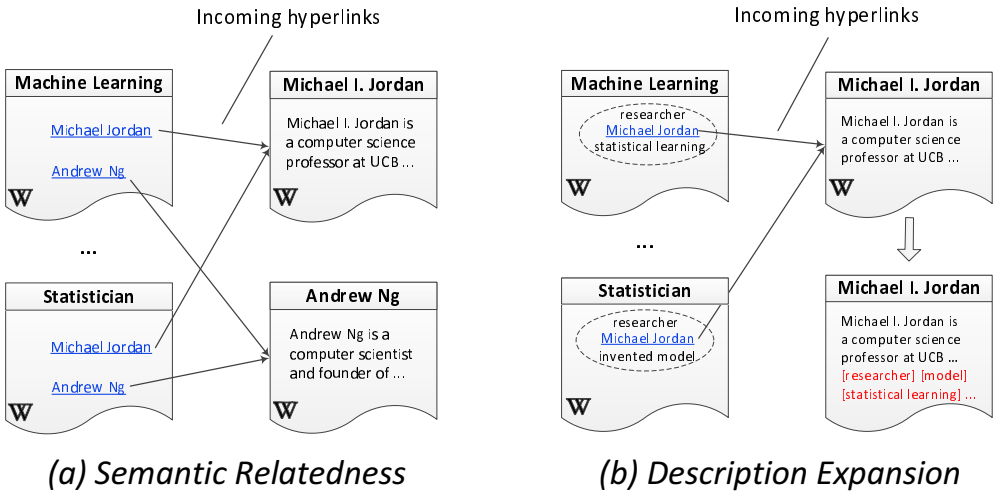


Figure 4.2: Effects of hyperlinks in NED features

mine helpful disambiguation evidences from the reference knowledge base which contains no cross-document hypelinks. The mined evidences can mimic the role of those links and boost the LNED performance. Mining evidences is not trivial, since without hyperlinks the only labeled data available are the entity candidates' own description pages. Mentions in other documents are not disambiguated; yet it is still possible to extract new evidences from them, through our model. Experimental results show that our proposed method can mine evidences to improve linkless knowledge base's disambiguation ability and substantially improve the disambiguation accuracy over the baseline approaches.

4.1.1 Problem Statement

We formalize the *Named Entity Disambiguation with Linkless Knowledge Bases* (LNED) problem as follows.

Definition 3 (Named Entity Disambiguation with Linkless Knowledge Bases)**.** *Named Entity Disambiguation with Linkless Knowledge Bases (LNED) is the process of associating an entity name mentioned in a text to an entry, representing that entity, in a "linkless" reference knowledge base K. K is comprised of a set of isolated documents D with each document $d \in D$ describing one entity e. There are no*[1] *cross-document or intra-document hyperlinks among the documents in D.*

In the Big Data and Big Knowledge age, more and more closed domain knowledge bases will emerge and most of them are likely to be linkless. Meanwhile, many domain-specific entity mentions can only be resolved to these knowledge bases. Therefore it is necessary and critical to study the LNED problem and find a good solution to it. In the next section we will describe our approach to tackle the LNED problem via evidence mining.

[1]In this work, we study a general setting with minimal requirements on the underlying knowledge bases. Our proposed method is also applicable to the knowledge bases with a few hyperlinks.

4.2 The Evidence Mining Approach

In order to solve the LNED problem, we have to figure out a way to bridge the information gap caused by the absence of cross-document hyperlinks. One straightforward solution is employing an existing NED algorithm (with link-based features removed) to recover the links. Namely, one can perform NED on mentions found in the reference knowledge base and use the disambiguation results to serve as the links. Such a method has a critical drawback. Without hyperlinks, none of the existing NED algorithms can achieve satisfactory results (see Section 4.5.3 and 4.5.5). Therefore, a large amount of the recovered links are likely to be incorrect and the features built on these "false links" will do harm to the ultimate disambiguation performance.

In this work, we propose to bridge the gap by collecting *word-level* disambiguation evidences scattered in the knowledge bases. Compared with the above link-recovery approach, mining fine-grained word evidences has the advantage of being more robust. We will jointly model mention's link destination (i.e. referent entity) and entity's supporting evidences (i.e. words) in a probabilistic manner. Instead of explicitly predicting the link, we aim at harvesting some useful word evidences from the mention context, through analyzing the word co-occurrence patterns.

4.2.1 Documents

The reference knowledge base K is comprised of a set of isolated documents (called *entity documents*), with each document describing one specific entity. For a given target mention m, all the possible entities it can refer to form a candidate entity set. Their main description documents are called *candidate documents*. Since the mention m may also appear in other documents, these additional documents are named as m's *mention documents*. Figure 4.3 illustrates the candidate documents and mention documents of "Michael Jordan". We aim at mining evidences jointly from these two kinds of documents, to disambiguate m's appearances in *query documents*. Below are the brief summaries of these three types of documents.

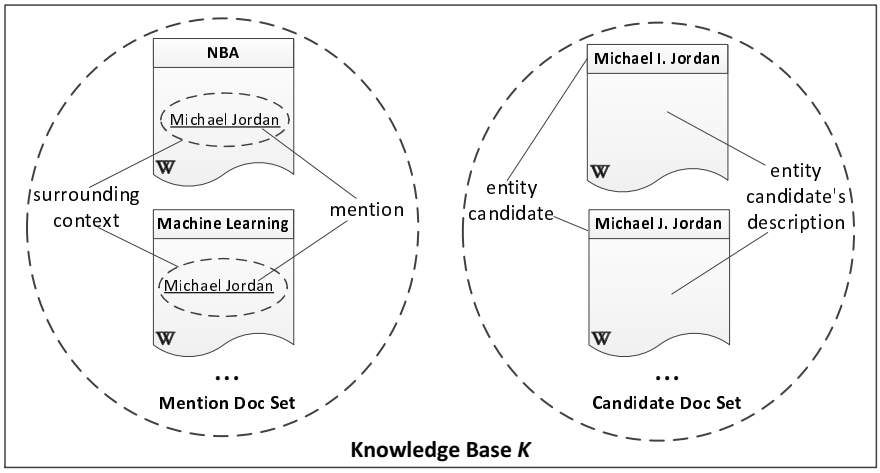


Figure 4.3: Illustration of mention *Michael Jordan*'s candidate documents and mention documents

1. *candidate documents*: m's referent entities' description documents in the knowledge base. Each document is associated with the corresponding entity it describes.

2. *mention documents*: Other documents in K whose contents contain mention m. These documents could be entity documents with titles different from m.

3. *query documents*: documents containing the target mention m and its query context.

4.2.2 Word Evidences

The NED problem arises from the fact that the same textual mention can represent multiple different entities depending on the context of its appearance. The reason why context can help disambiguate mention is that each referent entity candidate can be distinguished by a set of representative words. Those representative words can be seen as the disambiguation evidences for those entity candidates. The candidate documents can explicitly provide some basic evidences (i.e. entity descriptions). However, to achieve good NED performance, we still need some auxiliary information (e.g. semantic relatedness). Therefore, in the LNED problem, we hope to mine additional word evidences from mention documents, to

mimic the following effects of the cross-document hyperlinks and thus supply the auxiliary information.

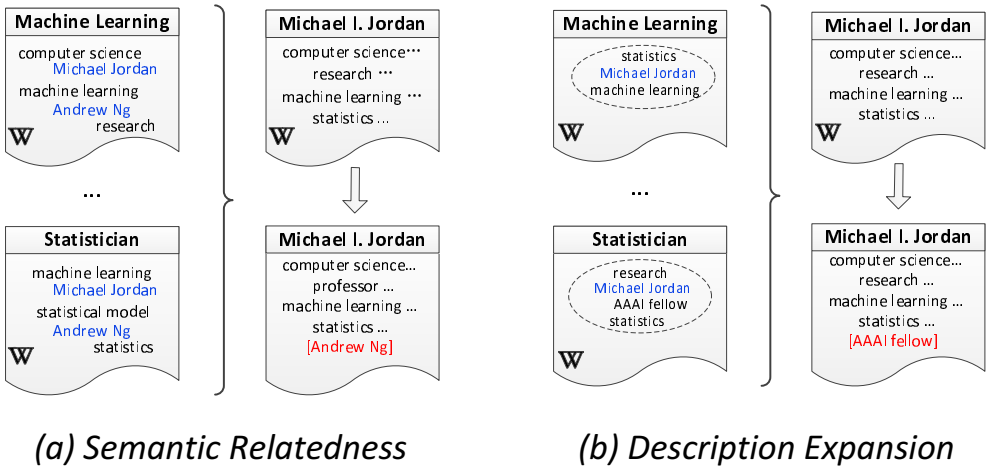


Figure 4.4: Mimicing the effects of hyperlinks

1. *semantic relatedness*. If two entities are semantic related, they share many common incoming hyperlinks, which can be used to measure their relatedness. Without hyperlinks, we can still capture their relatedness, via adding their names into each other's supporting evidences. Then the semantic relatedness effect can be revealed through context comparison. For instance, as shown in Figure 4.4(a), entity *Michael I. Jordan* and *Andrew Ng* are semantic related, so they co-occur in many documents. Meanwhile, some words (e.g. "research", "machine learning") appearing in *Michael I. Jordan*'s descriptions may also appear in these documents. As we know these words are supporting evidences for *Michael I. Jordan*, by analyzing the word co-

occurrence patterns, we can associate the words "Andrew Ng" as *Michael I. Jordan*'s disambiguation evidences as well, since they co-occur with *Michael I. Jordan*'s representative words. Now, given a query containing a mention of "Michael Jordan", with "Andrew Ng" being part of the query context, even we know nothing about the hyperlinks, it is still possible to correctly disambiguate the mention to *Michael I. Jordan*, by comparing the query context with *Michael I. Jordan*'s word-level disambiguation evidences.

2. *description expansion for context similarity.* If entity e_1 appears in the entity document of e_2, d_{e_2}, via hyperlinks, one can expand e_1's entity document by adding e_1's surrounding words in d_{e_2}. Without hyperlinks, we can still perform such expansions, via directly mining those surrounding words from knowledge base and adding them as e_1's supporting evidences. For instance, as shown in Figure 4.4(b), the critical descriptive words "AAAI fellow" of entity *Michael I. Jordan* are expanded from a document where *Michael I. Jordan* appears via hyperlink. Now without links, we don't know to which entity the mention "Michael Jordan" really refers in the document. However, we notice that some words (e.g. "research", "statistics") appearing in *Michael I. Jordan*'s descriptions also appear in this document. As we know these words are supporting evidences for *Michael I. Jordan*, by analyzing the word co-occurrence patterns, we can associate "AAAI fellow" as *Michael I.*

Jordan's disambiguation evidences. Now, a query containing "AAAI fellow" can be easily disambiguated via context comparison.

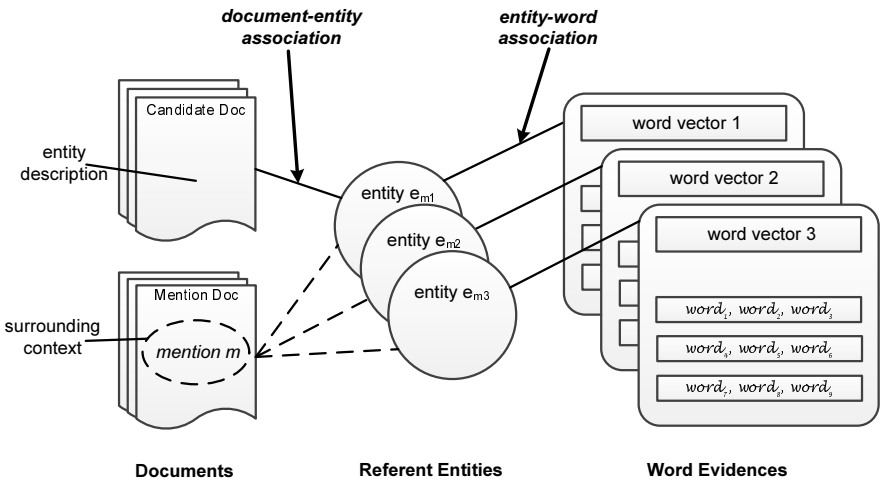


Figure 4.5: Documents, Entities and Words

Figure 4.5 shows the association among referent entities, word evidences, and documents. Given a target mention m, the document-entity association θ_d for document d is a distribution over m's entity candidates, with each component $\theta_d^{e_{mi}}$ indicating the likelihood that mention m's referent entity in d is e_{mi}. Similarly, we have an entity-word association ϕ_e for each entity candidate e so that the words with high probabilities in ϕ_e are precisely the critical disambiguation evidences for e. In Section 4.3.2 we develop a generative model to automatically learn θ and ϕ.

4.2.3 LNED via Evidence Mining

Algorithm 4 provides a high-level description of our approach to tackle the LNED problem by leveraging the useful disambiguation evidences scattered across the reference knowledge base. We will first generate the entity candidates list for the target mention m. Then we will mine disambiguation evidences jointly from m's candidate documents and mention documents, via utilizing the word co-occurrence patterns. Upon the completion of evidence mining, we can utilize the mined evidences to rank entity candidates and choose the top-ranked candidate as disambiguation result.

Note that the evidence mining step (Step 4 in Algorithm 4) is independent of the query context. For each named entity mention m, evidence mining is performed only once, regardless of different query contexts for the same mention. In practice the set of ambiguous mentions can be pre-fetched from the knowledge base K. Therefore the evidence mining step shall run **offline** as a preprocessing step.

4.3 Mining Evidences

In this section we formally introduce our proposed model, for mining evidences from the knowledge base.

Algorithm 4 LNED via Evidence Mining

Input: Reference knowledge base K (with no links), named entity mention m, query q.

1: Generate candidates list C for mention m

2: Fetch candidate documents set D_C from K

3: Fetch m's mention documents set D_M from K

4: Mine evidences from $D_C \cup D_M$

5: Use mined evidences to rank candidate $c \in C$ for m in q

6: Return top-ranked candidate c_{top} as the answer

4.3.1 Model Intuitions

Based on the assumption that disambiguation evidences are entity-specific representative words, it is natural to model each entity as a topic/label and imagine those representative words are generated from such topics. For a given target mention m, we model each of its entity candidates as a regular topic and introduce the following three special topics to capture some noisy or useless words.

1. *background*. Some words in the documents might be general to more than one candidate. Therefore we introduce a special background topic to capture those non-representative words.

2. *undefined*. Since a knowledge base K is very likely incomplete, some entities named after m may not be indexed by K. Therefore we introduce a special topic called "undefined" to capture the words that are associated with these undefined entities.

3. *master*. Since mention documents themselves could be description documents for other entities, words in these documents might be generated from these entities instead of target mention's candidate entities. Thus we introduce a special "master" topic to capture those words. Note that each mention document will have one unique "master" topic.

4.3.2 Model Details

We now explain the details of our generative model. Figure 4.6 shows the graphical structure of dependencies of our model. Each node in the figure corresponds to a random variable or prior parameter. The shaded nodes represent observed variables while other nodes represent latent variables. A plate means the nodes within it are replicated for multiple times. A directed edge from node a to node b indicates that the variable represented by b is dependent on the the variable represented by a.

Table 4.2 summarizes the notations used in our model. Given a named entity mention m, we will first find all of its possible referent entity candidates and denote

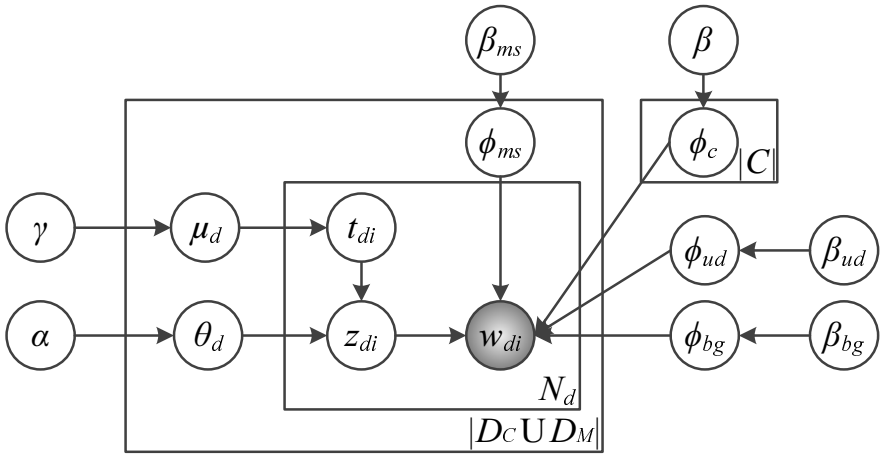


Figure 4.6: Proposed Model for LNED

the candidates set as C. Each referent entity candidate will then be treated as a regular topic/label and the total number of them is $|C|$.

1. Given a candidate document $d_{candidate}$, its underlying entity e for m is already identified. For each word w in $d_{candidate}$, its label z is either e, or the "background" label. The selection is controlled by an indicator variable t drawn from a multinomial distribution μ.

2. For a mention document $d_{mention}$, we further split it into two sub-documents: surrounding context $d_{surround}$ and the rest d_{other}. $d_{surround}$ represents the limited size context (e.g. a width-W word window surrounding m), while d_{other} captures the out-of-window words in $d_{mention}$. For each word w in $d_{surround}$, its label z is chosen from "background", the "master" entity of

$d_{mention}$, or a label (from $C \cup$ "$undefined$") drawn from the multinomial distribution θ. The selection is also controlled by an indicator variable t drawn from a multinomial distribution μ. For each word w in d_{other}, its label z is fixed as the "master" entity of $d_{mention}$.

