Named entity recognition and linking with knowledge base

Phan, Cong Minh

2019

Phan, C. M. (2019). Named entity recognition and linking with knowledge base. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/136585>

<https://doi.org/10.32657/10356/136585>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 17:11:19 SGT

NAMED ENTITY RECOGNITION AND LINKING WITH KNOWLEDGE BASE

PHAN CONG MINH

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

2019

NAMED ENTITY RECOGNITION AND LINKING WITH KNOWLEDGE BASE

PHAN CONG MINH

School of Computer Science and Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2019

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

22/07/2019

Date

Phan Cong Minh

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

22/07/2019

Date

A/Prof. Sun Aixin

Authorship Attribution Statement

This thesis contains material from 5 papers published in the following peer-reviewed journal and conference in which I am listed as an author. The contributions of the co-authors are listed as follows:

Chapter 3 is accepted as Minh C. Phan and Aixin Sun. Collective Named Entity Recognition in User Comments via Parameterized Label Propagation. *The Journal of the Association for Information Science and Technology (JASIST)*, doi:10.1002/asi.24282, 2019.

- Prof. Sun Aixin provides the initial project direction. We also discuss about several model designs at the early stage.
- I propose and implement the model. I prepare the manuscript and it is then revised by Prof. Sun Aixin.

Chapter 4 is published as Minh C. Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. NeuPL: Attention-based Semantic Matching and Pair-Linking for Entity Disambiguation. *The 26th ACM International Conference on Information and Knowledge Management (CIKM)*, 1667-1676, 2017.

- Prof. Sun Aixin provides the initial project direction. We also discuss about several model designs at the early stage.
- I propose and implement the model. Tay Yi gives feedback about the model design and assists in the implementation.
- I prepare the manuscript. It is then edited by Prof. Sun Aixin, and revised by Dr. Han Jialong, Dr. Li Chengliang, and Tay Yi.

Chapter 5 (key idea and experiment sections) is published as Minh C. Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. Pair-Linking for Collective Entity Disambiguation: Two Could Be Better Than All. *The IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 30(1): 59-72, 2019.

- Prof. Sun Aixin suggests the project direction.
- I perform data analysis and formulate the problem. I co-design with Prof. Sun Aixin the Pair-Linking algorithm. I implement the model.
- I prepare the manuscript. Dr. Han Jialong, Dr. Li Chengliang, and Tay Yi revise the manuscript.

Chapter 5 (demo system section) is published as Minh C. Phan, Aixin Sun. CoNEREL: Collective Information Extraction in News Articles. *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR, demo paper)*, 1273-1276, 2018.

- I co-design with Prof. Sun Aixin the system architecture and user interface.
- I implement the system. I prepare the manuscript, which is subsequently revised by Prof. Sun Aixin.

Chapter 6 is accepted as Minh C. Phan, Aixin Sun and Yi Tay. Robust Representation Learning of Biomedical Names. *The 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, 3275-3285, 2019.

- I formulate the problem. I propose and implement the model. Prof. Sun Aixin discusses with me about the model design.
- I prepare the manuscript, which is then revised by Prof. Sun Aixin and Tay Yi.

22/07/2019

Date

....

Phan Cong Minh

Acknowledgements

I would like to give my first and foremost special gratitude to Prof. Sun Aixin for his guidance and support throughout my PhD. Working under his supervision was a fruitful and enjoyable experience, which allowed me to not just gain substantial knowledge about my research topic, but also broaden my perspective to related fields.

I also want to thank my seniors, Dr. Han Jialong, Dr. Li Chengliang, and my TAC members, Prof. Zhang Jie and Prof. Qui Lin for their advice and insightful feedback on my earlier work.

I am thankful to my collaborators and labmates, Tay Yi, Pham Nguyen Tuan Anh, and Grace E. Lee for their sharing of knowledge and experience. I also like to thank all my fellows for their help, support, knowledge sharing, and for our joyful moments. They are (in chronological order): Luu Anh Tuan, Surendra Sedhai, Lin Xi, Zheng Xin, Han Jianglei, Huang Keke, Tu Hongkui, Han Peng, Wang Yequan, Li Jing, Lin Ting, Chen Zhe, Lucas Vinh Tran, Nguyen Thanh Tung, Jarana Manotumruksa, Kaibo Gong, and Parisa Kaghazgaran. Thank you all!

Last but not least, I would like to thank my parents and my younger sister for their love and encouragement. This thesis is dedicated to them who are the motivation for my hard work.

Abstract

Named entities such as people, organizations, and locations appear in various kinds of textual contexts and under different surface forms. Successful extraction of these entities enables machines to understand and organize information in a systematic manner. This thesis addresses both named entity recognition (NER) and entity linking (EL) processes. The former aims at recognizing mentions of specific classes such as persons, organizations, and locations, while the latter maps these mentions to their associated entities in a knowledge base. Different from humans who can quickly identify these named entities using their commonsense knowledge and inference-making ability, machines do not have that intelligence. The main challenges arise when the mentions and local contexts are ambiguous. Moreover, the variance of entity names also introduces additional difficulty in resolving the mentions' identities. As such, the recognition and disambiguation of these entity mentions greatly depend on machine understanding of the input contexts, knowledge base entities, as well as the relations between them.

In this thesis, we introduce several novel approaches to tackle these challenges in both NER and EL. First, we propose a collective NER framework for the recognition task. Apart from local contexts, our approach utilizes relevant contexts in related documents to perform NER in a collective manner. The proposed model demonstrates superior performance on user comments in which the context of each individual comment is often limited. Second, we tackle the EL problem by first addressing the ambiguity of mentions. We study a local context-based approach that disambiguates each mention individually based on its local context. We propose an attention-based neural network architecture to estimate the semantic similarity between a mention's local context and its entity candidates. Our model utilizes Wikipedia hyperlinks as the training data and obtains competitive performance on different benchmark datasets. Third, we investigate a collective EL approach,

which utilizes the semantic relatedness between entities to collectively resolve the mentions' ambiguity. We first analyze the semantic coherence between entities in a document. In contrast to the assumptions made in previous works, our analysis reveals that not all entities (in a document) are highly related to each other. This insight leads us to relax the coherence constraint and develop a significantly faster and more effective collecting linking algorithm. Finally, we study a special setting of EL in which the disambiguation is based on the matching between the mentions and entity names. This setting is commonly seen in particular applications such as biomedical concept, product name, and job title normalizations. In this setting, we focus on learning semantic representations for entity names such that representations of synonymous names are close to each other. We then evaluate the learned representations in the biomedical concept linking task. All in all, despite the problems of NER and EL have been established and investigated for the last decade, this thesis contributes several key ideas that could further improve the performance and shed light on few potential directions for future work.

Contents

Acknowledgements	v
Abstract	vi
List of Figures	xi
List of Tables	xvi
Acronyms	xx
1 Introduction	1
1.1 Motivation	1
1.2 Approaches	6
1.3 Research Contributions	8
1.4 Thesis Outline	9
2 Literature Review	10
2.1 Named Entity Recognition	11
2.1.1 Local Context-based Named Entity Recognition	13
2.1.2 Collective Named Entity Recognition	15
2.2 Entity Linking	17
2.2.1 Knowledge Base	19
2.2.2 Candidate Selection	21
2.2.3 Local Context-based Entity Linking	22
2.2.4 Collective Entity Linking	25
2.2.5 Entity Name Normalization	29

3	Collective Named Entity Recognition	33
3.1	Introduction	33
3.2	Collective NER Framework	36
3.2.1	Mention Co-reference Graph	36
3.2.2	Parameterized Label Propagation	39
3.3	Experiments	41
3.3.1	Experimental Settings	41
3.3.2	Datasets and Baselines	43
3.3.3	Overall Performance	45
3.3.4	Analysis of Collective NER	46
3.3.5	Analysis of Parameterized Label Propagation	49
3.4	Summary	52
4	Local Context-based Entity Linking	54
4.1	Introduction	54
4.2	Joint Learning of Word and Entity Embeddings	55
4.3	Attention-based Semantic Matching Architecture	57
4.4	Experiments	61
4.4.1	Experimental Settings	61
4.4.2	Datasets and Baselines	65
4.4.3	Overall Performance	69
4.4.4	Ablation Study and Analysis	70
4.5	Summary	72
5	Collective Entity Linking	73
5.1	Introduction	73
5.2	Semantic Coherence of Entities	75
5.2.1	Semantic Coherence Analysis	75
5.2.2	Tree-based Objective	79
5.3	Pair-Linking Algorithm	82
5.3.1	Idea and Algorithm	82
5.3.2	Computational Complexity	85
5.4	Experiments	86
5.4.1	Experimental Settings	86

5.4.2	Datasets and Baselines	87
5.4.3	Overall Performance	89
5.4.4	Robustness to Not-in-list Entities	95
5.5	Demo System and Pair-Linking Visualization	96
5.6	Summary	98
6	Entity Name Normalization	100
6.1	Introduction	100
6.2	Representation Learning of Entity Names	103
6.2.1	Context-based Skip-gram Model	103
6.2.2	Representation Learning with Context, Concept, and Synonym- based Objectives	105
6.3	Experiments	109
6.3.1	Experimental Settings	109
6.3.2	Datasets and Baselines	111
6.3.3	Overall Performance	112
6.3.4	Qualitative Analysis	114
6.4	Summary	118
7	Conclusion and Future Work	120
7.1	Conclusion	120
7.2	Future Work	123
7.2.1	Incorporate Language and Structured Knowledge Modelings	123
7.2.2	Long-tail Mentions and Entities	124
	List of Publications	125
	References	126

List of Figures

1.1	An example of named entity recognition and linking results for a sentence. The recognition step identifies two people mentions (‘Pacquiao’ and ‘Bradley’), and one location mention (‘Las Vegas’). The linking step then maps each mention to its associated entity in a knowledge base (Wikipedia in this case).	1
1.2	Results of named entity recognition and linking are commonly used for knowledge population. The results also enable many downstream applications to benefit from the structured information in knowledge base. . . .	2
1.3	Example of Google search result for the query ‘woods’ (captured on June 19, 2019). The right panel lists some potential entities that share the same or similar name with the input query. Users may choose to click on one of these entities to clarify their information need.	3
1.4	Two main challenges of named entity recognition and linking: the ambiguity of mentions and contexts, and the variance of entity names.	4
2.1	General pipeline architecture for named entity recognition and linking. The dashed line separates between alternative approaches used in the recognition and linking processes.	10
2.2	Two categories of NER approaches: local context-based and collective NER. Local context-based NER relies on local contexts and performs the recognition independently for each input text. On the other hand, collective NER utilizes relevant contexts in related sentences or documents to perform NER in a collective manner. A context in this illustration refers to a sentence, a short paragraph, a user comment, or a tweet.	12

2.3	Illustration of NER as a sequence labeling task. Each input token is assigned a BIO tag and a mention class label. The B tag indicates that the token is the beginning of a mention. The I tag indicates that the token is inside a mention. The O tag indicates that the token does not belong to any mention. These token labels are convertible to the NER expected output.	13
2.4	A simple illustration of an NER model based on recurrent neural network (RNN) and conditional random fields (CRF). The RNN is used to automatically extract hidden representations given the token embeddings as input. The hidden representations are then converted into structured label predictions using a CRF layer.	14
2.5	Entity linking results of four entity mentions. The ground-truth entity of each mention is highlighted in bold-face in its candidate list.	17
2.6	Example of description and anchor texts in Wikipedia KB for the entity Tiger Woods. The description text provides concise information about the entity. Furthermore, mentions of Tiger Woods in other Wikipedia pages and their local context are often utilized to train a semantic matching model for EL. The hyperlinks can also be used to estimate the semantic similarity between two entities, based on their common citing pages.	19
2.7	An example of a mention-entity graph consisting of three mentions and their entity candidates. The weights between the mentions and entity candidates represent the local relevance scores, while the weights between the entity candidates represent the pairwise semantic relatedness scores.	28
2.8	Alignment in word mover's distance measure (WMD) for two biomedical names belonging to the same entity. The arrows illustrate the flows between word pairs that have high semantic similarity scores.	30
3.1	Examples of named entity mention in two user comments in two new articles. The extracted mentions are underlined.	34
3.2	Overall architecture of our proposed collective NER framework. A mention co-reference graph is constructed from the sets of mentions that are initially extracted from the main articles and their user comments. Parameterized label propagation is then applied on the constructed graph to refine the initial mention labels.	37

3.3	Illustration of label propagation for the mention ‘ <i>curry</i> ’ in a comment. The propagation weights between mentions are learned automatically based on the features extracted from the mentions’ initial labels and their local contexts.	39
3.4	F_1 Performance of CoNER: CRF + Y (Y is an inference method such as KNN, ABSORD, MAD, GCN, GAT, PLP) in different expansions the co-reference graph G (controlled by k -nearest neighbors). Note that when $k = 0$, CoNER: CRF + Y is associated with the base model CRF	48
3.5	Performance of CoNER: CRF + PLP in different settings of number of iteration step ρ in PLP.	50
3.6	Distributions of propagation weights on two types of edges: those from article mentions to candidates mentions in user comments, and those connecting between candidate mentions (in user comments).	51
3.7	Case studies of propagation weights between candidate mentions in user comments. The mentions are shown with their local contexts. The labels in square brackets indicate the initial predictions by CRF model.	52
4.1	Neural network architecture for learning the semantic relevance score between a mention’s local context and an entity candidate. Two unidirectional LSTMs are used to encode the left and right-side local contexts. On the other hand, entity embedding and another LSTM unit are used to construct the representation for an entity candidate. Attention mechanism and feed-forward neural network (FFNN) are used to capture the matching between these two representations. Finally, the sigmoid matching score $\sigma(m_i, e_i)$ is combined with prior probability score $P(e m)$ to obtained the final semantic relevance score.	57
4.2	F_1 performance of NeuL with different settings of α . A larger α value indicates that the disambiguation will favor prior probability knowledge more than semantic matching scores.	71
5.1	Sparse forms of semantic coherence among entities in two example sentences. Only the edges that connect between two strongly related entities are shown.	74

5.2	Four different forms of connections among entities in a document. In the dense form, all these entities are pairwise related to each other. In the tree- and chain-like forms, there are minimal coherent connections among these entities. On the other hand, in the forest-like form, the entity connections are relatively sparse.	76
5.3	An example entity candidate graph for a document consisting of 4 mentions, each mention has 2 entity candidates. The edge weights represent the distance between the pairs of entities. The weight of the minimum spanning tree derived from the selected entities (represented by the filled points) is used as the MINTREE coherence measure.	80
5.4	An example of an entity candidate graph with 5 mentions, each mention has 2 entity candidates. The edges between the entity candidates are weighted by the semantic distance. Only the edges with the lowest semantic distances are illustrated. The solid edges are the ones selected by the Pair-Linking process.	83
5.5	Main GUI of our demo system. The left panel displays the statistics about the extracted entities. The right panel highlights the mentions where they are referred to.	96
5.6	A graphical visualization of Pair-Linking process, after the 7 th linking step, and the completion. The left panel details the local relevance and pairwise relatedness scores corresponding to each step. The right panel visualizes the pairs of linking assignments that have been made at each step. The edge width represents the pairwise confidence score (see Equation 5.5). The current step of Pair-Linking is highlighted by the orange edge.	97
5.7	Visualization of Pair-Linking results for a news article and its user comments. The entities that appear in comments are with gray borders, while the ones in the main article text have red borders.	98
6.1	Illustration of three aspects, which are related to three training objectives, for computing representation of entity name (surface form) s . Intuitively, the representation is supposed to be similar to its synonym's as well as its conceptual and contextual representations.	102

- 6.2 Our proposed entity name encoding framework. The main encoder (ENE) uses two-level BiLSTM architecture to capture both character and word-level information of an input name. ENE parameters are learned by considering three training objectives. Synonym-based objective $\mathcal{L}_{syn}$ enforces similar representations of two synonymous names (s and s'). Concept-based objective $\mathcal{L}_{def}$, and context-based objectives $\mathcal{L}_{ctx}$ apply similarity constraints on the representations of names (s and s' , which are interchangeable) and their conceptual and contextual representations ($g(e)$ and $g(x)$, respectively). Details about $g(e)$ and $g(x)$ calculations are discussed in Section 6.2.2. 105
- 6.3 Mean coverage at k : average ratio of correct synonyms that are found in k -nearest neighbors, which are estimated by cosine similarity of name embeddings. Note that names in these disease and chemical test sets are not seen in the training data. 114
- 6.4 t-SNE visualization of 254 name embeddings. These names belong to 10 disease concepts in which 5 of these concepts appear in the training data, while the other 5 concepts (marked with (*)) do not. It can be observed that ENE projects names of the same concept close to each other. The model also retains closeness between names of related concepts, such as ‘parkinson disease’ and ‘paranoid disorders’ (see the red square and green cross signs). 115

List of Tables

2.1	A set of hand-crafted features that are commonly used for named entity recognition.	14
2.2	Key information stored in UMLS (a biomedical metathesaurus) for ‘Leiner disease’ entity.	20
2.3	Summary of existing local context-based entity linking models. The categorization of different models is based on the methods used to represent a mention (with its local context), an entity candidate, matching between a mention and an entity candidate, and the learning models.	23
3.1	Features for learning propagation weight between two mentions m_i and m_j	39
3.2	Statistics of three partitions in our annotated Yahoo! user comment dataset. 1500 articles sampled with their associated user comments. The article mentions (M_A) and candidate mentions (M_C) are extracted by pre-trained NER annotators. We randomly select 1 comment from each sampled article to annotate.	44
3.3	Performance of baselines and the best configuration of CoNER on Yahoo! comment test set. † indicates performance difference against the one in boldface is statistically significant by one-tailed paired t-test ($p < 0.05$).	46
3.4	Performance of different configurations of CoNER: X + Y on Yahoo! comment test set. X denotes the base model used to obtain the initial labels, and Y denotes the employed inference method. † indicates that the performance difference against the one in boldface (within a row group) is statistically significant by one-tailed paired t-test ($p < 0.05$).	47
3.5	F_1 performance of collective NER on CoNLL03 test set. Different percentage of CoNLL03 training data is used to train the base model. The improvement is shown in terms of absolute increment score and relative error reduction.	49

3.6	F_1 performance of collective NER when additional percentages of the development data is used to train the base model.	50
4.1	Hyperparameter setting used in our proposed semantic matching model. .	63
4.2	Statistics of the 7 test datasets used in experiments. $ D $, $ M $, Avg_m , and $Length$ are the number of documents, number of mentions, average number of mentions per document, and document length in number of words, respectively.	64
4.3	F_1 performance of NeuL and all baselines. The best results are in boldface and the second-best ones are underlined.	68
4.4	Micro-averaged precision, recall and F_1 performance of NeuL.	70
4.5	F_1 performance of our proposed model and two variants: one with single-directional LSTM used to encode the local context, and one without the attention mechanism.	70
5.1	Average denseness of entity coherence calculated on each EL dataset. Only the documents having more than 3 mentions are considered. The results are reported with three pairwise relatedness measures: Wikipedia link-based measure (WLM), normalized Jaccard similarity (NJS), and entity embedding similarity (EES).	78
5.2	Spearman's correlations (ρ) between the disambiguation quality (represented by the number of correct linking decisions) and three collective linking objective scores: ALL-Link (AL), SINGLE-Link (SL) and MINTREE (MT). The correlations are averaged across 8 datasets. The results are reported with three relatedness measures: Wikipedia Link-based Measure (WLM), Normalized Jaccard Similarity (NJS) and Entity Embedding Similarity (EES). For each relatedness measure, we also analyze the correlation between the every pairs of objectives.	81
5.3	Statistics of the 8 test datasets used in our evaluation. $ D $, $ M $, Avg_m , and $Length$ are the number of documents, number of mentions, the average number of mentions per document, and the average number of words per document, respectively.	87

5.4	Micro-averaged F_1 of different collective EL algorithms with different pairwise relatedness measures. The best scores are in boldface and the second-best ones are underlined. The numbers of win and runner-up each method performs across different datasets are also illustrated. Significance test is performed on Reuters123, RSS500 and Micro14 datasets (denoted by *) which contain a sufficient number of documents. † indicates the difference against the Pair-Linking's F_1 score is statistically significant by one-tailed paired t -test (with $p < 0.05$).	90
5.5	Micro-averaged F_1 of different collective linking algorithms with different pairwise relatedness measures. The best scores are in boldface and the second-best ones are underlined. Significance test is performed on Reuters123, RSS500 and Micro14 datasets (denoted by *) which contain a sufficient number of documents. † indicates the difference against the Pair-Linking's F_1 score is statistically significant by one-tailed paired t -test (with $p < 0.05$).	91
5.6	Time complexity of different linking algorithms. N is the number of mentions, k is the average number of candidates per mention, and $\mathcal{I}$ is the number of iterations for convergence.	91
5.7	Average time to disambiguate mentions in one document (in milliseconds) for each dataset. The time for preprocessing steps such as candidate generation is not included.	92
5.8	Micro-averaged precision, recall, and F_1 of Pair-Linking with NJS&EES as the pairwise relatedness measure.	94
5.9	Micro-averaged F_1 of Pair-Linking (using NJS&EES pairwise relatedness measure) and other disambiguation systems. The 'local' annotations indicate that the associated approaches are solely based on the local relevance scores and do not implement any collective EL methods. (PL: Pair-Linking, Avg: Average)	94
5.10	Micro-averaged F_1 of Pair-Linking (with NJS&EES as the pairwise relatedness measure) with different percentage of non-linkable mentions (as noise). The F_1 score is calculated on the linkable mentions.	95
6.1	Example of entities and their names (multi-word expressions). These name include both official names in a knowledge base or unofficial names mentioned in texts.	100

- 6.2 Biomedical context linking accuracy on disease and chemical datasets. The last row group includes the results of supervised models that utilize training annotations in each specific dataset. The ‘exact match’ rule indicates the use of annotation in the training partition to overwrite the original disambiguation result if a query mention is found in the training data. † indicates the results reported in [1]. 113
- 6.3 Mean average precision (MAP) performance on the synonym retrieval task. The best and second best results are in boldface and underlined, respectively. 117
- 6.4 Spearman’s rank correlation coefficient between cosine similarity scores of name embeddings and human judgments, reported on semantic similarity and relatedness benchmarks. 118

Acronyms

CNN	Convolutional neural network
CRF	Conditional random fields
EL	Entity linking
FFNN	Feed-forward neural networks
GRU	Gated recurrent unit
HNN	Hidden Markov model
IR	Information retrieval
KB	Knowledge base
LP	Label propagation
LSTM	Long short-term memory
LTR	Learning-to-rank
NER	Named entity recognition
NLP	Natural language processing
RNN	Recurrent neural network

Chapter 1

Introduction

1.1 Motivation

Named entities such as people, organizations, and locations are commonly found in various forms of natural languages including written text and speech. The extraction of these named entities enables machines to understand and organize information in a systematic manner. This extraction generally consists of two consecutive processes: named entity recognition (NER) and entity linking¹ (EL). The former aims at identifying the mention locations and classifying the semantic types of the mentions, while the latter maps the

FIGURE 1.1: An example of named entity recognition and linking results for a sentence. The recognition step identifies two people mentions ('Pacquiao' and 'Bradley'), and one location mention ('Las Vegas'). The linking step then maps each mention to its associated entity in a knowledge base (Wikipedia in this case).

¹Entity linking is also known as named entity linking, or named entity disambiguation. However, the term 'entity linking' is more commonly used.

FIGURE 1.2: Results of named entity recognition and linking are commonly used for knowledge population. The results also enable many downstream applications to benefit from the structured information in knowledge base.

extracted mentions to their associated entities in a knowledge base. Consider an example illustrated in Figure 1.1, the recognition step outputs two people mentions (‘Pacquiao’ and ‘Bradley’) and one location mention (‘Las Vegas’). Since a knowledge base can contain multiple entities that have the same or similar names, the linking step will resolve the ambiguity. It then assigns to each mention one corresponding entity in the knowledge base. In this example, ‘Pacquiao’ and ‘Bradley’ are linked to two boxers, Manny Pacquiao and Timothy Bradley, respectively, and ‘Las Vegas’ is mapped to the well-known city in the United States. Note that, if a mention is not associated with any entity in the knowledge base, a pseudo not-in-list (NIL) entity is assigned to the mention.

Why is it essential? Named entity recognition and linking is known as the first and crucial step in the attempt to extract structured knowledge from unstructured texts. Since new information is created at a faster pace than ever before, updating an existing knowledge bases (KB) has become increasingly significant and demanding. Regarding the example in Figure 1.1, given the result that two entities Manny Pacquiao and Timothy Bradley are correctly identified, the related facts about them can be extracted and added into the KB. This process is known as knowledge base population [2], which has been a fruitful research area for the last decade.²

The results of named entity recognition and linking also enable the use of structured information in the KB to support multiple applications in information retrieval (IR), content analysis, and question-answering (see Figure 1.2). Semantic search is one of the tasks that benefit from the NER and EL results. As 40-70% of natural language queries in web

²Knowledge base population is one of the main tracks in the annual Text Analysis Conference (TAC).

FIGURE 1.3: Example of Google search result for the query ‘woods’ (captured on June 19, 2019). The right panel lists some potential entities that share the same or similar name with the input query. Users may choose to click on one of these entities to clarify their information need.

searches or question answering systems contain named entities [3, 4], correctly extracting these enclosed entities contributes greatly to the successful query understanding [5–8]. Moreover, the extraction results also help to enhance the users’ experience when interacting with the search process. Nowadays, popular search engines such as Google, Bing, and Yahoo allow users to clarify their queries by suggesting the entities that the queries may refer or relate to. For example, as shown in Figure 1.3, Google search suggests several entity candidates for the ambiguous query ‘woods’. This suggestion not only assists users in clarifying their information need, but also helps the systems to return accurate documents, thus improving the users’ overall satisfaction.

Text mining tasks such as entity-based sentiment analysis [9, 10] and relation extraction [11, 12] even use the results of named entity recognition and linking as inputs. For example, biomedical relation extraction is a significant task which aims to automatically extract valuable biomedical interactions between protein-protein, drug-drug, or chemical-disease. Most of these approaches [13–15] presume that the biomedical concepts are extracted beforehand and provided as inputs. These models then focus on classifying whether there is a biomedical interaction between a pair of entities. As such, the effectiveness of these models greatly depends on the quality of the NER and EL processes.

FIGURE 1.4: Two main challenges of named entity recognition and linking: the ambiguity of mentions and contexts, and the variance of entity names.

Moreover, the construction of answers in question answering (QA) systems [7, 16–18] usually needs to identify the mentioned entities and retrieve their related information from a KB. All in all, the significant role of named entity recognition and linking in information retrieval (IR) and natural language processing (NLP) is unquestionable.

What are the challenges? Named entity recognition and linking contributes to an ultimate goal that is to help machines to understand what people say or write. Different from humans who can use their prior knowledge to quickly interpret and understand various natural language contexts, machines do not have such commonsense knowledge. Furthermore, the inference-making ability of machines is far from comparable to humans' brains. Consider a piece of text: 'make us great again'. In most cases, the mention 'us' in this text is a pronoun, and hence it should not be recognized as a named entity mention. However, for the rest of the cases, it can be a mention of the United States or a mention that refers to another named entity, such as an album, a movie, or an organization which shares the same name³. Thus, named entity recognition and linking is not simply an indexing and retrieval task; it further requires semantic understanding of the input texts (including the mentions and their contexts) as well as the structured information resided in the knowledge base.

Illustrated in Figure 1.4, challenges of named entity recognition and linking arise mainly because of (1) *the ambiguity of mentions and local contexts*, and (2) *the variance of entity names*. First, since people tend to use the least effort when communicating, they often

³According to Wikipedia (captured on June 19, 2019), there are more than 30 different entities that can be referred to by the string 'us'. Reference: [https://en.wikipedia.org/wiki/US_\(disambiguation\)](https://en.wikipedia.org/wiki/US_(disambiguation)).

assume that the receivers know the background information. Therefore, they often cite named entities using relatively short names and limited local contexts. However, for machine understanding, these mentions and local contexts can be highly ambiguous. For example, the same entity mention (surface form⁴) can refer to multiple entities in a knowledge base. Moreover, the local contexts do not always contain useful evidence that reflects the identity of the mentioned entities. Popular entities such as politicians can be mentioned in various contexts including political, sport, or entertainment news, which are not necessarily matched with their descriptions in the KB. The ambiguity of the local contexts is even more serious in social media texts such as user comments or tweets because of the shortness and noisy nature of these kinds of texts. Thus, the performance in these domains declines significantly compared to formal texts such as news articles.

Second, the fact that an entity can be referred to by different surface forms (or names) introduces another challenge for EL. Natural language offers various ways to express the same entity using different combinations of words (or tokens). However, many of these multi-word expressions may not be found in an existing knowledge base. As a result, linking these mentions to their associated entities is more challenging, especially in specific domains such as biomedical texts, product names, or job titles. For example, one biomedical concept, such as ‘Exudative retinopathy’, is often associated with many alternative names (*e.g.*, ‘Coats’ disease’ and ‘Abnormal retinal vascular development’). Note that, these multi-word expressions are generally less ambiguous than names of people or locations because of their creation intents. However, the lexical mismatches between these mentions and existing names in the KB remains a key challenge for EL in this domain.

Moreover, as a task that involves the knowledge base, NER and EL should take into consideration the structured information offered by the KB. Most EL systems use Wikipedia, which contains natural language descriptions for popular entities. The anchor texts and hyperlinks in Wikipedia pages serve as valuable data for the model training. However, such a well-constructed knowledge base is not yet available in many specific domains such as biomedical texts, products, or points of interest. In these domains, the existing knowledge bases are still in their early stages, and they mostly have the forms of synonym

⁴We refer to the string used to represent an entity or mention as a surface form.

dictionaries (*i.e.*, each entity is associated with a list of alternative names). Therefore, entity linking in these domains demands additional techniques to make the best use of the limited KB information. All in all, as natural language and knowledge bases are being created and evolved, they will introduce new challenges for named entity recognition and linking. However, with its significant benefits for a wide range of IR and NLP applications, this problem will continue to receive considerable attention in both industrial and academic research communities.

1.2 Approaches

In this thesis, we study the problem of named entity recognition and linking. Similar to most existing work, we separate the problem into two sub-tasks for investigation, namely named entity recognition (NER), and entity linking (EL). The key theme of our approach is to utilize the contexts in the input and the knowledge base to improve the performances of both recognition and linking processes.

Named entity recognition. We propose a collective NER approach to address the shortness and noisiness of social media texts such as user comments. We first construct a mention co-reference graph by collecting the co-reference evidence from all the relevant contexts found in the related comments and articles. Each mention is initialized a soft-label based on its local context. Our collective NER model then performs inference on a mention co-reference graph such that the labels of mentions are propagated from more confident cases to less confident cases. We propose parameterized label propagation (PLP) to be used as a semi-supervised inference algorithm. In PLP, the propagation weights between mentions are automatically learned given their local contexts and initial labels as input. We create a dataset that contains annotated comments collected from Yahoo! News articles for training and testing. PLP's parameters are learned by gradient descent on the training set. We compare the performance of our collective NER model with approaches that process each local context independently. We then evaluate the effectiveness of PLP and analyze its behavior in the NER task.

Entity linking. We first focus on the semantic matching between a mention’s local context and its entity candidates to disambiguate the mention. We propose a deep neural network that uses two unidirectional LSTMs to encode the left- and right-side local contexts of the mention. On the other hand, entity embeddings and another unidirectional LSTM are used to construct the representation for the entity candidate. We employ attention mechanism to emphasize the relevant matches in the mention’s local context with regards to the entity candidate’s representation. We then use a two-layer feed-forward neural network to capture the matching between the local context’s representation and the entity candidate’s representation. For training, we utilize the anchor texts and hyperlinks available in Wikipedia. The trained model can be used to disambiguate mentions in documents from general domains such as web pages, news articles, and tweets.

Second, we investigate a collective approach for EL. In addition to the local context-based matching, this collective approach relies on the semantic relatedness between entities to perform disambiguation in a collective manner. We first study the degree of semantic coherence among the entities that appear in a document. Our analysis shows that not all entities (in a document) are highly related to each other. We then design a new objective that relaxes the coherence constraint, and propose Pair-Linking as a fast and effective collective EL algorithm. Pair-Linking performs disambiguation by iteratively selecting a mention pair that has the highest confidence for decision making at each step. We evaluate Pair-Linking on 8 popular benchmark datasets and compare it with state-of-the-art baselines. To further investigate the entity coherence and understand the effectiveness of Pair-Linking, we develop a demonstration system that simulates the disambiguation process and visualizes the results of Pair-Linking.

Finally, we address the EL challenge that resides in the high variance of entity names. This challenge is commonly seen in particular domains such as biomedical texts, product names, or job titles. We focus on learning semantic representations for multi-word expressions such that the representations associated with the same entity will be similar to each other. To this end, we propose three objectives to be used in the representation learning, namely context, concept, and synonym-based objectives. These objectives not only enforce the similarity between synonymous representations but also aim to encode

conceptual and contextual information into the learned representations. As such, our proposed name encoder can derive meaningful representations for un-seen names. We then evaluate our model in the biomedical concept linking task.

