Distribution Agreement

In presenting this thesis or dissertation as a partial fulfillment of the requirements for an advanced degree from Emory University, I hereby grant to Emory University and its agents the non-exclusive license to archive, make accessible, and display my thesis or dissertation in whole or in part in all forms of media, now or hereafter known, including display on the world wide web. I understand that I may select some access restrictions as part of the online submission of this thesis or dissertation. I retain all ownership rights to the copyright of the thesis or dissertation. I also retain the right to use in future works (such as articles or books) all or part of this thesis or dissertation.

Signature:

_____ _____

Zhexiong Liu Date

Hierarchical Entity Extraction and Ranking with Unsupervised Graph Convolutions

By

Zhexiong Liu
Master of Science

Computer Science

Jinho D. Choi, Ph.D.
Advisor

Shun Yan Cheung, Ph.D.
Committee Member

Michelangelo Grigni, Ph.D.
Committee Member

Accepted:

Lisa A. Tedesco, Ph.D.
Dean of the James T. Laney School of Graduate Studies

Date

Hierarchical Entity Extraction and Ranking with Unsupervised Graph Convolutions

By

Zhexiong Liu

Advisor: Jinho D. Choi, Ph.D.

An abstract of
A thesis submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Master of Science
in Computer Science
2020

Abstract

Hierarchical Entity Extraction and Ranking with Unsupervised Graph Convolutions

By Zhexiong Liu

Entity extraction problems have been extensively studied in terms of investigating the capability of extracting entities from text using natural language processing (NLP). Most research involves training learnable models on a large amount of corpus to extract entities and determine their salience. Typically, these systems aim to retrieve an array of ranked entities from a set of documents while giving queries, which mainly measure the relevance between queries and entities. However, this thesis leverages semantic and syntactic information within the documents to perform entities extraction as well as entity ranking. In particular, given document corpus, constituency parsing trees are constructed to extract entity mentions (phrases) for each article. Meanwhile, dependency parsing trees and entity coreference clusters are employed to build a relation graph, of which nodes denote entity mentions and edges denote mention relations. Moreover, graph convolution is performed on the relation graph to normalize the mention representation with respect to mention embeddings. Hierarchical density-based clustering and ranking mechanism are applied to compute entity priors. To evaluate this work, three models are proposed and evaluated on 60 annotated articles. Preliminary results illustrate that the usage of parsing trees, along with entity coreference relations improves the effectiveness of entity extraction and ranking. The interesting hierarchical trees for entity extraction, the principles for graph construction, as well as the system architecture serve as main contributions of this thesis.

Hierarchical Entity Extraction and Ranking with Unsupervised Graph Convolutions

By

Zhexiong Liu

Advisor: Jinho D. Choi, Ph.D.

A thesis submitted to the Faculty of the
James T. Laney School of Graduate Studies of Emory University
in partial fulfillment of the requirements for the degree of
Master of Science
in Computer Science
2020

Acknowledgments

This thesis marks my significant trajectory at Emory University, which is both inspiring and challenging. Completing a Master's program abroad is a journey on its own, but the experience of doing research and working in the Emory community is a set of invaluable treasures that would consistently encourage me to archive my highest potential.

I would love to thank all the amazing people who have always been supporting me in my research adventure. Especially, I would love to have my greatest appreciation to Dr. Jinho D. Choi, for being the best advisor I could ever have hoped for. This thesis would not be achievable without all his constructive feedback, availability, and motivation. I would also love to sincerely thank Dr. Shun Yan Cheung and Dr. Michelangelo Grigni for their extensively reviewing this thesis and presenting constructive suggestions without any reservation. Moreover, I would love to thank Dr. Dorian Arnold, Dr. Ulf Nilsson, Matt Davidson, Tamika F Hairston, and Yvette Hilaire who have always been encouraging me to complete this thesis.

I would appreciate my great colleagues Jiaying Lu and Shaun Lim for their collaborations on annotating the dataset utilized in this thesis. Meanwhile, I would like to thank excellent colleagues and friends in the Emory NLP group who offered me helpful discussions on my research. Particularly, I would like to thank my friends Jing Zhang and his wife Elle Hao for granting me substantial support for my studies. Besides, I would love to extend my sincere appreciation to Dr. Frederic Bien, Dr. Hojin Kim, Dr. Sarah Milla, Waugh Dorothy, and Edward Brenna who are incredible people I could ever work with.

I have always been grateful for my wonderful parents, grandparents, brothers, and sisters for all the love they possess and all the privilege they award me. I have always been blessed in gaining extraordinary opportunities and experience in such a great country.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Information extraction has recently drawn significant attention due to a tremendous amount of information accumulated in collections of unstructured documents. Natural language processing (NLP) community has facilitated the development of text processing applications to handle this literature, of which entity extraction constitutes crucial components of these pipelined systems. Moreover, automatic extraction of useful entities, such as organizations, persons, locations, and dates in given documents, can be employed to tackle downstream tasks such as event surveillance, question, and answering, information summarization [4][27][45]. In addition, the research of ranking entities has significantly emerged thanks to the increasing number of entities contained in a document [59]. Therefore, several tasks of ranking entities have been recently organized, such as WSDM Cup 2016 that calls for an entity ranking challenge using a large heterogeneous graph [58]. These tasks demonstrate the significance of entity extraction and ranking.

Although NLP research that focuses on extracting entities from a variety of documents has been well studied, determining the importance of these entities hypo-

thetically remains challenging tasks, especially for ranking entities [59]. A number of these research have involved identifying main entity or entity salience that can be considered as binary classifications, which determine relevant or irrelevant entities to the document [15]. Other research has paid attention to retrieving an array of entities from given documents, along with measuring the relevance between these entities to given queries [61]. Such research either heavily relies on training learnable models using a large volume of annotated data [17] or the way of determining main entities are mostly based on external queries outside of documents [10].

To bridge this gap, an unsupervised framework is proposed in this thesis to handle entities extraction and ranking. The unsupervised approaches not only offer an alternative strategy to retain model generalization from a limited dataset but also maintain robust performance without any hand-engineered features and specialized external knowledge. These benefits make the unsupervised entity extraction and ranking framework, which extensively minimizes the human annotation efforts and maintains satisfactory effectiveness, a desirable approach for this task.

1.2 Research Questions

The objective of this work is to extract significant entities and determine their importance with ranking scores for a given document. This thesis handles the task of entity extraction and ranking by examining the following research questions.

Determine the effectiveness of unsupervised approaches for extracting main entities from unstructured documents. Entity extraction research has recently spawned into two domains: unsupervised extraction and supervised extraction. As for the unsupervised extraction, a group of candidates is first extracted from documents though rules or heuristics. Several entity candidates are scored by their features, such as term frequency and term offsets. Other research leverages graph-based approaches, such as

PageRank, and TextRank [62], to build a graph in which nodes denote entities, and edges represent entity relations [38]. I devise these processes by leveraging unsupervised approaches to prune entity candidates with syntactic and semantic encodings. In particular, this thesis investigates the effectiveness of constituency parsing, dependency parsing, graph-based embedding and clustering for the entity extraction and ranking problems.

Determine the effectiveness of contextualized embeddings for entity or mention representation. Word embeddings are crucial steps in natural language processing pipelined tasks, which captures word (token) semantics information amongst sentences. However, despite the success of word embeddings have demonstrated their effectiveness in capturing word semantics, the flexibility of entity mention (phrase) embeddings remain challenges due to their inability that has important limitation to discriminate meanings of different entity mentions regarding the same entity. For instance, the phrase *Kansas City Chiefs* and *Super Bowl Winner* refer to the same entity but do not have the same phrase representation. Meanwhile, *The Champion* can refer to ambiguous meanings according to given context: a Super Bowl Champion or a fiction movie. In other words, embeddings usually remain sensitive to semantics in a variety of contexts. Therefore, accurately capturing the semantics of ambiguous words or phrases can be crucial in NLP applications. This thesis leverages the contextualized word or phrase embedding, which dynamically updates the representation of words and phrases according to their contexts [1][7]. Concretely, this thesis aims to investigate the effectiveness of contextualized embedding for capturing entity relations in terms of entity coreference and entity dependencies.

Determine the effectiveness of graph-based convolution for entity mention representation. Graphs have been well employed to represent entity relations because of their good properties of capturing the node-to-node relationship and other relevant graph information [19]. A number of the research make use of graph con-

volutions proposed in spectral graph theory [6][23] to extract featured information in the dataset spectrum by using parameterized filters in graph convolutional neural networks (GCNN). Generally, GCNN shares filter parameters over graph nodes and transform node information among their neighborhoods though performing convolution operations. This thesis takes advantage of spectral graph convolutions to implement an unsupervised framework that leverages the dependency parsing and coreference convolution relations to dynamically normalize the entity mention representation. In particular, this work aims to evaluate the effectiveness of graph convolution for improving phrase representation, and in addition to further improving density-based spatial clustering in terms of model efficiency and accuracy.

1.3 Contribution

This thesis investigates the entity extraction and ranking tasks with an unsupervised learning framework, and reports attempt to contextualized embedding, graph-based convolution, and density-based clusters. Extensive experiments have been conducted to evaluate the effectiveness of proposed models. The contributions of this thesis fall into the following folds.

This thesis introduced a problem of extracting main entities with hierarchical ranking, which has been less investigated in an unsupervised way. To incorporate multiple NLP and graph theories to efficiently solve hand-on tasks, end-to-end entity extraction and ranking framework has been implemented, which show high potential for downstream NLP applications.

Accompanying with the unsupervised framework, a fresh annotated dataset based on NELA2017 [26], consisting of almost 1000 human-annotated entities with respect to top 3/5/10 entities, are released. These annotations not only remain useful for evaluating entity extraction and ranking tasks, but also fruitful for other NLP appli-

cations such as event extraction.

This thesis proposed a simple yet effective normalization mechanism for entity mention embedding that covers both positional information and entity coreference. This mechanism significantly improves extraction performance by eliminating entity redundancies and taking advantage of contextualized noun phrases, pronouns, and single terms to enrich entity candidates.

This thesis takes an initial step in constructing an entity relation graph with nodes as entities and edges as entity distance. To enhance the representation of the entity mentions, the framework leveraged dependency parser and entity coreference to weight edges. In addition, a k-block graph convolution is employed to aggregate neighbor information and improve the performance of the density-based cluster algorithm.

1.4 Organization

This Thesis is organized as 5 chapters: Chapter 1 discusses the motivation, research questions, and main contribution. Chapter 2 discusses the background and related work on word embedding, keyword extraction, followed by entity ranking and graph construction related to work in each of the subsequent chapters. Chapter 3 goes into the approach pipeline that provides details on an unsupervised framework, language parsing, coreference resolution, embedding normalization, graph convolution, clustering, and ranking. Chapter 4 discusses the experimental settings including gold standard the data collection, inter-annotator agreement, data preprocessing, approach evaluation, comparison with other approaches, and result visualization. Chapter 5 discusses the quantity and quality analysis, effectiveness examination, along with error analysis, and further work.

Chapter 2

Backgrounds

2.1 Word Embeddings

Word embeddings are approaches that encode words (tokens) into fixed-length vectors, which are found very useful in NLP tasks. Word embeddings can be classified into count-based embeddings and prediction-based embeddings [2].

The count-based embeddings mainly leverage word-context co-occurrence counts in a corpus. These embeddings are represented as word-context matrices [56]. For example, Deerwester et al. (1990) proposed Latent Semantic Analysis (LSA) to produce word-context matrices that can be employed to encode words [12]. Afterward, Lund et (1996) al. introduced the Hyperspace Analogue to Language (HAL) approaches [40] to inversely compute the co-occurrence between context words and target words[3]. Rohde et al. (2004) improved HAL by performing a normalization mechanism that can differentiate word frequency [52]. Lebret and Collobert (2013) have improved count-based approaches by applying a Hellinger PCA transformation to a word-context matrix [33]. Recently, Pennington et al. (2014) proposed a significant word embedding model named GloVe [48], which leverages ratios of co-occurrences to semantically represent a pair of words and trains a linear relation that

can maximize the similarity between each word pair. For instance, given a GloVe vector $w(x)$ for a word x, a linear relation amongst *king, queen, man* and *woman* can be interpreted as: $w(queen) = w(king) - w(man) + w(woman)$[48].