The multinomial distribution θ is drawn from a Dirichlet prior with α and α_{ud} as the hyperparameters. The difference between α and α_{ud} should reflect how conservatively we choose between regular topics and the special "undefined" one. The multinomial distribution μ is drawn from a Dirichlet prior with γ_1, γ_2 and γ_3 as the hyperparameters. The difference between γ_1, γ_2 and γ_3 should reflect the proportion of "background" topic, regular topics and "master" topics.

For each topic/label in $C \cup$ "$undefined$" $\cup$ "$background$" $\cup$ "$master$", it is associated with a multinomial distribution ϕ over words, which is drawn from the Dirichlet prior with β, β_{ud}, β_{bg} and β_{ms} as the hyperparameters. The difference among β, β_{ud}, β_{bg} and β_{ms} should reflect the content difference among regular labels, the "undefined" label, the "background" label and the "master" labels.

Finally, each word w is drawn from the multinomial distribution ϕ_z, where z is the word label for w. **Our goal** is to infer the document-label association θ and the label-word association ϕ from this model. To summarize, the detailed generative process of our model is as follows:

1. Draw the multinomial distribution over words $\phi_c \sim \mathit{Dirichlet}(\beta)$ for each regular topic c.
2. Draw the multinomial distribution over words $\phi_{bg} \sim \mathit{Dirichlet}(\beta_{bg})$ for the background topic.
3. Draw the multinomial distribution over words $\phi_{ud} \sim \mathit{Dirichlet}(\beta_{ud})$ for the undefined topic.
4. Draw the multinomial distribution over words $\phi_{ms} \sim \mathit{Dirichlet}(\beta_{ms})$ for each master topic.
5. For each document $d \in D_C$:
   (a) Let e_{candi} = candidate label of d
   (b) Choose a background topic proportion $\mu_d \sim \mathit{Dirichlet}(\gamma_1, \gamma_2)$.
   (c) For each word position i in document d:
      i. Choose a background indicator $t_{di} \sim \mathit{Multinomial}(\mu_d)$.
      ii. if $t_{di} = 0$:

          Choose topic $z_{di} = bg$.

      iii. else:

          Choose topic $z_{di} = e_{candi}$.

      iv. Choose a word $w_{di} \sim \mathit{Multinomial}(\phi_{z_{di}})$.
6. For each document $d \in D_{surround}$:

(a) Draw a topic distribution $\theta_d \sim Dirichlet(\boldsymbol{\alpha})$, where $\boldsymbol{\alpha} = (\alpha_{ud}, \alpha_1, ..., \alpha_c)$ and $\alpha_1 = \ldots = \alpha_c = \alpha$.

(b) Let e_{ms} = master entity label of the mention document from which d is extracted

(c) Choose a background/regular/master topic proportion $\mu_d \sim Dirichlet(\gamma_1, \gamma_2, \gamma_3)$.

(d) For each word position i in document d:

i. Choose a background/regular/master indicator $t_{di} \sim Multinomial(\mu_d)$.

ii. if $t_{di} = 0$:

Choose topic $z_{di} = bg$.

iii. else if $t_{di} = 1$:

Choose topic $z_{di} \sim Multinomial(\theta_d)$.

iv. else if $t_{di} = 2$:

Choose topic $z_{di} = e_{ms}$.

v. Choose a word $w_{di} \sim Multinomial(\phi_{z_{di}})$.

7. For each document $d \in D_{other}$:

(a) Let e_{ms} = master entity label of the mention document from which d is extracted

(b) For each word position i in document d:

i. Choose topic $z_{di} = e_{ms}$.

ii. Choose a word $w_{di} \sim \textit{Multinomial}(\phi_{z_{di}})$.

Our model is similar to the MENED model proposed in [65], which aims at mining evidences from external corpus to bridge the information gap between the queries and the reference knowledge base. Compared with the MENED model, our model is different in two aspects:

1. In MENED, each document has only one label. The model will first sample the document label. Once the document label is chosen, every word in the document can only have two possible labels: foreground or background, and the foreground label is restricted by the document label. While this is an effective constraint for the problem studied in [65], it is inappropriate for our LNED problem. In LNED, the number of labeled documents is much less than that of unlabeled documents. Therefore it is very likely to assign a wrong label to an unlabeled document. If this constraint is applied, then all the words inside that document will get wrong labels, which will in turn confuse the label-word association and get more documents wrongly labeled. To avoid this, we model each document as a mixture of different labels and directly infer the label for each word.

2. In MENED, every word within a mention document is considered as either a background word, or a supporting evidence for one of the entity candidates. In LNED problem, each mention document itself is a description page for some "master" entity. Hence, some words in the mention document may be generated from the "master" entity instead of one of the entity candidates. To properly handle these words, we introduce a special "master" topic and assume all such words are generated from the corresponding "master" topics.

In Section 4.5.4, we will conduct experiments to compare our proposed model with MENED and demonstrate our model's advantages over MENED.

4.3.3 Inference Algorithm

4.3.3.1 Likelihood Function

The joint likelihood function of our model is:

$$\begin{aligned} & p(\boldsymbol{w}, \boldsymbol{t}, \boldsymbol{z} | \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \\ = & \int_{\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\mu}} p(\boldsymbol{\theta}|\boldsymbol{\alpha}) p(\boldsymbol{\phi}|\boldsymbol{\beta}) p(\boldsymbol{\mu}|\boldsymbol{\gamma}) p(\boldsymbol{t}|\boldsymbol{\mu}) p(\boldsymbol{z}|\boldsymbol{\theta}, \boldsymbol{t}) p(\boldsymbol{w}|\boldsymbol{z}, \boldsymbol{\phi}) d\boldsymbol{\theta} d\boldsymbol{\phi} d\boldsymbol{\mu} \end{aligned} \tag{4.1}$$

Given the hyperparameters $\Gamma = \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}\}$, and the observed words $\boldsymbol{w}$, we will calculate the posterior probability of $p(\boldsymbol{t}, \boldsymbol{z} | \boldsymbol{w}, \Gamma)$, and use the maximal marginal probability to infer each word's topic assignment z_{di} and label category indicator

t_{di}. After that we can make use of the inferred $\boldsymbol{t}$ and $\boldsymbol{z}$ to estimate the document-label association $\boldsymbol{\theta}$ and the label-word association $\boldsymbol{\phi}$.

4.3.3.2 Approximate Inference via Gibbs Sampling

Similar to many other topic models with conjugate prior (e.g LDA [8]), exact inference is intractable for our model. Here we use Gibbs Sampling as an approximate inference method. Compared with other approximate inference methods such as Variational Inference, Gibbs Sampling is easy to extend and has been proved to be quite effective in avoiding local optima. In Gibbs Sampling, each hidden variable will be iteratively sampled and the corresponding marginal probability can later be estimated with the samples.

In our model, the word topic assignment variable z_{di} and the label category indicator variable t_{di} are highly correlated since t_{di} controls the selection of z_{di}. Once t_{di} is assigned some value, z_{di} can only be sampled from the corresponding distribution indicated by t_{di}. Therefore we design a blocked Gibbs Sampler to group z_{di} and t_{di} together, and sample from their joint distribution conditional on all other variables, instead of sampling from each one individually.

Algorithm 5 describes the blocked Gibbs Sampling process. Note that we will only sample the words in document $d \in D_C \cup D_{surround}$. For words in document $d \in D_{other}$, the word topic is fixed as e_{ms} and therefore no sampling is needed.

Algorithm 5 Blocked Gibbs Sampling

for all *iter* from 1 to $MaxIter$ **do**

  for all $d \in D_C \cup D_{surround}$ **do**

    for all i from 1 to N_d **do**

      sample $\{z_{di}, t_{di}\}$ together according to $p(z_{di}, t_{di}|\boldsymbol{w}, \boldsymbol{z}_{-di}, \boldsymbol{t}_{-di}, \Gamma)$

    end for

  end for

end for

The sampling function $p(z_{di}, t_{di}|\boldsymbol{w}, \boldsymbol{z}_{-di}, \boldsymbol{t}_{-di}, \Gamma)$ for different document sets D_C and $D_{surround}$ are only slightly different from each other. Due to the space limit, here we only describe the detailed sampling functions for documents in $D_{surround}$, which is more complicated than those in D_C. For each word w_{di} in $d \in D_{surround}$:

1. z_{di} is sampled to the "background" topic with:

$$p(t_{di} = 0, z_{di} = bg|\boldsymbol{w}, \boldsymbol{z}_{-di}, \boldsymbol{t}_{-di}, \Gamma) \propto \frac{|w = w_{di}, t_w = 0| + \beta_{bg}}{|t_w = 0| + V \cdot \beta_{bg}}$$

$$\cdot(|t_w = 0, w \in d| + \gamma_1) \tag{4.2}$$

2. z_{di} is sampled to the "undefined" topic or one of the regular topics with:

$$p(t_{di} = 1, z_{di} = c|\boldsymbol{w}, \boldsymbol{z}_{-di}, \boldsymbol{t}_{-di}, \Gamma) \propto$$

$$\frac{|w \in d, t_w = 1, z_w = c| + \alpha_c}{|w \in d, t_w = 1| + |C| \cdot \alpha + \alpha_{ud}} \cdot \frac{|w = w_{di}, t_w = 1, z_w = c| + \beta_c}{|t_w = 1, z_w = c| + V \cdot \beta_c}$$

$$\cdot(|t_w = 1, w \in d| + \gamma_2)\ , \tag{4.3}$$

where $c \in C \cup$ "$undefined$". If $c =$ "$undefined$", $\alpha_c = \alpha_{ud}$ and $\beta_c = \beta_{ud}$; otherwise, $\alpha_c = \alpha$ and $\beta_c = \beta$.

3. z_{di} is sampled to the "master" topic with:

$$p(t_{di} = 2, z_{di} = e_{ms} | \boldsymbol{w}, \boldsymbol{z}_{-di}, \boldsymbol{t}_{-di}, \Gamma) \propto$$
$$\frac{|w = w_{di}, t_w = 2, w \in d| + \beta_{ms}}{|t_w = 2, w \in d| + V \cdot \beta_{ms}} \cdot (|t_w = 2, w \in d| + \gamma_3) \tag{4.4}$$

Sampling functions for documents in D_C can be derived similarly, according to the corresponding generative process.

4.3.3.3 Estimating Document-Label Association

After enough iterations of sampling for $\boldsymbol{z}$ and $\boldsymbol{t}$, the document-label association can be estimated by *maximum a posteriori* (MAP) inference:

$$\theta_d^{(c)} = \frac{|w \in d, t_w = 1, z_w = c| + \alpha_c}{|w \in d, t_w = 1| + |C| \cdot \alpha + \alpha_{ud}}, \tag{4.5}$$

where $c \in C \cup$ "$undefined$". If $c =$ "$undefined$", $\alpha_c = \alpha_{ud}$ and $\beta_c = \beta_{ud}$; otherwise, $\alpha_c = \alpha$ and $\beta_c = \beta$.

4.3.3.4 Estimating Label-Word Association

Similarly, we can infer the label-word association by MAP inference:

$$\phi_{bg}^{(v)} = \frac{|w = v, t_w = 0| + \beta_{bg}}{|t_w = 0| + V \cdot \beta_{bg}}, \tag{4.6}$$

$$\phi_{ud}^{(v)} = \frac{|w = v, t_w = 1, z_w = ud| + \beta_{ud}}{|t_w = 1, z_w = ud| + V \cdot \beta_{ud}}, \tag{4.7}$$

$$\phi_{c}^{(v)} = \frac{|w = v, t_w = 1, z_w = c| + \beta}{|t_w = 1, z_w = c| + V \cdot \beta} \tag{4.8}$$

For the LNED task, we are particularly interested in the label-word association $\phi_c^{(v)}$, which reveals the disambiguation evidences for each referent entity candidate.

4.4 Ranking Referent Candidates

Our ultimate goal for the LNED problem is to disambiguate entity mentions in query documents. Upon the completion of the evidence mining step, we can make use of the knowledge learned from our evidence mining model to rank referent entity candidates, and choose the top-ranked candidate as disambiguation result. Given a query document, we predict its word labels $\boldsymbol{z}$ using the incremental Gibbs Sampling algorithm described in [60]. Namely, we iteratively update the word topic assignments of a query document using the above inference process, but with the previously learned global knowledge (i.e. $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$) fixed. As the sampling is operated only on the words in the query document, it converges very fast (e.g. less than 30 iterations).

After the sampling converges, we infer the document-label association θ_d for each query document d, using Equation 4.5. The disambiguation result can then be predicted with the maximal marginal probability:

$$LNED(d) = argmax_c \theta_d^{(c)}. \tag{4.9}$$

4.5 Experiments

In this section, we evaluate the effectiveness of our proposed method for the LNED problem on two real-life query datasets [65]: one from news, and the other from Twitter. We will: (1) illustrate the effectiveness of mining evidences for LNED, by comparing our model against a similar generative model which has no evidence mining component; (2) demonstrate the superiority of our method by comparing it with a baseline method which also performs NED via evidence mining; (3) compare the end-to-end disambiguation accuracy of our method, with two state-of-the-art NED methods (with their link-based features disabled); (4) show the quality of the evidences harvested by our model; (5) show how the performance of our method changes with respect to surrounding window size of entity mentions. All the experiments, if not specifically mentioned, are conducted on a server with 2.40GHz Intel Xeon CPU and 48GB RAM.

4.5.1 Datasets

We adopt the two datasets used in [65] to test our method. The first one is derived from the TAC-KBP2009 dataset, which is created for the Entity Linking task [57] in the Knowledge Base Population track at the Text Analysis Conference. The queries in this dataset are all news articles. Therefore the queries are relatively long and the writing quality is good. *Note that our experiments setting is more challenging than the TAC-KBP competition [103] since we don't assume the availability of various kinds of annotations (e.g. entity type, Wikipedia infobox).* The second dataset is generated from Twitter. Since tweets have the 140-character constraint and the words used in them are often irregular, the queries are usually very short and the writing quality is not well-expected. Table 4.1 shows some basic statistics of these two datasets. As can be seen, it is quite challenging to conduct NED on these two datasets since there are many entity candidates for the query mentions.

	TAC-KBP2009	Twitter
# of Queries	424	340
Avg Length of Queries	53.15 words	16.46 words
Avg # of Candidates	24.024	19.279

Table 4.1: Basic statistics of test datasets for LNED

4.5.2 Experiments Setup

While more and more closed domain knowledge bases are emerging, most of them are restricted to inside domain access and thus not publicly available. Moreover, to make it practical to compare with previous methods which use Wikipedia as the knowledge base, the *linkless version of Wikipedia* is used in this work, where we keep all entity description pages in Wikipedia but discard all link-related information such as cross-document hyperlinks and entity categories. In such a case, for each referent entity candidate, the only labeled data we have is its own Wikipedia page. This modification can establish a **public benchmark** for algorithm comparison. Following previous work [89, 85, 16] on NED, we make use of the "Disambiguation Pages" and "Redirect Pages" in Wikipedia to find all the entity candidates that a given mention can be mapped to. Note that these two types of pages are not related with the cross-document hyperlinks. For fetching mention documents from the knowledge base, we make use of the Wikipedia Search API and collect all the Wikipedia pages that contain the query mention.

Our model has some hyperparameters, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. In this work, we use the following parameter settings: $\alpha = 0.01, \alpha_{ud} = 0.1, \beta = 0.01, \beta_{ud} = 0.1, \beta_{bg} = 0.1, \beta_{ms} = 0.01, \gamma_1 = 0.01, \gamma_2 = 1, \gamma_3 = 2$. These parameters are tuned on a separate develop dataset containing 15 queries and then reused in all the experiments

without any further tuning. Besides, the surrounding window size W (see Section 4.3.2) is set as 40 for Twitter dataset and 30 for TAC-KBP2009 dataset.

To train the evidence mining model, we run 2000 iterations of our Gibbs sampling algorithm to its convergence. The training time varies from several seconds to a few hours for different entity mentions, depending on the corresponding number of candidate documents and mention documents. As discussed in Section 4.2.3, this training process shall run offline. Therefore we believe that the training time is not critical to the real world applications. After training, the online disambiguation of query documents (Section 4.4) is very quick (usually within seconds).

4.5.3 Effectiveness of Evidence Mining

We first illustrate the effectiveness of mining evidences to bridge the information gap (caused by the missing links), by comparing the LNED accuracy of our model (we name it as *Linking Evidences in Not Well Linked Sources*, **LENS**), with a baseline model, **Labeled-LDA** [84], which has no evidence mining component. Labeled-LDA is also extended from the standard LDA [8]. In Labeled-LDA, each document can have multiple labels and the label-word correspondences can be inferred. Labeled-LDA can be directly used for LNED purpose by treating each referent entity candidate as a unique label. In this way, the model can learn the

label-word association from the candidate documents and then use it to rank referent entities with respect to the query. Note that Labeled-LDA's disambiguation decisions are made from candidate documents. All other documents containing mentions are discarded without processing.