1.3 Research Contributions

We summarize our key contributions as follows:

- First, we study a collective NER idea that utilizes the external relevant contexts to improve NER performance. We show that this approach is effective in tackling the shortness and noisiness of social media texts such as user comments. We further propose parameterized label propagation (PLP). Different from existing propagation methods, PLP can learn the propagation weights automatically given a set of annotated training data. The experiment result demonstrates the advantage of the collective NER approach as well as the superior performance of PLP over other inference methods. This result is published in the Journal of the Association for Information Science and Technology (JASIST) [19].
- Second, we study a local context-based approach for EL. We carefully design an attention-based neural network model that takes into consideration the mention's location (within its local context) and the semantic representation of an entity candidate. To the best of our knowledge, we are the first to employ neural network with attention mechanism for entity linking. The experiment shows that the proposed model achieves competitive and even state-of-the-art performance on multiple benchmark datasets. This result is reported in the Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM) [20].
- Third, we investigate a collective linking approach for EL. For the first time, we study the degree of semantic coherence among the entities that appear in a document. In contrast to the assumptions used in previous works, our result reveals that not all entities (in a document) are highly related to each other. This insight leads us to develop a new objective that relaxes the coherence constraint. We then propose Pair-Linking as a fast and effective collective linking algorithm. In our evaluation,

Pair-Linking is significantly faster while yielding comparable and even better disambiguation accuracy. This result is published in IEEE Transactions on Knowledge and Data Engineering (TKDE) [21]. Furthermore, our Pair-Linking demonstration system is reported in the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR) [22].

- Finally, we study a special setting of EL in biomedical text, in which the disambiguation is mostly based on the mentions and entity names (multi-word expressions). We propose a robust framework for learning entity name representations. Through experiments with the semantic similarity and relatedness, we show that the learned representations have encoded useful semantic information and benefit the EL task. This work is reported in the Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL) [23].

1.4 Thesis Outline

This thesis contains the introduction (this chapter), a literature review, four main contributions, and the conclusion. In Chapter 2, we provide the readers with a review of related work in named entity recognition and linking. Chapter 3 starts to investigate the first sub-task, which is named entity recognition. In this chapter, we propose a collective NER approach and verify its effectiveness on an user comments dataset. Chapters 4 to 6 begin to study different approaches for entity linking. Specifically, Chapter 4 studies a local context-based EL approach in which we propose a deep neural network model to estimate the local relevance score between a mention’s local context and an entity candidate. Chapter 5 investigates a collective EL approach. We study the semantic relatedness between the entities in a document. We then propose a fast and effective collecting linking algorithm called Pair-Linking. Next, Chapter 6 studies a special setting of EL in which the disambiguation is mostly based on the mentions and entity names. Finally, Chapter 7 concludes the thesis and discusses several potential directions for future work.

Chapter 2

Literature Review

In this chapter, we present an overview of existing approaches in named entity recognition and linking. As is illustrated in Figure 2.1, general pipeline architecture for named entity recognition and linking consists of two processes: recognition and linking. For the recognition process, we categorize the NER models into two groups: local context-based and collective NER models. The local context-based approach splits an input text into sentences (or smaller chunks) and processes each sentence independently. On the other hand, the collective approach utilizes the relevant contexts in the related sentences and documents to perform NER in a collective manner. We will discuss these two approaches in the first section of this chapter. We then provide a literature survey about entity linking with a consideration of the target knowledge base. Generally, the linking process consists of two main steps. The first step is candidate selection that will retrieve for each mention

FIGURE 2.1: General pipeline architecture for named entity recognition and linking. The dashed line separates between alternative approaches used in the recognition and linking processes.

a list of potential entity candidates from the knowledge base. Afterward, the disambiguation process maps each mention to an entity candidate based on local context-based EL, collective EL, or entity name normalization. Each of these approaches has its own pros and cons, which will be discussed in the second section of this chapter.

We also acknowledge several models that try to perform the recognition and linking jointly [24–27]. However, this approach has one limitation regarding the computational complexity. The extraction models, in this case, need to consider KB entities while performing the mention recognition, thus resulting in many possible combinations of mentions and entity candidates. Furthermore, NER and EL have their own use cases. Some downstream applications only need the result of NER, while in other settings, the mentions are provided as inputs. Therefore, the separation of these two tasks makes the employed techniques more generalizable and applicable to a wide range of applications.

2.1 Named Entity Recognition

Named entity recognition is a long-standing problem in NLP. The task aims at identifying locations (or mentions) of named entities that appear in texts, and classifying each mention into one of predefined mention classes. The input for NER is usually a short text such as a sentence or a short paragraph. The NER outputs are the extracted mentions’ locations and their associated mention classes. These extracted mentions will be the inputs of the entity linking process (see Figure 2.1). Specifically, the mentions’ locations specify the surface forms that need to be linked, and the associated mention classes are often used in the candidate selection to filter potential entity candidates. Thus, the NER step plays an important role in the overall performance of the entity extraction.

Most NER systems consider a small set of mention classes. For example, NER in open text domain usually uses four mention classes, which are person name (PER), organization (ORG), location (LOC), and miscellaneous (MISC, referring to other types of named entities such as language, product, and event). On the other hand, NER in biomedical texts often considers only mentions of diseases, chemicals, and genes. There is also another group of research works that pay more attention to fine-grained mention classes [28–30]. In this setting, the mention class set contain a much larger number of classes. These classes are

FIGURE 2.2: Two categories of NER approaches: local context-based and collective NER. Local context-based NER relies on local contexts and performs the recognition independently for each input text. On the other hand, collective NER utilizes relevant contexts in related sentences or documents to perform NER in a collective manner. A context in this illustration refers to a sentence, a short paragraph, a user comment, or a tweet.

often organized into a hierarchical structure. For example, FIGER dataset [28] contains 112 classes in which the person class is divided into several sub-classes including artist, engineer and politician sub-classes. As the number of mention classes increases, NER systems need to have a deeper semantic understanding of the mentions' contexts. For this reason, fine-grained NER is still challenging and its performance scores are generally lower than the results of coarse-grained NER.

Performance of NER systems also varies from domain to domain. At some points in the past, NER is considered as a solved problem because of its high performance in formal texts such as news articles. For example, the best annotator obtains 0.96 F_1 score on MUC-6 dataset [31] which contains 318 annotated Wall Street journal articles. On a newer test set, which is also in newswire text (CoNLL03 [32]), recently proposed neural network models achieve more than 0.92 F_1 score [33, 34]. However, NER performance in non-formal texts such as user comments or tweets declines significantly. For example, a decent NER model that is designed for NER in social media texts (tweets and user comments) only achieves 0.49 F_1 score [35]. As such, there are still remaining challenges for NER in these kinds of user-generated texts.

Two categories of NER approaches are local context-based and collective NER. Local context-based NER performs recognition based on each individual local context, which is usually limited to a single sentence. Each sentence is assumed to contain sufficient contextual information for most NLP tasks such as parsing, POS tagging, and NER. Therefore,

Input:	<u>Pacquiao</u>	,	37	,	easily	won	his	third	battle	with	<u>Bradley</u>	in	<u>Las Vegas</u>	.	
BIO Tag:	B-PER	O	O	O	O	O	O	O	O	O	B-PER	O	B-LOC	I-LOC	O
Output:	PER: Pacquiao										PER: Bradley		LOC: Las Vegas		

FIGURE 2.3: Illustration of NER as a sequence labeling task. Each input token is assigned a BIO tag and a mention class label. The B tag indicates that the token is the beginning of a mention. The I tag indicates that the token is inside a mention. The O tag indicates that the token does not belong to any mention. These token labels are convertible to the NER expected output.

most NER systems split the input text into sentences and process each sentence independently. This approach performs especially well on formal texts. On the other hand, collective NER approach utilizes the relevant contexts in related documents (or comments) to perform the recognition in a collective manner (see Figure 2.2). Collective NER is often used to tackle the shortness and noisiness when performing NER on user-generated texts. In the following sub-sections, we will review the works related to these two NER approaches. We will also discuss the remaining challenges of NER.

2.1.1 Local Context-based Named Entity Recognition

Local context-based approaches treat NER as a sequence labeling task. As is illustrated in Figure 2.3, a set of labeling tags is used to indicate if a token (in the input sentence) belongs to an entity mention, and also to denote the mention class it associates with. Given this labeling scheme, the NER is equivalent to the task of predicting tag for each input token.

Feature Engineering Approach. Since each of those tokens has some dependency on the previous and the next tokens, structured prediction techniques are commonly used for NER task. Early approaches are based on hidden Markov models (HMM) [37–39] and conditional random fields (CRF) [40–45]. These models require hand-crafted features to be extracted and used as the input presentation for each token. These hand-crafted features are designed to capture the surface form, syntactic, semantic information of the token and its local context. We list some common features in Table 2.1. Given the extracted features, HMM or CRF-based NER models learn their parameters from a set of annotated sentences (training data). Once trained, these supervised models demonstrate robust NER performance in different domains. However, as these models are feature-dependent, their

TABLE 2.1: A set of hand-crafted features that are commonly used for named entity recognition.

Feature	Description
Tokens	The current token and surrounding tokens (within a fixed-length window such as 2).
Prefixes and suffixes	Prefixes and suffixes up to a fixed length (<i>e.g.</i> , 6). Suffixes such as ‘-land’ are highly correlated with location entities (<i>e.g.</i> , Feuerland, Nederland).
Orthography	The shape of the token’s surface form. For example, indications of digits and uppercase letters.
Part-of-Speech tags	The POS tag of the token (<i>e.g.</i> , noun, verb, adjective, and preposition).
Lexical matches	Indication of lexical matching between the token and entity names in a given dictionary.
Token clusters	Clustering result of the token. There are several clustering methods that can be used such as Brown, Clark or LDA clustering (see [36] for more details).

FIGURE 2.4: A simple illustration of an NER model based on recurrent neural network (RNN) and conditional random fields (CRF). The RNN is used to automatically extract hidden representations given the token embeddings as input. The hidden representations are then converted into structured label predictions using a CRF layer.

performances highly rely on quality of the extracted features. Thus, in more challenging text domains including user-generated texts in social media, the NER performance degrades significantly [35]. The reason is that the syntactic and semantic features in these kinds of texts are more difficult to extract.

Neural Network Approach. Recently proposed NER models are based on neural networks [46–48]. Specifically, recurrent neural networks (RNN) such as long short-term memory [49] (LSTM) and gated recurrent units [50] (GRU) are utilized to encode sequence information of the input tokens. As is illustrated in Figure 2.4, instead of using hand-crafted features, these RNN models use word and character embeddings to construct the representation for each input token. The output of RNN encoders is a sequence of hidden representations associated with all tokens in the input sentence. These hidden representations are passed to a CRF layer to obtain the structural label predictions. Different from the feature-engineering based approaches, the RNN approaches automatically extract useful features for NER from the input tokens which are represented by the pre-trained embeddings. Furthermore, these models can capture long-term syntactic and semantic dependency among the input tokens, thus enriching the information encoded in the hidden representations. However, these advantages of RNN models also come with a cost. These RNN models usually demand a notably larger amount of training data to effectively learn the model parameters. However, this need can be partially relieved by utilizing unlabeled data or transfer learning techniques [34, 51, 52].

As mentioned earlier, the vast majority of local context-based NER models process the input texts at sentence level. Since natural language is complicated, in some text domains such as user-generated texts, a single local context may not provide sufficient contextual information for machines to perform NER effectively. In the following sub-section, we will discuss a collective approach that utilizes relevant contexts to perform the recognition.

2.1.2 Collective Named Entity Recognition

In user-generated texts such as user comments and tweets, the context of each individual document is usually limited. This is because writers often assume that the readers know the relevant contexts. Therefore, their posts/comments are usually short and do not provide sufficient contextual information when being interpreted separately. However, there are potentially relevant contexts in other comments (or documents) that can benefit NER. By this assumption, collective NER focuses on collecting these relevant contexts and extract useful information for NER.

There are relatively fewer research works that adopt this collective NER idea. The model proposed in [53] is for NER in Chinese user comments. The authors propose a CRF-based feature engineering approach. Apart from the common features listed earlier, authors further introduce a new set of co-reference features to indicate the lexical matching between a span in user comments and an entity name in the main articles. Specifically, authors first construct a dictionary based on a set of confidently extracted mentions in the news articles. The dictionary is then used to create the lexical features for a CRF model. The proposed approach demonstrates a simple way of using additional information (*i.e.*, co-reference evidence) from relevant contexts (*i.e.*, the main articles) to assist the NER in a local region. However, the proposed model faces several limitations if useful information does not exist in the main articles. Furthermore, since these lexical matching features do not consider surrounding contexts, the co-referent information can be incorrect, thus making these features become less effective.

In another work [54], the authors propose a relation phrase-based NER framework. The input to their model is a set of documents such as news articles, Yelp comments or tweets. The model first uses a data-driven phrase mining method to generate mention candidates and relation phrases. These relation phrases are then used to infer the class labels for the mention candidates. For example, consider ‘A *won the game over* B’, the relation phrase ‘*won the game over*’ indicates that both A and B are likely two sport teams. To perform inference, the relation phrases are clustered such that if two relation phrases share the same cluster, the mention classes of their head and tail arguments (mention candidate) will be similar. Propagation of mention classes is performed together with the relation phrase clustering. In the implementation, a small set of labeled data is used as seeds in the propagation. The inference is then performed in a semi-supervised setting. In their proposed model, the performance greatly depends on the relation phrase mining step. If a mention is not linked to any relation phrase or the associated relation phrase is less frequent in the corpus, the mention’s prediction is less accurate. The authors also report that NER performance of their proposed model on Yelp comments and tweets is still worse than the performance on news articles.

In summary, collective NER attempts to mine useful information from relevant contexts to assist NER in a local region. Compared to local context-based NER, collective

FIGURE 2.5: Entity linking results of four entity mentions. The ground-truth entity of each mention is highlighted in bold-face in its candidate list.

NER approach generally yields better results when working with noisy user-generated texts. However, the challenge still remains in the ways of collecting relevant contexts and extracting useful information from these contexts.

2.2 Entity Linking

Given the extracted mentions from NER, entity linking assigns to each mention a correct entity in a knowledge base. Formally, suppose that document d contains a set of mentions $\mathcal{M} = \{m_1, \dots, m_N\}$, the task of entity linking is to derive a mapping $\mathcal{M} \mapsto \mathcal{KB}$ that links each mention in $\mathcal{M}$ to a correct entity in knowledge base $\mathcal{KB}$. We denote the output of the matching as an N -tuple $\Gamma = (e_1, \dots, e_N)$ where e_i is the assigned entity for mention m_i and $e_i \in \mathcal{KB}$. As is illustrated in Figure 2.5, the challenges arise as there are multiple entity candidates that have the same or similar names with the input mentions. In general, entity linking is based on the local contexts to perform the disambiguation. However, in some settings where the local contexts are not available or the entity profile is limited, EL will need to rely on the semantic similarity between the mentions and entity names.

Most entity linking systems consist of two main steps: candidate selection, and disambiguation. For efficiency, candidate selection is used to retrieve a small set of entity candidate to be considered in the disambiguation step. The disambiguation is based on the semantic relevance between a mention (with its local context) and an entity candidate’s profile. Note that, disambiguation can be performed independently for each mention, or collectively for all mentions in a paragraph or a document. The next sub-sections will detail each of these approaches.

Entity linking is usually performed on input texts coming from general domains such as news article, reports, tweets. The knowledge graph is usually Wikipedia, which contains comprehensive descriptions of popular entities. Moreover, the anchor texts and hyperlinks in Wikipedia serve as a valuable source of annotated data used to train semantic matching models [55, 56]. On the other hand, entity linking in specific domains, such as product names, POIs, or biomedical concepts is more challenging. In these domains, the knowledge base usually contains very limited information about the entities. Therefore, the use of local contexts and entity profiles for the semantic matching is less effective. Furthermore, entities in these domains are usually mentioned in text under different surface forms, thus creating a serious challenge for both candidate selection and disambiguation. Most existing works tackle this challenge by focusing on the matching between mentions and entity names. This setting of EL is also known as entity name normalization [57–59], which will also be detailed shortly.

Not-in-list Entity. Entity linking needs to perform disambiguation for all the input mentions. However, if a mention does not have a corresponding entity in the knowledge base, a ‘dummy’ not-in-list (NIL) entity will be assigned. There are several research works that aim to cluster these NIL mentions such that the mentions belonging to the same unseen entities will be grouped together [60, 61]. However, we do not consider this setting in this thesis. Instead, we focus on the linkable mentions, similar to most other EL works.

Word-sense Disambiguation. Entity linking is highly related to word sense disambiguation (WSD) [62], which aims to identify the correct sense for a word given its local context (*e.g.*, a sentence). However, there are still two key differences between these two problems. First, although EL and WSD both tackle the ambiguity of natural language, EL focuses on disambiguating the mentions to specific entities/objects while WSD works with the abstract concepts (senses). Second, EL needs to address the variance of entity names, *i.e.*, a mention may completely dissimilar with its formal entity name. On the other hand, WSD can confidently retrieve the sense candidates for an input word from a predefined sense dictionary. Because of these differences, we will not further discuss WSD in the rest

FIGURE 2.6: Example of description and anchor texts in Wikipedia KB for the entity Tiger Woods. The description text provides concise information about the entity. Furthermore, mentions of Tiger Woods in other Wikipedia pages and their local context are often utilized to train a semantic matching model for EL. The hyperlinks can also be used to estimate the semantic similarity between two entities, based on their common citing pages.

of this thesis. However, it is worth mentioning that there are several common ideas that can be applied interchangeably in both problems [63, 64].

We have briefly described the setting of our entity linking problem. In the following sub-sections, we will detail each component and review existing EL models.

2.2.1 Knowledge Base

Knowledge base (KB) is the first component that needs to be considered in any EL systems. Different KBs store different types of information about entities. We category knowledge bases into general and domain-specific KBs. General domain KBs cover most popular entities that are often mentioned in news articles, reports, and social media texts. In contrast, domain-specific KBs cover the entities of a specific type such as diseases, genes, movies or authors.

General-domain Knowledge Bases. Wikipedia is the most popular KB used for EL when processing with texts in general domains. Entity descriptions in Wikipedia are stored in the form of natural language texts. As shown in Figure 2.6, these descriptions provide comprehensive topical information about the associated entity. Furthermore, the

TABLE 2.2: Key information stored in UMLS (a biomedical metathesaurus) for ‘Leiner disease’ entity.

Entity ID - Name	C0343047 - Leiner disease
Semantic type	Disease or Syndrome [T047]
Definition	NCI/null - A rare genetic disorder with an autosomal recessive pattern of inheritance. It is caused by the ineffective or decreased biosynthesis of the fifth complement component, C5...
Synonyms	C5 Deficiency; C5D; Complement 5 dysfunction; Erythroderma desquamativum; Generalised seborrhoeic dermatitis of infants; Leiner’s disease; Seborrheic infantile dermatitis; desquamativum, erythroderma; infantile seborrheic dermatitis; seborrheic dermatitis infantile; ...
Relations	<i>isa</i> : C2030721 - hereditary serum complement C5 dysfunction <i>finding_site_of</i> : C0020962 - Immune system <i>ssociated_with</i> : C0021376 - Chronic inflammation <i>associated_with</i> : C0232403 - Increased desquamation <i>classifies</i> : C0036508 - Seborrheic dermatitis ...

hyperlinks in Wikipedia can be used to extract statistical features such as entity popularity, or prior popularity of an entity given a mention ($P(e|m)$). The anchor texts and their hyperlinks in Wikipedia also serve as training data for semantic matching models. Wikipedia also contains structured information about entities such as categories. There are also disambiguation pages and redirect pages in Wikipedia, which can be used to construct the mapping between surface forms and potential entity candidates. Most EL systems utilize these available Wikipedia data to generate entity candidates in the candidate selection and to generate EL features for the disambiguation.

Apart from Wikipedia, DBpedia[65], Freebase [66], and YAGO [67] can also be used as the knowledge base. Different from Wikipedia, the other KBs store structured information that is extracted from Wikipedia and/or other text sources. For example, DBpedia contains facts that are extracted from the Wikipedia information boxes of about 5 million entities. Some examples of these facts are a person’s birthplace and nationality, or a country’s area, population, and GDP. Freebase is a much larger KB and contains about 43 million entities collected from multiple sources including Wikipedia and MusicBrainz. A portion of Freebase facts is manually created or revised by public users.

Domain-specific Knowledge Bases. Although multiple variants of knowledge bases are constructed, most general-domain EL systems choose Wikipedia as the KB because of its comprehensive and high-quality data. However, in domain-specific EL applications, Wikipedia may not fully cover all the entities of interest. For example, the most popular domain-specific EL is biomedical concept linking. In this domain, the CTD (Comparative Toxicogenomic Database [68]) or UMLS (Unified Medical Language System [69]) are two commonly used public KBs. CDT covers about 70 thousand biomedical entities (or concepts) including diseases, chemicals, and genes. Each entity is associated with a list of synonyms and a short definition. This KB also contains associations between entities such as chemical-disease or gene-disease iterations. On the other hand, UMLS is a much bigger KB, which is the result of combining nearly 200 different biomedical vocabularies including CDT. As a result, UMLS have information for about 1 million biomedical entities. Many of these entities do not have adequate definitions. One of the most valuable resources in UMLS is the synonym sets (see an example in Table 2.2). Most state-of-the-art biomedical concept linking systems rely on this resource in candidate selection and disambiguation [70–73]. UMLS also stores interactions and hierarchical relationships among entities. However, all these associations are relatively sparse hence their effectiveness in the EL task is not obvious [30].

There are also other domain-specific KBs including locations (*e.g.*, Foursquare), product names or job titles. While most of these KBs are not publicly available, it can be assumed that these KBs have forms of dictionaries, *i.e.*, each entity is associated with a list of synonyms (alternative multi-word expressions). The descriptions and associations may also be available but this information is less useful for EL because of their sparsity and the lack of sufficient training data.

2.2.2 Candidate Selection

Candidate selection retrieves for each mention a small set of entity candidates from the knowledge base. For efficiency, subsequent disambiguation steps only consider the entities in these candidate sets to make the final linking decisions. To this end, candidate

selection aims at high recall, while also keeping the size of the candidate sets to be manageable (usually from 20 to 50 candidates for each mention). Candidate selection should be less computationally expensive than the main disambiguation process. The selection is usually done by retrieving entities whose names (or their synonyms) are similar to a given mention. The retrieval is based on word or n-gram character level with a scoring function such as TFIDF or BM25. For Wikipedia, most EL systems [74] utilize the Wikipedia hyperlinks, disambiguation and redirect pages to retrieve the potential candidates for each mention. To ensure high recall, several EL systems [75–77] further resolve the mention’s abbreviation before the retrieval. The entity candidates can also be collected from the results of web search engines such as Google and Bing. Several EL systems [77–79] perform candidate selection by making mention queries to these search engines and obtain the entity candidates associated with the returned Wikipedia pages. To keep the size of the candidate set small, the retrieved results are truncated by a certain threshold. Combining candidate sets outputted from different retrieval methods can also be done by using a ranker. Note that entities in a candidate set usually share the same or similar names with the given mention. Therefore, the key challenge of entity linking is to select from each candidate set one entity that is referred by the mention and its local context. Next, we will discuss several approaches to perform this disambiguation.

2.2.3 Local Context-based Entity Linking

Given a mention and its local context, local context-based entity linking estimates the semantic relevance to each entity candidate. It then selects the candidate with the highest relevance score as the disambiguation entity. The linking process is performed independently for each mention and the linking result can be formulated as follows:

$$\Gamma^* = \arg \max_{\Gamma} \sum_{i=1}^N \phi(m_i, e_i) \quad (2.1)$$

where $\phi(m_i, e_i)$ denotes the local relevance score between a mention m_i (with its local context) and an entity candidate e_i . The optimization can be decomposed into the searching

TABLE 2.3: Summary of existing local context-based entity linking models. The categorization of different models is based on the methods used to represent a mention (with its local context), an entity candidate, matching between a mention and an entity candidate, and the learning models.

Approach	Mention	Entity	Mention-Entity	Learning method
Vector space model	Bags of words	Bags of words		KL-divergence [80], dot product [81], TF-IDF [82]
Feature engineering approach	Mention class	Category, popularity	Matching features, prior probability	Binary classification [79, 83, 84], or learning-to-rank (LTR) [85–87]
Neural network	Doc2vec	Doc2vec		Cosine-sim [88]
	Autoencoder	Autoencoder		LTR [89]
	CNN, or LSTM	CNN, or LSTM		LTR [90–93]
	LSTM	Embeddings	FFNN	LTR [94]

for optimal candidate for each individual mention, *i.e.*, $e_i^* = \arg \max_{e_i \in \mathcal{C}_i} \phi(m_i, e_i)$. Different local context-based EL approaches differ by the way of estimating the local relevance score. As shown in Table 2.3, we categorize these approaches into three main groups, namely vector space models, feature engineering approaches and neural networks. This categorization is based on the way of representing the mentions (with their local contexts), entity candidates, their matching features, and the learning models.

Vector Space Model. Early entity linking systems are based on the vector space model. Bag-of-words representation for a mention is formed by collecting words in its surface form and local context. On the other side, the representation of an entity candidate is derived from its description. At this point, the relevance score can be estimated through a simple scoring function such as TF-IDF, as in [82]. In another work [81], the authors utilize entities (instead of words) to construct the vector representations. Specifically, for each mention, they first identify a set of entity names appearing in its local context by a simple heuristic. In a similar way, another set of entity names are extracted from hyperlinks in the entity candidate’s Wikipedia page. The relevance between the two vector representations is estimated by the dot product. Although these vector space models are simple and easy to implement, their performance is limited if the local contexts and the entity descriptions do not share many common words/entities. Furthermore, other matching information

including the string similarity (between a mention and an entity name) and statistical features are not captured in the introduced vector representations. For this reason, most EL systems are based on the following feature engineering approach, which allows hand-crafting additional features used to model different aspects of the semantic matching.

Feature Engineering Approach. Feature engineering approach extracts a set of features for each mention-entity candidate pair. A binary classification or learning-to-rank model is trained with these extracted features and a set of labeled (training) data. The most commonly used features are the lexical matching signals between the mention and the entity candidate’s name, including string edit distance, abbreviation-matching indication, and first-name-matching indication. Furthermore, statistical features such as entity popularity and prior probability of an entity given a mention ($P(entity|mention)$) are also commonly employed in this approach. These statistical features are estimated by utilizing the Wikipedia anchor texts and hyperlinks, which are publicly available. Furthermore, there are also context-based matching features that capture the similarity between a mention’s local context and an entity candidate’s description. Apart from conventional textual similarity measures such as TF-IDF or dot product, recent EL systems [85] further utilize word and entity embeddings to capture the semantic similarity.

Neural Network Approach. Neural network based approach shows promising improvement in semantic matching tasks and entity linking. The key idea is to learn latent representations for the mention’s local context and entity candidate’s description. In [95], stacked denoising auto-encoders are used to compute these representations. On the other hand, Doc2vec [96] is utilized in [96]. In order to capture the sequence information in the mention’s local context and entity candidate’s description, other models leverage convolutional or recurrent neural network [90–93]. Given the learned representations, the relevance score can be estimated using a similarity measure such as cosine similarity. Authors in [94] further introduce an additional layer of feed-forward neural network (FFNN) with non-linear activation functions to capture the matching between the two representations. This layer is practically useful because FFNN can be trained to capture more complicated matching signals than the cosine similarity.

Discussion. Similar to the feature engineering-based models, these neural networks usually require a large amount of annotated data for training their parameters. In general, estimating the semantic relevance between a mention and an entity candidate is still a challenging problem. This is because the mention’s local context can be different from the entity candidate’s description. Furthermore, the local context can contain information that is relevant to entities other than the ground-truth entity in the candidate set. In this case, local context-based entity linking will assign a wrong entity to the given mention. There are two potential solutions to alleviate this problem. First, the semantic matching model can focus on the sequence information of words in the local context, especially the mention’s location. Furthermore, a mechanism to emphasize relevant information will benefit the semantic matching. In Chapter 4 of this thesis, we will introduce a novel attention-based neural network that employs these ideas. Second, semantic coherence between entities can be utilized to collectively disambiguate multiple mentions in a document, thus leading to a more robust and effective linking. The following subsection will discuss about this collective EL approach.

2.2.4 Collective Entity Linking

In contrast to local context-based EL approach that disambiguates each mention individually, collective EL method resolves multiple mentions in a document in a collective manner. With the assumption that entities mentioned in a document are strongly related to each other, semantic coherence was first introduced in [97]. Existing works related to collective EL can be characteristically dichotomized into two families: optimization-based approach and graph-based approach. The optimization-based approach formulates the EL as an optimization problem with additional constraints on the coherence among the selected entities. On the other hand, the graph-based approach directly approximates the EL solution by performing propagation on an entity candidate graph, thus simulating the influence of semantic coherence on the disambiguation results.

Optimization-based Approach. A common technique for finding the optimal disambiguation, denoted by Γ^* , is to maximize the local relevance of each individual assignment

$\phi(m_i, e_i)$, while enforcing pairwise semantic relatedness between all pairs of selected entities $\psi(e_i, e_j)$. The associated linking solution is expressed as the following optimization problem:

$$\Gamma^* = \arg \max_{\Gamma} \left[\sum_{i=1}^N \phi(m_i, e_i) + \sum_{i=1}^N \sum_{j=1, j \neq i}^N \psi(e_i, e_j) \right] \quad (2.2)$$

We refer to this objective as **ALL-Link** since the semantic coherence component involves all pairwise semantic relatedness. The local relevance $\phi(m_i, e_i)$ reflects the confidence of mapping mention m_i to entity e_i based on the local relevance score. As described earlier, the local relevance is computed through the string similarity between the entity mention and the entity candidate’s name, and/or the semantic similarity between the mention’s local context and the entity candidate’s description [98]. On the other hand, the pairwise relatedness $\psi(e_i, e_j)$ is often computed based on the incoming links and categories of the entities [99], normalized Google distance [100–102], or cosine similarity of entity embeddings [88].

The optimization expressed in Equation 2.2 is NP-Hard, therefore, Shen *et al.* [103] propose to use iterative substitution method (*i.e.*, hill climbing technique) to find an approximate solution. Specifically, the final assignment is obtained by iteratively substituting a linking assignment $m_i \mapsto e_i$ with another assignment $m_i \mapsto e_j$ as long as it improves the objective score. In the other works [104, 105], *Loopy Belief Propagation* (LBP) [106] is utilized. Both approaches have the complexity of $\mathcal{O}(\mathcal{I} \times N^2 k^2)$ where $\mathcal{I}$ is the number of iterations required for convergence, N and k are the numbers of mentions and candidates per mention, respectively. As such, for long documents contains hundreds of entity mentions, these algorithms can be computationally inefficient.

Other optimization approaches follow the idea proposed in [100]. These models first extract a set of unambiguous mentions and their associated entities based on the local relevance score $\phi(m_i, e_i)$. The set of confidently disambiguated entities will be used as a disambiguation context Γ' . The optimization task is decomposed into the optimization of each individual assignment. Specifically, the selected entity for each mention needs to maximize not only the relevance regarding the local context but also the coherence to the

disambiguation context, expressed as follows:

$$\Gamma^* = \arg \max_{\Gamma} \sum_{i=1}^N \left[\phi(m_i, e_i) + \sum_{e_j \in \Gamma'} \psi(e_i, e_j) \right] \quad (2.3)$$

The challenge with this approach is that the unambiguous set of mention is not always obtainable beforehand. In many cases, all mentions within a document can be ambiguous because of the noisy and ambiguous local contexts. As a remedy, models proposed in [105, 107] disambiguate a mention by considering the evidence from not only the unambiguous mentions but also the ambiguous ones. Consider the assignment $m_i \mapsto e_i$ and let $S_{ij}(e_i)$ denote the support for e_i from another mention m_j , then $S_{ij}(e_i)$ is defined as follows:

$$S_{ij}(e_i) = \max_{e_j} [\phi(m_j, e_j) + \psi(e_i, e_j)] \quad (2.4)$$

The disambiguated entity e_i for mention m_i is extracted as follows:

$$e_i = \arg \max_{e_i} \left[\phi(m_i, e_i) + \sum_{j=1, j \neq i}^N S_{ij}(e_i) \right] \quad (2.5)$$

Interestingly, the work in [105] reveals that the best performance is obtained by considering evidence from not all but only *top-k* supporting mentions. Furthermore, the authors also study the **SINGLE-Link**, which considers only the most related evidence. The associated optimization problem is expressed as follows:

$$\Gamma^* = \arg \max_{\Gamma} \sum_{i=1}^N \left[\phi(m_i, e_i) + \max_{j=1}^N \psi(e_i, e_j) \right] \quad (2.6)$$

In another work [108], fast collective linking is achieved by only considering the neighbouring connections *i.e.*, the previous and subsequent mentions of a mention. The proposed model aims to solve the following optimization:

$$\Gamma^* = \arg \max_{\Gamma} \left[\sum_{i=1}^N \phi(m_i, e_i) + \sum_{i=1}^{N-1} \psi(e_i, e_{i+1}) \right] \quad (2.7)$$

Dynamic programming, specifically Forward-Backward algorithm [109], is utilized to

FIGURE 2.7: An example of a mention-entity graph consisting of three mentions and their entity candidates. The weights between the mentions and entity candidates represent the local relevance scores, while the weights between the entity candidates represent the pairwise semantic relatedness scores.

find the optimal solution that maximizes the objective score. Although this approach works well on short text (*i.e.*, query) [108], it does not consider long-distance coherence which can be important for EL in long documents.

Graph-based Approach. Graph-based approaches solve the disambiguation problem by performing inference on a mention-entity graph. The graph is constructed with edges connecting mentions and their entity candidates. These edges are weighted by the local relevance score, *i.e.*, $\phi(m_i, e_i)$. There are also edges connecting among entity candidates, which reflect the semantic relatedness between entity pairs, *i.e.*, $\psi(e_i, e_j)$. An example of such a mention-entity graph is illustrated in Figure 2.7.