The count-based embeddings express meanings of words by considering its global information but fail to disambiguate the different meanings of the same word by analyzing its surrounding context. For example, the *bank* in the sentence *a robber robs a bank near the Hudson bank* has different meanings. To handle the problem of word ambiguities, contextualized embedding models have emerged in recent studies. Peters et al. (2018) trained a language model named ELMo [50] that has the capability of creating context-based word embeddings. Moreover, Devlin et al. (2018) proposed a more powerful contextualized model named BERT [14] that can greatly disambiguate contextualized word representations in different contexts [49]. Several downstream tasks have achieved significant improvement by using these contextualized word embeddings, thus BERT embeddings are integrated into the proposed models to represent mention embeddings.

2.2 Keyphrase Extraction

The keyphrase extraction task aims to extract an array of phrases that contain important and topical information in a given document [60]. The keyphrase extraction task mainly consists of two steps: extracting a set of candidates from a given document and determining the importance of these candidates [28]. Usually, binary classification is applied in terms of marking important or unimportant phrases but cannot distinguish their importance levels. To handle this problem, several features such as syntactic and external knowledge-based features are employed to quantify phrase importance [21].

Commonly, a definition of the importance of a phrase refers to its relations to other

candidate phrases in a given document. Intuitively, a candidate phrase is significant if it has a large group of related phrases in the document [21] or it is significantly related to a topic discussed in the document [37]. Therefore, topic-based approaches have been prevailing in recent research as they propose to extract keyphrases that have both strong relation to the topics and maintain the phrase generality that can cover multiple topics in a given document.

Besides, researchers also measure phrase importance by leveraging semantic and syntactic relatedness based on a graph. Typically, the graph-based approaches involve building a graph, of which each node represents a phrase and an edge that connects two phrases is measured by their syntactic and semantic relevance. For example, graph ranking, such as PageRank [46] proposed by Page et al. (1999) and TextRank proposed by Mihalcea and Tarau (2004) [48], have become prominent for text processing in several NLP research. These approaches leverage graph-based theories to score candidate phrases and consequentially select the most significant ones, but cannot be directly applied to entity extraction and ranking. To bridge this gap, this thesis focus on the extracting and ranking entity with graph-based approaches.

2.3 Entities Ranking

An entity is denoted by a sequence of tokens that carry substantial information in a sentence [22]. The main difference between phrases or entity mentions and entities is that an entity might consist of several entity mentions [25]. For example, regarding the sentence, *"Kansas City Chiefs defeated the National Football Conference Champion, San Francisco 49ers, in February 2020"*, the *Francisco 49ers* and *National Football Conference Champion* are both entity mentions but reference to the same entity. Considering the disambiguation and coreference of these phrases is the main difference between keyphrase extraction and entity extraction. Formally, the task of identifying

entity mentions that refer to the same entity is named entity coreference resolution [20][43][55].

In most research, entity extraction tasks leverage entity coreference resolution form mention clusters that significantly eliminate phrase ambiguities. Entity ranking tasks usually employ a given query to measure the significance of extracted entities. In other words, the more relevant to the given query, the more significant an extracted entity is [16]. However, entities have underlying importance based solely on syntactic and semantic information in given documents. For example, the main entities in a document[1] regarding Super Bowl 2020 can be directly extracted without any queries. Since the entity such as *Kansas City Chiefs*, *San Francisco 49ers*, *National Football League*, *Patrick Mahomes*, *Hard Rock Stadium* and so on play significant roles in conveying main information in the document, deeply investigate the semantic and syntactic relations in the given document might be beneficial. Therefore, this thesis addresses the task of entity extraction and ranking based on semantic and syntactic information in the document regardless of any external knowledge or given queries. The challenge of this task is that the significance of entities is implicitly expressed in the context of the document [59], which can hardly measure and quantify, and further ensure the difficulties of entity ranking task.

Nevertheless, entity ranking have several applications in information extraction and NLP fields [13] [18] [43]. For example, entity-oriented web search leverage the entity ranking to improve web entity recommendation; web sanctification utilizes the entity importance to help semantic tagging; entity ranking also potentially enhances the knowledge extraction by performing ranks and links to given knowledge [59]. Typically, entity ranking and extraction can be mutually enhanced and solved, thus the entity ranking procedure is integrated into the clustering components in this thesis that rank entity importance by leveraging the hierarchical information of the minimal

[1]The article 51167 in NELA2017 dataset

spanning tree.

2.4 Graph-based Approaches

Previous studies have explored graph-based modeling to share entity representations across entity relation in information extraction tasks [39]. Recently, neural networks with graph-based approaches have achieved significant performance compared to rule-based approaches on several downstream NLP tasks. For instance, Katiyar et al. (2018) proposed an approach that extracts entities by constructing a hypergraph [30]. Yang et al. (2015) have constructed a knowledge graph to embedding entities and relations [64]. As for the relation extraction, Christopoulou et al. (2018) have modeled entity relations on a graph and perform a random walk to integrate node representations [11], which has demonstrated the effectiveness in terms of incorporating features such as syntactic relations [47]. Conclusively, most of these works leverage the graph nodes to update node embeddings through learning processes.

Many graph-based models for entity extraction focus on leveraging syntactic and semantic information to construct entity dependency within a sentence [65][66]. These approaches suffer from significant errors from ignoring inter-sentence relations and are limited to construct graphs to capture comprehensive information for given documents. More recently, an intern-sentence relation extraction approach has been proposed by Sunil Kumar Sahu et al. [53] to handle the relation extraction tasks. These research significantly exploit the graph-based approaches for entity relation classifications, but few of them involve entity extraction and ranking tasks.

Therefore, this thesis bridge this gap and pay attention to utilizing graph convolution to explore entity extractions and ranking. Meanwhile, motivated by the state-of-the-art coreference resolution results in [29] [53], the proposed framework incorporates both the dependency parsing relations within sentences and coreference

relations across sentence as inputs for graph construction. The model uses a simple yet effective convolution operations to dynamically normalize the node embeddings for entity mentions. In other words, the proposed graph-based model does not require a time-consuming training process, instead, it only allows a limited dataset without any external information.

Chapter 3

Approaches

The entity extraction and ranking task can be defined as a ranking problem. In other words, the objective of the entity extraction and ranking is to extract an array of ranked entities that dominate the significant information in a given document. Ranking extracted entity brings benefits for several entity-related tasks such as question and answering, topic modeling, and text summarization [21][63].

Given a document D, the goal is to extract a group of entities denoted as E, of which each entity e has a ranking score *r(e)* that marks the importance of the entity in document D. Since an entity consists of several entity mentions in given document, the definitions of entity mention and entity clusters are respectively formulated. In particular, let x be an array of word (tokens) $[x_1, x_2, \ldots, x_n]$ in a document, the entity mention (span) consisting of a subarray of X is denoted as $m = [x_s, \ldots, x_e]$, in which the tuple (s, e) denotes a start and end offset of the mention respectively. Normally, (s, e) is considered as global offsets in document D. In addition, the entity clusters $c(e)$ for document D that represent the entity is formulated as a group of entity mentions (spans) $c(e) = \{m_1, m_2, ..., m_k\}$ where k is a positive integer. Therefore, the objective of this project is to extract an array of entity cluster $\{c(e_1), c(e_2), ...c(e_n)\}$ and a list of corresponding ranking scores $\{r(e_1), r(e_2), ...r(e_n)\}$ from given document

D, where n is a positive integer.

3.1 Constituency Parsing

Constituency parsing aims to explore syntactic structure of a given sentence, which plays a substantial role in mediating between linguistic expression and meaning [54]. For example, the sentence[1] *Kansas City Chiefs won their first Super Bowl on February 2020* can be parsed along with parts of speech (POS) tags [41] and constituency [5] tags as shown in Figure 3.1.

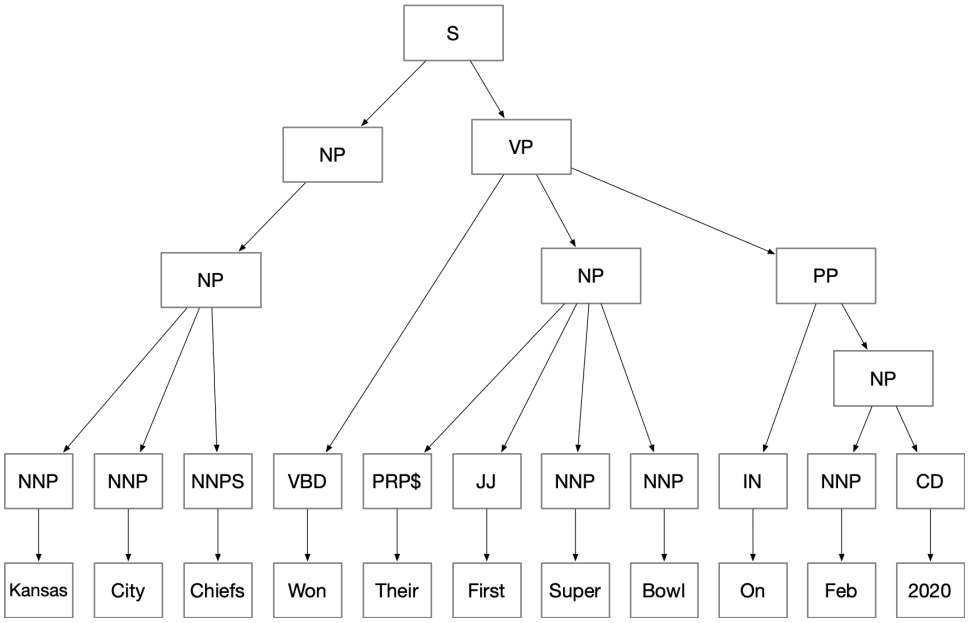

Figure 3.1: Constituency parsing tree.

In this tree, the abbreviations *S, NP, VP, PP, JJ, IN, CD, NNP, NNPS, VBD,* and PRPS refer to tags described in Table 3.1. The words (tokens) in the sentence are denoted as tree leaves, and play a role of being substitutes of their immediate parents. The rest of the nodes are formed out of constituency parsing by following

[1] A sentence from article 51167 in NELA2017 dataset

Context-Free Grammars [9]:

$$\begin{aligned} S &\Rightarrow NP|VP \\ NP &\Rightarrow NP|NNP|NNPS|PRP\$|JJ|CD \\ VP &\Rightarrow VBD|NP|PP \\ PP &\Rightarrow IN|NP \end{aligned} \tag{3.1}$$

As constituency trees ensure the capability of parsing sentence structure in the phrase level, the state-of-the-art constituency parser is employed in this thesis to trunk noun phrases. As stated in Table 3.1, the *NP* tag denotes the noun phrases in given sentences, thus all the *NP* on the parsing tree would be collected as a set of raw entity mentions. The mention set for document D is formulated as

$$M(D) = \{m_1, m_2, ..., m_l\} \tag{3.2}$$

where l is a positive integer, representing the number of mentions in document D. Moreover, the mention m_i can be denoted as an array of words x, refering

$$m_i = (x_s, ..., x_e) \tag{3.3}$$

where $i = 1, 2, ..., l$, and (s, e) represent the start offset and end offset of mention m_i in given document D, respectively.

3.2 Dependence Parsing

Dependency parsing has been well-studied in NLP research, which dominates mainstream tasks regarding the syntactic structure of a sentence. In a dependency tree (referring Figure 3.2) for sentence *Kansas City Chiefs won their first Super Bowl on*

Table 3.1: The table of POS & constituency tags and examples

Tags	Description	Examples
S	sentence	-
NP	noun phrase	Kansas City Chiefs
VP	verb phrase	won their first Super Bowl
PP	prepositional phrase	on Febrary 2020
JJ	adjective	first
IN	conjunction, preposition	on
CD	cardinal number	2020
NNP	noun, proper singular	Kansas, City, Chiefs, Super, Bowl
VBD	verb, past tense	won
NNPS	noun, proper plural	Chiefs
PRP$	pronoun, possessive	their

February 2020, each note represents a word and each edge represents a dependency relation between two words. The labels *nsubj, prep, dobj, pobj, nn, poss, amod,* and num refer the dependency relations in Table 3.2.