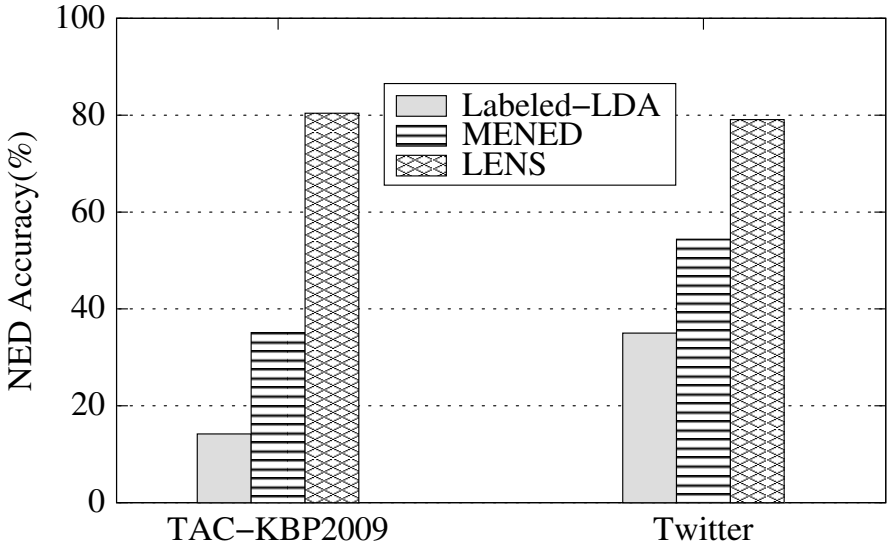


Figure 4.7: LENS vs. Labeled-LDA and MENED

Figure 4.7 shows the results for LENS and Labeled-LDA on both datasets. Suffering from the information gap caused by the missing links, Labeled-LDA works badly in LNED task. Conversely, our LENS model works much better. That is because LENS utilizes not only the candidate documents, but also useful evidences scattered across mention documents. The performance gain clearly illustrates the effectiveness of mining evidences to bridge the information gap.

4.5.4 Comparison of Evidence Mining Methods

We then conduct experiments to compare the disambiguation accuracy of LENS, against the **MENED** model proposed in [65]. MENED was originally designed to mine additional evidences from external corpus to help NED. In the LNED problem setting, we can treat the mention documents as the "external corpus". By doing so, MENED can perform LNED via mining evidences from those mention documents. For MENED, we use the parameters suggested in [65].

Figure 4.7 shows that LENS outperforms MENED on both datasets. Like LENS, MENED also uses the mention documents in the knowledge base to obtain additional disambiguation evidences. However, it assumes that all the surrounding words contribute to the entity candidates. In fact, some surrounding words are only used to describe the "master" entity of the mention document, and just accidentally co-occur with the mention. LENS is aware of this phenomenon and can utilize other words in the mention documents to help eliminate the effect of those words.

4.5.5 End-to-end NED Accuracy

We then conduct experiments to compare the end-to-end disambiguation accuracy of LENS, against two state-of-the-art NED methods: **Wikifier** [85, 21], a widely-used NED system using a machine learning based hybrid strategy to com-

bine various kinds of features together, and **AIDA** [51], a robust NED system making use of weighted mention-entity graph to find the best joint mention-entity mapping. To make Wikifer and AIDA fit the LNED problem setting, we modified them to disable all the link-based features (e.g. semantic relatedness) and then re-trained the models. The modified linkless versions are denoted as **Wikifier(w/o link)** and **AIDA(w/o link)**[2]. All three methods use a Wikipedia repository of late 2012 as the reference knowledge base.

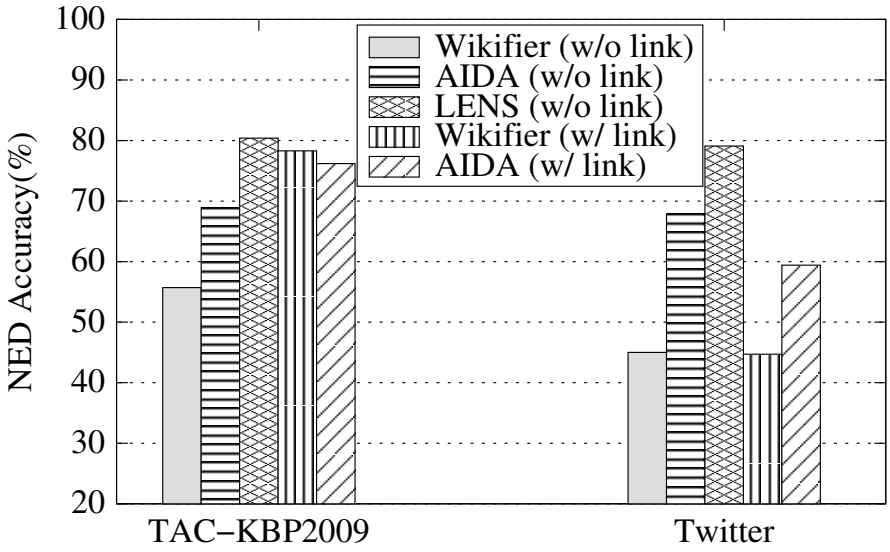


Figure 4.8: LENS vs. Wikifier vs. AIDA

Figure 4.8 shows that LENS significantly outperforms Wikifier and AIDA on both datasets. Compared with Wikifier and AIDA, LENS can collect disambigua-

[2]AIDA extracts keyphrases from various sources to describe entities. As these keyphrases are pre-extracted and indexed offline, we are unable to modify them. So precisely speaking, AIDA(w/o link) still utilizes few link-based features.

tion evidences scattered in the reference knowledge bases to mimic the role of the missing cross-document hyperlinks and thus achieve better disambiguation results. To demonstrate the helpfulness of such links, we also present the NED accuracy of the original Wikifier and AIDA (with all features enabled and using the parameter settings suggested by their authors) in Figure 4.8 (denoted as **Wikifier(w/ link)** and **AIDA(w/ link)**, respectively). As we can see, on the TAC-KBP2009 dataset, the link-based features are quite helpful in boosting the NED accuracy. LENS can harvest evidences to mimic the effects of these links and achieve similar NED accuracy with the original Wikifier and AIDA. On the Twitter dataset, the full version Wikifier and AIDA perform even worse than the modified linkless versions. This is because both Wikifier and AIDA utilize entity popularity information to give more preferences to popular entity candidates, while in Twitter dataset, many queries' disambiguation answers are actually non-famous entities. LENS can still achieve good performance on the Twitter dataset, due to the following two reasons: (1) LENS makes NED decisions without applying any prior preferences; (2) the evidences mined from "description expansion" are very helpful for short texts like tweets.

4.5.6 Quality of Mined Evidences

Next we conduct experiments to show the quality of the disambiguation evidences harvested through our model. We collect all Wikipedia pages in which the target mentions from the two datasets are used as hyperlink anchor texts. Then we treat the paragraphs where such anchor texts appear as queries and use the entities to which the anchor texts link as the ground truth. In total, we obtained 30,544 queries for mentions in TAC-KBP2009 dataset and 8,523 queries for mentions in Twitter dataset. As all these queries are directly formed from the cross-document hyperlinks, the disambiguation accuracy on them can roughly reflect how well the mined evidences can mimic the role of these links.

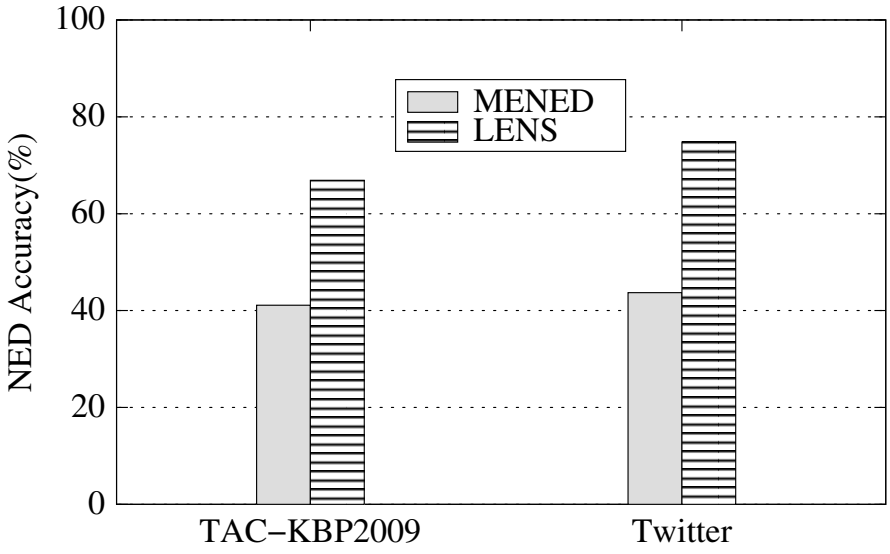


Figure 4.9: Quality of Evidences: LENS vs. MENED

Since MENED is the only baseline model which has the functionality of mining evidences, in this part, we will only compare LENS with the MENED model. Here the parameter settings are the same as those described in Section 4.5.4. Figure 4.9 shows the comparison results. Again, LENS outperforms MENED, indicating that the intuitions we adopted to build LENS are reasonable and effective. Considering the high ambiguities of the mentions, the quality of the mined evidences is acceptable.

4.5.7 Impact of Surrounding Window Size

In this part we conduct experiments to illustrate how the performance of our LENS model changes with respect to the surrounding window size W in mention documents (see Section 4.3.2). Here we still use the same parameter settings as in previous experiments. Figure 4.10 shows that as W increases, the NED accuracy will increase to a peak point and decrease afterwards. There are two factors inside this phenomenon. As the window size increases, more evidences are exposed to the model. Hence, the NED accuracy will increase. On the other hand, with the increase of W, more noisy words may be wrongly judged as supporting evidences. Our model incorporates the "master" topic to filter the helpless words. However, when the window is too large, most words in the mention document $d_{mention}$ will be split into $d_{surround}$. Therefore the very few words in d_{other} are insufficient to filter

out the noisy words in $d_{surround}$, which results in the performance degradation. In practice, W usually works very well at $20 \sim 40$.

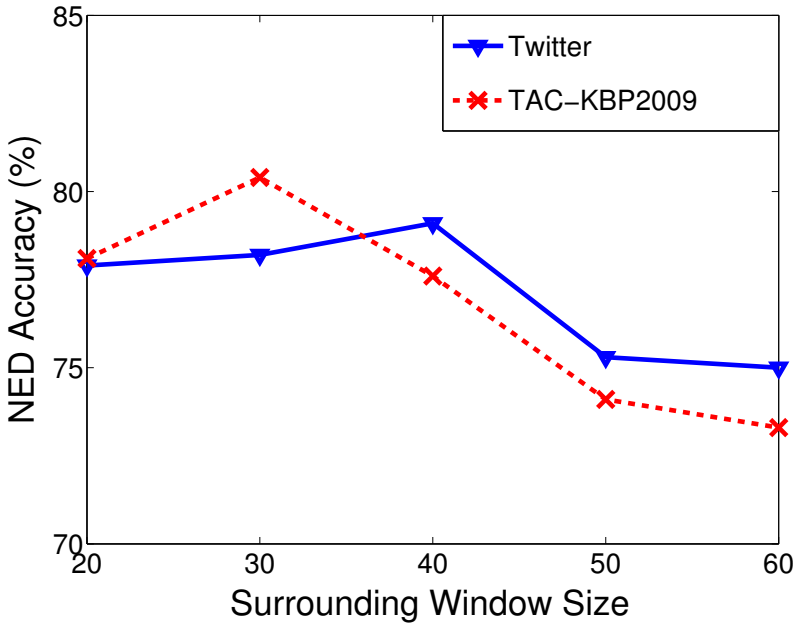


Figure 4.10: Varying Surrounding Window Size for LENS

4.6 Summary

In this chapter, we studied the problem of *Named Entity Disambiguation with Linkless Knowledge Bases* (LNED). We proposed a generative model to automatically mine useful evidences from the reference knowledge base so that the mined evidences can help mimic the role of the missing links. With a specific modeling of "background topic", "undefined entities" and "master entities", our model is able

to harvest useful evidences from noisy text. To evaluate the effectiveness of our model, a thorough experimental study was conducted. The experimental results demonstrated that our proposed method can mine useful evidences to bridge the information gap caused by the absence of cross-document links, thus significantly boosting the NED performance for a linkless knowledge base.

Symbols	Descriptions
D_C	the set of referent entity candidate documents
D_M	the set of mention documents
$D_{surround}$	the set of surrounding context documents (extracted from D_M)
D_{other}	the set of non-surrounding context documents (extracted from D_M)
C	the set of regular entity labels (i.e. entity candidates)
V	vocabulary size
N_d	the number of words in document d
w_{di}	the i-th word of document d
z_{di}	the label associated with w_{di}
t_{di}	the background/regular/master topic indicator for w_{di}
μ_d	the background/regular/master topic proportion for document d
θ_d	the topic distribution for document d
$\phi_{bg}, \phi_{ud}, \phi_{ms}$	the word distribution for the background, undefined and master topic
ϕ_c	the word distribution for the c-th regular topic ($1 \leq c \leq \|C\|$)
α_{ud}, α	the hyperparameters for Dirichlet prior of θ
$\beta_{bg}, \beta_{ud}, \beta_{ms}$	the hyperparameter for Dirichlet prior of ϕ_{bg}, ϕ_{ud} and ϕ_{ms}
β	the hyperparameters for Dirichlet prior of ϕ_c ($1 \leq c \leq \|C\|$)
γ	the hyperparameters for Dirichlet prior of μ

Table 4.2: Notations used in proposed model for LNED

Chapter 5

Acronym Disambiguation for Enterprises

In this chapter, we move one step further to explore the entity disambiguation problem where there are no reference knowledge bases being available. This setting further looses the underlying requirements for connecting text with knowledge, as it can be potentially applied to any domains. But on the other hand, it is much more challenging since the knowledge has to be directly mined and organized from plain text. As a first step towards this goal, we study the acronym disambiguation for enterprises problem. Acronyms are abbreviations formed from the initial components of words or phrases. In enterprises, people often use acronyms to make communications more efficient. However, acronyms could be difficult to

understand for people who are not familiar with the subject matter (new employees, collaborators from other groups, etc.), thereby affecting productivity. To alleviate such troubles, we study the acronym disambiguation problem, which is to automatically resolve the true meanings of acronyms in a given context. Acronym disambiguation for enterprises is challenging for several reasons. First, acronyms may be highly ambiguous since an acronym used in the enterprise could have multiple internal and external meanings. Second, there are usually no comprehensive knowledge bases such as Wikipedia available in enterprises, so all the signals for disambiguation (including potential meanings, and their popularity scores, context representations, etc.) need to be mined from plain text. Finally, the system should work for any enterprise automatically, so it shall not rely much on manually labeled data. In this chapter we propose a general end-to-end framework to tackle all these challenges. The framework takes the enterprise corpus as input and produces a high-quality acronym disambiguation system as output. Our disambiguation models are trained via distant supervised learning, without requiring any manually labeled training examples. Therefore, our proposed framework can be easily deployed to any enterprise or closed-domain corpus to support high-quality acronym disambiguation. Experimental results on real world data justified the effectiveness of our system.

5.1 Background and Preliminary Material

Acronyms are abbreviations formed from the initial components of words or phrases (e.g., "AI" from "Artificial Intelligence", "ACM" from "Association for Computing Machinery"). As acronyms can shorten long names and make communications more efficient, they are widely used at almost everywhere in enterprises, including notifications, emails, project reports and social network posts. Figure 5.1 shows a sample post from Microsoft's Yammer social network. As we can see, acronyms are very frequently used in the post.

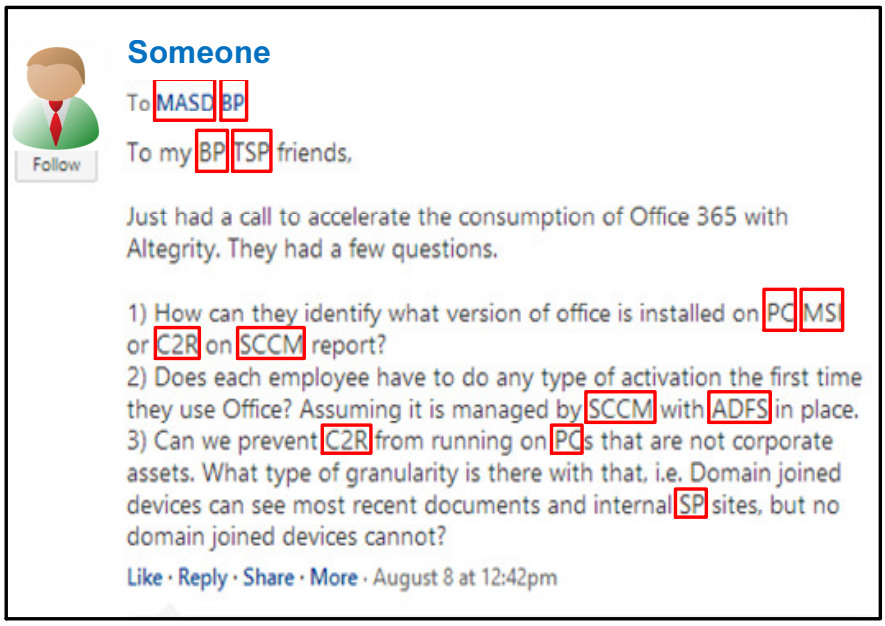


Figure 5.1: Acronyms in Enterprises

Despite the fact that acronyms can make communications more efficient, sometimes they could be very difficult to understand, especially for people who are not very familiar with the specific areas, such as new employees, collaborators from other groups and patent lawyers. In such situations, acronyms could negatively affect productivity. We randomly sampled 1000 documents from a Microsoft internal question answering forum and found out that only 7% of the acronyms co-occur with the corresponding meanings in the same document, which means 93% of the time when the user does not understand an acronym, she will need to find clues outside of the document. Therefore, it is particularly useful to develop a system that can automatically resolve the true meanings of acronyms in enterprise documents. Figure 5.2 shows an example: in the first document, "AI" means "Asset Intelligence" [1], while in the second document, "AI" means "Artificial Intelligence"; the goal of our system would be automatically figuring out the correct mapping from the context. Such system could be run online as a querying tool to handle any ad-hoc document, or run offline to annotate acronyms with their true meanings in a large amount of documents. In the offline mode, the true meanings can be further indexed by an enterprise search engine, so that when users search for the true meaning, documents that only contain the acronym can also be found.