Authors in [110] cast the joint disambiguation into the problem of identifying dense subgraph that contains exactly one entity candidate for each mention. Many other works are based on random walk and PageRank [111–116] to propagate the influence from one assignment on another assignment based on the semantic coherence assumption. Specifically, authors in [117] introduce a new ‘pseudo’ topic node into the mention-entity graph to enforce the agreement between the linked entities and the topic node’s context in which the topic node is initialized using the set of confidently linked entities. In DoSeR [117], personalized PageRank is iteratively performed on the mention-entity graph. At each iteration, entity candidates having high stabilized scores are selected and added into the pseudo topic node. In general, these graph-based approaches have been shown to produce

competitive performance. However, these approaches are computationally expensive because of the cost of constructing the entity-mention graph and performing inference on it.

Discussion. Existing collective EL approaches usually assume that all pairs of entities within a document are highly related. Therefore, the proposed models often need to consider all possible pairs of entity candidates between any two mentions, which will result in a very high computational complexity, especially for long documents. To the best of our knowledge, there is no prior work that studies the coherence structure of the entities in a document. Specifically, the research question is “*to what extent the mentioned entities are related to each other? (by a specific relatedness measure)*“. In this thesis, we will address this research problem in Chapter 5. We will also propose a new tree-based objective and Pair-Linking algorithm, which are used to derive the linking results more effectively and efficiently.

2.2.5 Entity Name Normalization

Entity name normalization refers to a group of entity linking approaches that are based on the matching between the mention’s surface form and entity candidate’s name for disambiguation. Formally, given two multi-word expressions, the EL models will estimate the local relevance score based on the lexical and semantic matching between the two surface forms. Common applications of entity name normalization are product name [57], organization name [118], and job title normalizations [119]. In some specific domain such as biomedical concept linking, where the mentions and entity names usually consist of multiple words, performances of both the candidate selection and disambiguation heavily rely on the name matching performance [70–73].

The lexical similarity between names can be estimated using a simple measure such as Jaccard similarity, TF-IDF score, or string edit distance. When the semantics of words is taken into consideration, word mover’s distance (WMD) [120] is often selected as a measure. As is illustrated in Figure 2.8, WMD is based on word semantic similarity to compute the maximum alignment score between two multi-word expressions. The WMD score is high if all words in one name are semantically aligned with all words in the other name.

FIGURE 2.8: Alignment in word mover's distance measure (WMD) for two biomedical names belonging to the same entity. The arrows illustrate the flows between word pairs that have high semantic similarity scores.

As a result, WMD allows two synonymous names to have a high similarity score even they share very few words in common. Other approaches used to capture the semantic similarity are based on neural networks [73, 121]. Most of these models focus on learning the name semantic representations such that names of the same entity will have similar representations. These models will be trained on a set of annotated synonym pairs. The key objective is to learn sophisticated information in the names such as abbreviation and word morphology. Given the learned representations, candidate selection and disambiguation can be performed by retrieving the closest names in the embedding space.

There are several options to encode variable-length names/phrases into fixed-sized vector representations. Existing approaches range from simple compositions of pre-trained word representations to sequence encoding neural networks.

Average of Contextual Word Embeddings. A simple method to compute name embeddings is taking the average of their constituent word embeddings. This approach is effective for long entities names such as biomedical names since the words in these names are usually descriptive about its meaning. FastText [122] further leverages this idea by considering character n-grams instead of words. Therefore, the model can derive representations for names that contain unseen words. The effectiveness of simple compositions such as taking the average or power mean has also been verified in the problem of phrase or sentence embeddings [123–125].

Sequence Encoding Models. Sequence encoding models aim to capture more sophisticated semantics of character and word sequences. These models range from multilayer

feed-forward networks [126] to convolutional [127], recursive and recurrent neural networks [128, 129]. They also differ from the types of supervision used in training. Context-based sentence encoders [130–132] are based on the distributional hypothesis. The training utilizes sentences and their contexts (surrounding sentences), which can be extracted from an unlabeled corpus. Similar to contextual word embeddings, the derived sentence embeddings are expected to carry the contextual information. However, this contextual information does not fully reflect paraphrastic characteristic, *i.e.*, semantically similar sentences do not necessarily have identical meanings. These embeddings, therefore, are not favorable in applications that demand strong synonym identification. In contrast, supervised or semi-supervised representation learning requires annotated corpora, such as paraphrastic sentences or natural language inference data [51, 133–136]. However, most of these works focus on learning representations for sentences. In another model [137], the authors utilize pairs of paraphrastic phrases as training data, *e.g.*, ‘does not exceed’ and ‘is no more than’. To prevent the trained model from over-fitting, authors introduce regularization terms that applied on encoder’s parameters as well as the difference between the initial and trainable word embeddings.

Discussion. Sequence encoding models are trained using pairs of synonyms (two names that belong to the same entity). Since the names are usually short (containing few words), these supervised neural network models easily overfit to the training data and yield an undesired performance in test cases. This challenge has also been emphasized in previous work [123]. In Chapter 6 of this thesis, we will introduce a novel representation learning framework which is more robust and effective. We will also evaluate the learned representations in the biomedical concept linking task.

Summary. We have thus far reviewed most of existing works related to named entity recognition and linking. We start with a literature review for NER. We show that the vast majority of NER models focus on extracting mentions in a local region, within a sentence or short paragraph. There are very few works that try to aggregate the relevant contexts and perform recognition in a collective manner. We have also presented three approaches

to tackle the EL problem. The local context-based approach focuses on estimating the relevance score between a mention and an entity candidate. The collective linking approach further utilizes semantic coherence of entities to collectively disambiguate mentions in documents. On the other hand, the name normalization approach focuses on learning the name semantic representations, which are subsequently used to estimate the similarity between a mention and an entity name.

Chapter 3

Collective Named Entity Recognition

3.1 Introduction

Named entity recognition (NER) aims to extract mentions about named entities in documents and to classify each of them into one of the predefined mention classes, such as person, organization, or location. NER in the past has been focusing on extracting mentions in a local region, within a sentence or short paragraph. When dealing with user-generated text, the diverse and informal writing style makes traditional approaches much less effective. On the other hand, in many types of texts on social media such as user comments, tweets or question-answer posts, the contextual connections between documents do exist. Examples include posts in a thread discussing the same topic, and tweets that share a hashtag about the same entity. In this chapter¹, we investigate a collective NER framework that utilizes external relevant contexts to collectively perform the mention recognition.

Considering two example user comments in Figure 3.1, in the first comment, most off-the-shelf NER systems fail to extract the two mentions ‘golden state’ and ‘curry’ due to their lowercase surface forms; hence these mentions are mistakenly viewed as common nouns. Even if some systems are able to extract them, it is challenging to correctly classify their semantic types because of the limited context presented in each individual comment. However, given the fact that two mentions ‘Golden State Warriors’ and ‘Stephen Curry’ do

¹This chapter is accepted as Minh C. Phan and Aixin Sun. Collective Named Entity Recognition in User Comments via Parameterized Label Propagation. *The Journal of the Association for Information Science and Technology (JASIST)*, accepted in 2019.

FIGURE 3.1: Examples of named entity mention in two user comments in two new articles. The extracted mentions are underlined.

appear in the associated news article, an NER model can utilize the co-reference evidence to assist the recognition in this comment. In the second example, the semantic type of a new mention ‘Mother’s Day’ is also ambiguous unless the context from the main article is utilized, where the mention is described as a movie acted by Jennifer Aniston. By further investigation, our analysis on Yahoo! News user comments reveals that the average sentence length in user comments is much shorter than that in the main articles, *i.e.*, 14 words in comparison to 22 words. Moreover, 84 percent of named entity mentions in news articles are titlecase (consisting of words beginning with uppercase letters), while the percentage in user comments is 67 percent². The noisy nature of user comments, therefore, creates serious challenges for NER systems that only consider local contexts.

The presence of supporting context can be found in other types of text such as multi-format documents (reports, slides, descriptive articles) where mentions appearing in headlines or introductions can be referred later in tables or notes. Other examples are tweets that share the same hashtag, comments posted on Facebook, or even conversational text. In all these kinds of user-generated data, there are potentially relevant contexts that NER can use to support the recognition. However, based on our literature review, there have been very limited works that study the effect of supporting contexts for NER. One reason could be the lack of annotated data, where related contexts need to be included. With our effort on collecting and annotating a subset of comments in Yahoo! News, we focus our study on user comments in news articles. However, our proposed model can be applied to other domains which have the similar setting. In this chapter, we will address two

²The analysis is performed on the annotated mentions in Yahoo! News comments (details in the experiment section), and the article mentions in CoNLL03 dataset [138].

research questions: *To what degree the collective NER approach can improve the recognition performance in user comments?* and *How to effectively model and perform NER across comments?*

Our Approach. To answer the first question, we construct a mention co-reference graph for all related comments and articles. Each mention is initialized a label based on its local context. Our collective NER then performs inference on the constructed graph such that mentions that are co-referent with each other should have the same label. We evaluate the collective NER setting with several semi-supervised algorithms (*e.g.*, K-nearest neighbors, label propagation, and graph convolutional networks) and compare the performance with other non-collective NER approaches.

Second, on the road of evaluating the performance with existing inference algorithms, we observe that the results of these methods are sensitive to the quality of the constructed graph including the measures used to set the edge weights. This observation is aligned with the past reports [139, 140]. Furthermore, traditional inference methods such as label propagations are heavily used for detecting community structure in networks. In this setting, the graph is given as part of the input and the graphical contextual information is important for the task. On the other hand, in our collective NER, we want to make use of the co-reference evidence while its quality and effectiveness are not guaranteed. This is because determining whether two mentions are co-referent with each other is another challenging task; hence relying solely on preset edge weights to propagate will be ineffective. To tackle this challenge, we further propose a *parameterized label propagation* model that automatically learns the weights given the initial labels and local contexts of mentions as input. Our propagation model is trained by back-propagation using both labeled and unlabeled data. We study the performance of our approach in the Yahoo! News dataset, where comments and articles within a thread share similar context. The results show that our model significantly outperforms all other non-collective NER baselines.

3.2 Collective NER Framework

Given a news collection $\mathcal{D}$, where each document $(a, \{c_i\}) \in \mathcal{D}$ consists of an article a and a set of comments $\{c_1, c_2, \dots, c_n\}$ associated with it, we use $\mathcal{A}$ and $\mathcal{C}$ to denote the sets of all articles and all comments, respectively. The task of named entity recognition in **user comments** is to detect all text spans in $\mathcal{C}$ which refer to real-world entities; and classify each of them to one of the semantic classes in $\mathcal{T}$ (e.g., $\mathcal{T} = \{\text{person, location, organization, miscellaneous}\}$). We treat NER problem as a classification task, similar to the work in [54]. That is, given a candidate mention m , it can be either an entity mention of one class in $\mathcal{T}$, or a non-valid entity mention. Consequently, we use a label vector $y_m \in [0, 1]^T$, where $T = |\mathcal{T}| + 1$, to represent the type probability of the mention m .

Furthermore, let M be the set of (candidate) mentions extracted from all articles and comments, and G be a graph representing co-reference evidence between the mentions. Suppose that mentions in M are assigned initial labels Y^0 that are obtained from pre-trained NER annotators, our research focus is to find an inference method Φ such that the output label Y , i.e., $Y = \Phi(Y^0, G)$, can be used to derive the correct types for mentions in M . Since we focus on NER in user comments, the evaluation is performed on a subset of mentions (in M) which belong to the user-generated text. Next, we describe the construction of the mention set M , the mention co-reference graph G , and the initial label matrix Y^0 in our proposed model.

3.2.1 Mention Co-reference Graph

The mention set M contains mentions in main articles and user comments. For the ease of presentation, we use *article mentions* to refer to the mentions in main articles, and *candidate mentions* for the ones in user comments. As is illustrated in Figure 3.2, we first use an off-the-shelf NER annotator, $NER_{\mathcal{A}}$, to extract article mentions and their labels in the article set $\mathcal{A}$. We use $M_{\mathcal{A}}$ and $Y_{\mathcal{A}}^0$ to denote the set of extracted article mentions and their initial labels, respectively. Since NER in formal text has been well studied and reached a high level of accuracy, the article mentions and their labels can serve as good seeds in our model. Second, for extracting candidate mentions in comments, we employ a dictionary-based approach and aim at high recall (details about this extraction will be presented in the

FIGURE 3.2: Overall architecture of our proposed collective NER framework. A mention co-reference graph is constructed from the sets of mentions that are initially extracted from the main articles and their user comments. Parameterized label propagation is then applied on the constructed graph to refine the initial mention labels.

experiment section). The extracted mention set is denoted by M_C . Another NER annotator called NER_C is used to assign labels for M_C , and the result is denoted by Y_C^0 . Note that NER_C is one of the existing NER models that is carefully tuned to perform NER in user comments. We combine M_A and M_C as M , and stack Y_A^0 and Y_C^0 as Y^0 . The size of Y^0 is $|M| \times T$ and its i^{th} row, *i.e.*, Y_i^0 , represents the label vector for the mention m_i .

Next, we build a mention co-reference graph $G(M, E)$ which consists of all article mentions and candidate mentions (*i.e.*, $M = M_A \cup M_C$). Ideally, the edges in E connect between mentions that are co-referent with each other. However, co-reference resolution is a difficult task, especially when working with noisy user-generated text. Furthermore, one motivation of our model design is to make the model less dependent on the quality of the co-reference graph construction. Therefore, in our model, the co-reference evidence is simply determined by Jaccard similarity of mentions' surface forms (measured at the word level). Specifically, if the Jaccard similarity score between two mentions is greater than a threshold ϵ (which is set to 0.3 based on development set), we include its associated edge into the edge set E .

The number of edges in the co-reference graph can increase quadratically for long article threads that have a high degree of mention co-reference. We reduce the size of G by pruning the connections of each mention to its top k nearest neighbors (by the Jaccard similarity score). We also remove the connections between article mentions, *i.e.*, there is no edge pointing to the mentions in the main articles. This is because we assume that

the article mentions are confidently extracted. Thus, we are not going to further improve their predictions, and the focus of this work is the mentions in user comments. Note that the graph pruning step is mainly for improving the efficiency of the model without taking any significant advantage in the accuracy gain. Further analysis of the pruning’s impact will be discussed in the experiment section.

Given the initial node labels encoded in Y^0 and the co-reference graph G , collective inference can be performed in multiple ways. One approach is aggregating the labels from k-nearest neighbors (KNN) to update the label of each node in G . This KNN inference method has one limitation: that is it only considers information from 1-hop neighbors while ignoring the influence from further nodes. Another effective solution is to utilize label propagation (LP) to perform the inference on G . The propagation works based on the assumption that two nodes that are close in the graph should have similar labels. To this end, LP iteratively refines each node’s label based on its initial and neighbor’s labels. Mathematically, let α_{ij} represent the weight for propagating label from the mention m_j to the mention m_i . The formula used to update the node label y_i^t (of mention m_i) in each time step t can be expressed as follows:

$$y_i^t = \gamma \sum_{j \in \eta(i)} \alpha_{ij} y_j^{t-1} + (1 - \gamma) y_i^{t-1} \quad (3.1)$$

where $\eta(i)$ represents indexes of neighboring mentions, *i.e.*, the ones that connect to m_i in G . Furthermore, α_{ij} represents the weight for propagating label from mention m_j to m_i , and γ is the weight vector that controls the balance between the current and updated labels. Note that γ is a non-negative vector of sized T , which is parameterized for each mention class.

In our baseline, we use the Jaccard similarity scores calculated between mention surface forms as propagation weights. However, as mentioned in the introduction, these weights will have a significant impact on the performance of label propagation. Using the Jaccard similarity solely might be insufficient to capture the rich mention context that is available. Therefore, we propose a novel parameterized label propagation (PLP) which incorporates multiple contextual features (see Figure 3.3) and automatically learns the propagation weights. The next section will detail the proposed propagation method.

FIGURE 3.3: Illustration of label propagation for the mention ‘curry’ in a comment. The propagation weights between mentions are learned automatically based on the features extracted from the mentions’ initial labels and their local contexts.

TABLE 3.1: Features for learning propagation weight between two mentions m_i and m_j .

Feature	Description
Surface form similarity	Jaccard similarity between m_i and m_j with regard to their original and lowercase words.
Brown clusters	Jaccard similarity between Brown clusters [141] of words in m_i and m_j . Clusters are defined based on the path prefix of lengths 4, 8, and 12.
Context similarity	Cosine similarity of the averaged Glove embeddings [142] of context words surrounding m_i and m_j (the window size is set to 10).
Context quality	Ratio between lengths of m_i ’s and m_j ’s belonging comments (or articles).
Lexical feature	Whether m_i is substring of m_j .

3.2.2 Parameterized Label Propagation

Given a connection between two mentions, we expect PLP to produce a larger weight if the co-reference evidence between them is stronger. Furthermore, if the quality of local contexts of m_i and m_j are different, *e.g.*, m_i comes from a comment that is much shorter or noisier than m_j , the model should favor the propagation from m_i to m_j rather than the opposite direction. To this end, we define a set of features (listed in Table 3.1) to represent co-reference evidence as well as the quality of local contexts for each pair of mentions. Let $\phi(m_i, m_j)$ denote the feature vector for the directional connection from mention m_j to mention m_i . We then concatenate $\phi(m_i, m_j)$ with the initial labels of m_j and m_i to

form a contextual feature vector f_{ij} . The propagation weight α_{ij} for the connection from m_j to m_i is then calculated as follows:

$$\begin{aligned} Z_{ij} &= \tanh(f_{ij}W + B) \\ \alpha_{ij} &= \sigma(Z_{ij}v^\top + b) \end{aligned}$$

where f_{ij} is an input feature vector, which is the concatenation of $\phi(m_i, m_j)$, Y_i^0 , and Y_j^0 . Four trainable parameters are $W \in \mathbb{R}^{l \times l}$, $B \in \mathbb{R}^l$, $v \in \mathbb{R}^l$, and $b \in \mathbb{R}$ in which $l = |\phi(m_i, m_j)| + 2 \times T$. $\sigma(x) = \frac{1}{1+e^{-x}}$ is *sigmoid* activation function, which is used to rescale the propagation weight into the value range of $(0, 1)$. We also try to apply *softmax* function on top of the raw scores of incoming edges (for each mention). However this implementation yields comparable performance while being more computationally expensive. Therefore, in our proposed design and final implementation, we directly use the scores obtained after the *sigmoid* activation as the propagation weights.

Propagation Function. Equation 3.1 describes a propagation towards a single mention. In order to perform propagation for all mentions in G efficiently, we utilize matrix multiplication and rewrite the equation as follows:

$$Y^t = \gamma \odot PY^{t-1} + (1 - \gamma) \odot Y^{t-1} \quad (3.2)$$

where P is a sparse matrix that stores the propagation weights ($P_{ij} = \alpha_{ij}$). $\odot$ denotes element-wise multiplication applied to γ and each row of P . After each iteration, we normalize the label vector of each mention to maintain its probability interpretation:

$$Y_i^t = Y_i^t / \sum Y_i^t$$

Finally, after a specific ρ propagation steps, Y^ρ is used to derive the mention predictions. The class prediction for mention m_i is obtained by $\arg \max_{x=0,1,\dots,T-1} Y_i^\rho[x]$.

Loss Function. We define a loss function used to train the parameters in our model. Suppose that the ground-truth labels of all mentions in G are known and these labels are represented by Y^* . The loss function consists of cross entropy loss and L_2 regularization

of parameters:

$$L(\gamma, \theta) = -\frac{1}{N} \sum (Y^* \odot \log(Y^\rho)) + \frac{\lambda}{2} \|\theta\|^2$$

where N is the number of mentions with gold labels; $\odot$ denotes element-wise multiplication; λ is a regularization hyper-parameter. Back-propagation algorithm is used to update the model's parameters (γ , W , B , v , and b). Since obtaining gold labels for all the mentions in G can be labor-expensive, we use a subset which has annotated labels to calculate the loss. In the implementation, we set Y_i^* to the gold label of m_i if m_i appears in a manually annotated comment. On the other hand, Y_i^* is set to zero vector if m_i belongs to an unannotated comment. Intuitively, our propagation is performed on a graph consisting of both known and unknown labels in which only the known labels are used to compute the gradients and update the model's parameters.

3.3 Experiments

Our experiments are designed to study the effectiveness of the proposed collective NER framework (CoNER), and to evaluate the performance of parameterized label propagation (PLP) in comparison to other inference methods. We use our annotated Yahoo! user comment dataset to report the model's performance and analyze its behaviors.

3.3.1 Experimental Settings

As described earlier, CoNER requires two pre-trained NER annotators, NER_A and NER_C (see Figure 3.2), which are used to generate the initial mention labels. The model also relies on a candidate mention extraction module to generate a set of candidate mentions. We will describe in details each component as follows.

NER_A and NER_C. We use Stanford NER [143] as NER_A . The annotator is pre-trained on CoNLL 2003 dataset. It has been shown to achieve more than 90% accuracy for NER in formal text. The raw prediction scores returned by NER_A are used to assign the initial label vectors for the article mentions. Specifically, the label vector for a mention is calculated as the average of its tokens' scores. On the other hand, NER_C is another annotator

that is specifically trained on the annotated user comments. The annotator is also used to obtain the initial labels for all candidate mentions in user comments. In our work, we study the performance of CoNER in different settings of NER_C , including the uses of CRF and BiLSTM-CRF-based models.

Candidate Mention Extraction. Due to the noisy nature of the user-generated text, syntactical features become less effective for NER in user comments. To ensure high recall in the mention extraction step, we adopt a dictionary-based approach. We first build a mention dictionary by collecting all mentions in all main articles and user comments. The mentions are extracted by using the pre-trained NER annotators NER_A and NER_C . For each input comment, we extract all candidate mentions (text spans) that match to an entry in the dictionary. To ensure high recall, our extractor allows partial overlapping, *i.e.*, we extract $\{w_1, w_2\}$ and $\{w_2, w_3\}$ if both are found in the dictionary. However, if a match is a substring of another match, then the shorter one will be ignored. Without considering the type labels, *i.e.*, only considering the boundary of the extracted candidate mentions, the dictionary-based approach obtains the precision and recall of 0.745 and 0.843, respectively (measured on the development data). We acknowledge that more advanced techniques could be used with potentially better extraction performance [54, 144]. However, this candidate mention extraction is not the key focus of our work, and we mainly study about the effectiveness of utilizing relevant contexts to perform NER collectively.

PLP Implementation. Our proposed propagation model is implemented using TensorFlow. In training, the model’s parameters are initialized randomly, and the regularization hyper-parameter λ is set to 0.01. Adam optimizer with the learning rate of 0.001 is used to update the parameters. The hyper-parameters k and ϵ are set to 15 and 0.3 respectively, based on the development set. Similarly, the number of iterations for propagation (ρ) is set to 5. For efficiency, we use a sparse matrix data structure to store the graph connections and the propagation weights. Therefore, the model’s complexity is proportional to the graph size ($|E|$), and the training in our implementation takes less than a second per epoch.

CoNER in Different Configurations. CoNER works on top of the initial labels of mentions in the co-reference graph G (see section 3.2). The model performs inference to refine the initial labels based on information collected from neighbors. We denote the model setting as **CoNER: X + Y** where X is the base annotator NER_C used to initialize the mentions’ labels, and Y is the employed inference method.

For the base model X, we evaluate the performance with several feature-based *CRF* and BiLSTM-CRF models. For the inference method Y, we evaluate our proposed parameterized label propagation (**PLP**). We also study other inference methods, including k-nearest neighbor (**KNN**), two label propagation variants (**ADSORP**) [145] and **MAD** [146], graph convolutional network (**GCN**) [147], and graph attention network (**GAT**) [148]. Same as PLP, the input to KNN, ADSORP, and MAD is the mention co-reference graph G . The edge weights are preset by the Jaccard similarities of the mention surface forms (calculated at word level). Based on the development set, we set $k = 5$ for KNN. Other hyper-parameters in ADSORP and MAD are left as default. For neural network approaches GCN and GAT, we use the initial labels as the node feature vector. We also try to include contextual features (POS tags, orthographic signals, and Brown clusters), but this attempt leads to a poorer performance.

3.3.2 Datasets and Baselines

The collective NER approach assumes that there are shared contexts among documents. However, most existing NER datasets only contain individual documents that are sampled from a text collection. These datasets do not include connections between related documents that can be used to extract related contexts for collective NER. Hence, to evaluate the effectiveness of the collective NER approach, we create a new dataset by collecting articles and user comments from Yahoo! News website. These user comments are assumed to share the same context with their main articles.

Yahoo! User Comment Dataset. We use a dataset collected from 13,958 Yahoo! News articles including their associated user comments. The articles come from four different domains *i.e.*, ‘https://sg.news.yahoo.com/domain’ where $domain \in \{\text{singapore, malaysia, philippines, world}\}$. As this news collection is for different targeted readers, one should

TABLE 3.2: Statistics of three partitions in our annotated Yahoo! user comment dataset. 1500 articles sampled with their associated user comments. The article mentions (M_A) and candidate mentions (M_C) are extracted by pre-trained NER annotators. We randomly select 1 comment from each sampled article to annotate.

Data	Train	Dev	Test
#news articles $ \mathcal{D} $	500	500	500
#comments $ \mathcal{C} $	16,924	19,067	20,817
#article mentions $ M_A $	17,870	17,376	18,875
#candidate mentions $ M_C $	36,101	37,021	47,801
#annotated comments	500	500	500
#annotated mentions	993	1082	1131

expect that the comments include different writing styles and even different languages. We only consider comments in English and filter out a small number of comments having fewer than 3 words or more than 500 words. NLTK Tweet tokenizer is used to extract words/tokens. The average lengths of news articles and comments are 550 and 40 words, respectively.

We sample 1500 news articles from the crawled corpus. From each article, we randomly select one comment for annotating. We follow the guideline in [138] to identify and categorize each entity mention into one of four classes: person, organization, location, and miscellaneous. Statistics about the dataset are shown in Table 3.2. Note that, we do not manually annotate any news articles since the focus of this work is to improve NER in the user-generated text. The mentions in news articles are detected by an off-the-shelf NER_A , *i.e.*, Stanford NER, to be detailed shortly. We use the annotated comments in the train and development sets to train and validate the model, respectively. We report the model’s performance on the 500 annotated comments in the test set.

Baselines. We evaluate CoNER against the following NER baselines:

- **StanfordNER** [143] is a pre-trained CRF-based model trained on CoNLL03. We also evaluate another variant, *i.e.*, **CRF**, which is trained using the annotations in our annotated user comment dataset. In the retrained model, we include two sets of new features in addition to the default feature set in Stanford NER. One set is gazette-based matching features with the consideration of fuzzy matches. The gazette is built from the extracted article mentions (*i.e.*, M_A by NER_A in Figure 3.2).

The second set of features is derived from the results of Brown clustering. These features are shown to be effective for NER in informal text [53, 149].

- **TwitterNLP** [150] and **TwitterNER** [151] are another two CRF-based models with well-designed sets of features for NER on tweets.
- **BiLSTM-CRF** is a neural network model based on bidirectional long short-term memory and conditional random field. We use the implementation provided in [152], which considers both word and character-level information of the input text. We train the model using on the Yahoo! user comments training data.
- **Pattern** [153] is a pattern-based bootstrapping method which iteratively extracts new patterns and new entity mentions based on a set of initial seed mentions. We use the gold mentions in training data and the ones extracted from main articles as seeds.
- **ClusType** [54] is a relation-phase based NER method. The model performs label propagation for mentions simultaneously with multi-view relation phrase clustering. The original implementation utilizes an entity linking service (DBpedia Spotlight) to generate seed mentions. In our experiment, we reset the seeds by the gold mentions in our training data and the ones extracted from main articles.

Evaluation Metric. We report the performance regarding the micro-averaged precision, recall, and F_1 , calculated at mention level on the test set. Specifically, an extracted mention is considered correct if both its boundary and mention class are correctly identified. For all the measures, we report the micro-averaged score, i.e., aggregated across mentions but not documents, and refer the F_1 as the main metric for comparison.

3.3.3 Overall Performance

The results of the best CoNER configuration (CoNER: CRF + PLP) and baselines are reported in Table 3.3. Compare between the two CRF-based models, the retrained *CRF* with additional features notably surpasses Stanford NER model which is pre-trained on

TABLE 3.3: Performance of baselines and the best configuration of CoNER on Yahoo! comment test set. † indicates performance difference against the one in boldface is statistically significant by one-tailed paired t-test ($p < 0.05$).

Model (trained on CoNLL03 or Tweets)	P	R	F_1
Stanford_NER [143]	0.586	0.562	0.574†
BiLSTM_CRF [152]	0.594	0.631	0.612†
TwitterNLP [150]	0.517	0.355	0.421†
TwitterNER [151]	0.673	0.317	0.431†
Model (trained on Yahoo! user comments)	P	R	F_1
CRF	0.739	0.601	0.663†
BiLSTM_CRF [152]	0.654	0.639	0.646†
BiLSTM_CRF [152] (with ELMo [34])	0.653	0.649	0.651†
Pattern [153]	0.407	0.603	0.486†
Clustype [54]	0.462	0.403	0.431†
CoNER: CRF + PLP	0.768	0.660	0.710

CoNLL03 data. Furthermore, two models TwitterNLP and TwitterNER are specially designed for NER in tweets. However, they do not show a good performance in user comments (in news articles). On the other hand, BiLSTM-CRF-based model also suffers from the limited amount of training data. Its F_1 score is lower than the result of *CRF* model that uses the same set of training data. The BiLSTM-CRF with ELMo embedding is slightly better than the base model, but the trade-off is longer training and inferencing time. The noisiness in the user-generated texts also degrades the performance of the pattern-mining based approach.

On the other hand, both ClusType and our CoNER aim to recognize and classify mentions collectively by propagating labels within a mention graph. The difference is that ClusType utilizes relation phrases as evidence to infer the mention labels while our approach mostly focuses on the co-reference signals to propagate the labels. The results show that CoNER is better suited for NER task in user comments, where the relation phrase extracting is more challenging.

3.3.4 Analysis of Collective NER

We evaluate our collective NER framework with different propagation methods. We then analyze how the co-reference signals can improve the NER performance in user comments.

TABLE 3.4: Performance of different configurations of CoNER: $X + Y$ on Yahoo! comment test set. X denotes the base model used to obtain the initial labels, and Y denotes the employed inference method. † indicates that the performance difference against the one in boldface (within a row group) is statistically significant by one-tailed paired t-test ($p < 0.05$).

X is trained on CoNLL03	P	R	F_1
$X = \text{Stanford_NER}$	0.586	0.562	0.574†
CoNER: Stanford_NER + KNN	0.640	0.577	0.607†
CoNER: Stanford_NER + ADSORP	0.685	0.547	0.609†
CoNER: Stanford_NER + MAD	0.694	0.551	0.614†
CoNER: Stanford_NER + GCN	0.664	0.514	0.596†
CoNER: Stanford_NER + GAT	0.675	0.526	0.591†
CoNER: Stanford_NER + PLP	0.701	0.617	0.656
$X = \text{BiLSTM_CRF}$	0.594	0.631	0.612†
CoNER: BiLSTM_CRF + KNN	0.642	0.631	0.637†
CoNER: BiLSTM_CRF + ADSORP	0.683	0.606	0.642†
CoNER: BiLSTM_CRF + MAD	0.648	0.608	0.644†
CoNER: BiLSTM_CRF + GCN	0.726	0.587	0.649†
CoNER: BiLSTM_CRF + GAT	0.730	0.574	0.643†
CoNER: BiLSTM_CRF + PLP	0.751	0.619	0.679
X is trained on Yahoo! user comments	P	R	F_1
$X = \text{CRF}$	0.739	0.601	0.663†
CoNER: CRF + KNN	0.768	0.621	0.687†
CoNER: CRF + ADSORP	0.783	0.616	0.690†
CoNER: CRF + MAD	0.780	0.615	0.688†
CoNER: CRF + GCN	0.749	0.649	0.696†
CoNER: CRF + GAT	0.776	0.611	0.683†
CoNER: CRF + PLP	0.768	0.660	0.710
$X = \text{BiLSTM_CRF}$	0.654	0.639	0.646†
CoNER: BiLSTM_CRF + KNN	0.695	0.668	0.681†
CoNER: BiLSTM_CRF + ADSORP	0.729	0.643	0.683†
CoNER: BiLSTM_CRF + MAD	0.729	0.643	0.683†
CoNER: BiLSTM_CRF + GCN	0.757	0.624	0.684†
CoNER: BiLSTM_CRF + GAT	0.750	0.634	0.687†
CoNER: BiLSTM_CRF + PLP	0.762	0.645	0.699

Furthermore, we also discuss the effectiveness of the collective NER approach in formal text.

Performance of Different Label Propagation Methods. As shown in Table 3.4, CoNER with a collective inference method outperforms all the associated base models. Among

FIGURE 3.4: F_1 Performance of CoNER: CRF + Y (Y is an inference method such as KNN, ABSORD, MAD, GCN, GAT, PLP) in different expansions the co-reference graph G (controlled by k -nearest neighbors). Note that when $k = 0$, CoNER: CRF + Y is associated with the base model CRF .

these inference methods, PLP beats other traditional label propagation and graph neural network based approaches. In PLP, the propagation weights are learned to maximize the correctness of predictions in a semi-supervised fashion. On the other hand, the weights in other LP methods are set by heuristic rules which may not be optimal. As a result, CoNER with PLP, significantly improves the initial predictions of CRF and BiLSTM-CRF by about 7% and 8% respectively, on micro-averaged F_1 . On the other hand, a non-parametric inference method like KNN or MAD only helps to increase 3-5% in F_1 performance. More analysis about how the propagation weights in PLP are learned will be detailed in the sub-section 3.3.5.

