MINING ENTITY AND RELATION STRUCTURES FROM TEXT: AN EFFORT-LIGHT APPROACH

BY

XIANG REN

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Doctoral Committee:

Professor Jiawei Han, Chair & Director of Research
Professor Tarek F. Abdelzaher
Professor Heng Ji
Professor ChengXiang Zhai

ABSTRACT

In today's computerized and information-based society, text data is rich but often also "messy". We are inundated with vast amounts of text data, written in different genres (from grammatical news articles and scientific papers to noisy social media posts), covering topics in various domains (e.g., medical records, corporate reports, and legal acts). Can computational systems automatically identify various real-world entities mentioned in a new corpus and use them to summarize recent news events reliably? Can computational systems capture and represent different relations between biomedical entities from massive and rapidly emerging life science literature? How might computational systems represent the factual information contained in a collection of medical reports to support answering detailed queries or running data mining tasks?

While people can easily access the documents in a gigantic collection with the help of data management systems, they struggle to gain insights from such a large volume of text data: document understanding calls for in-depth content analysis, content analysis itself may require domain-specific knowledge, and over a large corpus, a complete read and analysis by domain experts will invariably be subjective, time-consuming and relatively costly. To turn such massive, unstructured text corpora into machine-readable knowledge, one of the grand challenges is to gain an understanding of the typed entity and relation structures in the corpus. This thesis focuses on developing principled and scalable methods for extracting typed entities and relationship with light human annotation efforts, to overcome the barriers in dealing with text corpora of various domains, genres and languages. In addition to our *effort-light methodologies*, we also contribute effective, noise-robust *models* and real-world *applications* in two main problems:

- **Identifying Typed Entities**: We show how to perform data-driven text segmentation to *recognize* entities mentioned in text as well as their surrounding relational phrases, and infer types for entity mentions by *propagating* "distant supervision" (from external knowledge bases) via relational phrases. In order to resolve data sparsity issue during propagation, we complement the type propagation with *clustering* of functionally similar relational phrases based on their redundant occurrences in large corpus. Apart from entity recognition and coarse-grained typing, we claim that fine-grained entity typing is beneficial for many downstream applications and very challenging due to the context-agnostic label assignment in distant supervision, and we present principled, efficient models and algorithms for inferring fine-grained type path for entity

mention based on the sentence context.

- **Extracting Typed Entity Relationships**: We extend the idea of entity recognition and typing to extract relationships between entity mentions and infer their relation types. We show how to effectively model the noisy distant supervision for relationship extraction, and how to avoid the error propagation usually happened in incremental extraction pipeline by integrating typing of entities and relationships in a principled framework. The proposed approach leverages noisy distant supervision for both entities and relationships, and simultaneously learn to uncover the most confident labels as well as modeling the semantic similarity between true labels and text features.

In practice, text data is often highly variable: corpora from different domains, genres or languages have typically required for effective processing a wide range of language resources (e.g., grammars, vocabularies, and gazetteers). The massive and messy nature of text data poses significant challenges to creating tools for automated extraction of entity and relation structures that scale with text volume. State-of-the-art information extraction systems have relied on large amounts of task-specific labeled data (e.g., annotating terrorist attack-related entities in web forum posts written in Arabic), to construct machine-learning models (e.g., deep neural networks). However, even though domain experts can manually create high-quality training data for specific tasks as needed, both the scale and efficiency of such a manual process are limited. This thesis harnesses the power of "big text data" and focuses on creating generic solutions for efficient construction of customized machine-learning models for mining typed entities and relationships, relying on only limited amounts of (or even no) task-specific training data. The approaches developed in the thesis are thus general and applicable to all kinds of text corpora in different natural languages, enabling quick deployment of data mining applications. We provide scalable algorithmic approaches that leverage external knowledge bases as sources of supervision and exploit data redundancy in massive text corpora, and we show how to use them in large-scale, real-world applications, including structured exploration and analysis of life sciences literature, extracting document facets from technical documents, document summarization, entity attribute discovery, and open-domain information extraction.

To my wonderful parents, Yanhong Wang and Yongkui Ren, for their love and support.

ACKNOWLEDGMENTS

I am greatly indebted to my Ph.D. advisor, Jiawei Han, one of the kindest and nicest people I have ever met. His passion and dedication on research has deeply influence me. Even during weekend nights, he spent hours remotely to give me advice on my research projects and plans, and to teach me how to do research, write scientific papers and manage all sorts of things in my Ph.D. career. He spent his sabbatical on campus, working closely with every student and providing detailed advices. Over the past five years, Jiawei did not just prepare me for every step of the Ph.D. degree, but also for an independent academic career by involving me in a comprehensive set of academic activities including grant proposal writing, paper reviewing, PI meetings, guest lectures, student mentoring, and following up with many constructive and timely feedback and advice. He has also been extremely supportive and open-mined, always encouraging me to explore various new problems, to think from a bigger picture, and to learn from other people's work. Moreover, Jiawei has been always fair, giving credit and generous; he always supported our trips to different conferences (even when we only got short papers or demo papers accepted there). During my last year at UIUC, he spent lots of efforts on my job search, by providing timely feedbacks on my application documents and job talk slides, giving me tips during the whole application process, and connecting me to relevant folks at different places. Even until the final version of this thesis, Jiawei has been providing insightful comments and suggestions. No matter how much I write in this note, it is impossible to express all my gratitude to him.

I would also like to say thank you to all the other thesis committee members, Tarek F. Abdelzaher, Heng Ji and Cheng-Xiang Zhai, for giving me valuable feedback and asking insightful questions during my thesis proposal and defense. I would like to particularly thank Cheng for spending a lot of time to give me detailed comments on this thesis, help improve the content, and also provide lots of tips and advices on my job search process. I am grateful to Heng and Tarek for not only sitting on my dissertation committee, but also advising me through my job application process, answering a lot of questions, and suggesting interesting future directions for my work.

I have had the awesome opportunity to work with numerous great mentors during summer internships. I went back to Beijing, China in my first summer and spent the summer at Microsoft Research Asia (where I stayed for over a year during my college time). I got the chances to know new and industry-scale problems, and learned different ways of thinking about solution and styles of doing research. My second and third summers were at Microsoft

Research Redmond, working with the data management group and ISRC group respectively. I want to thank Tao Cheng, Bolin Ding, Yuanhua Lv, Kuan-san Wang, Yang Song and Hao Ma, for their mentorship and guidance during my internship.

I really appreciate the time working with the amazing peers in the Data Mining group at UIUC. I thank all my friends in the group: Jialu Liu (thank you for all the discussions we had on many research ideas as well as views on future directions of our research), Xiao Yu (I still remember we started collaborating on recommender system projects right after my first project ended, we spent lots of time together doing experiments and writing papers; it turns out working quite well and we got some nice work done before you graduated), Fangbo Tao (it is great to have you in the group that you are actively connecting folks together and bring joy to the group), Hyungsul Kim (thank you for being my very first student mentor and guiding me on how to do research in my first year). I would also like to thank all the master students and junior Ph.D. students that I am honored to work with, including Ellen Wu, Wenqi He, Meng Qu, Qi Zhu, Hongkun Yu, Urvashi Khandelwal, Bradley Sturt and Tobias Lei. All of you did an amazing job on the collaboration projects and I am very grateful to enjoy the research together with you.

Grad school was not just about work. I was fortunate to have wonderful friends around me who made the experience more fun, and less daunting: Gong Chen and Annlin Lee, Ray Wang, Qi Wang, De Liao, Wenzhu Tong; my early officemates, Aston Zhang, Hongwei Wang and Huan Gui. Special thanks to my closest friend and partner, Han Zheng, for being patient and supporting me, and for making future career decisions with me.

Last but not least, many thanks are due to the most amazing parents, who have been extremely supportive during the whole process. They have always been there ready to answer my video calls to hear my news, provide encouragement and advice, celebrate my successes, help me recover fast after failures, and cheer me up.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

The success of data mining technology is largely attributed to the efficient and effective analysis of structured data. However, the majority of existing data generated in our computerized society is unstructured or loosely-structured, and is typically "text-heavy". People are soaked with vast amounts of natural-language text data, ranging from news articles, social media posts, online advertisements, scientific publications, to a wide range of textual information from various domains (*e.g.*, medical notes and corporate reports). Big data leads to big opportunities to uncover structures of real-world entities (*e.g.*, `person`, `company`, `product`) and relations (*e.g.*, `employee_of`, `manufacture`) from massive text corpora. Can machines automatically identify person, organization and location entities in a news corpus and use them to summarize recent news events (Fig. 1.1)? Can we mine different relations between proteins, drugs and diseases from massive and rapidly emerging life science literature? How would one represent entity and relation structures hidden in a collection of medical reports to support answering precise queries or running data mining tasks?

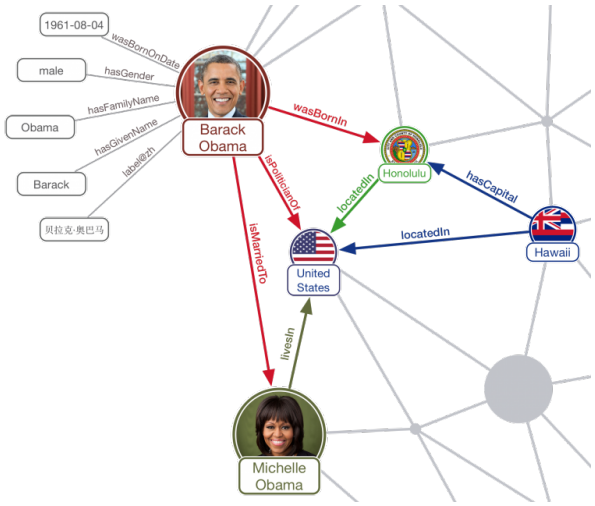


Figure 1.1: An illustration of entity and relation structures extracted from some text data. The nodes correspond to entities and the links represent their relationships.

While accessing documents in a gigantic collection is no longer a hard thing with the help of data management systems, people, especially those who are not domain experts, struggle to gain insights from such a large volume of text data: document understanding calls for in-depth content analysis, content analysis itself may require domain-specific knowledge, and over a large corpus, a complete read and analysis by domain experts will invariably be subjective, time-consuming and relatively costly. Moreover, text data is highly variable: corpora from different domains, genres or languages have typically required for effective processing a wide range of language resources (*e.g.*, grammars, vocabularies, gazetteers). The "massive" and "messy" nature of text data poses significant challenges to creating tools for automated processing and algorithmic analysis of content that scale with text volume.

This thesis develops principled and scalable methods for the mining of typed entity and

relation structures from unstructured text corpora, with a focus on overcoming the barriers in dealing with text corpora of various domains, genres and languages. State-of-the-art information extraction (IE) approaches have relied on large amounts of task-specific labeled data (*e.g.*, annotating terrorist attack-related entities in web forum posts written in Arabic), to construct machine-learning models (*e.g.*, deep neural networks). However, even though domain experts can manually create high-quality training data for specific tasks as needed, both the *scale* and *efficiency* of such a manual process are limited. My thesis research harnesses the power of "big text data" and focuses on creating generic solutions for *efficient construction of customized machine-learning models for factual structure extraction*, relying on only limited amounts of (or even no) task-specific training data.

The main directions of our work are: (1) *entity recognition and typing*, which automatically identifies token spans of real-world entities of interests from text and classifies them into a set of coarse-grained entity types; (2) *fine-grained entity typing*, which assigns the most appropriate type path in a given type hierarchy to entity mention based on their sentence context; and (3) *relation extraction*, which determine what kind of relations is expressed between two entities based on the sentences where they co-occur. We provide scalable algorithmic approaches that leverage external knowledge bases as sources of supervision and exploit data redundancy in massive text corpora, and we show how to use them in large-scale, real-world applications, including structured exploration and analysis of life sciences literature, extracting document facets from technical documents, document summarization, entity attribute discovery, and open-domain information extraction.

1.1 OVERVIEW AND CONTRIBUTIONS

This thesis studies how to automate the process of extracting typed entity and relation structures from a large corpus *with light human efforts* (*i.e.*, **Effort-Light StructMine**), that is, with no task-specific manual annotation on the corpus. In contrast to existing knowledge base population approaches (*e.g.*, Google Knowledge Vault [1], NELL [2], KnowItAll [3], DeepDive [4]) that harvests facts incrementally from the whole Web to cover common knowledge in the world, my approach aim to generate a structured (typed) view of all the entities and their relationships in a given corpus, to enable semantic, holistic and fast analysis of all content in the full corpus. Thus the extraction of a corpus-specific entity/relation structures is distinct from, but also complements the task of knowledge base population. As a result, the application of effort-light StructMine techniques for extracting entity and relation structures focuses on establishing only corpus-specific factual knowledge (e.g. identifying the entities and relations disambiguated for that corpus), a task that is outside the scope of

general knowledge bases or graphs.

Challenges. We have witnessed the great success of machine-learning approaches in yielding state-of-the-art performance on information extraction when abundant amounts of training data are available. In contrast to rule-based systems, supervised learning-based systems shift the human expertise in customizing systems away from the complex handcrafting of extraction rules to the annotation of training data and feature engineering. The resulting effectiveness of supervised learning systems largely depends on the amount of available annotated training data and complexity of the task. When the quantity of annotated data is limited and the complexity of the task is high, these factors become bottlenecks in extracting corpus-specific entity/relation structures. Recent advances in bootstrapping pattern learning (*e.g.*, NELL [2], KnowItAll [3], OpenIE [5]) aim to reduce the amount of human involvement—only an initial set of annotated examples/patterns is required from domain experts, to iteratively produce more patterns and examples for the task. Such a process, however, still needs manual spot-checking of system intermediate output on a regular basis to avoid error propagation, and suffers from low coverage on "implicit relations", *i.e.*, those that are not overtly expressed in the corpus and so fail to match textual patterns generated by the systems.

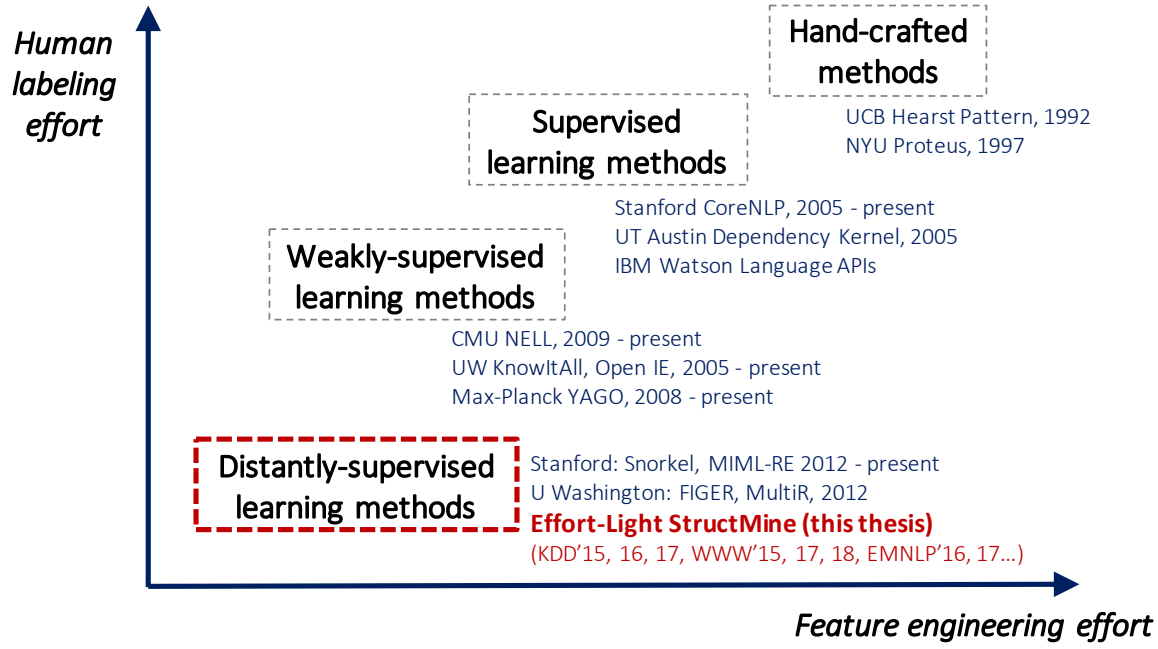


Figure 1.2: Overview of the related work. Our method, effort-light StructMine, has relied on lightest efforts on human laboring and feature engineering when compared with prior arts.

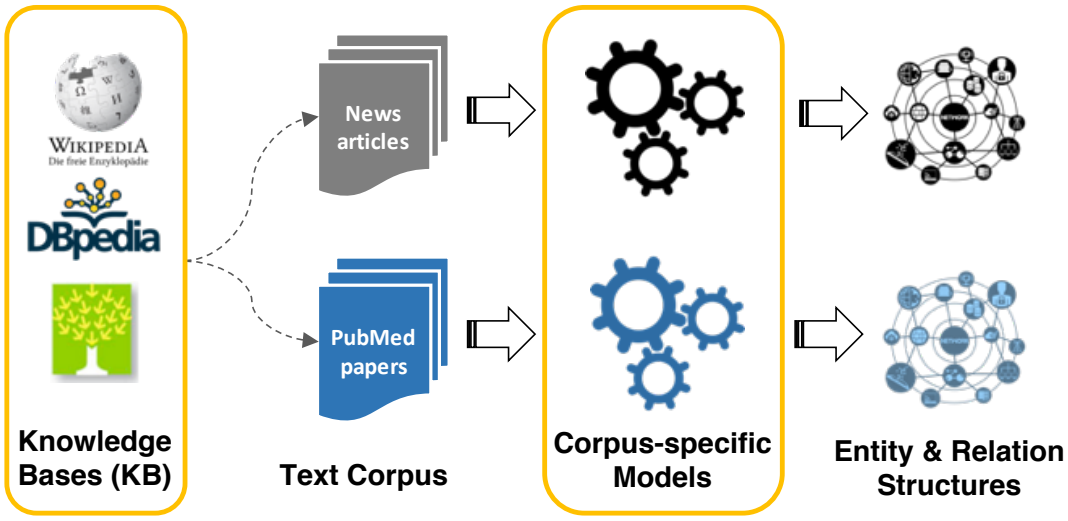


Figure 1.3: Illustration of the proposed framework. Effort-light StructMine leverages existing structures stored in external knowledge bases to automatically generate large amounts of corpus-specific, potentially noisy training data, and builds corpus-specific models for extracting entity and relation structures.

Proposed Framework. Our solution to effort-light StructMine aims to bridge the gap between customized machine-learning models and the absence of high-quality task-specific training data. It leverages the information overlap between background facts stored in external knowledge bases (KBs) (*e.g.*, Freebase [6], BioPortal [7]) and the given corpus to automatically generate large amounts of (possibly noisy) task-specific training data; and it exploits redundant text information within the massive corpus to reduce the complexity of feature generation (*e.g.*, sentence parsing). This solution is based on two key intuitions which are described below.

First, in a massive corpus, structured information about some of the entities (*e.g.*, entity types, relationships to other entities) can be found in external KBs. Can we align the corpus with external KBs to automatically generate training data for extracting entity and relation structures at a large scale? Such retrieved information supports the automated annotation of entities and relations in text and labeling of their categories, yielding (possibly noisy) corpus-specific training data (Figure 1.3). Although the overlaps between external KBs and the corpus at hand might involve only a small proportion of the corpus, the scale of the automatically labeled training data could still be much larger than that of manually annotated data by domain experts.

Second, text units (*e.g.*, word, phrase) co-occur frequently with entities and other text units in a massive corpus. Can we exploit the textual co-occurrence patterns to characterize the semantics of text units, entities, and entity relations? For example, having observed

that "*government*", "*speech*", "*party*" co-occur frequently with `politician` entities in the training data, the next time these text units occur together with an unseen entity in a sentence, the algorithm can more confidently guess that entity is a `politician`. As such patterns become more apparent in a massive corpus with rich data redundancy, big text data leads to big opportunities in representing semantics of text unit without complex feature generation.

To systematically model the intuitions above, effort-light StructMine approaches the structure extraction tasks as follows: (1) annotate the text corpus automatically with target factual structure instances (*e.g.*, entity names, entity categories, relationships) by referring to external KBs, to create a task-specific training data (*i.e.*, distant supervision); (2) extract shallow text units (*e.g.*, words, n-grams, word shapes) surrounding the annotated instances in local context; (3) learn semantic vector representations for target instances, text units, and their category labels based on distant supervision and corpus-level co-occurrence statistics, through solving joint optimization problems; and (4) apply learned semantic vector representations to extract new factual instances in the remaining part of the corpus. The resulting framework, which integrate these ideas, has minimal reliance on human efforts, and thus can be ported to solve structure extraction tasks on text corpora of different kinds (*i.e.*, **domain-independent**, **language-independent**, **genre-independent**).

The thesis is organized into two main parts: (1) identifying typed entities, and (2) extracting entity relationships. We summarize the main problems of each part in the form of questions in Table 1.1.

Part	**Research Problem**	**Chapter**
I: Identifying Typed Entities	**Entity Recognition and Typing:** How can we identify token spans of real-world entities and their types from text?	4
	Fine-grained Entity Typing: How can we assign fine-grained types to mentions of entities in text?	5
II: Extracting Typed Entity Relationships	**Joint Extraction of Entities and Relations:** What types of entities are mentioned in text and what typed of relationships are expressed between them?	6

Table 1.1: Thesis Organization

1.1.1 Part I: Identifying Typed Entities

Real-world entities are important factual structures that can be identified from text to represent the factual knowledge embedded in massive amounts of text documents, and can server as fundamental building blocks for many downstream data mining and natural language processing tasks such as knowledge base construction and recommender systems. At a macroscopic level, how can we extract entities of types of interests from text with minimal reliance on labeled training data? At a microscopic level, how can we determine the fine-grained categories of an entity based on the context where it occurs? Our work proposes effective and distantly-supervised methods for recognizing and typing entities from text by leveraging external knowledge bases and exploiting rich data redundancy in large corpora. With the entities and their coarse-grained types extracted, we further look into how to assign more fine-grained entity types given the context and noisy distant supervision.

Entity Recognition and Typing

How can we identify token spans of real-world entities and their categories from text?

One of the most important factual structures in text is entity. Recognizing entities from text and labeling their types (*e.g.*, `person`, `location`) enables effectives structured analysis of unstructured text corpora (Chapter 4). Traditional named entity recognition (NER) systems are usually designed for several major types and general domains, and so require additional steps for adaptation to a new domain and new types. Our method, ClusType [8], aims at identifying typed entities of interests from text without task-specific human supervision. While most existing NER methods treat the problem as sequence tagging task and require significant amounts of manually labeled sentences (with typed entities), ClusType makes use of entity information stored in freely-available knowledge bases to create large amounts of (yet potentially noisy) labeled data and infers types of other entities mentioned in text in a robust and efficient way.

We formalize the entity recognition and typing task as a distantly-supervised learning problem. The solution workflow is: (1) detect entity mentions from a corpus; (2) map candidate entity mentions to KB entities of target types; and (3) use those confidently mapped {mention, type} pairs as labeled data to infer the types of remaining candidate mentions. ClusType runs data-driven phrase mining to generate entity mention candidates and relation phrases (thus having no reliance on pre-trained name recognizer), and enforces the principle that relation phrases should be softly clustered when propagating type information between

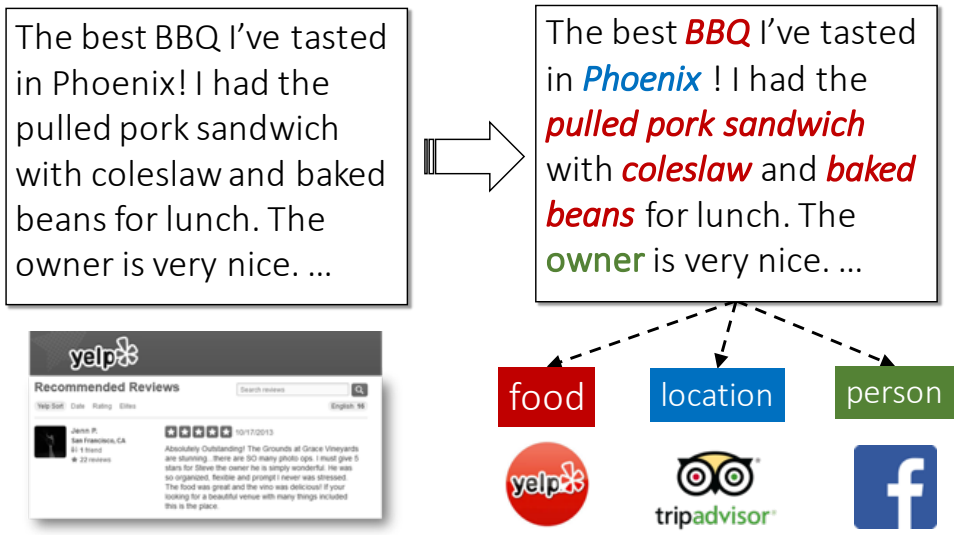


Figure 1.4: An example for illustration of entity recognition and typing.

their argument entities. We formulate a joint optimization to integrate type propagation via relation phrases and clustering of relation phrases.

Contributions:

- *Problem*: We study the problem of distantly-supervised entity recognition and typing in a domain-specific corpus, where only a corpus and a reference knowledge base are given as input.
- *Methodology*: We develop an efficient, domain-independent phrase segmentation algorithm for extracting entity mentions and relation phrases. Entity types can be estimated for entity mentions by solving the clustering-integrated type propagation.
- *Effectiveness on real-world corpora*: Our experiments on three datasets of different genres – news, reviews and tweets – demonstrate that ClusType achieves significant improvement over the state-of-the-art.

Impact:

- ClusType system was transferred to US Army Research Lab, Microsoft and National Institute of Health.
- It was taught in a graduate class at University of Illinois at Urbana-Champaign, covered as a major part of the keynote at ACL 2015, and presented in conference tutorials at SIGKDD, WWW and SIGMOD [9, 10].

Fine-grained Entity Typing

How can we assign fine-grained types to mentions of entities in text?

ClusType provides a data-driven way to identify typed entities from text with the helps from external knowledge bases. It is able to distinguish types of entities at a coarse-grained level (*e.g.*, `location` versus `organization`) based on the context surrounding the entity mention. However, many downstream applications will benefit if one can assign type to an entity mention from a much larger set of fine-grained types (for example, over 100) from a tree-structured hierarchy. Fine-grained entity typing allows one entity mention to have multiple types, which together constitute a type path in the given type hierarchy, depending on the local context. This task require in-depth modeling of local context and thus becoming very challenging for relation phrase-based method like ClusType.

How can we build models to automatically estimate the fine-grained type path for entity mention, without heavy reliance on human supervision? This is the problem we address in Chapter 5. When external knowledge bases are available for generating fine-grained labels, a key issue with such distant supervision is that it assign types to entity mentions in a *context-agnostic* way – one entity (*e.g.*, *Barack Obama*) can have multiple entity types (*e.g.*, `person`, `politician`, `writer`) in KB but within a specific context only some of these types may describe the entity properly. While prior arts follow supervised learning methods to train typing models on noisy distant supervision, we propose noise-aware embedding approaches [11, 12] to *denoise* the set of labels given by distant supervision as we are learning the embeddings for text features. Such weakly-supervised learning setting yields model the

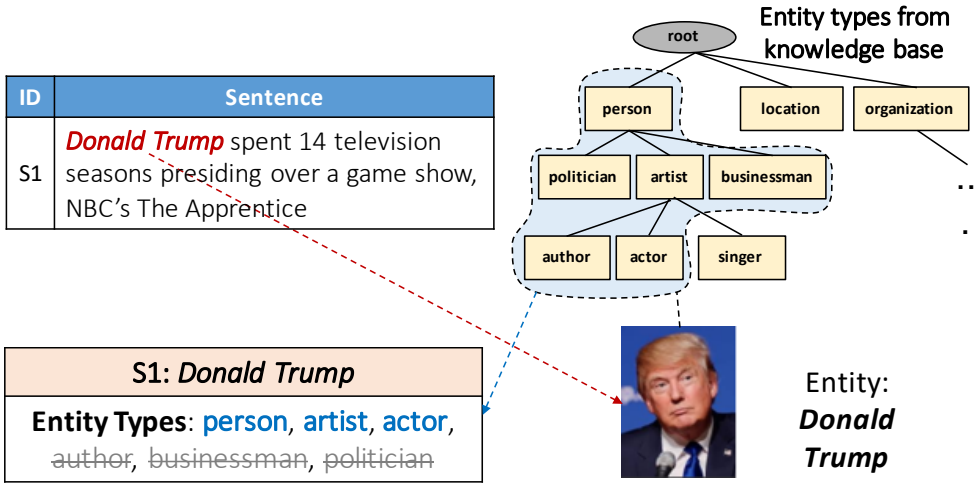


Figure 1.5: An illustration of the fine-grained entity typing problem.

correlation between "true" labels and features in a more reliable way and thus can produce more effective feature embeddings for prediction.

Contributions:

- *Problem*: The first systematic study of noisy labels in distant supervision for entity typing problem.
- *Methodology 1*: An embedding-based framework, PLE [11], to model and measure semantic similarity between text features and type labels, and is robust to noisy labels.
- *Methodology 2*: A novel rank-based optimization problem is formulated to model noisy type label and type correlation [12].
- *Effectiveness on real data*: The proposed methods achieve significant improvement over the state of the art on multiple fine-grained typing datasets, and demonstrate the effectiveness on recovering true labels from the noisy label set.

Impact:

- It was taught in a graduate class at University of Illinois at Urbana-Champaign, and presented in conference tutorials at WWW and SIGMOD [9, 10].

1.1.2 Part II: Extracting Typed Entity Relationships

Our studies on entity structure mining (Part I) provide the basic building blocks of entity relationships, *i.e.*, the entity arguments mentioned in text. To further structure the text corpus the next step is to identify typed relationships between entities based on the local context in sentences (*i.e.*, relation extraction). Identifying typed relationships is key to structuring content from text corpora for fa wide range of downstream applications. For example, when an extraction system finds a "`produce`" relation between "`company`" and "`product`" entities in news articles, it supports answering questions like "*what products does company X produce?*". Once extracted, such relationship information is used in many ways, *e.g.*, as primitives in knowledge base population and question-answering systems. Traditional relation extraction systems have relied on human-annotated corpora for training supervised learning models. Here we ask: how can we design a domain-independent relation extraction system that can apply to text corpora from different domains in the absence of human-annotated, domain training data? To address this question, we propose a distant-supervised relation extraction method in Chapter 6, which is able to reference existing relationship information stored in external KBs as a source of supervision and integrates the extraction models for both entities and relationships.

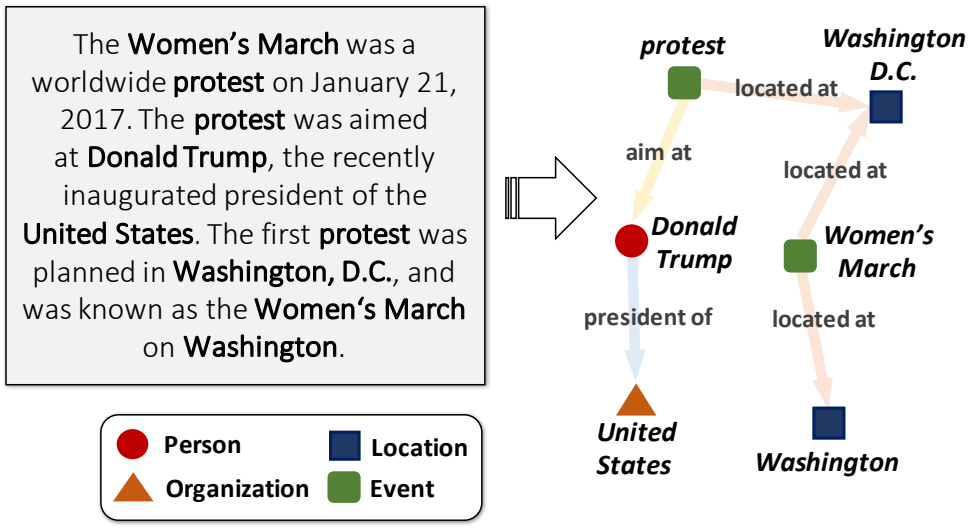


Figure 1.6: An illustration of the joint entity and relationship extraction problem.

Joint Extraction of Entities and Relationships

What types of entities are mentioned in text and what typed of relationships are expressed between them?

With facts about entities, their types and the relationships between them stored in external knowledge bases, one can automatically generate large amounts of (potentially noisy) labeled training data for building entity recognition models (Part I) and relation extraction models. Prior arts focusing on this task have two major limitations: (1) partition the relation extraction process into several subtasks and solve them incrementally (thus there are errors propagating cascading down the pipeline); and (2) ignore the noises brought in during the automatic label generation process and directly train machine learning models over the noisy labeled data.

To overcome these challenges, we study the problem of joint extraction of typed entities and relationships with knowledge bases. Given a domain-specific corpus and an external knowledge base, we aim to detect relation mentions together with their entity arguments from text, and categorize each in context by relation types of interests. Our method, CoType [13], approaches the joint extraction task as follows: (1) it designs a domain-agnostic text segmentation algorithm to detect candidate entity mentions with distant supervision (*i.e.*, minimal linguistic assumption); (2) models the mutual constraints between the types of relation mentions and the types their entity arguments to enable feedbacks between the two subtasks; (3) model the true type labels in a candidate type set as latent variables and require only the most confident type to be relevant to the mention. CoType achieved the

state-of-the-art relation extraction performance under distant supervision, and demonstrates robust domain-independence across various datasets.

Contributions:

- *Methodology*: We propose a novel distant supervision framework, CoType, to extract typed entities and relationships in domain-specific corpora with minimal linguistic assumption.
- *Effectiveness on real data*: Experiments with three public datasets demonstrate that CoType improves the performance of state-of-the-art systems of entity typing and relation extraction significantly, demonstrating robust domain-independence.
- It was taught in a graduate class at University of Illinois at Urbana-Champaign and presented in conference tutorials at WWW, CIKM and SIGMOD.
- *Software and demo system*: We open sourced the CoType software and datasets used in our experiment, and further developed a demo system to facilitate exploring and analyzing over the typed entity and relation structures extracted using CoType from PubMed papers.