[1] `http://technet.microsoft.com/en-us/library/gg681998.aspx`

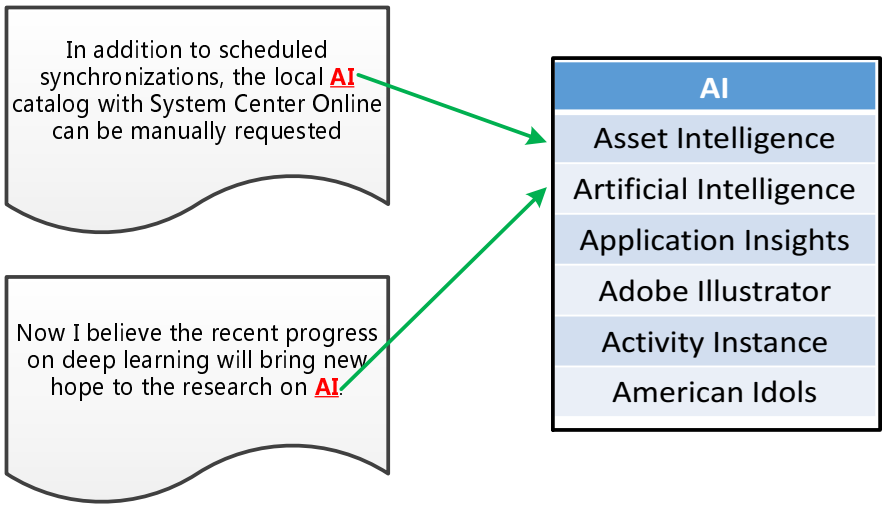


Figure 5.2: Acronym Disambiguation Example

The enterprise acronym disambiguation task is very challenging due to the high ambiguity of acronyms, e.g., "SP" could stand for "Service Pack", "SharePoint" or "Surface Pro" in Microsoft. And there is one additional challenge compared with previous disambiguation tasks: in an enterprise document, an acronym could refer to either an internal meaning (concepts created by the enterprise that may or may not be found outside, like project names in Microsoft) or an external meaning (all concepts that are not internal). For the same example in Figure 5.2, "Asset Intelligence" is an internal meaning, so it rarely appears on the web and there is almost no evidence that people use "AI" as its acronym except in a few Microsoft internal documents about the product. On the other hand, "Artificial Intelligence" is an external meaning and it is the most popular meaning for

"AI" even in Microsoft documents, same as the rest of the world. Therefore, a good acronym disambiguation system should be able to handle both internal and external meanings. As we will explain in details, it is important to make such distinction and different strategies are needed for such two cases.

For internal meanings, there are some previous work on word sense disambiguation [75] and acronym disambiguation [38, 77, 83, 109] on a closed-domain corpus. The main challenge here is that there are rarely any domain-specific knowledge bases available in enterprises, therefore all the signals for disambiguation (including potential meanings, and their popularity scores, context representations, etc.) need to be mined from plain text. Training data should also be automatically generated to make the system easily scale out to all enterprises. Compared with previous work, we developed a more comprehensive and advanced set of features in the disambiguation model, and also used a much less restrictive way to discover meaning candidates and training data, so that both precision and recall can be improved. Moreover, one main limitation of all previous work is that they do not distinguish internal and external meanings. They merely rely on the enterprise corpus to discover information about external meanings, which we observe is quite ineffective. The reason is that for popular external meaning like "Artificial Intelligence", people often directly use its acronym in enterprises without explanation, therefore there is limited information about the connection between the acronym

and the external meaning, as well as the external meaning itself in the enterprise corpus. On the other hand, there are much more such information available in the public domain, which should be leveraged by the system.

If we consider utilizing a public knowledge base such as Wikipedia to better handle external meanings of acronyms, the problem becomes very related to the well studied Entity Linking [30, 33, 48, 51, 65, 85, 89] problem, which is to map entity mentions in texts to their corresponding entities in a reference knowledge base (e.g. Wikipedia). But our disambiguation task is different from the entity linking task, because the system also needs to handle internal meanings which are not covered by any knowledge bases, and ultimately needs to decide whether an acronym refers to an internal meaning or an external meaning. It is nontrivial to combine the information mined from the enterprise corpus and the public knowledge base so that the system can get the best of both worlds. Even for external meanings, it is also important to leverage signals from the enterprise corpus since the context surrounding them could be quite different from that in the external world, and context is one of the most important factor for disambiguation. For example, in public world, when people mention "Operating System" they mainly talk about how to install or use it; while within Microsoft, when people mention "Operating System" most of the time they focus on how to design or implement it.

In this chapter, we design a novel, end-to-end framework to address all the above challenges. Our framework takes the enterprise corpus and certain public knowledge base as input and produces a high-quality acronym disambiguation system as output. The models are all trained via distant supervised learning [29], therefore our system requires no manually labeled training examples and can be easily deployed to any enterprise to support high-quality acronym disambiguation. Experimental study on Microsoft enterprise corpus justified the effectiveness of our system.

5.1.1 Problem Statement

We formalize the problem of *Enterprise Acronym Disambiguation* as follows.

Definition 4 (Enterprise Acronym Disambiguation)**.** *Given acronym mention a with its surrounding context t, and an enterprise corpus C, the goal of acronym disambiguation is to map acronym mention a to its referent meaning m mined from C.*

The *Enterprise Acronym Disambiguation* problem is comprised of two sub-problems. The first one is *Acronym Meaning Mining*, which aims at mining acronym/meaning pairs from the enterprise corpus. Each meaning m should contain the full name expansion e, popularity score p (indicating how often meaning m is used as the genuine referent meaning of acronym a) and context words W

(i.e. words frequently used in context of the meaning). The popularity score and context words could provide critical information for making the disambiguation decisions. The second one is *Meaning Candidate Ranking*, whose goal is to rank the distinct meanings associated with the target acronym a and select the genuine referent meaning m based on the given context.

In this paper we assume the acronyms for disambiguation are provided as input to the system, either by the user or by an existing acronym detection module. We do not try to optimize the performance of acronym detection (e.g. identifying acronyms beyond the simple capitalized rule, or distinguishing cases where a capitalized term is not an acronym but a regular English word, such as "OK"). The task of acronym detection is also interesting and important. But due to the space limit, it is beyond the scope of this paper.

The *Acronym Disambiguation for Enterprises* problem is more challenging than the well-studied *Named Entity Disambiguation (NED)*[33] problem. In the setting of NED, a reference knowledge base (e.g. Wikipedia) is given. The entity candidate list, popularity scores and context representations are explicitly provided by the knowledge base. However, in enterprises, there are rarely any reference knowledge bases being available. The potential meanings of the acronyms need to be mined from plain text in the enterprise corpus. Moreover, the various

kinds of structural information (e.g. entity taxonomy, cross-document hyperlinks) within the reference knowledge base, is not available in enterprises.

5.2 Framework

We propose a novel end-to-end framework to solve the *Enterprise Acronym Disambiguation* problem. Our framework takes the enterprise corpus as input and produces a high-quality acronym disambiguation system as output. Figure 5.3 shows the details of our proposed framework. In the mining module, we will sequentially perform Candidates Generation, Popularity Calculation, Candidates Deduplication, Candidates Filtering and Context Harvesting on the input enterprise corpus. The details of these steps will be discussed in Section 5.3. After mining steps, we will get an acronym/meaning repository storing all the mined acronym/meaning pairs. Feed this repository together with the training data (which are automatically generated via distant supervision from the enterprise corpus) to the training module, we will get a candidate ranking model, a confidence estimation model and a final selection model. These models form the final acronym disambiguator and will be used in the testing module for actual acronym disambiguation. In the testing module, given the target acronym along with some context as input, the system will output the predicted meaning. Note that the mining and training module run offline once for the entire corpus or periodically

when the corpus update, while the testing (disambiguation) can be run online repeatedly for processing new documents.

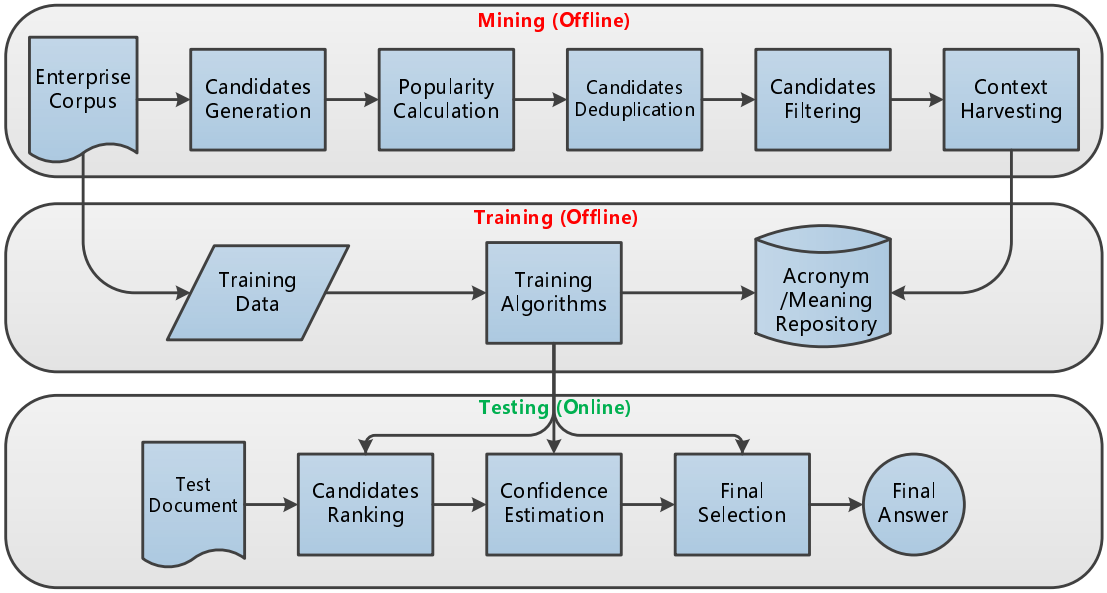


Figure 5.3: Framework

Our framework is a general framework. All the machine learning models in the framework are trained via distant supervised learning, thereby not requiring any manually labeled training data. The framework can be easily deployed to any enterprise domain to support high-quality acronym disambiguation, without requiring any domain knowledge.

5.3 Acronym Meaning Mining

In this section we describe the five steps of the offline mining module in details. The output of each step is the input to the subsequent step. The initial input to the first step is the enterprise corpus, and the eventual output of the last step is an acronym/meaning repository storing all the mined acronym/meaning pairs.

5.3.1 Candidates Generation

As there is no reference dictionary or knowledge base available in enterprise telling us the potential meanings of acronyms, we have to mine them from plain text. In this step we aim at generating an initial set of meaning candidates for the acronyms appearing in the enterprise corpus.

There are several strategies for candidate generation. The first one is *Aggressive Generation*. Using this strategy, whenever we see a phrase (within the enterprise corpus) whose initial letters match an acronym, we make it a meaning candidate for that acronym. Clearly this approach can cover all the candidates. But on the other hand, it will also introduce a huge amount of false candidates. For example, with this strategy, we will treat "Ancient India" as a candidate of "AI", which is clearly incorrect. To solve this problem, we may resort to another strategy called *Conservative Generation*. Under this strategy, a phrase is considered as a meaning candidate for an acronym if: (1) the initial letters of the

phrase match the acronym; **and** (2) the phrase and the acronym co-occur in at least one document in the enterprise corpus. The candidates generated from this strategy are much cleaner. However, due to the data sparsity issue in enterprises, many valid candidates (especially those public meanings such as "Artificial Intelligence") cannot be captured since they may never co-occur with the acronyms. For example, in Microsoft Corpus, "Artificial Intelligence" never co-occurs with "AI", but clearly we should not miss it. This shows that we cannot merely rely on the enterprise corpus to discover all the valid meanings for acronyms.

To achieve a balance between these two strategies, we propose a strategy called *Hybrid Generation* to be neither too aggressive nor too conservative. Namely, we treat a phrase as a meaning candidate for an acronym if: (1) it meets the condition of *Conservative Generation*; **or** (2) it is a valid candidate for the acronym in public knowledge bases (e.g. Wikipedia). The insight of this strategy is that the valid candidates missed by the *Conservative Generation* strategy are mainly public meanings which can be found in public knowledge bases. With this strategy we can make our system understand both the internal world and the external world, and get a good balance between precision and recall. In Section 5.5.4 we will experimentally evaluate the quality of the generated meaning candidates.

5.3.2 Popularity Calculation

As mentioned in Section 5.1.1, for each candidate meaning, we need to calculate its popularity score, which reveals how often the candidate meaning is used as the genuine referent meaning of the acronym. Previous research on *Entity Linking (EL)* shows that popularity is a very reliable indicator of correct entity disambiguation. In EL, popularity is calculated as the fraction of times a candidate being the target page for an anchor text in a reference knowledge base (e.g. Wikipedia). However, in enterprises, we do not have a knowledge base with anchor links. Therefore we cannot calculate popularity in the same way. Here we propose to calculate two types of popularity to mimic the effect.

1. *Marginal Popularity.*

   For each meaning candidate m_i, its marginal popularity score is defined as:

   $$MP(m_i) = \frac{Count(m_i)}{\sum_{j=1}^{n} Count(m_j)}, \tag{5.1}$$

   where m_1, m_2, ..., m_n are the n meaning candidates of acronym a and $Count(m_i)$ is the number of occurrences for meaning m_i in the corpus.

2. *Conditional Popularity.*

For each meaning candidate m_i, its conditional popularity score is defined as:

$$CP(m_i) = \frac{Count(m_i, a)}{\sum_{j=1}^{n} Count(m_j, a)}, \tag{5.2}$$

where m_1, m_2, ..., m_n are the n meaning candidates of acronym a and $Count(m_i, a)$ is the number of document-level co-occurrences for m_i and a in the corpus.

Marginal Popularity indicates the raw frequency of each meaning candidate, while Conditional Popularity indicates the popularity of each meaning candidate around the corresponding acronym. Conditional Popularity can more reasonably reveal how often the acronym is used to represent each meaning candidate. However, due to the data sparsity issue in enterprises, many valid candidates may get zero value for conditional popularity since they may never co-occur with the acronyms in the enterprise corpus. For example, in Microsoft Corpus, "Artificial Intelligence" never co-occurs with "AI", therefore its conditional popularity score is zero. The Marginal Popularity does not have this problem since it is calculated from the raw counts of the candidates. Therefore each candidate will have a non-zero marginal popularity score. But on the other hand, high marginal popularity score does not necessarily indicate high correlation between the candidate and the acronym. For example, among the candidates for acronym "IP", "IntPtr" has the largest marginal popularity score in our corpus, but actually it is not a popu-

lar meaning for "IP". As we can see, both Marginal Popularity and Conditional Popularity have their own advantages and disadvantages, and it is unclear how to combine the two scores into one popularity score. Therefore, we use both of them as features in the machine learning model for disambiguation.

5.3.3 Candidates Deduplication

In enterprises, people often create many variants (including abbreviations, plurals or even misspellings) for the same meaning, therefore many mined meaning candidates are actually equivalent. For example, for the meaning "Certificate Authority" of the acronym "CA", the variants include "Cert Auth", "Certificate Authorities" and many others. It is important to deduplicate these variants before sending them to the disambiguation module. The deduplication helps aggregate disambiguation evidences and reduce noises. Therefore in this step we aim at clustering the mined meaning candidates into groups such that each group corresponds to a distinct meaning. We design several rules based on string similarity, word stemming, word lemmatization and common prefix to perform the deduplication. Experiments show that the rules can accurately group the variants together. After grouping, we sort the variants within the same group based on their marginal popularity. Here we use marginal popularity instead of conditional popularity, because conditional popularity is very sparse and therefore many can-

didates may have zero values. The candidate with the largest marginal popularity is selected as the canonical candidate for the group. Other variants in the group will be deleted from the candidate list and their popularity scores will be aggregated to the canonical candidate. We maintain a table to record the variants for each canonical candidate. The table will be used for context harvesting in Section 5.3.5. Figure 5.4 illustrates an example of this process.