The relation between words in the dependency tree can be intuitively interpreted. For example, the words of *their*, *first* and *super* are served as modifiers for the word *bowl*, which points out a heading information in the phrase *their first super bowl*. In other words, the headword *bowl* hypothetically conveys more significance than the other words. Formally, the headword in mention m can be formulated as

$$h(m) = x_i \tag{3.4}$$

where x_i is a head token in mention m and the other tokens x_j ($j \neq i, and s <= j <= e$) are all the modifiers of x_i in dependency parsing tree.

Another advantage of the dependency tree is that the dependency tree can capture syntactic relations in terms of sentence subjects and objects, which convey significant information. Besides, the dependency parsing tree also performs a hierarchical structure that can be used to present entity hierarchy, and correspondingly measure entity importance.

In order to leverage syntactic information in sentences, a state-of-the-art dependency parsing is performed to document D. Meanwhile, with the mention set $M(D)$ obtained in Equation 3.2, the parsed tokens in Table 3.2 are merged into noun phrases, which conforms to mention set $M(D)$ trunked through constituency parsing tree in Section 3.1. In particular, the dependency parsing tree with merged mentions is shown in Figure 3.3.

To leverage and quantify dependency relations shown in the tree (Figure 3.3), the tree-based path is defined. Specifically, the distance of mention m_i and m_j on a dependency tree is defined as the distance between head word $h(m_i)$ and $h(m_j)$ defined in Equation 3.4 for mention m_i and m_j respectively. Thus, the distance between mention m_i and m_j is defined as the number of edges on the shortest path that connects two mentions. Specifically,

$$dist_p(m_i, m_j) =< h(m_i), h(m_j) > \tag{3.5}$$

where $i \neq j$ and $i <= l$, $j <= l$. Meanwhile the distance between the same mentions is defined as 0.

Table 3.2: The table of dependency relations and examples

Relations	Description	Examples
Nsubj	represent nominal subject	Chiefs
Prep	prepositional phrase	On
Dobj	direct object	Bowl
Pobj	object of a preposition	February
NN	noun compound modifier	Kansas, City, Super
Poss	possession modifier	Their
Amod	adjectival modifier	First
Num	numeric modifier	2020

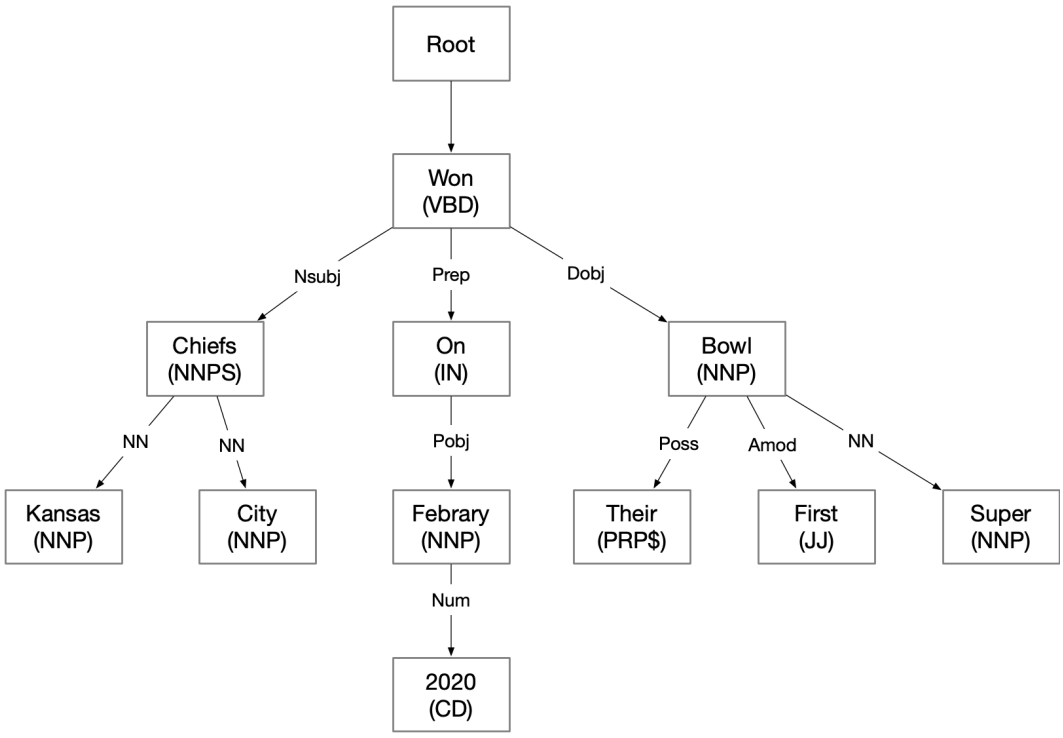

Figure 3.2: Dependency parsing tree for tokens.

3.3 Coreference Resolution

Even if the dependency tree successfully measures syntactic relations between two words in a given sentence, it cannot measure syntactic relation between two words that come from different sentences. In order to handle this issue, The entity coreference relations are employed to handle inter-sentence entities [43]. Generally, entity coreference resolution aims to determine if a pair of entity mentions (m_i and m_j) refers to the same entity e or the same entity cluster $c(e)$. For instance, given the following sentence[2]:

> *"Kansas City Chiefs defeated the National Football Conference champion, San Francisco 49ers, in February 2020. This team made their first Super Bowl victory since Super Bowl IV and the Chiefs' first NFL championship since joining the league".*

[2]A sentence from article 51167 in NELA2017 dataset

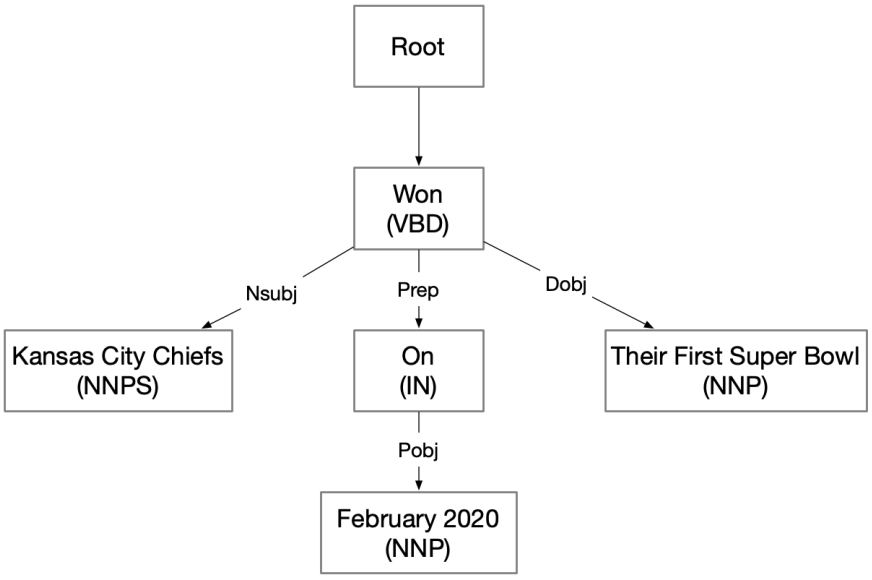

Figure 3.3: Dependency parsing tree for noun phrases.

A person can easily recognize the coreference of the entity mentions in given sentences and partition them into the four clusters: (*Kansas City Chiefs, This team, their*), (*National Football Conference champion, San Francisco 49ers*), (*Super Bowl victory, the Chiefs' first NFL championship*), (*Super Bowl IV, the league*), along with singleton *February 2020*. However, it is significantly challenging for an automatic coreference resolver.

In this thesis, a state-of-the-art coreference resolution model is employed to resolve coreference in documents. Particularly, SpanBERT [29], a pre-training model that is trained to represent entity mentions is leveraged to perform entity coreference for mention set $M(D)$ of document D. SpanBERT is an extending work of BERT [14], which achieved 79% F1 score on OntoNotes dataset [51] for coreference tasks.

To implement this, the training process of the coreference resolution system based on SpanBERT is partitioned into two phrases, of which the first one is to train a better word embedding based on SpanBERT; and the second one is to generate coreference clusters for each entity e.

For the first phrase, the Transformer [57] is employed to embed an array of tokens

$x_i \in X$ into a d-dimensional vector denoted by $\mathbf{x}_i \in \mathbb{R}^d$ $(i = 1, 2, ..., n)$. Based on the SpanBERT, the embedding for each token x_i is further represented as

$$\mathbf{y}_i = f\left(\mathbf{x}_{s-1}, \mathbf{x}_{e+1}, \mathbf{p}_{i-s+1}\right) \tag{3.6}$$

where $\mathbf{x}_{s-1}, \mathbf{x}_{e+1}$ represent the words at right before the start and after the end words of the mention m in an array X, and $\mathbf{p}_{i-s+1}$ is the position embedding in d dimension that marking the relative global offsets of the start and end boundary of word x_i in mentions m. $f(\cdot)$ is a simple Feed-forward Network (FFN), *GeLU* is the activation function, and *Norm* represents a normalization layer [24]. Specific, the formulation of the FFN network is represented as

$$\begin{aligned} \mathbf{s}_0 &= [\mathbf{x}_{s-1}; \mathbf{x}_{e+1}; \mathbf{p}_{i-s+1}] \\ \mathbf{s}_1 &= \text{Norm}\left(\text{GeLU}\left(\mathbf{W}_1 \mathbf{s}_0\right)\right) \\ \mathbf{y}_i &= \text{Norm}\left(\text{GeLU}\left(\mathbf{W}_2 \mathbf{s}_1\right)\right) \end{aligned} \tag{3.7}$$

where $\mathbf{W}_1$ and $\mathbf{W}_2$ are learnable parameter matrices in FFN. In order to employ the vector representation $\mathbf{y}_i$ to predict word x_i in mention m, a loss function is formulated as

$$\mathcal{L}\left(x_i\right) = -\log P\left(x_i|\mathbf{x}_i\right) - \log P\left(x_i|\mathbf{y}_i\right) \tag{3.8}$$

where $P\left(x_i|\mathbf{y}_i\right)$ is the condition probability of word x_i for given vector $\mathbf{y}_i$.

The second phase is to learn the entity mention clusters. Specifically, the task goes to find a set of the most possible antecedent mentions $m_j \in \mathcal{Y}(i)$ of current mention m_i, where $\mathcal{Y}(i) = \{\epsilon, m_1, \ldots, m_{i-1}\}$, and a dummy antecedent ϵ represents that there is no corresponding antecedent for m_i. In other words, [34][35] is going to learn a conditional probability distribution $P(\epsilon, m_1, \ldots, m_{i-1}|D)$ for m_i in document D to figure out the most likely antecedent mentions and produce them as clusters.

Specifically, the distribution can be formulated as

$$\begin{aligned} P\left(\epsilon, m_1, \ldots, m_{i-1}|D\right) &= \prod_{j=1}^{i-1} P\left(m_j|D\right) \\ &= \prod_{j=1}^{i-1} \frac{\exp\left(s\left(m_i, m_j\right)\right)}{\sum_{y' \in \mathcal{Y}(i)} \exp\left(s\left(m_i, y'\right)\right)} \end{aligned} \tag{3.9}$$

where $s(m_i, m_j)$ is the coreference scores for mention m_i and m_j, which is further formulated as

$$s(m_i, m_j) = \begin{cases} 0 & j = \epsilon \\ s_{\mathrm{m}}(m_i) + s_{\mathrm{m}}(m_j) + s_{\mathrm{a}}(m_i, m_j) & j \neq \epsilon \end{cases} \tag{3.10}$$

where $s_{\mathrm{m}}(m_i)$ is a mention score for m_i, and $s_{\mathrm{a}}(m_i, m_j)$ is an antecedent score for mention m_j being an antecedent of mention m_i.