Impact:

- The work was selected as 2017 SIGMOD Research Competition finalist.
- The open-sourced software on GitHub receives over 100 subscriptions from users.
- The typed entities and relationships extracted from 21 millions PubMed publications was adopted by researchers at Stanford Medical School to facilitate the development of drug re-purposing and adverse drug event discovery techniques.

1.2 OVERALL IMPACT

The core of the thesis focuses on developing effective, human effort-light and scalable methods for extracting typed entities and relationships from massive, domain-specific text corpora. Our contributions are in the area of text mining and information extraction, within which we focus on domain-independent and noise-robust approaches using distant supervision (in conjunction with publicly-available knowledge bases). The work has broad impact on a variety of applications: knowledge base construction, question-answering systems, structured search and exploration of text data, recommender systems, network analysis, and many other text mining tasks. Finally, our work has been used in the following settings:

- **Used in real world**:
  - Our entity recognition and typing technique (ClusType [14]) has been transferred to U.S. Army Research Lab, Microsoft Bing Ads and NIH Big Data to Knowledge Center to identify typed entities of different kinds from low-resource, domain-specific text corpora. ClusType is also used by Stanford sociologists to identify scientific concepts from 37 millions of scientific publications in Web of Science database to study innovation and translation of scientific ideas.
  - A biomedical knowledge graph (*i.e.*, Life-iNet [15]) constructed automatically from millions of PubMed publications using our effort-light StructMine pipeline is used by researchers at Stanford Medical school to facilitate drug re-purposing. It yields significant improvement of performance on new drugs and rare diseases.
  - Our effort-light StructMine techniques (ClusType, PLE, CoType) is adopted by veterinarians at Veterinary Information Network Inc. (VIN) to construct the first veterinary knowledge graph from multiple sources of information including research articles, books, guidelines, drug handbooks and message board posts.
- **Taught in classes and conference tutorials**: Our methods on entity recognition and typing (ClusType), fine-grained entity typing (PLE [11], AFET [12]), and relation extraction (CoType [13]) are being taught in graduate courses, *e.g.*, University of Illinois at Urbana-Champaign (CS 512), and are introduced as major parts of the conference tutorial in top data mining and database conferences such as SIGKDD, WWW, CIKM and SIGMOD.
- **Awards**: The thesis work on effort-light StructMine has been awarded a Google PhD fellowship in 2016 (sole winner in the category of Structured Data and Data Management in the world) and a Yahoo!-DAIS Research Excellence Award, and a C. W. Gear Outstanding Graduate Student Award from University of Illinois.

Next, we present background in mining structured factual information and introduce useful factual structure notions and definitions.

CHAPTER 2: BACKGROUND

In this chapter we introduce the key definitions and notions in information extraction and knowledge graph construction that are useful for understanding the methods and algorithms described in this thesis. At the end of this chapter we give a table with the common notations and their descriptions.

2.1 ENTITY STRUCTURES

We start with the definition of common text structures, followed by entity, and other entity-related concepts (*e.g.*, entity mention, entity types).

Phrase: A phrase is a group of words (or possibly a single word) that functions as a constituent in the syntax of a sentence, a single unit within a grammatical hierarchy. Phrase acts as a semantic unit in a sentence with some special idiomatic meaning or other significance (*e.g.*, "*machine learning*", "*watch TV*", "*before that happened*", "*too slowly*").

Noun Phrase: A noun phrase (or nominal phrase) is a phrase which has a noun as its head (*i.e.*, the word that determines the syntactic category of that phrase). It usually consists of groups made up of nouns – a person, place, thing or idea – and the modifiers such as determiners, adjectives, conjunctions. When looking at the structure of language, we treat a noun phrase the same way we treat a common noun. Like all nouns, a noun phrase can be a subject, object, or complement in a sentence.

Example 2.1 (Noun Phrase) *In the sentence "The quick, brown fox jumped over the lazy dog", there are two noun phrases: "the quick, brown fox" and "the lazy dog". "the quick, brown fox" is the subject of the sentence and "the lazy dog" is the object.*

Proper Name: A proper name is a noun phrase that in its primary application refers to a unique entity (*e.g.*, *University of Southern California*, *Computer Science*, *United States*), as distinguished from a common noun which usually refers to a class of entities (*e.g.*, *city*, *person*, *company*), or non-unique instances of a specific class (*e.g.*, *this city*, *other people*, *our company*). When a noun refers to a unique entity, it is also called proper noun.

Entity: In information extraction and text mining, an *entity* (or *named entity*) is a real-world object, such as persons, locations, organizations, products, scientific concepts, etc., that can be *denoted* with a proper name. It can be abstract or have a physical existence.

Examples of named entities include *Barack Obama*, *Chicago*, *University of Illinois*. An entity is denoted as e in this thesis.

Remark 2.1: Ambiguous Proper Names for Named Entity

In the expression of "named entity", the word *named* restricts the scope to those entities for which one or many strings stands consistently for some referent. In practice, one named entity may be referred by many proper names and one proper name may refer to multiple named entity. For example, the entity, *automotive company created by Henry Ford in 1903*, can be referred to using proper names "*Ford*" or "*Ford Motor Company*", although "*Ford*" can refer to many other entities as well.

Entity Mention: An entity mention, denoted by m, is a token span (*i.e.*, a sequence of words) in text that refers to a named entity. It consists of the proper name and the token index in the sentence.

Example 2.2 (Entity Mention) *In the sentences "I had the **pulled pork sandwich** with **coleslaw** and **baked beans** for lunch. The **pulled pork sandwich** is the best Ive tasted in **Phoenix**!", the entity mentions are bold-faced. The proper name "pulled pork sandwich" appear twice in the sentence, corresponding to the same named entity but different entity mentions (thus will have different entity mention IDs).*

Entity Type: An entity type (or *entity class*, *entity category*), is a conceptual label for a collection of entities that share the same characteristics and attributes (*e.g.*, `person`, `artist`, `singer`, `location`). Entities with the same entity types are similar to one another. Entity type instances refer to entities that are assigned with the specific entity type. In many applications, a set of entity types of interests are usually pre-specified by domain experts via providing example entity type instances. There also exists cases that entity types are related to each other (vs. mutually exclusive), forming a complex, tree-structured type hierarchy.

Example 2.3 (Entity Types in ACE Shared Task) *The Automatic Content Extraction (ACE) Program [16] was to develop information extraction technology to support automatic processing of natural language data. In the Entity Detection and Tracking (EDT) task of ACE, it focuses on seven types of entities:* `Person`*,* `Organization`*,* `Location`*,* `Facility`*,* `Weapon`*,* `Vehicle`*, and* `Geo-Political Entity`*. Each type was further divided into subtypes (for instance,* `Organization` *subtypes include* `Government`*,* `Commercial`*,* `Education`*,* `Non-profit`*,* `Other`*).*

2.2 RELATION STRUCTURES

This section introduces the basic concepts on relations. We start with the definition of relation, followed by definitions of relation instance and mention.

Relation: a *relation* (or *relation type*, *relation class*), denoted as r, is a (pre-defined) predication about two or more entities. For example, from the sentence fragment "*Facebook co-found Mark Zuckerberg*" one can extract the `FounderOf` relation between entities *Mark Zuckerberg* and *Facebook*. In this thesis, we focus on binary relations, that is, relations between two entities.

Example 2.4 (Relations in ACE Shared Task) *Much of the prior work on extracting relations from text is based on the task definition from ACE program [16]. A set of major relation types and their subtypes are defined by ACE. Examples of ACE major relation types include* ***`physical`*** *(an entity is physically near another entity),* ***`personal/social`*** *(a person is a family member of another person),* ***`employment/affiliation`*** *(a person is employed by an organization).*

Relation Instance: A relation instance denotes a relationship over two or more entities in a specific relation. When only considering binary relations, a relation instance can be represented as a triple with a pair of entities e_i and e_j, and their relation type r, *i.e.*, (e_i, r, e_j). For example,

Entity Argument: The two entities involved in a relation instance are referred to as entity arguments. The former one is also referred to as *head entity* while the later is also referred to as *tail entity*.

Relation Mention: A relation mention, z, denotes a specific occurrence of some relation instance in text. It records the two entity mentions for the pair of entity arguments, the relation type between these two entities, and the sentence s where the relation mention is found, *i.e.*, $z = (m_i, r, m_j; s)$.

Example 2.5 (Relation Mention) *Suppose we are given two sentences: "Obama was born in Hawaii, USA" (s_1) and "Barack Obama, the president of United States" (s_2). There are two relation mentions between entities Barack Obama and United States: z_1 = (Obama,* ***`BirthPlace`****, USA; s_1) and z_2 = (Barack Obama,* ***`PresidentOf`****, United States; s_2). Although the entity arguments are the same, the two relation mentions have different relation types based on the sentence context.*

2.3 DISTANT SUPERVISION FROM KNOWLEDGE BASES

Knowledge bases are emerging as a popular and useful way to represent and leverage codified knowledge for a variety of use cases. For example, codifying the key entities and relationships of a particular domain can greatly accelerate a variety of tasks from providing semantic and natural language search over more traditional business intelligence data, to providing enabling query expansion and matching, to discovery and exploration of related entities and relations extracted from a large corpus of unstructured documents.

In information extraction, the term "*knowledge base*" is most frequently used with large collections of curated, structured knowledge, such as WikiData [17], Freebase [6], YAGO [18], DBpedia [19] or CYC [20]. It is also applied when such structured knowledge is automatically extracted, such as NELL [2], SnowBall [21] or OpenIE [5]. We use it to refer to any collection of relation triples, no matter their source or underlying ontology, if any. This can include subject-verb-object triples automatically extracted from large text corpora or curated from domain experts.

Fact: A fact in knowledge base (KB) can refer to either a binary relation triple in the form of (e_i, r, e_j) or an is-A relation between an entity and a concept, such as *Facebook* is a `company`.

Formally, A knowledge base, Ψ, consists of a set of entities $\mathcal{E}_\Psi$ and curated facts on both relation instances $\mathcal{I}_\Psi$ and entity types $\mathcal{T}_\Psi$ (*i.e.*, is-A relation between entities and their entity types). The set of relation types in the KB is denoted as $\mathcal{R}_\Psi$.

Example 2.6 (Freebase) *Curating a universal knowledge graph is an endeavor which is infeasible for most individuals and organizations. Thus, distributing that effort on as many shoulders as possible through crowdsourcing is a way taken by Freebase [6], a public, editable knowledge graph with schema templates for most kinds of possible entities (i.e., persons, cities, movies, etc.). After MetaWeb, the company running Freebase, was acquired by Google, Freebase was shut down on March 31st, 2015. The last version of Freebase contains roughly 50 million entities and 3 billion facts. Freebases schema comprises roughly 27,000 entity types and 38,000 relation types* [a]*. types*

[a] `https://developers.google.com/freebase/`

Example 2.7 (DBpedia) *DBpedia is a knowledge graph which is extracted from structured data in Wikipedia [19]. The main source for this extraction are the key-value pairs in the Wikipedia infoboxes. In a crowd-sourced process, types of infoboxes are mapped to the DBpedia ontology, and keys used in those infoboxes are mapped to properties in that ontology. Based on those mappings, a knowledge graph can be extracted. The most recent version of the main DBpedia (i.e., DBpedia 2016-10, extracted from the English Wikipedia based on dumps from October 2016) contains 6.6 million entities and 13 billion facts about those entities. The ontology comprises 735 classes and 2,800 relations.*

2.4 MINING ENTITY AND RELATION STRUCTURES

Here we describe the basic tasks in mining factual structures from text corpora, followed by short introduction on related information extraction tasks.

Entity Recognition and Typing: Entity recognition and typing (or named entity recognition) addresses the problem of identification (detection) and classification of pre-defined types of entities, such as `organization` (*e.g.*, "*United Nation*"), `person` (*e.g.*, "*Barack Obama*"), `location` (*e.g.*, "*Los Angeles*"), etc. The detection part aims to find the token span of entities mentioned in text (*i.e.*, entity mention) and the classification part aims to assign the suitable type to entity mention based on its sentence context.

Fine-grained Entity Typing: The goal of fine-grained entity typing is to classify each entity mention m (based on its sentence context s) into a pre-defined set of types where the types are correlated and organized into a tree-structured type hierarchy $\mathcal{Y}$. Each entity mention with be assigned with an entity type path – a path in the given type hierarchy that may not end at a leaf node. For example, in Figure 2.1, the entity mention "*Donald Trump*" is assigned with the type path "`person-artist-actor`" based on the given sentence.

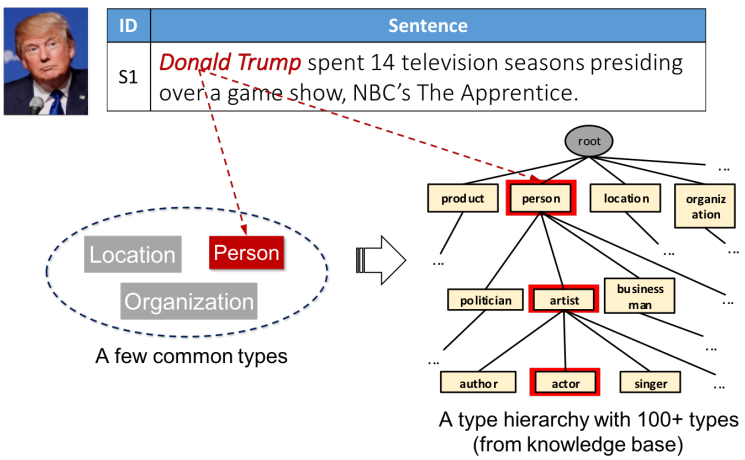


Figure 2.1: Illustration example for fine-grained entity typing.

Relation Extraction: The task of relation extraction aims to detect and classify predefined relationships between entities recognized in text. In the corpus-level relation extraction setting, all sentences $\{s\}$ where a pair of entities (e_i, e_j) (proper names) occur are collected as evidences for determining the appropriate relation type r. In the mention-level relation extraction, the correct label for a relation mention m is determined based on the sentence it occurs, *i.e.*, s. In particular, a label (class) called "`None`" is included into the label set so as to classify a false positive candidate as "no relation".

2.5 COMMON NOTATIONS

We provide the most common notations and their brief definitions in Table 6.4. More specific notations used to explain proposed methods are introduced in the corresponding chapters.

Notation	Definition
s	sentence
d, $\mathcal{D}$	document, corpus
e	entity
t, $\mathcal{T}$	entity type, entity type set
m	entity mention
r, $\mathcal{R}$	relation type, relation type set
(e_i, r, e_j)	relation instance of type r between entities e_i and e_j
z	relation mention
$\mathcal{E}$	finite set of entities in a corpus
$\mathcal{Z}$	finite set of relationships between entities in a corpus
$G = (\mathcal{E}, \mathcal{Z})$	directed, labeled graph that represents StructNet
Ψ	knowledge base (*e.g.*, Freebase, DBpedia)
$\mathcal{E}_\Psi$	set of entities in KB
$\mathcal{T}_\Psi$	entity types in KB
$\mathcal{I}_\Psi$	set of relation instances in KB
$\mathcal{R}_\Psi$	relation types in KB
$\mathcal{Y}$	tree-structured entity type hierarchy

Table 2.1: Common notations and definitions used throughout the thesis.

CHAPTER 3: LITERATURE REVIEW

This chapter provides an overview of prior arts and related studies in mining typed entities and relationships from text. Methods are categorized and organized based on the amounts of human labeled data required in the model training process, which also demonstrate the trajectory of research on reducing human supervision in entity and relation structure mining. We also review techniques developed for learning with noisy labeled data as well as open-domain information extraction, followed by a summary of our contributions.

The existing methods on mining entity and relation structures can be roughly categorized along two dimensions – (1) the amount of human supervision required and (2) the extraction task (problem formulation) it is solving. Table 3.1 gives a few examples for each category. A method can be fully hand-crafted, supervised, weakly supervised or distantly supervised. The second dimension is that the problem formulation of the task can be either sequence labeling (*e.g.*, CRFs), transdutive classification (*e.g.*, pattern bootstrapping, and label propagation), and inductive classification (*e.g.*, SVM). More in-depth discussion about the literature related to concrete tasks and the proposed approaches can be found in each chapter.

Method Category	**Prior Work on Entity Extraction**	**Prior Work on Relation Extraction**
Hand-crafted methods	DIPRE [22], FASTUS [23]	Hearst Pattern [24]
Supervised learning methods	Sequence tagging models like CRFs, HMMs, CMMs [25, 26]	Inductive classifiers like SVM, kernel methods [27, 28], and various deep neural networks
Weakly-supervised methods	KnowItAll [29], SEISA [30], Gupta *et al.* [31], semi-supervised CRFs [32]	SnowBall, NELL [33]
Distantly-supervised methods	[34, 35, 36]	Mintz *et al.* [37], MIME-RE [38], [39]
Open-domain extraction methods	Liberal IE [40], OpenIE systems [5, 41, 42, 43]	OpenIE systems [5, 41, 42, 43]

Table 3.1: A few examples to give an idea about the categories of methods developed based on the amount of human supervision and the task.

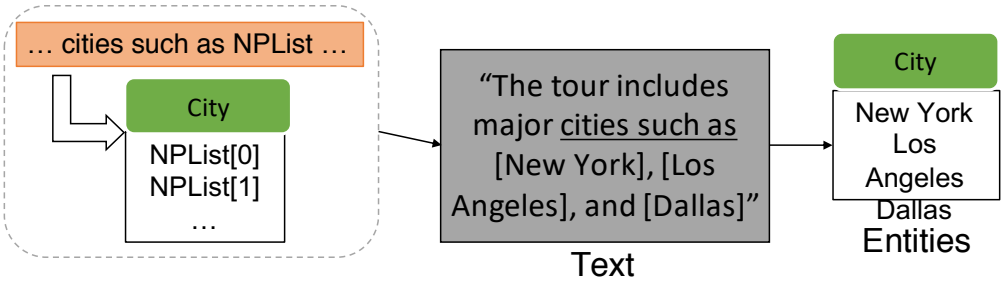


Figure 3.1: An illustration of hand-crafted extraction methods for extracting entity structures.

3.1 HAND-CRAFTED METHODS

One straightforward way of finding entities and relationships in text is to write a set of textual patterns or rules (*e.g.*, regular expression) for the different types of entities (or relationships between entities), with each pattern using some clue to identify the type of entity and relation. For example, in Figure 3.1, the pattern “city such as NPList” is designed to extract a list of `city` entities from text. The NPList matches different noun phrases such as “*New York*”. Therefore, from the sentence in Figure 3.1, this pattern is able to extract three entity mentions of `city` type, *i.e.*, ‘*New York*”, “*Los Angeles*” and “*Dallas*”. Such patterns can be further enhanced with various lexical and syntactic constraints including part-of-speech tags and dependency parse structures.

Hand-crafted systems often rely on extensive lists of people, organizations, locations, and other entity or relation types. Many such lists are now available from the Web. These lists can be quite helpful, and for some entity types they are essential because good contextual patterns are not available, but they should be used with some caution. Systems which are primarily list based will suffer when new names arise, either because of changes in the input texts or the passage of time. Also, large lists may include entries which are proper names but more often are capitalized forms of common words. By building up a set of rules for each entity or relation type, it is possible to create a quite effective extraction system. However, obtaining high performance on corpus of specific domain, genre or language does require some degree of skill. It also generally requires an annotated corpus which can be used to evaluate the rule set after each revision; without such a corpus there is a tendency after a certain point for added rules to actually worsen overall performance.It requires human experts to define rules or regular expressions or program snippets for performing the extraction. That person needs to be a domain expert and a programmer, and possess descent linguistic understanding to be able to develop robust extraction rules.

3.2 TRADITIONAL SUPERVISED LEARNING METHODS

If the creation of a high-quality entity and relation extractor by hand requires an annotated corpus, it is natural to ask whether extraction models can be trained automatically from such a corpus. Such supervised methods for training an fully-supervised extraction model will be considered in this section.

3.2.1 Sequence Labeling Methods

Many statistical learning-based named entity recognition algorithms treat the task as a sequence labeling problem. Sequence labeling is a general machine learning problem and has been used to model many natural language processing tasks including part-of-speech tagging, chunking and named entity recognition. It can be formulated as follows. We are given a sequence of observations, denoted as $x = (x_1, x_2, ..., x_n)$. Usually each observation is represented as a feature vector. We would like to assign a label y_i to each observation x_i. While one may apply standard classification to predict the label y_i based solely on x_i, in sequence labeling, it is assumed that the label y_i depends not only on its corresponding observation xi but also possibly on other observations and other labels in the sequence. Typically this dependency is limited to observations and labels within a close neighborhood of the current position i.

To map entity recognition to a sequence labeling problem, we treat each word in a sentence as an observation. The class labels have to clearly indicate both the boundaries and the types of named entities within the sequence. Usually the BIO notation, initially introduced for text chunking, is used. With this notation, for each entity type T, two labels are created,

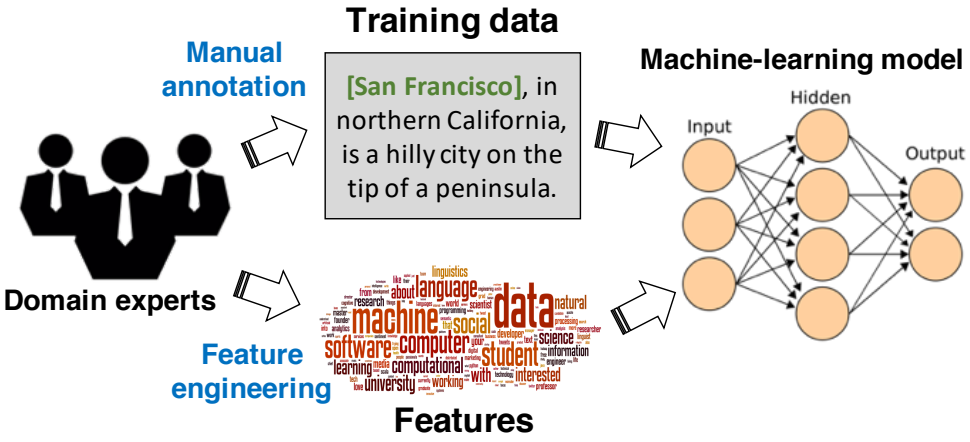


Figure 3.2: An illustration of supervised learning methods for extracting entity structures.

namely, $B-T$ and $I-T$. A token labeled with B-T is the beginning of a named entity of type T while a token labeled with $I-T$ is inside (but not the beginning of) a named entity of type T. In addition, there is a label O for tokens outside of any named entity.

Such supervised methods [26, 44] use fully annotated documents and different linguistic features to train sequence labeling model. To obtain an effective model, the amount of labeled data is significant [26], despite of semi-supervised sequence labeling [32].

3.2.2 Supervised Relation Extraction Methods

Instead of creating and refining these patterns by hand, we can build a relation extractor from an annotated corpus. We will need a corpus which has been annotated both for entity mentions and for relations. We then want to convert relation tagging into a classification problem. To handle a single type of relation, we will train a classifier which classifies each pair of entity mentions appearing in the same sentence as either having or not having this relation. To handle n relation types, we can train an $n+1$-way classifier. Alternatively, we can create two classifiers: a binary classifier which determines whether the entities bear some relation, and an n-way classifier (applied to instances which are passed by the first classifier) which determines which relation is involved. To apply this to new data, we first run an entity tagger and then apply the relation classifier to every pair of entities.

Traditional systems for relation extraction [45, 46, 47] partition the process into several subtasks and solve them incrementally (*i.e.*, detecting entities from text, labeling their types and then extracting their relations). Such systems treat the subtasks independently and so may propagate errors across subtasks in the process. Recent studies [48, 49, 50] focus on joint extraction methods to capture the inherent linguistic dependencies between relations and entity arguments (*e.g.*, the types of entity arguments help determine their relation type, and vice versa) to resolve error propagation.

3.3 WEAKLY-SUPERVISED EXTRACTION METHODS

Weakly-supervised methods utilize a small set of typed entities or relation instances as seeds, and extract more entities or relationships of target types, which can largely reduce the amount of required labeled data.

3.3.1 Semi-supervised Learning

Supervised methods spare us the task of writing rules by hand but still require substantial labor to prepare an annotated corpus. Can we reduce the amount of corpus which we need to

annotate? This is possible through semi-supervised learning methods, including in particular those based on co-training [51, 52]. Semi-supervised methods make use of limited amounts of labeled data together with large amounts of unlabeled data. The basic observation, already made use of in the long-range features of the supervised tagger, is that multiple instances of the same name have the same type. Some of these instances may occur in contexts which are indicative of a particular name type, and these may be used to tag other instances of the same name. In semi-supervised learning we apply these principles repeatedly: starting from a few common names of each type, we look for the contexts in which each of these names appears. If a context appears only with names of one type, we treat this as a predictive context; we look for other names which appear in this context and tag them with the predicted type. We add them to the set of names of known type, and the process repeats.

This is an example of co-training. In co-training, we have two views of the data - two sets of features which can predict the label on the data. In our case the two views are the context of a name and the "spelling" of a name (this includes the complete name and the individual tokens of the name). For each view, we are able to build a model from the labeled data; then we can use this model to label the unlabeled data and associate a confidence with each label. We build a model based on the first view, generate a label for each unlabeled datum, and keep the most confident labels, thus growing the labeled set. Then we do the same using the second view. Gradually the labeled set grows and the models are refined.

3.3.2 Pattern-based Bootstrapping

Pattern-based bootstrapping [31, 53] derives patterns from contexts of seed entities or relation instances, and uses them to incrementally extract new entities / relationships and new patterns unrestricted by specific domains, but often suffers low recall and semantic drift [36].

Iterative bootstrapping, such as probabilistic method [51] and label propagation [54] softly assign multiple types to an entity name and iteratively update its type distribution, yet cannot decide the exact type for each entity mention based on its local context.

The DIPRE system [22] was one of the first systems for weakly-supervised, pattern-based relation extraction, and one of the first designed to operate on the Web. The context patterns were based on character sequences before, between, and after the entities, and so could make use of both lexical contexts and XML mark-up contexts. The patterns were associated with particular web sites. There was no ranking of patterns or entity pairs; instead, some heuristics were used to insure that the patterns were sufficiently specific. The entire procedure was demonstrated on the bookauthor relation. The Snowball system [21]

introduced several improvements, including the use of name types, the ranking of patterns and entity pairs (as described above, using negative examples) and more relaxed patten matching. It was applied to the companyheadquarters relation, a functional relation.

3.4 DISTANTLY-SUPERVISED LEARNING METHODS

Distantly supervised methods [37, 39, 38, 34, 35, 36] avoid expensive human labels by leveraging type information of entity and relation mentions which are confidently mapped to entries in KBs. Linked mentions are used to type those unlinkable ones in different ways, including training a contextual classifier [34], learning a sequence labeling model [36] and serving as labels in graph-based semi-supervised learning [35].

In the context of distant supervision, label noise issue has been studied for other information extraction tasks such as relation extraction [55]. In relation extraction, label noise is introduced by the false positive textual matches of entity pairs. In entity typing, however, label noise comes from the assignment of types to entity mentions without considering their contexts. The forms of distant supervision are different in these two problems. Recently, [56] has tackled the problem of label noise in fine-grained entity typing, but focused on how to generate a clean training set instead of doing entity typing.

3.5 LEARNING WITH NOISY LABELED DATA

Our proposed framework incorporates embedding techniques used in modeling words and phrases in large text corpora [57, 58, 59] ,and nodes and links in graphs/networks [60, 61]. Theses methods assume links are all correct (in unsupervised setting) or labels are all true (in supervised setting). CoType seeks to *model the true links and labels* in the embedding process (*e.g.*, see our comparisons with LINE [60], DeepWalk [61] and FCM [62] in Sec. 6.4.2). Different from embedding structured KB entities and relations [63, 64], our task focuses on embedding entity and relation mentions in *unstructured* contexts.

In the context of modeling noisy labels, our work is related to partial-label learning [56, 65, 66] and multi-label multi-instance learning [38], which deals with the problem where each training instance is associated with a set of noisy candidate labels (where *only one is correct*). Unlike these formulations, our *joint* extraction problem deals with both *classification with noisy labels* and *modeling of entity-relation interactions*. In Sec 6.4.2, we compare our full-fledged model with its variants CoType-EM and CoType-RM to validate the Hypothesis on entity-relation interactions.

3.6 OPEN-DOMAIN INFORMATION EXTRACTION

Traditional techniques for mining entity and relation structures usually work on a corpus from a single domain, *e.g.*articles describing terrorism events, because the goal is to discover the most salient relations from such a domain-specific corpus. In some cases, however, our goal is to find all the potentially useful facts from a large and diverse corpus such as the Web. This is the focus of open information extraction, first introduced in [5], then followed by several other open IE systems [41, 42, 43].

Open information extraction does not assume any specific target relation type. It makes a single pass over the corpus and tries to extract as many relations as possible. Because no relation type is specified in advance, part of the extraction results is a phrase that describes the relation extracted. In other words, open information extraction generates (e_i, r, e_j) tuples where r is not from a finite set of pre-defined relation types, but can be arbitrary predicate or relation phrases. In [5], Banko and Etzioni introduced an unlexicalized CRF-based method for open information extraction. The method is based on the observation that although different relation types have very different semantic meanings, there exists a small set of syntactic patterns that cover the majority of semantic relation mentions. It is therefore possible to train a relation extraction model that extracts arbitrary relations. The key is not to include lexical features in the model.

Extracting textual relation between subjective and objective from text has been extensively studied [41] and applied to entity typing [35]. Fader *et al.* [41] utilize POS patterns to extract verb phrases between detected noun phrases to form relation assertion. Schmitz *et al.* [42] further extend the textual relation by leveraging dependency tree patterns. These methods rely on linguistic parsers that may not generalize across domains. They also do not consider significance of the detected entity mentions in the corpus (see comparison with NNPLB [35]). There have been some studies on clustering and and canonicalizing synonymous relations generated by open information extraction [67]. These methods either ignore entity type information when resolving relations, or assume types of relation arguments are already given.

3.7 CONTRIBUTIONS OF THIS THESIS

There have been extensive studies on extracting typed entities and relations in text (*i.e.*, context-dependent extraction). Most existing work follows an *incremental* diagram—they first perform entity recognition and typing [44, 26] to extract typed entity mentions, and then solve relation extraction [45, 47] to identify relation mentions of target types. Work

along both lines can be categorized in terms of the degree of supervision. While supervised entity recognition systems [25, 44] focus on a few common entity types, weakly-supervised methods [31, 34] and distantly-supervised methods [8, 58, 36] use large text corpus and a small set of seeds (or a knowledge base) to induce patterns or to train models, and thus can apply to different domains without additional human annotation labor. For relation extraction, similarly, weak supervision [68, 3] and distant supervision [69, 70, 38, 71, 39, 37] approaches are proposed to address the domain restriction issue in traditional supervised systems [45, 28, 47]. However, such a "pipeline" diagram ignores the dependencies between different sub tasks and may suffer from error propagation between the tasks.

Some recent efforts have been done on integrating entity extraction with relation extraction by performing global sequence labeling for both entities and relations [48, 49, 72], incorporating type constraints between relations and their arguments [50], or modeling factor graphs [73]. However, these methods require human-annotated corpora (cleaned and general) for model training, and rely on existing entity detectors to provide entity mentions. In particular, [72] integrates entity classification with relation extraction using distant supervision but it ignores label noise issue in the automatically labeled training corpora.

By contrast, our effort-light StructMine framework runs domain-agnostic text segmentation algorithm to mine entity mentions, and adopts a label noise-robust objective to train models using distant supervision in conjunction with external knowledge bases – it carefully models the noisy labels given by distant supervision by leverage the rich data redundancy in the massive corpus. Our method combines the best of two worlds—it leverages the noisy distant supervision in a robust way to address domain restriction (vs. existing joint extraction methods [48, 49]), and models entity-relation interactions jointly with other signals to resolve error propagation (vs. current distant supervision methods [38, 37]).

Part I

Identifying Typed Entities

CHAPTER 4: ENTITY RECOGNITION AND TYPING WITH KNOWLEDGE BASES

4.1 PROPOSED METHOD: OVERVIEW AND MOTIVATION

Entity recognition is an important task in text analysis. Identifying token spans as entity mentions in documents and labeling their types (*e.g.*, people, product or food) enables effective structured analysis of unstructured text corpus. The extracted entity information can be used in a variety of ways (*e.g.*, to serve as primitives for information extraction [42] and knowledge base (KB) population [1]. Traditional named entity recognition systems [26, 44] are usually designed for several major types (*e.g.*, person, organization, location) and general domains (*e.g.*, news), and so require additional steps for adaptation to a new domain and new types.

Entity linking techniques [74] map from given entity mentions detected in text to entities in KBs like Freebase [6], where type information can be collected. But most of such information is manually curated, and thus the set of entities so obtained is of limited coverage and freshness (*e.g.*, over 50% entities mentioned in Web documents are unlinkable [35]). The rapid emergence of large, domain-specific text corpora (*e.g.*, product reviews) poses significant challenges to traditional entity recognition and entity linking techniques and calls for methods of recognizing entity mentions of target types with minimal or no human supervision, and with no requirement that entities can be found in a KB.

There are broadly two kinds of efforts towards that goal: weak supervision and distant supervision. Weak supervision relies on manually-specified seed entity names in applying pattern-based bootstrapping methods [31, 75] or label propagation methods [54] to identify more entities of each type. Both methods assume the seed entities are unambiguous and sufficiently frequent in the corpus, which requires careful seed entity selection by human [76]. Distant supervision is a more recent trend, aiming to reduce expensive human labor by utilizing entity information in KBs [34, 35] (see Fig. 6.1). The typical workflow is: i) detect entity mentions from a corpus, ii) map candidate mentions to KB entities of target types, and iii) use those confidently mapped {mention, type} pairs as labeled data to infer the types of remaining candidate mentions.