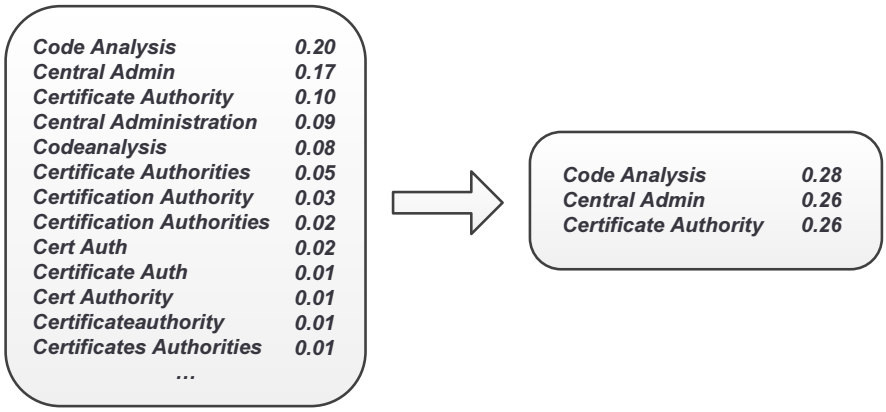


Figure 5.4: Candidates Deduplication Example

5.3.4 Candidates Filtering

After candidates deduplication, we observe that there still could be some remaining invalid candidates, most of which can be divided into the following three categories.

1. *Conflict*, i.e., the candidate contains a substring which is already a valid candidate for that acronym. For example, "Sharepoint Pro" is not a valid candidate for "SP" since its substring "Sharepoint" is already a valid candidate for "SP".

2. *Self Contained*, i.e., the candidate recursively contains the acronym. For example, "AD Display" is not a valid candidate for "AD" since it recursively contains "AD".

3. *All Acronyms*, i.e., all the component words of the candidate are acronyms. For example, "AD FS" is not a valid meaning for acronym "AF" since both "AD" and "FS" themselves are acronyms.

Note that not all candidates containing acronyms are invalid. For example, "SQL Server Configuration Manager" is a valid candidate for the acronym "SSCM". Therefore it is necessary to further divide candidates containing acronyms into Category 2, Category 3 and valid ones.

In this step we design a filtering function to test if a meaning candidate belongs to any of the above three categories. The candidates failing to pass the test will be filtered out. After this step we will get the final candidate list.

5.3.5 Context Harvesting

In this step, our goal is to harvest context words for each meaning candidate. These context words could be used to calculate context similarity with the query context and therefore provide important information for disambiguation. The context harvesting process is quite straightforward. For each distinct meaning candidate m, we put its canonical form and all its variants (from the variants table in Section 5.3.3) into set S. Then we scan the enterprise corpus, each time we find a match of any $e \in S$, we harvest the words in a width-W word window surrounding e as the context words of m. In our experiments we set the window size as 30.

As mentioned before, some popular public meanings (e.g., "Personal Computer") might be mentioned very rarely by their full names in the enterprise corpus since people directly use their acronyms most of the time. For example, the phrase "Personal Computer" appears less than ten times in our corpus, while "PC" are mentioned much more often and most of them indeed mean "Personal Computer". Therefore, the above context harvesting process can only get very few context words for those public meanings. Intuitively, we need to get more information from the external world about these public meanings. Since we are already leveraging Wikipedia to get the additional public meaning candidates, here we take one step further: for each public meaning, we add its Wikipedia

page's content as complementary context. By doing so, we ensure almost all valid candidates get a reasonable amount of context words.

5.4 Meaning Candidate Ranking

In this section we describe how to utilize the information from the mining steps to perform the actual disambiguation. We first train a candidate ranking model to get the initial ranking results. Then we train a confidence estimation model to decide whether the top 1 ranked result is reliable. And finally we train a final selection model to refine the ranking results for unconfident cases. These three models work together to form the ultimate acronym disambiguator, which can be used to perform high quality disambiguation for acronyms in enterprises.

5.4.1 Candidate Ranking

We first train a candidate ranking model. The goal of this model is to rank candidates with respect to the likelihood that each meaning candidate is the genuine meaning for the target acronym. In our system we make use of the learning-to-rank technique to train a supervised ranking model.

5.4.1.1 Training Data Generation

In order to train a robust ranking model, we need to get adequate amount of training data. The straightforward way to obtain training data is through manual labeling. However, the labeling process is very expensive and it requires a lot of domain knowledge. For each enterprise domain, we may need to find some domain experts to manually label a large amount of documents in that enterprise, which severely limits our framework's generalization capability. To tackle this problem, we propose to automatically generate training data via distant supervision. The intuition is that since acronyms and the corresponding meanings are semantically equivalent, people use them interchangeably in enterprises. Therefore we can fetch documents containing the meaning, then replace the meaning with the corresponding acronym and treat the meaning as ground truth. Figure 5.5 shows an example of this automatic training data generation process.

5.4.1.2 Training Algorithm

There are two strategies to train the ranking model. The first one is via classification. Namely, we label the ground truth meaning as "1" and all other candidate meanings as "0", then we adopt a classification algorithm (e.g. SVM [28]) to train a classifier. In testing, we run the classifier on each meaning candidate and rank them with respect to the likelihood of getting label "1". The other strategy is

utilizing the learning-to-rank technique. Namely, for each training instance, we label the ground truth meaning as "good" and all other candidate meanings as "bad", and then feed the full order ranking to a leaning-to-rank algorithm (e.g. Ranking SVM [59]) to train a ranking model. In testing, we run the ranker on the set of meaning candidates and directly obtain the ranking result. In our experiments, we tried both strategies on a validation dataset and found that the learning-to-rank version achieved a much better performance on our data. A possible explanation is that decisions for different acronyms can be quite different, e.g., for some acronyms, all the meaning candidates including incorrect ones could have relatively high values on features, while for some other acronyms, even the correct meanings may have low feature values. So it is more stable to learn a ranking for each instance rather than learn a global decision boundary for all the acronyms. In our system, we use the learning-to-rank algorithm LambdaMART [15] to train the ranking model. The model will calculate a score (indicating the likelihood of being the genuine meaning) for each meaning candidate, and then rank the candidates with respect to the scores.

5.4.1.3 Features

Now we explain the features we developed for the candidate ranking model. First, we have the *Marginal Popularity* score and *Conditional Popularity* score as

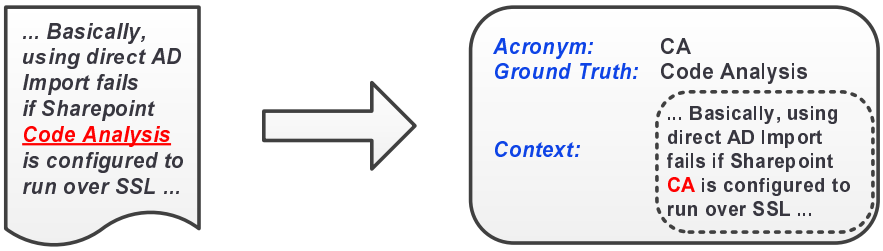


Figure 5.5: Distant Supervision Example

two context-independent features, which could compensate for each other. However, as discussed in the previous section, some popular public meanings (e.g., "Artificial Intelligence") can be rarely mentioned in enterprise corpus by their full names, therefore both their marginal popularity score and conditional popularity score can be very low. To address this, we add a third feature called *Wiki Popularity*, which is calculated from Wikipedia anchor texts to capture how often an acronym refers to a public meaning in Wikipedia. This feature is a very important feature commonly used in Entity Linking [57]. The fourth feature we adopt is *Context Similarity*. We convert the harvested context for the meaning and the query context of the target acronym into TFIDF vectors and then compute their cosine similarity. This feature is inspired by the observation that acronyms and their true meanings usually share similar context words. We also include two features (i.e. *LeftNeighborScore* and *RightNeighborScore*) to capture the effect of the immediate neighboring words, which are more important than further context words since immediate words could form phrases with the acronym. For example,

Feature	Description
MarginalPopularity	The meaning candidate's marginal popularity score
ConditionalPopularity	The meaning candidate's conditional popularity score
WikiPopularity	The meaning candidate's Wiki popularity score
ContextSimilarity	TFIDF cosine similarity between meaning context and acronym context
LeftNeighborScore	Probability that the acronym's immediate left word occurs to the immediate left of the meaning
RightNeighborScore	Probability that the acronym's immediate right word occurs to the immediate right of the meaning
FullNamePercentage	The percentage of the meaning candidate's component words appearing in the acronym context

Table 5.1: Candidate Ranking Features

if we see an acronym "SP" followed by the word "2", then likely it stands for "Service Pack". However, if we see "SP" followed by "2003", then probably its genuine meaning is "SharePoint". The last feature we use is *FullNamePercentage*. This feature is defined as the percentage of the meaning candidate's component words appearing in the context of the target acronym. For example, if the context for acronym "SP" contains the word "Service" but not the word "Pack", then this

feature score for candidate "Service Pack" is 0.5; and if "Service Pack" actually appears in the context, this feature will be 1. Table 1 summarizes the features used to train the candidate ranking model.

5.4.2 Confidence Estimation

After getting the ranking results, instead of directly returning the top ranked answer, we would like to apply a confidence estimation step to decide whether to trust the top ranked answer. There are two motivations behind. First, our candidate generation approach is not perfect, therefore we could encounter cases in which the genuine meaning is not in our candidate list. For such cases, the top ranked answer is obviously incorrect. Second, our training data is biased towards the internal meanings since external meanings may rarely appear with full names. As a result, the learned ranking model may lack the capability to properly rank the external meanings. We have tried to add some external resources (e.g., training examples sampled from Wikipedia) to balance the training data, but it does not help (actually making result worse) since within enterprises the usage of public meanings could be quite different from that in the external world. Therefore, the ranker may fail to rank the genuine public meaning at the top position. In such cases, we would better have the system return nothing or warn the user that the top result is not reliable rather than directly provide a wrong answer to mislead

the user. In this step, we train a confidence estimation model, which will estimate the top result's confidence, i.e., the probability of being correct.

Feature	**Description**
Top1Score	Top 1 ranked meaning candidate's ranking score
Top1&2ScoreDiff	Difference between 1st and 2nd ranked meaning candidates' ranking score
Top1&2CtxSimDiff	Difference between 1st and 2nd ranked meaning candidates' context similarity score
Top1WikiPopularity	Top 1 ranked meaning candidate's Wiki popularity score
MaxWikiPopularity	Maximum Wiki popularity score among all the meaning candidates
MaxWP&MPGap	Maximum difference between Wiki and marginal popularity among all the meaning candidates
MaxWP&CPGap	Maximum difference between Wiki and conditional popularity among all the meaning candidates

Table 5.2: Confidence Estimation Features

5.4.2.1 Training Data Generation

Similar to the ranker training, here the training data is also automatically generated. We just run the learned ranker on some distant labeled data (generated from a different corpus), and then check if the top ranked answer is correct or not. If it is correct, we generate a positive training example; otherwise we make a negative training example.

5.4.2.2 Training Algorithm

Any classification algorithms can be used to train the confidence estimation model. In our system we utilize the MART boosted tree algorithm [41] to train the model.

5.4.2.3 Features

We design 7 features to train the confidence estimation model. Table 2 summarizes these features. Basically there are two intuitions behind: (1) If the top-ranked answer's ranking score is very small, or the top-ranked answer's score is very close to the second-ranked answer's score, then the ranking is not very confident; (2) If the acronym has a dominating candidate in the public domain (e.g., "Personal Computer" is the dominating candidate for "PC"), and the candidates' Wiki popularity distribution is significantly different from their marginal/conditional

popularity distribution, then the ranker's output is not very confident. The first intuition handles Case 1 in Section 5.4.2 and covers the first 3 features in Table 2, while the second intuition handles Case 2 in Section 5.4.2 and covers the last 4 features in Table 2.

5.4.3 Final Selection

We have discussed that one of the most important motivations for confidence estimation is that the candidate ranking stage has some bias so it does not always rank public meanings at top when they are correct. Therefore, assuming the confidence estimation model can remove incorrect top-ranked result, we still need one additional step to decide if any public meaning is correct, which we call a final selection model. Specifically, in this step, we determine whether to return the most popular public meaning (based on Wiki Popularity) as the final answer, and this step is only triggered when the confidence estimator judges that the ranking result is unconfident. By combining confidence estimation and final selection, we are able to remove the bias as much as possible and make our system well understand both the internal world and the external world.

The goal of the final selection model is very similar to that of the confidence estimation model. In confidence estimation model, we aim at judging whether the top-ranked answer is the genuine answer; while in final selection model, we

want to know whether the most popular meaning in external world is the genuine answer. Thanks to this similarity, we can reuse the data and features in confidence estimation model training. We can take the same training data in Section 5.4.2.1 and update the labels correspondingly: if the genuine answer is the most popular meaning in the external world, we generate a positive training example; otherwise we make a negative training example. The training features are exactly the 7 features in Table 2 and the training algorithm is also the same as the confidence estimation model.

5.4.4 Iterative Popularity Calculation

As discussed in Section 5.3.2, neither *Marginal Popularity* nor *Conditional Popularity* can perfectly reveal how often the candidate meaning is used as the genuine referent meaning of the acronym. As a trade-off, we add both of them as ranking features. Now, with the joint work of candidate ranking, confidence estimation and final selection models, our acronym disambiguation system can achieve reasonably good performance. Therefore it is possible to run our system on the enterprise corpus for acronym disambiguation, and then use the results to recalculate popularity. We name this new popularity as *Iterative Popularity*. For each meaning candidate m_i of acronym a, its iterative popularity score is defined

as:

$$IP(m_i) = \frac{Count(a \text{ being disambiguated to } m_i)}{Count(a)}, \tag{5.3}$$

Compared with *Marginal Popularity*, *Iterative Popularity* can more accurately reflect the correlations between acronyms and meanings. Compared with *Conditional Popularity*, *Iterative Popularity* is less likely to suffer from the data sparsity problem. Figure 5.6 shows the comparison of conditional popularity and iterative popularity for the meaning candidates[2] of acronym "PC". Clearly the *Iterative Popularity* can more reasonably reveal how often the candidate meaning is used as the genuine referent meaning of the acronym.

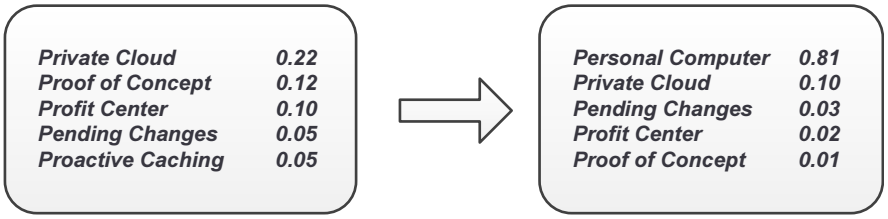


Figure 5.6: Conditional vs. Iterative

We add the *Iterative Popularity* as a new feature for candidate ranking and re-train all three models. Ideally we can use the new models to calculate new *Iterative Popularity* and iterative the process for several rounds. However, in practice we only execute one iteration due to efficiency reasons.

[2]We only show the top 5 candidates for each popularity

5.5 Experiments

In this section, we evaluate the effectiveness of our proposed method for the Acronym Disambiguation problem on several real-life datasets from Microsoft. We will: (1) demonstrate the quality of the acronym/meaning pairs harvested through our offline mining module; (2) compare the helpfulness of the features used in our ranking model; (3) show the performance gain from adding the iteratively updated popularity information; (4) illustrate the effectiveness of the confidence estimation module and the final selection module in our system; (5) compare the end-to-end disambiguation accuracy of our method, with two state-of-the-art Entity Linking methods. All the experiments, if not specifically mentioned, are conducted on a server with 2.40GHz Intel Xeon CPU and 12GB RAM.

5.5.1 Data

5.5.1.1 Mining and Training Corpus

We use both the Microsoft Answer Corpus (MAC) and the Microsoft Yammer Corpus (MYC) as the mining corpus. MAC contains 0.3 million web pages from a Microsoft internal question answering forum. While MYC is consisted of 6.8 million posts from Microsoft's Yammer social network. In total, our mining module harvested 5287 acronyms and 17258 meaning candidates from this joint corpus.

For model training, the confidence estimation model and final selection model need to be trained on a different corpus than the candidate ranking model. So we train the candidate ranking model on the Microsoft Answer Corpus, with 12500 training examples being automatically generated; and train the confidence estimation and final selection model on the Microsoft Yammer Corpus, with 40000 training instances being automatically generated.

5.5.1.2 Evaluation Datasets

We prepared four datasets for evaluation purposes. The first one is obtained from the recent pages of Microsoft answer forum. Note these pages are disjoint from those used as mining/training corpus. We randomly sampled 300 pages and filtered out pages which do not contain acronyms or only contain unambiguous acronyms (i.e., acronyms with a single meaning). After filtering, 240 test cases were left and we manually labeled the acronyms to their corresponding meanings.

The second one is generated via distant labeling on a corpus of Microsoft Office365 documents. We sampled 2000 documents which contain at least one occurrence of a meaning candidate. Then we replaced the meanings with the corresponding acronyms and treat the meanings as ground truths. We noticed that this automatically generated dataset contain a few bad cases where the acronym is unlikely to be used to represent the meaning (e.g., "AS" for "App Store"). We

manually checked through this dataset to remove such bad cases. This resulted in a test set of 1949 test cases.