Given mention representations, the scoring functions are computed by another FFN:

$$\begin{aligned} s_{\mathrm{m}}(m_i) &= \boldsymbol{w}_{\mathrm{m}} \cdot \mathrm{FFN}_{\mathrm{m}}\left(\boldsymbol{g}_i\right) \\ s_{\mathrm{a}}(m_i, m_j) &= \boldsymbol{w}_{\mathrm{a}} \cdot \mathrm{FFN}_{\mathrm{a}}\left(\left[\boldsymbol{g}_i, \boldsymbol{g}_j, \boldsymbol{g}_i \circ \boldsymbol{g}_j\right]\right) \end{aligned} \tag{3.11}$$

where $\cdot$ represents the dot product, $\boldsymbol{w}_{\mathrm{m}}$ and $\boldsymbol{w}_{\mathrm{a}}$ denote the learnable weight parameters, $\circ$ represents element-wise multiplication between two vectors, and $\boldsymbol{g}_i$ is the mention representation for m_i. Specifically, $\boldsymbol{g}_i$ is a concatenation of the two transformer encoder of start and end point of mention m_i along with an attention vector that is computed over the spenBERT representation $\mathbf{y}_i$. The formula is defined as

$$\boldsymbol{g}_i = [\boldsymbol{x}_{\mathrm{s}-1}; \boldsymbol{x}_{\mathrm{e}+1}; \hat{\boldsymbol{x}}_i] \tag{3.12}$$

where the $\hat{\boldsymbol{x}}_i$ is the sum of word embeddings in mention m_i with attention weights.

Normally, the weighted sum of word embeddings is formulated as

$$\begin{aligned} \alpha_t &= \boldsymbol{w}_\alpha \cdot \mathrm{FFN}_\alpha\left(\mathbf{y}_i\right) \\ a_{i,t} &= \frac{\exp\left(\alpha_t\right)}{\sum_{k=\mathrm{e}}^{\mathrm{s}} \exp\left(\alpha_k\right)} \\ \hat{\boldsymbol{x}}_i &= \sum_{t=\mathrm{s}}^{\mathrm{e}} a_{i,t} \cdot \boldsymbol{x}_t \end{aligned} \tag{3.13}$$

where $\boldsymbol{w}_\alpha$ is a trainable parameter, which is learned by the objective function 3.9, and $\mathbf{y}_i$ is obtained in Equation 3.7. Thus, a list of the coreference clusters $C(D)$ for document D is formulated as

$$C(D) = \bigcup_{e \in E} c(e) \tag{3.14}$$

where e and E are the entity and entity set for D, respectively.

Thus, the inter-sentence entity relations that play an important role of indicating of local and non-local dependencies [53] are obtained. To quantify these inter-sentence relations, the distance between mention m_i and m_j that belong to different sentences but fall into the same coreference cluster $c(e)$ is formulated as

$$dist_c(m_i, m_j) = \begin{cases} 0 & \text{otherwise} \\ |c(e)| & m_j \in c(e), m_j \in c(e), i \neq j \end{cases} \tag{3.15}$$

where $|c(e)|$ denotes the number of mentions in the entity mention cluster $c(e)$.

3.4 Embedding Normalization

To encode extracted entity mention m_i in Equation 3.3, mention embeddings are applied based on BERT [14]. For each word (token) $x_{i,j}$ in mention m_i where $j \in [s, e]$, the word embedding of $x_{i,j}$ is performed as the sum of the last four layers of the BERT architecture, denoting as $\mathbf{x}_i$. In particular, the embedding of mention m_i is formulated

as

$$\mathbf{x}_u(m_i) = \sum_{k=s}^{e} \alpha_k \mathbf{x}_k \tag{3.16}$$

where the α_k is the attention score obtained through BERT between token x_k and x_{root}, where x_{root} is the root node in the dependency tree discussed in Session 3.2, s and e are the global offset of the mention m_i in documnet D.

Besides, coreference can also be used to improve mention embeddings. In other words, the mentions in the same coreference cluster can share similar embedding features. Thus, the normalized embedding of mention $m_i \in c(e)$ based on coreference resolution is formulated as

$$\mathbf{x}_n(m_i) = \mathbf{x}(m_{e,i}) + \bar{\mathbf{x}}(m_e) \tag{3.17}$$

where $\bar{\mathbf{x}}(m_e)$ is the average embeddings of entity cluster $c(e)$, which is formulated as

$$\bar{\mathbf{x}}(m_e) = \frac{1}{|c(e)|} \sum_{k=1, m_k \in c(e)}^{|c(e)|} \mathbf{x}_u(m_k) \tag{3.18}$$

To incorporate Equation 3.17 and 3.18, the embedding of mention m_i can be deduced as

$$\mathbf{x}(m_i) = \begin{cases} \mathbf{x}_u(m_i) & \text{if } m_i \notin c(m_i) \\ \mathbf{x}_n(m_i) & \text{otherwise} \end{cases} \tag{3.19}$$

In other words, the mention embedding of m_i is the mean of the embeddings of mentions in cluster $c(e)$ plussing its own embedding from BERT if $m_i \in c(e)$; otherwise, the mention embedding of m_i is only from the last four layers of BERT.

3.5 Graph Convolutions

The dependency-based graphs represent each sentence as a tree with modifier dependency and relation edges between words [44]. Thus, the same mechanism can be applied to mentions. Each mention on the graphs is surrounded by its directly related entity mentions, and connected mentions have hypothetically strong relations in terms of similar mention embeddings. Performing convolution operation on the connected nodes on a graph can smooth the node representation. Therefore, the graph convolution can be employed to capture the most relevant mention neighbors for the current mention and avoid the modeling of unrelated nodes [44].

Inspired by the recently favored graph convolutional neural networks [31][32] in the NLP community, an unsupervised graph convolution approaches are performed to further capture syntactic relations. In other words, the dependency and coreference relations are once again to form connections between mention nodes on the graph. In particular, a dependency-based graph are constructed, of which nodes are a mention in $M(D)$, along with corresponding mention embeddings in 3.19 and edges denote the distance between two mention nodes defined in 3.5. The convolution operations are performed to update these nodes in terms of mention embeddings. Moreover, a k-order convolutions can be used to update the nodes in the spectrum of k-order neighbors. Specifically, a graph $\mathcal{G}_p$ based on the dependency tree is formulated as

$$\mathcal{G}_p = \{\mathcal{V}, \mathcal{E}\} \tag{3.20}$$

where $\mathcal{V}$ is an array of nodes with respect to mention set $M(D)$ of document D, and $\mathcal{E}$ denotes the edge set computed by the formula 3.5. More concretely, each entry a_{m_i,m_j} in adjacency matrix $A \in \mathbb{R}^{n \times n}$ of the graph $\mathcal{G}_p$ is defined as

$$a_{m_i,m_j} = dist_p(m_i, m_j) \tag{3.21}$$

In addition, the degree matrix of graph $\mathcal{G}_p$ can be deduced as $D = \text{diag}(d_1, d_2, ..., d_n)$, where d_i $(i = 1, 2, ..., n)$ counts the number of times an edge terminates at the node m_i. Therefore, the Laplacian matrix $L \in \mathbb{R}^{n \times n}$ can be denoted as

$$L = D - A \tag{3.22}$$

Meanwhile, the symmetric normalized Laplacian matrix $L_s \in \mathbb{R}^{n \times n}$ is formulated as

$$L_s = D^{-\frac{1}{2}} L D^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \tag{3.23}$$

where $I \in \mathbb{R}^{n \times n}$ is the identity matrix of A. As indicated in [36], a graph filter G_{pf} can be performed to integrate graph structure and node features, which is defined as

$$G_{pf} = I - L_s \tag{3.24}$$

Thus, a linear graph convolution can be formulated as

$$\bar{X} = G_{pf} X^{\mathsf{T}} \tag{3.25}$$

where $X = [\mathbf{x}(m_1), \mathbf{x}(m_2), ..., \mathbf{x}(m_l)]$, and $\mathbf{x}(m_i)$, $i \in [1, n]$ is the mention embedding in equation 3.19. Even if Equation 3.25 can update current node (mention) embedding by performing convolution to its direct neighbors, a first-order on graph $\mathcal{G}_p$ convolution 3.25 cannot exploit nodes without direct connections to itself. It is necessary to aggregate k-order neighbors to globally update node embeddings, which normally happens in large and sparse graphs. Therefore, a k-order graph filter G_{pf}^k is denoted as

$$G_{pf}^k = (I - L_s)^k \tag{3.26}$$

Consequencially, a k-order graph convolution can be applied obtain the updated men-

tion embeddings $\widetilde{X}_p$

$$\widetilde{X}_p = G_{pf}^k X \tag{3.27}$$

where k is a positive integer denoting k order connections between current nodes and their k-order neighbors.

Similarly, the graph $\mathcal{G}_p$ with the convolution for the parsing tree can be adopted to a coreference graph $\mathcal{G}_c$ with coreference resolution. The former one captures the mentions regarding relations within sentences; the latter one captures the mentions regarding relations amongst sentences. Therefore, the mention representation $\widetilde{X}$ in document D regarding both the within sentences and cross sentences relations can be formulated as

$$\widetilde{X} = \widetilde{X}_p + \widetilde{X}_c \tag{3.28}$$

where $\widetilde{X}_c$ is the updated mention embedding based on the coreference graph $\mathcal{G}_c$.

3.6 Clustering

Given the mention embedding $\widetilde{X}$ for document D, a fully connected graph $\mathcal{G}_{cluster}$ can be constructed to perform clustering. Formally,

$$\mathcal{G}_{cluster} = \{\mathcal{V}, \mathcal{E}\} \tag{3.29}$$

where $\mathcal{V}$ is mention set in $M(D)$ for document D with updated mention embeddings in 3.28 and $\mathcal{E}$ is edge set with weights that measure the Euclidian distance between two nodes.

Furthermore, in order to enlarge the embeddings features between mention m_i and mention m_j, comparable embeddings $\mathbf{r}(m_i)$ and $\mathbf{r}(m_j)$ is defined, which evolves

concatenate of three kinds of embeddings as follows

$$\begin{aligned}\mathbf{r}(m_i) &= [\mathbf{x}(m_i); \mathbf{x}(m_i) - \mathbf{x}(m_j); \mathbf{x}(m_i) \circ \mathbf{x}(m_j)] \\ \mathbf{r}(m_j) &= [\mathbf{x}(m_j); \mathbf{x}(m_j) - \mathbf{x}(m_i); \mathbf{x}(m_i) \circ \mathbf{x}(m_j)]\end{aligned} \tag{3.30}$$

where $\mathbf{x}(m_i)$ and $\mathbf{x}(m_j)$ are the embeddings obtained from 3.28, and $\circ$ is the pairwise production. Afterwards, the adjacency matrix $A \in \mathbb{R}^{n \times n}$ for graph $\mathcal{G}_{cluster}$ can be defined in terms of each entry $a_{i,j}$ as

$$a_{i,j} = \|\mathbf{r}(m_i) - \mathbf{r}(m_j)\|_2^2 \tag{3.31}$$

Having graph $\mathcal{G}_{cluster}$, a clustering algorithm aims to partition $\mathcal{G}_{cluster}$ into several components, of which each part represents an entity cluster. As the adjacency matrix shows the property that dense areas are separated by sparse areas, the Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [8] algorithm is performed to make graph clusters. Meanwhile, HDBSCAN does not need a parameter to indicate the number of clusters, which is useful and significant to cluster entity mentions with noises as singletons. In addition, finding correct cluster numbers is extremely bias as the entity number varies in different documents. Therefore, the HDBSCAN cluster is a good fit for this research.