In this paper, we study the problem of *distantly-supervised entity recognition in a domain-specific corpus*: Given a domain-specific corpus and a set of target entity types from a KB, we aim to effectively and efficiently detect entity mentions from that corpus, and categorize each by target types or Not-Of-Interest (NOI), with distant supervision. Existing distant supervision methods encounter the following limitations when handling a large, domain-

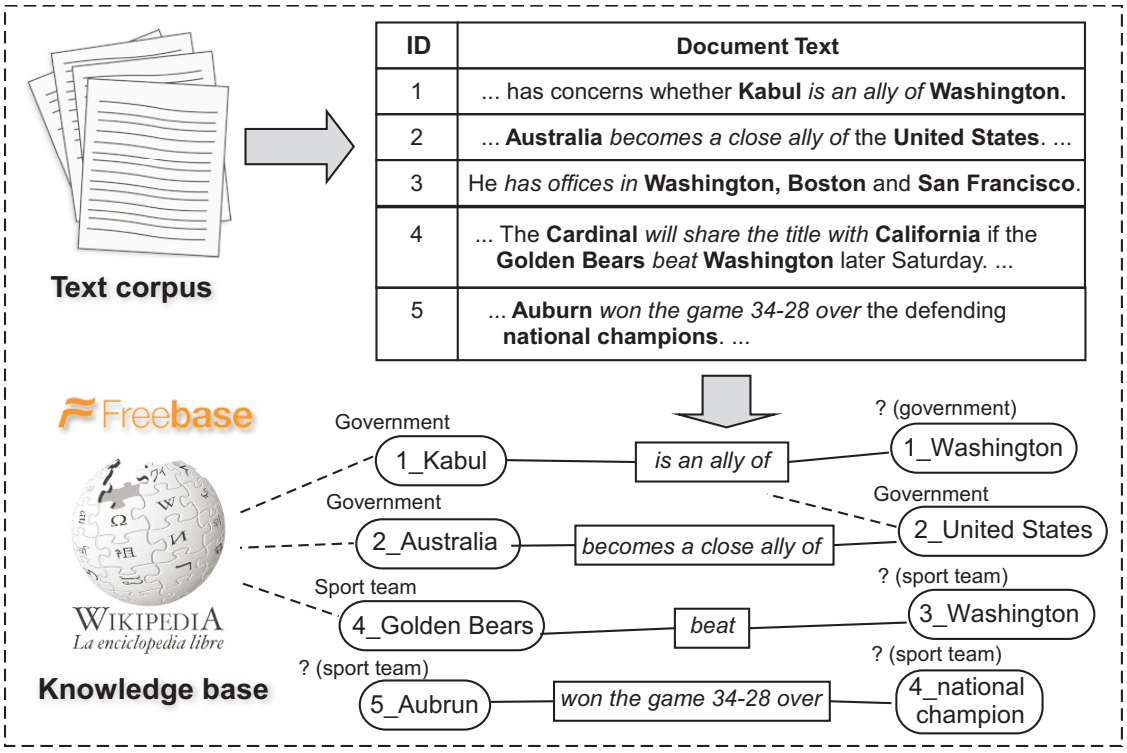


Figure 4.1: An example of distant supervision.

specific corpus.

• **Domain Restriction:** They assume entity mentions are already extracted by existing entity detection tools such as noun phrase chunkers. These tools are usually trained on general-domain corpora like news articles (clean, grammatical) and make use of various linguistic features, but do not work well on specific, dynamic or emerging domains (*e.g.*, tweets or restaurant reviews).

• **Name Ambiguity:** Entity names are often ambiguous—multiple entities may share the same surface name. In Fig. 6.1, for example, the surface name "*Washington*" can refer to either the U.S. government, a sport team, or the U.S. capital city. However, most existing studies [77, 75] simply output a type distribution for each surface name, instead of an exact type for *each* mention of the entity.

• **Context Sparsity:** Previous methods have difficulties in handling entity mentions with sparse context. They leverage a variety of contextual clues to find sources of shared semantics across different entities, including keywords [54], Wikipedia concepts [77], linguistic patterns [34] and textual relations [35]. However, there are often many ways to describe even the same relation between two entities (*e.g.*, "*beat*" and "*won the game 34-28 over*" in Fig. 6.1). This poses challenges on typing entity mentions when they are isolated from other entities or only share infrequent (sparse) context.

We address these challenges with several intuitive ideas. First, to address the domain restriction, we consider a domain-agnostic phrase mining algorithm to extract entity mention candidates with minimal dependence of linguistic assumption (*e.g.*, part-of-speech (POS) tagging requires fewer assumptions of the linguistic characteristics of a domain than semantic parsing). Second, to address the name ambiguity, we do not simply merge the entity mention candidates with identical surface names but model each of them based on its surface name and contexts. Third, to address the context sparsity, we mine *relation phrases* co-occurring with the mention candidates, and infer synonymous relation phrases which share similar type signatures (*i.e.*, express similar types of entities as arguments). This helps to form connecting bridges among entities that do not share identical context, but share synonymous relation phrases.

To systematically integrate these ideas, we develop a novel solution called **ClusType**. First, it mines both entity mention candidates and relation phrases by POS-constrained phrase segmentation; this demonstrates great cross-domain performance (Sec. 6.3.1). Second, it constructs a heterogeneous graph to faithfully represent candidate entity mentions, entity surface names, and relation phrases and their relationship types in a unified form (see Fig. 4.2). The entity mentions are kept as individual objects to be disambiguated, and linked to surface names and relation phrases (Sec. 4.3.2-4.3.4). With the heterogeneous graph, we formulate a graph-based semi-supervised learning of two tasks jointly: (1) type propagation on graph, and (2) relation phrase clustering. By clustering synonymous relation phrases, we can propagate types among entities bridged via these synonymous relation phrases. Conversely, derived entity argument types serve as good features for clustering relation phrases. These two tasks mutually enhance each other and lead to quality recognition of unlinkable entity mentions. In this paper, we present an alternating minimization algorithm to efficiently solve the joint optimization problem, which iterates between type propagation and relation phrase clustering (Sec. 6.3). To our knowledge, this is the first work to *integrate entity recognition with textual relation clustering*.

The major novel contributions of this paper are as follows: (1) we develop an efficient, domain-independent phrase mining algorithm for entity mention candidate and relation phrase extraction; (2) we propose a relation phrase-based entity recognition approach which models the type of each entity mention in a scalable way and softly clusters relation phrases, to resolve name ambiguity and context sparsity issues; (3) we formulate a joint optimization problem for clustering-integrated type propagation; and (4) our experiments on three datasets of different genres—news, Yelp reviews and tweets— demonstrate that the proposed method achieves significant improvement over the state-of-the-art (*e.g.*, 58.3% enhancement in F1 on the Yelp dataset over the best competitor from existing work).

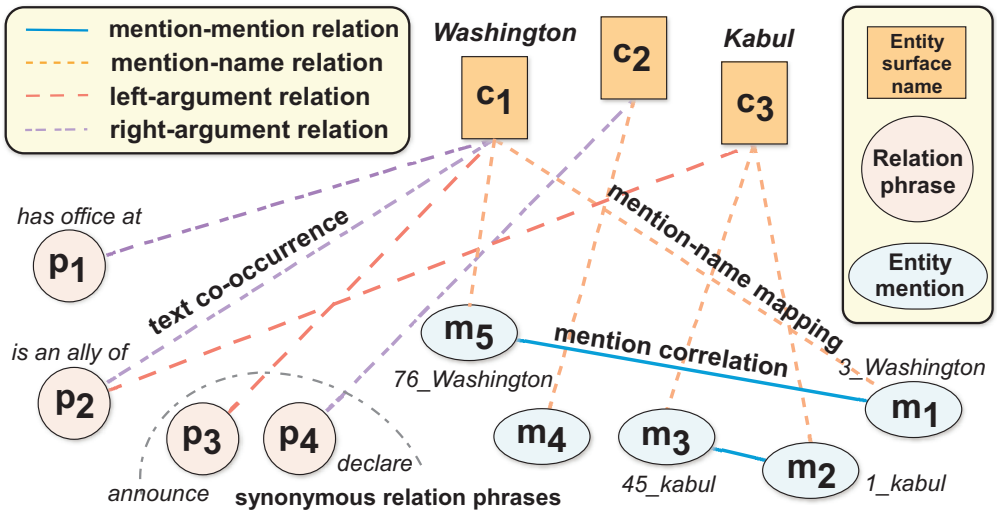


Figure 4.2: The constructed heterogeneous graph.

4.2 PROBLEM DEFINITION

The input to our proposed ER framework is a document collection $\mathcal{D}$, a knowledge base Ψ with type schema $\mathcal{T}_\Psi$, and a *target type set* $\mathcal{T} \subset \mathcal{T}_\Psi$. In this work, we use the type schema of Freebase [6] and assume $\mathcal{T}$ is covered by Freebase.

An *entity mention*, m, is a token span in the text document which refers to a real-world entity e. Let c_m denote the *surface name* of m. In practice, people may use multiple surface names to refer to the same entity (*e.g.*, "*black mamba*" and "*KB*" for *Kobe Bryant*). On the other hand, a surface name c could refer to different entities (*e.g.*, "*Washington*" in Fig. 6.1). Moreover, even though an entity e can have multiple types (*e.g.*, *J.F.K. airport* is both a location and an organization), the type of its specific mention m is usually unambiguous. We use a type indicator vector $\mathbf{y}_m \in \{0,1\}^T$ to denote the entity type for each mention m, where $T = |\mathcal{T}| + 1$, *i.e.*, m has type $t \in \mathcal{T}$ or is Not-of-Interest (NOI). By estimating $\mathbf{y}_m$, one can predict type of m as type $(m) = \text{argmax}_{1 \le i \le T}\ y_{m,i}$.

Extracting textual relations from documents has been previously studied [41] and applied to entity typing [34, 35]. A *relation phrase* is a phrase that denotes a unary or binary relation in a sentence [41] (see Fig. 4.3 for example). We leverage the rich semantics embedded in relation phrases to provide type cues for their entity arguments. Specifically, we define the *type signature* of a relation phrase p as two indicator vectors $\mathbf{p}_L, \mathbf{p}_R \in \mathbb{R}^T$. They measure how likely the left/right entity arguments of p belong to different types ($\mathcal{T}$ or NOI). A large positive value on $p_{L,t}$ ($p_{R,t}$) indicates that the left/right argument of p is likely of type t.

Let $\mathcal{M} = \{m_1, ..., m_M\}$ denote the set of M candidate entity mentions extracted from $\mathcal{D}$. Suppose a subset of entity mentions $\mathcal{M}_L \subset \mathcal{M}$ can be confidently mapped to entities in Ψ.

The type of a linked candidate $m \in \mathcal{M}_L$ can be obtained based on its mapping entity $\kappa_e(m)$ (see Sec. 4.4.1). This work focuses on predicting the types of *unlinkable candidate mentions* $\mathcal{M}_U = \mathcal{M} \backslash \mathcal{M}_L$, where $\mathcal{M}_U$ may consist of (1) mentions of the emerging entities which are not in Ψ; (2) new names of the existing entities in Ψ; and (3) invalid entity mentions. Formally, we define the problem of **distantly-supervised entity recognition** as follows

Problem 4.1: Entity Recognition and Typing

Given a document collection $\mathcal{D}$, a target type set $\mathcal{T}$ and a knowledge base Ψ, our task aims to: (1) extract candidate entity mentions $\mathcal{M}$ from $\mathcal{D}$; (2) generate seed mentions $\mathcal{M}_L$ with Ψ; and (3) for each unlinkable candidate mention $m \in \mathcal{M}_U$, estimate its type indicator vector $\mathbf{y}_m$ to predict its type.

In our study, we assume each mention within a sentence is only associated with a single type $t \in \mathcal{T}$. We also assume the target type set $\mathcal{T}$ is given (It is outside the scope of this study to generate $\mathcal{T}$). Finally, while our work is independent of entity linking techniques [74], our ER framework output may be useful to entity linking.

Framework Overview. Our overall framework is as follows:

1. Perform phrase mining on a POS-tagged corpus to extract candidate entity mentions and relation phrases, and construct a heterogeneous graph G to represent available information in a unified form, which encodes our insights on modeling the type for each entity mention (Sec. 4.3).
2. Collect seed entity mentions $\mathcal{M}_L$ as labels by linking extracted candidate mentions $\mathcal{M}$ to the KB Ψ (Sec. 4.4.1).
3. Estimate type indicator $\mathbf{y}$ for unlinkable candidate mention $m \in \mathcal{M}_U$ with the proposed type propagation integrated with relation phrase clustering on G (Sec. 6.3).

4.3 RELATION PHRASE-BASED GRAPH CONSTRUCTION

We first introduce candidate generation in Sec. 6.3.1, which leads to three kinds of objects, namely candidate entity mentions $\mathcal{M}$, their surface names $\mathcal{C}$ and surrounding relation phrases $\mathcal{P}$. We then build a heterogeneous graph G, which consists of multiple types of objects and multiple types of links, to model their relationship. The basic idea for constructing the graph is that: the more two objects are likely to share the same label (*i.e.*, $t \in \mathcal{T}$ or NOI), the larger the weight will be associated with their connecting edge.

Over:RP the weekend the system:EP dropped:RP nearly inches of snow in:RP western Oklahoma:EP and at:RP [Dallas Fort Worth International Airport]:EP sleet and ice caused:RP hundreds of [flight cancellations]:EP and delays. ..... It is forecast:RP to reach:RP [northern Georgia]:EP by:RP [Tuesday afternoon]:EP, Washington:EP and [New York]:EP by:RP [Wednesday afternoon]:EP, meteorologists:EP said:RP.

`EP: entity mention candidate; RP: relation phrase.`

Figure 4.3: Example output of candidate generation.

Specifically, the constructed graph G unifies three types of links: *mention-name link* which represents the mapping between entity mentions and their surface names, *entity name-relation phrase link* which captures corpus-level co-occurrences between entity surface names and relation phrase, and *mention-mention link* which models distributional similarity between entity mentions. This leads to three subgraphs $G_{\mathcal{M},\mathcal{C}}$, $G_{\mathcal{C},\mathcal{P}}$ and $G_{\mathcal{M}}$, respectively. We introduce the construction of them in Secs. 4.3.2–4.3.4.

4.3.1 Candidate Generation

To ensure the extraction of informative and coherent entity mentions and relation phrases, we introduce a scalable, data-driven phrase mining method by incorporating both corpus-level statistics and syntactic constraints. Our method adopts a *global significance score* to guide the filtering of low-quality phrases and relies on a set of generic POS patterns to remove phrases with improper syntactic structure [41]. By extending the methodology used in [78], we can partition sentences in the corpus into non-overlapping segments which meet a significance threshold and satisfy our syntactic constraints. In doing so, entity candidates and relation phrases can be jointly extracted in an effective way.

First, we mine *frequent contiguous patterns* (*i.e.*, sequences of tokens with no gap) up to a fixed length and aggregate their counts. A greedy agglomerative merging is then performed to form longer phrases while enforcing our syntactic constraints. Suppose the size of corpus $\mathcal{D}$ is N and the frequency of a phrase S is denoted by $\upsilon(S)$. The phrase-merging step selects the most significant merging, by comparing the frequency of a potential merging of two consecutive phrases, $\upsilon(S_1 \oplus S_2)$, to the expected frequency assuming independence, $N\frac{\upsilon(S_1)}{N}\frac{\upsilon(S_2)}{N}$. Additionally, we conduct syntactic constraint check on every potential merging by applying an entity check function $I_e(\cdot)$ and a relation check function $I_p(\cdot)$. $I_e(S)$ returns one if S is *consecutive nouns* and zero otherwise; and $I_p(S)$ return one if S (partially) matches one of the patterns in Table 4.2. Similar to Student's t-test, we define a score function $\rho_X(\cdot)$ to measure the significance and syntactic correctness of a merging [78], where X can be e

Table 4.1: Performance on entity detection.

Method	**NYT**		**Yelp**		**Tweet**	
	Prec	Recall	Prec	Recall	Prec	Recall
Our method	**0.469**	**0.956**	**0.306**	**0.849**	0.226	**0.751**
NP chunker	0.220	0.609	0.296	0.247	**0.287**	0.181

(entity mention) or p (relation phrase).

$$\rho_X(S_1, S_2) = \frac{\upsilon(S_1 \oplus S_2) - N\frac{\upsilon(S_1)}{N}\frac{\upsilon(S_2)}{N}}{\sqrt{\upsilon(S_1 \oplus S_2)}} \cdot I_X(S_1 \oplus S_2) \tag{4.1}$$

At each iteration, the greedy agglomerative algorithm performs the merging which has highest scores (ρ_e or ρ_p), and terminates when the next highest-score merging does not meet a pre-defined significance threshold. Relation phrases without matched POS patterns are discarded and their valid sub-phrases are recovered. Because the significance score can be considered analogous to hypothesis testing, one can use standard rule-of thumb values for the threshold (*e.g.*, Z-score$\geq$2) [78]. Overall the threshold setting is not sensitive in our empirical studies. As all merged phrases are frequent, we have fast access to their aggregate counts and thus it is efficient to compute the score of a potential merging.

Fig. 4.3 provides an example output of the candidate generation on New York Times (NYT) corpus. We further compare our method with a popular noun phrase chunker[1] in terms of entity detection performance, using the extracted entity mentions. Table 4.1 summarizes the comparison results on three datasets from different domains (see Sec. 6.4 for details). Recall is most critical for this step, since we can recognize false positives in later stages of our framework, but no chance to later detect the misses, *i.e.*, false negatives.

4.3.2 Mention-Name Subgraph

In practice, directly modeling the type indicator for each candidate mention may be infeasible due to the large number of candidate mentions (*e.g.*, $|\mathcal{M}| > 1$ million in our experiments). This results in an intractable size of parameter space, *i.e.*, $\mathcal{O}(|M|T)$. Intuitively, both the entity name and the surrounding relation phrases provide strong cues on the type of a candidate entity mention. In Fig. 6.1, for example, the relation phrase "*beat*" suggests "*Golden Bears*" can mention a person or a sport team, while the surface name "*Golden Bears*" may

[1]TextBlob: `http://textblob.readthedocs.org/en/dev/`

Table 4.2: POS tag patterns for relation phrases.

Pattern	Example
V	disperse; hit; struck; knock;
P	in; at; of; from; to;
V P	locate in; come from; talk to;
VW*(P)	caused major damage on; come lately

`V-verb; P-prep; W-{adv | adj | noun | det | pron}`
`W* denotes multiple W; (P) denotes optional.`

refer to a sport team or a company. We propose to model the type indicator of a candidate mention based on the type indicator of its surface name and the type signatures of its associated relation phrases (see Sec. 6.3 for details). By doing so, we can reduce the size of the parameter space to $\mathcal{O}\big((|\mathcal{C}|+|\mathcal{P}|)T\big)$ where $|\mathcal{C}|+|\mathcal{P}| \ll |\mathcal{M}|$ (see Table 6.5 and Sec 6.4.1). This enables our method to scale up.

Suppose there are n unique surface names $\mathcal{C} = \{c_1, ..., c_n\}$ in all the extracted candidate mentions $\mathcal{M}$. This leads to a biadjacency matrix $\Pi_{\mathcal{C}} \in \{0,1\}^{M\times n}$ to represent the subgraph $G_{\mathcal{M},\mathcal{C}}$, where $\Pi_{\mathcal{C},ij} = 1$ if the surface name of m_j is c_j, and 0 otherwise. Each column of $\Pi_{\mathcal{C}}$ is normalized by its ℓ_2-norm to reduce the impact of popular entity names. We use a T-dimensional type indicator vector to measure how likely an entity name is subject to different types ($\mathcal{T}$ or NOI) and denote the type indicators for $\mathcal{C}$ by matrix $\mathbf{C} \in \mathbb{R}^{n\times T}$. Similarly, we denote the type indicators for $\mathcal{M}$ by $\mathbf{Y} \in \mathbb{R}^{M\times T}$.

4.3.3 Name-Relation Phrase Subgraph

By exploiting the aggregated co-occurrences between entity surface names and their surrounding relation phrases across multiple documents *collectively*, we weight the importance of different relation phrases for an entity name, and use their connected edge as bridges to propagate type information between different surface names by way of relation phrases. For each mention candidate, we assign it as the *left (right, resp.) argument* to the closest relation phrase appearing on its right (left, resp.) in a sentence. The *type signature* of a relation phrase refers to the two type indicators for its left and right arguments, respectively. The following hypothesis guides the type propagation between surface names and relation phrases.

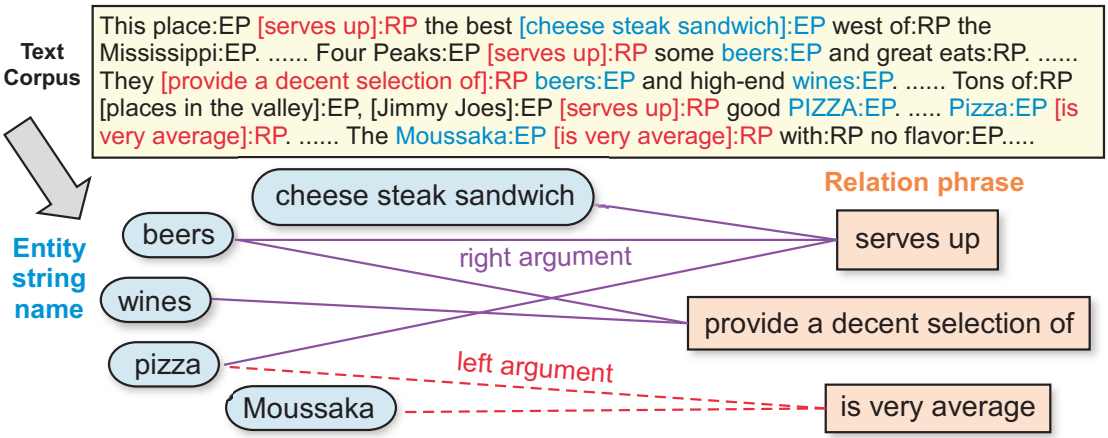


Figure 4.4: Example entity name-relation phrase links from Yelp reviews.

Hypothesis 4.1: Entity-Relation Co-occurrences

If surface name c often appears as the left (right) argument of relation phrase p, then c's type indicator tends to be similar to the corresponding type indicator in p's type signature.

In Fig. 4.4, for example, if we know "*pizza*" refers to food and find it frequently co-occurs with the relation phrase "*serves up*" in its right argument position, then another surface name that appears in the right argument position of "*serves up*" is likely food. This reinforces the type propagation that "*cheese steak sandwich*" is also food.

Formally, suppose there are l different relation phrases $\mathcal{P} = \{p_1, ..., p_l\}$ extracted from the corpus. We use two biadjacency matrices $\Pi_L, \Pi_R \in \{0,1\}^{M \times l}$ to represent the co-occurrences between relation phrases and their left and right entity arguments, respectively. We define $\Pi_{L,ij} = 1$ ($\Pi_{R,ij} = 1$) if m_i occurs as the *closest* entity mention on the left (right) of p_j *in a sentence*; and 0 otherwise. Each column of Π_L and Π_R is normalized by its ℓ_2-norm to reduce the impact of popular relation phrases. Two bipartite subgraphs $G_{\mathcal{C},\mathcal{P}}$ can be further constructed to capture the aggregated co-occurrences between relation phrases $\mathcal{P}$ and entity names $\mathcal{C}$ across the corpus. We use two biadjacency matrices $\mathbf{W}_L, \mathbf{W}_R \in \mathbb{R}^{n \times l}$ to represent the edge weights for the two types of links, and normalize them.

$$\mathbf{W}_L = \Pi_{\mathcal{C}}^T \Pi_L \quad \text{and} \quad \mathbf{W}_R = \Pi_{\mathcal{C}}^T \Pi_R;$$
$$\mathbf{S}_L = \mathbf{D}_L^{(\mathcal{C})^{-\frac{1}{2}}} \mathbf{W}_L \mathbf{D}_L^{(P)^{-\frac{1}{2}}} \text{ and } \mathbf{S}_R = \mathbf{D}_R^{(\mathcal{C})-\frac{1}{2}} \mathbf{W}_R \mathbf{D}_R^{(P)-\frac{1}{2}},$$

where $\mathbf{S}_L$ and $\mathbf{S}_R$ are normalized biadjacency matrices. For left-argument relationships, we define the diagonal surface name degree matrix $\mathbf{D}_L^{(\mathcal{C})} \in \mathbb{R}^{n \times n}$ as $D_{L,ii}^{(\mathcal{C})} = \sum_{j=1}^{l} W_{L,ij}$ and

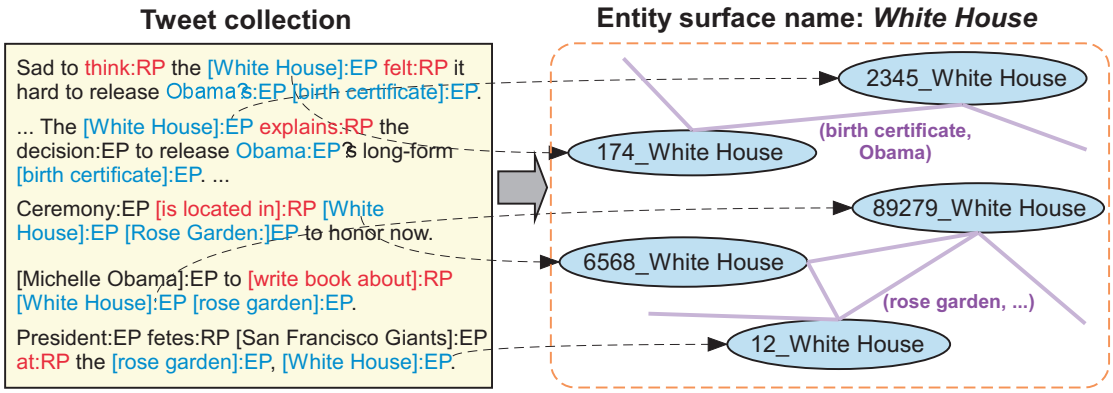


Figure 4.5: Example mention-mention links for entity surface name "*White House*" from Tweets.

the relation phrase degree matrix $\mathbf{D}_L^{(\mathcal{P})} \in \mathbb{R}^{l \times l}$ as $D_{L,jj}^{(\mathcal{P})} = \sum_{i=1}^{n} W_{L,ij}$. Likewise, we define $\mathbf{D}_R^{(\mathcal{C})} \in \mathbb{R}^{n \times n}$ and $\mathbf{D}_R^{(\mathcal{P})} \in \mathbb{R}^{l \times l}$ based on $\mathbf{W}_R$ for the right-argument relationships.

4.3.4 Mention Correlation Subgraph

An entity mention candidate may have an ambiguous name as well as associate with ambiguous relation phrases. For example, "*White House*" mentioned in the first sentence in Fig. 4.5 can refer to either an organization or a facility, while its relation phrase "*felt*" can have either a person or an organization entity as the left argument. It is observed that other co-occurring entity mentions (*e.g.*, "*birth certificate*" and "*rose garden*" in Fig. 4.5) may provide good hints to the type of an entity mention candidate. We propose to propagate the type information between candidate mentions of *each* entity name based on the following hypothesis.

Hypothesis 4.2: Mention correlation

If there exists a strong correlation (*i.e.*, within sentence, common neighbor mentions) between two candidate mentions that share the same name, then their type indicators tend to be similar.

Specifically, for each candidate entity mention $m_i \in \mathcal{M}$, we extract the set of entity surface names which co-occur with m_i *in the same sentence*. An n-dimensional TF-IDF vector $\mathbf{f}^{(i)} \in \mathbb{R}^n$ is used to represent the importance of these co-occurring names for m_i where $f_j^{(i)} = v_s(c_j) \cdot \log(|\mathcal{D}|/v_{\mathcal{D}}(c_j))$ with term frequency in the sentence $v_s(c_j)$ and document frequency $v_{\mathcal{D}}(c_j)$ in $\mathcal{D}$. We use an affinity subgraph to represent the mention-mention link based on k-nearest neighbor (KNN) graph construction [79], denoted by an adjacency matrix $\mathbf{W}_{\mathcal{M}} \in \mathbb{R}^{M \times M}$. Each mention candidate is linked to its k most similar mention candidates

which share the same name in terms of the vectors $\mathbf{f}$.

$$W_{\mathcal{M},ij} = \begin{cases} \text{sim}(\mathbf{f}^{(i)}, \mathbf{f}^{(j)}), & \text{if } \mathbf{f}^{(i)} \in N_k(\mathbf{f}^{(j)}) \text{ or } \mathbf{f}^{(j)} \in N_k(\mathbf{f}^{(i)}) \\ & \text{and } c(m_i) = c(m_j); \\ 0, & \text{otherwise.} \end{cases}$$

where we use the heat kernel function to measure similarity, *i.e.*, $\text{sim}(\mathbf{f}^{(i)}, \mathbf{f}^{(j)}) = \exp\left(-\|\mathbf{f}^{(i)} - \mathbf{f}^{(j)}\|^2/t\right)$ with $t = 5$ [79]. We use $N_k(\mathbf{f})$ to denote k nearest neighbors of $\mathbf{f}$ and $c(m)$ to denote the surface name of mention m. Similarly, we normalize $\mathbf{W}_{\mathcal{M}}$ into $\mathbf{S}_{\mathcal{M}} = \mathbf{D}_{\mathcal{M}}^{-\frac{1}{2}} \mathbf{W}_{\mathcal{M}} \mathbf{D}_{\mathcal{M}}^{-\frac{1}{2}}$ where the degree matrix $\mathbf{D}_{\mathcal{M}} \in \mathbb{R}^{M \times M}$ is defined by $D_{\mathcal{M},ii} = \sum_{j=1}^{M} W_{\mathcal{M},ij}$.

4.4 CLUSTERING-INTEGRATED TYPE PROPAGATION ON GRAPHS

This section introduces our unified framework for joint type propagation and relation phrase clustering on graphs.

A straightforward solution is to first perform hard clustering on the extracted relation phrases and then conduct type propagation between entity names and relation phrase clusters. Such a solution encounters several problems. One relation phrase may belong to multiple clusters, and the clusters so derived do not incorporate the type information of entity arguments. As such, the type prediction performance may not be best optimized by the mined clusters.

In our solution, we formulate a joint optimization problem to minimize both a graph-based semi-supervised learning error and a multi-view relation phrase clustering objective.

4.4.1 Seed Mention Generation

We first collect type information for the extracted mention candidates $\mathcal{M}$ by linking them to the KB. This yields a set of type-labeled mentions $\mathcal{M}_L$. Our goal is then to type the remaining unlinkable mention candidates $\mathcal{M}_U = \mathcal{M}/\mathcal{M}_L$.

We utilize a state-of-the-art entity name disambiguation tool[2] to map each candidate mention to Freebase entities. Only the mention candidates which are mapped with high confidence scores (*i.e.*, $\eta \geq 0.8$) are considered as valid output. We denote the mapping entity of a linked mention m as $\kappa_e(m)$, and the set of types of $\kappa_e(m)$ in Freebase as $\mathcal{T}(m)$. The linked mentions which associate with multiple target types (*i.e.*, $|\mathcal{T}(m) \cap \mathcal{T}| > 1$) are discarded to avoid type ambiguity. This finally leads to a set of labeled (seed) mentions $\mathcal{M}_L$. In our experiments, we found that only a very limited amount of extracted candidate entity

[2] `http://spotlight.dbpedia.org/`

Table 4.3: Statistics for seed generation.

Dataset	NYT	Yelp	Tweet
#Extracted mentions	4.88M	1.32M	703k
%Seed mentions	6.98	4.57	1.83
#Entities	17,326	5,662	12,211

mentions can be confidently mapped to Freebase entities (*i.e.*, $|\mathcal{M}_L|/|\mathcal{M}| < 7\%$). We define the type indicator $\mathbf{y}_m$ for a linked mention $m \in \mathcal{M}_L$ as $\mathbf{y}_{m,t} = 1$ if $\mathcal{T}(m) \cap \mathcal{T} = \{t\}$ and 0 otherwise, for $t \in \mathcal{T}$. Meanwhile, $\mathbf{y}_{m,\text{NOI}}$ is assigned with 1 if $\mathcal{T}(m) \cap \mathcal{T} = \emptyset$ and 0 otherwise.

4.4.2 Relation Phrase Clustering

In practice, we observe that many extracted relation phrases have very few occurrences in the corpus. This makes it hard to model their type signature based on the aggregated co-occurrences with entity names (*i.e.*, Hypothesis 4.3.3). In our experimental datasets, about 37% of the relation phrases have less than 3 unique entity surface names (in right or left arguments) in $G_{\mathcal{C},\mathcal{P}}$. Intuitively, by softly clustering synonymous relation phrases, the type signatures of frequent relation phrases can help infer the type signatures of infrequent (sparse) ones that have similar cluster memberships, based on the following hypothesis.

Hypothesis 4.3: Type signature consistency

If two relation phrases have similar cluster memberships, the type indicators of their left and right arguments (type signature) tend to be similar, respectively.

There has been some studies [67, 80] on clustering synonymous relation phrases based on different kinds of signals and clustering methods. We propose a general relation phrase clustering method to incorporate different features for clustering, which can be integrated with the graph-based type propagation in a mutually enhancing framework, based on the following hypothesis.

Hypothesis 4.4: Relation phrase similarity

Two
relation phrases tend to have similar cluster memberships, if they have similar (1) strings; (2) context words; and (3) left and right argument type indicators.

In particular, type signatures of relation phrases have proven very useful in clustering of relation phrases which have infrequent or ambiguous strings and contexts [67]. In contrast to previous approaches, our method leverages the type information derived by the type propagation and thus does not rely strictly on external sources to determine the type information for all the entity arguments.

Formally, suppose there are n_s (n_c) unique words $\{w_1, ..., w_{n_s}\}$ ($\{w'_1, ..., w'_{n_c}\}$) in all the relation phrase strings (contexts). We represent the strings and contexts of the extracted relation phrases $\mathcal{P}$ by two feature matrices $\mathbf{f}_s \in \mathbb{R}^{l \times n_s}$ and $\mathbf{f}_c \in \mathbb{R}^{l \times n_c}$, respectively. We set $F_{s,ij} = 1$ if p_i contains the word w_j and 0 otherwise. We use a text window of 10 words to extract the context for a relation phrase from each sentence it appears in, and construct context features $\mathbf{f}_c$ based on TF-IDF weighting. Let $\mathbf{P}_L, \mathbf{P}_R \in \mathbb{R}^{l \times T}$ denote the type signatures of $\mathcal{P}$. Our solution uses the derived features (*i.e.*, $\{\mathbf{f}_s, \mathbf{f}_c, \mathbf{P}_L, \mathbf{P}_R\}$) for multi-view clustering of relation phrases based on joint non-negative matrix factorization, which will be elaborated in the next section.