Comparing the manually labeled evaluation data with the automatically generated evaluation data, the manually labeled one, although in smaller size, should more accurately evaluate the end-to-end system performance and real user experience, since the target acronyms in the manually labeled data are sampled from the real distribution, while in the automatic dataset acronyms are artificially generated from randomly sampled meanings. In another word, in the manual dataset the distribution of ground truth meanings follows the real conditional probability given the acronym, while in the automatic dataset, the distribution of ground truth follows the probability that the meaning is directly mentioned, which is very different from the real distribution. Three biases exist there: first, tail cases (meanings that are less likely to be represented by the acronym) occur more often than they should; second, public meanings occur less often than they should since their full names are less likely to be mentioned directly; and third, there will be no cases where the ground truth meaning is not discovered by the mining module, which could happen in practice. As we can see, the last two biases are directly related to what we want to address with final selection and confidence estimation, so the automatic evaluation data cannot accurately evaluate the performance of

these two modules. However, it can still help evaluate the candidate ranking module.

We also want to compare our method with the state-of-the-art Entity Linking (EL) systems based on public knowledge bases such as Wikipedia. However, it is unfair to directly compare with them as most enterprise specific meanings are unknown to them. Therefore, we need to only consider cases where the true meaning is a public meaning covered by both our system and the compared system. By filtering the distant dataset from Office365, we get the third dataset for comparing with the popular EL system Wikifier [85], and the fourth dataset for comparing with the popular EL system AIDA [51]. Since AIDA does not index any meanings for two-letter acronyms, the number of test cases in this dataset is much smaller and the disambiguation is much easier (reflected by the average number of candidates). Table 5.5.1.2 shows some basic statistics of these datasets.

	Manual	Distant	JoinW	JoinA
#Test Cases	240	1949	1659	237
Avg #Words	222.3	456	757.6	752.7
Avg #Candidates	15.94	22.39	29.55	4.52

Table 5.3: Basic statistics of the evaluation datasets for acronym disambiguation

5.5.2 Compared Methods

5.5.2.1 Baseline

Previous research on Entity Linking [85] shows that the popularity itself is a very reliable indicator of correct disambiguation, so we use it as a baseline in our experiments.

- **Max Marginal Popularity (MMP)**: Always selecting the candidate with maximum marginal popularity[3] as the disambiguation answer.

5.5.2.2 Ablations of Our System

We compare the following ablations of our system, to illustrate the effectiveness of the features and components.

- **Internal Popularity (IP)**: Only using the internal popularity (i.e., marginal popularity and conditional popularity) to train the ranking model.
- **Popularity (P)**: Using the internal popularity plus Wiki popularity to train the ranking model.
- **Popularity+Context (P+C)**: Using the popularity features plus the context similarity feature in ranking.

[3]Due to data sparsity, we didn't use conditional popularity.

- **Popularity+Context+Neighbbor (P+C+N)**: Using the popularity features, the context similarity feature and the immediate neighbor features to train the ranking model.

- **Popularity+ Context+ Neighbbor+ FullnamePerc (P+C+N+F)**: Using all previous features plus the fullname percentage feature to train the ranking model.

- **Popularity+Context+ Neighbbor+FullnamePerc+ IterPopularity (a.k.a. Candidate Ranker, or CR)**: Using all previous features plus the iterative popularity feature to train the ranking model.

- **Candidate Ranker+Confidence Estimator (CR+ CE)**: Using the candidate ranking model plus the confidence estimation model to make disambiguation decision.

- **Candidate Ranker+Confidence Estimator+Final Selector (a.k.a. Acronym Disambiguator, or AD)**: Using the candidate ranking model, the confidence estimation model and the final selection model to make decision. This is the full version of our system.

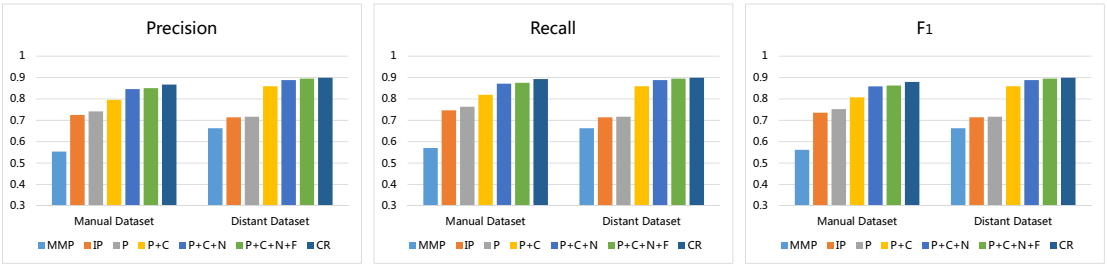


Figure 5.7: Ranking Performance

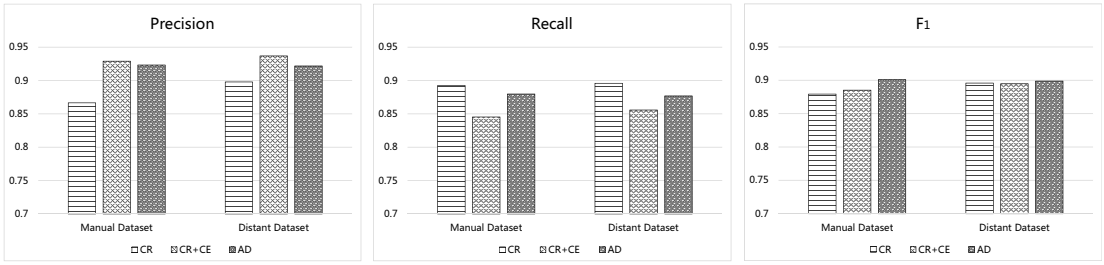


Figure 5.8: Effectiveness of Confidence Estimator and Final Selector

5.5.2.3 State-of-the-art EL Systems

We also compare our method with two state-of-the-art Entity Linking (EL) systems.

- **Wikifier**: a popular EL system using machine learning based strategy to combine various features together.
- **AIDA**: a robust EL system using weighted mention-entity graph to find the best joint mention-entity mapping.

5.5.3 Evaluation Measures

We use the following measures for evaluating the acronym disambiguation performance. Let $N_{Predicted}$ be the total number of test cases for which the system's disambiguation result is not empty, $N_{Correct}$ be the number of cases correctly disambiguated by the system, and N_{Total} be the total number of test cases.

- **Precision**: The Precision measures the accuracy of the disambiguation. It is defined as the percentage of correctly disambiguated cases among all predicted cases:

$$Precision = \frac{N_{Correct}}{N_{Predicted}}. \tag{5.4}$$

- **Recall**: The Recall measures the coverage of the disambiguation. It is defined as the percentage of correctly disambiguated cases among all test cases:

$$Recall = \frac{N_{Correct}}{N_{Total}}. \tag{5.5}$$

- $\boldsymbol{F_1}$: The F_1 measures the balance of precision and recall. It is defined as the harmonic mean of precision and recall:

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}. \tag{5.6}$$

5.5.4 Quality of Mined Acronym/Meaning Pairs

We first conduct experiments to evaluate the quality of the acronym/meaning pairs harvested through our offline mining module. Out of the 17258 mined pairs, we randomly sampled 2000 of them and asked 5 domain experts to manually check their validness. An acronym/meaning pair is considered as valid if the majority of the experts think the acronym is indeed used to abbreviate the meaning. For example, (*AS*, *Analysis Service*) is a valid pair, but (*AS*, *App Store*) is considered as invalid because people will not actually use *AS* to represent *App Store*. Among the sampled 2000 pairs, **94.5%** are labeled as valid, indicating our offline mining module can accurately extract acronym/meaning pairs from enterprise corpus. It is hard to precisely evaluate the coverage/recall of our mining method, since it is very difficult to obtain the complete meaning list for a given acronym. To get a rough idea, we randomly picked up 100 acronyms and asked the 5 domain experts to enumerate the valid meanings for these acronyms. In total we got 230 valid meanings and **all** of them are covered by the mined pairs. In Table 4 we present the mined meanings for several acronyms.

5.5.5 Disambiguation Performance

We then conduct experiments to evaluate the disambiguation performance of our ranking model against the baseline, and compare the helpfulness of the

Acronym	**Mined Meanings**
AI	Asset Intelligence, Application Insights, Artificial Intelligence, Appreciative Inquiry, Action Items, Activity Instance, Adobe Illustrator, American Idols, Air India
CA	Code Analysis, Central Administration, Certificate Authority, Client Authentication, Conditional Access, Corporate Accounts
ESA	European Space Agency, Entertainment Software Association, Employment and Support Allowance, External System Adaptor
MSE	Microsoft Security Essential, Managed Services Engine, Media Source Extension, Mean Squared Error, Madrid Stock Exchange
SP	SharePoint, Security Policy, Search Platform, Service Pack, Surface Pro, Stored Procedure, System Process, Service Point, Source Port

Table 5.4: Mined meanings for sample acronyms

features used in our ranking model. Figure 5.7 shows the precision, recall and F_1 of the compared methods on the *Manual* dataset and the *Distant* dataset. Our ranking model (**CR**) significantly outperforms the simple baseline (**MMP**) which barely ranks candidates with respect to the marginal popularity. In terms of the helpfulness of the features, the context similarity feature and the immediate

neighbor features contribute most to the performance gain. Other features are less helpful, yet still bring improvements to the overall performance. The Wiki popularity feature and the iterative popularity feature are more useful on the *Manual* dataset than on the *Distant* dataset. The reason is that the *Manual* dataset contains more public meanings (e.g., "Personal Computer") and these two features are specifically designed to help resolve acronyms to public meanings. In contrary, the *Distant* dataset is automatically generated via distant labeling. Since the public meanings may rarely appear in enterprise corpus with full names, they are less likely to appear in the *Distant* dataset.

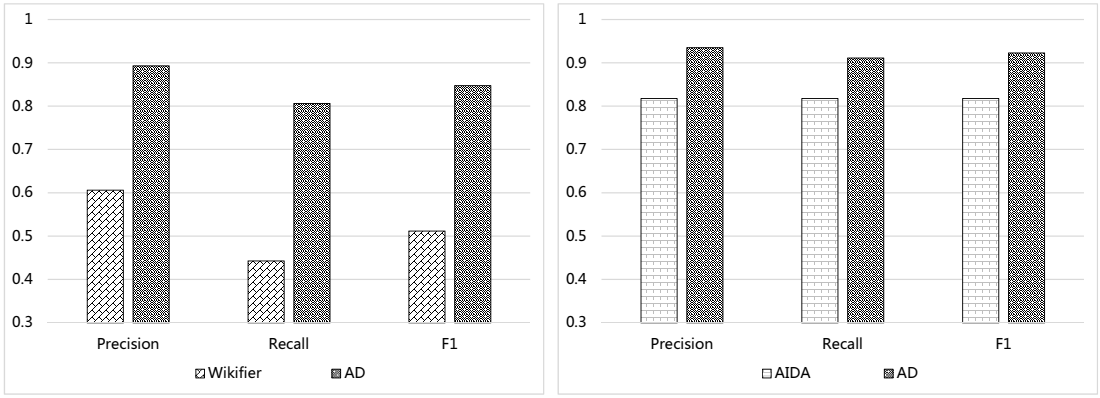
Next we conduct experiments to illustrate the effectiveness of the confidence estimation module and the final selection module in our system. Figure 5.8 shows the precision, recall and F_1 of the compared system configurations on the *Manual* dataset and the *Distant* dataset. As can be seen, the confidence estimation module can successfully reject unconfident predictions and therefore improve precision. On the other hand, few correct predictions (known as false negatives) may also be rejected due to the imperfectness of the confidence estimator. As a result, adding the confidence estimation module will decrease recall. Fortunately, the trained final selection module can help recover some of these false negatives. Therefore it will greatly improve recall, without sacrificing too much on precision. In terms of the F_1 measure, the final system (including both the confidence estimation module

and the final selection module) achieves the best performance. Again, these two modules are more helpful on the *Manual* dataset than on the *Distant* dataset. As we have discussed in Section 5.5.1.2, it is because of the bias in the *Distant* dataset.

5.5.6 Comparison with EL Systems

We also compare our system (**AD**) with two state-of-the-art Entity Linking (EL) systems: **Wikifier** and **AIDA**. As explained in Section 5.5.1.2, we made two datasets (i.e., *JoinW* and *JoinA*) for fair comparisons: all genuine meanings in the *JoinW* dataset are public meanings covered by both AD and Wikifier, and all genuine meanings in the *JoinA* dataset are public meanings covered by both AD and AIDA. Figure 5.9(a) and Figure 5.9(b) present the comparison of our AD system against Wikifer and AIDA, respectively. As we can see from the figures, AD significantly outperforms both Wikifier and AIDA on all three measures. The reason is that even for public meanings indexed by Wikifier and AIDA, the usage of them could be quite different in enterprises. Wikifier and AIDA utilize information from public knowledge bases (e.g., Wikipedia) to generate features, therefore can hardly capture such enterprise-specific signals. In contrast, our AD system mines disambiguation features directly from the enterprise corpus. As a result, it can

more accurately represent the characteristics of the enterprise and lead to much better disambiguation performances.



(a) *Wikifier vs. AD* (b) *AIDA vs. AD*

Figure 5.9: Comparison with EL Systems

5.6 Summary

In this chapter, we studied the *Enterprise Acronym Disambiguation* problem. We proposed a novel, end-to-end framework to solve this problem. Our framework takes the enterprise corpus as input and produces a high-quality acronym disambiguation system as output. The disambiguation models are trained via distant supervised learning, without requiring any manually labeled training examples. And the system is capable of resolving acronyms to both enterprise-specific meanings and public meanings. To evaluate the effectiveness of our method, a

thorough experimental study was conducted on Microsoft enterprise data. The experimental results demonstrated that our proposed method can effectively construct acronym/meaning repositories and accurately disambiguate acronyms to their genuine meanings with over 90% precision. Furthermore, our proposed framework can be easily deployed to any enterprises or closed-domain corpus to support high-quality acronym disambiguation, without requiring any domain knowledge.

Chapter 6

Answering Elementary Questions via Coherent Scene Extraction

In this chapter, we study how to leverage the existing background knowledge for better understanding and answering elementary science questions. Elementary grade science tests are interesting as they test a wide variety of commonsense knowledge that human beings largely take for granted, yet are quite challenging for machines. One particular hardness there is that not all knowledge required to answer the questions is explicitly stated in the question context. Inspired by the fact that people can use background knowledge to fill in the implicit information, we study how to leverage the existing knowledge bases as machine's brain so that we can connect the information gap. In this chapter we propose an approach to

build knowledge graphs jointly from question context and background knowledge bases, and then extract coherent scenes from the graphs as a proper interpretation of the question/answer pair. The scene's coherent score can naturally reflect each answer option's possibility of being correct. Experimental results on real elementary science questions justified the effectiveness of our method. The work in this chapter is published in [64].

6.1 Background and Preliminary Material

Elementary grade science tests are interesting as they test a wide variety of commonsense knowledge that human beings largely take for granted, yet are very challenging for machines [23, 26]. A system's ability to answer such elementary science questions can naturally reflect its level of intelligence.

The elementary science questions are quite different from the entity-centric factoid questions [40, 108] that are extensively studied in the question answering (QA) community. As can be seen from the two examples in Figure 6.1, the entity-centric questions are about explicit entities and the answer are usually directly searchable from the factoid databases (e.g. Freebase). While for elementary science questions, they are mainly about general concepts. Its answer lookup also relies on many implicit knowledge and needs more logical reasoning.

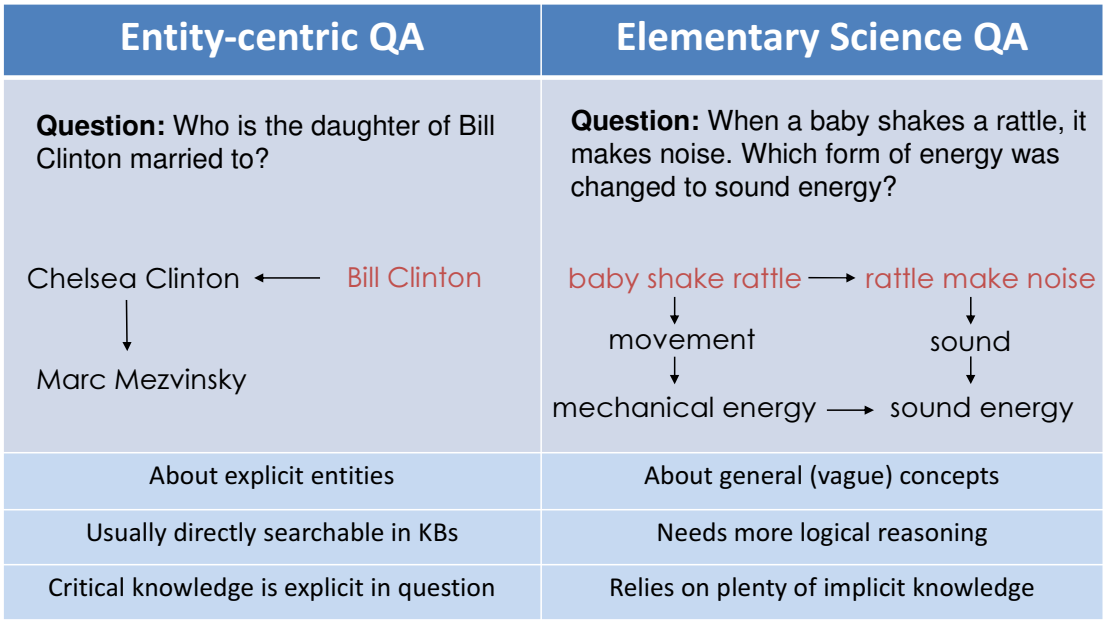


Figure 6.1: Entity-centric QA vs. Elementary QA

One particular challenge for elementary science QA is that not all knowledge required to answer the questions is explicitly stated in the question context. However, our human beings can still easily understand and solve the questions, by leveraging the background knowledge in our brains to fill in the implicit knowledge and reconstruct the scene of the problem. Inspired by this, we study how to make machines mimic the process to better answer these questions. In this work we leverage the existing knowledge bases (KBs) as machine's brain for connecting the information gap. Then we can build a knowledge graph jointly from question context and relational tuples (e.g. animal <eat> food) in background KBs. In such a graph nodes are words (or concepts) and edges encode the relationships be-

tween words. The correct answer to the question should be richly connected with other nodes in the graph so that they can form a dense coherent scene. While for wrong answer options, their connections with other nodes should be much weaker. For example, consider the following multiple choice question taken from a 4th grade exam:

Question 1. *Animals get energy for growth and repair from (A) food (B) air (C) soil (D) water*

Figure 6.2 shows the coherent scene for answer option "food" and "air". As cen be seen, the correct answer "food" is densely connected in the scene, while the wrong answer "air"'s connection in the scene is much sparser.