Specifically, HDBSCAN leverages the concept of core distance $\text{core}_k(m_i)$, which is defined with parameter k and a mention node m_i, drawing a circle at m_i with the core distance as the radius and its area covers the k nearest neighbor nodes. In other words, small core distance represents a dense area of the nodes and large core distance represents a relatively a sparse area of the nodes. In order to spread apart nodes with low density, a mutual reachability distance d_k for mention m_i and m_j is defined as

$$d_k(i,j) = \max\{\text{core}_k(m_i), \text{core}_k(m_j), a_{i,j}\} \tag{3.32}$$

where $a_{i,j}$ is defined in 3.31.

With the mutual reachability distance as a new distribution of graph nodes, the HDBSCAN first generates a Minimum Spanning Tree (MST) and then leverages a single linkage to convey this MST to a hierarchical tree. Specifically, each mention node has a score of $\lambda(m_i)$

$$\lambda(m_i) = \frac{1}{dist(m_i)} \tag{3.33}$$

where $dist(m_i)$ is the distance on the hierarchy tree for each mention node. Moreover, this hierarchical tree can be condensed and clustered by giving a minimum cluster size parameter while maintaining and splitting each linkage [8]. Note that, the score for each entity that may be a cluster or a singleton is defined as

$$\mathbf{s}(e_i) = \begin{cases} \lambda_{e_i,max} - \lambda_{e_i,min} & \text{if } e_i \text{ is a cluster} \\ \lambda_{m_i} & \text{otherwise} \end{cases} \tag{3.34}$$

where $\lambda_{e_i,max}$ and $\lambda_{e_i,min}$ are the maximum and minimum value of λ for the mentions in cluster $c(e_i)$. If the entity e_i is a singleton, its score is a mention score. Thus, the cluster hierarchy can be well formulated to rank entity clusters, of which entity importance is decreasing while traversing the tree from its root to leaves.

3.7 Models

To validate the effectiveness of the proposed frameworks and approaches, three models are constructed to solve the entity extraction and ranking.

3.7.1 Baseline Model

The baseline model is a phrase embedding model, which leverages the constituency parsing to obtain candidate mentions set $M(D)$ for document D. The embedding

formula is indicated in 3.16, which sum each token embedding in the phrase. Afterward, these candidate mentions are used to construct the cluster graph $\mathcal{G}_{cluster}$ and run cluster algorithms. The model architecture is shown in figure 3.4.

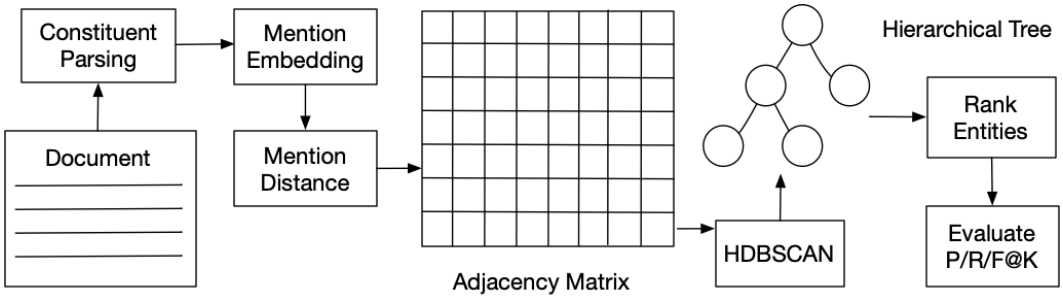

Figure 3.4: Baseline model architecture.

3.7.2 Coreference Model

The second model is a coreference-aided model, which not only leverages the constituency parsing to obtain candidate mentions in a document but also employs the coreference resolution to normalize the mention embedding. The embedding normalization formula is indicated in 3.17, which averages two mention embeddings, including the original mention embedding and the mean of mention embeddings in a cluster. Afterward, these mentions embeddings are used to compute the euclidean distance and construct the adjacency matrix for graph $\mathcal{G}_{cluster}$, which is then fit to the clustering. The framework for the coreference model is shown in Figure 3.5.

3.7.3 Convolutional Model

The third model is a convolutional model, which leverages the graph convolution to moderate candidate mentions embeddings. In particular, the graph convolution components utilize the coreference resolution and dependency parsing to represent the mention relations within sentences and between sentences respectively. The convolution operation aims to smooth the node embeddings and relatively smooth the

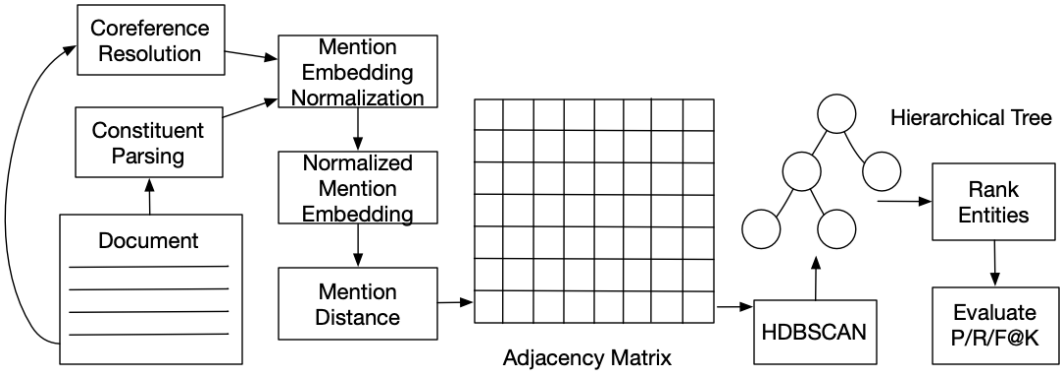

Figure 3.5: Model with conreference architecture.

graph. Normally, a 1-order convolution leverages the embedding information in the first-order neighbors of a node through a linear mapping to update the embeddings of that node. Higher-order convolution considers second- or higher-order neighbors of that node. The graph convolution formula is indicated in 3.25 and 3.27. Afterward, a cluster graph $\mathcal{G}_{cluster}$ is constructed, and the hierarchy is employed to rank the entity clusters. The structure of the convolutional model is shown in Figure 3.6

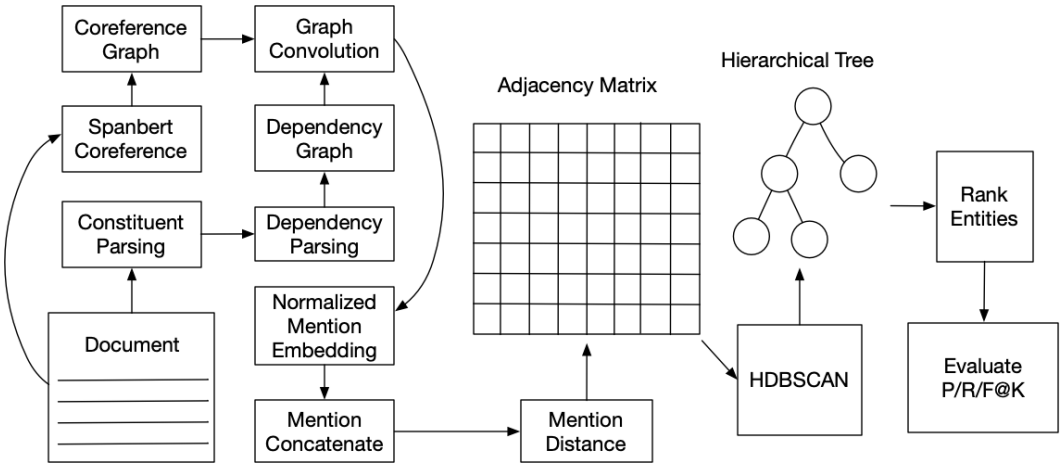

Figure 3.6: Model with dependency parsing and coreference architecture.

Chapter 4

Experiments

This chapter introduces the experiments of the proposed models, and exams the effectiveness of these models using NELA2017 dataset [26]. The implementation of this research is mainly based on Pytorch[1], TensorFlow[2], SpaCy[3], and NLTK[4] in Python 3.6+.

4.1 Experimental Setup

PyTorch is a deep learning framework that provides libraries to implement the SpanBERT coreference as well as BERT Embeddings. The work of [29] and [14] are employed to develop TensorFlow-based coreference resolution and mention embedding components respectively. As PyTorch relies on tensor-based computation that is incorporated with Graphics Processing Units (GPUs), all the vectors and matrices are necessarily converted to tensor formats to perform efficient computation on GPUs.

SpaCy is a Python open-source framework for NLP research and NLP industry

[1]https://pytorch.org/
[2]http://tensorflow.org/
[3]https://spacy.io/
[4]https://www.nltk.org/

production. It supplies implementations for data preprocessing, named entity recognition, part-of-speech tagging, and so on. Threfore, SpaCy is leveraged to build constituency parsing and dependency parsing trees in this thesis.

The Natural Language Toolkit (NLTK) is another package used to process text data. It has similar functions compared to SpaCy and also provides various interfaces such as tokenization, POS tagging and so on. For the implementation, NLTK is used to split sentences in given articles.

4.2 Data Exploration

The dataset used in this thesis is the NEws LAndscape (NELA2017) [26]. It presents a large political news dataset with over 136K articles in 92 sources from April 2017 to October 2017. Compared to the specific events or topics of news in other event-based news datasets, NELA2017 instead incorporates all political news from 92 sources, which perfectly matches the needs of entity extraction and ranking regardless of events and topics in the documents.

As unsupervised learning for these models is performed in this thesis, the volume of the dataset rarely affects the model performance. So a sample of 60 articles is used to exam the effectiveness of proposed models by comparing their outputs with gold entities. The gold entities are manually annotated by annotators with respect to top 3, top 5 and top 10. The number of annotated entities in the sampled articles is shown in Figure 4.1.

In general, the average number of annotated entities is 16.24; however, there are 8 articles, in which the total annotated entities are less than 10. For the articles with total entities less than 10, the top 10 entities are the whole number of entities annotated in the articles. Meanwhile, the top 5 entities are the whole number of entities annotated in the articles if the whole number is less than 5. The articles with

human-annotated entities less than 3 are dropped. Moreover, each entity consists of several tokens. The average number of tokens in an entity is 1.79. As shown in Figure 4.2, the longest entity has 6 tokens, and most of the entities consist of 1 or 2 tokens.

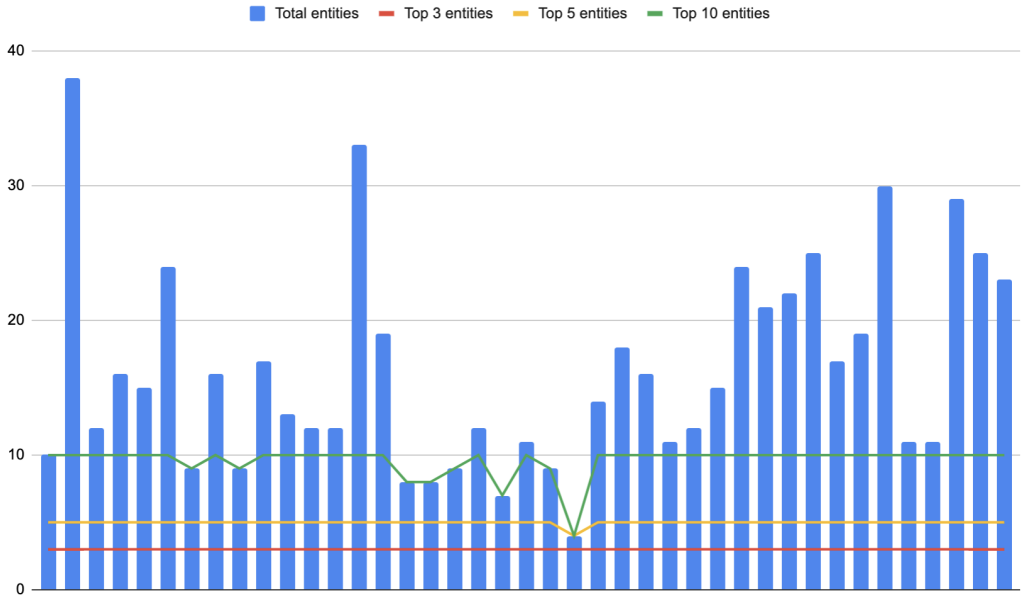

Figure 4.1: Number of entities in sampled articles

The x-axis is the sampled article and y-axis is the number of entities in the articles

4.3 Evaluation Metrics

The accuracy assessment for entity extraction involves the comparison between entity candidates extracted from systems and gold entity annotated by annotators with respect to precision, recall, and F1-score. Traditionally, entity extraction systems can be evaluated by the number of top-N candidates that exactly match corresponding gold entities. In particular, inexact matches, or near-misses, have been considered.