Effect of Co-reference Graph Formation. Recall that the connections from other mentions to each mention in G is pruned to its k -nearest neighbors which have the largest Jaccard similarity scores. When k increases, it establishes more connections in G , thus providing richer contexts for extracting the mention labels. However, the expansion also potentially brings in more noise. As shown in Figure 3.4, in general, all the propagation methods benefit from expanding the co-reference graph (by increasing k). The result indicates that aggregating more information from other co-referent mentions is useful for NER in user comments.

We also evaluate the setting of removing all the connections from article mentions so that the propagation is performed among only candidate mentions in user comments.

TABLE 3.5: F_1 performance of collective NER on CoNLL03 test set. Different percentage of CoNLL03 training data is used to train the base model. The improvement is shown in terms of absolute increment score and relative error reduction.

Model	20%	60%	100%
BiLSTM_CRF	0.850	0.876	0.905
BiLSTM_CRF + PLP	0.860	0.881	0.907
Absolute improvement (error reduction)	0.010 (7%)	0.005 (4%)	0.002 (2%)

This new setting results in the F_1 performance of 0.684 (by CoNER: CRF + PLP) compared to 0.710 with the original setting. On the other hand, if we only consider the connection coming from the article mentions in the co-reference graph, the result is slightly better, at 0.695 F_1 score.

Collective NER on Formal Text. We apply the collective NER approach for news articles in CoNLL03 dataset. Different from the original setting for user comments, we only consider article mentions and the connections between them. Propagation is performed among mentions in the same main articles and there is no candidate mention (in user comments). In this setting, the task becomes refining the semantic category of the article mentions. As shown in Table 3.5, collective NER marginally improves the initial predictions. Therefore, in this kind of formal text, the base NER model can confidently extract the mentions with high accuracy; therefore, there is much less room for PLP to further improve the performance.

3.3.5 Analysis of Parameterized Label Propagation

We analyze the effect of the initial label quality on the performance of PLP. We also study the effect of number of propagation step k . Finally, we present some case studies about the propagation weights learned by our model.

Effect of Initial Labels. In PLP, the initial mention labels are obtained from a supervised NER model (such as *CRF* or BiLSTM_CRF). When more training data is available,

TABLE 3.6: F_1 performance of collective NER when additional percentages of the development data is used to train the base model.

Model	+20%	+50%	+80%
CRF	0.680	0.682	0.692
CoNER: CRF + PLP	0.716	0.721	0.722
BiLSTM_CRF	0.684	0.689	0.708
CoNER: BiLSTM_CRF + PLP	0.705	0.710	0.724

FIGURE 3.5: Performance of CoNER: CRF + PLP in different settings of number of iteration step ρ in PLP.

or more advanced NER model is used, we expect PLP still be able to improve NER performance on top of the initial labels. To validate this hypothesis, we supplement the supervised base models with additional training data taken from the development set (from 20% to 80%). Table 3.6 shows that when more training data is used, the neural network model BiLSTM_CRF starts demonstrating its advantage over the CRF-based model. In this setting, CoNER with PLP still improves the performance of the base models although the improvement is less than the case with user comment dataset.

Effect of Number of Propagation Steps ρ in PLP. As ρ increases, CoNER with PLP allows evidence from further hops in the mention co-reference graph to propagate to the target mention. At first, we expect that the peak performance is obtained after a few propagation steps. However, the result in Figure 3.5 shows that the best performance is seen at around 7 steps. It hints that either the model has learned to propagate useful signals from further nodes and/or the increase of k also results in a positive regularization

FIGURE 3.6: Distributions of propagation weights on two types of edges: those from article mentions to candidates mentions in user comments, and those connecting between candidate mentions (in user comments).

impact. Furthermore, we also observe that with a larger number of propagation steps, the recall score increases while there is a small sacrifice in the precision.

Analysis of Propagation Weights. The propagation weights in PLP are automatically learned based on a set of pre-defined features which reflect the co-reference evidence and quality of local context. As such, we expect the weights are larger when the propagations involve reliable signals. To study this behavior of PLP, we extract the propagation weights of edges starting from the article mentions to candidate mentions in user comments, and compare to those weights between candidate mentions in user comments (see Figure 3.3 for the illustration). We plot the weight distributions in Figure 3.6. It shows that propagations coming from the main articles are weighted more than those coming from user comments. This result is aligned with the observation that initial labels of the article mentions are more reliable.

In the case that an entity mention in user comments does not have corresponding co-reference in the associated main article, co-reference evidence will mainly come from other user comments. We further extract some propagation weights between candidate mentions in user comments as case studies (see Figure 3.7). In case I, mention ‘Kobe’ is initially mistakenly labeled as MISC by *CRF*. However, in another comment that is related, there is stronger evidence for recognizing and classifying the co-referential mention

FIGURE 3.7: Case studies of propagation weights between candidate mentions in user comments. The mentions are shown with their local contexts. The labels in square brackets indicate the initial predictions by *CRF* model.

‘Kobe Bryant’. PLP successfully learns to give more weight for the connection from the less ambiguous mention to the more ambiguous mention. A similar explanation can be derived for case II.

3.4 Summary

In this chapter, we have studied the idea of utilizing relevant contexts to perform NER in a collective manner. We show that this approach is effective when dealing with the noisiness of user-generated texts such as user comments. Our proposed collective NER framework (CoNER) is based on a mention co-reference graph and label propagation to refine the mention labels. Different from other inference approaches, our parameterized label propagation (PLP) allows the propagation weights to be learned automatically based on the mentions’ initial labels and their contextual features. The experiment on Yahoo! News user comments has demonstrated the robustness and effectiveness of the proposed model.

On the other hand, one limitation of CoNER is that the framework still relies on the candidate mention generation and pre-trained NER annotators to obtain the initial mention labels. Therefore, the proposed approach could be less appealing in some practical scenarios where an end-to-end extraction is required. However, one key contribution

of our work is that we have verified the effectiveness of using relevant contexts for NER. Furthermore, our proposed propagation PLP is potentially applicable as a semi-supervised inference method for other classification tasks such as text classification, or node classification in network graphs.

Chapter 4

Local Context-based Entity Linking

4.1 Introduction

Mentions of named entities that appear in texts are usually ambiguous because of their polymorphic nature *i.e.*, the same entity may be mentioned under different surface forms, and the same surface form can refer to different named entities. We use a sentence from Wikipedia as an example: “*Before turning seven, Tiger won the Under Age 10 section of the Drive, Pitch, and Putt competition, held at the Navy Golf Course in Cypress, California*”. Without considering the local context, the word ‘Tiger’ can refer to American golfer *Tiger Woods*, budget airline *Tiger Air*, or beer brand *Tiger Beer*. Considering its context, the mention ‘Tiger’ in this sentence should be linked to *Tiger Woods*.

Formally, for each mention extracted from the NER process, local context-based entity linking attempts to disambiguate the mention to a correct entity based on the semantic relevance between the mention’s local context and an entity candidate’s profile. Various methods have been proposed to model this semantic relevance, ranging from simple vector space models, IR-based approaches, to supervised models such as binary classification, learning-to-rank, and probabilistic models [98]. Furthermore, deep neural network (DNN) has also been investigated for EL and promising results have been achieved [85, 95, 154, 155]. These works are based on the general idea that the disambiguation problem is a semantic matching problem. However, the existing solutions do not fully utilize the information presented in the mention’s context. First, the mention’s

position is ignored in most previous models. For example, in [95], a DNN learns the representation of the local context without specifying which mention to be focused on. If the context consists of two or more mentions, all the mentions are viewed as identical in the disambiguation. Second, existing approaches (*i.e.*, bag-of-word) often ignore the word order in local contexts while word ordering is critical for natural language understanding.

In this chapter¹, we present a neural network that takes into consideration the mention’s positional information, and word order to model the mention’s local context. On the other side, entity embeddings and their textual descriptions are utilized to construct the semantic representations for entity candidates. To assist the matching in challenging cases where there are noises in the mention’s context, we further employ the *attention mechanism* into the designed model. To the best of our knowledge, we are the first to employ attention mechanism to model the semantic relevance in EL task.

4.2 Joint Learning of Word and Entity Embeddings

To start with, we describe a way to obtain semantic representations for words and entities. The learned representations will be used as inputs in our proposed semantic matching model. Our idea is to jointly learn word and entity embeddings. There are two key motivations for this approach. First, the semantic relations between entities and context words are encoded in their embeddings during the representation learning process, thus enabling a semantic matching model to utilize this valuable information. Second, joint training has been shown to improve the quality of both word and entity embeddings [85, 156].

Word embedding models [157] generate a continuous representation for every word. Two words that are close in meaning are also close in the embedding vector space. These models are based on the distributional hypothesis, which states that words are semantically similar if they co-occur often with the same context words [157]. Correspondingly, we extend this assumption to entities, *i.e.*, two entities are semantically related if they are found in analogous contexts. Here, the context is defined by the surrounding words or entities.

¹This chapter is published as Minh C. Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. NeuPL: Attention-based Semantic Matching and Pair-Linking for Entity Disambiguation. *The 26th ACM International Conference on Information and Knowledge Management (CIKM)*, 1667-1676, 2017.

We employ skip-gram model [157] to jointly learn the distributional representations of word and entity. Formally, let $\mathcal{T}$ denote the set of tokens. Token $\tau \in \mathcal{T}$ can be either a word (e.g., Tiger, Wood) or an *entityID* (e.g., [Tiger_Wood]). Suppose $\tau_1, \dots, \tau_N$ is a given sequence of tokens, the model aims to maximize the following average log probability:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^N \sum_{-c \leq j \leq c, j \neq 0} \log P(\tau_{i+j} | \tau_i) \quad (4.1)$$

where c is the size of context window, τ_i denotes the target token, and τ_{i+j} is a context token. The conditional probability $P(\tau_{i+j} | \tau_i)$ is defined by the following softmax function:

$$P(\tau_O | \tau_I) = \frac{\exp(v'_{\tau_O} \top v_{\tau_I})}{\sum_{\tau \in \mathcal{T}} \exp(v'_{\tau} \top v_{\tau_I})} \quad (4.2)$$

where v_{τ} and v'_{τ} are the ‘input’ and ‘output’ vector representations of τ , respectively. After training, we use the ‘output’ v'_{τ} as the embedding for a word or entity.

The training requires a corpus consisting of sequences of tokens (including both words and entities). We create a ‘token corpus’ by exploiting the hyperlinks in Wikipedia. Specifically, for each sentence in Wikipedia that contains at least one hyperlink to another Wikipedia entity, we create an additional sentence by replacing each anchor text with its associated entity. Furthermore, for each Wikipedia page, we also create a ‘pseudo sentence’ formed by the sequence of entityIDs mentioned in this page, in the order of their appearances. For example, assume that the Wikipedia page of *Tiger Wood* contains only 2 sentences: “*Woods*_[Tiger_Woods] was born in *Cypress*_[Cypress,_California]. He has a niece, *Cheyenne Woods*_[Cheyenne_Woods].”, the following sentences are added into our corpus.

- Wood was born in Cypress. He has a niece, Cheyenne Woods.
- [Tiger_Woods] was born in [Cypress,_California]. He has a niece, [Cheyenne_Woods].
- [Tiger_Woods] [Cypress,_California] [Cheyenne_Woods].

These token sequences are subsequently fed as inputs to the skip-gram model (the training details are presented in Section 4.4.1). Outputs of this baseline are word and entity embeddings. We refer them as pre-trained embeddings and they are used to construct representations for the mention’s local context and entity candidate.

FIGURE 4.1: Neural network architecture for learning the semantic relevance score between a mention’s local context and an entity candidate. Two unidirectional LSTMs are used to encode the left- and right-side local contexts of the mention. On the other hand, entity embeddings and another LSTM unit are used to construct the representation for an entity candidate. Attention mechanism and feed-forward neural network (FFNN) are used to capture the matching between these two representations. Finally, the sigmoid matching score $\sigma(m_i, e_i)$ is combined with prior probability score $P(e|m)$ to obtain the final semantic relevance score.

4.3 Attention-based Semantic Matching Architecture

Our proposed semantic matching model is depicted in Figure 4.1. The model estimates semantic relevance score for a pair of mention’s local context and entity candidate. The model first uses two LSTM networks to encode the left- and right-side local contexts of the mention. On the other hand, entity embeddings and another LSTM network are used to construct the latent representation for the entity candidate. This representation of entity candidate is then used by an attention module to emphasize the relevant matches in the mention’s local context. All the representations are aggregated (by max-pooling and concatenation) and passed to a feed-forward neural network (FFNN) which aims to capture the semantic matching between the representations. The output of the FFNN is a scalar that represents the semantic matching score. This semantic matching score is linearly combined with a prior probability score $P(e|m)$ to obtain the final semantic relevance score. The whole model is trained using the texts and hyperlinks in Wikipedia. We detail each individual component in our proposed model as follows.

Representation for A Mention’s Local Context. The local context of a mention is defined by the words that appear within a window size c on both sides of the mention. Specifically, let $\langle w_\ell^c, \dots, w_\ell^1, m, w_r^1, \dots, w_r^c \rangle$ be the local context for mention m , we define its *left-side context* and *right-side context* as $\langle w_\ell^c, \dots, w_\ell^1, m \rangle$ and $\langle m, w_r^1, \dots, w_r^c \rangle$, respectively. For simplicity, we use a single token m to represent a mention here (and also in Figure 4.1). However, in implementation, it can consist of multiple tokens.

Shown in Figure 4.1, two separate LSTM networks are used to encode the mention’s left- and right-side contexts respectively. The left-side context is passed in forward direction *i.e.*, $w_\ell^c \rightarrow w_\ell^{c-1} \rightarrow \dots \rightarrow w_\ell^1 \rightarrow m$, while the right-side context is passed in backward direction *i.e.*, $m \leftarrow w_r^1 \leftarrow \dots \leftarrow w_r^{c-1} \leftarrow w_r^c$. By doing so, we align mention m at the end of each sequence, so that LSTM is aware of its position. This is important because the local context may contain more than one mention, and the model needs to focus on the right mention for correct linking. For example, given two (underlined) mentions in a sentence: ‘The Tiger Woods Foundation was established in 1996 by Woods’, without specifying the mention locations, the context can matches both entities *Tiger Woods Foundation* and *Tiger Woods*, thus leading to wrong disambiguations of each individual mention.

Compared with the model proposed in [158], where additional positional embedding is used to encode a mention’s location, the method we used to align the mention at the end of each side of local context is simpler, and does not add much computational complexity to the model. Our strategy is similar to the idea of target-dependent LSTM used for sentiment classification in [159]. However, there with two key differences. First, the model in [159] uses the last hidden vector of LSTM’s output to represent the context, while we max-pool all hidden vectors of the LSTM on each side of the context. Second, we employ the attention mechanism to improve the quality of representation. Specifically, the hidden vectors obtained from LSTM will pass through an attention module to emphasize the relevant matches. Then, max-pooling is applied over the new hidden states to produce a fixed-sized representation for the local context. We employ max-pooling instead of weighted summation because we want to emphasize only the most relevant matches in the local context. Furthermore, max-pooling yields better performance in our experiment.

Representation for An Entity Candidate. To build the representation for an entity candidate, we exploit the pre-trained embeddings of entities in Section 4.2. Because we treat entities similar to words in training, an entity embedding encodes not only semantic information but also syntactic knowledge about how the associated entity is mentioned. For example, entities about geographic location will be more likely to be placed close to prepositions such as ‘in’ or ‘at’ in the embedding space. We further complete the entity candidate representation by including its description. Specifically, we take the first 150 words from its Wikipedia page and use a unidirectional LSTM network with max-pooling (see Figure 4.1) to learn the description representation. The learned vector is concatenated with the pre-trained entity embeddings to form the final entity candidate representation.

Because the entire DNN model is pre-trained using Wikipedia data, the training samples are biased toward the standard way of putting hyperlinks in Wikipedia. In the designed model, we do not explicitly specify the entity’s name in an entity description, neither declare any mention’s boundary in its local context. This prevents the model from over-fitting the training data; hence the model generalizes better for semantic matching with data from outside of Wikipedia.

Attention Mechanism. Our goal is to estimate the matching score given: (i) a vector representing a mention’s local context, and (ii) a vector encoding an entity candidate. However, the local context usually contains irrelevant information that can downgrade the matching performance. As a remedy, we propose to use the attention mechanism to alleviate this problem.

The idea of attention is to learn to emphasize the more relevant parts of the inputs to a given attention vector. The mechanism has been successfully applied in most NLP tasks including translation [160], summarization [161] and question-answer matching [162–164]. In our model, we use entity candidate’s representation as the attention vector to highlight the relevant parts in mention’s local context. Specifically, given a hidden vector h_t from a LSTM block at time step t , and an attention vector p (*i.e.*, an entity candidate’s

representation, see Figure 4.1), the re-weighted hidden vector $\bar{h}_t$ is calculated as follows:

$$z_t = \tanh(V_p p + V_h h_t) \quad (4.3)$$

$$s_t = \frac{\exp(v^\top z_t)}{\sum_{t'=1}^n \exp(v^\top z_{t'})} \quad (4.4)$$

$$\bar{h}_t = s_t h_t \quad (4.5)$$

where V_p , V_h and v are the attention parameters that will be learned during the model training. Intuitively, h_t will be given more weight if it is more relevant to the attention vector p . In our context, the attention mechanism emphasizes the embedded information in the local context (*i.e.*, h_i) that are more relevant to the entity candidate.

Matching by FFNN. Given the representations of a mention’s context and an entity candidate, we concatenate the representations and pass the concatenated vector to a two-layer feed-forward neural network (FFNN) with a tanh activation after each layer. Finally, another linear transformation with sigmoid activation is applied to get a scalar value that represents the semantic matching score.

Training Objective. Let $0 < o < 1$ denote the final output and $g \in \{0, 1\}$ be the groundtruth label indicating positive/negative sample. The proposed deep neural network is trained with the following binary cross-entropy (CE) loss function:

$$L(o, g) = g \log(o) + (1 - g) \log(1 - o) \quad (4.6)$$

where $0 < o < 1$ is the output of the last hidden layer after going through sigmoid activation.

Incorporating Prior Probability Knowledge. Prior probability $P(e|m)$ represents the likelihood of a mention with a given surface form m being linked to an entity e . It can be estimated from the anchor texts and hyperlinks in Wikipedia [98]. Although $P(e|m)$ completely ignores the surrounding context of m , it is recognized as one of the important features for entity linking [165]. Especially, for popular entities such as countries, famous people, they generally do not require supporting context when being mentioned in

texts. Therefore, the local contexts in these cases can be general, thus making the semantic matching based on context becomes less effective. On the other hand, such mentions can be easily disambiguated by adopting the prior probability knowledge. To this end, in our proposed model, we use a linear combination of the semantic matching score computed by DNN and the prior probability as a final semantic relevance score:

$$\phi(m_i, e_i) = (1 - \alpha)\sigma(m_i, e_i) + \alpha P(e_i|m_i) \quad (4.7)$$

where α is a weight factor, and $\sigma(m_i, e_i)$ is the output of DNN given m_i 's context and entity e_i (see Figure 4.1).

4.4 Experiments

We aim to evaluate the performance of our proposed local context-based entity linking approach by comparing it with other semantic matching baselines. We also include the performance of several existing collective EL systems. To this end, we first describe our experiential setting including the entity candidate selection and model training details. We then detail the datasets and our baselines. Finally, we report and analyze the model performances.

4.4.1 Experimental Settings

Entity Candidate Selection. Similar to the candidate selection in other works [98, 117, 166], we select the entity candidates purely based on the surface form similarity between a mention and an entity name including all its synonyms. We used a dictionary-based technique for this candidate retrieval [98]. The dictionary is built by exploiting the entity titles, anchor texts, redirect pages, and disambiguation pages in Wikipedia. The entries in the dictionaries are indexed using their n-grams. If a given mention does not present in the dictionary, we use its n-grams to retrieve the potential candidates. We further improve the recall of candidate generation by correcting the mention's boundary. In several situations, a given mention may contain trivial words (*e.g.*, the, Mr. , CEO, president) that are

not indexed by the dictionary. We use an off-the-shelf NER annotator to refine the mention’s boundary in these cases.² As in [167], we also utilize the NER output to expand the mention’s surface form. Specifically, if mention m_1 appears before m_2 and m_1 contains m_2 as a substring, we consider m_1 as an expanded form of m_2 , and candidates of m_1 will be included to the candidate set of m_2 .

We train a learning-to-rank model (Gradient Boosted Regression Trees [168]) to prune the candidate set to a manageable size. For each pair of (mention, candidate) *i.e.*, (m, e) , we use the following statistical and textual features for ranking.

- Prior probability $P(e|m)$: $P(e|m)$ is the likelihood that the mention with surface form m being mapped to entity e . As mentioned earlier, $P(e|m)$ is pre-calculated based on the anchor texts and hyperlinks in Wikipedia.
- String similarity features: These features include (i) string edit distance, (ii) whether m exactly matches entity e ’s name, (iii) whether m is a prefix or suffix of the entity name, and (iv) whether m is an abbreviation of the entity name. Note that the string similarity features are calculated for the original mention as well as the boundary-corrected mention and the expanded mention described earlier.

We use the IITB labeled dataset [169] to train the learning-to-rank model and take the top 20 scored entities as the final candidate set for each mention. Considering fewer candidates per mention will lead to a low recall while using more candidates can degrade the disambiguation accuracy. Similar observations are also reported in [104, 117]. Note that the learning-to-rank model is used to rank and select entity candidates, thus it aims at high recall. Since the model does not consider the contextual information, it does not guarantee that the first-ranked entity is the optimal match for each mention. Therefore, we still need a local context-based semantic matching model to re-rank the candidate set.

Pre-trained Word and Entity Embeddings. We use Gensim, an efficient skip-gram library to train word and entity embeddings. The training corpus is the ‘token corpus’ described in Section 4.2. The embedding dimension is set to 400, the window size is 5, and the number of training iteration is 5. For preprocessing, we remove all numeric tokens and

²We used the Stanford NER tool in this work.

TABLE 4.1: Hyperparameter setting used in our proposed semantic matching model.

Neural network setting		Training setting	
LSTM hidden state size	384	Dropout	0.3
No of hidden layers in FFNN	2	Epochs	30
Size of each hidden layer in FFNN	700	Batch size	256
Activation for hidden layer in FFNN	<i>tanh</i>	Optimizer	<i>Adam</i>

the tokens that appear less than 5 times in the whole corpus. All the words are lowercased. The final vocabulary contains 2,689,534 words and 2,863,704 entities. For the entities that are less frequent and not included in the vocabulary, we use a zero vector to represent their embeddings in the semantic matching model.

Neural Network Setting. We set the context window size $c = 20$, meaning that the left 20 and the right 20 words around a mention are used as the local context words. We ignore the sentence boundary and try to include as much as local context information. Zero paddings are employed if the context has fewer words. On the other hand, the first 150 words in the entity’s Wikipedia page are taken as the entity description. All these words are lowercased and the numbers are removed. We use the pre-trained word/entity embeddings as the inputs to the neural network. The embeddings are fixed during training and only the neural network’s parameters are updated.

We use unidirectional LSTM to encode word sequences. We also try bidirectional LSTM. However, it results in a negligible improvement while doubling the number of network parameters and taking longer time to train. Hence, we adopt the unidirectional LSTM. Finally, we add dropout layers after each input layer and hidden layer to prevent the model over-fitting. The setting of other neural network hyperparameters is listed in Table 4.1.

Neural Network Training. We use Wikipedia (dumped on 01-Jul-2016) as the target knowledge base. Wikipedia consists of 5,187,458 entities. The neural network model is trained using data solely from Wikipedia. Once trained, it can be used to disambiguate mentions in documents from general domains, *e.g.*, web pages, news, and tweets. Specifically, we utilize the text and hyperlinks in Wikipedia. For each entity, we randomly collect

TABLE 4.2: Statistics of the 7 test datasets used in experiments. $|D|$, $|M|$, Avg_m , and $Length$ are the number of documents, number of mentions, average number of mentions per document, and document length in number of words, respectively.

Dataset	Type	$ D $	$ M $	Avg_m	Length
Reuters128	news	111	637	5.74	136
ACE2004	news	35	257	7.34	375
MSNBC	news	20	658	32.90	544
DBpedia	news	57	331	5.81	29
RSS500	RSS-feeds	343	518	1.51	30
KORE50	short sentences	50	144	2.88	12
Micro2014	tweets	696	1457	2.09	18

up to 100 of its anchor texts (*i.e.*, mentions) together with context words as positive training samples. Because of computational resource limitation, we train the model on the aggregated entity candidates derived from the seven datasets (see Table 4.2). Since candidate generation for each mention is independent of each other, all entity candidates can be pre-determined for a given test dataset (see the candidate selection setting). The aggregated candidate set contains 27,444 entities, from which we collect 1,108,524 positive samples through Wikipedia hyperlinks. For each positive sample, we create 4 negative samples by replacing the correct entity with another randomly selected entity from the mention’s candidate set. As a result, the neural network is trained by using 5,542,620 samples in total. Training of the neural network model takes about 4 days using a single Nvidia K40 GPU. Note that the training uses data from Wikipedia. As shown in Table 4.2, no test datasets contain Wikipedia pages. That is, the test datasets are not seen in training.

Given the trained semantic matching model, we combine the prediction score with the prior probability as expressed in Equation 4.7. We set the value for α using 5-fold cross-validation. Specifically, we search for the optimal α value in a range from 0 to 1, with a step of 0.05.

Rule-based Disambiguation. Because numeric tokens are removed in preprocessing, all numeric mentions are mapped to numeric entities regardless of the local context. For example, mention ‘12’ will be disambiguated to entity *12 (number)*. Another rule we used is for several mentions of news agencies. These mentions usually appear at the beginning or the ending of a news article (in some benchmark datasets). They can be easily detected

and disambiguated by using the following rule-based mappings: ‘reuters’ $\mapsto$ *Reuters*, ‘associated press’ $\mapsto$ *Associated Press*, and ‘afp’ $\mapsto$ *Agence France-Presse*.

4.4.2 Datasets and Baselines

Datasets. We evaluate our proposed model on 7 benchmark datasets. These datasets come from different domains and contain both short and long texts, as well as formal and informal texts (see Table 5.3). For all the datasets, we only consider the mentions whose linked entities present in the Wikipedia dump, hence we do not address the not-in-list entity in this work. The same setting has been used in most of existing works [85, 104, 117, 166]. Next, we describe each dataset in our experiment.

- **Reuters128** contains 128 economic news articles taken from the Reuters-21587 corpus³. All the mentions are carefully labeled by experts. Thus, this dataset can be viewed as having the highest quality. There are 111 documents in this dataset that contain linkable mentions, *i.e.*, can be mapped to Wikipedia entities.
- **ACE2004** is a subset of ACE2004 co-reference documents annotated by Amazon Mechanical Turk. The corpus has 35 documents in which each document contains 7 mentions on average.
- **MSNBC** is created from MSNBC news articles. Each document contains a lot of mentions and many of them refer to the same entity. This dataset has the highest number of mentions per document (among 7 datasets used in this experiment). Each document contains 33 mentions on average.
- **DBpedia Spotlight (DBpedia)** is another news corpus. Apart from the named entity mentions, this dataset also contains many common nouns such as parents, car, dance, etc. We will retain this dataset in this experiment to evaluate the generalizability of our proposed model to these kinds of mentions. Note that these common nouns also have the corresponding Wikipedia entities.
- **RSS500** is RSS feeds - short formal text collected from a wide range of topics *e.g.*, world, business, science, etc.

³<http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html>

- **KORE50** is a subset of the AIDA corpus. It contains 50 short sentences of various topics including music, celebrities, and business. Most mentions are the first names referring to persons. This dataset is purposely created with highly ambiguous mentions. Thus, it is the most challenging short-text dataset.
- **Microposts2014 (Micro2014)** is a collection of tweets, introduced in the ‘Making Sense of Microposts 2014’ challenge. Each document contains very few mentions. Furthermore, the surrounding context of each mention is limited because of the shortness of tweets.

Baselines. We compare our proposed model against 5 state-of-the-art EL systems whose results are also reported with the Gerbil benchmark framework. We acknowledge that there are other systems whose results are not reported with the framework. As the evaluation itself is a complicated issue [170, 171], we adopt Gerbil following the most recent studies [104, 117], such that all systems can be compared in future through the common protocol.

Note that, apart from the local relevance score, most EL baselines further use semantic coherence of entities to improve the disambiguation accuracy. At this point, we do not evaluate our proposed model with this setting. Instead, we will investigate it in the next chapter where we focus on collective EL approach. In this experiment section, we report the performance of our proposed model which is a local-context based semantic matching model. We compare our model with the following baselines:

- **AIDA** [110] uses prior probability and local context-based similarity to estimate the local relevance score. Different from our model, their local context-based similarity is calculated through keyphrase-base similarity (between context words and entity keyphrases) and syntax-based similarity (between the mention and entity categorical information). The proposed approach further uses a dense subgraph algorithm to collectively identify mention-entity mappings based on the semantic coherence of entities.

- **Kea** [172] builds fine-granular context model and utilizes heterogeneous text sources as well as the text created by an automated multimedia analysis. The disambiguation is performed by selecting the entity candidate that has the highest probability according to a predetermined context.
- **WAT** [114] is based on n-gram Jaccard similarity between the mention's surface form and then entity name, in addition to the prior probability. Other string-matching scores such as BM25 calculated between the mention's context and the entity's description are also considered. This baseline further implements the collective linking idea with two configurations: using graph-based algorithms (PageRank or HITS), and a voting-based algorithm.
- **PBoH** [104] is a light-weight disambiguator that is based on a probabilistic graphical model to perform collective EL. The model extracts Wikipedia statistics about the co-occurrence of words and entities to disambiguate the mentions.
- **DoSeR** [117] estimates the semantic relevance by the cosine similarity between Doc2vec representations of the mention's local context and the entity candidate's description. This approach carefully designs a novel collective disambiguation algorithm based on Personalized PageRank, which contributes greatly to the overall EL performance.

Apart from these existing EL models, we also implement three more baselines. First, we include a simple baseline, **Prior**, which ranks the entity candidates solely based on the prior probability $P(e|m)$. Note that the performance of this Prior baseline can be used to judge the ambiguity of mentions in each dataset. Second, we consider **AvgEmb**, which estimates the semantic similarity between a mention's local context and an entity candidate's description by the cosine similarity of two average word embedding representations. The cosine similarity score replaces the semantic matching score $\sigma(m_i, e_i)$ in Equation 4.7. It is worth mentioning that this bag-of-word-embedding approach is also used in several other works [85, 173, 174]. Third, we implement **Xgb**, a feature engineering-based baseline that uses Gradient Boosting Tree (GBT) as a learning-to-rank model. Similar to [85], our feature set includes the prior probability $P(e|m)$, several string-similarity features based on

TABLE 4.3: F_1 performance of NeuL and all baselines. The best results are in boldface and the second-best ones are underlined.

System	Reuters128	ACE2004	MSNBC	DBpedia	RSS500	KORE50	Micro2014	Average
AIDA [110]	0.599	0.820	0.759	0.249	0.722	0.660	0.433	0.606
Kea [172]	0.654	0.796	0.854	0.736	0.709	0.620	0.639	0.715
WAT [114]	0.660	0.809	0.795	0.671	0.700	0.599	0.604	0.691
PBoH [104]	0.759	0.876	0.897	0.791	0.711	<u>0.646</u>	0.725	0.772
DoSeR [117]	0.873	0.921	0.912	<u>0.816</u>	<u>0.762</u>	0.550	0.756	0.798
Prior	0.697	0.861	0.781	0.752	0.702	0.354	0.650	0.685
AvgEmb	0.793	0.896	0.823	0.780	0.708	0.419	0.725	0.735
Xgb	0.776	0.872	0.834	0.818	0.756	0.496	0.789	0.763
NeuL	<u>0.869</u>	<u>0.906</u>	<u>0.904</u>	0.807	0.770	0.551	<u>0.766</u>	<u>0.796</u>

the mention’s surface form and the entity’s title, and the embedding similarity between the mention’s local context and the entity candidate. The raw scores obtained from this GBT model will be used to rank the entity candidates.

Evaluation Metrics. For evaluation, we use Gerbil benchmark framework (Version 1.2.4) [170]. We run Gerbil on a local PC, with the same setting for all the EL baselines. Note that some of the results of our baseline models are slightly different from the ones reported in previous works [104, 117]. This is because Gerbil (Version 1.2.4) has improved the entity matching and entity validation procedures to adapt to the knowledge base’s changes over time.⁴

We consider three commonly used metrics: precision P , recall R , and their harmonic mean F_1 . Specifically, let Γ_g be the set of groundtruth assignments $m_i \mapsto t_i$ and Γ^* be the linkings produced by an EL system, the three measures are computed as follows:

$$P = \frac{|\Gamma^* \cap \Gamma_g|}{|\Gamma^*|} \quad R = \frac{|\Gamma^* \cap \Gamma_g|}{|\Gamma_g|} \quad F_1 = \frac{2 \times P \times R}{P + R}$$

For all these measures, we report the micro-averaged score (*i.e.*, aggregated across mentions, not documents), and refer the micro-averaged F_1 as the main metric for comparison.