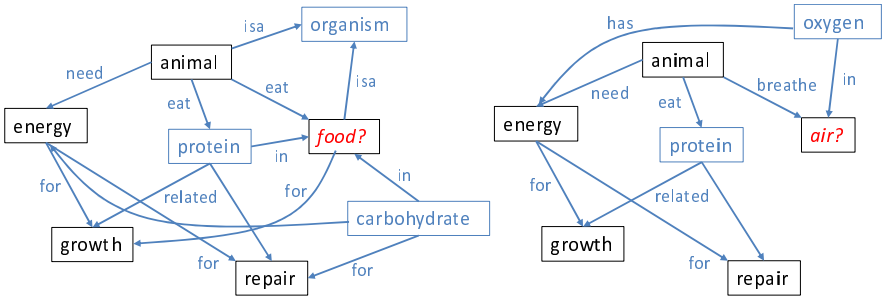


Figure 6.2: Coherent Scene Examples

In this chapter, we design a novel, end-to-end framework to solve *Elementary Science Question Answering* problem, via extracting coherent scene from knowledge graph. To the best of our knowledge, this is the first work of leveraging

knowledge graph for filling in implicit knowledge to better answer elementary science questions. The extracted coherent scene can also be viewed as the proper interpretation of the question/answer pairs.

6.1.1 Problem Statement

We formalize the *Elementary Science Question Answering via Coherent Scene Extraction* problem as follows.

Definition 5 (Elementary Science Question Answering via Coherence Scene Extraction)**.** *Elementary Science Question Answering via Coherent Scene Extraction is the task of finding the correct answer to an elementary-level science question, by extracting the coherent scene as interpretation of each question/answer pair. Given an elementary science question q, four answer options O_i (i=1,2,3,4), one or more background knowledge base $\{K\}$, the task is to build a knowledge graph G jointly from q and $\{K\}$, and then extract a coherent dense subgraph S_i from G for each answer option O_i. Finally the four answer options will be ranked with respect to the density of the corresponding S_i.*

6.2 Approach

We propose a novel end-to-end framework to solve the *Elementary Science Question Answering* problem via coherent scene extraction from knowledge graph. Our framework takes a multiple choice elementary science question along with one or more background knowledge base(s) as input and produces a ranked list of answer options as output. Figure 6.3 shows the details of our proposed framework. Given an elementary science question, we first use the *question analyzer* to get keywords from it. Then we apply the *elaborator* to inject implicit knowledge from the background knowledge bases, for forming an elaborated knowledge graph together with the question keywords. After that we sequentially put each answer option into the graph and utilize *scene extractor* to filter out unrelated knowledge and get the final coherent scene. Finally we rank the answer options with respect to a score derived from the coherent scene and choose the top ranked one as the predicted answer.

6.2.1 Question Analyzer

In this step we aim at doing a basic analysis for the question and extracting keywords (along with their importance scores) from it. There are different ways for keywords extraction, ranging from lightweight frequency based methods to heavyweight parsing based methods. For efficiency reason, here we adopt a

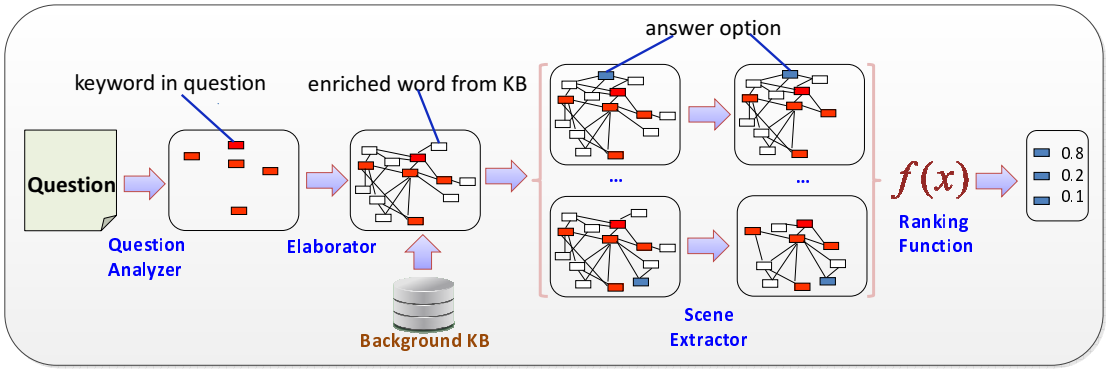


Figure 6.3: Framework

frequency based method via leveraging the Web corpus. We send the question to a web search engine (e.g. Google) and group the top 20 snippets as a single document d. For each non-stop word w in the question, we calculate its importance score (IS) as follows:

$$IS(w) = \frac{tf_d(w)}{df_Q(w)}, \tag{6.1}$$

where $tf_d(w)$ is the term frequency of w in document d, and $df_Q(w)$ is the document frequency of w in question set Q containing all the elementary science questions. The intuition here is that the words frequently mentioned in variations of the question should be important, while the words (e.g. "following", "example") which are widely used in many questions should be penalized.

6.2.2 Elaborator

In this step our goal is to inject necessary implicit knowledge from the background KBs, and utilize them to form an elaborated knowledge graph together with the question keywords. We take the keywords set K from *Question Analyzer* as input and fetch all the neighbor words which are directly connected with any keyword $kw \in K$ in the background KBs. As we may get huge number of such neighbor words, we perform an initial ranking of them and only retain the top ranked ones. Here we use the aggregated relatedness with question keywords as the ranking measure. For each neighbor word w, its ranking score is:

$$score(w) = \sum_{kw \in K} IS(kw) * rel(kw, w), \tag{6.2}$$

where $IS(kw)$ is the importance score of keyword kw and $rel(kw, w)$ is the relatedness score between kw and w. In this work we use the cosine similarity between word vectors to measure two words' relatedness. After the ranking, the retained top-N[1] neighbor words will form a knowledge graph together with the question keywords. In the graph, two words are connected if they are related in the background KBs. The edge weights in the graph is defined as the relatedness score between the two end words.

[1]N is empirically set as 6*(#keywords in question) in our tests.

Note that the elaboration process is independent of any answer option. As we don't know which answer option is correct, it is inappropriate to involve any of them into the elaboration process.

6.2.3 Scene Extractor

Since the answer options are independent of the previous elaboration process, we have to sequentially incorporate each of them into the knowledge graph. There will be an edge between the option word and any of the existing word in the graph if they are related in the background KBs.

After incorporating the answer option, we look into the elaborated graph and find that the knowledge in the graph tends to be too broad. Some of the enriched words are related to the question, but not quite related to finding the answer of the question. In order to facilitate the question answering goal, only the necessary words bridging the question and the answer option should be kept. Therefore we introduce a scene extraction step to further remove useless knowledge from the graph and make it bias towards the question/option pair. Our goal is to compute a dense subgraph (i.e. the coherent scene) that would ideally contain all the question keywords, the answer option and a few enriched words which are richly connected with them. As we assume each node in the dense subgraph should be critical and essential, we need to capture even the weakest node in the desired

subgraph. For this purpose, we define the weighted degree of a node in the graph as the summed weight of its incident edges. Then we define the density of a subgraph to be the *minimum weighted degree* among its nodes. Our goal is then equivalent to extracting a subgraph with maximum density from the elaborated graph.

Inspired by Sozio et al's work [95] on finding strongly interconnected subgroups in social networks, we develop an iterative node removal algorithm for extracting coherent scene. The algorithm starts from the elaborated graph and iteratively removes the non-keywords node with the minimum weighted degree, until the answer option node is about to be removed. Among the subgraphs obtained during the iterative steps, the one with the largest *minimum weighted degree* will be returned as the coherent scene.

6.2.4 Ranking Function

After obtaining the coherent scene for each answer option, we design a ranking function to calculate a score from the coherent scene and then use the score to rank the answer options. Here our assumption is that the answer option being more densely connected in the coherent scene is more likely to be the correct answer. So we define the ranking score as the density or the *minimum weighted*

degree of each answer option's coherent scene. The option with largest ranking score is selected as the predicted answer to the question

6.3 Experiments

In this section, we evaluate the effectiveness of our proposed method on several real elementary science question datasets. All the experiments, if not specifically mentioned, are conducted on a laptop with 2.8GHz Intel Core i7 CPU and 16GB RAM.

6.3.1 Evaluation Datasets

We prepare four real elementary science question datasets for evaluation purpose. The first three are for grade 4 and the last one is for grade 5. Since our approach requires that the answer option can be represented as a single node in the knowledge graph, we only retain the questions where all four answer options are single words or phrases. Table 6.3.1 shows some basic statistics of these datasets.

6.3.2 Baselines

We compare the end-to-end question answering accuracy of our method (denoted as **SceneQA**) with the following baselines:

	Regents1	Regents2	Viceroy	Grade5
Grade	4	4	4	5
# questions	47	23	26	197

Table 6.1: Basic statistics of the evaluation datasets

- **Aggregated Relatedness (AR)**: Rank answer options with respect to the aggregated relatedness with question keywords (Equation 6.2).

- **Aggregated Relatedness with Elaboration (ARE)**: Rank answer options with respect to the aggregated relatedness with all the nodes in the elaborated knowledge graph.

- **Scene Extraction without using Elaboration (Scene-)**: Rank answer options with respect to the density of the coherent scene extracted from the original keywords knowledge graph.

- **Arizona**: Combines language models (likelihood of answer option given question) and information retrieval scores (score of top retrieved sentence matching question plus option) using an SVM ranker, trained on a set of questions with known answers. [56]

- **A***: "Prove" answer option from question by applying lexical inference rules automatically extracted from science texts. Select the option with the strongest "proof". [24]

- **1Rule**: A* restricted to use just a single rule (question → answer option) in the proof. [24]

6.3.3 Experiments Setup

In this work, we utilize the union of WordNet [68], FreeAssociation [76] and DART [25] as our method's background KBs. For relatedness measure, we train word vectors on 4th grade textbook (along with the similar sentences retrieved from Web search engine) using the word2vec [66] tool and then calculate the cosine similarity between word vectors to measure two words' relatedness. For all the baseline methods, we adopt the default settings if there are any.

6.3.4 Performance Comparison

Table 6.3.4 shows the performance comparison (in terms of question answering accuracy) of our proposed method with all the baseline methods, on the evaluation datasets.

From the comparison we can see that our proposed method **SceneQA** outperforms the baselines on almost all the datesets, except on *Regents2* it performs

	Regents1	Regents2	Viceroy	Grade5
AR	59.57	65.22	42.31	50.25
ARE	65.96	**69.57**	42.31	51.78
Scene-	70.74	57.61	47.12	50.13
Arizona	65.96	58.70	28.85	30.08
A*	65.96	67	47	29.22
1Rule	73.57	63.74	42.92	31.21
SceneQA	**83.51**	66.30	**65.38**	**55.20**

Table 6.2: QA Accuracy Comparison

slightly worse than **ARE**. The comparison of **SceneQA** and **Scene-** clearly illustrates that the implicit knowledge is critical for correctly answering the elementary science questions. And the performance gain of **SceneQA** over **ARE** justifies that our scene extraction algorithm is effective in leveraging the most useful implicit knowledge from the background KBs.

6.3.5 Error Analysis

We then conduct some error analysis for our method and find that the false answered questions mainly belong to two categories: (1) There are two answer options with exactly the opposite meanings and one of them is correct. Examples

include Question 2 and Question 3. Since the relatedness measure we use (i.e. word2vec) cannot distinguish words with similar distributional semantics (e.g. antonyms), our method cannot confidently identify which one is correct. (2) The question is about a "process". Examples include Question 4 and Question 5. Since in our method, the representation of the coherent scene is still based on bag of words, it is not quite aware of word orders and therefore incapable of representing a "process" or "sequence".

Question 2. *The process that changes a gas to liquid is called (A) condensation (B) melting (C) evaporation (D) vaporization*

Question 3. *An animal that has a backbone is called a(n) (A) invertebrate (B) vertebrate (C) exoskeleton (D) sponge*

Question 4. *A pot is heated on a stove. Which process causes the metal handle of the pot to also become hot? (A) combustion (B) convection (C) radiation (D) conduction*

Question 5. *Baby chicks peck their way out of their shells when they hatch. This activity is an example of which of the following types of behavior? (A) instinctive (B) learn (C) plan (D) social*

6.4 Summary

In this chapter we studied the *Elementary Science Question Answering* problem. We proposed a novel, end-to-end framework to solve the problem via extracting coherent scene from knowledge graph. The experimental results on real elementary science questions demonstrated that our proposed method can effectively fill in the implicit knowledge required to answer questions. The extracted coherent scene can also facilitate the interpretation of the question/answer pairs.

Chapter 7

Conclusions and Future Directions

7.1 Conclusions

Connecting text with knowledge is an important research topic with many interesting real-world applications. Text can be seen as both the source and the destination of knowledge. In one direction, knowledge can be extracted from text, in the other direction, knowledge can be leveraged to understand text. In between is the entity linking process to enrich text with knowledge. It links text mentions to corresponding entries in a reference knowledge base, so that semantic information can be transferred from KB to text, and then new knowledge can be distilled

from text to complete the KB. In this thesis, we studied the connecting text with knowledge problem from two perspectives: (1) For enriching text with knowledge via entity linking, this thesis extended the current literature by alleviating the complex requirements for the underlying reference knowledge bases. Efficient and robust algorithms are further introduced for mining evidences from plain text to bridge the information gap between text and knowledge. The fundamental motivation is to overcome the limitations of existing entity linking methods and make linking text to knowledge possible at any circumstances. (2) For leveraging knowledge for understanding text, this thesis described the coherent scene extraction algorithm to utilize background knowledge for filling in the implicit yet critical information in text, and showed its helpfulness in answering elementary science questions.

Chapter 3 studied mining additional evidences from external corpus to overcome the incompleteness of reference knowledge bases. A generative model and an incremental algorithm was proposed to automatically mine useful evidences across documents. With a specific modeling of "background topic" and "unknown entities", the model is able to harvest useful evidences from noisy text, which can be leveraged to significantly boost the disambiguation performance.

Chapter 4 studied linking entity mentions to linkless knowledge bases. A generative model was introduced to leverage the latent evidences scattered across

the reference knowledge base to mimic the effects of the missing cross-document links. Experimental results on real data justified the effectiveness of the model.

Chapter 5 studied linking acronyms in enterprises to their corresponding meanings. The challenge is that there are no reference knowledge bases being available. An end-to-end framework was designed to mine acronym meanings from enterprise corpus and then disambiguate acronyms to the mined meanings. The disambiguation models are trained via distant supervised learning, without requiring any manually labeled training examples.

Chapter 6 studied leveraging background knowledge for better understanding and answering elementary science questions. The coherent scene extraction algorithm was developed to utilize background knowledge for filling in the implicit information in question. The extracted coherent scene can not only help rank answer options, but also facilitate the interpretation of each question/answer pair.

7.2 Future Directions

An important future direction is leveraging deep learning inspired text embeddings [5, 72, 50, 27, 11, 12, 54, 93, 67, 49, 81, 94, 63] for faster and better entity linking. The distributed representation learning of textual objects such as words, phrases, and documents with deep learning techniques has received a lot of research interests recently. Such embeddings are able to encode the semantic and

structural information of textual objects in a high dimensional space. Then the various kinds of similarity measures (e.g. context similarity, topical coherence) used in entity linking can be potentially replaced by fast calculation of embedding distances. However, challenges arise with utilization of such embeddings. First, those embeddings are based on distributional semantics, therefore it is hard to distinct text segments with opposite meanings. Second, the current techniques for phrase and document embeddings are still not quite satisfactory. And finally, such embedding techniques can hardly be applied to tail entities with very limited descriptions. How to tackle these challenges is an interesting future work.

Another interesting future direction is injecting more structural or relational information into the entity linking process. We observed that in many cases the critical evidences for correct entity linking are relational information (e.g. coreference chain [21, 34, 45], relations between entity mentions [21, 19]), which can hardly be captured by bag-of-words methods. Topical coherence is helpful for partially and implicitly covering such relational information, however, the link structure required by topical coherence calculation is not always available. It is interesting to study how to explicitly leverage the coreference chains and relations extracted by existing NLP tools to facilitate better entity linking. How to jointly perform entity linking with other NLP tasks (e.g. coreference resolution [34, 45], relation extraction [19]) is also worth investigating.