4.4.3 The Joint Optimization Problem

Our goal is to infer the label (type $t \in \mathcal{T}$ or NOI) for *each* unlinkable entity mention candidate $m \in \mathcal{M}_U$, *i.e.*, estimating $\mathbf{Y}$. We propose an optimization problem to unify two different tasks to achieve this goal: (i) type propagation over both the type indicators of entity names $\mathbf{C}$ and the type signatures of relation phrases $\{\mathbf{P}_\mathrm{L}, \mathbf{P}_\mathrm{R}\}$ on the heterogeneous graph G by way of graph-based semi-supervised learning, and (ii) multi-view relation phrase clustering. The seed mentions $\mathcal{M}_L$ are used as initial labels for the type propagation. We formulate the objective function as follows.

$$\begin{aligned}\mathcal{O}_{\alpha,\gamma,\mu} = \mathcal{F}(\mathbf{C}, \mathbf{P}_\mathrm{L}, \mathbf{P}_\mathrm{R}) + \mathcal{L}_\alpha\big(\mathbf{P}_\mathrm{L}, \mathbf{P}_\mathrm{R}, \{\mathbf{U}^{(v)}, \mathbf{V}^{(v)}\}, \mathbf{U}^*\big) \\ + \Omega_{\gamma,\mu}(\mathbf{Y}, \mathbf{C}, \mathbf{P}_\mathrm{L}, \mathbf{P}_\mathrm{R}).\end{aligned} \tag{4.2}$$

The first term $\mathcal{F}$ follows from Hypothesis 4.3.3 to model *type propagation* between entity names and relation phrases. By extending local and global consistency idea [79], it ensures that the type indicator of an entity name is similar to the type indicator of the left (or right) argument of a relation phrase, if their corresponding association is strong.

$$\mathcal{F}(\mathbf{C}, \mathbf{P}_\mathrm{L}, \mathbf{P}_\mathrm{R}) = \sum_{Z \in \{L,R\}} \sum_{i=1}^{n} \sum_{j=1}^{l} W_{Z,ij} \left\| \frac{\mathbf{C}_i}{\sqrt{D^{(\mathcal{C})}_{Z,ii}}} - \frac{\mathbf{P}_{Z,j}}{\sqrt{D^{(\mathcal{P})}_{Z,jj}}} \right\|_2^2, \tag{4.3}$$

The second term $\mathcal{L}_\alpha$ in Eq. (4.2) follows Hypotheses 4.4.2 and 4.4.2 to model the *multi-view relation phrase clustering* by joint non-negative matrix factorization. In this study, we consider each derived feature as one *view* in the clustering, *i.e.*, $\{\mathbf{f}^{(0)}, \mathbf{f}^{(1)}, \mathbf{f}^{(2)}, \mathbf{f}^{(3)}\} = \{\mathbf{P}_{\mathrm{L}}, \mathbf{P}_{\mathrm{R}}, \mathbf{f}_s, \mathbf{f}_c\}$ and derive a four-view clustering objective as follows.

$$\begin{aligned}&\mathcal{L}_\alpha\big(\mathbf{P}_{\mathrm{L}}, \mathbf{P}_{\mathrm{R}}, \{\mathbf{U}^{(v)}, \mathbf{V}^{(v)}\}, \mathbf{U}^*\big) \\ &= \sum_{v=0}^{d} \beta^{(v)} \big(\| \mathbf{f}^{(v)} - \mathbf{U}^{(v)}\, \mathbf{V}^{(v)T} \|_F^2 + \alpha \| \mathbf{U}^{(v)}\, \mathbf{Q}^{(v)} - \mathbf{U}^* \|_F^2 \big).\end{aligned} \tag{4.4}$$

The first part of Eq. (4.4) performs matrix factorization on each feature matrix. Suppose there exists K relation phrase clusters. For each view v, we factorize the feature matrix $\mathbf{f}^{(v)}$ into a cluster membership matrix $\mathbf{U}^{(v)} \in \mathbb{R}_{\geq 0}^{l \times K}$ for all relation phrases $\mathcal{P}$ and a type indicator matrix $\mathbf{V}^{(v)} \in \mathbb{R}_{\geq 0}^{T \times K}$ for the K derived clusters. The second part of Eq. (4.4) enforces the consistency between the four derived cluster membership matrices through a *consensus matrix* $\mathbf{U}^* \in \mathbb{R}_{\geq 0}^{l \times K}$, which applies Hypothesis 4.4.2 to incorporate multiple similarity measures to cluster relation phrases. As in [81], we normalize $\{\mathbf{U}^{(v)}\}$ to the same scale (*i.e.*, $\| \mathbf{U}^{(\mathrm{v})}\, \mathbf{Q}^{(\mathrm{v})} \|_F \approx 1$) with the diagonal matrices $\{\mathbf{Q}^{(v)}\}$, where $Q_{kk}^{(v)} = \sum_{i=1}^{T} V_{ik}^{(v)} / \|\mathbf{f}^{(v)}\|_F$, so that they are comparable under the same consensus matrix. A tuning parameter $\alpha \in [0, 1]$ is used to control the degree of consistency between the cluster membership of each view and the consensus matrix. $\{\beta^{(v)}\}$ are used to weight the information among different views, which will be automatically estimated. As the first part of Eq. (4.4) enforces $\{\mathbf{U}^{(0)}, \mathbf{U}^{(1)}\} \approx \mathbf{U}^*$ and the second part of Eq. (4.4) imposes $\mathbf{P}_L \approx \mathbf{U}^{(0)}\, \mathbf{V}^{(0)T}$ and $\mathbf{P}_R \approx \mathbf{U}^{(1)}\, \mathbf{V}^{(1)T}$, it can be checked that $U_i^* \approx U_j^*$ implies both $P_{L,i} \approx P_{L,j}$ and $P_{R,i} \approx P_{R,j}$ for any two relation phrases, which captures Hypothesis 4.4.2.

The last term $\Omega_{\gamma,\mu}$ in Eq. (4.2) models the type indicator for each entity mention candidate, the mention-mention link and the supervision from seed mentions.

$$\begin{aligned}&\Omega_{\gamma,\mu}(\mathbf{Y}, \mathbf{C}, \mathbf{P}_L, \mathbf{P}_R) = \big\| \mathbf{Y} - f(\Pi_{\mathcal{C}}\, \mathbf{C}, \Pi_L\, \mathbf{P}_L, \Pi_R\, \mathbf{P}_R) \big\|_F^2 \\ &+ \frac{\gamma}{2} \sum_{i,j=1}^{M} W_{\mathcal{M},ij} \left\| \frac{\mathbf{Y}_i}{\sqrt{D_{ii}^{(\mathcal{M})}}} - \frac{\mathbf{Y}_j}{\sqrt{D_{jj}^{(\mathcal{M})}}} \right\|_2^2 + \mu \| \mathbf{Y} - \mathbf{Y}_0 \|_F^2.\end{aligned} \tag{4.5}$$

In the first part of Eq. (4.5), the type of each entity mention candidate is modeled by a function $f(\cdot)$ based on the the type indicator of its surface name as well as the type signatures of its associated relation phrases. Different functions can be used to combine the information from surface names and relation phrases. In this study, we use an equal-weight linear combination, *i.e.*, $f(\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3) = \mathbf{X}_1 + \mathbf{X}_2 + \mathbf{X}_3$. The second part follows Hypothesis 4.3.4 to model the mention-mention correlation by graph regularization, which ensures

the consistency between the type indicators of two candidate mentions if they are highly correlated. The third part enforces the estimated $\mathbf{Y}$ to be similar to the initial labels from seed mentions, denoted by a matrix $\mathbf{Y}_0 \in \mathbb{R}^{\mathcal{M}\times T}$ (see Sec. 4.4.1). Two tuning parameters $\gamma, \mu \in [0, 1]$ are used to control the degree of guidance from mention correlation in $G_{\mathcal{M}}$ and the degree of supervision from $\mathbf{Y}_0$, respectively.

To derive the exact type of each candidate entity mention, we impose the 0-1 integer constraint $\mathbf{Y} \in \{0, 1\}^{M\times T}$ and $\mathbf{Y}\,\mathbf{1} = \mathbf{1}$. To model clustering, we further require the cluster membership matrices $\{\mathbf{U}^{(\mathrm{v})}\}$, the type indicator matrices of the derived clusters $\{\mathbf{V}^{(\mathrm{v})}\}$ and the consensus matrix $\mathbf{U}^*$ to be non-negative. With the definition of $\mathcal{O}$, we define the joint optimization problem as follows.

$$\begin{aligned} &\min_{\substack{\mathbf{Y},\mathbf{C},\mathbf{P}_{\mathrm{L}},\mathbf{P}_{\mathrm{R}},\mathbf{U}^* \\ \{\mathbf{U}^{(v)},\mathbf{V}^{(v)},\beta^{(v)}\}}} \quad \mathcal{O}_{\alpha,\gamma,\mu} \\ \text{s.t.} \quad &\mathbf{Y} \in \{0,1\}^{M\times T}, \mathbf{Y}\,\mathbf{1} = \mathbf{1},\ \mathbf{U}^* \geq 0, \\ &\{\mathbf{U}^{(v)},\ \mathbf{V}^{(v)}\} \geq 0,\ \textstyle\sum_{v=0}^{d} e^{-\beta^{(v)}} = 1, \end{aligned} \tag{4.6}$$

where $\sum_{v=0}^{d} e^{-\beta^{(v)}} = 1$ is used for avoiding trivial solution, *i.e.*, solution which completely favors a certain view.

4.4.4 The ClusType Algorithm

The optimization problem in Eq. (4.6) is mix-integer programming and thus is NP-hard. We propose a two-step approximate solution: first solve the real-valued relaxation of Eq. (4.6) which is a non-convex problem with $\mathbf{Y} \in \mathbb{R}^{M\times T}$; then impose back the constraints to predict the exact type of each candidate mention $m_i \in \mathcal{M}_U$ by type $(m_i) = \text{argmax}\ Y_i$.

Directly solving the real-valued relaxation of Eq. (4.6) is not easy because it is non-convex. We develop an alternating minimization algorithm to optimize the problem with respect to each variable alternatively, which accomplishes two tasks iteratively: type propagation on the heterogeneous graph, and multi-view clustering of relation phrases.

First, to learn the type indicators of candidate entity mentions, we take derivative on $\mathcal{O}$ with respect to $\mathbf{Y}$ while fixing other variables. As links only exist between entity mentions sharing the same surface name in $\mathbf{W}_{\mathcal{M}}$, we can efficiently estimate $\mathbf{Y}$ with respect to each entity name $c \in \mathcal{C}$. Let $\mathbf{Y}^{(c)}$ and $\mathbf{S}^{(c)}_{\mathcal{M}}$ denote the sub-matrices of $\mathbf{Y}$ and $\mathbf{S}_{\mathcal{M}}$, which correspond to the candidate entity mentions with the name c, respectively. We have the update rule for $\mathbf{Y}^{(c)}$ as follows:

$$\mathbf{Y}^{(c)} = \big[(1+\gamma+\mu)\,\mathbf{I}_c - \gamma\,\mathbf{S}_{\mathcal{M}}^{(c)}\big]^{-1}\big(\,\mathbf{\Theta}^{(c)} + \mu\,\mathbf{Y}_0^{(c)}\,\big), \quad \forall c \in \mathcal{C}, \tag{4.7}$$

where $\mathbf{\Theta} = \Pi_{\mathcal{C}}\,\mathbf{C} + \Pi_L\,\mathbf{P}_{\mathrm{L}} + \Pi_R\,\mathbf{P}_{\mathrm{R}}$. Similarly, we denote $\Theta^{(c)}$ and $\mathbf{Y}_0^{(c)}$ as sub-matrices of Θ and $\mathbf{Y}_0$ which correspond to the candidate mentions with name c, respectively. It can be shown that $[(1+\gamma+\mu)\,\mathbf{I}_c - \gamma\,\mathbf{S}_{\mathcal{M}}^{(c)}]$ is positive definite given $\mu > 0$ and thus is invertible. Eq. (4.7) can be efficiently computed since the average number of mentions of an entity name is small (*e.g.*, < 10 in our experiments). One can further parallelize this step to reduce the computational time.

Second, to learn the type indicators of entity names and the type signatures of relation phrases, we take derivative on $\mathcal{O}$ with respect to $\mathbf{C}$, $\mathbf{P}_{\mathrm{L}}$ and $\mathbf{P}_{\mathrm{R}}$ while fixing other variables, leading to the following closed-form update rules.

$$\begin{aligned}
\mathbf{C} &= \frac{1}{2}\big[\,\mathbf{S}_L\,\mathbf{P}_{\mathrm{L}} + \mathbf{S}_R\,\mathbf{P}_{\mathrm{R}} + \Pi_{\mathcal{C}}^T(\mathbf{Y} - \Pi_L\,\mathbf{P}_{\mathrm{L}} - \Pi_R\,\mathbf{P}_{\mathrm{R}})\big]; \\
\mathbf{P}_{\mathrm{L}} &= \mathbf{X}_0^{-1}\,\big[\,\mathbf{S}_L^T\,\mathbf{C} + \Pi_L^T(\mathbf{Y} - \Pi_{\mathcal{C}}\,\mathbf{C} - \Pi_R\,\mathbf{P}_{\mathrm{R}}) + \beta^{(0)}\,\mathbf{U}^{(0)}\,\mathbf{V}^{(0)T}\,\big]; \\
\mathbf{P}_{\mathrm{R}} &= \mathbf{X}_1^{-1}\,\big[\,\mathbf{S}_R^T\,\mathbf{C} + \Pi_R^T(\mathbf{Y} - \Pi_{\mathcal{C}}\,\mathbf{C} - \Pi_L\,\mathbf{P}_{\mathrm{L}}) + \beta^{(1)}\,\mathbf{U}^{(1)}\,\mathbf{V}^{(1)T}\,\big];
\end{aligned} \tag{4.8}$$

where we define $\mathbf{X}_0 = [(1+\beta^{(0)})\,\mathbf{I}_l + \Pi_L^T\Pi_L]$ and $\mathbf{X}_1 = [(1+\beta^{(1)})\,\mathbf{I}_l + \Pi_R^T\Pi_R]$ respectively. Note that the matrix inversions in Eq. (4.8) can be efficiently calculated with linear complexity since both $\Pi_L^T\Pi_L$ and $\Pi_R^T\Pi_R$ are diagonal matrices.

Finally, to perform multi-view clustering, we first optimize Eq. (4.2) with respect to $\{\mathbf{U}^{(\mathrm{v})}, \mathbf{V}^{(\mathrm{v})}\}$ while fixing other variables, and then update $\mathbf{U}^*$ and $\{\beta^{(\mathrm{v})}\}$ by fixing $\{\mathbf{U}^{(\mathrm{v})}, \mathbf{V}^{(\mathrm{v})}\}$ and other variables, which follows the procedure in [81].

We first take the derivative of $\mathcal{O}$ with respect to $\mathbf{V}^{(\mathrm{v})}$ and apply Karush-Kuhn-Tucker complementary condition to impose the non-negativity constraint on it, leading to the multiplicative update rules as follows:

$$V_{jk}^{(v)} = V_{jk}^{(v)}\,\frac{[\mathbf{f}^{(v)T}\,\mathbf{U}^{(\mathrm{v})}]_{jk} + \alpha\sum_{i=1}^{l} U_{ik}^{*}U_{ik}^{(v)}}{\mathbf{\Delta}_{jk}^{(v)} + \alpha\big(\sum_{i=1}^{l} U_{ik}^{(v)2}\big)\big(\sum_{i=1}^{T} V_{ik}^{(v)}\big)}, \tag{4.9}$$

where we define the matrix $\mathbf{\Delta}^{(v)} = \mathbf{V}^{(\mathrm{v})}\,\mathbf{U}^{(v)T}\,\mathbf{U}^{(\mathrm{v})} + \mathbf{f}^{(v)^-}\,\mathbf{U}^{(\mathrm{v})}$. It is easy to check that $\{\mathbf{V}^{(\mathrm{v})}\}$ remains non-negative after each update based on Eq. (4.9).

We then normalize the column vectors of $\mathbf{V}^{(\mathrm{v})}$ and $\mathbf{U}^{(\mathrm{v})}$ by $\mathbf{V}^{(\mathrm{v})} = \mathbf{V}^{(\mathrm{v})}\,\mathbf{Q}^{(v)-1}$ and $\mathbf{U}^{(\mathrm{v})} = \mathbf{U}^{(\mathrm{v})}\,\mathbf{Q}^{(\mathrm{v})}$. Following similar procedure for updating $\mathbf{V}^{(\mathrm{v})}$, the update rule for $\mathbf{U}^{(\mathrm{v})}$ can be derived as follows:

$$U_{ik}^{(v)} = U_{ik}^{(v)}\,\frac{[\mathbf{f}^{(v)^+}\,\mathbf{V}^{(\mathrm{v})} + \alpha\,\mathbf{U}^{*}]_{ik}}{[\mathbf{U}^{(\mathrm{v})}\,\mathbf{V}^{(v)T}\,\mathbf{V}^{(\mathrm{v})} + \mathbf{f}^{(v)^-}\,\mathbf{V}^{(v)} + \alpha\,\mathbf{U}^{(\mathrm{v})}]_{ik}}. \tag{4.10}$$

In particular, we make the decomposition $\mathbf{f}^{(v)} = \mathbf{f}^{(v)^+} - \mathbf{f}^{(v)^-}$, where $A_{ij}^{+} = (|A_{ij}| + A_{ij})/2$ and $A_{ij}^{-} = (|A_{ij}| - A_{ij})/2$, in order to preserve the non-negativity of $\{\mathbf{U}^{(v)}\}$.

Algorithm 4.1 The **ClusType** algorithm

Input: biadjacency matrices $\{\Pi_{\mathcal{C}}, \Pi_L, \Pi_R, \mathbf{W}_L, \mathbf{W}_R, \mathbf{W}_{\mathcal{M}}\}$, clustering features $\{\mathbf{f}_s, \mathbf{f}_c\}$, seed labels $\mathbf{Y}_0$, number of clusters K, parameters $\{\alpha, \gamma, \mu\}$

Initialize $\{\mathbf{Y}, \mathbf{C}, \mathbf{P}_L, \mathbf{P}_R\}$ with $\{\mathbf{Y}_0, \Pi_{\mathcal{C}}^T \mathbf{Y}_0, \Pi_L^T \mathbf{Y}_0, \Pi_R^T \mathbf{Y}_0\}$, $\{\mathbf{U}^{(\mathrm{v})}, \mathbf{V}^{(\mathrm{v})}, \beta^{(\mathrm{v})}\}$ and $\mathbf{U}^*$ with positive values.
repeat
  Update candidate mention type indicator $\mathbf{Y}$ by Eq. (4.7)
  Update entity name type indicator $\mathbf{C}$ and relation phrase type signature $\{\mathbf{P}_L, \mathbf{P}_R\}$ by Eq. (4.8)
  for $v = 0$ to 3 **do**
    repeat
      Update $\mathbf{V}^{(\mathrm{v})}$ with Eq. (4.9)
      Normalize $\mathbf{U}^{(\mathrm{v})} = \mathbf{U}^{(\mathrm{v})}\,\mathbf{Q}^{(\mathrm{v})}$, $\mathbf{V}^{(\mathrm{v})} = \mathbf{V}^{(\mathrm{v})}\,\mathbf{Q}^{(v)-1}$
      Update $\mathbf{U}^{(\mathrm{v})}$ by Eq. (4.10)
    until Eq. (4.11) converges
  end for
  Update consensus matrix $\mathbf{U}^*$ and relative feature weights $\{\beta^{(\mathrm{v})}\}$ using Eq. (4.12)
until the objective $\mathcal{O}$ in Eq. (4.6) converges
Predict the type of $m_i \in \mathcal{M}_U$ by $\text{type}(m_i) = \text{argmax}\ Y_i$.

The proposed algorithm optimizes $\{\mathbf{U}^{(\mathrm{v})}, \mathbf{V}^{(\mathrm{v})}\}$ for each view v, by iterating between Eqs. (4.9) and (4.10) until the following reconstruction error converges.

$$\delta^{(v)} = \|\mathbf{f}^{(v)} - \mathbf{U}^{(\mathrm{v})}\,\mathbf{V}^{(v)T}\|_F^2 + \alpha\|\,\mathbf{U}^{(\mathrm{v})}\,\mathbf{Q}^{(\mathrm{v})} - \mathbf{U}^*\,\|_F^2 \tag{4.11}$$

With optimized $\{\mathbf{U}^{(\mathrm{v})}, \mathbf{V}^{(\mathrm{v})}\}$, we update $\mathbf{U}^*$ and $\{\beta^{(\mathrm{v})}\}$ by taking the derivative on $\mathcal{O}$ with respect to each of them while fixing all other variables. This leads to the closed-form update rules as follows:

$$\mathbf{U}^* = \frac{\sum_{v=0}^{d} \beta^{(\mathrm{v})}\,\mathbf{U}^{(\mathrm{v})}\,\mathbf{Q}^{(\mathrm{v})}}{\sum_{v=0}^{d} \beta^{(\mathrm{v})}}; \quad \beta^{(\mathrm{v})} = -\log\Big(\frac{\delta^{(v)}}{\sum_{i=0}^{d} \delta^{(i)}}\Big). \tag{4.12}$$

Algorithm 4.1 summarizes our algorithm. For convergence analysis, ClusType applies block coordinate descent on the real-valued relaxation of Eq. (4.6). The proof procedure in [82] (not included for lack of space) can be adopted to prove convergence for ClusType (to the local minimum).

4.4.5 Computational Complexity Analysis

Given a corpus $\mathcal{D}$ with $N_{\mathcal{D}}$ words, the time complexity for our candidate generation and generation of $\{\Pi_{\mathcal{C}}, \Pi_L, \Pi_R, \mathbf{f}_s, \mathbf{f}_c\}$ is $\mathcal{O}(N_{\mathcal{D}})$. For construction of the heterogeneous graph G, the costs for computing $G_{\mathcal{C},\mathcal{P}}$ and $G_{\mathcal{M}}$ are $\mathcal{O}(nl)$ and $\mathcal{O}(MM_{\mathcal{C}}d_{\mathcal{C}})$, respectively, where $M_{\mathcal{C}}$

denotes average number of mentions each name has and $d_{\mathcal{C}}$ denotes average size of feature dimensions ($M_{\mathcal{C}} < 10, d_{\mathcal{C}} < 5000$ in our experiments). It takes $\mathcal{O}(MT)$ and $\mathcal{O}(MM_{\mathcal{C}}^2 + l^2)$ time to initialize all the variables and pre-compute the constants in update rules, respectively.

We then study the computational complexity of ClusType in Algorithm 4.1 with the pre-computed matrices. In each iteration of the outer loop, ClusType costs $\mathcal{O}(MM_{\mathcal{C}}T)$ to update $\mathbf{Y}$, $\mathcal{O}(nlT)$ to update $\mathbf{C}$ and $\mathcal{O}\big(nT(K+l)\big)$ to update $\{\mathbf{P}_L, \mathbf{P}_R\}$. The cost for inner loop is $\mathcal{O}\big(t_{in}lK(T+n_s+n_c)\big)$ supposing it stops after t_{in} iterations ($t_{in} < 100$ in our experiments). Update of $\mathbf{U}^*$ and $\{\beta^{(\mathrm{v})}\}$ takes $\mathcal{O}(lK)$ time. Overall, the computational complexity of ClusType is $\mathcal{O}\big(t_{out}nlT + t_{out}t_{in}lK(T+n_s+n_c)\big)$, supposing that the outer loop stops in t_{out} iterations ($t_{out} < 10$ in our experiments).

4.5 EXPERIMENTS

4.5.1 Data Preparation

Our experiments use three real-world datasets[3]: (1) **NYT:** constructed by crawling 2013 news articles from New York Times. The dataset contains 118,664 articles (57M tokens and 480k unique words) covering various topics such as Politics, Business and Sports; (2) **Yelp:** We collected 230,610 reviews (25M tokens and 418k unique words) from the 2014 *Yelp dataset challenge*; and (3) **Tweet:** We randomly selected 10,000 users in Twitter and crawled at most 100 tweets for each user in May 2011. This yields a collection of 302,875 tweets (4.2M tokens and 157k unique words).

1. Heterogeneous Graphs. We first performed lemmatization on the tokens using NLTK WordNet Lemmatizer[4] to reduce variant forms of words (*e.g.*, eat, ate, eating) into their lemma form (*e.g.*, eat), and then applied Stanford POS tagger [83] on the corpus. In candidate generation (see Sec. 6.3.1), we set maximal pattern length as 5, minimum support as 30 and significance threshold as 2, to extract candidate entity mentions and relation phrases from the corpus. We then followed the introduction in Sec. 4.3 to construct the heterogeneous graph for each dataset. We used 5-nearest neighbor graphs when constructing the mention correlation subgraph. Table 6.5 summarizes the statistics of the constructed heterogeneous graphs for all three datasets.

2. Clustering Feature Generation. Following the procedure introduced in Sec. 4.4.2, we used a text window of 10 words to extract the context features for each relation phrase (5

[3]Code and datasets used in this paper can be downloaded at: `http://web.engr.illinois.edu/~xren7/clustype.zip`.

[4] `http://www.nltk.org/`

Table 4.4: Statistics of the heterogeneous graphs.

Data sets	**NYT**	**Yelp**	**Tweet**
#Entity mention candidates (M)	4.88M	1.32M	703k
#Entity surface names (n)	832k	195k	67k
#Relation phrases (l)	743k	271k	57k
#Links	29.32M	8.64M	3.59M
Avg#mentions per string name	5.86	6.78	10.56

Table 4.5: Target type sets $\mathcal{T}$ for the datasets.

NYT	*person*, *organization*, *location*, *time_event*
Yelp	*food*, *time_event*, *job_title*, *location*, *organization*
Tweet	*time_event*, *business_consumer_product*, *person*, *location*, *organization*, *business_job_title*, *time_year_of_day*

words on the left and the right of a relation phrase), where stop-words are removed. We obtained 56k string terms (n_s) and 129k context terms (n_c) for the NYT dataset, 58k string terms and 37k context terms for the Yelp dataset and 18k string terms and 38k context terms for the Tweet dataset, respectively all unique term counts. Each row of the feature matrices were then normalized by its ℓ-2 norm.

3. Seed and Evaluation Sets. For evaluation purposes, we selected entity types which are popular in the dataset from Freebase, to construct the target type set $\mathcal{T}$. Table 4.5 shows the target types used in the three datasets. To generate the set of seed mentions $\mathcal{M}_L$, we followed the process introduced in Sec. 4.4.1 by setting the confidence score threshold as $\eta = 0.8$. To generate the evaluation sets, we randomly selected a subset of documents from each dataset and annotated them using the target type set $\mathcal{T}$ (each entity mention is tagged by one type). 1k documents are annotated for the NYT dataset (25,451 annotated mentions). 2.5k reviews are annotated for the Yelp dataset (21,252 annotated mentions). 3k tweets are annotated for the Tweet dataset (5,192 annotated mentions). We removed the mentions from the seed mention sets if they were in the evaluation sets.

4.5.2 Experimental Settings

In our testing of ClusType and its variants, we set the number of clusters $K = \{4000, 1500, 300\}$ for NYT, Yelp and Tweet datasets, respectively, based on the analyses in Sec. 6.4.2. We set $\{\alpha, \gamma, \mu\} = \{0.4, 0.7, 0.5\}$ by five-fold cross validation (of classification accuracy) on the seed

Table 4.6: Performance comparisons on three datasets in terms of Precision, Recall and F1 score.

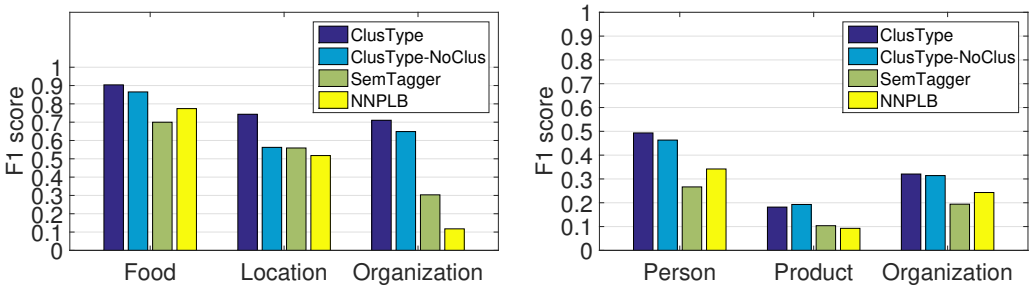
Data sets	**NYT**			**Yelp**			**Tweet**		
Method	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Pattern [31]	0.4576	0.2247	0.3014	0.3790	0.1354	0.1996	0.2107	0.2368	0.2230
FIGER [36]	0.8668	0.8964	0.8814	0.5010	0.1237	0.1983	**0.7354**	0.1951	0.3084
SemTagger [75]	0.8667	0.2658	0.4069	0.3769	0.2440	0.2963	0.4225	0.1632	0.2355
APOLLO [77]	0.9257	0.6972	0.7954	0.3534	0.2366	0.2834	0.1471	0.2635	0.1883
NNPLB [35]	0.7487	0.5538	0.6367	0.4248	0.6397	0.5106	0.3327	0.1951	0.2459
ClusType-NoClus	0.9130	0.8685	0.8902	0.7629	0.7581	0.7605	0.3466	0.4920	0.4067
ClusType-NoWm	0.9244	0.9015	0.9128	0.7812	0.7634	0.7722	0.3539	**0.5434**	0.4286
ClusType-TwoStep	0.9257	0.9033	0.9143	0.8025	0.7629	0.7821	0.3748	0.5230	0.4367
ClusType	**0.9550**	**0.9243**	**0.9394**	**0.8333**	**0.7849**	**0.8084**	0.3956	0.5230	**0.4505**

mention sets. For convergence criterion, we stop the outer (inner) loop in Algorithm 4.1 if the relative change of $\mathcal{O}$ in Eq. (4.6) $\big($reconstruction error in Eq. (4.11)$\big)$ is smaller than 10^{-4}, respectively.

Compared Methods: We compared the proposed method (ClusType) with its variants which only model part of the proposed hypotheses. Several state-of-the-art entity recognition approaches were also implemented (or tested using their published codes): (1) **Stanford NER** [25]: a CRF classifier trained on classic corpora for several major entity types; (2) **Pattern** [31]: a state-of-the-art pattern-based bootstrapping method which uses the seed mention sets $\mathcal{M}_L$; (3) **SemTagger** [75]: a bootstrapping method which trains contextual classifiers using the seed mention set $\mathcal{M}_L$ in a self-training manner; (4) **FIGER** [36]: FIGER trains sequence labeling models using automatically annotated Wikipeida corpora; (5) **NNPLB** [35]: It uses ReVerb assertions [41] to construct graphs and performs entity name-level label propagation; and (6) **APOLLO** [77]: APOLLO constructs heterogeneous graphs on entity mentions, Wikipedia concepts and KB entities, and then performs label propagation.

All compared methods were first tuned on our seed mention sets using five-fold cross validation. For ClusType, besides the proposed full-fledged model, **ClusType**, we compare (1) **ClusType-NoWm**: This variant does not consider mention correlation subgraph, *i.e.*, set $\gamma = 0$ in ClusType; (2) **ClusType-NoClus**: It performs only type propagation on the heterogeneous graph, *i.e.*, Eq. (4.4) is removed from $\mathcal{O}$; and (3) **ClusType-TwoStep**: It first conducts multi-view clustering to assign each relation phrase to a single cluster, and then performs ClusType-NoClus between entity names, candidate entity mentions and relation phrase clusters.

Evaluation Metrics: We use F1 score computed from Precision and Recall to evaluate the entity recognition performance. We denote the #system-recognized entity mentions as J and the # ground truth annotated mentions in the evaluation set as A. Precision is calculated by $\text{Prec} = \sum_{m \in J \cap A} \omega(t'_m = t_m)/|J|$ and Recall is calculated by $\text{Rec} = \sum_{m \in J \cap A} \omega(t'_m = t_m)/|A|$.



(a) Yelp (b) Tweet

Figure 4.6: Performance breakdown by types.

Here, t_m and t'_m denote the true type and the predicted type for m, respectively. Function $\omega(\cdot)$ returns 1 if the predicted type is correct and 0 otherwise. Only mentions which have correct boundaries and predicted types are considered correct. For cross validation on the seed mention sets, we use classification accuracy to evaluate the performance.

4.5.3 Experiments and Performance Study

1. Comparing ClusType with the other methods on entity recognition. Table 4.6 summarizes the comparison results on the three datasets. Overall, ClusType and its three variants outperform others on all metrics on NYT and Yelp and achieve superior Recall and F1 scores on Tweet. In particular, ClusType obtains a 46.08% improvement in F1 score and 168% improvement in Recall compared to the best baseline FIGER on the Tweet dataset and improves F1 by 48.94% compared to the best baseline, NNPLB, on the Yelp dataset.

FIGER utilizes a rich set of linguistic features to train sequence labeling models but suffers from low recall moving from a general domain (*e.g.*, NYT) to a noisy new domain (*e.g.*, Tweet) where feature generation is not guaranteed to work well (*e.g.*, 65% drop in F1 score). Superior performance of ClusType demonstrates the effectiveness of our candidate generation and of the proposed hypotheses on type propagation over domain-specific corpora. NNPLB also utilizes textual relation for type propagation, but it does not consider entity surface name ambiguity. APOLLO propagates type information between entity mentions but encounters severe context sparsity issue when using Wikipedia concepts. ClusType obtains superior performance because it not only uses semantic-rich relation phrases as type cues for each entity mention, but also clusters the synonymous relation phrases to tackle the context sparsity issues.

2. Comparing ClusType with its variants. Comparing with ClusType-NoClus and ClusType-TwoStep, ClusType gains performance from integrating relation phrase clustering with type

Table 4.7: F1 score comparison with trained NER.