In evaluation, the inexact matches between candidates and gold entities are allowed if most of the tokens matching within the entities. In other words, the gold entity is matched with candidate entity if at least half of the tokens in the gold entities are matched. In particular, the outputs of the extraction system are entity

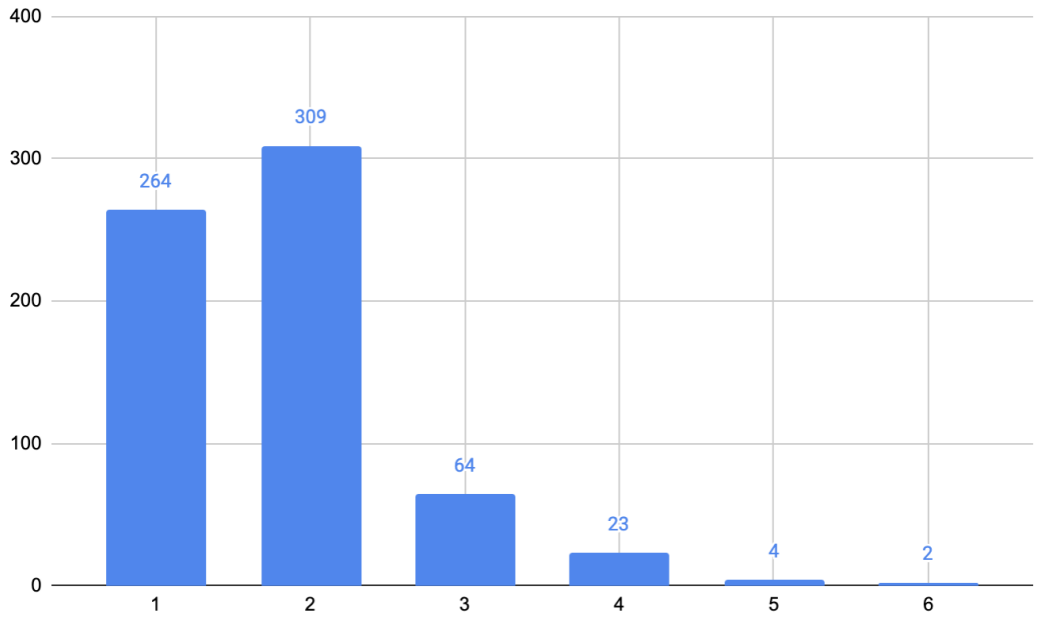

Figure 4.2: The entity length in the articles

The x-axis represents the entity length category and y-axis is the number of the entity length

clusters but the gold entities are phrases, so the evaluation of the matches between the entity cluster is challenging. The criteria for matching between entity cluster and entity phrases is that the gold entity phrase needs to match at least a half number of the entity mentions in the candidate entity cluster. All the evaluation is calculated as micro-averaged precision, recall and F-score regarding the top 5, 10 and 15 candidates.

In classification experiments, precision is a measure of the class matches between the labels in gold data and the labels given by the classifier. In experiments, the micro precision for top K is measured as the matching agreement between candidate entity and gold entities instead. Thus, the formula is

$$P_{\text{micro}}@K = \frac{\bar{TP}}{\bar{TP} + \bar{FP}} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + \sum_{i=1}^{n} FP_i} \tag{4.1}$$

where the TP_i is the number of matchings in top K, and FP_i is the number of non-

matchings in top K. Recall measures the effectiveness of a classifier in identifying positive labels, which can be calculated as the number of matchings (TP_i) in top K dividing the total number of correctly matched entities generated through the system. The micro recall is formulated as

$$R_{\text{micro}}\ @K = \frac{\bar{TP}}{\bar{TP} + \bar{FN}} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + \sum_{i=1}^{n} FN_i} \tag{4.2}$$

where the FN_i is the number of incorrectly ordered entities after top K. The micro F1-score is a particularization of the F-Measure, which is a combination of micro precision and micro recall and is formulated as follows

$$F_{\text{micro}}@K = \frac{2 \times P_{\text{micro}} \times R_{\text{micro}}}{P_{\text{micro}} + R_{\text{micro}}} \tag{4.3}$$

All the evaluation is based on the experiments with the NALA2017 dataset with respect to sampled articles. The inputs are raw texts and the outputs are ranked entity clusters, of which there can be clusters with one mention, representing singletons, and cluster with multiple mentions, representing entity clusters.

4.4 Model Evaluation

As discussed in the previous session, the evaluation metrics include the micro precision (P@K), micro recall (R@K), and micro F1-score (F1@K) for top 3, top5, and top10 candidates respectively. The precision was obtained based on the proportion of matched entities in the total entities generated through the models. The recall was obtained based on the proportion of matched entities in the annotated golden entities. The F score performs a trade-off between precision and recall.

From the results presented in Table 4.1, the conclusion that models proposed in this thesis surpass the baseline on NELA2017 dataset can be made. The coreference-

Table 4.1: Evaluation results for proposed models

		P_{micro}	R_{micro}	$F1_{micro}$
Baseline	Top@3	0.382	0.127	0.191
	Top@5	0.393	0.218	0.280
	Top@10	0.311	0.345	0.327
Coreference	Top@3	**0.521**	**0.179**	**0.266**
	Top@5	0.480	0.275	0.350
	Top@10	0.396	0.454	0.423
Graph Covolution	Top@3	0.438	0.153	0.227
	Top@5	**0.481**	**0.280**	**0.354**
	Top@10	**0.463**	**0.538**	**0.498**

based model has the most satisfactory results for Top 3 entities as it directly leverages the phrase embeddings in the cluster, of which each phrase has coreference relations. It is important to notice that, the coreference resolution is significant to improve the entity clustering as well. The graph-based model also archived satisfactory performance as it aims to normalize the node information from its k-order neighbor, which could verify the assumption that entity nodes did share information with their neighbors on dependency and coreference graphs.

On the one hand, the top 10 has relatively higher precision compared to the top 3 and top 5 in terms of F1 scores, which indicates that proposed frameworks have the potential to extract entities. On the other hand, the recall of these models maintains increasing trends in all the three models for top 3, top 5, and top 10, which indicates that these models could generate fewer noises with the number of the candidates increased. In general, the top 10 candidates achieved considerably higher results compared to the top 3 and top 10. This finding suggests that the top 10 candidates not only improve the coverage of the candidate entities but also maintain considerable accuracy.

Moreover, the graph-based model slightly improves the performance of coreference model in terms of F1 score for the top 5 and top 10, as it leverages both the coreference resolution normalization and the dependency parsing trees. This finding can verify

the effectiveness of the contextualized embeddings and graph-based convolutions for entity extraction and ranking. Besides, the precision of the graph-based model is higher than coreference model in terms of top 5 and top 10 but lower than the coreference model in terms of top 3, so it may conclude that graph-based model is more sensitive to the top key entities; however, the coreference model may be more sensitive to the coverage of the entities. In general, the pipelined framework that integrates the contextualized embeddings, the coreference resolution, graph convolutional, and the density-based clusters, has demonstrated its effectiveness on this task, which further indicates that the joint entity extraction and entity ranking could to some degree demonstrate the success of the unsupervised learning in this application.

Chapter 5

Analysis

In this thesis, pipelined approaches are developed to extract entities from given texts. The proposed models involve the contextualized embeddings, dependency parsing and constituency parsing, coreference resolution, graph convolution, and density-based clustering. The evaluation of these proposed models have demonstrated their effectiveness and achieved satisfactory results on NELA 2017 dataset. This chapter further exams the proposed models in terms of error analysis.

Given texts, the unsupervised frameworks can directly generate ranked entities without any external resources or any model training, which are competitive to learnable models with multiple deep layers. The case analysis is based on the following article[1]:

> *"WASHINGTON NFL Sunday kicked off in London with more kneeling during the national anthem, as President Trump continues to admonish players who don't stand for the flag. Three Miami Dolphins players were spotted taking a knee on the sideline during the singing of the national anthem: Kenny Stills, Michael Thomas, and Julius Thomas. The three kneeled side-by-side at one of the sidelines while the rest of their team*

[1]Number 51167 article in NELA2017 dataset

stood for the anthem. Meanwhile, their opposing team, the New Orleans Saints, took a unified knee on the sidelines but rose in time for the national anthem. Many of Saints linked arms as Grammy-winning artist Darius Rucker sang the anthem in Wembley Stadium, the first televised game of the day on Fox. The cameras didn't zoom in on the Dolphins kneelers, but attendees and journalists in London quickly posted pictures of the trio of kneelers on social media. President Trump has blasted a kneeling player as a son of a bitch and urged owners to fire athletes who don't stand up for the flag"

In this research, the use of constituency parsing is sufficient to generate a list of entity mentions, which contains both singletons and phrases candidates. The contextualized BERT embeddings for each phrase are shown in Figure 5.1.

As seen in Figure 5.1, the contextualized embeddings map each noun phrase extracted from constituency parsing to space where related phrases remain close to each other. For example, the two *President Trump* phrases are overlapped as they refer to the same entity, but the related phrases *a knee* and *kneelers* are yet considerably far from each other. Besides, the phrase *a keen* and *kneelers* also keep distance with each other and the phrase *the flag* and *the anthem* are even further even if they may have close relations based on knowledge. Therefore, the contextualized embeddings can somewhat map a similar phrase to a close area but fail to precisely recognize similar terms in different sentences.

To solve this problem, the entity coreference resolution is employed to leverage inter-sentence relations for embeddings. For example, as shown in figure 5.2, the marked phrase *the flag* and *the anthem*, *a keen* and *kneelers* are in close positions, which indicates that the normalized embeddings based on coreference resolution can potentially encode phases based on their semantics and contexts. However, there are a few outliers such as marked *fire athletes* and *more kneeling*, which introduce noises

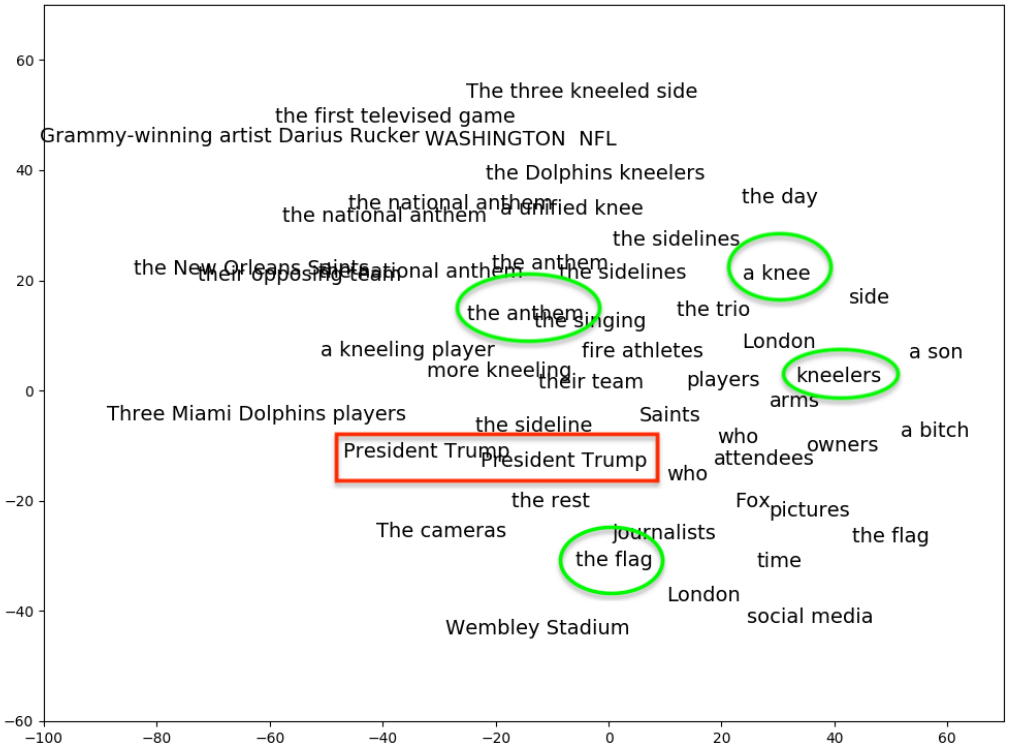

Figure 5.1: Phrase embeddings based on BERT

The x- and y-axis represent the distance of phrases in BERT embeddings

to the embedding space. Therefore, coreference-based embeddings have potentials to improve.