4.4.3 Overall Performance

Table 4.3 reports the micro-averaged F_1 performances of NeuL and all the baselines. Compare with other three local context-based EL methods (Prior, AvgEmb and Xgb), our proposed model NeuL outperforms the rest. On the other hand, compare with other baselines that implement collective EL idea (such as Kea, WAT, PBoH, and DoSeR), NeuL is only worse than DoSeR on the average performance score. On formal-text datasets (ACE2004, DBpedia and MSNBC) where the mentions are relatively popular and the local contexts are clear, NeuL yields comparable performance with DoSeR. However, on short-text datasets such as RSS and Micro2014, where the number of entities in a document is limited, collective linking baselines including DoSeR become less effective. In these cases, disambiguation performance greatly relies on the estimation of local semantic relevance score. Technically, DoSeR uses Doc2vec [175] to encode the mention’s local context. Doc2vec is simple but it ignores the word order as well as the mention’s location in the local context. On the other hand, NeuL implements two LSTMs to capture the positional information and incorporates the attention mechanism to filter out noise. As expected, NeuL outperforms DoSeR in these short-text datasets although NeuL does not employ the collective EL idea.

Performance of the probabilistic graphical model PBoH is better than the first three baselines (in Table 4.3) but it is worse than the methods powered by neural networks such as NeuL and DoSeR. PBoH utilizes the pairwise co-occurrence statistics of words and entities. Semantic similarity between different words/entities is not well captured. In contrast, both NeuL and DoSeR utilize word and entity embeddings to estimate the semantic relevance score.

Detailed performance of NeuL regarding the precision, recall and F_1 metrics is shown in Table 4.4. Since we only consider linkable mentions (see Section 4.4.2), in most cases, our proposed model will take the entity candidate with the highest relevance score as the disambiguation for each mention. There are only a few cases where the candidate selection is empty. These situations can happen if the mention’s surface form is completely unseen

⁴<http://svn.aksw.org/papers/2016/ISWCGerbilUpdate/public.pdf>

TABLE 4.4: Micro-averaged precision, recall and F_1 performance of NeuL.

Data set	Precision	Recall	F_1
Reuters128	0.877	0.861	0.869
ACE2004	0.911	0.901	0.906
MSNBC	0.905	0.903	0.904
DBpedia	0.811	0.803	0.807
RSS500	0.771	0.769	0.770
KORE50	0.551	0.551	0.551
Micro2014	0.772	0.760	0.766

TABLE 4.5: F_1 performance of our proposed model and two variants: one with single-directional LSTM used to encode the local context, and one without the attention mechanism.

Dataset	Full-model	Single LSTM	No Attention
Reuters128	0.869	0.846	0.832
ACE2004	0.906	0.898	0.894
MSNBC	0.904	0.901	0.895
DBpedia	0.807	0.792	0.795
RSS500	0.770	0.772	0.721
KORE50	0.551	0.572	0.572
Micro2014	0.766	0.775	0.769
Average	0.796	0.794	0.783

and does not match with the existing names in Wikipedia. Because of this reason, the precision scores are often equal or slightly higher than the recall scores (in Table 4.4).

4.4.4 Ablation Study and Analysis

Ablation Study. In our proposed neural network model, we use two LSTMs to encode positional information of the mention’s local context as well as exploit attention mechanism in the semantic matching problem. To evaluate their impact on the EL performance, we perform two ablation studies. First, we use only one unidirectional LSTM to encode the mention’s local context. Second, we try to abandon the use of attention mechanism in our proposed model. The micro-averaged F_1 of our originally proposed model and the two variants are reported in Table 4.5. As expected, the use of two LSTMs and attention mechanism are more effective in the cases of the long-text datasets such as Reuters, ACE2004, and MSNBC. However, the improvement is marginal on the short-text datasets such as

FIGURE 4.2: F_1 performance of NeuL with different settings of α . A larger α value indicates that the disambiguation will favor prior probability knowledge more than semantic matching scores.

RSS500 and Micro2014. Especially, on challenging short-text dataset KORE50, using two LSTMs and attention mechanism even harms the model performance. One potential reason is that the full-model is over-fitted to the training data. It is also worth mentioning that KORE50 is a special dataset in which the collective EL baselines demonstrate much better performance than local context-based models (see Table 4.3). Furthermore, the local contexts in this dataset are highly ambiguous, thus creating a more serious challenge for local context-based EL methods.

Hyperparameter Study (α). Recall that NeuL requires the setting of hyperparameters α , which balances the contribution between the prior probability and the semantic matching score (see Equation 4.7). A larger value of α indicates that the model will focus more on the prior probability knowledge in the disambiguation. This setting is favorable if the ground-truth entity is popular. To this point, we analyze the F_1 performance with different settings of α value ranging from 0 to 1. As shown in Figure 4.2, for the long, formal text datasets such as Reuters128, ACE2004, and MSNBC, the peak performance is obtained with a larger α values. In contrast, the optimal values are smaller for the short and more challenging datasets such as KORE50. The analysis result indicates that the local contexts are more important for the disambiguation in the KORE50 dataset. However, the current performance of local context-based approaches including NeuL is still low on this dataset.

In the next chapter, we will investigate a collective EL idea that utilizes semantic coherence of entities (in addition to the local semantic relevance) to improve the disambiguation.

4.5 Summary

We have presented a neural network architecture for local context-based entity linking. Our architecture uses recurrent neural network (LSTM, to be specific) and attention mechanism to model the semantic matching between a mention’s local context and an entity candidate. After training on Wikipedia data, the proposed model demonstrates more effective disambiguation performance than other local context-based baselines. However, as context understanding is a challenging problem in NLP, the performance of our proposed model is still limited on difficult test dataset such as KORE50. In the next chapter, we will study another aspect to improve the disambiguation accuracy. Instead of only focusing on the semantic relevance between a mention’s local context and an entity candidate, we will also consider the semantic coherence between the entities. Intuitively, entities within a document are assumed to be semantically related. Thus, the semantic coherence can be used as an additional constraint to improve the EL performance.

Chapter 5

Collective Entity Linking

5.1 Introduction

The previous chapter has studied the local context-based entity linking approach, which resolves the ambiguity of each mention independently. Because an entity can appear in various local contexts, even in the ones that are different from training data, modeling the semantic matching between a mention’s local context and an entity candidate is a non-trivial and challenging task. If a mention’s local context is short, general, or does not contain specific information that reflects the identity of the referred entity, the local context-based EL models will face a serious challenge in disambiguating the associated mention. In this chapter¹, we study a collective EL approach which alleviates this problem by considering the semantic coherence of entities (in a document) to jointly disambiguate the mentions. For example, two people name mentions ‘Pacquiao’ and ‘Bradley’ can be confidently mapped to two boxers Manny Pacquiao and Timothy Bradley, respectively because these two entities are semantically related (can be derived from the KB).

As mentioned in the literature review (see Section 2.2.4 of Chapter 2), most existing collective EL methods are based on the underlying assumption that entities in the same document are pairwise related. These models usually need to consider all possible pairs

¹This chapter is published as Minh C. Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. Pair-Linking for Collective Entity Disambiguation: Two Could Be Better Than All. *The IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 30(1): 59-72, 2019. The demonstration system section is published as Minh C. Phan, Aixin Sun. CoNEREL: Collective Information Extraction in News Articles. *The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR, demo paper)*, 1273-1276, 2018.

FIGURE 5.1: Sparse forms of semantic coherence among entities in two example sentences. Only the edges that connect between two strongly related entities are shown.

of entity candidates (in a document) while performing the disambiguation, thus resulting in an unnecessary high computational complexity. Although the semantic coherence between entities are shown to be useful for EL task, the extent to which these entities are actually connected in reality and the necessity of considering all the pair-wise connections are not yet studied. Consider the two examples in Figure 5.1, in the first example, entities are related in a pairwise form. However, in the second, they are connected in a chain-like form. Both examples illustrate sparse forms of semantic coherence which is commonplace in generic documents. Therefore, these examples show that the fundamental assumption in previous collective EL approaches leaves much to be desired.

For the first time, we study the form of semantic coherence among mentioned entities (*i.e.*, whether it is sparse or dense). We will show that the semantic relationships between mentioned entities in a document are in fact less dense than expected. This could be attributed to several reasons such as noise, data sparsity, and knowledge base incompleteness. As a remedy, we introduce MINTREE, a new tree-based objective for the problem of entity disambiguation. The key intuition behind MINTREE is the concept of *coherence relaxation*, which utilizes the weight of a minimum spanning tree to measure the semantic coherence. With this new objective, we further design Pair-Linking as an approximate solution for the MINTREE optimization problem. The idea of Pair-Linking is simple: instead of considering all the given mentions, Pair-Linking iteratively selects a pair with the highest confidence at each step for disambiguation. Via extensive experiments on 8 benchmark

datasets, we show that our approach is not only more accurate but also surprisingly faster than many state-of-the-art collective linking algorithms.

5.2 Semantic Coherence of Entities

5.2.1 Semantic Coherence Analysis

As is illustrated by the two examples in the introduction, documents (in general) can contain non-salient entities, or the entities that do not have complete connections in the knowledge base. Therefore, the basic assumption used by conventional collective linking approaches (all the entities mentioned should be densely related) leaves much to be desired. In this section, we will study the degree of semantic coherence among the entities. We will calculate the pairwise semantic relatedness between all pairs of entities (in a document) and analyze the denseness of the entity connections. To this end, we first present several pairwise relatedness measures used to estimate the semantic relatedness between two entities. We then introduce a method to measure the degree of coherence. Finally, we report our analysis results on 8 EL datasets (details of each dataset are presented in Section 5.4.2).

Pairwise Relatedness Measure. For the semantic relatedness measures for a pair of entities, denoted by $\psi(e_i, e_j)$, we study the Wikipedia links-based and entity embedding-based measures. The Wikipedia link-based measure (WLM) [99] is widely used in previous EL systems. This measure is based on the incoming Wikipedia hyperlinks. Specifically, two entities will have a higher semantic relatedness score if they share more common Wikipedia pages that cite both these entities. Mathematically, the WLM score is calculated as follows:

$$WLM(e_1, e_2) = 1 - \frac{\log(\max(|U_1|, |U_2|) + 1) - \log(|U_1 \cap U_2| + 1)}{\log(|W| + 1) - \log(\min(|U_1|, |U_2|) + 1)} \quad (5.1)$$

where $|U_1|$ and $|U_2|$ are the set of Wikipedia articles that have hyperlinks to e_1 and e_2 , respectively. W is the set of all Wikipedia articles.

We further exploit a Jaccard-based measure which is also based on the set of incoming Wikipedia hyperlinks. However, different from the original calculation introduced

FIGURE 5.2: Four different forms of connections among entities in a document. In the dense form, all these entities are pairwise related to each other. In the tree- and chain-like forms, there are minimal coherent connections among these entities. On the other hand, in the forest-like form, the entity connections are relatively sparse.

in [176], here we take the logarithm scale of the set size in the calculation:

$$NJS(e_1, e_2) = \frac{\log(|U_1 \cap U_2| + 1)}{\log(|U_1 \cup U_2| + 1)} \quad (5.2)$$

In our experiments, this modified formula with the logarithm yields better performance than the original measure. We name the new measure as Normalized Jaccard Similarity (NJS). Furthermore, we study the entity embedding similarity (EES) which is calculated by the cosine similarity of entity embeddings:

$$EES(e_1, e_2) = \text{cosine}(\text{embedding}(e_1), \text{embedding}(e_2)) \quad (5.3)$$

where the entity embeddings are trained with word embeddings on the Wikipedia data (see Section 4.2 of Chapter 4). Recent works [85, 108, 117] have also verified the effectiveness of the jointly trained entity embeddings in EL task.

Degree of Entity Coherence. We aim to analyze the degree of coherence among entities that appear in a document. Specifically, we are interested to know where these entities are densely (or sparsely) connected. To this end, we propose a new measure to estimate the denseness of the entity connections. Suppose a graph $G(V, E)$ contains all the entities in a document. The edges between each pair of entities are weighted by one of the pairwise relatedness measures introduced earlier: Wikipedia link-based measures (WLM), normalized Jaccard similarity (NJS), or entity embedding similarity (EES). Figure 5.2 illustrates four standard forms of the entity coherence. Consider the denseness of the entity connections, if these entities are pairwise related at the same relatedness level (can be at a

high or a low semantic relatedness level), we will conclude that these entities are densely connected (see Figure 5.2(a)). On the other hand, if there are only a few entity pairs that have much higher pairwise relatedness scores in comparison to the other pairs, we will classify these entity connections, in this case, as sparse (see Figures 5.2(d), 5.2(b), 5.2(c)).

We propose to estimate the denseness of the entity connections through the average degree of an entity filtered graph $G_\theta(V, E_\theta)$, which contains only the connections between the top semantically related entities (*i.e.*, $E_\theta = \{e | e \in E \wedge weight(e) \geq \theta\}$). The threshold θ needs to be carefully set for each entity graph. If this threshold is set to an unnecessarily high value, a small number of edges will be left in the filtered graph, thus leading to a biasedly low denseness score. On the other hand, if the value is unreasonably small, the average degree of the filtered graph will be high. because of this reason, we propose a scheme to set the threshold θ dynamically for each entity graph. Specifically, the θ is chosen as the largest value such that every vertex (or entity) in V is incident to at least one edge in E_θ . As such, each entity graph is pruned to the same ‘standard form’ before calculating its average degree. In other words, the associated filtered edge set E_θ will be a valid *edge cover*² of the graph G . Finally, we calculate the average degree of $G_\theta(V, E_\theta)$ and refer to it as the denseness of entity connections (for the entity set V). Mathematically, the measure is expressed as follows:

$$Denseness(V) = Average_degree(G_\theta) = \frac{2 \times |E_\theta|}{|V|} \quad (5.4)$$

Note that the filtered graph G_θ contains highly related entity connections. The average degree of G_θ reflects the density of the connections. A higher value indicates that the entities in V are densely connected, and a lower value hints that the entity connections are sparse. Illustrated in Figure 5.2(d), in the forest-like form (*i.e.*, every entity is strongly related to only one other entity), the theoretical average degree of G_θ , is 1. On the other hand, if entities in G_θ are connected in tree- or chain-like fashion (see Figures 5.2(b) and 5.2(c)), their corresponding denseness value is about $2 * (n - 1)/n$. Furthermore, the expected value for densely connected case (see Figure 5.2(a)) is close to $(n - 1)$, where n is the number of entities in G_θ .

²https://en.wikipedia.org/wiki/Edge_cover

TABLE 5.1: Average denseness of entity coherence calculated on each EL dataset. Only the documents having more than 3 mentions are considered. The results are reported with three pairwise relatedness measures: Wikipedia link-based measure (WLM), normalized Jaccard similarity (NJS), and entity embedding similarity (EES).

Dataset	$ D $	<i>Coh_deg</i> (theoretical)			<i>Coh_deg</i> (calculated)		
		Forest	Tree	Dense	WLM	NJS	EES
Reuters128	30	1.00	1.64	5.93	3.21	2.13	2.68
ACE2004	25	1.00	1.69	7.20	3.23	2.83	2.75
MSNBC	19	1.00	1.83	14.89	6.35	4.48	7.08
Dbpedia	35	1.00	1.71	6.60	3.08	2.55	2.92
KORE50	9	1.00	1.54	3.44	1.36	1.58	1.36
Micro14	80	1.00	1.53	3.33	1.81	1.72	1.82
AQUAINT	50	1.00	1.84	12.82	5.78	3.39	4.53

Analysis Results. We report the degree of entity coherence for 7 benchmark datasets in Table 5.1. We consider only the documents that have at least 4 mentions. This is because the documents with 3 or a fewer number of mentions have a fixed denseness score by the calculation described above. It is also worth mentioning that for short text datasets like KORE50 or Micro14, the edge pruning step likely produces a tree- or forest-like filtered graph G_θ , thus leading to a bias in the denseness score. However, for completeness, we still report the denseness scores on these datasets.

As shown in Table 5.1, in general, the denseness scores calculated on most of the datasets lie closer to the tree (or chain) form’s expected values rather than the dense form. The same observation is seen in all configurations of the relatedness measures (WLM, NJS, and EES). In details, for long text datasets such as MSNBC and AQUAINT, each entity is highly related to only 3 to 5 other entities (by the NJS measure) although the number of entities in each document in the two datasets is more than 13 (on average). The result reveals that not all the entities appeared in a document are highly related to each other. Therefore, considering all the pairwise connections, as in previous collective EL approaches, is not necessary. Next, we introduce a new graph-based objective that better adapts to the sparse form of the entity coherence. Based on this new objective, we then propose a fast and effective collective linking algorithm.

5.2.2 Tree-based Objective

We introduce **MINTREE**, a new tree-based objective to effectively model the entity disambiguation problem. First, we define a new coherence measure for a set of entities.

MINTREE Coherence Measure. Given an entity set V , we construct a fully-connected entity graph $G(V, E)$. The edges in E are weighted by a specific semantic measure that reflects the distance between entities. The coherence of the entity set V is defined as the weight of the minimum spanning tree (MST) that is derived from G . With this proposed MINTREE coherence measure, we formulate the collective EL problem as an optimization problem as follows

MINTREE Problem Statement. Suppose that the input document consists of N mentions, these mentions are associated with N entity candidate sets $C_1, \dots, C_N$, where C_i represents the entity candidate set for mention m_i . An undirected *entity candidate graph* $\mathcal{G}(V, E)$ is built. The set of vertices V contains all the entity candidates in $C_1, \dots, C_N$. The edges in E connect between two entity candidates: $e_i \in C_i$ and $e_j \in C_j$ ($i \neq j$). The edges are weighted by the semantic distance, which is computed from the local relevance ($\phi(m_i, e_i)$ and $\phi(m_j, e_j)$) and pairwise relatedness ($\psi(e_i, e_j)$) scores:

$$d(e_i, e_j) = 1 - \frac{\phi(m_i, e_i) + \psi(e_i, e_j) + \phi(m_j, e_j)}{3} \quad (5.5)$$

These edge weights not only reflect the pairwise relatedness between the two entity candidates but also encode the local relevance of the two disambiguations. As such, the edge weights defined in this manner can be viewed as the confidence scores for the pairs of linking assignments. Given the entity candidate graph $\mathcal{G}(V, E)$, we aim to find in each candidate set C_i an entity e_i such that the MINTREE coherence score of the selected entity set $\Gamma = \{e_1, \dots, e_N\}$ is minimized.

The MINTREE problem is equivalent to finding the minimum spanning tree on an N -partite graph G such that each of N entity candidate subsets contributes one representative for the tree. Note that the desired output of MINTREE problem is the same as EL, which is the selected entity set Γ , not the MST although the associated MST can be derived easily

FIGURE 5.3: An example entity candidate graph for a document consisting of 4 mentions, each mention has 2 entity candidates. The edge weights represent the distance between the pairs of entities. The weight of the minimum spanning tree derived from the selected entities (represented by the filled points) is used as the MINTREE coherence measure.

from Γ (by using Prim or Kruskal's algorithm). An illustration of a MINTREE output is shown in Figure 5.3. In this example, the illustrated document contains 4 mentions and 4 associated sets of entity candidates. The assigned entity for each mention is represented by a filled point. Furthermore, a sample of a spanning tree is also illustrated with the solid edges. The weight of the associated MST reflects the goodness of the selected entity set.

Quantitative Study of MINTREE. The objective score of an EL model should be correlated to its disambiguation quality. Specifically, given a set of disambiguated entities in a document, MINTREE objective score should be lowered as the number of correct mention-entity assignments increases. We simulate this disambiguation quality by considering $N+1$ disambiguation results in which the number of correct assignments incrementally increases from 0 to N :

- The first disambiguation result has all mentions linking to all wrong entities.
- The second disambiguation result differs from the first result by having the first mention linking to its correct entity.
- The k^{th} ($2 < k \leq N + 1$) result differs with the $(k - 1)^{th}$ result by having the $(k - 1)^{th}$ mention linking to its correct entity.

TABLE 5.2: Spearman’s correlations (rho) between the disambiguation quality (represented by the number of correct linking decisions) and three collective linking objective scores: ALL-Link (AL), SINGLE-Link (SL) and MINTREE (MT). The correlations are averaged across 8 datasets. The results are reported with three relatedness measures: Wikipedia Link-based Measure (WLM), Normalized Jaccard Similarity (NJS) and Entity Embedding Similarity (EES). For each relatedness measure, we also analyze the correlation between the every pairs of objectives.

rho	WLM			NJS			EES		
	AL	SL	MT	AL	SL	MT	AL	SL	MT
Quality	0.924	0.925	-0.927	0.954	0.952	-0.951	0.947	0.945	-0.947
AL	–	0.986	-0.983	–	0.995	-0.994	–	0.989	-0.990
SL		–	-0.985		–	-0.992		–	-0.986
MT			–			–			–

We calculate the MINTREE objective score associated with each of the $N+1$ results. Then we compute the Spearman’s rank correlation coefficient between the calculated objective scores and their associated numbers of correct decisions (made in each of the $N+1$ disambiguation results). In the ideal case, the rank-based correlation coefficient value is equal to -1 because the MINTREE score should be inversely correlated with the disambiguation quality. We compare the results of MINTREE with other two collective EL objectives, namely ALL-Link and SINGLE-Link. Similar to MINTREE, we report the Spearman’s correlation coefficient between each of these two objectives and the disambiguation quality. Furthermore, to show that MINTREE is highly correlated with other objectives, we also calculate the correlation between each pair of objectives.

As mentioned in the literature review (see Section 2.2.4 of Chapter 2), **ALL-Link** considers all the pairwise entity relatedness into its objective function:

$$\Gamma^* = \arg \max_{\Gamma} \left[\sum_{i=1}^N \phi(m_i, e_i) + \sum_{i=1}^N \sum_{j=1, j \neq i}^N \psi(e_i, e_j) \right] \quad (5.6)$$

On the other hand, **SINGLE-Link** considers only the most related connection for each entity, expressed as follows:

$$\Gamma^* = \arg \max_{\Gamma} \sum_{i=1}^N \left[\phi(m_i, e_i) + \max_{j=1}^N \psi(e_i, e_j) \right] \quad (5.7)$$

The analysis result of all the EL objectives is reported in Table 5.2. The result shows that the Spearman’s correlation score between MINTREE and the disambiguation result is as high as the other objectives. The score is about 0.92 for WLM measure and more than 0.94 for NJS and EES measures. Moreover, MINTREE is highly correlated to ALL-Link and SINGLE-Link. The pairwise correlation scores are more than 0.98 across different relatedness measure. We conclude that MINTREE is reasonably as effective as other objectives when being used to model the disambiguation quality. Note that the correlations between the objective score and the disambiguation quality by WLM measure are lower than the ones by NJS and EES measures. As a result, we expect that NJS and EES to be more effective when being used as a relatedness measure for a collective EL algorithm. We will discuss this observation in the experiment (see Section 5.4.3). Next, we propose Pair-Linking as a heuristic solution for the MINTREE optimization problem.

5.3 Pair-Linking Algorithm

5.3.1 Idea and Algorithm

As mentioned earlier, finding the set of linked entities Γ is equivalent to finding the minimum spanning tree in the associated entity candidate graph. Two well-known algorithms for finding MST in a general graph is Kruskal’s [177] and Prim’s [178]. However, the special setting of MINTREE problem makes any direct application of Kruskal’s or Prim’s becoming infeasible. Specifically, the MST in our problems is a subgraph of the entity candidate graph, which involves only one entity node for each candidate set. In this following, we introduce Pair-Linking, a heuristic solution to find the entity set Γ through the process of constructing its associated MST.

Idea. Similar to Kruskal’s algorithm, the main idea of Pair-Linking is iteratively taking the edge that has the smallest weight into consideration. Specifically, Pair-Linking works on the entity candidate graph G (see the MINTREE problem statement, Section 5.2.2). It iteratively takes an edge of the least possible weight that connects two entities e_i^x, e_j^y (in two candidate sets C_i and C_j respectively) to form the tree. The difference compared to the original Kruskal’s algorithm is that after e_i^x is selected, Pair-Linking removes other

FIGURE 5.4: An example of an entity candidate graph with 5 mentions, each mention has 2 entity candidates. The edges between the entity candidates are weighted by the semantic distance. Only the edges with the lowest semantic distances are illustrated. The solid edges are the ones selected by the Pair-Linking process.

vertex $e_i^{\bar{x}}$ from G such that $e_i^{\bar{x}} \neq e_i^x \wedge e_i^{\bar{x}} \in C_i$. Similar removal is performed for e_j^y . This removal steps will ensure that no other entity candidates in the same candidate set will be selected. The algorithm stops when each of the candidate sets has one selected entity.

Intuitively, each step of Pair-linking aims to find and resolve the most confident pair of mentions (represented by the least weighted edge on the entity candidate graph G). Furthermore, once the edge (e_i^x, e_j^y) is selected, it implies that two mentions m_i and m_j are mapped to the entities e_i^x and e_j^y , respectively.

Our Pair-Linking algorithm simulates the Kruskal's but not the Prim's algorithm. The reason is twofold. First, instead of building the MST by merging smaller trees (like Kruskal's algorithm), Prim's grows the tree from a root node. However, Prim's algorithm is less effective than Kruskal's in the EL task. This is because, with Kruskal's algorithm, Pair-linking performs the disambiguation by the order of confidence scores, thus enforcing the subsequent and less confident assignments to be consistent with the previously made and more confident assignments. This strategy is also used in several previous EL works [100, 117, 179] and has been shown to improve the EL performance noticeably. Another advantage of using Kruskal's over Prim's is that if the entity candidate graph is not well-connected (sparse form), the Kruskal-based Pair-Linking process will return multiple coherent trees (see Figure 5.2(d)), which better reflects the sparseness of entity connections in some informal and noisy texts.

Algorithm 1: Pair-Linking algorithm

```

input :  $N$  mentions  $(m_1, \dots, m_N)$ . Mention  $m_i$  has candidate set  $\{C_i \subset W\}$ 
output:  $\Gamma = (e_1, \dots, e_N)$ 
1  $e_i \leftarrow null, \forall e_i \in \Gamma$ 
2 for each pair  $(m_i, m_j) \wedge m_i \neq m_j$  do
3    $Q_{m_i, m_j} \leftarrow \text{top\_pair}(m_i, C_i, m_j, C_j)$ 
4    $Q.\text{add}(Q_{m_i, m_j})$ 
5 end
6 while  $(\exists e_i \in \Gamma, e_i = null)$  do
7    $(m_i, e_i^x, m_j, e_j^y) \leftarrow \text{most\_confident\_pair}(Q)$ 
8    $e_i^x \mapsto e_i$  (Disambiguate  $m_i$  to  $e_i^x$ )
9    $e_j^y \mapsto e_j$  (Disambiguate  $m_j$  to  $e_j^y$ )
10  for  $k := 1 \rightarrow N \wedge e_k = null$  do
11     $Q_{m_k, m_i} \leftarrow \text{top\_pair}(m_k, C_k, m_i, \{e_i\})$ 
12     $Q_{m_k, m_j} \leftarrow \text{top\_pair}(m_k, C_k, m_j, \{e_j\})$ 
13  end
14 end

```

Pair-Linking Example. We will explain the Pair-Linking process by using an example illustrated in Figure 5.4. In this example, the given document consists of 5 mentions, each mention has 2 entity candidates. The edges between the entities are weighted by the semantic distance. Pair-Linking traverses through the list of edges by the order of their weights. In the first step, Pair-Linking considers the edge with the lowest semantic distance (e_1^2, e_2^2) and makes a pair of linking assignments with the highest confidence: $m_1 \mapsto e_1^2$ and $m_2 \mapsto e_2^2$. The edge with the second lowest semantic distance is (e_2^1, e_3^1) . However, since m_2 is already mapped to e_2^2 , any entity other than e_2^2 will be removed from the m_2 's candidate set, including the associated edges. Therefore, the next edge to be considered is (e_4^1, e_5^1) . As a result, m_4 and m_5 are disambiguated to e_4^1 and e_5^1 , respectively. Lastly, (e_3^1, e_4^1) is taken into consideration. As a result, another linking assignment is made *i.e.*, $m_3 \mapsto e_3^1$. Pair-Linking stops at this point because all the 5 mentions are mapped to their associated entities (see Figure 5.4). Note that for the EL task, it is not necessary to construct the minimum spanning tree associated with the linked entity set, although it can be done by continuing picking up additional edges until a fully connected tree is formed.

Pair-Linking Algorithm. We detail Pair-Linking procedure in Algorithm 1. Specifically, Pair-Linking maintains a priority queue Q . Each element Q_{m_i, m_j} tracks the most

confident linking pairs involving mentions m_i and m_j . Q_{m_i, m_j} is initialized by calling function $top_pair(m_i, C_i, m_j, C_j)$, where C_i is the set of entity candidates that mention m_i can link to. The function returns a pair assignment $m_i \mapsto e_i^x$ and $m_j \mapsto e_j^y$, such that $e_i^x \in C_i$, $e_j^y \in C_j$, and the confidence score of the pair assignment is the highest among $C_i \times C_j$ (i.e., the edge distance is the smallest according to Equation 5.5). After initialization, Pair-Linking iteratively retrieves the most confident pair assignment from Q (Line 7) and links the pair of mentions to the associated entities (Lines 8-9). After that, Pair-Linking updates Q , more precisely, Q_{m_k, m_i} and Q_{m_k, m_j} (Lines 10-13). For Q_{m_k, m_i} , the possible pairs of assignments between m_k and m_i are now conditioned by $m_i \mapsto e_i^x$, and the same applies to Q_{m_k, m_j} .

5.3.2 Computational Complexity

The most computationally expensive part of the algorithm is the initialization of Q . This initialization requires the call of $top_pair(m_i, C_i, m_j, C_j)$ function for every pair of mentions. A straightforward implementation of $top_pair(m_i, C_i, m_j, C_j)$ will scan through all possible candidate pairs between the two mentions. This implementation has the time complexity of $\mathcal{O}(k^2)$, where k is the number of candidates per mention. This leads to an overall complexity of $\mathcal{O}(N^2 k^2)$ for the Q 's initialization (Lines 2-5), where N is the number of mentions. However, since only the pair of candidates with the highest confidence score is recorded for a pair of mentions m_i and m_j , Pair-Linking uses *early stop* to avoid scanning through all possible candidate pairs. Specifically, it first sorts each of N candidate set by the local scores ($\mathcal{O}(Nk \log k)$) then traverses the sorted list in descending order. The early stop is applied if the current score is worse than the highest score by a specific margin, i.e., the largest possible value of $\psi(e_i, e_j)$, see Equation 5.5. In the best case, if early stop is applied right after getting the first score, the complexity of $top_pair(m_i, C_i, m_j, C_j)$ is $\mathcal{O}(1)$ and the overall time complexity becomes $\mathcal{O}(N^2 + Nk \log k)$. Indeed, early stop significantly reduces the running time of Pair-Linking in real test cases while still maintaining the correctness of the algorithm.

5.4 Experiments

We compare our proposed Pair-Linking algorithm with other collective EL methods including the optimization-based and graph-based approaches (see Section 2.2.4 of Chapter 2). Since collective EL requires the estimation of local relevance scores, we implement a commonly used feature engineering-based approach for this estimation. Most of our analysis will use this setting because of its easy implementation and training. Furthermore, we also evaluate another setting of Pair-Linking in which our previously proposed semantic matching model, *i.e.*, NeuL (see Chapter 4), is employed. We keep most of the experimental settings about candidate selection, word/entity embeddings, datasets, evaluation metrics to be similar to the settings used in the previous chapter. However, for the ease of presentation, we will describe the main configurations and highlight the difference (if any).

5.4.1 Experimental Settings

Entity Candidate Selection. Same as the procedure described in the previous chapter, the entity candidates are first retrieved based on the surface form similarity between a mention and an entity name. These candidates are then ranked by a pre-trained learning-to-rank model to select the top-20 potential entities. The subsequent disambiguation step only considers the entities in this candidate set.

Local Relevance Score. We adopt the approach proposed in [85] to estimate the local relevance score between a mention (with its local context) and an entity candidate. Specifically, a learning-to-rank model (we use Gradient Boosting Tree in our experiment) is trained to predict the likelihood that a mention m_i will be mapped to an entity candidate e_i . The set of features to be used includes the prior probability $P(e|m)$, several string-similarity features between a mention and an entity name, and the semantic similarity between the mention’s surrounding context and the entity candidate. The raw output obtained from the ranking model will be used as the local relevance score. We also consider another configuration where the local relevance score is obtained from our previously proposed semantic matching model, *i.e.*, NeuL (see Chapter 4).

TABLE 5.3: Statistics of the 8 test datasets used in our evaluation. $|D|$, $|M|$, Avg_m , and $Length$ are the number of documents, number of mentions, the average number of mentions per document, and the average number of words per document, respectively.