Finally, it is interesting to study the interaction between text corpus and knowledge bases via entity linking. On one hand, linking entity mentions in text corpus to entries in knowledge bases could add more semantic information to text and facilitate better understanding of text. On the other hand, the accurate understanding of text could lead to more high-quality knowledge extraction from text to complete existing knowledge bases. How to design effective framework to make text understanding and knowledge base completion iteratively benefit each other is a future work we would like to explore.

Bibliography

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

[2] Eytan Adar. Sarad: A simple and robust abbreviation dictionary. *Bioinformatics*, 20(4):527–533, 2004.

[3] Hiroko Ao and Toshihisa Takagi. Alice: an algorithm to extract abbreviations from medline. *Journal of the American Medical Informatics Association*, 12(5):576–586, 2005.

[4] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. *The Semantic Web*, pages 722–735, 2007.

[5] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

[6] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *Proceedings of EMNLP*, pages 1533–1544, 2013.

[7] I. Bhattacharya and L. Getoor. A latent dirichlet model for unsupervised entity resolution. pages 509–518, 2006.

[8] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[9] John Blitzer, Mark Dredze, Fernando Pereira, et al. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL*, volume 7, pages 440–447, 2007.

[10] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of SIGMOD*, pages 1247–1250, 2008.

[11] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. Joint learning of words and meaning representations for open-text semantic parsing. In *International Conference on Artificial Intelligence and Statistics*, pages 127–135, 2012.

[12] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of NIPS*, pages 2787–2795, 2013.

[13] Eric Brill, Susan Dumais, and Michele Banko. An analysis of the askmsr question-answering system. In *Proceedings of EMNLP*, pages 257–264, 2002.

[14] R. Bunescu and M. Pasca. Using encyclopedic knowledge for named entity disambiguation. In *Proceedings of EACL*, pages 9–16, 2006.

[15] Christopher JC Burges. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11:23–581, 2010.

[16] Zhiyuan Cai, Kaiqi Zhao, Kenny Q Zhu, and Haixun Wang. Wikification via link co-occurrence. In *Proceedings of CIKM*, pages 1087–1096, 2013.

[17] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E.R. Hruschka Jr, and T.M. Mitchell. Toward an architecture for never-ending language learning. In *Proceedings of AAAI*, pages 1306–1313, 2010.

[18] C. Chemudugunta and P.S.M. Steyvers. Modeling general and specific aspects of documents with a probabilistic topic model. In *Proceedings of NIPS*, pages 241–248, 2007.

[19] Liwei Chen, Yansong Feng, Jinghui Mo, Songfang Huang, and Dongyan Zhao. Joint inference for knowledge base population.

[20] Z. Chen and H. Ji. Collaborative ranking: A case study on entity linking. In *Proceedings of EMNLP*, pages 771–781, 2011.

[21] X. Cheng and D. Roth. Relational inference for wikification. In *Proceedings of EMNLP*, pages 1787–1796, 2013.

[22] R.L. Cilibrasi and P.M.B. Vitanyi. The google similarity distance. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):370–383, 2007.

[23] Peter Clark. Elementary school science and math tests as a driver for ai: Take the aristo challenge! In *Twenty-Seventh IAAI Conference*, 2015.

[24] Peter Clark, Niranjan Balasubramanian, Sumithra Bhakthavatsalam, Kevin Humphreys, Jesse Kinkead, Ashish Sabharwal, and Oyvind Tafjord. Automatic construction of inference-supporting knowledge bases. In *Proceedings of AKBC*, 2014.

[25] Peter Clark and Phil Harrison. Large-scale extraction and use of knowledge from text. In *Proceedings of the fifth international conference on Knowledge capture*, pages 153–160, 2009.

[26] Peter Clark, Philip Harrison, and Niranjan Balasubramanian. A study of the knowledge base requirements for passing an elementary science test. In *Proceedings of AKBC*, pages 37–42, 2013.

[27] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[28] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[29] Mark Craven, Johan Kumlien, et al. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of ISMB*, pages 77–86, 1999.

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

[31] Nilesh Dalvi, Ravi Kumar, and Bo Pang. Object matching in tweets with spatial models. In *Proceedings of WSDM*, pages 43–52, 2012.

[32] Nilesh Dalvi, Ravi Kumar, and Mohamed Soliman. Automatic wrappers for large scale web extraction. *Proceedings of the VLDB Endowment*, 4(4):219–230, 2011.

[33] M. Dredze, P. McNamee, D. Rao, A. Gerber, and T. Finin. Entity disambiguation for knowledge base population. In *Proceedings of ICCL*, pages 277–285, 2010.

[34] Greg Durrett and Dan Klein. A joint model for entity analysis: Coreference, typing, and linking. *Transactions of the Association for Computational Linguistics*, 2:477–490, 2014.

[35] A. Fader, S. Soderland, and O. Etzioni. Identifying relations for open information extraction. In *Proceedings of EMNLP*, pages 1535–1545, 2011.

[36] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open question answering over curated and extracted knowledge bases. In *Proceedings of SIGKDD*, pages 1156–1165, 2014.

[37] Miao Fan, Deli Zhao, Qiang Zhou, Zhiyuan Liu, Thomas Fang Zheng, and Edward Y Chang. Distant supervision for relation extraction with matrix completion. In *Proceedings of ACL*, volume 1, pages 839–849, 2014.

[38] Shicong Feng, Yuhong Xiong, Conglei Yao, Liwei Zheng, and Wei Liu. Acronym extraction and disambiguation in large-scale organizational web pages. In *Proceedings of CIKM*, pages 1693–1696, 2009.

[39] Paolo Ferragina and Ugo Scaiella. Tagme: on-the-fly annotation of short text fragments (by wikipedia entities). In *Proceedings of CIKM*, pages 1625–1628, 2010.

[40] David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79, 2010.

[41] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[42] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, pages 1–12, 2009.

[43] S. Gottipati and J. Jiang. Linking entities to a knowledge base with query expansion. In *Proceedings of EMNLP*, pages 804–813, 2011.

[44] A. Haghighi and L. Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of ACL-HLT*, pages 362–370, 2009.

[45] Hannaneh Hajishirzi, Leila Zilles, Daniel S Weld, and Luke S Zettlemoyer. Joint coreference resolution and named-entity linking with multi-pass sieves. In *Proceedings of EMNLP*, pages 289–299, 2013.

[46] X. Han and L. Sun. A generative entity-mention model for linking entities with knowledge base. In *Proceedings of ACL-HLT*, pages 945–954, 2011.

[47] X. Han and L. Sun. An entity-topic model for entity linking. In *Proceedings of EMNLP*, pages 105–115, 2012.

[48] X. Han, L. Sun, and J. Zhao. Collective entity linking in web text: a graph-based method. In *Proceedings of SIGIR*, pages 765–774, 2011.

[49] Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. Learning entity representation for entity disambiguation. In *Proceedings of ACL*, pages 30–34, 2013.

[50] Geoffrey Hinton and Ruslan Salakhutdinov. Discovering binary codes for documents by learning deep generative models. *Topics in Cognitive Science*, 3(1):74–91, 2011.

[51] J. Hoffart, M.A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *Proceedings of EMNLP*, pages 782–792, 2011.

[52] Johannes Hoffart, Stephan Seufert, Dat Ba Nguyen, Martin Theobald, and Gerhard Weikum. Kore: keyphrase overlap relatedness for entity disambiguation. In *Proceedings of CIKM*, pages 545–554, 2012.

[53] Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of ACL*, pages 541–550, 2011.

[54] Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of ACL*, pages 873–882, 2012.

[55] Alpa Jain, Silviu Cucerzan, and Saliha Azzam. Acronym-expansion recognition and ranking on the web. In *Information Reuse and Integration*, pages 209–214, 2007.

[56] Peter Jansen, Mihai Surdeanu, and Peter Clark. Discourse complements lexical semantics for non-factoid answer reranking. In *Proceedings of ACL*, pages 977–986, 2014.

[57] H. Ji and R. Grishman. Knowledge base population: Successful approaches and challenges. In *Proceedings of ACL*, pages 1148–1158, 2011.

[58] Yuzhe Jin, Emre Kıcıman, Kuansan Wang, and Ricky Loynd. Entity linking at the tail: Sparse signals, unknown entities and phrase models. In *Proceedings of WSDM*, 2014.

[59] Thorsten Joachims. Optimizing search engines using clickthrough data. In *Proceedings of SIGKDD*, pages 133–142, 2002.

[60] S.S. Kataria, K.S. Kumar, R. Rastogi, P. Sen, and S.H. Sengamedu. Entity disambiguation with hierarchical topic models. In *Proceedings of SIGKDD*, pages 1037–1045, 2011.

[61] Jeongwoo Ko, Eric Nyberg, and Luo Si. A probabilistic graphical model for joint answer ranking in question answering. In *Proceedings of SIGIR*, pages 343–350, 2007.

[62] Leah S Larkey, Paul Ogilvie, M Andrew Price, and Brenden Tamilio. Acrophile: an automated acronym extractor and server. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 205–214, 2000.

[63] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of ICML*, pages 1188–1196, 2014.

[64] Yang Li and Peter Clark. Answering elementary science questions by constructing coherent scenes using background knowledge. In *Proceedings of EMNLP*, 2015.

[65] Yang Li, Chi Wang, Fangqiu Han, Jiawei Han, Dan Roth, and Xifeng Yan. Mining evidences for named entity disambiguation. In *Proceedings of SIGKDD*, pages 1070–1078, 2013.

[66] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of ICLR*, 2013.

[67] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, pages 3111–3119, 2013.

[68] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[69] D. Milne and I.H. Witten. Learning to link with wikipedia. In *Proceedings of CIKM*, pages 509–518, 2008.

[70] Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. Distant supervision for relation extraction with an incomplete knowledge base. In *Proceedings of NAACL-HLT*, pages 777–782, 2013.

[71] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of ACL*, pages 1003–1011, 2009.

[72] Andriy Mnih and Geoffrey Hinton. Three new graphical models for statistical language modelling. In *Proceedings of ICML*, pages 641–648, 2007.

[73] Alexander A Morgan, Zhiyong Lu, Xinglong Wang, Aaron M Cohen, Juliane Fluck, Patrick Ruch, Anna Divoli, Katrin Fundel, Robert Leaman, Jörg Hakenberg, et al. Overview of biocreative ii gene normalization. *Genome biology*, 9(Suppl 2):S3, 2008.

[74] David Nadeau and Peter D Turney. A supervised learning approach to acronym identification. In *Proceedings of the 18th Canadian Society conference on Advances in Artificial Intelligence*, pages 319–329, 2005.

[75] Roberto Navigli. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):10:1–10:69, February 2009.

[76] Douglas L Nelson, Cathy L McEvoy, and Thomas A Schreiber. The university of south florida free association, rhyme, and word fragment norms. *Behavior Research Methods, Instruments, & Computers*, 36(3):402–407, 2004.

[77] Serguei Pakhomov, Ted Pedersen, and Christopher G Chute. Abbreviation and acronym disambiguation in clinical discourse. In *AMIA Annual Symposium Proceedings*, pages 589–593, 2005.

[78] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of EMNLP*, pages 79–86, 2002.

[79] Patrick Pantel and Ariel Fuxman. Jigs and lures: Associating web queries with structured entities. In *Proceedings of ACL-HLT*, pages 83–92, 2011.

[80] Youngja Park and Roy J Byrd. Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of EMNLP*, pages 126–133, 2001.

[81] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. *Proceedings of EMNLP*, 12, 2014.

[82] Matthew Purver and Stuart Battersby. Experimenting with distant supervision for emotion classification. In *Proceedings of the 13th Conference of the*

European Chapter of the Association for Computational Linguistics, pages 482–491, 2012.

[83] James Pustejovsky, Jose Castano, Brent Cochran, Maciej Kotecki, Michael Morrell, and Anna Rumshisky. Extraction and disambiguation of acronym-meaning pairs in medline. *Medinfo*, 10(2001):371–375, 2001.

[84] D. Ramage, D. Hall, R. Nallapati, and C.D. Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings EMNLP*, pages 248–256, 2009.

[85] L. Ratinov, D. Roth, D. Downey, and M. Anderson. Local and global algorithms for disambiguation to wikipedia. In *Proceedings of ACL*, pages 1375–1384, 2011.

[86] Alan Ritter, Luke Zettlemoyer, Oren Etzioni, et al. Modeling missing data in distant supervision for information extraction. *Transactions of the Association for Computational Linguistics*, 1:367–378, 2013.

[87] A. Schwartz and M. Hearst. A simple algorithm for identifying abbreviation definitions in biomedical text. In *Pacific Symposium on Biocomputing*, pages 451–462, 2002.

[88] P. Sen. Collective context-aware topic models for entity disambiguation. In *Proceedings of WWW*, pages 729–738, 2012.

[89] W. Shen, J. Wang, P. Luo, and M. Wang. Linden: linking named entities with knowledge base via semantic knowledge. In *Proceedings of WWW*, pages 449–458, 2012.

[90] Wei Shen, Jiawei Han, and Jianyong Wang. A probabilistic model for linking named entities in web text with heterogeneous information networks. In *Proceedings of SIGMOD*, 2014.

[91] Wei Shen, Jianyong Wang, and Jiawei Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge & Data Engineering*, 27(2):443–460, 2014.

[92] Avirup Sil, Ernest Cronin, Penghai Nie, Yinfei Yang, Ana-Maria Popescu, and Alexander Yates. Linking named entities to any database. In *Proceedings of EMNLP/CoNLL*, pages 116–127, 2012.

[93] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of NIPS*, pages 926–934, 2013.

[94] Richard Socher, Christopher D Manning, and Andrew Y Ng. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9, 2010.

[95] Mauro Sozio and Aristides Gionis. The community-search problem and how to plan a successful cocktail party. In *Proceedings of SIGKDD*, pages 939–948, 2010.

[96] Mark Stevenson, Yikun Guo, Abdulaziz Al Amri, and Robert Gaizauskas. Disambiguation of biomedical abbreviations. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, pages 71–79, 2009.

[97] F.M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *Proceedings of WWW*, pages 697–706, 2007.

[98] Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D Manning. Multi-instance multi-label learning for relation extraction. In *Proceedings of EMNLP*, pages 455–465, 2012.

[99] Jared Suttles and Nancy Ide. Distant supervision for emotion classification with discrete binary values. In *Computational Linguistics and Intelligent Text Processing*, pages 121–136. 2013.

[100] Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of ACL*, pages 721–729, 2012.

[101] Bilyana Taneva, Tao Cheng, Kaushik Chakrabarti, and Yeye He. Mining acronym expansions and their meanings using query click log. In *Proceedings of WWW*, pages 1261–1272, 2013.

[102] Christina Unger, Lorenz Bühmann, Jens Lehmann, Axel-Cyrille Ngonga Ngomo, Daniel Gerber, and Philipp Cimiano. Template-based question answering over rdf data. In *Proceedings of WWW*, pages 639–648, 2012.

[103] Vasudeva Varma, Vijay Bharat, Sudheer Kovelamudi, Praveen Bysani, GSK Santosh, Kiran Kumar, Kranthi Reddy, Karuna Kumar, and Nitin Maganti. Iiit hyderabad at tac 2009. In *Proceedings of TAC*, 2009.

[104] Ellen M Voorhees and Dawn M Tice. Building a question answering test collection. In *Proceedings of SIGIR*, pages 200–207, 2000.

[105] Jonathan D Wren, Harold R Garner, et al. Heuristics for identification of acronym-definition patterns within text: towards an automated construction of comprehensive acronym-definition dictionaries. *Methods of information in medicine*, 41(5):426–434, 2002.

[106] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 481–492, 2012.

[107] Wei Xu, Raphael Hoffmann, Le Zhao, and Ralph Grishman. Filling knowledge base gaps for distant supervision of relation extraction. In *Proceedings of ACL*, pages 665–670, 2013.

[108] Xuchen Yao and Benjamin Van Durme. Information extraction over structured data: Question answering with freebase. In *Proceedings of ACL*, 2014.

[109] Hong Yu, Won Kim, Vasileios Hatzivassiloglou, and John Wilbur. A large scale, corpus-based approach for automatically disambiguating biomedical abbreviations. *ACM Transactions on Information Systems (TOIS)*, 24(3):380–404, 2006.

[110] Ce Zhang, Vidhya Govindaraju, Jackson Borchardt, Tim Foltz, Christopher Ré, and Shanan Peters. Geodeepdive: Statistical inference using familiar data-processing languages. In *Proceedings of SIGMOD*, pages 993–996, 2013.

[111] W. Zhang, Y.C. Sim, J. Su, and C.L. Tan. Entity linking with effective acronym expansion, instance selection and topic modeling. In *Proceedings of IJCAI*, pages 1909–1914, 2011.

[112] Zhicheng Zheng, Xiance Si, Fangtao Li, Edward Y Chang, and Xiaoyan Zhu. Entity disambiguation with freebase. In *Proceedings of WI-IAT*, pages 82–89, 2012.

[113] Lei Zou, Ruizhe Huang, Haixun Wang, Jeffer Xu Yu, Wenqiang He, and Dongyan Zhao. Natural language question answering over rdf: a graph data driven approach. In *Proceedings of SIGMOD*, pages 313–324, 2014.