Figure 5.3 shows the density of the cluster graph defined in 3.29. Generally, these graphs show density clusters with sparse spectrums; therefore, performing density-based clustering on these graphs is effective. Moreover, the upper left subplot represents the cluster graph with only contextualized embeddings, of which density areas are not significantly separated by the sparse domains. This caused that HDBSCAN algorithm makes fewer clusters. In the upper right graph, the density and spars areas are well identified. In other words, the cluster algorithm achieves considerable performance, which on the other hand demonstrates the effectiveness of the coreference resolution model. Meanwhile, the lower-left subplot shows that the cluster graph constructed by using normalized phrase embedding with graph-convolution can further

Figure 5.2: Phrase embeddings based on BERT and Coreference

The x- and y-axis represent the distance of phrases in BERT embeddings with coreference normalizations

smooth the node distribution as the graph convolution aims to update the mention embedding by conducting a convolutional operation on its neighbor nodes, which consequentially differentiates the nodes embeddings. The lower right subplot shows the graph constructed by conducting graph convolution based on phrase embeddings with coreference. Notably, the convolution-based cluster graph has a similar distribution to the graph that leverages both graph convolution and entity coreference resolution. Therefore, the phrase embeddings based on coreference resolution do not make many benefits to graph convolutions. These findings also demonstrate the generalization of the graph convolutions that do not heavily rely on pre-coreferenced entities.

The ranked entities of these documents are shown in Table 5.1. The entities are represented as a list of entity mentions, and each of them (in the table cell) is considered as a cluster that contains corresponding entity mentions. The entities are ranked

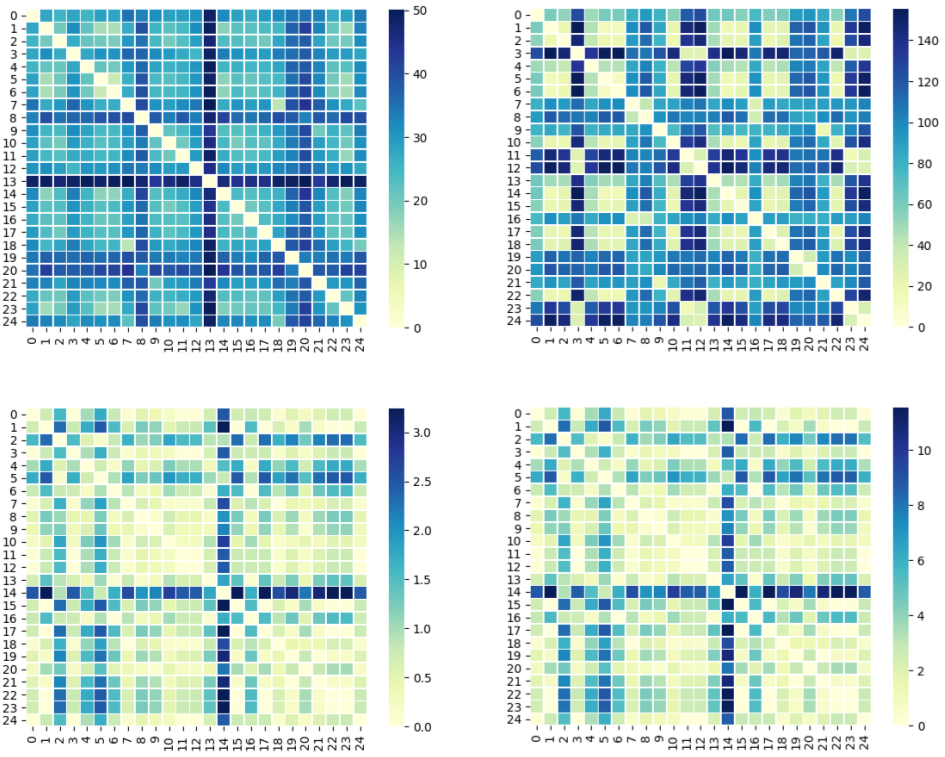

Figure 5.3: Density graph based on BERT embeddings, coreference resolution, graph convolution, and graph convolution with normalizations, respectively

from top 1 to top 10. As for the baseline model with contextualized embeddings, the clustered entities have several noises. For example, the *Three Miami Dolphins players* and *New Orleans Saints* are opposing teams that cannot belong to a cluster. On the other hand, the top 2 of the baseline model can handle the *the national anthem* showing in different sentences, which once again shows the effectiveness of contextualized embeddings for entity representation. Moreover, throwing *the flag* and *the anthem* into a cluster is yet reasonable. The significant entities are extracted and ranked with acceptable orders. For the coreference resolution model, the top 1 entity cluster is related to the national anthem, of which *time* and *Grammy-winning artist Darius Rucker* are irrelevant; however, the second entity is perfectly generated as *the New Orleans Saints* is exactly the opposing team of *Three Miami Dolphins players* shown in the fourth entity cluster. This finding demonstrates that coreference relations play

a significant role in improving clusterings. Even the fourth and seventh contain many noises, the rest of the entity clusters have considerable accuracy. The graph-based convolution model shows that the entity clusters have fewer noises compared to the coreference model, but have the limitation of the ranking for the top 3 and top 5 entities. For the fifth entity, it exactly extracts *President Trump* as an entity cluster that could be also important. As noticed, the graph convolution model is not sensitive to the entity coreference, therefore, it does not have rich entity mentions extracted from different sentences for an entity cluster; however, this may reduce errors and noises with relatively higher recall as shown in Table 4.1.

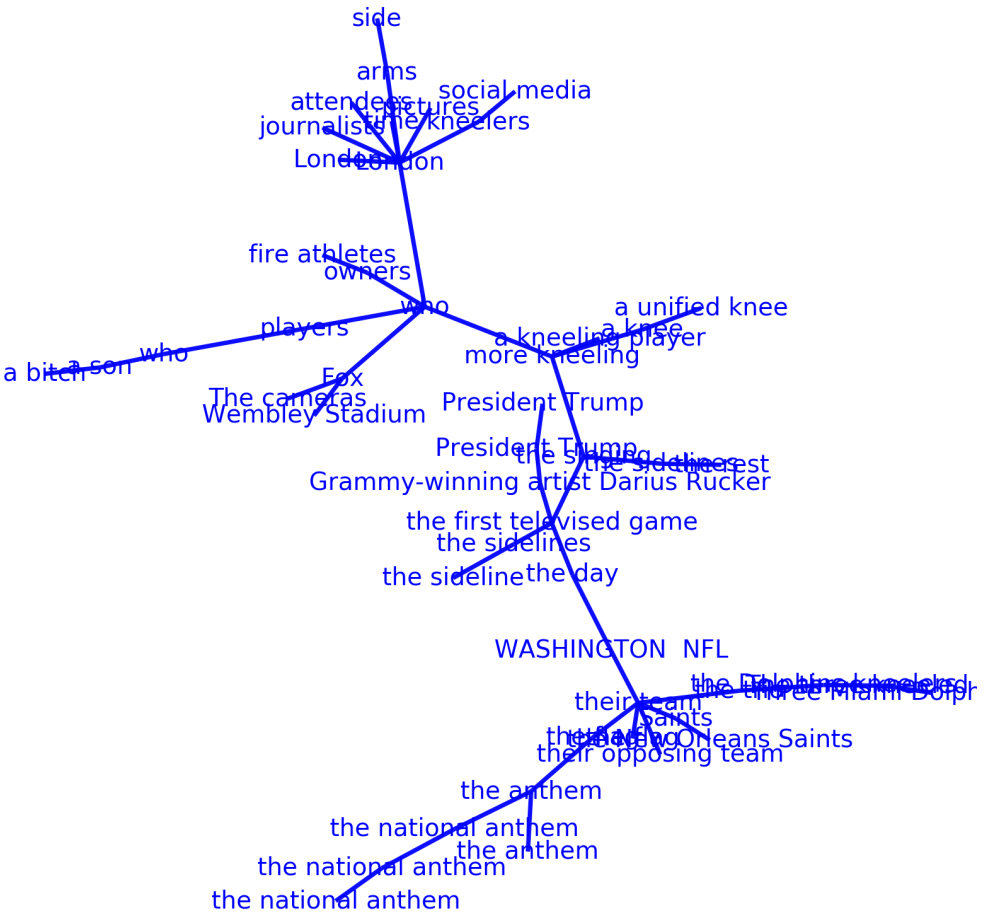

Figure 5.4: Minimum spanning tree of a cluster graph

Besides, Figure 5.4 shows the minimum spanning tree (MST) of the cluster graph,

Table 5.1: The results of ranked entities for three models

Rank	Baseline Model	Coreference Resolution Model	Graph Convolution Model
1	Three Miami Dolphins players, the New Orleans Saints	the national anthem, the singing, the national anthem, time, the national anthem, Grammy-winning artist Darius Rucker	side, fire athletes
2	the national anthem, the national anthem	their opposing team, the New Orleans Saints	more kneeling, players
3	the flag, the anthem, the national anthem	a knee, a unified knee	their opposing team, a unified knee
4	a knee, a unified knee	the flag, Three Miami Dolphins players, the rest, pictures, the trio	WASHINGTON NFL, the national anthem, the singing, the national anthem, the national anthem, The cameras
5	the sideline, the sidelines	the sideline, the sidelines	President Trump, President Trump
6	players, who	players, who	the flag, the anthem, the anthem, the flag
7	Saints, arms, Grammy-winning artist Darius Rucker, the anthem, Wembley Stadium, the first televised game	Saints, arms, the anthem, Wembley Stadium, the day, Fox, The cameras, the Dolphins kneelers, attendees, journalists, London, kneelers, President Trump, a kneeling player, a son, a bitch, owners, fire athletes, who, the flag	who, the rest, who
8	The three kneeled side	The three kneeled side	the sidelines, their team, the sidelines
9	their opposing team	WASHINGTON NFL	attendees, pictures
10	WASHINGTON NFL	the first televised game	the sideline, ox, the Dolphins kneelers, social media

seeing formula 3.29, which is a direct interpretation of the clusters and edge weights. Inspired by the hierarchical clustering algorithm, this MST can be converted to a hierarchy by performing single linkages [42], which is also the essential step in HDBSCAN algorithm. This hierarchy tree can be further condensed to obtain the clustering hierarchy with ranked entities [8], showing in Figure 5.5, in which the hierarchy tree represents the clusters in a ranked manner. In particular, each node denotes a cluster, and the X node denotes a failed cluster that has many noises, referring Table 5.1. From this tree, node *The National Anthem* and node *The New Orleans Saints* are the direct children of *Root*, which share the significance of the entities. Consequently, nodes on the lower levels share less important information.