Dataset	Type	$ D $	$ M $	Avg_m	Length
Reuters128	news	111	637	5.74	136
ACE2004	news	35	257	7.34	375
MSNBC	news	20	658	32.90	544
DBpedia	news	57	331	5.81	29
RSS500	RSS-feeds	343	518	1.51	30
KORE50	short sentences	50	144	2.88	12
Micro14	tweets	696	1457	2.09	18
AQUAINT	news	50	726	14.52	220

Pairwise Relatedness Measure. We evaluate the performance with three pairwise relatedness measures: Wikipedia link-based measure (WLM), normalized Jaccard similarity (NJS), and entity embedding similarity (EES). These measures are described in section 5.2.1. When each of these measures is used to obtain the pairwise relatedness scores in other collective EL baselines such as ALL-Link or SINGLE-Link, we use a hyper-parameter β to control the balance between the local relevance and the entity coherence terms in their EL objectives. For example, the updated objective function for the ALL-Link based model (originally expressed in Equation 5.6) is re-written as follows:

$$\Gamma^* = \arg \max_{\Gamma} \left[(1 - \beta) \sum_{i=1}^N \phi(m_i, e_i) + \beta \sum_{i=1}^N \sum_{j=1, j \neq i}^N \psi(e_i, e_j) \right] \quad (5.8)$$

5.4.2 Datasets and Baselines

Datasets. In addition to 7 datasets used in the previous chapter, we include an additional long-text dataset in this experiment. The new datasets is **AQUAINT** [99], which contains 50 news documents collected from Xinhua News Service, the New York Times and Associated Press news corpus. The statistics of all datasets is summarized in Table 5.3. Note that, we only consider the mentions whose linked entities appear in Wikipedia. The same setting has been used in most existing EL works [85, 104, 108, 117].

Collective Entity Linking Baselines. We compare Pair-Linking algorithm with the following state-of-the-art collective EL algorithms.

- **Iterative substitution (Itr_Sub (AL))** [103] is an approximate solution for the ALL-Link-based EL model (see Equation 5.6). First, each mention is initially assigned to an entity candidate which has the highest local score. The algorithm iteratively substitutes an assignment $m_i \mapsto e_i^x$ with another mapping $m_i \mapsto e_j^y$ as long as it improves the objective score. We also study the performance of this iterative substitution algorithm with the Sing-Link objective (Equation 5.7) and refer to this configuration as **IterSub (SL)**.
- **Loopy belief propagation (LBP(AL))** [104, 105] solves the inference problem (Equation 2.2) using loopy belief propagation technique [106]. Similar to the iterative substitution algorithm, we also study another setting with the SINGLE-Link objective and refer to it as **LBP(SL)**.
- **Forward-backward (FwBw)** [109] considers only the coherence within a small local region in the disambiguation objective. It uses dynamic programming to derive optimal assignments. The work in [108] shows that this approach is effective and efficient for entity extraction in short texts such as search queries.
- **Densest subgraph (DensSub)** [110] applies a dense subgraph algorithm to prune irrelevant candidates in the mention-candidate graph. Subsequently, a local search method is used to derive the final mention-entity assignments based on an objective function similar to ALL-Link.
- **Personalized PageRank (PageRank)** is used by DoSeR [117]. It performs personalized PageRank algorithm on a mention-candidate graph and utilizes the stabilized scores for the disambiguation. Additionally, DoSeR introduces a 'pseudo' topic node to enforce the coherence among entity candidates and the main topic's context.

We acknowledge a relevant work [105] that also addresses the issue of mentioned entities that are not salient or not well-connected in KB. To perform collective linking, the proposed model considers only top-k most related connections for each entity. However, the model is trained in end-to-end fashion in which the parameters of the local relevance and pairwise relatedness estimations are also learned. In contrast, our work only focuses on the coherence of entities and the collective EL component. We instead use existing

techniques to estimate the local relevance and pairwise relatedness scores. Because of this reason, we do not conduct a comparison with their work in our study.

Evaluation Metrics. Similar to the evaluation protocol used in the previous chapter, we use Gerbil benchmarking framework [170] (Version 1.2.4) to report the EL performance. We consider three measures: precision, recall, and F_1 . Specifically, let Γ_g be the set of groundtruth assignments $m_i \mapsto t_i$, and Γ^* be the linkings produced by an EL system, the three performance metrics are computed as follows:

$$P = \frac{|\Gamma^* \cap \Gamma_g|}{|\Gamma^*|} \quad R = \frac{|\Gamma^* \cap \Gamma_g|}{|\Gamma_g|} \quad F_1 = \frac{2 \times P \times R}{P + R}$$

For all the measures, we report the micro-averaged score (*i.e.*, aggregated across mentions, not documents), and refer the micro-averaged F_1 as the main metric for comparison.

5.4.3 Overall Performance

Collective Entity Linking Performance. We study the performances of different collective EL algorithms in different settings of the pairwise relatedness measure. As shown in Tables 5.4 and 5.5, the pairwise relatedness measure significantly affects the performance of all collective EL algorithms. The normalized Jaccard similarity (NJS) and entity embedding similarity (EES) are shown to be more effective than the Wikipedia Link-based measure (WLM). Furthermore, we try to combine these measures by taking the average of their scores. Among all possible combinations, the combination involved two former measures (*i.e.*, NJS and EES) is the most effective. The combined scheme outperforms other individual schemes.

The approximation algorithm loop belief propagation (LBP) is consistently better than the iterative substitution algorithm in both objective settings ALL-Link (AL) and SINGLE-Link (SL). Furthermore, comparing between ALL-Link and SINGLE-Link, the iterative substitution and LBP algorithms give a comparable performance with different pairwise relatedness measures. On the other hand, the graph based algorithms such as DensSub and PageRank are sensitive to the selection of the relatedness measure. For example, PageRank only yields good results when working with the NJS measure, *i.e.*, 0.825 F_1 score versus

TABLE 5.4: Micro-averaged F_1 of different collective EL algorithms with different pairwise relatedness measures. The best scores are in boldface and the second-best ones are underlined. The numbers of win and runner-up each method performs across different datasets are also illustrated. Significance test is performed on Reuters123, RSS500 and Micro14 datasets (denoted by *) which contain a sufficient number of documents. † indicates the difference against the Pair-Linking’s F_1 score is statistically significant by one-tailed paired t -test (with $p < 0.05$).

(a) WLM as the pairwise relatedness measure.

Collective EL	Reuters128*	ACE2004	MSNBC	Dbpedia	RSS500*	KORE50	Micro14*	AQUAINT	Average
Iter_Sub(AL)	0.795	<u>0.873</u>	0.809	0.821	0.775 [†]	0.506	0.798	0.857	0.779
Iter_Sub(SL)	0.778 [†]	0.849	0.874	0.827	0.758 [†]	0.484	0.794	0.849	0.777
LBP(AL)	<u>0.800</u>	0.867	0.847	0.837	<u>0.776</u>	0.487	0.798	0.855	0.783
LBP(SL)	0.793	0.865	0.850	0.828	0.772	0.496	0.805	0.868	0.785
FwBw	0.788	0.876	0.850	0.844	0.772 [†]	<u>0.526</u>	<u>0.799</u>	0.859	<u>0.789</u>
DensSub	0.788	<u>0.873</u>	0.831	0.823	0.766 [†]	0.523	0.790	0.853	0.781
PageRank	0.767 [†]	0.832	0.791	0.722	0.769 [†]	0.490	0.772 [†]	0.812	0.744
Pair-Linking	0.802	0.871	<u>0.864</u>	<u>0.842</u>	0.785	0.535	0.796	<u>0.862</u>	0.795

(b) NJS as the pairwise relatedness measure.

Collective EL	Reuters128*	ACE2004	MSNBC	Dbpedia	RSS500*	KORE50	Micro14*	AQUAINT	Average
Iter_Sub(AL)	<u>0.840</u>	0.877	0.882	0.810	0.783 [†]	0.689	0.814	0.869	0.821
Iter_Sub(SL)	0.821	0.876	0.878	0.812	<u>0.795</u>	0.671	0.812	0.859	0.815
LBP(AL)	0.839	0.883	0.883	0.825	0.790	0.728	0.812	<u>0.871</u>	<u>0.829</u>
LBP(SL)	0.813	<u>0.886</u>	<u>0.886</u>	<u>0.833</u>	0.788	0.726	0.818	0.868	0.827
FwBw	0.813 [†]	0.883	0.870	0.849	0.792	0.728	<u>0.815</u>	0.869	0.827
DensSub	0.835	0.881	0.855	0.820	0.778 [†]	<u>0.731</u>	0.806 [†]	0.853	0.820
PageRank	0.835	0.897	0.864	<u>0.833</u>	0.783	0.707	0.808	0.875	0.825
Pair-Linking	0.846	0.876	0.892	0.831	0.797	0.764	0.814	0.870	0.836

0.744 and 0.789 when working with WLM and EES measures, respectively. On the other hand, Pair-Linking is more robust with all three measures. Pair-Linking outperforms other baselines on more challenging and short text datasets such as Reuters128, RSS500, and KORE50. Forward-backward algorithm (FwBw) is more effective on short text datasets (RSS and Micro14) than long text datasets (Reuters and AQUAINT). One reason is that in long documents, the useful disambiguation evidence for a mention may not be presented in its local context.

Collective EL Running Time. The theoretical time complexity of different collective EL methods is listed in Table 5.6. FwBw has the lowest time complexity in the worst case because it only considers adjacent mentions. By using dynamic programming [109], FwBw calculates the score of each assignment $m_i \mapsto e_i$ by considering all possible states

TABLE 5.5: Micro-averaged F_1 of different collective linking algorithms with different pairwise relatedness measures. The best scores are in boldface and the second-best ones are underlined. Significance test is performed on Reuters123, RSS500 and Micro14 datasets (denoted by *) which contain a sufficient number of documents. $\dagger$ indicates the difference against the Pair-Linking's F_1 score is statistically significant by one-tailed paired t -test (with $p < 0.05$).

(a) Entity Embedding Similarity (EES) as the pairwise relatedness measure.

Collective EL	Reuters128*	ACE2004	MSNBC	Dbpedia	RSS500*	KORE50	Micro14*	AQUAINT	Average
Iter_Sub(AL)	0.852	0.905	0.875	0.837	0.795	0.556	0.806	0.872	0.812
Iter_Sub(SL)	0.807 $\dagger$	0.871	0.864	0.820	0.801	0.565	0.809	0.860	0.800
LBP(AL)	0.852	0.884	0.897	0.851	0.801	0.581	0.809	0.877	0.819
LBP(SL)	0.846	0.889	0.882	0.836	0.802	0.631	0.817	0.872	0.822
FwBw	0.834 $\dagger$	0.885	0.891	0.850	0.805	0.587	0.809 $\dagger$	0.870	0.816
DensSub	0.825 $\dagger$	0.836	0.840	0.805	0.796 $\dagger$	0.586	0.779 $\dagger$	0.858	0.791
PageRank	0.817 $\dagger$	0.874	0.877	0.827	0.768 $\dagger$	0.503	0.790 $\dagger$	0.860	0.789
Pair-Linking	0.856	0.879	0.894	0.846	0.806	0.637	0.817	0.885	0.827

(b) Average of NJS and EES scores as the pairwise relatedness measure.

Collective EL	Reuters128*	ACE2004	MSNBC	Dbpedia	RSS500*	KORE50	Micro14*	AQUAINT	Average
Iter_Sub(AL)	0.856	0.894	0.879	0.839	0.793 $\dagger$	0.682	0.811	0.876	0.829
Iter_Sub(SL)	0.807 $\dagger$	0.883	0.870	0.835	0.809	0.653	0.808	0.850	0.814
LBP(AL)	0.864	0.861	0.895	0.833	0.777 $\dagger$	0.715	0.822	0.877	0.831
LBP(SL)	0.823 $\dagger$	0.875	0.900	0.843	0.814	0.762	0.824	0.872	0.839
FwBw	0.830 $\dagger$	0.895	0.905	0.832	0.802 $\dagger$	0.749	0.818	0.866	0.837
DensSub	0.851	0.886	0.887	0.835	0.806 $\dagger$	0.738	0.809	0.878	0.836
PageRank	0.837 $\dagger$	0.882	0.888	0.822	0.785 $\dagger$	0.512	0.797 $\dagger$	0.872	0.799
Pair-Linking	0.859	0.883	0.910	0.845	0.823	0.787	0.813	0.879	0.850

TABLE 5.6: Time complexity of different linking algorithms. N is the number of mentions, k is the average number of candidates per mention, and $\mathcal{I}$ is the number of iterations for convergence.

Collective EL	Best case	Worst case
ItrSub	$\mathcal{O}(N^3k)$	$\mathcal{O}(\mathcal{I} \times N^3k)$
LBP	$\mathcal{O}(N^2k^2)$	$\mathcal{O}(\mathcal{I} \times N^2k^2)$
FwBw	$\mathcal{O}(Nk^2)$	$\mathcal{O}(Nk^2)$
DensSub	$\mathcal{O}(N^3k^2 + N^2k^2)$	$\mathcal{O}(N^3k^2 + \mathcal{I} \times N^2k^2)$
PageRank	$\mathcal{O}(N^2k^2)$	$\mathcal{O}(\mathcal{I} \times N^2k^2)$
Pair-Linking	$\mathcal{O}(Nk \log k + N^2)$	$\mathcal{O}(Nk \log k + N^2k^2)$

TABLE 5.7: Average time to disambiguate mentions in one document (in milliseconds) for each dataset. The time for preprocessing steps such as candidate generation is not included.

Collective EL	Reuters128	ACE2004	MSNBC	Dbpedia	RSS500	KORE50	Micro14	AQUAINT
Iter_Sub(AL)	97.515	21.369	3010.214	12.922	0.127	2.235	0.682	293.271
Iter_Sub(SL)	67.772	20.183	3211.341	11.603	0.108	2.284	0.684	107.640
LBP(AL)	40.049	41.911	1584.504	42.673	0.331	11.515	3.667	269.854
LBP(SL)	92.625	43.173	4421.172	44.263	0.289	8.627	3.170	403.140
FwBw	0.940	<u>1.975</u>	8.880	<u>2.034</u>	<u>0.103</u>	<u>1.190</u>	<u>0.367</u>	<u>4.959</u>
DensSub	166.862	221.437	12714.782	168.716	1.196	13.719	7.402	1121.231
PageRank	110.572	77.398	4293.670	132.009	5.436	64.982	15.796	375.239
Pair-Linking	<u>1.721</u>	0.590	<u>28.699</u>	0.491	0.025	0.951	0.117	3.105

in the previous decision (*i.e.*, $m_{i-1} \mapsto e_{i-1}$), which lead to the complexity of $\mathcal{O}(k)$ where k is the number of entity candidates per mention. Therefore, the overall time complexity of FwBw is $\mathcal{O}(Nk^2)$ where N is the number of mentions.

Not surprisingly, the optimization-based (Iter_Sub and LBP) and graph-based (DensSub and PageRank) methods have the highest time complexity. While Iter_Sub and LBP algorithms require multiple iterations to solve the optimization problems, two graph-based algorithms DensSub and PageRank work on a mostly complete entity graph that has N^2k^2 edges. DensSub also requires a graph pre-processing step (*i.e.*, filter noisy entities by shortest path distances) which takes $\mathcal{O}(N^3k^2)$. Furthermore, PageRank iteratively operates on the mention-entity matrix for convergence and it leads to the complexity of $\mathcal{O}(\mathcal{I} \times N^2k^2)$ where $\mathcal{I}$ is the number of iterations required. On the other hand, Pair-Linking only needs to traverse all possible pairs of linking assignment (*i.e.*, $(m_i, e_i), (m_j, e_j)$) at most once, thus leading to the computational complexity of $\mathcal{O}(N^2k^2)$. The worst case of Pair-Linking is the prerequisite of any graph-based algorithm (*e.g.*, DensSub, PageRank) because building the mention-entity graph for N mentions, each has k entity candidate will require Nk vertices and N^2k^2 edges.

It is also worth mentioning that Pair-Linking only considers the pairs of linking assignments that have the highest pairwise confidence scores. Therefore, by using a priority queue to keep track of the top confident pairs, it can avoid traversing through every pair at each step. Our empirical results show that Pair-Linking with this priority queue and “early stop” (see Section 5.3.2) significantly improves the algorithm’s speed. Because only a few pairs of linking assignments dominate the Pair-Linking scores. On the other hand, a large number of candidate pairs are ignored because of the early stop. Table 5.7 shows

that the running time of Pair-Linking (including the time used to construct the priority queue) is smaller than FwBw on 6 out of 8 datasets, thus making Pair-Linking the most effective and efficient collective EL algorithm.

Considering a long text dataset MSNBC, Pair-Linking is nearly 50-100 times faster than the next effective algorithm LBP(AL)(see Table 5.7). On the other hand, FwBw is faster than Pair-Linking but its linking accuracy is worse than Pair-Linking on several datasets (see Tables 5.4 and 5.5). Different from Pair-Linking, FwBw only considers the entity coherence that is limited to the neighboring mentions. Thus, the coherence objective ignores the connections between entities that are far away (e.g., across paragraphs). All in all, the good performance of both FwBw and Pair-Linking hints that a hybrid algorithm that incorporates both FwBw and Pair-Linking’s ideas can further improve the EL performance.

Comparison with other EL Systems. We compare the EL performance of the best setting of Pair-Linking (the one that takes the average score of NJS and EES as the pairwise relatedness measure) with other state-of-the-art EL systems:

- **PBoH** [104] is a probabilistic graphical model and based on *loopy belief propagation* to derive the EL results. The model utilizes Wikipedia statistics about the co-occurrence of words and entities to compute the local relevance and pairwise relatedness scores.
- **DoSeR** [117] carefully designs a collective EL algorithm by applying personalized the PageRank algorithm on a mention-candidate graph in which the edges are weighted by the cosine similarity between the mention’s context and its entity candidate embeddings. DoSeR heavily relies on the proposed collective EL algorithm to produce accurate disambiguation.

We also report the performances of two simple baselines. The first one is a simple probabilistic model which is based on the prior probability $P(e|m)$. This baseline simply disambiguates a mention based on the statistics extracted from Wikipedia hyperlinks. The other baseline is a learning-to-rank gradient boosting tree model. Both the baselines are based on the local relevance score to rank and select the entity candidates. Furthermore,

TABLE 5.8: Micro-averaged precision, recall, and F_1 of Pair-Linking with NJS&EES as the pairwise relatedness measure.

Data set	Precision	Recall	F_1
Reuters128	0.866	0.853	0.859
ACE2004	0.888	0.877	0.883
MSNBC	0.910	0.910	0.910
Dbpedia	0.847	0.842	0.845
RSS500	0.823	0.823	0.823
KORE50	0.787	0.787	0.787
Micro14	0.820	0.806	0.813
AQUAINT	0.882	0.875	0.879

TABLE 5.9: Micro-averaged F_1 of Pair-Linking (using NJS&EES pairwise relatedness measure) and other disambiguation systems. The 'local' annotations indicate that the associated approaches are solely based on the local relevance scores and do not implement any collective EL methods. (PL: Pair-Linking, Avg: Average)

System	Reuters128	ACE2004	MSNBC	Dbpedia	RSS500	KORE50	Micro14	AQUAINT	Avg
PBoH [104]	0.759	0.876	0.897	0.791	0.711	0.646	0.725	0.841	0.781
DoSeR [117]	<u>0.873</u>	<u>0.921</u>	<u>0.912</u>	0.816	0.762	0.550	0.756	0.847	0.805
$P(e m)$ (local)	0.697	0.861	0.781	0.752	0.702	0.354	0.650	0.835	0.704
Xgb (local)	0.776	0.872	0.834	0.818	0.756	0.496	<u>0.789</u>	0.855	0.775
Xgb + PL	0.859	0.883	0.910	0.845	0.823	<u>0.787</u>	0.813	<u>0.879</u>	<u>0.850</u>
NeuL (local)	0.869	0.906	0.904	0.807	0.770	0.551	0.766	0.862	0.804
NeuL + PL	0.916	0.929	0.918	<u>0.828</u>	<u>0.800</u>	0.794	0.776	0.887	0.856

since each mention is disambiguated in isolation with other mentions, these two baselines are viewed as local (non-collective) EL models.

The performance of Pair-Linking is detailed in Table 5.8 and the comparison with other EL systems is shown in Table 5.9. As mentioned in the previous chapter (see Section 4.4.3), our previously proposed local context-based semantic matching model NeuL outperforms the feature engineering-based baseline Xgb. However, when Pair-Linking is employed on top of these two methods, the new configurations yield similar EL performance. In general, Pair-Linking is more effective on short texts, *i.e.*, RSS500, KORE50, Micro14, where the local (non-collective) EL models face a more serious challenge. On KORE50, Pair-Linking improves the disambiguation performance by 0.30 F_1 compared to the local approach $P(e|m)$. Furthermore, Pair-Linking also outperforms PBoH by 0.14 F_1 score on the same dataset.

TABLE 5.10: Micro-averaged F_1 of Pair-Linking (with NJS&EES as the pairwise relatedness measure) with different percentage of non-linkable mentions (as noise). The F_1 score is calculated on the linkable mentions.

Dataset	0%	20%	40%	60%
Reuters128	0.859	0.842	0.850	0.848
ACE2004	0.883	0.879	0.900	0.869
MSNBC	0.910	0.890	0.887	0.893
AQUAINT	0.879	0.873	0.875	0.863

5.4.4 Robustness to Not-in-list Entities

In this work, we do not consider the case where a mention refers to a not-in-link (NIL) entity (*i.e.*, the entity that does not present in the given knowledge base). However, one possible solution to detect such non-linkable mentions is to base on the local relevance scores. Specifically, a mention is assigned to a NIL label if the highest local relevance score among its entity candidates is less than a predefined threshold. Since the performance of this threshold-based approach relies on the local relevance modeling which is not the focus of this work, we skip this NIL detection in our experiment. Instead, we will address a more interesting research question: “*How robust is Pair-Linking if non-linkable mentions are presenting in a document?*”.

For each document, we randomly sample a few mentions and remove the ground-truth entities from their candidate sets. We report the disambiguation performance of Pair-Linking in this new setting. Note that in this experiment, we only consider medium-to-long text documents that contain a sufficient number of mentions. Furthermore, the linking performance is measured only on the linkable mentions. As shown in Table 5.10, the presence of non-linkable mentions does not downgrade the performance of Pair-Linking on other linkable mentions, even in the case that 60% of the input mentions are non-linkable. The robust disambiguation performance of Pair-Linking can be explained as follows. Since the local relevance score of a NIL mention and its entity candidate is usually low, any pair of linking assignments that involves this non-linkable mention will have a low pairwise confidence score. As a result, such a pair will be selected only in the latest steps in the Pair-Linking process (see Section 5.3.1). Therefore, the assignments of these non-linkable mentions are unlikely to affect the assignments of other linkable mentions.

FIGURE 5.5: Main GUI of our demo system. The left panel displays the statistics about the extracted entities. The right panel highlights the mentions where they are referred to.

5.5 Demo System and Pair-Linking Visualization

We have shown that Pair-Linking yields a competitive EL performance while being significantly fast and efficient. The effectiveness of Pair-Linking can be explained by its design to adapt with the sparseness of the entity connections. To further study the entity coherence and the behavior of Pair-Linking, we implement a demonstration system. This demo will focus on simulating Pair-Linking process and visualizing the disambiguation results. We use Yahoo! news articles as the input texts. We also include the comments created by public users collected on the Yahoo! News website.

For each news article and user comment, a sequence of processing steps is applied, which include named entity recognition, candidate selection, local relevance score estimation, and Pair-Linking. The outputs are the mentions and their mapped entities, as shown in Figure 5.5. Although we do not have ground-truth labels to qualitatively evaluate the extraction performance, by browsing through some test cases, we observe that the NER and EL performance on the news article texts is relatively reasonable.

We further implement an interactive visualization of Pair-Linking process. Figure 5.6(a) shows the status of the entity linking graph after the 7th step, where the left panel lists the details of the 7 steps, and the right panel shows the 7 edges in the graph. The edges

FIGURE 5.6: A graphical visualization of Pair-Linking process, after the 7th linking step, and the completion. The left panel details the local relevance and pairwise relatedness scores corresponding to each step. The right panel visualizes the pairs of linking assignments that have been made at each step. The edge width represents the pairwise confidence score (see Equation 5.5). The current step of Pair-Linking is highlighted by the orange edge.

linked in the earlier steps have wider edges for more confidence. On the other hand, Figure 5.6(b) shows the graph after all entities are linked. The complete graph shows three groups of entities, professional basketball players (sub-graph on the left), professional basketball teams (sub-graph on the right), and two cities. These three sub-graphs provide a concise summary of the entity connections between the entities in this news article.

The entity graphs can also include the entities mentioned by readers in comments which do not appear in the news article. Figure 5.7 gives an example, where the entities in comments are with gray borders. The visual animation illustrates that Pair-Linking maintains the entity coherence assumption by growing multiple entity relatedness trees.

FIGURE 5.7: Visualization of Pair-Linking results for a news article and its user comments. The entities that appear in comments are with gray borders, while the ones in the main article text have red borders.

Furthermore, the visualization tool is also useful to study the semantic relatedness between article entities and the ones discussed by users in comments.

5.6 Summary

In this chapter, we have studied collective EL approaches. Traditional collective EL models assume that all entities mentioned in a document are densely related. However, our study reveals that the low degree of coherence is not occasional in general texts (news, tweet, RSS). We propose to use the weight of the minimum spanning tree derived from an entity graph as a new EL objective. This tree-based objective allows us to model the sparseness of entity coherence more effectively. Finally, we have introduced Pair-Linking, an approximate solution for the EL problem with this new objective. Despite being simple, Pair-Linking performs notably fast and achieves comparable accuracy in comparison to other collective EL methods.

At this point, we have studied two EL approaches. The local context-based approach is based on the local relevance between a mention and its entity candidate to disambiguate the mention. On the other hand, the collective EL approach is based on the entity coherence to derive more accurate linking results. Note that the collective EL models still require a method to estimate the local relevance scores. All in all, both these EL approaches aim to address the ambiguity of mentions and their local contexts. In the next chapter, we will address another challenge of EL which is caused by the entity name variance. Specifically, an entity can be referred to using various surface forms, even the ones that do not present

in a knowledge base. This challenge is more serious for EL in specific text domains such as biomedical concepts, product names, or job titles. Furthermore, the lack of sufficient training data and well-constructed knowledge bases in these specific domains also downgrades the performance of existing context-based EL models. We will tackle this challenge by learning meaningful semantic representations for entity names (or surface forms) such that the ones of the same concepts will have similar representations. As such, the entity associated with a mention can be simply retrieved by searching in the name embedding space.

Chapter 6

Entity Name Normalization

6.1 Introduction

Different from names of general-domain entities such as people, locations, and organizations, entity names in specific domains such as biomedical concepts, product names, or job titles have a higher degree of name variance. For example, as shown in Table 6.1, different doctors can use different names to refer to the same biomedical concept (or entity)¹. In social media, people often mention the same products or job titles using different surface forms. The mismatch between these surface forms is problematic for entity linking because it results in difficulties pertaining to have effective candidate selection and ranking

TABLE 6.1: Example of entities and their names (multi-word expressions). These name include both official names in a knowledge base or unofficial names mentioned in texts.

Entity and their names	Type
Exudative retinopathy: coats' disease, abnormal retinal vascular development, unilateral retinal telangiectasis, coats telangiectasis	Disease
Hepatitis B surface antigen: hepatitis b virus surface antigen, hepatitis-b surface antigen, hbs ag, hbsag, hepatitis b surface antigen	Chemical
Samsung Galaxy S III: galaxy s3, s3 lte, s iii, sgs3, samsung galaxy s3, gs3, samsung i9300 galaxy s iii	Mobilephone
Software Tester: tester sip, test consultant, stress tester, kit tester, agile java tester, test engineer, QTP tester	Job title

¹The terms 'concept' and 'entity' can be used interchangeably. However, 'biomedical concept' is more commonly used than 'biomedical entity'. Thus, we will use the former throughout this chapter.

methods. Furthermore, in these specific domains, public annotated data such as the hyperlinks in Wikipedia is not yet available. The lack of such resources prevents the effective training of models used to estimate the local relevance and pairwise relatedness scores. Instead of relying on the local context and the entity coherence, an alternative approach is to focus on the semantic matching between the mentions and entity names, *i.e.*, entity name normalization. This approach is commonly used by EL systems in specific or private domains, especially for biomedical concepts (see Section 2.2.5 of Chapter 2). In fact, most existing works in biomedical concept linking are based on the name matching to achieve state-of-the-art performance [1, 24, 73].

To capture the semantic similarity between two entity names, we aim to learn their semantic representations such that names of the same entity will have similar representations. As such, the entity associated with a query mention can be retrieved by searching in the name embedding space. Although this approach is applicable for a wide range of EL applications in several domains, such as biomedical concepts, project names and job titles, this chapter² will focus on the biomedical domain because of the availability of evaluation datasets. However, the key idea introduced in this chapter is extensible to other domains which have the similar setting.

Idea. As shown in Table 6.1, biomedical concepts appear in the texts under various names. These biomedical names are different from standard words and sentences. These names have both contextual and conceptual meanings. Contextual meaning reflects the contexts where the names appear, and it is specifically granted to each name. The names of a broad and popular concept often have slightly different contextual meanings. On the other hand, conceptual meaning is associated with the definitions/contexts of the names' corresponding concepts. As such, names of the same concepts share common conceptual meanings, although they can own different contextual information.

Our goal is to derive meaningful and robust representations for biomedical names from their surface forms. Unfortunately, this task is not trivial. This is because two names can be strongly related but not necessarily belong to the same concept (*e.g.*, 'complement

²This chapter is accepted as Minh C. Phan, Aixin Sun and Yi Tay. Robust Representation Learning of Biomedical Names. *The 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, accepted in 2019.

FIGURE 6.1: Illustration of three aspects, which are related to three training objectives, for computing representation of entity name (surface form) s . Intuitively, the representation is supposed to be similar to its synonym's as well as its conceptual and contextual representations.

component 5 deficiency' and 'complement component 5'). Furthermore, the names of a concept can be completely different regarding their surface forms (*e.g.*, 'leiner's disease' and 'c5d'). As such, we establish the key desiderata for learning robust representations. First, the output representations need to be both conceptually and contextually meaningful. Second, name representations that belong to the same concepts should be similar to each other, *i.e.*, conceptual grounding.

To this end, our proposed encoding framework incorporates three training objectives, namely context, concept, and synonym-based objectives. We formulate the representation learning process as a synonym prediction task, with context and conceptual losses acting as regularizers, preventing two synonyms from collapsing into semantically meaningless representations. As is illustrated in Figure 6.1, synonym-based objective enforces similar representations between synonymous names, while concept-based objective pulls the name's representations closer to its concept's centroid. On the other hand, context-based objective aims to minimize the difference between the derived representation and its specific contextual representation. More concretely, our approach adopts a recurrent sequence encoding model to extract the semantics of entity names, and to learn the alternative naming of entities. In our experiment with biomedical names, our approach does not need any additional annotations on biomedical text. To be specific, we do not need the biomedical names to be pre-annotated in the text. Instead, we utilize available synonym sets in a metathesaurus vocabulary, such as UMLS (see Section 2.2.1 of Chapter 2), as the only additional resource for training.

6.2 Representation Learning of Entity Names

For ease of presentation, we use three generic terms, u_w , u_s and u_e , to denote pre-trained word, name and concept embeddings, respectively. These embeddings will be used as inputs in our encoding framework. Note that there are multiple ways to pre-train these embeddings. In this section, we will present several skip-gram-based approaches. Note that the name embeddings learned from each of these approaches can also serve as a baseline in our experiments.

6.2.1 Context-based Skip-gram Model

Skip-gram Embeddings with Context. We revisit skip-gram model [157], as one of the most popular context-based embedding approaches. The model computes the representations for both target word w_t , and context word w_c by maximizing the following log-likelihood:

$$\mathcal{L}_W = \sum_{w_t, w_c \in \mathcal{C}_{w_t}} \log p(w_c | w_t) \quad (6.1)$$

The probability of observing w_c in the local context of w_t is defined as follows:

$$p(w_c | w_t) = \frac{\exp(v_{w_c}^\top u_{w_t})}{\sum_{w \in \mathcal{W}} \exp(v_w^\top u_{w_t})}$$

where u_w and v_w are the ‘input’ and ‘output’ vector representations of w . In this work, we refer to the input representations as contextual representations of words, or in short, word embeddings.

The skip-gram model is extensible to names (or phrases) by treating them as special tokens:

$$\mathcal{L}_S = \sum_{w_t, w_c \in \mathcal{C}_{w_t}} \log p(w_c | w_t) + \sum_{s, w_c \in \mathcal{C}_s} \log p(w_c | s) \quad (6.2)$$

where s is a special name token. Training of this model results in word and name embeddings.

Another simple and effective method to compute name embeddings is taking the average of their constituent word embeddings. Since words in a biomedical name are usually

descriptive about its meaning, this simple baseline is expected to produce quality representations. FastText [122] leverages this idea by considering character n-grams instead of words. Therefore, the model can derive representations for names that contain unseen words. The effectiveness of simple compositions such as taking the average or power mean has also been verified in phrase and sentence embeddings [123–125].

Skip-gram Embeddings with Context and Concept. The skip-gram model described by Equation 6.2 uses context words to calculate embeddings for names. Apart from the context words, we also consider the name’s conceptual information in this new baseline. We leverage two sources of conceptual information: the words in a name, and the associated concept of a name. We assume that the names that contain similar words tend to have similar meanings. Furthermore, the names of the same concepts will also share a common meaning.

At this point, we introduce a new token type for concepts. The concept embeddings are trained in a similar way as name embeddings. Specifically, for this baseline, we utilize a pre-annotated corpus where names appearing in the training text are labeled with their associated concepts. We convert the annotated texts into sequences of words, name, and concept tokens to be used as inputs to the skip-gram model. For example, consider a pseudo-sentence that has 4 words and contains a bigram name: $w_l \ w_1 \ w_2 \ w_r$, we map the annotated name $w_1 \ w_2$ to a name token s_i , and its annotated concept is denoted by c_i . We create two sequences of tokens corresponding to this original sentence:

- $w_l, s_i, c_i, w_1, w_2, w_r$
- $w_l, w_1, w_2, s_i, c_i, w_r$

The name and concept tokens are placed on the left and right sides of the annotated name to avoid being biased toward any single side. These token sequences are fed as inputs to train a skip-gram baseline (the training details are presented in Section 6.3.1). Note that the outputs of this baseline are word, name and concept embeddings.