Method	**NYT**	**Yelp**	**Tweet**
Stanford NER [25]	0.6819	0.2403	0.4383
ClusType-NoClus	0.9031	0.4522	0.4167
ClusType	**0.9419**	**0.5943**	**0.4717**

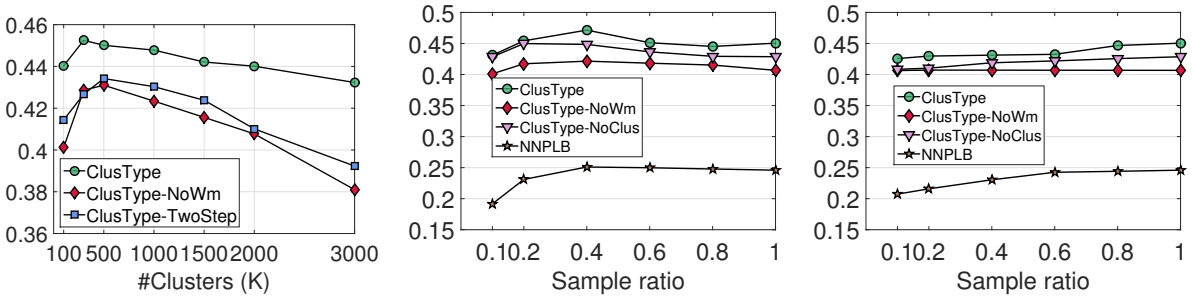
propagation in a mutually enhancing way. It always outperforms ClusType-NoWm on Precision and F1 on all three datasets. The enhancement mainly comes from modeling the mention correlation links, which helps disambiguate entity mentions sharing the same surface names.

3. Comparing on different entity types. Fig. 4.6 shows the performance on different types on Yelp and Tweet. ClusType outperforms all the others on each type. It obtains larger gain on *organization* and *person*, which have more entities with ambiguous surface names. This indicates that modeling types on entity mention level is critical for name disambiguation. Superior performance on *product* and *food* mainly comes from the domain independence of our method because both NNPLB and SemTagger require sophisticated linguistic feature generation which is hard to adapt to new types.

4. Comparing with trained NER. Table 4.7 compares ours with a traditional NER method, *Stanford NER*, trained using classic corpora like ACE corpus, on three major types—person, location and organization. ClusType and its variants outperform Stanford NER on the corpora which are dynamic (*e.g.*, NYT) or domain-specific (*e.g.*, Yelp). On the Tweet dataset, ClusType has lower Precision but achieves a 63.59% improvement in Recall and 7.62% improvement in F1 score. The superior Recall of ClusType mainly comes from the domain-independent candidate generation.

5. Testing on sensitivity over the number of relation phrase clusters, K. Fig. 4.7(a), ClusType was less sensitive to K compared with its variants. We found on the Tweet dataset, ClusType achieved the best performance when K=300 while its variants peaked at K=500, which indicates that better performance can be achieved with fewer clusters if type propagation is integrated with clustering in a mutually enhancing way. On the NYT and the Yelp datasets (not shown here), ClusType peaked at K=4000 and K=1500, respectively.

6. Testing on the size of seed mention set. Seed mentions are used as labels (distant supervision) for typing other mentions. By randomly selecting a subset of seed mentions as labeled data (sampling ratio from 0.1 to 1.0), Fig. 4.7(b) shows ClusType and its variants are not very sensitive to the size of seed mention set. Interestingly, using all the seed mentions



(a) #Clusters (b) Seed set size (c) Corpus size

Figure 4.7: Performance changes in F1 score with #clusters, #seeds and corpus size on Tweets.

does not lead to the best performance, likely caused by the type ambiguity among the mentions.

7. Testing on the effect of corpus size. Experimenting on the same parameters for candidate generation and graph construction, Fig. 4.7(c) shows the performance trend when varying the sampling ratio (subset of documents randomly sampled to form the input corpus). ClusType and its variants are not very sensitive to the changes of corpus size, but NNPLB had over 17% drop in F1 score when sampling ratio changed from 1.0 to 0.1 (while ClusType had only 5.5%). In particular, they always outperform FIGER, which uses a trained classifier and thus does not depend on corpus size.

4.6 RELATED WORK

Entity Recognition. Existing work leverages various levels of human supervision to recognize entities, from fully annotated documents (supervised), seed entities (weakly supervised), to knowledge bases (distantly supervised).

Traditional supervised methods [26, 44] use fully annotated documents and different linguistic features to train sequence labeling model (*e.g.*, CRF classifier). To obtain an effective model, the amount of labeled data is significant [26], despite the effort of semi-supervised sequence labeling methods [84, 85, 32].

Weakly-supervised methods utilize a small set of typed entities as seeds and extract more entities of target types, which can largely reduce the amount of required labeled data. Pattern-based bootstrapping methods [31, 53, 86] derive patterns from contexts of seed entities and use them to incrementally extract new entities and new patterns, which do not restrict to specific domains. However, assigning a single type to each entity name may cause semantic drift [36]. Also, such methods are constrained to the information matched

by the pattern and often suffer from recall [87]. Iterative bootstrapping methods such as probabilistic method [51] and label propagation method [88, 87, 54] softly assign multiple types to an entity name and iteratively update its type distribution. In particular, Web data (*e.g.*, Web lists, query logs) has been explored to conduct set expansion based on seed entities [89, 30, 90]. However, above methods simply derive a global type distribution for entity name and thus cannot decide the exact type for each entity mention based on its local context. Also, careful seed selection by human is required to ensure effective entity extraction [76].

Distantly supervised (unsupervised) methods [34, 35, 36] avoid expensive human labels by leveraging type information of entity mentions which are confidently mapped to entries in KBs. Linked mentions are used to type those unlinkable ones in different ways, including training a contextual classifier [34], learning a sequence labeling model [36] and serving as labels in graph-based semi-supervised learning [35]. In particular, Li *et al.* [91] conduct domain-specific, unsupervised entity recognition on tweet data based on Web data.

Aforementioned methods adopt general entity detection tools (*e.g.*, noun phrase chunker) to extract candidates and rely on linguistic processing (*e.g.*, dependency parser) to generate features, which may be hard to generalize across different domains. Also, they overlook entity name ambiguity and context sparsity issues when modeling type of entity mention by its contextual information (*e.g.*, keyword, context sequence, Wikipedia concept).

Our work uses a domain-independent phrase mining algorithm to generate candidate mentions, models the exact type for each entity mention based on its string name and surrounding relation phrases to handle name ambiguity, and conduct clustering-integrated type propagation to resolve context sparsity. Knowledge base population methods [92, 77] study entity linking and fine-grained categorization of unlinkable mentions in a unified framework, which shares the similar idea of modeling each entity mention individually to resolve name ambiguity. Our work is also related to noun phrase chunking [93] and keyphrase extraction [78] in terms of extracting noun phrase or significant phrases from corpus, but we focus on extracting candidate entity mentions which satisfy POS constraints and relation phrases in a joint manner.

Open Relation Mining. Extracting textual relation between subjective and objective from text has been extensively studied [41, 94] and applied to entity typing [35]. Fader *et al.* [41] utilize POS patterns to extract verb phrases between detected noun phrases to form relation assertion. Schmitz *et al.* [42] further extend the textual relation by leveraging dependency tree patterns. However, these methods reply on linguistic parsers, which may not generalize to different domains, and do not consider significance of the detected entity mentions in the

Table 4.8: Example output of ClusType and the compared methods on the Yelp dataset.

ClusType	SemTagger	NNPLB
The best **BBQ:Food** I've tasted in **Phoenix:LOC** ! I had the **[pulled pork sandwich]:Food** with **coleslaw:Food** and **[baked beans]:Food** for lunch. ...	The best BBQ I've tasted in **Phoenix:LOC** ! I had the pulled **[pork sandwich]:LOC** with **coleslaw:Food** and **[baked beans]:LOC** for lunch. ...	The best **BBQ:Loc** I've tasted in **Phoenix:LOC** ! I had the pulled pork **sandwich:Food** with coleslaw and baked **beans:Food** for **lunch:Food**. ...
I only go to **ihop:LOC** for **pancakes:Food** because I don't really like anything else on the menu. Ordered **[chocolate chip pancakes]:Food** and a **[hot chocolate]:Food**.	I only go to ihop for pancakes because I don't really like anything else on the menu. Ordered **[chocolate chip pancakes]:LOC** and a **[hot chocolate]:LOC**.	I only go to ihop for pancakes because I don't really like anything else on the menu. Ordered chocolate chip pancakes and a hot chocolate.

corpus (see comparison between NNPLB [35] and ClusType).

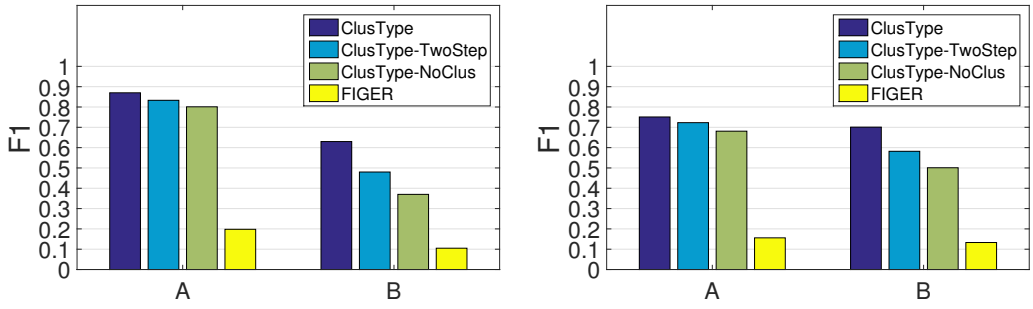
On the other hand, there has been some studies on clustering synonymous relations generated by open information extraction, and forming canonicalized relation phrase for each cluster [67, 80, 95]. However, these methods either ignore entity type information when resolving relations, or assume types of relation arguments are already given. Our work integrate entity type learning and relation phrase clustering in a joint framework where these two tasks can mutually enhance each other.

Liu *et al.* [81] formulate multi-view clustering based on joint non-negative matrix factorization. Ji *et al.* [96] propose a graph-based semi-supervised learning method for transductive classification on heterogeneous information network. To our knowledge, the proposed method is the first to model graph-based semi-supervised learning and multi-view clustering in a joint optimization problem, leading to a multi-task multi-view learning framework.

4.7 DISCUSSION

1. Example output on two Yelp reviews. Table 6.9 shows the output of ClusType, SemTagger and NNPLB on two Yelp reviews: ClusType extracts more entity mention candidates (*e.g.*, "*BBQ*", "*ihop*") and predicts their types with better accuracy (*e.g.*, "*baked beans*", "*pulled pork sandwich*").

2. Testing on context sparsity. The type indicator of each entity mention candidate is modeled in ClusType based on the type indicator of its surface name and the type signatures of its co-occurring relation phrases. To test the handling of different relation phrase sparsity,



(a) Context sparsity

(b) Surface name popularity

Figure 4.8: Case studies on context sparsity and surface name popularity on the Tweet dataset.

two groups of 500 mentions are selected from Yelp: mentions in *Group A* co-occur with frequent relation phrases (∼4.6k occurrences in the corpus) and those in *Group B* co-occur with sparse relation phrases (∼3.4 occurrences in the corpus). Fig. 4.8(a) compares their F1 scores on the Tweet dataset. In general, all methods obtained better performance when mentions co-occurring with frequent relation phrases than with sparse relation phrases. In particular, we found that ClusType and its variants had comparable performance in Group A but ClusType obtained superior performance in Group B. Also, ClusType-TwoStep obtained larger performance gain over ClusType-NoClus in Group B. This indicates that clustering relation phrases is critical for performance enhancement when dealing with sparse relation phrases, as expected.

3. Testing on surface name popularity. We generated the mentions in Group A with high frequency surface names (∼2.7k occurrences) and those in *Group B* with infrequent surface names (∼1.5). Fig. 4.8(b) shows the degraded performance of all methods in both cases—likely due to ambiguity in popular mentions and sparsity in infrequent mentions. ClusType outperforms its variants in Group B, showing it handles well mentions with insufficient corpus statistics.

4. Example relation phrase clusters. Table 4.9 shows relation phrases along with their corpus frequency from three example relation phrase clusters for the NYT dataset (K =

Table 4.9: Example relation phrase clusters and their corpus frequency from the NYT dataset.

ID	Relation phrase
1	recruited by (5.1k); employed by (3.4k); want hire by (264)
2	go against (2.4k); struggling so much against (54); run for re-election against (112); campaigned against (1.3k)
3	looking at ways around (105); pitched around (1.9k); echo around (844); present at (5.5k);

4000). We found that not only synonymous relation phrases, but also both sparse and frequent relation phrases can be clustered together effectively (*e.g.*, "*want hire by*" and "*recruited by*"). This shows that ClusType can boost sparse relation phrases with type information from the frequent relation phrases with similar group memberships.

4.8 SUMMARY

Entity recognition is an important but challenging research problem. In reality, many text collections are from specific, dynamic, or emerging domains, which poses significant new challenges for entity recognition with increase in name ambiguity and context sparsity, requiring entity detection without domain restriction. In this work, we investigate entity recognition (ER) with distant-supervision and propose a novel relation phrase-based ER framework, called **ClusType**, that runs *data-driven* phrase mining to generate entity mention candidates and relation phrases, and enforces the principle that relation phrases should be *softly* clustered when propagating type information between their argument entities. Then we predict the type of *each* entity mention based on the type signatures of its co-occurring relation phrases and the type indicators of its surface name, as computed over the corpus. Specifically, we formulate a joint optimization problem for two tasks, *type propagation with relation phrases* and *multi-view relation phrase clustering*. Our experiments on multiple genres—news, Yelp reviews and tweets—demonstrate the effectiveness and robustness of ClusType, with an average of 37% improvement in F1 score over the best compared method.

CHAPTER 5: FINE-GRAINED ENTITY TYPING WITH KNOWLEDGE BASES

5.1 PROPOSED METHOD: OVERVIEW AND MOTIVATION

Assigning types (*e.g.*, `person`, `organization`) to mentions of entities in context is an important task in natural language processing (NLP). The extracted entity type information can serve as primitives for relation extraction [37] and event extraction [97], and assists a wide range of downstream applications including knowledge base (KB) completion [1], question answering [35] and entity recommendation [98]. While traditional named entity recognition systems [26, 44] focus on a small set of coarse types (typically fewer than 10), recent studies [36, 99] work on a much larger set of fine-grained types (usually over 100) which form a tree-structured hierarchy (see the blue region of Fig. 6.1). Fine-grained typing allows one mention to have multiple types, which together constitute a *type-path* (not necessarily ending in a leaf node) in the given type hierarchy, *depending on the local context* (*e.g.*, sentence). Consider the example in Fig. 6.1, "*Arnold Schwarzenegger*" could be labeled as {`person`, `businessman`} in S3 (investment). But he could also be labeled as {`person`, `politician`} in S1 or {`person`, `artist`, `actor`} in S2. Such fine-grained type representation provides more informative features for other NLP tasks. For example, since relation and event extraction pipelines rely on entity recognizer to identify possible arguments in a sentence, fine-grained argument types help distinguish hundreds or thousands of different relations and events [36].

Traditional named entity recognition systems adopt manually annotated corpora as training data [44]. But the process of manually labeling a training set with large numbers of fine-grained types is too expensive and error-prone (hard for annotators to distinguish over 100 types consistently). Current fine-grained typing systems annotate training corpora automatically using knowledge bases (*i.e.*, *distant supervision*) [36, 100]. A typical workflow of distant supervision is as follows (see Fig. 6.1): (1) identify entity mentions in the documents; (2) link mentions to entities in KB; and (3) assign, to the candidate type set of each mention, all KB types of its KB-linked entity. However, existing distant supervision methods encounter the following limitations when doing automatic fine-grained typing.

• **Noisy Training Labels.** Current practice of distant supervision may introduce *label noise* to training data since it fails to take a mention's local contexts into account when assigning type labels (*e.g.*, see Fig. 6.1). Many previous studies ignore the label noises which appear in a majority of training mentions (see Table. 6.1, row (1)), and assume *all* types obtained by distant supervision are "correct" [58, 36]. The noisy labels may mislead the trained models and cause negative effect. A few systems try to denoise the training corpora using simple

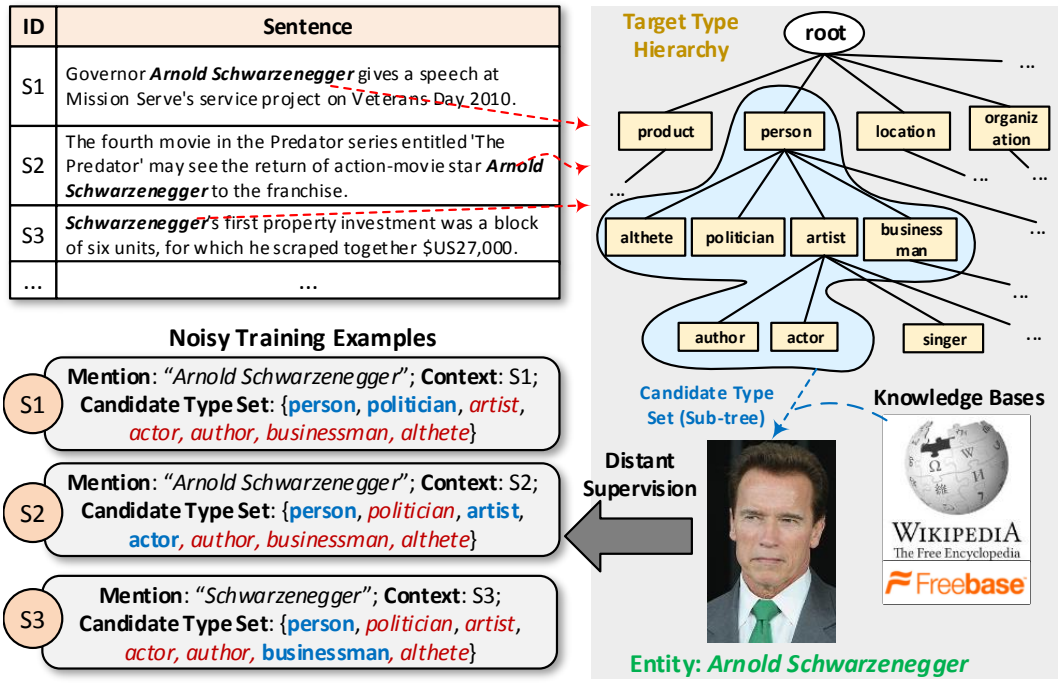


Figure 5.1: Current systems may detect *Arnold Schwarzenegger* in sentences S1-S3 and assign the same types to all (listed within braces), when only some types are correct for context (blue labels within braces).

pruning heuristics such as deleting mentions with conflicting types [101]. However, such strategies significantly reduce the size of training set (Table 6.1, rows (2a-c)) and lead to performance degradation (later shown in our experiments). The larger the target type set, the more severe the loss.

• **Type Correlation.** Most existing methods [58, 36] treat every type label in a training mention's candidate type set *equally* and *independently* when learning the classifiers but ignore the fact that types in the given hierarchy are semantically correlated (*e.g.*, `actor` is more relevant to `singer` than to `politician`). As a consequence, the learned classifiers may bias toward popular types but perform poorly on infrequent types since training data on infrequent types is scarce. Intuitively, one should pose smaller penalty on types which are semantically more relevant to the true types. For example, in Fig. 6.1 `singer` should receive a smaller penalty than `politician` does, by knowing that `actor` is a true type for "*Arnold Schwarzenegger*" in S2. This provides classifiers with additional information to distinguish between two types, especially those infrequent ones.

In this paper, we approach the problem of *automatic fine-grained entity typing* as follows: (1) Use different objectives to model training mentions with correct type labels and mentions with noisy labels, respectively. (2) Design a novel *partial-label loss* to model true types within

Dataset	Wiki	OntoNotes	BBN	NYT
# of target types	113	89	47	446
(1) noisy mentions (%)	27.99	25.94	22.32	51.81
(2a) sibling pruning (%)	23.92	16.09	22.32	39.26
(2b) min. pruning (%)	28.22	8.09	3.27	32.75
(2c) all pruning (%)	45.99	23.45	25.33	61.12

Table 5.1: A study of label noise. (1): %mentions with multiple *sibling types* (*e.g.*, `actor`, `singer`); (2a)-(2c): %mentions deleted by the three pruning heuristics [101] (see Sec. 6.4), for three experiment datasets and New York Times annotation corpus [102].

the noisy candidate type set which requires only the "*best*" candidate type to be relevant to the training mention, and progressively estimate the best type by leveraging various text features extracted for the mention. (3) Derive type correlation based on two signals: (i) the given type hierarchy, and (ii) the shared entities between two types in KB, and incorporate the correlation so induced by enforcing *adaptive margins* between different types for mentions in the training set. To integrate these ideas, we develop a novel embedding-based framework called **AFET**. First, it uses distant supervision to obtain candidate types for each mention, and extract a variety of text features from the mentions themselves and their local contexts. Mentions are partitioned into a "clean" set and a "noisy" set based on the given type hierarchy. Second, we embed mentions and types jointly into a low-dimensional space, where, in that space, objects (*i.e.*, features and types) that are semantically close to each other also have similar representations. In the proposed objective, an adaptive margin-based rank loss is proposed to model the set of clean mentions to capture type correlation, and a partial-label rank loss is formulated to model the "best" candidate type for each noisy mention. Finally, with the learned embeddings (*i.e.*, mapping matrices), one can predict the type-path for each mention in the test set in a top-down manner, using its text features. The major contributions of this paper are as follows:

1. We propose an automatic fine-grained entity typing framework, which reduces label noise introduced by distant supervision and incorporates type correlation in a principle way.

2. A novel optimization problem is formulated to jointly embed entity mentions and types to the same space. It models noisy type set with a partial-label rank loss and type correlation with adaptive-margin rank loss.

3. We develop an iterative algorithm for solving the joint optimization problem efficiently.

4. Experiments with three public datasets demonstrate that AFET achieves significant improvement over the state of the art.

5.2 PRELIMINARIES

Our task is to automatically uncover the type information for entity mentions (*i.e.*, token spans representing entities) in natural language sentences. The task takes a document collection $\mathcal{D}$ (automatically labeled using a KB Ψ in conjunction with a target type hierarchy $\mathcal{Y}$) as input and predicts a type-path in $\mathcal{Y}$ for each mention from the test set $\mathcal{D}_t$.

Type Hierarchy and Knowledge Base. Two key factors in distant supervision are the target type hierarchy and the KB. A *type hierarchy*, $\mathcal{Y}$, is a tree where nodes represent types of interests from Ψ. Previous studies manually create several clean type hierarchies using types from Freebase [36] or WordNet [99]. In this study, we adopt the existing hierarchies constructed using Freebase types[1]. To obtain types for entities $\mathcal{E}_\Psi$ in Ψ, we use the human-curated entity-type facts in Freebase, denoted as $\mathcal{F}_\Psi = \big\{(e, y)\big\} \subset \mathcal{E}_\Psi \times \mathcal{Y}$.

Automatically Labeled Training Corpora. There exist publicly available labeled corpora such as Wikilinks [103] and ClueWeb [104]. In these corpora, entity mentions are identified and mapped to KB entities using anchor links. In specific domains (*e.g.*, product reviews) where such public corpora are unavailable, one can utilize distant supervision to automatically label the corpus [36]. Specifically, an entity linker will detect mentions m_i and map them to one or more entity e_i in $\mathcal{E}_\Psi$. Types of e_i in KB are then associated with m_i to form its type set $\mathcal{Y}_i$, *i.e.*, $\mathcal{Y}_i = \big\{y \mid (e_i, y) \in \mathcal{F}_\Psi,\ y \in \mathcal{Y}\big\}$. Formally, a training corpus $\mathcal{D}$ consists of a set of extracted *entity mentions* $\mathcal{M} = \{m_i\}_{i=1}^N$, the *context* (*e.g.*, sentence, paragraph) of each mention $\{c_i\}_{i=1}^N$, and the candidate *type sets* $\{\mathcal{Y}_i\}_{i=1}^N$ for each mention. We represent $\mathcal{D}$ using a set of triples $\mathcal{D} = \big\{(m_i, c_i, \mathcal{Y}_i)\big\}_{i=1}^N$.

Problem Description. For each test mention, we aim to predict the correct type-path in $\mathcal{Y}$ based on the *mention's context*. More specifically, the test set $\mathcal{T}$ is defined as a set of mention-context pairs (m, c), where mentions in $\mathcal{T}$ (denoted as $\mathcal{M}_t$) are extracted from their sentences using existing extractors such as named entity recognizer [25]. We denote the gold type-path for a test mention m as $\mathcal{Y}^*$. This work focuses on learning a typing model from the noisy training corpus D, and estimating $\mathcal{Y}^*$ from $\mathcal{Y}$ for each test mention m (in set $\mathcal{M}_t$), based on mention m, its context c, and the learned model.

[1]We use the Freebase dump as of 2015-06-30.

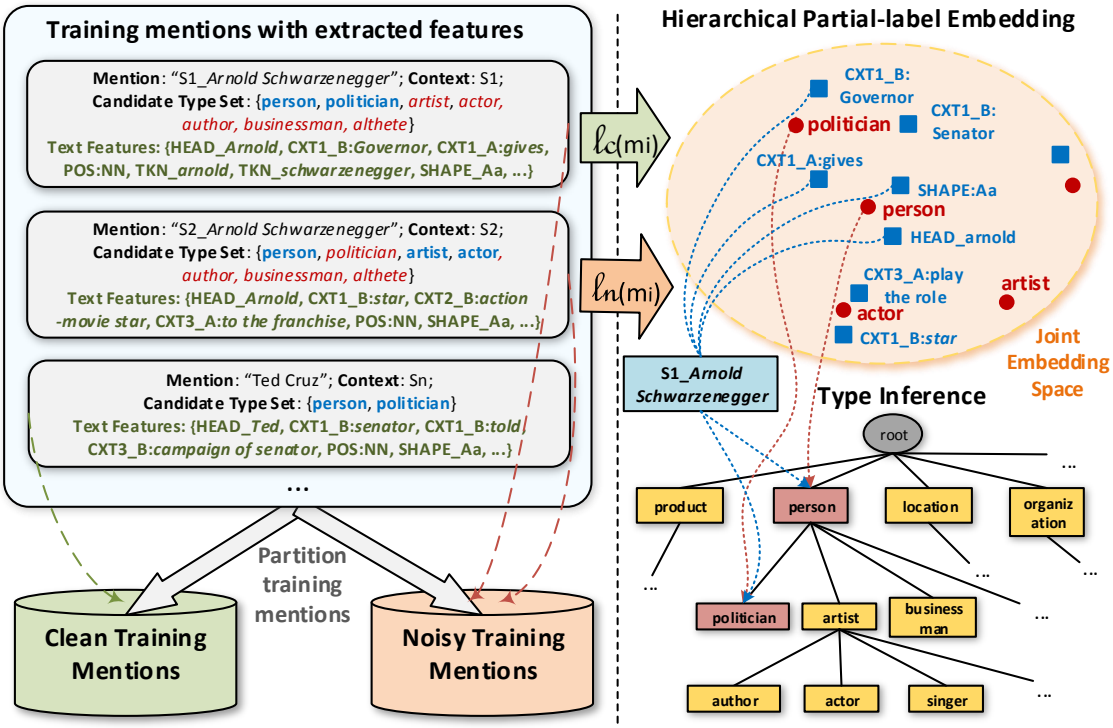


Figure 5.2: Framework Overview of AFET.

Framework Overview. At a high level, the AFET framework (see also Fig. 6.3) learns low-dimensional representations for entity types and text features, and infers type-paths for test mentions using the learned embeddings. It consists of the following steps:

1. Extract text features for entity mentions in training set $\mathcal{M}$ and test set $\mathcal{M}_t$ using their surface names as well as the contexts. (Sec. 5.3.1).

2. Partition training mentions $\mathcal{M}$ into a clean set (denoted as $\mathcal{M}_c$) and a noisy set (denoted as $\mathcal{M}_n$) based on their candidate type sets (Sec. 5.3.2).

3. Perform joint embedding of entity mentions $\mathcal{M}$ and type hierarchy $\mathcal{Y}$ into the same low-dimensional space where, in that space, close objects also share similar types (Secs. 5.3.3-5.3.6).

4. For each test mention m, estimate its type-path $\mathcal{Y}^*$ (on the hierarchy $\mathcal{Y}$) in a top-down manner using the learned embeddings (Sec. 5.3.6).

Feature	Description	Example
Head	Syntactic head token of the mention	"HEAD_Turing"
Token	Tokens in the mention	"Turing", "Machine"
POS	Part-of-Speech tag of tokens in the mention	"NN"
Character	All character trigrams in the head of the mention	":tu", "tur", ..., "ng:"
Word Shape	Word shape of the tokens in the mention	"Aa" for "Turing"
Length	Number of tokens in the mention	"2"
Context	Unigrams/bigrams before and after the mention	"CXT_B:Maserati ,", "CXT_A:and the"
Brown Cluster	Brown cluster ID for the head token (learned using $\mathcal{D}$)	"4_1100", "8_1101111"
Dependency	Stanford syntactic dependency [105] associated with the head token	"GOV:nn", "GOV:turing"

Table 5.2: Text features used in this paper. "*Turing Machine*" is used as an example mention from "*The bands former drummer Jerry Fuchs—who was also a member of Maserati,* ***Turing Machine*** *and The Juan MacLean—died after falling down an elevator shaft.*".

5.3 THE AFET FRAMEWORK

This section introduces the proposed framework and formulates an optimization problem for learning embeddings of text features and entity types jointly.

5.3.1 Text Feature Generation

We start with a representation of entity mentions. To capture the shallow syntax and distributional semantics of a mention $m_i \in \mathcal{M}$, we extract various features from both m_i itself and its context c_i. Table 6.3 lists the set of text features used in this work, which is similar to those used in [58, 36]. We denote the set of M unique features extracted from $\mathcal{D}$ as $\mathcal{F} = \{f_j\}_{j=1}^{M}$.

5.3.2 Training Set Partition

A training mention m_i (in set $\mathcal{M}$) is considered as a "clean" mention if its candidate type set obtained by distant supervision (*i.e.*, $\mathcal{Y}_i$) is not ambiguous, *i.e.*, candidate types in $\mathcal{Y}_i$ can form a *single* path in tree $\mathcal{Y}$. Otherwise, a mention is considered as "noisy" mention if its candidate types form *multiple* type-paths in $\mathcal{Y}$. Following the above hypothesis, we judge each mention m_i (in set $\mathcal{M}$) and place it in either the "clean" set $\mathcal{M}_c$, or the "noisy" set $\mathcal{M}_n$. Finally, we have $\mathcal{M} = \mathcal{M}_c \cup \mathcal{M}_n$.

5.3.3 The Joint Mention-Type Model

We propose to learn mappings into low-dimensional vector space, where, both entity mentions and type labels (in the training set) are represented, and in that space, *two objects are embedded close to each other if and only if they share similar types*. In doing so, we later can derive the representation of a test mention based on its text features and the learned mappings. Mapping functions for entity mentions and entity type labels are different as they have different representations in the *raw* feature space, but are jointly learned by optimizing a global objective of interests to handle the aforementioned challenges.

Each entity mention $m_i \in M$ can be represented by a M-dimensional feature vector $\mathbf{m}_i \in \mathbb{R}^M$, where $m_{i,j}$ is the number of occurrences of feature f_j (in set $\mathcal{F}$) for m_i. Each type label $y_k \in \mathcal{Y}$ is represented by a K-dimensional binary indicator vector $\mathbf{y}_k \in \{0,1\}^K$, where $\mathbf{y}_{k,k} = 1$, and 0 otherwise.

Specifically, we aim to learn a mapping function from the mention's feature space to a low-dimensional vector space, *i.e.*, $\Phi_{\mathcal{M}}(\mathbf{m}_i) : \mathbb{R}^M \mapsto \mathbb{R}^d$ and a mapping function from type label space to the same low-dimensional space, *i.e.*, $\Phi_{\mathcal{Y}}(\mathbf{y}_k) : \mathbb{R}^K \mapsto \mathbb{R}^d$. In this work, we adopt linear maps, as similar to the mapping functions used in [106].

$$\Phi_{\mathcal{M}}(\mathbf{m}_i) = \mathbf{U}\,\mathbf{m}_i; \quad \Phi_{\mathcal{Y}}(\mathbf{y}_k) = \mathbf{V}\,\mathbf{y}_k, \tag{5.1}$$

where $\mathbf{U} \in \mathbb{R}^{d \times M}$ and $\mathbf{V} \in \mathbb{R}^{d \times K}$ are the projection matrices for mentions and type labels, respectively.

5.3.4 Modeling Type Correlation

In type hierarchy (tree) $\mathcal{Y}$, types closer to each other (*i.e.*, shorter path) tend to be more related (*e.g.*, `actor` is more related to `artist` than to `person` in the right column of Fig. 6.3). In KB Ψ, types assigned to similar sets of entities should be more related to each other than those assigned to quite different entities [107] (*e.g.*, `actor` is more related to `director` than to `author` in the left column of Fig. 5.3). Thus, type correlation between y_k and $y_{k'}$ (denoted as $w_{kk'}$) can be measured either using the *one over the length of shortest path in* $\mathcal{Y}$, or using the *normalized number of shared entities in KB*, which is defined as follows.

$$w_{kk'} = \Big(\big|\mathcal{E}_k \cap \mathcal{E}_{k'}\big|/\big|\mathcal{E}_k\big| + \big|\mathcal{E}_k \cap \mathcal{E}_{k'}\big|/\big|\mathcal{E}_{k'}\big|\Big)/2. \tag{5.2}$$

Although a shortest path is efficient to compute, its accuracy is limited—It is not always true that a type (*e.g.*, `athlete`) is more related to its parent type (*i.e.*, `person`) than to its

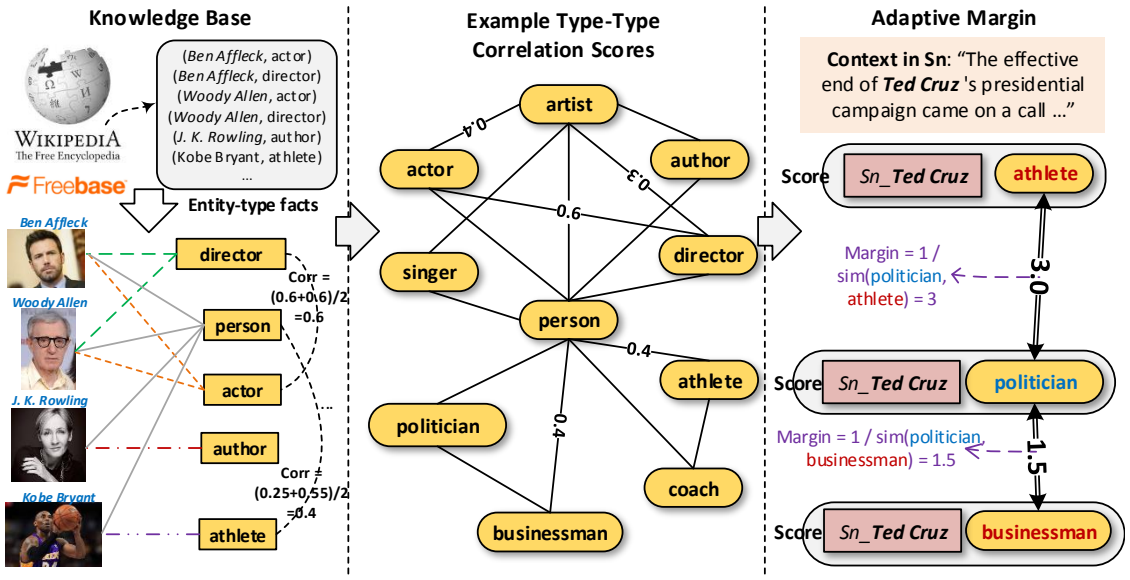


Figure 5.3: An illustration of KB-based type correlation computation, and the proposed adaptive margin.

sibling types (*e.g.*, `coach`), or that all sibling types are equally related to each other (*e.g.*, `actor` is more related to `director` than to `author`). We later compare these two methods in our experiments.