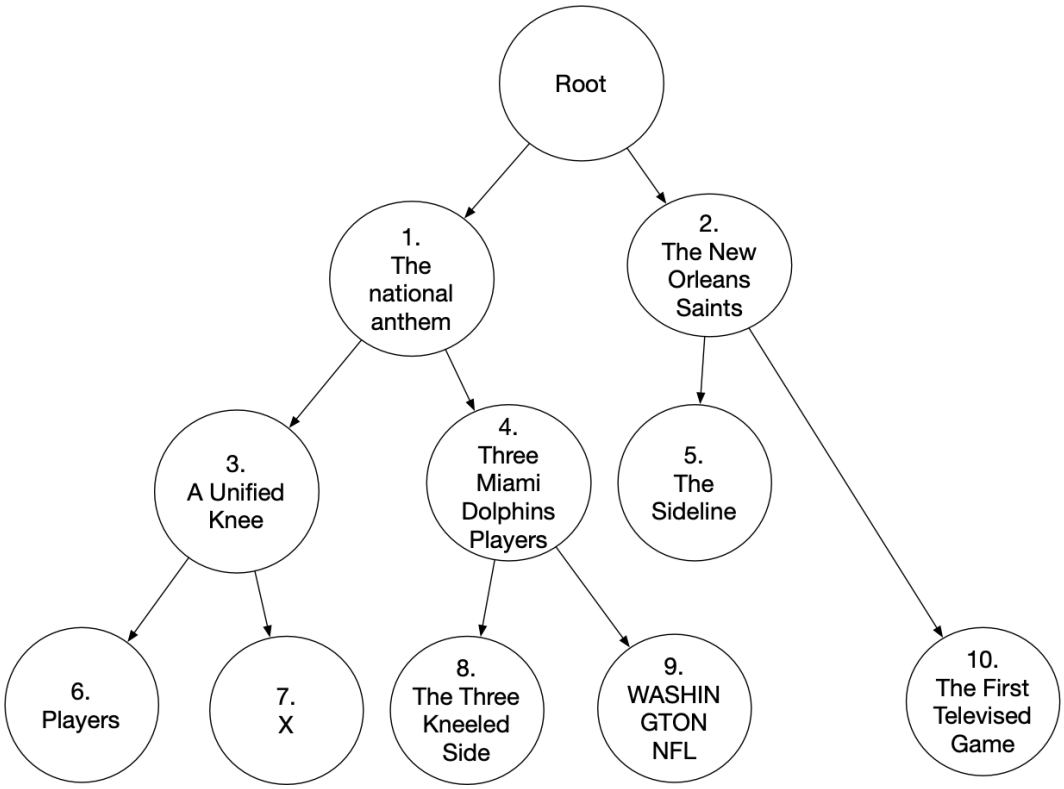

Figure 5.5: The hierarchy of a cluster graph

In conclusion, this thesis performs unsupervised frameworks to handle entity extraction and entity ranking problem. The parsing, entity coreference, and graph convolution jointly benefit the phrase normalization, resulting in a sparse-dense clus-

tering graph. Thus, the hierarchical density-based cluster algorithm can be applied to further cluster the graph nodes, along with the auxiliary entity ranking. This work not only demonstrates the effectiveness of proposed models but also further evaluates the successes of unsupervised learning for NLP application. The further work will be focusing on leveraging NLP parser to construct a more meaningful graph that could be used to update the node embeddings as well as directly training span-based model to improve mention representation will be another interesting direction. In addition, improving the hierarchical density-based cluster algorithm will be another exploration.

Bibliography

[1] Alan Akbik, Tanja Bergmann, and Roland Vollgraf. Pooled contextualized embeddings for named entity recognition. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 724–728, 2019.

[2] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey. *arXiv preprint arXiv:1901.09069*, 2019.

[3] Gabriel Bernier-Colborne. Identifying semantic relations in a specialized corpus through distributional analysis of a cooccurrence tensor. In *Proceedings of the Third Joint Conference on Lexical and Computational Semantics (* SEM 2014)*, pages 57–62, 2014.

[4] Parminder Bhatia, E Busra Celikkaya, and Mohammed Khalilia. End-to-end joint entity extraction and negation detection for clinical text. In *International Workshop on Health Intelligence*, pages 139–148. Springer, 2019.

[5] Ann Bies, Mark Ferguson, Karen Katz, Robert MacIntyre, Victoria Tredinnick, Grace Kim, Mary Ann Marcinkiewicz, and Britta Schasberger. Bracketing guidelines for treebank ii style penn treebank project. *University of Pennsylvania*, 97: 100, 1995.

[6] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[7] Jose Camacho-Collados and Mohammad Taher Pilehvar. From word to sense embeddings: A survey on vector representations of meaning. *J. Artif. Int. Res.*, 63(1):743–788, September 2018. ISSN 1076-9757. doi: 10.1613/jair.1.11259. URL `https://doi.org/10.1613/jair.1.11259`.

[8] Ricardo JGB Campello, Davoud Moulavi, and Jörg Sander. Density-based clustering based on hierarchical density estimates. In *Pacific-Asia conference on knowledge discovery and data mining*, pages 160–172. Springer, 2013.

[9] Eugene Charniak. Statistical parsing with a context-free grammar and word statistics. *AAAI/IAAI*, 2005(598-603):18, 1997.

[10] Jing Chen, Chenyan Xiong, and Jamie Callan. An empirical study of learning to rank for entity search. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 737–740, 2016.

[11] Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. A walk-based model on entity graphs for relation extraction. *arXiv preprint arXiv:1902.07023*, 2019.

[12] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[13] Gianluca Demartini, Tereza Iofciu, and Arjen P De Vries. Overview of the inex 2009 entity ranking track. In *International Workshop of the Initiative for the Evaluation of XML Retrieval*, pages 254–264. Springer, 2009.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[15] Jesse Dunietz and Dan Gillick. A new entity salience task with millions of training examples. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 205–209, 2014.

[16] Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S Weld, and Alexander Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artificial intelligence*, 165(1):91–134, 2005.

[17] Yuntian Feng, Hongjun Zhang, Wenning Hao, and Gang Chen. Joint extraction of entities and relations using reinforcement learning and deep learning. *Computational intelligence and neuroscience*, 2017, 2017.

[18] Kavita Ganesan and Chengxiang Zhai. Opinion-based entity ranking. *Information retrieval*, 15(2):116–150, 2012.

[19] Palash Goyal and Emilio Ferrara. Graph embedding techniques, applications, and performance: A survey. *Knowledge-Based Systems*, 151:78–94, 2018.

[20] Hannaneh Hajishirzi, Leila Zilles, Daniel S Weld, and Luke Zettlemoyer. Joint coreference resolution and named-entity linking with multi-pass sieves. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 289–299, 2013.

[21] Kazi Saidul Hasan and Vincent Ng. Automatic keyphrase extraction: A survey of the state of the art. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1262–1273, 2014.

[22] Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. Discovering relations among named entities from large corpora. In *Proceedings of the 42nd annual meeting on association for computational linguistics*, page 415. Association for Computational Linguistics, 2004.

[23] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[24] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[25] Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 782–792. Association for Computational Linguistics, 2011.

[26] Benjamin Horne. NELA2017, 2019. URL `https://doi.org/10.7910/DVN/ZCXSKG`.

[27] Alpa Jain and Marco Pennacchiotti. Open entity extraction from web search query logs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, page 510–518, USA, 2010. Association for Computational Linguistics.

[28] Xin Jiang, Yunhua Hu, and Hang Li. A ranking approach to keyphrase extraction. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 756–757, 2009.

[29] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *arXiv preprint arXiv:1907.10529*, 2019.

[30] Arzoo Katiyar and Claire Cardie. Nested named entity recognition revisited. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 861–871, 2018.

[31] Steven Kearnes, Kevin McCloskey, Marc Berndl, Vijay Pande, and Patrick Riley. Molecular graph convolutions: moving beyond fingerprints. *Journal of computer-aided molecular design*, 30(8):595–608, 2016.

[32] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[33] Rémi Lebret and Ronan Collobert. Word emdeddings through hellinger pca. *arXiv preprint arXiv:1312.5542*, 2013.

[34] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end neural coreference resolution. *arXiv preprint arXiv:1707.07045*, 2017.

[35] Kenton Lee, Luheng He, and Luke Zettlemoyer. Higher-order coreference resolution with coarse-to-fine inference. *arXiv preprint arXiv:1804.05392*, 2018.

[36] Qimai Li, Xiao-Ming Wu, Han Liu, Xiaotong Zhang, and Zhichao Guan. Label efficient semi-supervised learning via graph filtering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9582–9591, 2019.

[37] Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 366–376. Association for Computational Linguistics, 2010.

[38] Peter Lofgren, Siddhartha Banerjee, and Ashish Goel. Personalized pagerank estimation and search: A bidirectional approach. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, pages 163–172, 2016.

[39] Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. A general framework for information extraction using dynamic span graphs. *arXiv preprint arXiv:1904.03296*, 2019.

[40] Kevin Lund and Curt Burgess. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior research methods, instruments, & computers*, 28 (2):203–208, 1996.

[41] Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: The penn treebank. 1993.

[42] Fionn Murtagh and Pedro Contreras. Algorithms for hierarchical clustering: an overview. *WIREs Data Mining and Knowledge Discovery*, 2(1):86–97, 2012. doi: 10.1002/widm.53. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.53`.

[43] Vincent Ng and Claire Cardie. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 104–111. Association for Computational Linguistics, 2002.

[44] Thien Huu Nguyen and Ralph Grishman. Graph convolutional networks with argument-aware pooling for event detection. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[45] Xiaodong Ning, Lina Yao, Boualem Benatallah, Yihong Zhang, Quan Z Sheng, and Salil S Kanhere. Source-aware crisis-relevant tweet identification and key

information summarization. *ACM Transactions on Internet Technology (TOIT)*, 19(3):1–20, 2019.

[46] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[47] Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. Cross-sentence n-ary relation extraction with graph lstms. *Transactions of the Association for Computational Linguistics*, 5:101–115, 2017.

[48] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[49] 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. URL `http://arxiv.org/abs/1802.05365`.

[50] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[51] Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40. Association for Computational Linguistics, 2012.

[52] Douglas LT Rohde, Laura M Gonnerman, and David C Plaut. An improved method for deriving word meaning from lexical co-occurrence. *Cognitive Psychology*, 7:573–605, 2004.

[53] Sunil Kumar Sahu, Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. Inter-sentence relation extraction with document-level graph convolutional neural network. *arXiv preprint arXiv:1906.04684*, 2019.

[54] Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. Parsing with compositional vector grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 455–465, 2013.

[55] Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, 27(4):521–544, 2001.

[56] Peter D Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37:141–188, 2010.

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[58] Alex D Wade, Kuansan Wang, Yizhou Sun, and Antonio Gulli. Wsdm cup 2016: Entity ranking challenge. In *Proceedings of the ninth ACM international conference on web search and data mining*, pages 593–594, 2016.

[59] Chengyu Wang, Guomin Zhou, Xiaofeng He, and Aoying Zhou. Nerank+: a graph-based approach for entity ranking in document collections. *Frontiers of Computer Science*, 12(3):504–517, 2018.

[60] Yi-fang Brook Wu, Quanzhi Li, Razvan Stefan Bot, and Xin Chen. Domain-specific keyphrase extraction. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 283–284, 2005.

[61] Chenyan Xiong, Russell Power, and Jamie Callan. Explicit semantic ranking for academic search via knowledge graph embedding. In *Proceedings of the 26th international conference on world wide web*, pages 1271–1279, 2017.

[62] Ying Xiong, Yedan Shen, Yuanhang Huang, Shuai Chen, Buzhou Tang, Xiaolong Wang, Qingcai Chen, Jun Yan, and Yi Zhou. A deep learning-based system for pharmaconer. In *Proceedings of The 5th Workshop on BioNLP Open Shared Tasks*, pages 33–37, 2019.

[63] Vikas Yadav and Steven Bethard. A survey on recent advances in named entity recognition from deep learning models, 2019.

[64] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*, 2014.

[65] Meishan Zhang, Yue Zhang, and Guohong Fu. End-to-end neural relation extraction with global optimization. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1730–1740, 2017.

[66] Yuhao Zhang, Peng Qi, and Christopher D Manning. Graph convolution over pruned dependency trees improves relation extraction. *arXiv preprint arXiv:1809.10185*, 2018.