FIGURE 6.2: Our proposed entity name encoding framework. The main encoder (ENE) uses two-level BiLSTM architecture to capture both character and word-level information of an input name. ENE parameters are learned by considering three training objectives. Synonym-based objective $\mathcal{L}_{syn}$ enforces similar representations of two synonymous names (s and s'). Concept-based objective $\mathcal{L}_{def}$, and context-based objectives $\mathcal{L}_{ctx}$ apply similarity constraints on the representations of names (s and s' , which are interchangeable) and their conceptual and contextual representations ($g(e)$ and $g(x)$, respectively). Details about $g(e)$ and $g(x)$ calculations are discussed in Section 6.2.2.

6.2.2 Representation Learning with Context, Concept, and Synonym-based Objectives

Our proposed framework is illustrated in Figure 6.2. The encoder unit is based on bidirectional LSTM (BiLSTM) to aggregate information from both character and word levels. The encoded representations are constrained by three objectives, namely synonym, context, and concept-based objectives. The model utilizes synonym sets in UMLS as training data. We denote all the synonym sets as $\mathcal{U} = \{\mathcal{S}_e\}$, where $\mathcal{S}_e$ includes all names of concept (entity) e , i.e., $\mathcal{S}_e = \{s_i\}$.

Note that our proposed framework is not constrained on any neural network model used for the encoder. BiLSTM is chosen in this work because it has been shown to give robust performance on various short-text encoding tasks.

Entity Name Encoder (ENE). The encoder extracts a fixed-sized representation for a given name (or surface form) s . We use one BiLSTM unit with last-pooling to encode character-level information of each word. The representation is then concatenated with the pre-trained word embedding to form a word-level representation. Another BiLSTM unit with max-pooling is used to aggregate the semantics from the sequence of words'

representations. Finally, the aggregated representation is passed through a linear transformation. Mathematically, the encoding function is expressed as follows:

$$\begin{aligned} h_{w_i} &= [u_{w_i} \oplus \text{last}(\text{BiLSTM}(t_{i,1}, \dots, t_{i,m}))] \\ h_s &= \max(\text{BiLSTM}(h_{w_1}, \dots, h_{w_n})) \\ f(s) &= Wh_s + b \end{aligned}$$

where u_{w_i} represents the pre-trained word embedding of word w_i in name s . $t_{i,j}$ is a trainable character embedding in w_i . $\oplus$ denotes vector concatenation. W and b are parameters of the last transformation. Next, we detail three objectives used to train the encoder.

Synonym-based Similarity. Representations of names that belong to the same concept should be similar to each other. We formulate this objective using the following loss function:

$$\mathcal{L}_{syn} = \sum_{(s,s') \in \mathcal{S}_e \times \mathcal{S}_e} d(f(s), f(s')) \quad (6.3)$$

where $d(\cdot, \cdot)$ is a function that measures the difference between two representations.

As mentioned in the introduction, training the encoder using only this synonym-based objective will lead to biased representations. Specifically, the encoder will be trained to act like a hash function, which performs well on determining whether two names are the synonym of each other. However, it likely loses the semantics of names. As a remedy, we further introduce the concept and context-based objectives to regularize the representations.

Conceptual Meaningfulness. Representations of entity names should be similar to those of their associated concepts. This objective complements the synonym-based objective introduced earlier. The latter not only shifts the synonymous embeddings close to each other, but also pulls them near to its concept's centroid, expressed as:

$$\mathcal{L}_{def} = \sum_{c, s \in \mathcal{S}_e} d(f(s), g(c)) \quad (6.4)$$

where $g(e)$ returns a vector that encodes conceptual information of the corresponding concept c . There are several options for this representation. It can be a mapping to pre-trained concept embeddings learned from a large corpus, *i.e.*, $g(e) = u_e$. Another option is taking composition (*e.g.*, average) of all its name embeddings (see Table 6.1), *i.e.*, $g(e) = \frac{1}{|S_e|} \sum_{s \in S_e} u_s$. Furthermore, when definition of the concept is available, $g(e)$ can be modeled as another encoding function that extracts the conceptual meaning from the definition.

Contextual Meaningfulness. Each name representation should accommodate specific contextual information owned by the name, formulated as:

$$\mathcal{L}_{ctx} = \sum_{s, x \in \mathcal{X}_s} d(f(s), q(x)) \quad (6.5)$$

where $\mathcal{X}_s$ represents all local contexts of name s , and $q(x)$ returns contextual representation of local context x . A straightforward way to model $\mathcal{X}_s$ is using local context words of s . However, this modeling is computationally expensive since the training will need to iterate through all the context words of the name. Alternatively, the contextual information can be modeled using 1-hop approximation of the name's local contexts, which is mapped to the name's contextual representation, *i.e.*, $\mathcal{X}_s = \{s\}$ and $q(x) = q(s) = u_s$. We also consider another approximation where the contextual representation is further approximated by its pre-trained word embeddings, *i.e.*, $q(s) = \frac{1}{|\mathcal{T}(s)|} \sum_{w \in \mathcal{T}(s)} u_w$ where $\mathcal{T}(s)$ represents words in name s . Intuitively, in these two approximations, we assume that the pre-trained name or word embeddings carry local contextual information since they are trained by context-based approaches (see Section 6.2.1).

Combined Loss Function. The final loss function combines all the introduced losses:

$$\mathcal{L}_{ENE} = \mathcal{L}_{syn} + \mathcal{L}_{def} + \mathcal{L}_{ctx} \quad (6.6)$$

For simplicity, we ignore weighting factors that control the contribution of each loss. However, applying and fine-tuning these factors will shift the encoding results more on either semantic similarity or synonym-based similarity direction.

Choices of $g(e)$ and $q(x)$. Several options to calculate the conceptual and contextual representations are discussed earlier. Note that the two representations should be placed in the same distributional space. As such, the implicit relations between them are encoded in, and can be decoded from, their presentations. For efficiency, we model the local contexts $\mathcal{X}_s$ using contextual information encoded in the name itself, *i.e.*, $\mathcal{X}_s = \{s\}$ and $q(x) = q(s)$. To this end, we focus on studying two combinations of $g(e)$ and $q(s)$:

- Option 1: Both $g(e)$ and $q(s)$ directly map to the pre-trained concept and name embeddings, respectively, *i.e.*, $g(e) = u_e$ and $q(s) = u_s$. These embeddings are the outputs of our proposed extension of skip-gram model (see Section 6.2.1). This option requires an annotated corpus.
- Option 2: The contextual presentation $q(s)$ is approximated by the average of pre-trained words embeddings, *i.e.*, $q(s) = \frac{1}{|\mathcal{T}(s)|} \sum_{w \in \mathcal{T}_s} u_w$; and $g(e)$ is the average of all contextual presentations associated with the concept, *i.e.*, $g(e) = \frac{1}{|\mathcal{S}_e|} \sum_{s \in \mathcal{S}_e} q(s)$. These computations only require pre-trained word embeddings, and a dictionary of names and concepts, *e.g.*, UMLS.

Distance Function and Optimization. Distance function d can be Euclidean distance or Kullback-Leibler divergence. Alternatively, the optimization can be modeled as binary classification, motivated by its efficiency and effectiveness [132–134]. Another benefit of using classification is to align the encoded ENE vectors to the pre-trained word, name, and concept embeddings. The pre-trained embeddings are derived by skip-gram with negative sampling [157], which is also formulated as classification. In a similar way, we adopt logistic loss with dot product classifier for all the objectives. For example, the updated loss function for $\mathcal{L}_{syn}$ is rewritten as follows:

$$\ell(f(s')^\top f(s)) + \sum_{\bar{s} \in \mathcal{N}_s} \ell(-f(\bar{s})^\top f(s))$$

where ℓ is the logistic loss function $\ell : x \mapsto \log(1 + e^{-x})$. Negative name $\bar{s}$ is sampled from a mini-batch during optimization, similar to [137]. In a similar way, the loss functions $\mathcal{L}_{def}$ and $\mathcal{L}_{ctx}$ are also updated accordingly.

6.3 Experiments

We aim to evaluate our proposed entity name encoder in the biomedical concept linking task. There are two reasons for selecting the biomedical domain. First, biomedical concept linking is an active area of research, besides the EL for text in general domains which use Wikipedia KB. Second, high-quality evaluation datasets for this biomedical concept linking task are publicly available. At this point, although the current experiment design is for the biomedical domain, our proposed model potentially will demonstrate similar behaviors in other domains which have the similar setting.

In this section, we first detail the implementations of baselines and the proposed ENE model. We then evaluate all these models on several benchmark datasets. We also analyze the robustness and generalizability of these models in other test cases beside EL. Specifically, since our goal is to enforce the similar representations of synonymous names, we will perform a closeness analysis of these synonymous representations. Finally, we study the semantic similarity and relatedness of the name embeddings to verify the robustness of the learned embeddings.

6.3.1 Experimental Settings

Setting and Training of Skip-gram Embeddings. We consider three variants of skip-gram (with negative sampling). SG_W obtains word embeddings by training the very basic skip-gram model (see Equation 6.1). To get the representation for a name, we simply take the average of its associated word embeddings. SG_S is another variant that considers names as special tokens. The model obtains embeddings for word and names concurrently (see Equation 6.2). SG_S training requires input text to be segmented into names and regular words. $SG_{S,C}$ is our proposed extension of skip-gram model. As introduced in Section 6.2.1, this model requires an annotated corpus in which the names are labeled with their associated concepts.

We use PubMed corpus, which consists of 29 million biomedical abstracts, to train SG_W . For SG_S and $SG_{S,C}$, we further utilize the annotations provided in Pubtator [180]. The annotations (names and their associated concepts) come with five categories: disease, chemical, gene, species, and mutation. We use the annotations of two popular classes:

disease and chemical. In preprocessing, text is tokenized and lowercased. Words that appear less than 3 times are ignored. We use spaCy library for this parsing. In total, our vocabulary contains approximately 3 millions words, 700 thousand names, and 85 thousand concepts. We use Gensim library to train all the skip-gram models. The embedding dimension is 200, and the context window size is 6. Negative sampling is used with the number of negatives set to 5.

Setting and Training of Entity Name Encoder (ENE). We use single-layer BiLSTM for both character and word-level encoders. We set the character embedding dimension to 50, and initialize their values randomly. We use 200 dimensions for the outputted name embeddings. The hidden states' dimensions for both character and word-level BiLSTM are 200. We use Adam optimizer with the learning rate of 0.001, and gradient clipping threshold set to 5.0. Training batch size is 64. Dropout with the rate of 0.5 is used to regularize the model. Average performance on validation sets is used as a criteria to stop the model training.

Our proposed model is trained using only the synonym sets in UMLS³, *i.e.*, $\mathcal{U} = \{\mathcal{S}_c\}$. We limit the synonyms to those of disease concepts⁴. We intentionally leave the chemical concepts out for out-domain evaluation. As a result, approximately 16 thousand synonym sets (associated with that number of disease concepts) are collected for training. These synonym sets include 156 thousand disease names in total. In each training batch, one positive and one negative pair are sampled separately for each loss. The pre-trained word (or name/concept) embeddings are taken from the skip-gram models as described earlier. We denote two configurations, associated with Options 1 and 2 (see Section 6.2.2), as **ENE + SG_{s,c}** and **ENE + SG_w**, respectively.

Candidate Selection. We use the same candidate selection process for our proposed model and other baselines. For each mention, we retrieve a list of concept candidates. Since one concept is usually associated with multiple alternative names in the vocabulary, we retrieve the most similar names to the query mention and rank their associated

³We use the 2018AA version released in May, 2018.

⁴We consider the diseases that exist in the CTD's MEDIC disease vocabulary [68].

concepts by the retrieved scores (n-gram BM25 based). We then select the top-20 distinct concepts as the candidate set.

Evaluation Metric. Similar to most works in biomedical concept linking, we report the performance with the accuracy metric. It measures the ratio of mentions that are correctly disambiguated. Note that if a mention is finally not associated with any concept (because the associated candidate set is empty), it is counted as incorrect disambiguation.

6.3.2 Datasets and Baselines

Datasets. We use **NCBI-Disease** [181] and **BC5CDR** [69] datasets in this evaluation. NCBI-Disease contains 6892 disease mentions while BC5CDR dataset contains 5818 disease and 4409 chemical mentions. The texts in both these datasets are sub-sampled from PubMed abstracts. Note that these datasets come with three partitions for training, validation, and testing. We do not use the training data to train our encoder. As described earlier, we only use the UMLS disease synonym sets which are publicly available. Furthermore, it is worth mentioning that the chemical mentions are completely unseen during the model training.

Similar to previous works, we use Ab3P [182] to resolve local abbreviations. Composite mentions (such as ‘pineal and retinal tumors’) are split into separate mentions (‘pineal tumors’ and ‘retinal tumors’) using simple patterns as described in [183]. For each mention, we find the concept (in UMLS) that has the most similar name. The selected concept is then mapped to its associated MeSH or OMIM ID in the CTD dictionary for evaluation. We only consider mentions whose associated concepts exist in the CTD dictionary and report the accuracy aggregated from all mentions in the test set.

Baselines. Apart from three skip-gram baselines, *i.e.*, SG_W , SG_S and $SG_{S,C}$, we also re-implement **PARAGRAM**, a compositional paraphrase model proposed in [137]. The difference is that we use word-level BiLSTM, instead of recursive neural network, to obtain semantic representations of names. Furthermore, L_2 regularizations with the weights of 10^{-3} and 10^{-4} are applied on the BiLSTM’s parameters and the difference between the

trainable and initial word embeddings, respectively. Similar to our ENE encoder, we train this baseline using the same UMLS disease synonym sets.

We also consider several state-of-the-art baselines in biomedical concept linking:

- **Sieve-based** [183] is a sieve-based approach specifically designed for disease concept linking. The authors introduce a set of ten sieves focusing on the lexical similarity between surface forms of the mention and entity name. Some of the sieves are exact matching, abbreviation matching, numbers replacement, and partial matching.
- **TaggerOne** [24] is a semi-Markov-based model that jointly performs the mention recognition and concept linking. Since we do not consider NER, we report only the results in which the ground-truth mentions are given. In this configuration, the disambiguation is performed based on a supervised semantic indexing method [184], which converts both mention and entity candidate's name into two vectors and then uses a weight matrix to score this pair of vectors.
- **Coherence-based NN** [1] is a neural network model that uses bidirectional GRU (BiGRU) to encode the semantic coherence of mentions in a document. Specifically, embedding for a mention is obtained by taking the average of its word embeddings. Next, embedding sequence associated with all mentions in a document is passed into a BiGRU encoder to obtain the new context-aware representation for each mention. Finally, disambiguation is based on the similarity between the entity embedding and both the original and the context-aware representations.

6.3.3 Overall Performance

We report the overall linking accuracy in Table 6.2. Notably, Jaccard baseline demonstrates an effective performance on NCBI-disease and BC5CDR-chemical datasets. This result again verifies that the surface form similarity plays an important role in biomedical concept linking. However, embedding similarity-based baselines such as Word's Mover Distance (**WMD**) [120] and cosine similarity do not show a comparable performance regarding this accuracy metric. Note that this metric only considers the best matched-concept. On the other hand, these embedding-based similarity measures often emphasize

TABLE 6.2: Biomedical context linking accuracy on disease and chemical datasets. The last row group includes the results of supervised models that utilize training annotations in each specific dataset. The ‘exact match’ rule indicates the use of annotation in the training partition to overwrite the original disambiguation result if a query mention is found in the training data. † indicates the results reported in [1].

Models	NCBI (Disease)	BC5CDR (Disease)	BC5CDR (Chemical)
Jaccard similarity (token level)	0.843	0.772	0.935
Cosine similarity (with SG_W embs)	0.800	0.725	0.771
WMD [120] (with SG_W embs)	0.779	0.731	0.919
Cosine similarity (with SG_S embs)	0.815	0.790	0.929
Cosine similarity (with $SG_{S,C}$ embs)	0.838	0.811	0.929
ENE + SG_W	0.854	0.829	0.930
ENE + $SG_{S,C}$	0.857	0.829	0.934
PARAGRAM [137]	0.822	0.813	0.930
Sieve-based [183]	0.847	0.841	-
TaggerOne [24]	<u>0.877</u> †	0.889†	0.941
Coherence-based NN [1]	0.878 †	0.880†	-
ENE + SG_W + ‘exact match’ rule	0.873	<u>0.905</u>	<u>0.954</u>
ENE + $SG_{S,C}$ + ‘exact match’ rule	<u>0.877</u>	0.906	0.958

on the topical similarity. Thus, it does not guarantee that names of the same concept will have the highest similarity score (*i.e.*, conceptual similarity).

PARAGRAM baseline is trained on the synonym-based objective without considering the contextual and conceptual objectives. Although this baseline is trained on the same UMLS disease synonym sets as ENE, the baseline does not generalize well to the real test cases in NCBI-disease datasets. On the other hand, both configurations of ENE (ENE+ SG_W and ENE + $SG_{S,C}$) achieve comparable and the best performances.

Other baselines such as Sieve-based, TaggerOne and Coherence-based NN require EL training data. Furthermore, these models are specifically tuned for each dataset. In contrast, ENE utilizes only the existing synonym sets in UMLS for training. When the dataset-specific annotations are utilized, even the simple exact matching rule can boost the performance of our model to surpass other baselines (see the last two rows in Table 6.2).

Overall, we have shown that our proposed encoder, which considers synonym similarity as well as contextual and conceptual information in training, can achieve state-of-the-art performance in the biomedical concept linking task. Next, we will present our

FIGURE 6.3: *Mean coverage at k* : average ratio of correct synonyms that are found in k -nearest neighbors, which are estimated by cosine similarity of name embeddings. Note that names in these disease and chemical test sets are not seen in the training data.

analysis that details some characteristics of the learned embeddings such as the synonym closeness, semantic similarity and relatedness.

6.3.4 Qualitative Analysis

Closeness Analysis of Synonymous Embeddings. We propose a measure to estimate the closeness between name embeddings of the same concept. For each name, we consider its k most similar names estimated by cosine similarity of their embeddings. We define *coverage at k* as the ratio of correct synonyms that are found in the k -nearest neighbors. We report the average score of all query names, as *mean coverage at k* .

We create two test sets for this experiment: one for disease names and one for chemical names. Given the CTD’s MEDIC disease vocabulary, we randomly select 1000 concepts and all their corresponding names in UMLS. In this experiment, we exclude these 1000 concepts from the synonym sets used to train ENE encoder. Furthermore, to ensure the quality of the selected names, we only consider the ones that appear in the high-quality biomedical phrases collected in [185]. Similarly, we create another test set for chemical names. This chemical set is used to evaluate out-domain performance since our model is trained using only disease synonyms.

As shown in Figure 6.3, ENE outperforms other embedding baselines that do not consider the synonym-based objective. More importantly, the model also generalizes well to out-domain data (chemical names). Furthermore, different from the lexical (Jaccard) and

FIGURE 6.4: t-SNE visualization of 254 name embeddings. These names belong to 10 disease concepts in which 5 of these concepts appear in the training data, while the other 5 concepts (marked with (*)) do not. It can be observed that ENE projects names of the same concept close to each other. The model also retains closeness between names of related concepts, such as ‘parkinson disease’ and ‘paranoid disorders’ (see the red square and green cross signs).

semantic matching (WMD and SG_W) baselines, ENE obtains high scores in both accuracy and ranking-based (MAP) metrics (see Tables, **and** 6.3). It shows that our proposed encoder has encoded both lexical and semantic information of names into their embeddings. Among the skip-gram baselines, the context-based name embedding model (SG_S) is worse than the average word embedding baseline (SG_W). This result again indicates that words in biomedical names are more indicative about their conceptual identities.

The embedding plots in Figure 6.4 further illustrate the effectiveness of our encoder in enhancing the similarity between synonymous representations. By investigating name embeddings of an unseen concept ‘pseudotumor cerebri’, we observe that ENE is robust to the morphology of biomedical names, such as ‘benign hypertension intracranial’ and ‘benign intracran hypt’. The model is also aware of word importance in long names such as ‘intracranial pressure increased (benign)’. Moreover, since ENE is trained using

synonym sets, the encoder is equipped with knowledge about alternative expressions of biomedical terms, *e.g.*, ‘intracranial hypertension’ and ‘intracranial increased pressure’. The knowledge can be used to infer quality representations for new synonyms. However, similar to skip-gram baselines, ENE faces serious challenges if the names are unpopular and contain words that do not reflect their conceptual meanings. For example, for this ‘pseudotumor cerebri’ concept, the name “Nonne’s syndrome”⁵ is distant from its concept cluster (see the olive plus sign locating near the red squares in Figure 6.4).

Synonym Retrieval. We evaluate the embeddings in a synonym retrieval application: given a biomedical mention (or query), retrieving all its synonyms from a controlled vocabulary by ranking. We utilize both NCBI-Disease and BC5CDR datasets in this evaluation. Note that, different from the closeness evaluation presented earlier, a disease name may or may not appear in the synonym sets used to train ENE encoder. On the other hand, chemical queries are completely unseen during model training. Furthermore, this evaluation also differs from the biomedical concept linking experiment. The previous evaluation only reports the performance regarding the best-matched entities. In this synonym retrieval evaluation, we will consider the ranks of all synonyms names and report the mean average precision score (MAP). Specifically, we first retrieve a list of potentially associated concepts for each mention. A concept is retrieved if one of its names is similar to the query (estimated by BM25 score). We collect all names of the top-20 retrieved concepts as a synonym candidate set. Our proposed model and other baselines will rank the synonym candidates regarding their similarities to a given query.

As shown in Table 6.3, SG_W+WMD outperforms Jaccard baseline (in MAP score), mainly because of its ability to capture semantic matching. However, both baselines are non-parametric. In contrast, $ENE+SG_W$ learns additional knowledge about the synonym matching by using synonyms sets in UMLS as training data. Although the model is trained on only disease names, it also generalizes well to chemical names. Furthermore, comparing between the two configurations of ENE, both $ENE+SG_W$ and $ENE+SG_{SC}$ models yield comparable performances. However, $ENE+SG_W$ is simpler since it does not require the pre-trained name and concept embeddings.

⁵Dr. Max Nonne coined the name ‘pseudotumor cerebri’ in 1904.

TABLE 6.3: Mean average precision (MAP) performance on the synonym retrieval task. The best and second best results are in boldface and underlined, respectively.

Models	NCBI (Disease)	BC5CDR (Disease)	BC5CDR (Chemical)
Jaccard	0.424	0.410	0.607
Cosine similarity (with SG_W embs)	0.499	0.494	0.598
WMD [120] (with SG_W embs)	0.532	0.526	0.637
Cosine similarity (with SG_S embs)	0.487	0.472	0.623
Cosine similarity (with $SG_{S,C}$ embs)	0.531	0.510	0.628
ENE + SG_W	<u>0.695</u>	<u>0.718</u>	<u>0.664</u>
ENE + $SG_{S,C}$	0.713	0.734	0.672

Semantic Similarity and Relatedness. We evaluate the correlation between embedding cosine similarity and human judgments, regarding semantic similarity and relatedness. Different from previous evaluations, this experiment aims to evaluate the conceptual similarity and relatedness, as one way to analyze the generalizability of the encoder. We use two biomedical datasets: MayoSRS and UMNSRS. **MayoSRS** [186] consists of multi-word clinical term pairs whose relatedness was determined by nine medical coders and three physicians from the Mayo Clinic. For example, a pair with a high relatedness score is ‘morning stiffness’ (C0457086) and ‘rheumatoid arthritis’ (C0003873). **UMNSRS** [187] contains only single-word name pairs and is spitted into similarity and relatedness partitions. For example, a pair with a high similarity score is ‘weakness’ (C1883552) and ‘paresis’ (C0030552). For these two datasets, the names in each pair come from different concepts, hence they do not appear in the synonym pairs used to train our encoder. Furthermore, the coverage of pre-trained word embeddings in baselines such as SG_W are 100% and 97% for UMNSRS and MayoSRS, respectively.

Table 6.4 shows that ENE models perform especially well on the multi-word relatedness test set (MayoSRS). Conceptual information has been utilized by these models to enrich the name representations. On the other hand, when the training is performed solely on the synonym pairs (only use $\mathcal{L}_{syn}$), the trained model is overfitted to the training task and do not generalize to other test cases. On the other hand, SG_W is still a strong baseline in these benchmarks. Other skip-gram and fastText embeddings [187, 188], which are

TABLE 6.4: Spearman’s rank correlation coefficient between cosine similarity scores of name embeddings and human judgments, reported on semantic similarity and relatedness benchmarks.

Models	UMNSRS (similarity)	UMNSRS (relatedness)	MayoSRS (relatedness)
Cosine similarity (with SG_W embs)	<u>0.645</u>	0.584	0.518
Skip-gram based [187]	0.620	0.580	-
fastText based [188]	0.630	0.575	0.501
cui2vec [189]	0.411	0.334	0.427
Cosine similarity (with SG_S embs)	0.614	0.566	0.516
Cosine similarity (with $SG_{S,C}$ embs)	0.654	<u>0.592</u>	0.557
ENE + SG_W	0.606	0.580	0.626
ENE + $SG_{S,C}$	0.637	0.593	<u>0.602</u>
ENE + $SG_{S,C} (\mathcal{L}_{syn})$	0.496	0.445	0.564
PARAGRAM [137]	0.639	0.565	0.595

trained on a similar corpus, do not achieve better results. The authors in [189] use a SVD-based word2vec model [190] to compute embeddings for biomedical concepts. Although the embeddings are trained on a much larger multimodal medical data, their results are lower than other baselines. Further investigation reveals that many concepts in the test sets do not exist in their pre-trained concept embeddings.

6.4 Summary

By learning to encode names of the same concepts into similar representations, while preserving their conceptual and contextual meanings, our encoder is able to extract meaningful representations for unseen names. The learned embeddings can be used to identify the names of the same concepts directly based on the embedding similarity. The core unit of our encoder (in this work) is BiLSTM. Alternatively, sequence encoding models such as GRU, CNN, transformer, or even encoders with contextualized word embeddings like BERT [33], or ELMo [34] can be used to replace this BiLSTM, however, with additional computation cost. We also discuss different ways of representing the contextual and conceptual information in our framework. In the implementation, we use the simple aggregation of pre-trained embeddings. The experiment results show that this approach is both efficient and effective. We believe that the application of the proposed biomedical

name encoder is not only limited to the EL but can be extended to other tasks in IR such as biomedical literature retrieval [191–193].

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this thesis, our goal is to improve both NER and EL processes. We have presented several new ideas to effectively utilize the local contexts to resolve the ambiguity of the mentions, and make the best use of available resources (structured data, embeddings) to perform the recognition and linking. In the first chapter, Chapter 1 – *Introduction*, we highlight the several motivations of our research problem and its applications in knowledge base population, information retrieval, question answering, and content analysis. We also discuss the main challenges regarding the ambiguity of mentions and contexts, as well as the variance of entity names. Chapter 2 – *Literature Review* provides the readers with background information about existing approaches related to NER and EL. The next four chapters then detail our key contributions, in which one chapter serves NER, and three other chapters associate with EL.

Chaptep 3 – *Collective NER* presents a new idea that utilizes the external relevant contexts to perform NER in a collective manner. Our approach aims to handle the NER challenges caused by the shortness and noisiness of local contexts in user comments. We have shown that most existing NER approaches, which focus on a local region when performing NER, do not yield a desirable performance on these kinds of texts. On the other hand, through extensive experiments with the proposed collective NER framework, we have verified that the relevant contexts in related comments can provide useful information for

NER. We further propose parameterize label propagation (PLP) as a new collective inference method. PLP has demonstrated its desirable behaviors in distinguishing the external contexts which are more reliable then give more propagation weights to their associated mention labels. However, one limitation of our approach is that the proposed NER framework requires the initial NER labels obtained from a trained NER as part of the input. Thus, this strategy can limit the model performance if the initial predictions of the base NER annotator are low-quality. All in all, one key idea of this chapter is to convey the readers that utilizing the external relevant contexts is an effective approach to alleviate the NER challenges in social media texts.

Chapter 4 – *Local Context-based EL* starts to tackle the entity linking problem. This chapter addresses the ambiguity of mentions. As a mention can refer to different entities, disambiguation needs to rely on the mention’s local context to determine its identity. We first formulate the disambiguation as a semantic matching task in which one side is the mention (with its local context) and the other side is an entity candidate (with its description). We have presented two contributions in our proposed semantic matching model. First, we propose a way to jointly train the word and entities embeddings, which are used as pre-trained embeddings in our model. Second, we propose a neural network architecture that relies on LSTMs to encode the mention’s local context and entity description. The attention mechanism is also employed to emphasize the potential matches. At the time of writing this thesis, there are several similar approaches and architectures that have been proposed for semantic matching (in general) and entity linking. However, our work is one of the first which verifies the benefits of using neural network for the EL task. In the model training, we only use the public Wikipedia data. However, the trained model demonstrates competitive and even state-of-the-art performances in different benchmark datasets. We also observe that in some test cases where the mentions are highly ambiguous and their local contexts do not contain adequate matching signals, these local context-based EL approaches (including our proposed model) will fail to disambiguate these mentions correctly. This issue motives us to investigate another approach to improve the EL performance and lead us to study the collective EL approach.

Chapter 5 – *Collective EL* aims to utilize the semantic coherence of entities in a document to improve the linking performance. Our analysis on the entity coherence reveals

that the entities in a document are sparsely related. This is different from previous works which usually assume that these entities are densely connected. We point out that considering the semantic relatedness between all possible pairs of candidate entities often results in an unnecessary computational cost. On the other hand, we introduce a new collective EL objective, which is based on the weight of the minimum spanning tree derived from an entity graph. This new objective alleviates the need of considering all the possible pairwise entity connections in the collective linking process. Alternatively, we propose Pair-Linking as an approximate solution for the EL problem with the tree-based objective. The advantages of Pair-Linking are its simple implementation, effective performance, and robustness in various experimental settings.

Chapter 6 – *Entity Name Normalization* addresses a special setting of the EL in which the challenge arises because of the entity name variance. To tackle this problem, we focus on learning semantic representation for entity names such that the name representations of the same entity are similar to each other. We propose three key objectives used in the representation learning. These objectives not only enforces the similar representations between synonyms but also retains the conceptual and contextual information in this representations. We have shown that the learn embeddings effectively improve the EL performance in the biomedical concept linking task. Our proposed encoding framework can also adapt to the EL problem in other domains such as product names or job titles where a similar setting can be inferred. In practice, there are demands of EL techniques for these specific domains. However, they are not well studied by the research community, partly due to the lack of publicly available benchmark datasets.

In conclusion, named entity recognition and linking has been a fruitful research problem because of its significant impact on a wide range of downstream applications. The problem not only relates to natural language understanding but also requires the retrieval of information in a structured knowledge base. As such, the existing models usually need to consider techniques in both NLP and IR. In this thesis, we have walked the readers through different ideas to improve the NER and EL performance. However, in a bigger picture, as languages and knowledge bases are changing and evolving, more new challenges for NER and EL will arise. To conclude, we will briefly discuss several potential directions for future work.

7.2 Future Work

7.2.1 Incorporate Language and Structured Knowledge Modelings

In order to mimic human capabilities in identifying and interpreting named entities in natural language, NER and EL models should have encoded the commonsense and background knowledge in the natural language and knowledge base when performing the extraction. Chapter 2 has shown that, for NER in general domains, neural network approaches have outperformed the feature engineering approaches. One key advantage of using neural networks is that these models can utilize the semantics encoded in the pre-trained word embeddings. Note that these embeddings are pre-trained on a huge corpus. Thus, they have partly encoded the background knowledge presented in the corpus. Recent works [33, 34] show that pre-training these embeddings with language modeling, or adjusting these embeddings based on their local contexts can improve the NER performance. Regarding the structured information in the knowledge base, it is more challenging to incorporate such information into an NER model. A common way used by existing NER approaches [194, 195] is to include the lexical matching features with the entity names in the KB. This simple approach can produce a significant gain in the NER performance.

For entity linking, we have observed several attempts which jointly learn the word and entity embeddings by utilizing the annotations and linkage structure in Wikipedia KB. This joint training scheme improves the quality of both word and entity embeddings, thus benefiting the EL task [85]. In our previously proposed EL models, we also try to incorporate the information extracted from KB in the linking process. Specifically, our proposed semantic matching model takes the jointly pre-trained word and entity embeddings as input. Furthermore, the semantic relatedness between the entities is estimated based on the co-occurrence of entities in the KB. On the other hand, our proposed name encoder (in Chapter 6) utilizes the synonym sets (for training), which are also extracted from a given KB. All in all, the key idea is to supplement the models with additional information extracted from the KB. Although these current approaches and implementations are still far from what actually happens in human brains, the promising initial results have shed light on one potential direction that can further improve the performance of NER and EL.

7.2.2 Long-tail Mentions and Entities

Similar to words, frequency of entities that are mentioned in natural languages follows the power law distribution. Since most NER and EL approaches are based on the annotated training data to learn the model parameters, their performances on the long-tail, less frequent mentions and entities are relatively worse than the results on the popular ones. Recent analysis [196, 197] also reveals that the embeddings of infrequent words (trained by a popular embedding approach such as word2vec) are much less stable than the frequent-word embeddings. As a result, NER and EL models that use these pre-trained embeddings will have an unstable performance, especially when processing on the unpopular mentions or entities.