With the type correlation computed, we propose to apply *adaptive* penalties on different negative type labels (for a training mention), instead of treating all of the labels *equally* as in most existing work [106]. The hypothesis is intuitive: given the positive type labels for a mention, we force the negative type labels which are related to the positive type labels to receive smaller penalty. For example, in the right column of Fig. 5.3, negative label `businessman` receives a smaller penalty (*i.e.*, margin) than `athele` does, since `businessman` is more related to `politician`.

Hypothesis 5.1: Adaptive Margin

For a mention, if a negative type is correlated to a positive type, the margin between them should be smaller.

We propose an adaptive-margin rank loss to model the set of "clean" mentions (*i.e.*, $\mathcal{M}_c$), based on the above hypothesis. The intuition is simple: for each mention, rank all the positive types ahead of negative types, where the ranking score is measured by similarity between mention and type. We denote $f_k(m_i)$ as the similarity between (m_i, y_k) and is defined

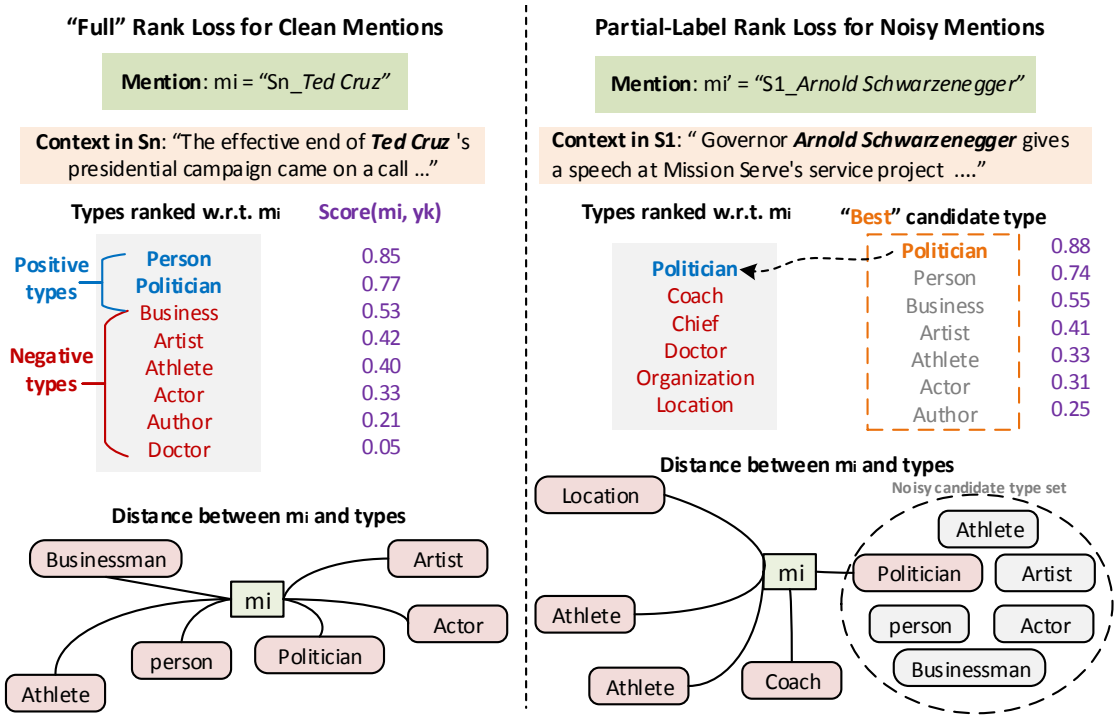


Figure 5.4: An illustration of the partial-label rank loss.

as the inner product of $\Phi_{\mathcal{M}}(\mathbf{m}_i)$ and $\Phi_{\mathcal{Y}}(\mathbf{y}_k)$.

$$\ell_c(m_i, \mathcal{Y}_i, \overline{\mathcal{Y}}_i) = \sum_{y_k \in \mathcal{Y}_i} \sum_{y_{\bar{k}} \in \overline{\mathcal{Y}}_i} L\Big\lfloor \text{rank}_{y_k}\Big(f(m_i)\Big)\Big\rfloor \Theta_{i,k,\bar{k}};$$

$$\Theta_{i,k,\bar{k}} = \max\Big\{0, \gamma_{k,\bar{k}} - f_k(m_i) + f_{\bar{k}}(m_i)\Big\};$$

$$\text{rank}_{y_k}\Big(f(m_i)\Big) = \sum_{y_{\bar{k}} \in \overline{\mathcal{Y}}_i} \mathbf{1}\Big(\gamma_{k,\bar{k}} + f_{\bar{k}}(m_i) > f_k(m_i)\Big).$$

Here, $\gamma_{k,\bar{k}}$ is the adaptive margin between positive type k and negative type $\bar{k}$, which is defined as $\gamma_{k,\bar{k}} = 1 + 1/(w_{k,\bar{k}} + \alpha)$ with a smooth parameter α. $L(x) = \sum_{i=1}^{x} \frac{1}{i}$ transforms rank to a weight, which is then multiplied to the max-margin loss $\Theta_{i,k,\bar{k}}$ to optimize precision at x [106].

5.3.5 Modeling Noisy Type Labels

True type labels for noisy entity mentions $\mathcal{M}_n$ (i.e., mentions with ambiguous candidate types in the given type hierarchy) in each sentence are not available in knowledge bases. To effectively model the set of noisy mentions, we propose not to treat *all* candidate types (*i.e.*, $\{\mathcal{Y}_i\}$ as true labels. Instead, we model the "true" label among the candidate set as *latent*

value, and try to infer that using text features.

Hypothesis 5.2: Partial-Label Loss

For a noisy mention, the maximum score associated with its candidate types should be greater than the scores associated with any other non-candidate types

We extend the partial-label loss in [66] (used to learn linear classifiers) to enforce Hypothesis 5.3.5, and integrate with the adaptive margin to define the loss for m_i (in set $\mathcal{M}_n$).

$$\ell_n(m_i, \mathcal{Y}_i, \overline{\mathcal{Y}}_i) = \sum_{\bar{k} \in \overline{\mathcal{Y}}_i} L\Big\lfloor \text{rank}_{y_{k^*}}\Big(f(m_i)\Big)\Big\rfloor \Omega_{i,\bar{k}};$$

$$\Omega_{i,k} = \max\Big\{0, \gamma_{k^*,\bar{k}} - f_{k^*}(m_i) + f_{\bar{k}}(m_i)\Big\};$$

$$\text{rank}_{y_{k^*}}\Big(f(m_i)\Big) = \sum_{y_{\bar{k}} \in \overline{\mathcal{Y}}_i} \mathbf{1}\Big(\gamma_{k^*,\bar{k}} + f_{\bar{k}}(m_i) > f_{k^*}(m_i)\Big)$$

where we define . $y_{k^*} = \text{argmax}_{y_k \in \mathcal{Y}_i} f_k(m_i)$ and $y_{\bar{k}^*} = \text{argmax}_{y_k \in \overline{\mathcal{Y}}_i} f_k(m_i)$.

Minimizing ℓ_n encourages a *large margin* between the maximum scores $\max_{y_k \in \mathcal{Y}_i} f_{y_k}(m_i)$ and $\max_{y_{\bar{k}} \in \overline{\mathcal{Y}}_i} f_{y_k}(m_i)$. This forces m_i to be embedded closer to the most "relevant" type in the noisy candidate type set, *i.e.*, $y^* = \text{argmax}_{y_k \in \mathcal{Y}_i} f_{y_k}(m_i)$, than to *any other* non-candidate types (*i.e.*, Hypothesis 5.3.5). This constrasts sharply with multi-label learning [99], where a large margin is enforced between *all* candidate types and non-candidate types without considering noisy types.

5.3.6 Hierarchical Partial-Label Embedding

Our goal is to embed the heterogeneous graph G into a d-dimensional vector space, following the three proposed hypotheses in the section. Intuitively, one can *collectively* minimize the objectives of the two kinds of loss functions ℓ_c and ℓ_n, across all the training mentions. To achieve the goal, we formulate a joint optimization problem as follows.

$$\min_{\mathbf{U},\ \mathbf{V}} \mathcal{O} = \sum_{m_i \in \mathcal{M}_c} \ell_c(m_i, \mathcal{Y}_i, \overline{\mathcal{Y}}_i) + \sum_{m_i \in \mathcal{M}_n} \ell_n(m_i, \mathcal{Y}_i, \overline{\mathcal{Y}}_i).$$

We use an alternative minimization algorithm based on block-wise coordinate descent [82] to *jointly* optimize the objective $\mathcal{O}$. One can also apply stochastic gradient descent to do online update.

Type Inference. With the learned mention embeddings $\{\mathbf{u}_i\}$ and type embeddings $\{\mathbf{v}_k\}$,

Data sets	Wiki	OntoNotes	BBN
#Types	113	89	47
#Documents	780,549	13,109	2,311
#Sentences	1.51M	143,709	48,899
#Training mentions	2.69M	223,342	109,090
#Ground-truth mentions	563	9,604	121,001
#Features	644,860	215,642	125,637
#Edges in graph	87M	5.9M	2.9M

Table 5.3: Statistics of the datasets.

we perform top-down search in the given type hierarchy $\mathcal{Y}$ to estimate the correct type-path $\mathcal{Y}_i^*$. Starting from the tree's root, we recursively find the best type among the children types by measuring the dot product of the corresponding mention and type embeddings, *i.e.*, $\text{sim}(\mathbf{u}_i, \mathbf{v}_k)$. The search process stops when we reach a leaf type, or the similarity score is below a pre-defined threshold $\eta > 0$.

5.4 EXPERIMENTS

5.4.1 Data Preparation

Datasets. Our experiments use three public datasets. (1) **Wiki** [36]: consists of 1.5M sentences sampled from Wikipedia articles; (2) **OntoNotes** [108]: consists of 13,109 news documents where 77 test documents are manually annotated [101]; (3) **BBN** [109]: consists of 2,311 Wall Street Journal articles which are manually annotated using 93 types. Statistics of the datasets are shown in Table 6.5.

Training Data. We followed the process in [36] to generate training data for the Wiki dataset. For the BBN and OntoNotes datasets, we used DBpedia Spotlight[2] for entity linking. We discarded types which cannot be mapped to Freebase types in the BBN dataset (47 of 93).

Table 6.3 lists the set of features used in our experiments, which are similar to those used in [58, 36] except for topics and ReVerb patterns. We used a 6-word window to extract context unigrams and bigrams for each mention (3 words on the left and the right). We applied the Stanford CoreNLP tool [105] to get POS tags and dependency structures. The word clusters were derived for each corpus using the Brown clustering algorithm[3]. Features

[2] `http://spotlight.dbpedia.org/`

[3] `https://github.com/percyliang/brown-cluster`

for a mention is represented as a binary indicator vector where the dimensionality is the number of features derived from the corpus. We discarded the features which occur only once in the corpus. The number of features generated for each dataset is shown in Table 6.5.

5.4.2 Evaluation Settings

For the Wiki and OntoNotes datasets, we used the provided test set. Since BBN corpus is fully annotated, we followed a 80/20 ratio to partition it into training/test sets. We report Accuracy (Strict-F1), Micro-averaged F1 (Mi-F1) and Macro-averaged F1 (Ma-F1) scores commonly used in the fine-grained type problem [36, 58]. Since we use the gold mention set for testing, the Accuracy (**Acc**) we reported is the same as the Strict F1.

Baselines. We compared the proposed method (AFET) and its variant with state-of-the-art typing methods, embedding methods and partial-label learning methods [4]: (1) **FIGER** [36]; (2) **HYENA** [99]; (3) **FIGER/HYENA-Min** [101]: removes types appearing only once in the document; (4) **ClusType** [8]: predicts types based on co-occurring relation phrases; (5) **HNM** [110]: proposes a hybrid neural model without hand-crafted features; (6) **DeepWalk** [61]: applies Deep Walk to a feature-mention-type graph by treating all nodes as the same type; (7) **LINE** [60]: uses a second-order LINE model on feature-type bipartite graph; (8) **PTE** [111]: applies the PTE joint training algorithm on feature-mention and type-mention bipartite graphs. (9) **WSABIE** [58]: adopts WARP loss to learn embeddings of features and types; (10) **PL-SVM** [66]: uses a margin-based loss to handle label noise. (11) **CLPL** [65]: uses a linear model to encourage large average scores for candidate types.

We compare AFET and its variant: (1) **AFET**: complete model with KB-induced type correlation; (2) **AFET-CoH**: with hierarchy-induced correlation (*i.e.*, shortest path distance); (3) **AFET-NoCo**: without type correlation (*i.e.*, all margin are "1") in the objective $\mathcal{O}$; and (4) **AFET-NoPa**: without label partial loss in the objective $\mathcal{O}$.

5.4.3 Performance Comparison and Analyses

Table 5.4 shows the results of AFET and its variants.

Comparison with the other typing methods. AFET outperforms both FIGER and HYENA systems, demonstrating the predictive power of the learned embeddings, and the effectiveness of modeling type correlation information and noisy candidate types. We also

[4]We used the published code for FIGER, ClusType, HNM, LINE, PTE, and DeepWalk, and implemented other baselines which have no public code. Our implementations yield comparable performance as those reported in the original papers.

observe that pruning methods do not always improve the performance, since they aggressively filter out rare types in the corpus, which may lead to low Recall. ClusType is not as good as FIGER and HYENA because it is intended for coarse types and only utilizes relation phrases.

Comparison with the other embedding methods. AFET performs better than all other embedding methods. HNM does not use any linguistic features. None of the other embedding methods consider the label noise issue and treat the candidate type sets as clean. Although AFET adopts the WARP loss in WSABIE, it uses an adaptive margin in the objective to capture the type correlation information.

Comparison with partial-label learning methods. Compared with PL-SVM and CLPL, AFET obtains superior performance. PL-SVM assumes that only *one* candidate type is correct and does not consider type correlation. CLPL simply averages the model output for all candidate types, and thus may generate results biased to frequent false types. Superior performance of AFET mainly comes from modeling type correlation derived from KB.

Comparison with its variants. AFET always outperforms its variant on all three datasets. It gains performance from capturing type correlation, as well as handling type noise in the embedding process.

5.5 RELATED WORK

There has been considerable work on named entity recognition (NER) [105], which focuses on three types (*e.g.*, `person`, `location`) and cast the problem as multi-class classification following the type mutual exclusion assumption (*i.e.*, one type per mention) [44].

Recent work has focused on a much larger set of fine-grained types [99, 36]. As the type mutual exclusion assumption no longer holds, they cast the problem as multi-label multi-class (hierarchical) classification problems [101, 99, 36]. Embedding techniques are also recently applied to jointly learn feature and type representations [58, 110]. Del Corro *et al.* [112] proposed an unsupervised method to generate context-aware candidates types, and subsequently select the most appropriate type. Gillick *et al.* [101] discuss the label noise issue in fine-grained typing and propose three pruning heuristics. However, these heuristics aggressively delete training examples and may suffer from low recall (see Table. 5.4).

In the context of distant supervision, label noise issue has been studied for other information extraction tasks such as relation extraction [55]. In relation extraction, label noise is introduced by the false positive textual matches of entity pairs. In entity typing, however,

Typing Method	Wiki Acc	Wiki Ma-F1	Wiki Mi-F1	OntoNotes Acc	OntoNotes Ma-F1	OntoNotes Mi-F1	BBN Acc	BBN Ma-F1	BBN Mi-F1
CLPL [65]	0.162	0.431	0.411	0.201	0.347	0.358	0.438	0.603	0.536
PL-SVM [66]	0.428	0.613	0.571	0.225	0.455	0.437	0.465	0.648	0.582
FIGER [36]	0.474	0.692	0.655	0.369	0.578	0.516	0.467	0.672	0.612
FIGER-Min [101]	0.453	0.691	0.631	0.373	0.570	0.509	0.444	0.671	0.613
HYENA [99]	0.288	0.528	0.506	0.249	0.497	0.446	0.523	0.576	0.587
HYENA-Min	0.325	0.566	0.536	0.295	0.523	0.470	0.524	0.582	0.595
ClusType [8]	0.274	0.429	0.448	0.305	0.468	0.404	0.441	0.498	0.573
HNM [110]	0.237	0.409	0.417	0.122	0.288	0.272	0.551	0.591	0.606
DeepWalk [61]	0.414	0.563	0.511	0.479	0.669	0.611	0.586	0.638	0.628
LINE [60]	0.181	0.480	0.499	0.436	0.634	0.578	0.576	0.687	0.690
PTE [111]	0.405	0.575	0.526	0.436	0.630	0.572	0.604	0.684	0.695
WSABIE [58]	0.480	0.679	0.657	0.404	0.580	0.527	0.619	0.670	0.680
AFET-NoCo	0.526	0.693	0.654	0.486	0.652	0.594	0.655	0.711	0.716
AFET-NoPa	0.513	0.675	0.642	0.463	0.637	0.591	0.669	0.715	0.724
AFET-CoH	0.433	0.583	0.551	0.521	0.680	0.609	0.657	0.703	0.712
AFET	**0.533**	**0.693**	**0.664**	**0.551**	**0.711**	**0.647**	**0.670**	**0.727**	**0.735**

Table 5.4: Study of typing performance on the three datasets.

label noise comes from the assignment of types to entity mentions without considering their contexts. The forms of distant supervision are different in these two problems.

Partial Label Learning. - averaging strategy: - directly model –¿ max-margin: PL-SVM Partial label learning (PLL) [113, 66, 65] deals with the problem where each training example is associated with a set of candidate labels, where *only one is correct*. One intuitive strategy to solve the problem is to assume equal contribution of each candidate label and average the outputs from all candidate labels for prediction [65]. Another strategy is to disambiguate the candidate label set by identifying the true label [113]. Existing disambiguation approaches treat true label as latent variable and adopt expectation-maximization procedure to optimize different objectives, such as maximum likelihood criterion [114] and maximum margin

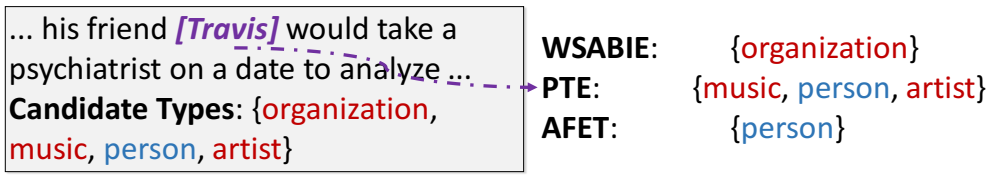


Figure 5.5: Example output of AFET and the compared methods on a training sentence from **OntoNotes** dataset.

Text	"... going to be an imminent easing of monetary policy," said Robert Dederick , chief economist at ***Northern Trust Co. in Chicago***.	...It's terrific for advertisers to know the reader will be paying more , " said Michael Drexler , ***national media director*** at Bozell Inc. ad agency.
Ground Truth	`organization`, `company`	`person`, `person_title`
FIGER	`organization`	`organization`
WSABIE	`organization`, `company`, `broadcast`	`organization`, `company`, `news_company`
PTE	`organization`	`person`
AFET	`organization`, `company`	`person`, `person_title`

Table 5.5: Example output of AFET and the compared methods on two news sentences from **OntoNotes** dataset.

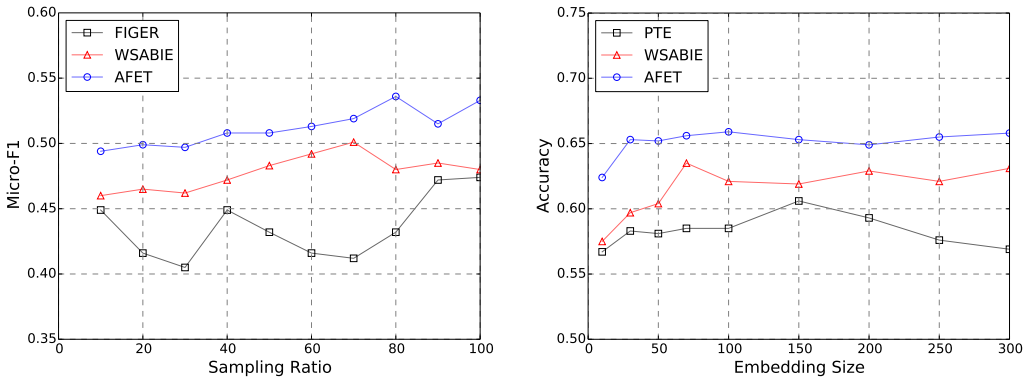
criterion [66]. There also exist methods which adopt error-correcting output codes [113] and instance-level label propagation [115] to disambiguate candidate labels. Unlike existing PLL methods, our method considers type hierarchy and correlation. We compare with simple extensions of PL-SVM [66] and CLPL [65] by applying the learned partial-label classifiers to predicted type-paths in a top-down manner (see Table. 5.4).

Text and Network Embedding. The proposed PLE framework incorporate embedding techniques used in modeling text data [58, 110, 57], and networks/graphs [111, 61, 79]. These methods can be generally classified as supervised [] or unsupervised [], based on how they use labeled data. differences However, existing methods assume links are all correct (unsupervised) or labels are all true (supervised)—our approach seeks to *delete noisy links and lables* in the embedding process. We compare with several embedding methods like PTE [60] to validate the proposed Hypothesis 5.3.5 on noisy labels. With respect to modeling type correlation, our work is related to KB embedding [116, 63], which focuses on embedding *global* information of the KB elements (*e.g.*, entities, relations, types) into a low-dimensional space, although ours incorporates *local* context information of entity mentions in text, and models KB-based type correlation jointly.

5.6 DISCUSSION AND CASE ANALYSIS

Example output on news articles. Table 6.9 shows the types predicted by AFET, FIGER, PTE and WSABIE on two news sentences from OntoNotes dataset: AFET predicts fine-grained types with better accuracy (*e.g.*, `person_title`) and avoids overly-specific predictions (*e.g.*, `news_company`). Figure 5.5 shows the types estimated by AFET, PTE and WSABIE on a training sentence from OntoNotes dataset. We found AFET could discover

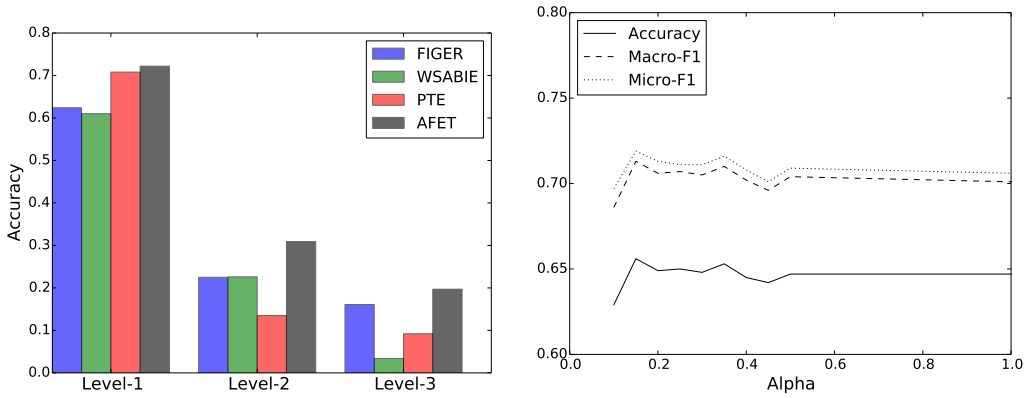
the best type from noisy candidate types.



(a) Varying training set size (b) Varying d

Figure 5.6: Performance change with respect to (a) sampling ratio of training mentions on the Wiki dataset; and (b) embedding dimension d on the BBN dataset.

Testing the effect of training set size and dimension. Experimenting with the same settings for model learning, Fig. 5.6(a) shows the performance trend on the Wiki dataset when varying the sampling ratio (subset of mentions randomly sampled from the training set $\mathcal{D}$). Fig. 5.6(b) analyzes the performance sensitivity of AFET with respect to d—the embedding dimension on the BBN dataset. Accuracy of AFET improves as d becomes large but the gain decreases when d is large enough.



(a) Test at different levels (b) Varying α

Figure 5.7: Performance change (a) at different levels of the type hierarchy on the OntoNotes dataset; and (b) with respect to smooth parameter α on the BBN dataset.

Testing sensitivity of the tuning parameter. Fig. 5.7(b) analyzes the sensitivity of AFET with respect to α on the BBN dataset. Performance increases as α becomes large. When α is large than 0.5, the performance becomes stable.

Testing at different type levels. Fig. 5.7(a) reports the Ma-F1 of AFET, FIGER, PTE and WSABIE at different levels of the target type hierarchy (*e.g.*, person and location on level-1, politician and artist on level-2, author and actor on level-3). The results show that it is more difficult to distinguish among more fine-grained types. AFET always outperforms the other two method, and achieves a 22.36% improvement in Ma-F1, compared to FIGER on level-3 types. The gain mainly comes from explicitly modeling the noisy candidate types.

5.7 SUMMARY

In this chapter, we study *automatic fine-grained entity typing* and propose a *hierarchical partial-label embedding* method, AFET, that models "clean" and "noisy" mentions separately and incorporates a given type hierarchy to induce loss functions. APEFT builds on a joint optimization framework, learns embeddings for mentions and type-paths, and iteratively refines the model. Experiments on three public datasets show that AFET is effective, robust, and outperforms other comparing methods.

As future work, the framework can be considered to incorporate additional language features, to conduct integrated modeling of multiple sources, and to be extended to relationship typing.

Part II

Extracting Typed Relationships

CHAPTER 6: JOINT EXTRACTION OF TYPED ENTITIES AND RELATIONSHIPS WITH KNOWLEDGE BASES

6.1 PROPOSED METHOD: OVERVIEW AND MOTIVATION

The extraction of entities and their relations is critical to understanding massive text corpora. Identifying the token spans in text that constitute entity mentions and assigning types (*e.g.*, `person`, `company`) to these spans as well as to the relations between entity mentions (*e.g.*, `employed_by`) are key to structuring content from text corpora for further analytics. For example, when an extraction system finds a "`produce`" relation between "`company`" and "`product`" entities in news articles, it supports answering questions like "*what products does company X produce?*". Once extracted, such structured information is used in many ways, *e.g.*, as primitives in information extraction, knowledge base population [1, 117], and question-answering systems [118, 119]. Traditional systems for relation extraction [45, 46, 47] partition the process into several subtasks and solve them incrementally (*i.e.*, detecting entities from text, labeling their types and then extracting their relations). Such systems treat the subtasks independently and so may propagate errors across subtasks in the process. Recent studies [48, 49, 50] focus on joint extraction methods to capture the inhereent linguistic dependencies between relations and entity arguments (*e.g.*, the types of entity arguments help determine their relation type, and vice versa) to resolve error propagation.

A major challenge in joint extraction of typed entities and relations is to design *domain-independent* systems that will apply to text corpora from different domains in the *absence of human-annotated, domain data*. The process of manually labeling a training set with a large number of entity and relation types is too expensive and error-prone. The rapid emergence of large, domain-specific text corpora (*e.g.*, news, scientific publications, social media content) calls for methods that can jointly extract entities and relations of target types with minimal or no human supervision.

Towards this goal, there are broadly two kinds of efforts: weak supervision and distant supervision. Weak supervision [68, 34, 3] relies on a small set of manually-specified seed instances (or patterns) that are applied in bootstrapping learning to identify more instances of each type. This assumes seeds are unambiguous and sufficiently frequent in the corpus, which requires careful seed selection by human [45]. Distant supervision [37, 39, 71, 38] generates training data automatically by aligning texts and a knowledge base (KB) (see Fig. 6.1). The typical workflow is: (1) detect entity mentions in text; (2) map detected entity mentions to entities in KB; (3) assign, to the candidate type set of each entity mention, all KB types of its KB-mapped entity; (4) assign, to the candidate type set of each entity

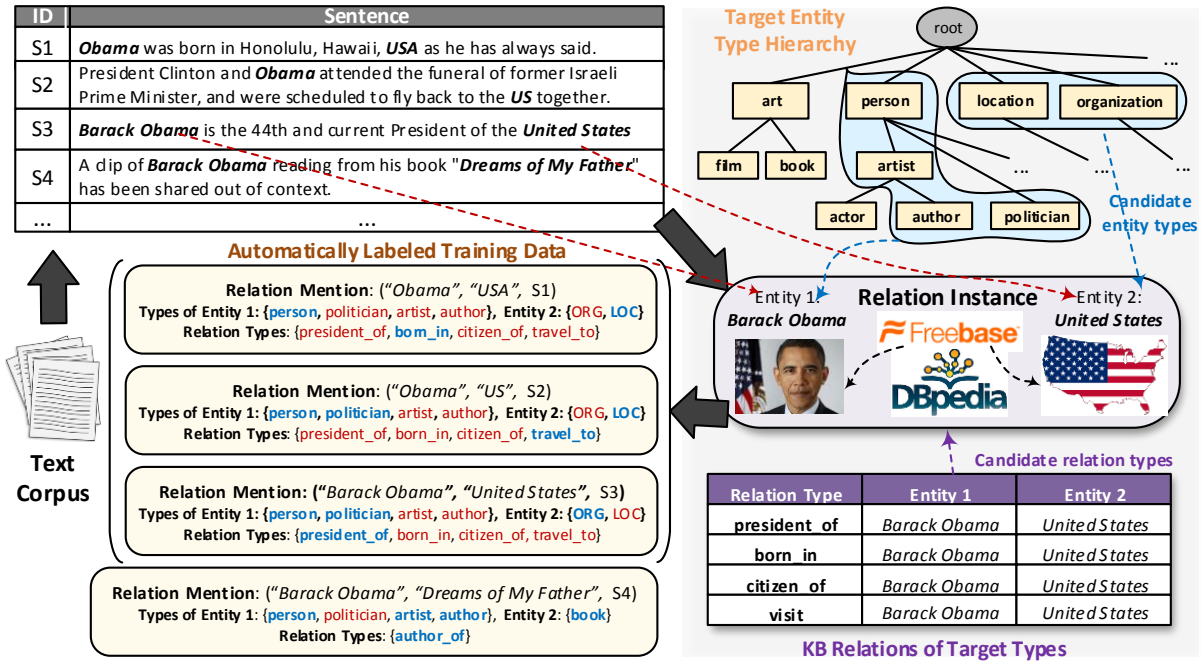


Figure 6.1: Current systems find relations (*Barack Obama*, *United States*) mentioned in sentences S1-S3 and assign the same relation types (entity types) to all relation mentions (entity mentions), when only some types are correct for context (highlighted in blue font).

mention pair, all KB relation types between their KB-mapped entities. The automatically labeled training corpus is then used to infer types of the *remaining* candidate entity mentions and relation mentions (*i.e.*, unlinkable candidate mentions).

In this paper, we study the problem of *joint extraction of typed entities and relations with distant supervision*. Given a domain-specific corpus and a set of target entity and relation types from a KB, we aim to detect relation mentions (together with their entity arguments) from text, and categorize each in context by target types or Not-Target-Type (`None`), with distant supervision. Current distant supervision methods focus on solving the subtasks separately (*e.g.*, extracting typed entities or relations), and encounter the following limitations when handling the joint extraction task.

• **Domain Restriction:** They rely on pre-trained named entity recognizers (or noun phrase chunker) to detect entity mentions. These tools are usually designed for a few general types (*e.g.*, `person`, `location`, `organization`) and require additional human labors to work on specific domains (*e.g.*, scientific publications).

• **Error Propagation:** In current extraction pipelines, incorrect entity types generated in entity recognition and typing step serve as features in the relation extraction step (*i.e.*, see Figure 6.2, errors are propagated from upstream components to downstream ones). Cross-task dependencies are ignored in most existing methods.

Dataset	NYT [39]	Wiki-KBP [120],	BioInfer [121]
# of entity types	47	126	2,200
noisy entity mentions (%)	20.32	28.31	59.80
# of relation types	24	19	94
noisy relation mentions (%)	15.54	8.54	41.12

Table 6.1: A study of type label noise. (1): %entity mentions with multiple *sibling entity types* (*e.g.*, `actor`, `singer`) in the given entity type hierarchy; (2): %relation mentions with multiple *relation types*, for the three experiment datasets.

• **Label Noise:** In distant supervision, the context-agnostic mapping from relation (entity) mentions to KB relations (entities) may bring false positive type labels (*i.e.*, label noise) into the automatically labeled training corpora and results in inaccurate models.

In Fig. 6.1, for example, all KB relations between entities *Barack Obama* and *United States* (*e.g.*, `born_in`, `president_of`) are assigned to the relation mention in sentence S_1 (while only `born_in` is correct within the context). Similarly, all KB types for *Barack Obama* (*e.g.*, `politician`, `artist`) are assigned to the mention "*Obama*" in S_1 (while only `person` is true). Label noise becomes an impediment to learn effective type classifiers. The larger the target type set, the more severe the degree of label noise (see Table 6.1).