In another work [198], the authors flag the lack of appropriate benchmark datasets and evaluation protocols used to analyze EL systems on the long-tail mentions and entities. As one application of NER and EL is to populate new information into an existing KB, the long-tail entities are as important as the popular ones. However, if the extracted information is biased toward the popular entities, very little new knowledge will be extracted (from texts) for the long-tail entities. Because of this reason, these long-tail mentions and entities should receive special attention in both the model design and evaluation of NER and EL systems.

List of Publications

The following is a list of my publications related to my PhD research.

1. **Minh C. Phan** and Aixin Sun. Collective Named Entity Recognition in User Comments via Parameterized Label Propagation. JASIST, doi:10.1002/asi.24282, 2019.
2. **Minh C. Phan**, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. Pair-Linking for Collective Entity Disambiguation: Two Could Be Better Than All. TKDE, 31(7): 1383-1396, 2019.
3. Jialong Han, Aixin Sun, Gao Cong, Wayne Xin Zhao, Zongcheng Ji, and **Minh C. Phan**. Linking Fine-Grained Locations in User Comments. TKDE, 30(1): 59-72, 2018.
4. **Minh C. Phan**, Aixin Sun, and Yi Tay. Robust Representation Learning of Biomedical Names. ACL, 3275-3285, 2019.
5. **Minh C. Phan** and Aixin Sun. CoNEREL: Collective Information Extraction in News Articles. SIGIR, 1273-1276 (demo paper), 2018.
6. Yi Tay, **Minh C. Phan**, Luu Anh Tuan, and Siu Cheung Hui. SkipFlow: Incorporating Neural Coherence Features for End-to-End Automatic Text Scoring. AAAI, 5948-5955, 2018.
7. **Minh C. Phan**, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. NeuPL: Attention-based Semantic Matching and Pair-Linking for Entity Disambiguation. CIKM, 1667-1676, 2017.
8. **Minh C. Phan**, Aixin Sun, and Yi Tay. Cross-Device User Linking: URL, Session, Visiting Time, and Device-log Embedding. SIGIR, 933-936 (short paper), 2017.
9. Yi Tay, **Minh C. Phan**, Luu Anh Tuan, and Siu Cheung Hui. Learning to Rank Question Answer Pairs with Holographic Dual LSTM Architecture. SIGIR, 695-704, 2017.

References

- [1] Dustin Wright, Yannis Katsis, Raghav Mehta, and Chun-Nan Hsu. Normco: Deep disease normalization for biomedical knowledge base construction. *1st Conference on Automated Knowledge Base Construction*, 2019.
- [2] Heng Ji and Ralph Grishman. Knowledge base population: Successful approaches and challenges. In *ACL*, pages 1148–1158, 2011.
- [3] Jiafeng Guo, Gu Xu, Xueqi Cheng, and Hang Li. Named entity recognition in query. In *SIGIR*, pages 267–274, 2009.
- [4] Jeffrey Pound, Peter Mika, and Hugo Zaragoza. Ad-hoc object retrieval in the web of data. In *WWW*, pages 771–780, 2010.
- [5] Chenyan Xiong, Jamie Callan, and Tie-Yan Liu. Word-entity duet representations for document ranking. In *SIGIR*, pages 763–772, 2017.
- [6] Grace E. Lee and Aixin Sun. Seed-driven document ranking for systematic reviews in evidence-based medicine. In *SIGIR*, pages 455–464, 2018.
- [7] Diego Mollá, Menno van Zaanen, and Daniel Smith. Named entity recognition for question answering. In *Australasian Language Technology Workshop*, page 51, 2006.
- [8] Mahboob Alam Khalid, Valentin Jijkoun, and Maarten De Rijke. The impact of named entity normalization on information retrieval for question answering. In *ECIR*, pages 705–710, 2008.
- [9] Namrata Godbole, Manja Srinivasaiah, and Steven Skiena. Large-scale sentiment analysis for news and blogs. *ICWSM*, 7(21):219–222, 2007.

- [10] Xiaowen Ding, Bing Liu, and Lei Zhang. Entity discovery and assignment for opinion mining applications. In *KDD*, pages 1125–1134, 2009.
- [11] Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. Discovering relations among named entities from large corpora. In *ACL*, page 415, 2004.
- [12] Chih-Hsuan Wei, Yifan Peng, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Jiao Li, Thomas C Wieggers, and Zhiyong Lu. Assessing the state of the art in biomedical relation extraction: overview of the biocreative v chemical-disease relation (cdr) task. *Database*, 2016, 2016.
- [13] Yijia Zhang, Hongfei Lin, Zhihao Yang, Jian Wang, Shaowu Zhang, Yuanyuan Sun, and Liang Yang. A hybrid model based on neural networks for biomedical relation extraction. *Journal of biomedical informatics*, 81:83–92, 2018.
- [14] Sunil Kumar Sahu and Ashish Anand. Drug-drug interaction extraction from biomedical texts using long short-term memory network. *Journal of biomedical informatics*, 86:15–24, 2018.
- [15] Yijia Zhang, Wei Zheng, Hongfei Lin, Jian Wang, Zhihao Yang, and Michel Dumontier. Drug-drug interaction extraction via hierarchical rnns on sequence and shortest dependency paths. *Bioinformatics*, 34(5):828–835, 2017.
- [16] Sherzod Hakimov, Soufian Jebbara, and Philipp Cimiano. Deep learning approaches for question answering on knowledge bases: an evaluation of architectural design choices. *CoRR*, abs/1812.02536, 2018.
- [17] Ahmad Aghaebrahimian and Filip Jurčiček. Open-domain factoid question answering via knowledge graph search. In *Proceedings of the Workshop on Human-Computer Question Answering*, pages 22–28, 2016.
- [18] Denis Lukovnikov, Asja Fischer, Jens Lehmann, and Sören Auer. Neural network-based question answering over knowledge graphs on word and character level. In *WWW*, pages 1211–1220, 2017.

- [19] Minh C. Phan and Aixin Sun. Collective named entity recognition in user comments via parameterized label propagation. *JASIST*. doi: 10.1002/asi.24282.
- [20] Minh C Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. Neupl: Attention-based semantic matching and pair-linking for entity disambiguation. In *CIKM*, pages 1667–1676, 2017.
- [21] Minh C Phan, Aixin Sun, Yi Tay, Jialong Han, and Chenliang Li. Pair-linking for collective entity disambiguation: Two could be better than all. *TKDE*, 31(7):1383–1396, 2018.
- [22] Minh C Phan and Aixin Sun. Conerel: Collective information extraction in news articles. In *SIGIR*, pages 1273–1276, 2018.
- [23] Minh C Phan, Aixin Sun, and Yi Tay. Robust representation learning of biomedical names. In *ACL*, pages 3275–3285, 2019.
- [24] Robert Leaman and Zhiyong Lu. Taggerone: joint named entity recognition and normalization with semi-markov models. *Bioinformatics*, 32(18):2839–2846, 2016.
- [25] Zongcheng Ji, Aixin Sun, Gao Cong, and Jialong Han. Joint recognition and linking of fine-grained locations from tweets. In *WWW*, pages 1271–1281, 2016.
- [26] Avirup Sil and Alexander Yates. Re-ranking for joint named-entity recognition and linking. In *CIKM*, pages 2369–2374, 2013.
- [27] Gang Luo, Xiaojiang Huang, Chin-Yew Lin, and Zaiqing Nie. Joint entity recognition and disambiguation. In *EMNLP*, pages 879–888, 2015.
- [28] Xiao Ling and Daniel S Weld. Fine-grained entity recognition. In *AAAI*, 2012.
- [29] Ndapandula Nakashole, Tomasz Tylenda, and Gerhard Weikum. Fine-grained semantic typing of emerging entities. In *ACL*, volume 1, pages 1488–1497, 2013.
- [30] Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. Hierarchical losses and new resources for fine-grained entity typing and linking. In *ACL*, pages 97–109, 2018.

- [31] Ralph Grishman and Beth Sundheim. Message understanding conference-6: A brief history. In *COLING*, volume 1, 1996.
- [32] Erik F Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *CoRR*, *abs/0306050*, 2003.
- [33] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *CoRR*, *abs/1810.04805*, 2018.
- [34] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *CoRR*, *abs/1802.05365*, 2018.
- [35] Gustavo Aguilar, Adrian Pastor López Monroy, Fabio González, and Tamar Solorio. Modeling noisiness to recognize named entities using multitask neural networks on social media. In *NAACL-HLT*, pages 1401–1412, 2018.
- [36] Maksim Tkachenko and Andrey Simanovsky. Named entity recognition: Exploring features. 2012.
- [37] Daniel M. Bikel, Scott Miller, Richard Schwartz, and Ralph Weischedel. Nymble: a high-performance learning name-finder. In *Fifth Conference on Applied Natural Language Processing*, pages 194–201, 1997.
- [38] Daniel M Bikel, Richard Schwartz, and Ralph M Weischedel. An algorithm that learns what’s in a name. *Machine learning*, 34(1-3):211–231, 1999.
- [39] GuoDong Zhou and Jian Su. Named entity recognition using an hmm-based chunk tagger. In *ACL*, pages 473–480, 2002.
- [40] Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *HLT-NAACL*, pages 188–191, 2003.

- [41] Vijay Krishnan and Christopher D Manning. An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *COLING-ACL*, pages 1121–1128, 2006.
- [42] Burr Settles. Biomedical named entity recognition using conditional random fields and rich feature sets. In *Joint Workshop on Natural Language Processing in Biomedicine and its Applications*, 2004.
- [43] Alan Ritter, Sam Clark, Oren Etzioni, et al. Named entity recognition in tweets: an experimental study. In *EMNLP*, pages 1524–1534, 2011.
- [44] Xiaohua Liu, Shaodian Zhang, Furu Wei, and Ming Zhou. Recognizing named entities in tweets. In *ACL-HLT*, pages 359–367, 2011.
- [45] Tim Rocktäschel, Michael Weidlich, and Ulf Leser. Chemspot: a hybrid system for chemical named entity recognition. *Bioinformatics*, 28(12):1633–1640, 2012.
- [46] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In *NAACL-HLT*, pages 260–270, 2016.
- [47] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL*, volume 1, pages 1064–1074, 2016.
- [48] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *COLING*, pages 1638–1649, 2018.
- [49] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [50] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078, 2014.
- [51] Kevin Clark, Minh-Thang Luong, Christopher D Manning, and Quoc V Le. Semi-supervised sequence modeling with cross-view training. *CoRR*, abs/1809.08370, 2018.

- [52] Zhilin Yang, Ruslan Salakhutdinov, and William Cohen. Multi-task cross-lingual sequence tagging from scratch. *CoRR, abs/1603.06270*, 2016.
- [53] Xiaojun Wan, Liang Zong, Xiaojiang Huang, Tengfei Ma, Houping Jia, Yuqian Wu, and Jianguo Xiao. Named entity recognition in chinese news comments on the web. In *IJCNLP*, pages 856–864, 2011.
- [54] Xiang Ren, Ahmed El-Kishky, Chi Wang, Fangbo Tao, Clare R. Voss, and Jiawei Han. Clustype: Effective entity recognition and typing by relation phrase-based clustering. In *KDD*, pages 995–1004, 2015.
- [55] Razvan Bunescu and Marius Paşca. Using encyclopedic knowledge for named entity disambiguation. In *EACL*, 2006.
- [56] Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and global algorithms for disambiguation to wikipedia. In *ACL-HLT*, pages 1375–1384, 2011.
- [57] Yangjie Yao and Aixin Sun. Product name recognition and normalization in internet forums.
- [58] Qiaoling Liu, Faizan Javed, and Matt Mcnair. Companydepot: Employer name normalization in the online recruitment industry. In *KDD*, pages 521–530, 2016.
- [59] Feroosh Jacob, Faizan Javed, Meng Zhao, and Matt Mcnair. scool: A system for academic institution name normalization. In *CTS*, pages 86–93, 2014.
- [60] Angela Fahrni and Michael Strube. Jointly disambiguating and clustering concepts and entities with markov logic. *COLING*, pages 815–832, 2012.
- [61] Heng Ji, Joel Nothman, Ben Hachey, et al. Overview of tac-kbp2014 entity discovery and linking tasks.
- [62] Roberto Navigli. Word sense disambiguation: A survey. *ACM computing surveys*, 41(2):10, 2009.
- [63] Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R Curran. Evaluating entity linking with wikipedia. *Artificial intelligence*, 194:130–150, 2013.

- [64] Andrea Moro, Alessandro Raganato, and Roberto Navigli. Entity linking meets word sense disambiguation: a unified approach. *TACL*, 2:231–244, 2014.
- [65] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. 2007.
- [66] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, pages 1247–1250, 2008.
- [67] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *WWW*, pages 697–706, 2007.
- [68] Allan Peter Davis, Cynthia J Grondin, Kelley Lennon-Hopkins, Cynthia Saraceni-Richards, Daniela Sciaky, Benjamin L King, Thomas C Wiegers, and Carolyn J Mattingly. The comparative toxicogenomics database’s 10th year anniversary: update 2015. *Nucleic acids research*, 43(D1):D914–D920, 2014.
- [69] Jiao Li, Yueping Sun, Robin J Johnson, Daniela Sciaky, Chih-Hsuan Wei, Robert Leaman, Allan Peter Davis, Carolyn J Mattingly, Thomas C Wiegers, and Zhiyong Lu. Biocreative v cdr task corpus: a resource for chemical disease relation extraction. *Database*, 2016, 2016.
- [70] Robert Leaman, Rezarta Islamaj Doğan, and Zhiyong Lu. Dnorm: disease name normalization with pairwise learning to rank. *Bioinformatics*, 29(22):2909–2917, 2013.
- [71] Luca Soldaini and Nazli Goharian. Quickumls: a fast, unsupervised approach for medical concept extraction.
- [72] Alan R Aronson. Metamap: Mapping text to the umls metathesaurus. 2006.
- [73] Haodi Li, Qingcai Chen, Buzhou Tang, Xiaolong Wang, Hua Xu, Baohua Wang, and Dong Huang. Cnn-based ranking for biomedical entity normalization. *BMC bioinformatics*, 18(11):385, 2017.

- [74] Wei Shen, Jianyong Wang, and Jiawei Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *TKDE*, 27(2):443–460, 2014.
- [75] Wei Zhang, Yan-Chuan Sim, Jian Su, and Chew-Lim Tan. Entity linking with effective acronym expansion, instance selection and topic modeling. In *IJCAI*, 2011.
- [76] John Lehmann, Sean Monahan, Luke Nezda, Arnold Jung, and Ying Shi. Lcc approaches to knowledge base population at tac 2010.
- [77] Xianpei Han and Jun Zhao. Nlpr_kbp in tac 2009 kbp track: A two-stage method to entity linking. Citeseer.
- [78] Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. Entity disambiguation for knowledge base population. In *COLING*, pages 277–285. ACL, 2010.
- [79] Sean Monahan, John Lehmann, Timothy Nyberg, Jesse Plymale, and Arnold Jung. Cross-lingual cross-document coreference with entity linking.
- [80] Swapna Gottipati and Jing Jiang. Linking entities to a knowledge base with query expansion. In *EMNLP*, pages 804–813, 2011.
- [81] Silviu Cucerzan. Large-scale named entity disambiguation based on wikipedia data. In *EMNLP-CoNLL*, pages 708–716, 2007.
- [82] Zheng Chen, Suzanne Tamang, Adam Lee, Xiang Li, Wen-Pin Lin, Matthew Snover, Javier Artiles, Marissa Passantino, and Heng Ji. Cuny-blender tac-kbp2010 entity linking and slot filling system description. 2010.
- [83] Wei Zhang, Jian Su, Chew Lim Tan, and Wen Ting Wang. Entity linking leveraging: automatically generated annotation. In *COLING*, pages 1290–1298, 2010.
- [84] Zheng Chen and Heng Ji. Collaborative ranking: A case study on entity linking. In *EMNLP*, pages 771–781, 2011.
- [85] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. Joint learning of the embedding of words and entities for named entity disambiguation. In *CoNLL*, 2016.

- [86] Zhicheng Zheng, Fangtao Li, Minlie Huang, and Xiaoyan Zhu. Learning to link entities with knowledge base. In *NAACL-HLT*, pages 483–491, 2010.
- [87] Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. Liege:: link entities in web lists with knowledge base. In *KDD*, pages 1424–1432, 2012.
- [88] Stefan Zwicklbauer, Christin Seifert, and Michael Granitzer. Robust and collective entity disambiguation through semantic embeddings. In *SIGIR*, pages 425–434, 2016.
- [89] Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. Learning entity representation for entity disambiguation. In *ACL*, volume 2, pages 30–34, 2013.
- [90] Matthew Francis-Landau, Greg Durrett, and Dan Klein. Capturing semantic similarity for entity linking with convolutional neural networks. In *NAACL-HLT*, pages 1256–1261, 2016.
- [91] Feng Nie, Yunbo Cao, Jinpeng Wang, Chin-Yew Lin, and Rong Pan. Mention and entity description co-attention for entity disambiguation. In *AAAI*, 2018.
- [92] Shengze Hu, Zhen Tan, Weixin Zeng, Bin Ge, and Weidong Xiao. Entity linking via symmetrical attention-based neural network and entity structural features. *Symmetry*, 11(4):453, 2019.
- [93] Zheng Fang, Yanan Cao, Dongjie Zhang, Qian Li, Zhenyu Zhang, and Yanbing Liu. Joint entity linking with deep reinforcement learning. *CoRR*, abs/1902.00330, 2019.
- [94] Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. End-to-end neural entity linking. *CoRR*, abs/1808.07699, 2018.
- [95] Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. Learning entity representation for entity disambiguation. In *ACL: Short Papers*, pages 30–34, 2013.
- [96] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *ICML*, pages 1188–1196, 2014.

- [97] Silviu Cucerzan. Large-scale named entity disambiguation based on wikipedia data. In *EMNLP-CoNLL*, pages 708–716, 2007.
- [98] Wei Shen, Jianyong Wang, and Jiawei Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *TKDE*, 27(2):443–460, 2015.
- [99] David N. Milne and Ian H. Witten. Learning to link with wikipedia. In *CIKM*, pages 509–518, 2008.
- [100] Lev-Arie Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and global algorithms for disambiguation to wikipedia. In *ACL*, pages 1375–1384, 2011.
- [101] Xianpei Han and Jun Zhao. Named entity disambiguation by leveraging wikipedia semantic knowledge. In *CIKM*, pages 215–224, 2009.
- [102] Rudi Cilibrasi and Paul M. B. Vitányi. The google similarity distance. *TKDE*, 19(3): 370–383, 2007.
- [103] Wei Shen, Jianyong Wang, Ping Luo, and Min Wang. LIEGE: : link entities in web lists with knowledge base. In *KDD*, pages 1424–1432, 2012.
- [104] Octavian-Eugen Ganea, Marina Ganea, Aurélien Lucchi, Carsten Eickhoff, and Thomas Hofmann. Probabilistic bag-of-hyperlinks model for entity linking. In *WWW*, pages 927–938, 2016.
- [105] Amir Globerson, Nevena Lazic, Soumen Chakrabarti, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. Collective entity resolution with multi-focal attention. In *ACL*, 2016.
- [106] Kevin P. Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *UAI*, pages 467–475, 1999.
- [107] Paolo Ferragina and Ugo Scaiella. TAGME: on-the-fly annotation of short text fragments (by wikipedia entities). In *CIKM*, pages 1625–1628, 2010.
- [108] Aasish Pappu, Roi Blanco, Yashar Mehdad, Amanda Stent, and Kapil Thadani. Lightweight multilingual entity extraction and linking. In *WSDM*, pages 365–374, 2017.

- [109] Steve Austin, Richard Schwartz, and Paul Placeway. The forward-backward search algorithm. In *ICASSP*, pages 697–700, 1991.
- [110] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. Robust disambiguation of named entities in text. In *EMNLP*, pages 782–792, 2011.
- [111] Xianpei Han, Le Sun, and Jun Zhao. Collective entity linking in web text: a graph-based method. In *SIGIR*, pages 765–774, 2011.
- [112] Ben Hachey, Will Radford, and James R. Curran. Graph-based named entity linking with wikipedia. In *WISE*, pages 213–226, 2011.
- [113] Zhaochen Guo and Denilson Barbosa. Robust entity linking via random walks. In *CIKM*, pages 499–508, 2014.
- [114] Francesco Piccinno and Paolo Ferragina. From tagme to WAT: a new entity annotator. In *ACM Workshop on Entity Recognition & Disambiguation*, pages 55–62, 2014.
- [115] Ayman Alhelbawy and Robert J. Gaizauskas. Graph ranking for collective named entity disambiguation. In *ACL: Short Papers*, pages 75–80, 2014.
- [116] Andrea Moro, Alessandro Raganato, and Roberto Navigli. Entity linking meets word sense disambiguation: a unified approach. *TACL*, 2:231–244, 2014.
- [117] Stefan Zwicklbauer, Christin Seifert, and Michael Granitzer. Robust and collective entity disambiguation through semantic embeddings. In *SIGIR*, pages 425–434, 2016.
- [118] Siddhartha Jonnalagadda and Philip Topham. Nemo: Extraction and normalization of organization names from pubmed affiliation strings. *Journal of Biomedical Discovery and Collaboration*, 5:50, 2010.
- [119] Paul Neculoiu, Maarten Versteegh, and Mihai Rotaru. Learning text similarity with siamese recurrent networks. In *RepL4NLP*, pages 148–157, 2016.

- [120] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In *ICML*, pages 957–966, 2015.
- [121] Miaofeng Liu, Jialong Han, Haisong Zhang, and Yan Song. Domain adaptation for disease phrase matching with adversarial networks. In *BioNLP 2018 workshop*, pages 137–141, 2018.
- [122] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *TACL*, 5:135–146, 2017.
- [123] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards universal paraphrastic sentence embeddings. In *ICLR*, 2016.
- [124] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. In *ICLR*, 2017.
- [125] Andreas Rücklé, Steffen Eger, Maxime Peyrard, and Iryna Gurevych. Concatenated p -mean word embeddings as universal cross-lingual sentence representations. *CoRR*, abs/1803.01400, 2018.
- [126] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. Deep unordered composition rivals syntactic methods for text classification. In *ACL-IJCNLP*, volume 1, pages 1681–1691, 2015.
- [127] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. In *ACL*, volume 1, pages 655–665, 2014.
- [128] Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y Ng. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *NIPS*, pages 801–809, 2011.
- [129] Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. In *ACL-IJCNLP*, volume 1, pages 1556–1566, 2015.

- [130] Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In *NIPS*, pages 3294–3302, 2015.
- [131] Felix Hill, Kyunghyun Cho, and Anna Korhonen. Learning distributed representations of sentences from unlabelled data. In *NAACL HLT*, pages 1367–1377, 2016.
- [132] Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. *CoRR*, *abs/1803.02893*, 2018.
- [133] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*, pages 670–680, 2017.
- [134] John Wieting and Kevin Gimpel. Revisiting recurrent networks for paraphrastic sentence embeddings. In *ACL*, pages 2078–2088, 2017.
- [135] Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal. Learning general purpose distributed sentence representations via large scale multi-task learning. In *ICLR*, 2018.
- [136] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *CoRR*, *abs/1803.11175*, 2018.
- [137] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. From paraphrase database to compositional paraphrase model and back. *TACL*, 3:345–358, 2015.
- [138] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *HLT-NAACL*, pages 142–147, 2003.
- [139] Masayuki Karasuyama and Hiroshi Mamitsuka. Manifold-based similarity adaptation for label propagation. In *NIPS*, pages 1547–1555, 2013.

- [140] Jinxiu Chen, Donghong Ji, Chew Lim Tan, and Zhengyu Niu. Relation extraction using label propagation based semi-supervised learning. In *COLING-ACL*, pages 129–136, 2006.
- [141] Peter F. Brown, Vincent J. Della Pietra, Peter V. de Souza, Jennifer C. Lai, and Robert L. Mercer. Class-based n-gram models of natural language. *Computational Linguistics*, pages 467–479, 1992.
- [142] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543, 2014.
- [143] Jenny Rose Finkel, Trond Grenager, and Christopher D. Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*, pages 363–370, 2005.
- [144] Chenliang Li, Jianshu Weng, Qi He, Yuxia Yao, Anwitaman Datta, Aixin Sun, and Bu-Sung Lee. Twiner: named entity recognition in targeted twitter stream. In *ACM SIGIR*, pages 721–730, 2012.
- [145] Shumeet Baluja, Rohan Seth, D Sivakumar, Yushi Jing, Jay Yagnik, Shankar Kumar, Deepak Ravichandran, and Mohamed Aly. Video suggestion and discovery for youtube: taking random walks through the view graph. In *WWW*, pages 895–904, 2008.
- [146] Partha Pratim Talukdar and Koby Crammer. New regularized algorithms for transductive learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 442–457, 2009.
- [147] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *ICLR*, 2017.
- [148] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *ICLR*, 2018.
- [149] Jialong Han, Aixin Sun, Gao Cong, Wayne Xin Zhao, Zongcheng Ji, and Minh C. Phan. Linking fine-grained locations in user comments. *TKDE*, 30(1):59–72, 2018.

- [150] Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. Named entity recognition in tweets: an experimental study. In *EMNLP*, pages 1524–1534, 2011.
- [151] Shubhanshu Mishra and Jana Diesner. Semi-supervised named entity recognition in noisy-text. In *WNUT workshop*, pages 203–212, 2016.
- [152] Franck Dernoncourt, Ji Young Lee, and Peter Szolovits. Neuroner: an easy-to-use program for named-entity recognition based on neural networks. In *EMNLP: System Demonstrations*, pages 97–102, 2017.
- [153] Sonal Gupta and Christopher Manning. Improved pattern learning for bootstrapped entity extraction. In *CoNLL*, pages 98–108, 2014.
- [154] Rui Cai, Houfeng Wang, and Junhao Zhang. Learning entity representation for named entity disambiguation. In *CCL and NLP-NABD*, pages 267–278, 2015.
- [155] Matthew Francis-Landau, Greg Durrett, and Dan Klein. Capturing semantic similarity for entity linking with convolutional neural networks. In *NAACL-HLT*, pages 1256–1261, 2016.
- [156] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph and text jointly embedding. In *EMNLP*, pages 1591–1601, 2014.
- [157] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.
- [158] Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. Modeling mention, context and entity with neural networks for entity disambiguation. In *IJCAI*, pages 1333–1339, 2015.
- [159] Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. Effective lstms for target-dependent sentiment classification. In *COLING*, pages 3298–3307, 2016.
- [160] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

- [161] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *EMNLP*, pages 379–389, 2015.
- [162] Karl Moritz Hermann, Tomáš Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *NIPS*, pages 1693–1701, 2015.
- [163] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *NIPS*, pages 2440–2448, 2015.
- [164] Ming Tan, Bing Xiang, and Bowen Zhou. Lstm-based deep learning models for non-factoid answer selection. *CoRR*, *abs/1511.04108*, 2015.
- [165] Heng Ji and Ralph Grishman. Knowledge base population: Successful approaches and challenges. In *ACL*, pages 1148–1158, 2011.
- [166] Aasish Pappu, Roi Blanco, Yashar Mehdad, Amanda Stent, and Kapil Thadani. Lightweight multilingual entity extraction and linking. In *WSDM*, 2017.
- [167] Swapna Gottipati and Jing Jiang. Linking entities to a knowledge base with query expansion. In *EMNLP*, pages 804–813, 2011.
- [168] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.
- [169] Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. Collective annotation of wikipedia entities in web text. In *KDD*, pages 457–466, 2009.
- [170] Ricardo Usbeck, Michael Röder, Axel-Cyrille Ngonga Ngomo, Ciro Baron, Andreas Both, Martin Brümmer, Diego Ceccarelli, Marco Cornolti, Didier Cherix, Bernd Eickmann, Paolo Ferragina, Christiane Lemke, Andrea Moro, Roberto Navigli, Francesco Piccinno, Giuseppe Rizzo, Harald Sack, René Speck, Raphaël Troncy, Jörg Waitelonis, and Lars Wesemann. GERBIL: general entity annotator benchmarking framework. In *WWW*, pages 1133–1143, 2015.
- [171] Xiao Ling, Sameer Singh, and Daniel S. Weld. Design challenges for entity linking. *TACL*, 3:315–328, 2015.

- [172] Nadine Steinmetz and Harald Sack. Semantic multimedia information retrieval based on contextual descriptions. In *ESWC*, pages 382–396, 2013.
- [173] Wei Fang, Jianwen Zhang, Dilin Wang, Zheng Chen, and Ming Li. Entity disambiguation by knowledge and text jointly embedding. In *CoNLL*, 2016.
- [174] Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. Fast and space-efficient entity linking for queries. In *WSDM*, pages 179–188, 2015.
- [175] Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *ICML*, pages 1188–1196, 2014.
- [176] Stephen Guo, Ming-Wei Chang, and Emre Kiciman. To link or not to link? A study on end-to-end tweet entity linking. In *NAACL-HLT*, pages 1020–1030, 2013.
- [177] Joseph B Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society*, 7(1):48–50, 1956.
- [178] Robert Clay Prim. Shortest connection networks and some generalizations. *Bell Labs Technical Journal*, 36(6):1389–1401, 1957.
- [179] Chenliang Li, Aixin Sun, and Anwitaman Datta. TSDW: two-stage word sense disambiguation using wikipedia. *JASIST*, pages 1203–1223, 2013.
- [180] Chih-Hsuan Wei, Hung-Yu Kao, and Zhiyong Lu. Pubtator: a web-based text mining tool for assisting biocuration. *Nucleic acids research*, 41(W1):W518–W522, 2013.
- [181] Rezarta Islamaj Doğan, Robert Leaman, and Zhiyong Lu. Ncbi disease corpus: a resource for disease name recognition and concept normalization. *Journal of biomedical informatics*, 47:1–10, 2014.
- [182] Sunghwan Sohn, Donald C Comeau, Won Kim, and W John Wilbur. Abbreviation definition identification based on automatic precision estimates. *BMC bioinformatics*, 9(1):402, 2008.

- [183] Jennifer D’Souza and Vincent Ng. Sieve-based entity linking for the biomedical domain. In *ACL-IJCNLP*, volume 2, pages 297–302, 2015.
- [184] Bing Bai, Jason Weston, David Grangier, Ronan Collobert, Kunihiro Sadamasa, Yanjun Qi, Olivier Chapelle, and Kilian Weinberger. Learning to rank with (a lot of) word features. *Information retrieval*, 13(3):291–314, 2010.
- [185] Sun Kim, Lana Yeganova, Donald C Comeau, W John Wilbur, and Zhiyong Lu. Pubmed phrases, an open set of coherent phrases for searching biomedical literature. *Scientific data*, 5:180104, 2018.
- [186] Serguei VS Pakhomov, Ted Pedersen, Bridget McInnes, Genevieve B Melton, Alexander Ruggieri, and Christopher G Chute. Towards a framework for developing semantic relatedness reference standards. *Journal of biomedical informatics*, 44(2):251–265, 2011.
- [187] Serguei VS Pakhomov, Greg Finley, Reed McEwan, Yan Wang, and Genevieve B Melton. Corpus domain effects on distributional semantic modeling of medical terms. *Bioinformatics*, 32(23):3635–3644, 2016.
- [188] Qingyu Chen, Yifan Peng, and Zhiyong Lu. Biosentvec: creating sentence embeddings for biomedical texts. *CoRR*, abs/1810.09302, 2018.
- [189] Andrew L Beam, Benjamin Kompa, Inbar Fried, Nathan P Palmer, Xu Shi, Tianxi Cai, and Isaac S Kohane. Clinical concept embeddings learned from massive sources of medical data. *CoRR*, abs/1804.01486, 2018.
- [190] Omer Levy, Yoav Goldberg, and Ido Dagan. Improving distributional similarity with lessons learned from word embeddings. *TACL*, 3:211–225, 2015.
- [191] Xiangji Huang, Ming Zhong, and Luo Si. York university at trec 2005: Genomics track.
- [192] Xiaoshi Yin, Jimmy Xiangji Huang, Zhoujun Li, and Xiaofeng Zhou. A survival modeling approach to biomedical search result diversification using wikipedia. *TKDE*, 25(6):1201–1212, 2012.

- [193] Xiaoshi Yin, Jimmy Xiangji Huang, and Zhoujun Li. Mining and modeling linkage information from citation context for improving biomedical literature retrieval. *Information processing & management*, 47(1):53–67, 2011.
- [194] Thanh Hai Dang, Hoang-Quynh Le, Trang M Nguyen, and Sinh T Vu. D3ner: biomedical named entity recognition using crf-bilstm improved with fine-tuned embeddings of various linguistic information. *Bioinformatics*, 34(20):3539–3546, 2018.
- [195] Gustavo Aguilar, Suraj Maharjan, Adrián Pastor López-Monroy, and Thamar Solorio. A multi-task approach for named entity recognition in social media data. *CoRR*, abs/1906.04135, 2019.
- [196] Laura Wendlandt, Jonathan K Kummerfeld, and Rada Mihalcea. Factors influencing the surprising instability of word embeddings. In *NAACL-HLT*, pages 2092–2102, 2018.
- [197] Maria Antoniak and David Mimno. Evaluating the stability of embedding-based word similarities. *TACL*, 6:107–119, 2018.
- [198] Filip Ilievski, Piek Vossen, and Stefan Schlobach. Systematic study of long tail phenomena in entity linking. In *COLING*, pages 664–674, 2018.