We approach the joint extraction task as follows: (1) Design a domain-agnostic text segmentation algorithm to detect candidate entity mentions with distant supervision and mini-

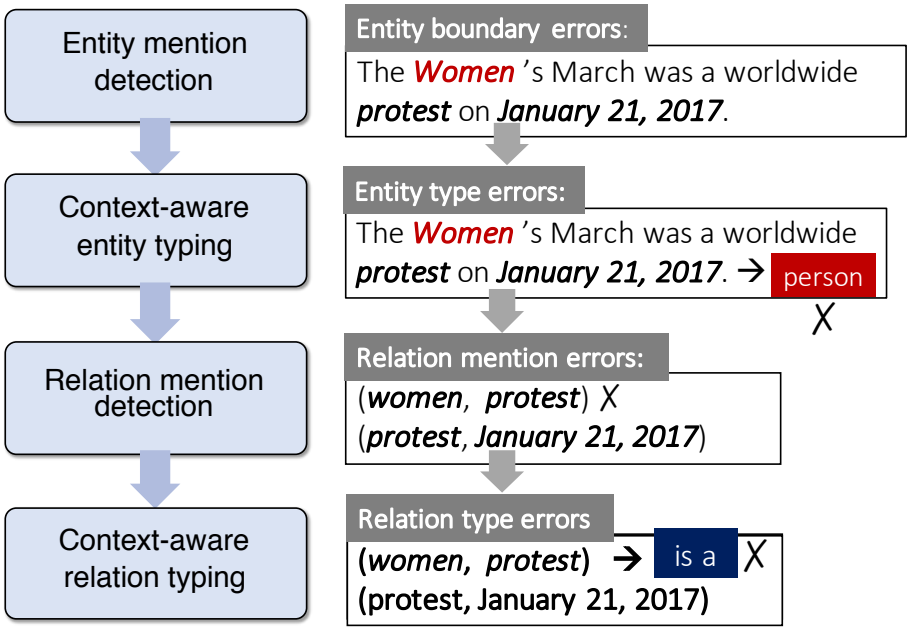


Figure 6.2: An illustration of error propagation in the incremental relation extraction pipeline.

mal linguistic assumption (*i.e.*, assuming part-of-speech (POS) tagged corpus is given [122]). (2) Model the mutual constraints between the types of the relation mentions and the types of their entity arguments, to enable feedbacks between the two subtasks. (3) Model the true type labels in a candidate type set as latent variables and require only the "*best*" type (progressively estimated as we learn the model) to be relevant to the mention—this is a less limiting requirement compared with existing multi-label classifiers that assume "*every*" candidate type is relevant to the mention.

To integrate these elements of our approach, a novel framework, CoType, is proposed. It first runs POS-constrained text segmentation using positive examples from KB to mine quality entity mentions, and forms candidate relation mentions (Sec. 6.3.1). Then CoType performs entity linking to map candidate relation (entity) mentions to KB relations (entities) and obtain the KB types. We formulate a global objective to *jointly* model (1) corpus-level co-occurrences between *linkable* relation (entity) mentions and text features extracted from their local contexts; (2) associations between mentions and their KB-mapped type labels; and (3) interactions between relation mentions and their entity arguments. In particular, we design a novel partial-label loss to model the noisy mention-label associations in a robust way, and adopt translation-based objective to capture the entity-relation interactions. Minimizing the objective yields two low-dimensional spaces (for entity and relation mentions, respectively), where, in each space, objects whose types are semantically close also have similar representation (see Sec. 6.3.2). With the learned embeddings, we can efficiently estimate the types for the remaining *unlinkable* relation mentions and their entity arguments (see Sec. 6.3.3).

The major contributions of this paper are as follows:

1. A novel distant-supervision framework, CoType, is proposed to extract typed entities and relations in domain-specific corpora with minimal linguistic assumption. (Fig. 6.3.)

2. A domain-agnostic text segmentation algorithm is developed to detect entity mentions using distant supervision. (Sec. 6.3.1)

3. A joint embedding objective is formulated that models mention-type association, mention-feature co-occurrence, entity-relation cross-constraints in a noise-robust way. (Sec. 6.3.2)

4. Experiments with three public datasets demonstrate that CoType improves the performance of state-of-the-art systems of entity typing and relation extraction significantly, demonstrating robust domain-independence.(Sec. 6.4)

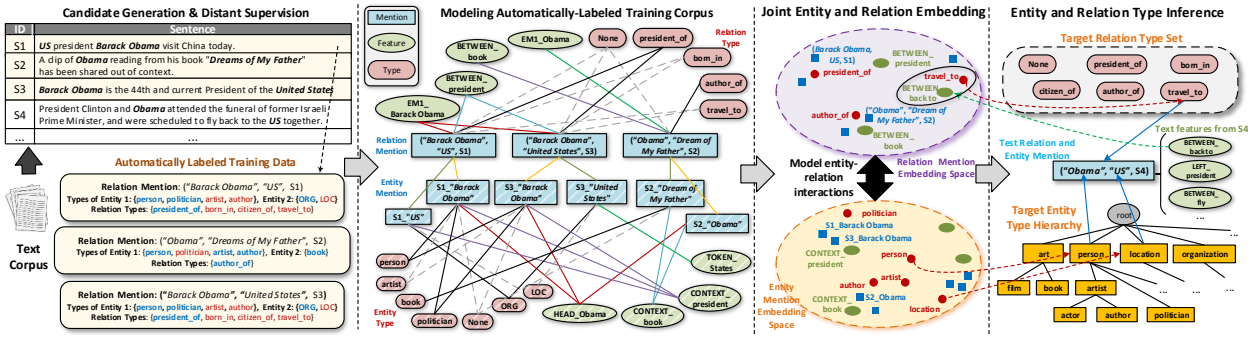


Figure 6.3: Framework Overview of CoType.

6.2 PRELIMINARIES

The input to our proposed CoType framework is a POS-tagged text corpus $\mathcal{D}$, a knowledge bases Ψ (*e.g.*, Freebase [6]), a target entity type hierarchy $\mathcal{Y}$ and a target relation type set $\mathbb{R}$. The target type set $\mathcal{Y}$ (set $\mathbb{R}$) covers a subset of entity (relation) types that the users are interested in from Ψ, *i.e.*, $\mathcal{Y} \subset \mathcal{Y}_\Psi$ and $\mathbb{R} \subset \mathbb{R}_\Psi$.

Entity and Relation Mention. An *entity mention* (denoted by m) is a token span in text which represents an entity e. A *relation instance* $r(e_1, e_2, \ldots, e_n)$ denotes some type of relation $r \in \mathbb{R}$ between multiple entities. In this work, we focus on binary relations, *i.e.*, $r(e_1, e_2)$. We define a *relation mention* (denoted by z) for some relation instance $r(e_1, e_2)$ as a (ordered) pair of entities mentions of e_1 and e_2 in a sentence s, and represent a relation mention with entity mentions m_1 and m_2 in sentence s as $z = (m_1, m_2, s)$.

Knowledge Bases and Target Types. A KB with a set of entities $\mathcal{E}_\Psi$ contains human-curated facts on both relation instances $\mathcal{I}_\Psi = \{r(e_1, e_2)\} \subset \mathbb{R}_\Psi \times \mathcal{E}_\Psi \times \mathcal{E}_\Psi$, and entity-type facts $\mathcal{T}_\Psi = \{(e, y)\} \subset \mathcal{E}_\Psi \times \mathcal{Y}_\Psi$. *Target entity type hierarchy* is a tree where nodes represent entity types of interests from the set $\mathcal{Y}_\Psi$. An entity mention may have multiple types, which together constitute one *type-path* (not required to end at a leaf) in the given type hierarchy. In existing studies, several entity type hierarchies are manually constructed using Freebase [123, 101] or WordNet [99]. *Target relation type set* is a set of relation types of interests from the set $\mathbb{R}_\Psi$.

Automatically Labeled Training Data. Let $\mathcal{M} = \{m_i\}_{i=1}^N$ denote the set of entity mentions extracted from corpus $\mathcal{D}$. Distant supervision maps $\mathcal{M}$ to KB entities $\mathcal{E}_\Psi$ with an entity disambiguation system [124, 125] and heuristically assign type labels to the mapped mentions. In practice, only a small number of entity mentions in set $\mathcal{M}$ can be mapped to entities in $\mathcal{E}_\Psi$ (*i.e.*, *linkable entity mentions*, denoted by $\mathcal{M}_L$). As reported in [8, 35], the ratios of $\mathcal{M}_L$ over $\mathcal{M}$ are usually *lower than 50%* in domain-specific corpora.

Between any two linkable entity mentions m_1 and m_2 in a sentence, a relation mention z_i is formed if there exists one or more KB relations between their KB-mapped entities e_1 and e_2. Relations between e_1 and e_2 in KB are then associated to z_i to form its candidate relation type set $\mathbb{R}_i$, *i.e.*, $\mathbb{R}_i = \{r \mid r(e_1, e_2) \in \mathbb{R}_\Psi\}$. In a similar way, types of e_1 and e_2 in KB are associated with m_1 and m_2 respectively, to form their candidate entity type sets $\mathcal{Y}_{i,1}$ and $\mathcal{Y}_{i,2}$, where $\mathcal{Y}_{i,x} = \{y \mid (e_x, y) \in \mathcal{Y}_\Psi\}$. Let $\mathcal{Z}_L = \{z_i\}_{i=1}^{N_L}$ denote the set of extracted relation mentions that can be mapped to KB. Formally, we represent the automatically labeled training corpus for the joint extraction task, denoted as $\mathcal{D}_L$, using a set of tuples $\mathcal{D}_L = \{(z_i, \mathbb{R}_i, \mathcal{Y}_{i,1}, \mathcal{Y}_{i,2})\}_{i=1}^{N_L}$.

Problem Description. By pairing up entity mentions (from set $\mathcal{M}$) within each sentence in $\mathcal{D}$, we generate a set of *candidate relation mentions*, denoted as $\mathcal{Z}$. Set $\mathcal{Z}$ consists of (1) linkable relation mentions $\mathcal{Z}_L$, (2) unlinkable (true) relation mentions, and (3) false relation mention (*i.e.*, no target relation expressed between).

Let $\mathcal{Z}_U$ denote the set of *unlabeled* relation mentions in (2) and (3) (*i.e.*, $\mathcal{Z}_U = \mathcal{Z} \setminus \mathcal{Z}_L$). Our main task is to determine the relation type label (from the set $\mathbb{R} \cup \{\texttt{None}\}$) for each relation mention in set $\mathcal{Z}_U$, and the entity type labels (either a single type-path in $\mathcal{Y}$ or `None`) for each entity mention argument in $z \in \mathcal{Z}_U$, using the automatically labeled corpus $\mathcal{D}_L$. Formally, we define the joint extraction of typed entities and relations task as follows.

Problem 6.1: Joint Entity and Relation Extraction

Given a POS-tagged corpus $\mathcal{D}$, a KB Ψ, a target entity type hierarchy $\mathcal{Y} \subset \mathcal{Y}_\Psi$ and a target relation type set $\mathbb{R} \subset \mathbb{R}_\Psi$, the joint extraction task **aims to** (1) detect entity mentions $\mathcal{M}$ from $\mathcal{D}$; (2) generate training data $\mathcal{D}_L$ with KB Ψ; and (3) estimate a relation type $r^* \in \mathbb{R} \cup \{\texttt{None}\}$ for each test relation mention $z \in \mathcal{Z}_U$ and a single type-path $\mathcal{Y}^* \subset \mathcal{Y}$ (or `None`) for each entity mention in z, using $\mathcal{D}_L$ and its context s.

Non-goals. This work relies on an entity linking system [124] to provide disambiguation function, but we do not address their limits here (*e.g.*, label noise introduced by wrongly mapped KB entities). We also assume human-curated target type hierarchies are given (It is out of the scope of this study to generate the type hierarchy).

6.3 THE COTYPE FRAMEWORK

This section lays out the proposed framework. The joint extraction task poses two unique challenges. First, type association in distant supervision between linkable entity (relation) mentions and their KB-mapped entities (relations) is *context-agnostic*—the candidate type

sets $\{\mathbb{R}_i, \mathcal{Y}_{i,1}, \mathcal{Y}_{i,2}\}$ contain "false" types. Supervised learning [47, 62] may generate models biased to the incorrect type labels [56]. Second, there exists dependencies between relation mentions and their entity arguments (*e.g.*, type correlation). Current systems formulates the task as a *cascading* supervised learning problem and may suffer from error propagation.

Our solution casts the type prediction task as *weakly-supervised learning* (to model the relatedness between mentions and their candidate types *in contexts*) and uses *relational learning* to capture interactions between relation mentions and their entity mention argument *jointly*, based on the redundant text signals in a large corpus.

Specifically, CoType leverages partial-label learning [66] to faithfully model mention-type association using text features extracted from mentions' local contexts. It uses the translation embedding-based objective [63] to model the mutual type dependencies between relation mentions and their entity (mention) arguments.

Framework Overview. We propose a *embedding-based* framework with distant supervision (see also Fig. 6.3) as follows:

1. Run POS-constrained text segmentation algorithm on POS-tagged corpus $\mathcal{D}$ using positive examples obtained from KB, to detect candidate entity mentions $\mathcal{M}$ (Sec. 6.3.1).

2. Generate candidate relation mentions $\mathcal{Z}$ from $\mathcal{M}$, extract text features for each relation mention $z \in \mathcal{Z}$ and their entity mention argument (Sec. 6.3.1). Apply distant supervision to generate labeled training data $\mathcal{D}_L$ (Sec. 6.2).

3. Jointly embed relation and entity mentions, text features, and type labels into two low-dimensional spaces (for entities and relations, respectively) where, in each space, close objects tend to share the same types (Sec. 6.3.2).

4. Estimate type labels r^* for each test relation mention $z \in \mathcal{Z}_U$ and type-path $\mathcal{Y}^*$ for each test entity mention m in $\mathcal{Z}_U$ from learned embeddings, by searching the target type set $\mathcal{Y}$ or the target type hierarchy $\mathbb{R}$ (Sec. 6.3.3).

6.3.1 Candidate Generation

Entity Mention Detection. Traditional entity recognition systems
[25, 44] rely on a set of linguistic features (*e.g.*, dependency parse structures of a sentence) to train sequence labeling models (for a few common entity types). However, sequence labeling models trained on automatically labeled corpus $\mathcal{D}_L$ may not be effective, as distant supervision only annotates a small number of entity mentions in $\mathcal{D}_L$ (thus generates a lot of "false

negative" token tags). To address domain restriction, we develop a distantly-supervised text segmentation algorithm for *domain-agnostic entity detection*. By using quality examples from KB as guidance, it partitions sentences into segments of entity mentions and words, by incorporating (1) corpus-level concordance statistics; (2) sentence-level lexical signals; and (3) grammatical constraints (*i.e.*, POS tag patterns).

We extend the methdology used in [126, 127] to model the *segment quality* (*i.e.*, "how likely a candidate segment is an entity mention") as a combination of *phrase quality* and *POS pattern quality*, and use positive examples in $\mathcal{D}_L$ to estimate the segment quality. The workflow is as follows: (1) mine frequent contiguous patterns for both word sequence and POS tag sequence up to a fixed length from POS-tagged corpus $\mathcal{D}$; (2) extract features including corpus-level concordance and sentence-level lexical signals to train two random forest classifiers [126], for estimating quality of candidate phrase and candidate POS pattern; (3) find the best segmentation of $\mathcal{D}$ using the estimated segment quality scores (see Eq. (6.1)); and (4) compute rectified features using the segmented corpus and repeat steps (2)-(4) until the result converges.

$$p\big(b_{t+1}, c \mid b_t\big) = p\big(b_{t+1} - b_t\big) \cdot p\big(c \mid b_{t+1} - b_t\big) \cdot Q(c) \tag{6.1}$$

Specifically, we find the best segmentation S_d for each document d (in $\mathcal{D}$) by maximizing the "joint segmentation quality", defined as $\sum_d^{\mathcal{D}} \log\, p(S_d, d) = \sum_d^{\mathcal{D}} \sum_{t=1}^{|d|} \log\, p(b_{t+1}^{(d)}, c^{(d)} \mid b_t^{(d)})$, where $p(b_{t+1}^{(d)}, c^{(d)} \mid b_t^{(d)})$ denote the probability that segment $c^{(c)}$ (with starting index $b_{t+1}^{(d)}$ and ending index in document d) is a good entity mention, as defined in Eq. (6.1). The first term in Eq. (6.1) is a segment length prior, the second term measures how likely segment c is generated given a length $(b_{t+1} - b_t)$ (to be estimated), and the third term denotes the segment quality. In this work, we define function $Q(c)$ as the *equally weighted combination of the phrase quality score and POS pattern quality score* for candidate segment c, which is estimated in step (2). The joint probability can be efficiently maximize using Viterbi Training with time complexity linear to the corpus size [126]. The segmentation result provides us a set of candidate entity mentions, forming the set $\mathcal{M}$.

Table 6.2 compares our entity detection module with a sequence labeling model [36] (linear-chain CRF) trained on the labeled corpus $\mathcal{D}_L$ in terms of F1 score. Fig. 6.4 show the high/low quality POS patterns learned using entity names found in $\mathcal{D}_L$ as examples.

Relation Mention Generation. We follow the procedure introduced in Sec. 6.2 to generate the set of candidate relation mentions $\mathcal{Z}$ from the detected candidate entity mentions $\mathcal{M}$: for each pair of entity mentions (m_a, m_b) found in sentence s, we form two candidate relation mentions $z_1 = (m_a, m_b, s)$ and $z_2 = (m_b, m_a, s)$. Distant supervision is then applied on $\mathcal{Z}$ to

	POS Tag Pattern	Example
Good (high score)	*NNP NNP*	San Francisco/Barack Obama/United States
	NN NN	comedy drama/car accident/club captain
	CD NN	seven network/seven dwarfs/2001 census
	JJ NN	crude oil/nucletic acid/baptist church
Bad (low score)	*DT JJ NND*	a few miles/the early stages/the late 1980s
	CD CD NN IN	2 : 0 victory over/1 : 0 win over
	NN IN NNP NNP	rating on rotten tomatoes
	VVD RB IN	worked together on/spent much of

Figure 6.4: Example POS tag patterns learned using KB examples.

generate the set of KB-mapped relation mentions $\mathcal{Z}_L$. Similar to [37, 71], we sample 30% unlinkable relation mentions between two KB-mapped entity mentions (from set $\mathcal{M}_L$) in a sentence as examples for modeling `None` relation label, and sample 30% unlinkable entity mentions (from set $\mathcal{M} \setminus \mathcal{M}_L$) to model `None` entity label. These negative examples, together with type labels for mentions in $\mathcal{Z}_L$, form the automatically labeled data $\mathcal{D}_L$ for the task.

Text Feature Extraction. To capture the shallow syntax and distributional semantics of a relation (or entity) mention, we extract various lexical features from both mention itself (*e.g.*, head token) and its context s (*e.g.*, bigram), in the POS-tagged corpus. Table 6.3 lists the set of text features for relation mention, which is similar to those used in [37, 128] (excluding the dependency parse-based features and entity type features). We use the same set of features for entity mentions as those used in [56, 36]. We denote the set of M_z (M_m) unique features extracted of relation mentions $\mathcal{Z}_L$ (entity mentions in $\mathcal{Z}_L$) as $\mathcal{F}_z = \{f_j\}_{j=1}^{M_z}$ (and $\mathcal{F}_m = \{f_j\}_{j=1}^{M_m}$).

6.3.2 Joint Entity and Relation Embedding

This section formulates a joint optimization problem for embedding different kinds of interactions between linkable relation mentions $\mathcal{Z}_L$, linkable entity mentions $\mathcal{M}_L$, entity and relation type labels $\{\mathbb{R}, \mathcal{Y}\}$ and text features $\{\mathcal{F}_z, \mathcal{F}_m\}$ into a d-dimensional *relation vector space* and a d-dimensional *entity vector space*. In each space, objects whose types are close

Dataset	NYT	Wiki-KBP	BioInfer
FIGER segmenter [36]	0.751	0.814	0.652
Our Approach	**0.837**	**0.833**	**0.785**

Table 6.2: Comparison of F1 scores on entity mention detection.

Feature	Description	Example
Entity mention (EM) head	Syntactic head token of each entity mention	"*HEAD_EM1_Obama*"
Entity Mention Token	Tokens in each entity mention	"*TKN_EM1_Barack*"
Tokens between two EMs	Each token between two EMs	"*was*", "*elected*", "*President*", "*of*", "*the*"
Part-of-speech (POS) tag	POS tags of tokens between two EMs	"*VBD*", "*VBN*", "*NNP*", "*IN*", "*DT*"
Collocations	Bigrams in left/right 3-word window of each EM	"*Honolulu native*", "*native Barack*", ...
Entity mention order	Whether EM 1 is before EM 2	"*EM1_BEFORE_EM2*"
Entity mention distance	Number of tokens between the two EMs	"*EM_DISTANCE_5*"
Entity mention context	Unigrams before and after each EM	"*native*", "*was*", "*the*", "*in*"
Special pattern	Occurrence of pattern "em1_in_em2"	"*PATTERN_NULL*"
Brown cluster (learned on $\mathcal{D}$)	Brown cluster ID for each token	"*8_1101111*", "*12_111011111111*"

Table 6.3: Text features for relation mentions used in this work [47, 39] (excluding dependency parse-based features and entity type features). ("*Barack Obama*", "*United States*") is used as an example relation mention from the sentence "*Honolulu native* ***Barack Obama*** *was elected President of the* ***United States*** *on March 20 in 2008.*".

to each other should have similar representation (*e.g.*, see the 3rd col. in Fig. 6.3).

As the extracted objects and the interactions between them form a heterogeneous graph (see the 2nd col. in Fig. 6.3), a simple solution is to embed the whole graph into a *single* low-dimensional space [79, 8]. However, such a solution encounters several problems: (1) False types in candidate type sets (*i.e.*, false mention-type links in the graph) negatively impact the ability of the model to determine mention's true types; and (2) a single embedding space cannot capture the differences in entity and relation types (*i.e.*, strong link between a relation mention and its entity mention argument does not imply that they have similar types).

In our solution, we propose a novel global objective, which extends a margin-based rank loss [66] to model *noisy mention-type associations* and leverages the second-order proximity idea [60] to model corpus-level *mention-feature co-occurrences*. In particular, to capture the *entity-relation interactions*, we adopt a translation-based embedding loss [63] to bridge the vector spaces of entity mentions and relation mentions.

Modeling Types of Relation Mentions. We consider both *mention-feature co-occurrences* and *mention-type associations* in the modeling of relation types for relation mentions in set Z_L.

Intuitively, two relation mentions sharing many text features (*i.e.*, with similar distribution over the set of text features $\mathcal{F}_m$) likely have similar relation types; and text features co-occurring with many relation mentions in the corpus tend to represent close type semantics. We propose the following hypothesis to guide our modeling of corpus-level mention-feature co-occurrences.

Hypothesis 6.1: Mention-Feature Co-occurrence

Two entity mentions tend to share similar types (close to each other in the embedding space) if they share many text features in the corpus, and the converse way also holds.

For example, in column 2 of Fig. 6.3, ("*Barack Obama*", "*US*", S_1) and ("*Barack Obama*", "*United States*", S_3) share multiple features including context word "*president*" and first entity mention argument "*Barack Obama*", and thus they are likely of the same relation type (*i.e.*, `president_of`).

Formally, let vectors $\mathbf{z}_i,\ \mathbf{c}_j \in \mathcal{R}^d$ represent relation mention $z_i \in \mathcal{Z}_L$ and text feature $f_j \in \mathcal{F}_z$ in the d-dimensional *relation embedding space*. Similar to the distributional hypothesis [57] in text corpora, we apply second-order proximity [60] to model the idea that *objects with similar distribution over neighbors are similar to each other* as follows.

$$\mathcal{L}_{ZF} = -\sum_{z_i \in \mathcal{Z}_L} \sum_{f_j \in \mathcal{F}_z} w_{ij} \cdot \log\, p(f_j|z_i), \tag{6.2}$$

where $p(f_j|z_i) = \exp(\mathbf{z}_i^T\, \mathbf{c}_j) / \sum_{f' \in \mathcal{F}_z} \exp(\mathbf{z}_i^T\, \mathbf{c}_{j'})$ denotes the probability of f_j generated by z_i, and w_{ij} is the co-occurrence frequency between (z_i, f_j) in corpus $\mathcal{D}$. Function $\mathcal{L}_{ZF}$ in Eq. (6.2) enforces the conditional probability specified by embeddings, *i.e.*, $p(\cdot|z_i)$ to be close to the empirical distribution.

To perform efficient optimization by avoiding summation over all features, we adopt negative sampling strategy [57] to sample multiple *false* features for each (z_i, f_j), according to some *noise distribution* $P_n(f) \propto D_f^{3/4}$ [57] (with D_f denotes the number of relation mentions co-occurring with f). Term $\log\, p(f_j|z_i)$ in Eq. (6.2) is replaced with the term as follows.

$$\log\, \sigma(\mathbf{z}_i^T\, \mathbf{c}_j) + \sum_{v=1}^{V} \mathbb{E}_{f_{j'} \sim P_n(f)} \big[\log\, \sigma(-\,\mathbf{z}_i^T\, \mathbf{c}_{j'})\big], \tag{6.3}$$

where $\sigma(x) = 1/\big(1 + \exp(-x)\big)$ is the sigmoid function. The first term in Eq. (6.3) models the observed co-occurrence, and the second term models the Z negative feature samples.

In D_L, each relation mention z_i is heuristically associated with a set of candidate types $\mathbb{R}_i$. Existing embedding methods rely on either the *local consistent assumption* [79] (*i.e.*, objects strongly connected tend to be similar) or the *distributional assumption* [57] (*i.e.*, objects sharing similar neighbors tend to be similar) to model object associations. However, some associations between z_i and $r \in \mathbb{R}_i$ are "false" associations and adopting the above assumptions may incorrectly yield mentions of different types having similar vector representations. For example, in Fig. 6.1, mentions ("*Obama*", "*USA*", S_1) and ("*Obama*", "*US*", S_2) have several candidate types in common (thus high distributional similarity), but their true types are different (*i.e.*, `born_in` vs. `travel_to`).

We specify the likelihood of "*whether the association between a relation mention and its candidate entity type being true*" as the *relevance* between these two kinds of objects

(measured by the similarity between their current estimated embedding vectors). To impose such idea, we model the associations between each linkable relation mention z_i (in set $\mathcal{Z}_L$) and its noisy candidate relation type set $\mathbb{R}_i$ based on the following hypothesis.

Hypothesis 6.2: Partial-Label Association

A relation mention's embedding vector should be more similar (closer in the low-dimensional space) to its "most relevant" candidate type, than to any other non-candidate type.

Specifically, we use vector $\mathbf{r}_k \in \mathcal{R}^d$ to represent relation type $r_k \in \mathbb{R}$ in the embedding space. The similarity between (z_i, r_k) is defined as the dot product of their embedding vectors, *i.e.*, $\phi(z_i, r_k) = \mathbf{z}_i^T \mathbf{r}_k$. We extend the margin-based loss in [66] and define a partial-label loss ℓ_i for each relation mention $z_i \in \mathcal{M}_L$ as follows.

$$\ell_i = \max\Big\{0, 1 - \Big[\max_{r \in \mathbb{R}_i} \phi(z_i, r) - \max_{r' \in \overline{\mathcal{R}}_i} \phi(z_i, r')\Big]\Big\}. \tag{6.4}$$

The intuition behind Eq. (6.4) is that: for relation mention z_i, the maximum similarity score associated with its candidate type set $\mathbb{R}_i$ should be greater than the maximum similarity score associated with any other *non-candidate types* $\overline{\mathcal{R}}_i = \mathbb{R} \setminus \mathbb{R}_i$. Minimizing ℓ_i forces z_i to be embedded closer to the *most* "*relevant*" type in $\mathbb{R}_i$, than to any other non-candidate types in $\overline{\mathcal{R}}_i$. This contrasts sharply with multi-label learning [36], where m_i is embedded closer to *every* candidate type than any other non-candidate type.

To faithfully model the types of relation mentions, we integrate the modeling of mention-feature co-occurrences and mention-type associations by the following objective.

$$O_Z = \mathcal{L}_{ZF} + \sum_{i=1}^{N_L} \ell_i + \frac{\lambda}{2} \sum_{i=1}^{N_L} \| \mathbf{z}_i \|_2^2 + \frac{\lambda}{2} \sum_{k=1}^{K_r} \| \mathbf{r}_k \|_2^2, \tag{6.5}$$

where tuning parameter $\lambda > 0$ on the regularization terms is used to control the scale of the embedding vectors.

By doing so, text features, as complements to mention's candidate types, also participate in modeling the relation mention embeddings, and help identify a mention's most relevant type—mention-type relevance is progressively estimated during model learning. For example, in the left column of Fig. 6.5, context words "*president*" helps infer that relation type `president_of` is more relevant (*i.e.*, higher similarity between the embedding vectors) to

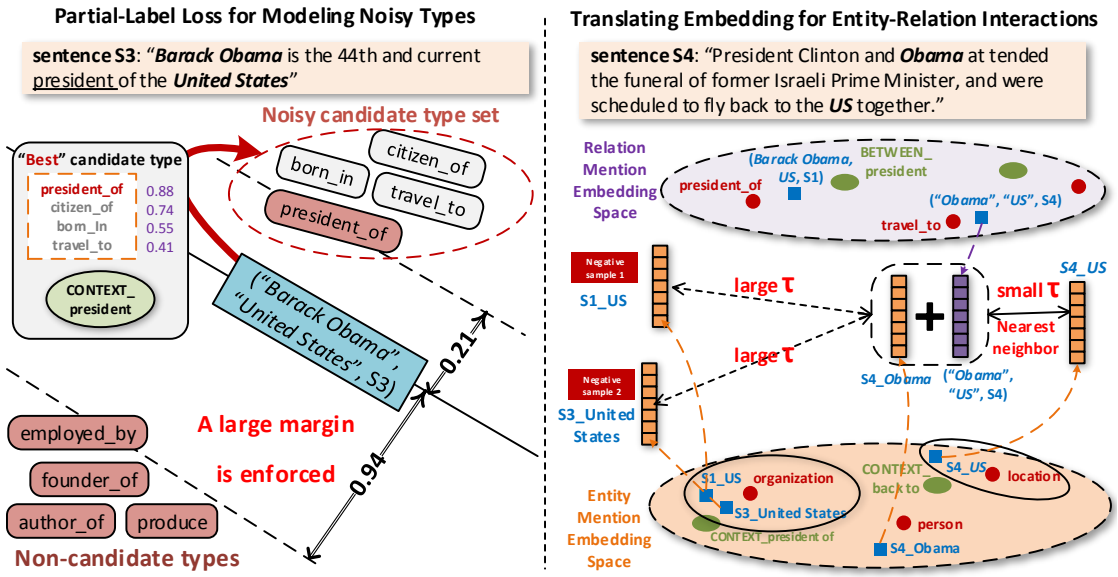


Figure 6.5: Illustrations of the partial-label associations, Hypothesis 6.3.2 (the left col.), and the entity-relation interactions, Hypothesis 6.3.2 (the right col.).

relation mention ("*Mr. Obama*", "*USA*", S_2), than type `born_in` does.

Modeling Types of Entity Mentions. In a way similar to the modeling of types for relation mentions, we follow Hypotheses 6.3.2 and 6.3.2 to model types of entity mentions. In Fig. 6.3 (col. 2), for example, entity mentions "S_1_*Barack Obama*" and "S_3_*Barack Obama*" share multiple text features in the corpus, including head token "*Obama*" and context word "*president*", and thus tend to share the same entity types like `politician` and `person` (*i.e.*, Hypothesis 6.3.2). Meanwhile, entity mentions "S_1_*Barack Obama*" and "S_2_*Obama*" have the same candidate entity types but share very few text features in common. This implies that likely their true type labels are different. Relevance between entity mentions and their true type labels should be progressively estimated based on the text features extracted from their local contexts (*i.e.*, Hypothesis 6.3.2).

Formally, let vectors $\mathbf{m}_i, \mathbf{c}'_j, \mathbf{y}_k \in \mathcal{R}^d$ represent entity mention $m_i \in \mathcal{M}_L$, text features (for entity mentions) $f_j \in \mathcal{F}_m$, and entity type $y_k \in \mathcal{Y}$ in a d-dimensional *entity embedding space*, respectively. We model the corpus-level co-occurrences between entity mentions and text features by second-order proximity as follows.

$$\mathcal{L}_{MF} = -\sum_{m_i \in \mathcal{M}_L} \sum_{f_j \in \mathcal{F}_m} w_{ij} \cdot \log\, p(f_j|m_i), \tag{6.6}$$

where the conditional probability term $\log\, p(f_j|m_i)$ is defined as $\log\, p(f_j|m_i) = \log\, \sigma(\mathbf{m}_i^T\, \mathbf{c}'_j) + \sum_{v=1}^{V} \mathbb{E}_{f_{j'} \sim P_n(f)} \big[\log\, \sigma(-\mathbf{m}_i^T\, \mathbf{c}'_{j'})\big]$. By integrating the term $\mathcal{L}_{MF}$ with partial-label loss $\ell'_i =$

$\max\{0, 1 - [\max_{y \in \mathcal{Y}_i} \phi(m_i, y) - \max_{y' \in \overline{\mathcal{Y}}_i} \phi(m_i, y')]\}$ for N'_L unique linkable entity mentions (in set $\mathcal{M}_L$), we define the objective function for modeling types of entity mentions as follows.

$$O_M = \mathcal{L}_{MF} + \sum_{i=1}^{N'_L} \ell'_i + \frac{\lambda}{2} \sum_{i=1}^{N'_L} \| \mathbf{m}_i \|_2^2 + \frac{\lambda}{2} \sum_{k=1}^{K_y} \| \mathbf{y}_k \|_2^2. \tag{6.7}$$

Minimizing the objective O_M yields an entity embedding space where, in that space, objects (*e.g.*, entity mentions, text features) close to each other will have similar types.

Modeling Entity-Relation Interactions. In reality, there exists different kinds of interactions between a relation mention $z = (m_1, m_2, s)$ and its entity mention arguments m_1 and m_2. One major kind of interactions is the correlation between relation and entity types of these objects—entity types of the two entity mentions provide good hints for determining the relation type of the relation mention, and vice versa. For example, in Fig. 6.5 (right column), knowing that entity mention "S_4_*US*" is of type `location` (instead of `organization`) helps determine that relation mention ("*Obama*", "*US*", S_4) is more likely of relation type `travel_to`, rather than relation types like `president_of` or `citizen_of`.

Intuitively, entity types of the entity mention arguments pose constraints on the search space for the relation types of the relation mention (*e.g.*, it is unlikely to find a `author_of` relation between a `organization` entity and a `location` entity). The proposed Hypotheses 6.3.2 and 6.3.2 model types of relation mentions and entity mentions by learning an entity embedding space and a relation embedding space, respectively. The correlations between entity and relation types (and their embedding spaces) motivates us to model entity-relation interactions based on the following hypothesis.

Hypothesis 6.3: Entity-Relation Interaction

For a relation mention $z = (m_1, m_2, s)$, embedding vector of m_1 should be a nearest neighbor of the embedding vector of m_2 plus the embedding vector of relation mention z.

Given the embedding vectors of any two members in $\{z, m_1, m_2\}$, say $\mathbf{z}$ and $\mathbf{m}_1$, Hypothesis 6.3.2 forces the "$\mathbf{m}_1 + \mathbf{z} \approx \mathbf{m}_2$". This helps regularize the learning of vector $\mathbf{m}_2$ (which represents the type semantics of entity mention m_2) in addition to the information encoded by objective O_M in Eq. (6.7). Such a "translating operation" between embedding vectors in a low-dimensional space has been proven effective in embedding entities and relations in

$\mathcal{D}$	Automatically generated training corpus
$\mathcal{M} = \{m_i\}_{i=1}^{N}$	Entity mentions in $\mathcal{D}$ (size N)
$\mathcal{Y} = \{y_k\}_{k=1}^{K}$	Target entity types (size K)
$\mathcal{Y}_i$	Candidate types of m_i
$\overline{\mathcal{Y}}_i$	Non-candidate types of m_i, *i.e.*, $\overline{\mathcal{Y}}_i = \mathcal{Y} \setminus \mathcal{Y}_i$
$\mathcal{F} = \{f_j\}_{j=1}^{M}$	Text features in $\mathcal{D}$ (size M)
$\mathbf{u}_i \in \mathbb{R}^d$	Embedding of mention m_i (dim. d)
$\mathbf{c}_j \in \mathbb{R}^d$	Embedding of feature f_j (dim. d)
$\mathbf{v}_k, \mathbf{v}'_k \in \mathbb{R}^d$	Embeddings of type y_k on two views (dim. d)

Table 6.4: Notations.

a structured knowledge baes [63]. We extend this idea to model the type correlations (and mutual constraints) between embedding vectors of entity mentions and embedding vectors of relation mentions, which are modeled in two different low-dimensional spaces.

Specifically, we define error function for the triple of a relation mention and its two entity mention arguments (z, m_1, m_2) using ℓ-2 norm: $\tau(z) = \| \mathbf{m}_1 + \mathbf{z} - \mathbf{m}_2 \|_2^2$. A small value on $\tau(z)$ indicates that the embedding vectors of (z, m_1, m_2) do capture the type constraints. To enforce small errors between linkable relation mentions (in set $\mathcal{Z}_L$) and their entity mention arguments, we use margin-based loss [63] to formulate a objective function as follows.

$$O_{ZM} = \sum_{z_i \in \mathcal{Z}_L} \sum_{v=1}^{V} \max\big\{0, 1 + \tau(z_i) - \tau(z_v)\big\}, \tag{6.8}$$

where $\{z_v\}_{v=1}^{V}$ are negative samples for z, *i.e.*, z_v is randomly sampled from the negative sample set $\{(z', m_1, m_2)\} \cup \{(z, m'_1, m_2)\} \cup \{(z, m_1, m'_2)\}$ with $z' \in \mathcal{Z}_L$ and $m' \in \mathcal{M}_L$ [63]. The intuition behind Eq. (6.8) is simple (see also the right col. in Fig. 6.5): embedding vectors for a relation mention and its entity mentions are modeled in the way that, the translating error τ between them should be *smaller* than the translating error of any negative sample.

A Joint Optimization Problem. Our goal is to embed all the available information for relation and entity mentions, relation and entity type labels, and text features into a d-dimensional entity space and a d-dimensional relation space, following the three proposed hypotheses. An intuitive solution is to *collectively* minimize the three objectives O_Z O_M and O_{ZM}, as the embedding vectors of entity and relation mentions are shared across them. To achieve the goal, we formulate a joint optimization problem as follows.

$$\min_{\{\mathbf{z}_i\},\{\mathbf{c}_j\},\{\mathbf{r}_k\},\{\mathbf{m}_i\},\{\mathbf{c}'_j\},\{\mathbf{y}_k\}} \mathcal{O} = \mathcal{O}_M + \mathcal{O}_Z + \mathcal{O}_{ZM}. \tag{6.9}$$

Optimizing the global objective O in Eq. (6.9) enables the learning of entity and relation embeddings to be *mutually* influenced, such that, errors in each component can be constrained and corrected by the other. The joint embedding learning also helps the algorithm to find the true types for each mention, besides using text features.

In Eq. (6.9), one can also minimize the weighted combination of the three objectives $\{O_Z, O_M, O_{ZM}\}$ to model the importance of different signals, where weights could be manually determined or automatically learned from data. We leave this as future work.

6.3.3 Model Learning and Type Inference

The joint optimization problem in Eq. (6.9) can be solved in multiple ways. One solution is to first learn entity mention embeddings by minimizing $\mathcal{O}_M$, then apply the learned embeddings to optimize $\mathcal{O}_{MZ}+\mathcal{O}_Z$. However, such a solution does not fully exploit the entity-relation interactions in providing mutual feedbacks between the learning of entity mention embeddings and the learning of relation mention embeddings (see CoType-TwoStep in Sec. 6.4).

We design a stochastic sub-gradient descent algorithm [129] based on edge sampling strategy [60], to efficiently solve Eq. (6.9). In each iteration, we alternatively sample from each of the three objectives $\{O_Z, O_M, O_{ZM}\}$ a batch of edges (*e.g.*, (z_i, f_j)) and their negative samples, and update each embedding vector based on the derivatives. The proof procedure in [129] can be adopted to prove convergence of the proposed algorithm (to the local minimum). Eq. (6.9) can also be solved by a mini-batch extension of the Pegasos algorithm [129], which is a stochastic sub-gradient descent method and thus can efficiently handle massive text corpora. Due to lack of space, we do not include derivation details here.

Type Inference. With the learned embeddings of features and types in relation space (*i.e.*, $\{\mathbf{c}_i\}$, $\{\mathbf{r}_k\}$) and entity space (*i.e.*, $\{\mathbf{c}'_i\}$, $\{\mathbf{y}_k\}$), we can perform nearest neighbor search in the target relation type set $\mathbb{R}$, or a top-down search on the target entity type hierarchy $\mathcal{Y}$, to estimate the relation type (or the entity type-path) for each (unlinkable) test relation mention $z \in \mathcal{Z}_U$ (test entity mention $m \in \mathcal{M} \setminus \mathcal{M}_L$). Specifically, on the entity type hierarchy, we start from the tree's root and recursively find the best type among the children types by measuring the *cosine similarity* between entity type embedding and the vector representation of m in our learned entity embedding space. By extracting text features from m's local context (denoted by set $\mathcal{F}_m(m)$), we represent m in the learned entity embedding space using the vector $\mathbf{m} = \sum_{f_j \in \mathcal{F}_m(m)} \mathbf{c}'_j$. Similarly, for test relation mention z, we represent it in our learned relation embedding space by $\mathbf{z} = \sum_{f_j \in \mathcal{F}_z(z)} \mathbf{c}_j$ where $\mathcal{F}_z(z)$ is the set of text features extracted

Data sets	NYT	Wiki-KBP	BioInfer
#Relation/entity types	24 / 47	19 / 126	94 / 2,200
#Documents (in $\mathcal{D}$)	294,977	780,549	101,530
#Sentences (in $\mathcal{D}$)	1.18M	1.51M	521k
#Training RMs (in $\mathcal{Z}_L$)	353k	148k	28k
#Training EMs (in $\mathcal{Z}_L$)	701k	247k	53k
#Text features (from $\mathcal{D}_L$)	2.6M	1.3M	575k
#Test Sentences (from $\mathcal{Z}_U$)	395	448	708
#Ground-truth RMs	3,880	2,948	3,859
#Ground-truth EMs	1,361	1,285	2,389

Table 6.5: Statistics of the datasets in our experiments.

from z's local context s. The search process stops when we reach to a leaf type on the type hierarchy, or the similarity score is below a pre-defined threshold $\eta > 0$. If the search process returns an empty type-path (or type set), we output the predicted type label as `None` for the mention.

Computational Complexity Analysis. Let E be the total number of objects in CoType (entity and relation mentions, text features and type labels). By alias table method [60], setting up alias tables takes $O(E)$ time for all the objects, and sampling a negative example takes constant time. In each iteration of the CoType algorithm, optimization with negative sampling (*i.e.*, optimizing second-order proximity and translating objective) takes $O(dV)$, and optimization with partial-label loss takes $O\big(dV(|\,\mathbb{R}\,|+|\mathcal{Y}|)\big)$ time. Similar to [60], we find the number of iterations for the algorithm to converge is usually *proportional to* the number of object interactions extracted from $\mathcal{D}$ (*e.g.*, unique mention-feature pairs and mention-type associations), denoted as R. Therefore, the overall time complexity of CoType is $O\big(dRV(|\,\mathbb{R}\,|+|\mathcal{Y}|)\big)$ (as $R \geq E$), which is *linear* to the total number of object interactions R in the corpus.

6.4 EXPERIMENTS

6.4.1 Data Preparation and Experiment Setting

Our experiments use three public datasets[1] from different domains. (1) **NYT** [39]: The training corpus consists of 1.18M sentences sampled from ∼294k 1987-2007 New York Times

[1]Codes and datasets used in this paper can be downloaded at: `https://github.com/shanzhenren/CoType`.

news articles. 395 sentences are manually annotated by authors of [71] to form the test data; (2) **Wiki-KBP** [36]: It uses 1.5M sentences sampled from ∼780k Wikipedia articles [36] as training corpus and 14k manually annotated sentences from 2013 KBP slot filling assessment results [120] as test data. (3) **BioInfer** [121]: It consists of 1,530 manually annotated biomedical paper abstracts as test data and 100k sampled PubMed paper abstracts as training corpus. Statistics of the datasets are shown in Table 6.5.

Automatically Labeled Training Corpora. The NYT training corpus has been heuristically labeled using distant supervision following the procedure in [39]. For Wiki-KBP and BioInfer training corpora, we utilized DBpedia Spotlight[2], a state-of-the-art entity disambiguation tool, to map the detected entity mentions $\mathcal{M}$ to Freebase entities. We then followed the procedure introduced in Secs. 6.2 and 6.3.1 to obtain candidate entity and relation types, and constructed the training data $\mathcal{D}_L$. For target types, we discard the relation/entity types which cannot be mapped to Freebase from the test data while keeping the Freebase entity/relation types (not found in test data) in the training data (see Table 6.5 for the type statistics).

Feature Generation. Table 6.3 lists the set of text features of relation mentions used in our experiments. We followed [36] to generate text features for entity mentions. Dependency parse-based features were excluded as only POS-tagged corpus is given as input. We used a 6-word window to extract context features for each mention (3 words on the left and the right). We applied the Stanford CoreNLP tool [105] to get POS tags. Brown clusters were derived for each corpus using public implementation[3]. The same kinds of features were used in all the compared methods in our experiments.

Evaluation Sets. For all three datasets, we used the provided training/test set partitions of the corpora. In each dataset, relation mentions in sentences are manually annotated with their relation types and the entity mention arguments are labeled with entity type-paths (see Table 6.5 for the statistics of test data). We further created a *validation set* by randomly sampling 10% mentions from each test set and used the remaining part to form the *evaluation set*.

Compared Methods. We compared CoType with its variants which model parts of the proposed hypotheses. Several state-of-the-art relation extraction methods (*e.g.*, supervised, embedding, neural network) were also implemented (or tested using their published codes): (1) **DS+Perceptron** [36]: adopts multi-label learning on automatically labeled training data $\mathcal{D}_L$. (2) **DS+Kernel** [28]: applies bag-of-feature kernel [28] to train a SVM classifier using

[2] http://spotlight.dbpedia.org/

[3] https://github.com/percyliang/brown-cluster

Method	NYT			Wiki-KBP			BioInfer		
	S-F1	Ma-F1	Mi-F1	S-F1	Ma-F1	Mi-F1	S-F1	Ma-F1	Mi-F1
FIGER [36]	0.40	0.51	0.46	0.29	0.56	0.54	0.69	0.71	0.71
Google [101]	0.38	0.57	0.52	0.30	0.50	0.38	0.69	0.72	0.65
HYENA [99]	0.44	0.49	0.50	0.26	0.43	0.39	0.52	0.54	0.56
DeepWalk[61]	0.49	0.54	0.53	0.21	0.42	0.39	0.58	0.59	0.61
WSABIE[58]	0.53	0.57	0.58	0.35	0.55	0.50	0.64	0.66	0.65
PLE [56]	0.56	0.60	0.61	0.37	0.57	0.53	0.70	0.71	0.72
CoType	**0.60**	**0.65**	**0.66**	**0.39**	**0.61**	**0.57**	**0.74**	**0.76**	**0.75**

Table 6.6: Performance comparison of entity recognition and typing (using strict, micro and macro metrics [36]) on the three datasets.

$\mathcal{D}_L$; (3) **DS+Logistic** [37]: trains a multi-class logistic classifier[4] on $\mathcal{D}_L$; (4) **DeepWalk** [61]: embeds mention-feature co-occurrences and mention-type associations as a homogeneous network (with binary edges); (5) **LINE** [60]: uses second-order proximity model with edge sampling on a feature-type bipartite graph (where edge weight w_{jk} is the number of relation mentions having feature f_j and type r_k); (6) **MultiR** [71]: is a state-of-the-art distant supervision method, which models noisy label in $\mathcal{D}_L$ by multi-instance multi-label learning; (7) **FCM** [62]: adopts neural language model to perform compositional embedding; (8) **DS-Joint** [48]: jointly extract entity and relation mentions using structured perceptron on human-annotated sentences. We used $\mathcal{D}_L$ to train the model.

For CoType, besides the proposed model, **CoType**, we compare (1) **CoType-RM**: This variant only optimize objective O_Z to learning feature and type embeddings for relation mentions; and (2) **CoType-TwoStep**: It first optimizes $\mathcal{O}_M$, then use the learned entity mention embedding $\{\mathbf{m}_i\}$ to initialize the minimization of $O_Z + O_{ZM}$—it represents a "pipeline" extraction diagram.

To test the performance on entity recognition and typing, we also compare with several entity recognition systems, including a supervised method **HYENA** [99], distant supervision methods (**FIGER** [36], **Google** [101], **WSABIE** [58]), and a noise-robust approach **PLE** [56].

Parameter Settings. In our testing of CoType and its variants, we set $\alpha = 0.025$, $\eta = 0.35$ and $\lambda = 10^{-4}$ based on the analysis on validation sets. For convergence criterion, we stopped the loop in the algorithm if the relative change of $\mathcal{O}$ in Eq. (6.9) is smaller than 10^{-4}. For fair comparison, the dimensionality of embeddings d was set to 50 and the number of negative samples V was set to 5 for all embedding methods, as used in [60]. For other tuning parameters in the compared methods, we tuned them on validation sets and picked

[4]We use liblinear package from `https://github.com/cjlin1/liblinear`

the values which lead to the best performance.

Evaluation Metrics. For entity recognition and typing, we to use strict, micro, and macro F1 scores, as used in [36], for evaluating both detected entity mention boundaries and predicted entity types. We consider two settings in evaluation of relation extraction. For relation classification, ground-truth relation mentions are given and `None` label is excluded. We focus on testing type classification accuracy. For relation extraction, we adopt standard Precision (**P**), Recall (**R**) and F1 score [28, 45]. Note that all our evaluations are *sentence-level* (*i.e.*, context-dependent), as discussed in [71].

6.4.2 Experiments and Performance Study

1. Performance on Entity Recognition and Typing. Among the compared methods, only FIGER [36] can detect entity mention. We apply our detection results (*i.e.*, $\mathcal{M}$) as input for other methods. Table 6.6 summarizes the comparison results on the three datasets. Overall, CoType outperforms others on all metrics on all three datasets (*e.g.*, it obtains a 8% improvement on Micro-F1 over the *next best method* on NYT dataset). Such performance gains mainly come from (1) a more robust way of modeling noisy candidate types (as compared to supervised method and distant supervision methods which ignore label noise issue); and (2) the joint embedding of entity and relation mentions in a mutually enhancing way (vs. the noise-robust method PLE [56]). This demonstrates the effectiveness of enforcing Hypothesis 6.3.2 in CoType framework.

2. Performance on Relation Classification. To test the effectiveness of the learned embeddings

Method	**NYT**	**Wiki-KBP**	**BioInfer**
DS+Perceptron [36]	0.641	0.543	0.470
DS+Kernel [28]	0.632	0.535	0.419
DeepWalk [61]	0.580	0.613	0.408
LINE [60]	0.765	0.617	0.557
DS+Logistic [37]	0.771	0.646	0.543
MultiR [71]	0.693	0.633	0.501
FCM [62]	0.688	0.617	0.467
CoType-RM	0.812	0.634	0.587
CoType-TwoStep	0.829	0.645	0.591
CoType	**0.851**	**0.669**	**0.617**

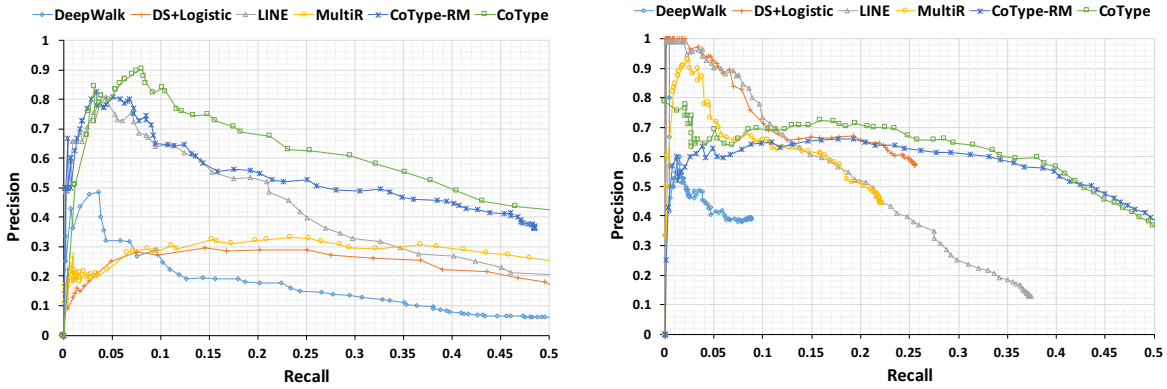
Table 6.7: Performance comparison on relation classification accuracy over ground-truth relation mentions on the three datasets.

Method	NYT [39, 71] Prec	Rec	F1	Time	Wiki-KBP [120, 36] Prec	Rec	F1	Time	BioInfer [121] Prec	Rec	F1	Time
DS+Perceptron [36]	0.068	**0.641**	0.123	15min	0.233	0.457	0.308	7.7min	0.357	0.279	0.313	3.3min
DS+Kernel [28]	0.095	0.490	0.158	56hr	0.108	0.239	0.149	9.8hr	0.333	0.011	0.021	4.2hr
DS+Logistic [37]	0.258	0.393	0.311	25min	0.296	0.387	0.335	14min	**0.572**	0.255	0.353	7.4min
DeepWalk [61]	0.176	0.224	0.197	1.1hr	0.101	0.296	0.150	27min	0.370	0.058	0.101	8.4min
LINE [60]	0.335	0.329	0.332	2.3min	0.360	0.257	0.299	1.5min	0.360	0.275	0.312	35sec
MultiR [71]	0.338	0.327	0.333	5.8min	0.325	0.278	0.301	4.1min	0.459	0.221	0.298	2.4min
FCM [62]	0.553	0.154	0.240	1.3hr	0.151	**0.500**	0.301	25min	0.535	0.168	0.255	9.7min
DS-Joint [48]	**0.574**	0.256	0.354	22hr	**0.444**	0.043	0.078	54hr	0.102	0.001	0.002	3.4hr
CoType-RM	0.467	0.380	0.419	2.6min	0.342	0.339	0.340	1.5min	0.482	0.406	0.440	42sec
CoType-TwoStep	0.368	0.446	0.404	9.6min	0.347	0.351	0.349	6.1min	0.502	0.405	0.448	3.1min
CoType	0.423	0.511	**0.463**	4.1min	0.348	0.406	**0.369**	2.5min	0.536	**0.424**	**0.474**	78sec

Table 6.8: Performance comparison on end-to-end relation extraction (at the highest F1 point) on the three datasets.

in representing type semantics of relation mentions, we compare with other methods on classifying the ground-truth relation mention in the evaluation set by target types $\mathbb{R}$. Table 6.7 summarizes the classification accuracy. CoType achieves superior accuracy compared to all other methods and variants (*e.g.*, obtains over 10% enhancement on both the NYT and BioInfer datasets over the next best method). All compared methods (except for MultiR) simply treat $\mathcal{D}_L$ as "perfectly labeled" when training models. The improvement of CoType-RM validates the importance on careful modeling of label noise (*i.e.*, Hypothesis 6.3.2). Comparing CoType-RM with MultiR, superior performance of CoType-RM demonstrates the effectiveness of partial-label loss over multi-instance learning. Finally, CoType outperforms CoType-RM and CoType-TwoStep validates that the propose translation-based embedding objective is effective in capturing entity-relation cross-constraints.

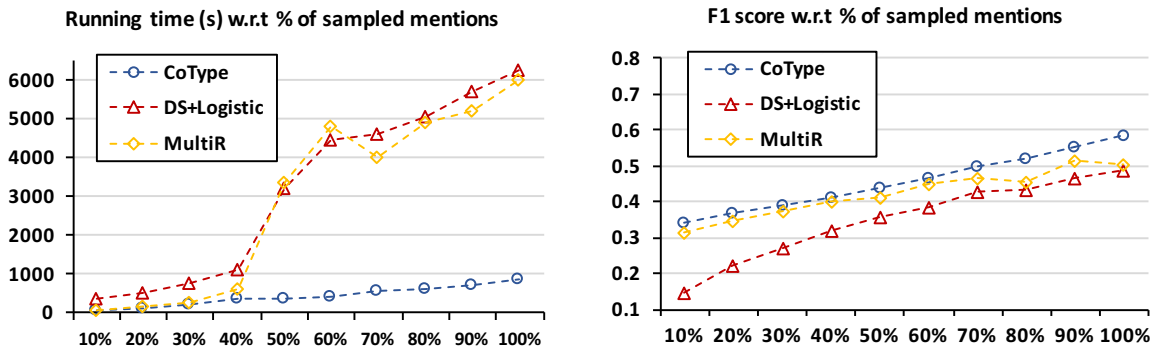
3. Performance on Relation Extraction. To test the domain independence of CoType frame-



(a) NYT (b) BioInfer

Figure 6.6: Precision-recall curves of relation extraction on NYT and BioInfer datasets. Similar trend is also observed on the Wiki-KBP dataset.

work, we conduct evaluations on the end-to-end relation extraction. As only MultiR and DS-Joint are able to detection entity and relation mentions in their own framework, we apply our detection results to other compared methods. Table 6.8 shows the evaluation results as well as runtime of different methods. In particular, results at each method's highest F1 score point are reported, after tuning the threshold for each method for determining whether a test mention is `None` or some target type. Overall, CoType outperforms all other methods on F1 score on all three datasets. We observe that DS-Joint and MultiR suffer from low recall, since their entity detection modules do not work well on $\mathcal{D}_L$ (where many tokens have false negative tags). This demonstrates the effectiveness of the proposed domain-agnostic text segmentation algorithm (see Sec. 6.3.1). We found that the incremental diagram of learning embedding (*i.e.*, CoType-TwoStep) brings only marginal improvement. In contrast, CoType adopts a "joint modeling" diagram following Hypothesis 6.3.2 and achieves significant improvement. In Fig. 6.6, precision-recall curves on NYT and BioInfer datasets further show that CoType can still achieve descent precision with good recall preserved.



(a) Scalability of CoType

(b) Effect of training set size

Figure 6.7: (a) Scalability study on CoType and the compared methods; and (b) Performance changes of relation extraction with respect to sampling ratio of relation mentions on the **Bioinfer** dataset.

4. Scalability. In addition to the runtime shown in Table 6.8, Fig. 6.7(a) tests the scalability of CoType compared with other methods, by running on BioInfer corpora sampled using different ratios. CoType demonstrates a linear runtime trend (which validates our time complexity in Sec. 6.3.3), and is the only method that is capable of processing the full-size dataset without significant time cost.

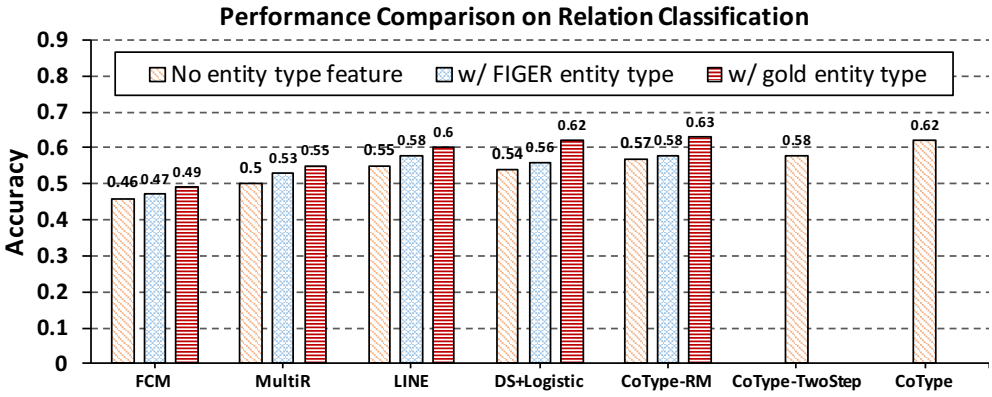


Figure 6.8: Study of entity type error propagation on the **BioInfer** dataset.

6.5 RELATED WORK

Entity and Relation Extraction. There have been extensive studies on extracting typed entities and relations in text (*i.e.*, context-dependent extraction). Most existing work follows an *incremental* diagram—they first perform entity recognition and typing [44, 26] to extract typed entity mentions, and then solve relation extraction [45, 47] to identify relation mentions of target types. Work along both lines can be categorized in terms of the degree of supervision. While supervised entity recognition systems [25, 44] focus on a few common entity types, weakly-supervised methods [31, 34] and distantly-supervised methods [8, 58, 36] use large text corpus and a small set of seeds (or a knowledge base) to induce patterns or to train models, and thus can apply to different domains without additional human annotation labor. For relation extraction, similarly, weak supervision [68, 3] and distant supervision [69, 70, 38, 71, 39, 37] approaches are proposed to address the domain restriction issue in traditional supervised systems [45, 28, 47]. However, such a "pipeline" diagram ignores the dependencies between different sub tasks and may suffer from error propagation between the tasks.

Recent studies try to integrate entity extraction with relation extraction by performing global sequence labeling for both entities and relations [48, 49, 72], incorporating type constraints between relations and their arguments [50], or modeling factor graphs [73]. However, these methods require human-annotated corpora (cleaned and general) for model training and rely on existing entity detectors to provide entity mentions. By contrast, the CoType framework runs domain-agnostic segmentation algorithm to mine entity mentions and adopts a label noise-robust objective to train models using distant supervision. In particular, [72] integrates entity classification with relation extraction using distant supervision but it ignores label noise issue in the automatically labeled training corpora.

CoType combines the best of two worlds—it leverages the noisy distant supervision in a robust way to address domain restriction (vs. existing joint extraction methods [48, 49]), and models entity-relation interactions jointly with other signals to resolve error propagation (vs. current distant supervision methods [38, 37]).

Learning Embeddings and Noisy Labels. Our proposed framework incorporates embedding techniques used in modeling words and phrases in large text corpora [57, 58, 59] ,and nodes and links in graphs/networks [60, 61]. Theses methods assume links are all correct (in unsupervised setting) or labels are all true (in supervised setting). CoType seeks to *model the true links and labels* in the embedding process (*e.g.*, see our comparisons with LINE [60], DeepWalk [61] and FCM [62] in Sec. 6.4.2). Different from embedding structured KB entities and relations [63, 64], our task focuses on embedding entity and relation mentions in *unstructured* contexts.

In the context of modeling noisy labels, our work is related to partial-label learning [56, 65, 66] and multi-label multi-instance learning [38], which deals with the problem where each training instance is associated with a set of noisy candidate labels (where *only one is correct*). Unlike these formulations, our *joint* extraction problem deals with both *classification with noisy labels* and *modeling of entity-relation interactions*. In Sec 6.4.2, we compare our full-fledged model with its variants CoType-EM and CoType-RM to validate the Hypothesis on entity-relation interactions.

6.6 DISCUSSION

1. Example output on news articles. Table 6.9 shows the output of CoType, MultiR and Logistic on two news sentences from the Wiki-KBP dataset. CoType extracts more relation mentions (*e.g.*, `children`), and predict entity/relation types with better accuracy. Also, CoType can jointly extract typed entity and relation mentions while other methods cannot (or need to do it incrementally).

2. Testing the effect of training corpus size. Fig. 6.7(b) shows the performance trend on Bioinfer dataset when varying the sampling ratio (subset of relation mentions randomly sampled from the training set). F1 scores of all three methods improves as the sampling ratio increases. CoType performs best in all cases, which demonstrates its robust performance across corpora of various size.

3. Study the effect of entity type error in relation classification. To investigate the "error propagation" issue of incremental pipeline, we test the changes of relation classification performance by (1) training models without entity types as features; (2) using entity types

predicted by FIGER [36] as features; and (3) using ground-truth ("perfect") entity types as features. Fig. 6.8 summarize the accuracy of CoType, its variants and the compared methods. We observe only marginal improvement when using FIGER-predicted types but significant improvement when using ground-truth entity types—this validates the error propagation issue. Moreover, we find that CoType achieves an accuracy close to that of the next best method (*i.e.*, DS + Logistic + Gold entity type). This demonstrates the effectiveness of our proposed joint entity and relation embedding.

Text	***Blake Edwards***, a prolific filmmaker who kept alive the tradition of slapstick comedy, died Wednesday of pneumonia at a hospital in ***Santa Monica***.	***Anderson*** is survived by his wife Carol, sons ***Lee*** and Albert, daughter Shirley Englebrecht and nine grandchildren.
MultiR [71]	r^*: person:country_of_birth, $\mathcal{Y}_1^*$: {N/A}, $\mathcal{Y}_2^*$: {N/A}	r^*: None, $\mathcal{Y}_1^*$: {N/A}, $\mathcal{Y}_2^*$: {N/A}
Logistic [37]	r^*: per:country_of_birth, $\mathcal{Y}_1^*$: {person}, $\mathcal{Y}_2^*$: {country}	r^*: None, $\mathcal{Y}_1^*$: {person}, $\mathcal{Y}_2^*$: {person, politician}
CoType	r^*: person:place_of_death, $\mathcal{Y}_1^*$: {person,artist,director}, $\mathcal{Y}_2^*$: {location, city}	r^*: person:children, $\mathcal{Y}_1^*$: {person}, $\mathcal{Y}_2^*$: {person}

Table 6.9: Example output of CoType and the compared methods on two news sentences from the **Wiki-KBP** dataset.

6.7 SUMMARY

This work studies domain-independent, joint extraction of typed entities and relations in text with distant supervision. The proposed CoType framework runs domain-agnostic segmentation algorithm to mine entity mentions, and formulates the joint entity and relation mention typing problem as a global embedding problem. We design a noise-robust objective to faithfully model noisy type label from distant supervision, and capture the mutual dependencies between entity and relation based on the translation embedding assumption. Experiment results demonstrate the effectiveness and robustness of CoType on text corpora of different domains.

Interesting future work includes incorporating pseudo feedback idea [70] to reduce false negative type labels in the training data, modeling type correlation in the given type hierarchy [56], and performing type inference for test entity mention and relation mentions jointly. CoType relies on minimal linguistic assumption (*i.e.*, only POS-tagged corpus is required) and thus can be extended to different languages where pre-trained POS taggers is available.

Part III

Conclusions and Future Directions

CHAPTER 7: APPLICATION DISCUSSION AND CONCLUSIONS

Entities and relationships are important structures that can be extracted from a text corpus to represent the factual knowledge inside the corpus. Effective and efficient mining of entity and relation structures from text helps gaining insights from large volume of text data (that are infeasible for human to read through and digest), and enables many downstream applications on understanding, exploring and analyzing the text content. Data analysts and government agents may want to identify person, organization and location entities in news everyday news articles and generate concise and timely summary of news events. Biomedical researchers who cannot digest large amounts of newly-published research papers in relevant areas would need an effective way to extract different relationships between proteins, drugs and diseases so as to follow the new claims and facts presented in the research community. However, text data is highly variable: corpora covering topics from different domains, written in different genres or languages have typically required for effective processing a wide range of language resources such as grammars, vocabularies, gazetteers. The *massive* and *messy* nature of text data post significant challenges to creating tools for automated structuring of unstructured content that scale with text volume.

7.1 EFFORT-LIGHT STRUCTMINE: SUMMARY

In this thesis, we focus on principled and scalable methods for the mining of typed *entity* and *relation* structures from unstructured text corpora in order to overcome the barriers in dealing with text corpora of various domains, genres and languages. As traditional information extraction approaches have relied on large amounts of task-specific labeled data, my thesis work harnesses the power of "big text data" and focuses on creating generic solutions for *efficient construction of customized machine-learning models for factual structure extraction*, relying on only limited amounts of (or even no) task-specific training data. Our proposed methods aim to bridge the gap between customized machine-learning models and the absence of high-quality task-specific training data. It leverages the information overlap between background facts stored in external knowledge bases (KBs) and the given corpus to automatically generate large amounts of (possibly noisy) task-specific training data; and it exploits redundant text information within the massive corpus to reduce the complexity of feature generation (*e.g.*, sentence parsing). This solution is based on two key intuitions which are described below. Overall, the thesis has made the contributions on mining entity and relation structures in the following aspects.

1. We propose **three key principles** on systematically mining typed entities and relationships from massive corpora, using distant supervision in conjunction with knowledge bases.

- **Automatic labeled data generation by aligning corpus with knowledge bases**. In a massive corpus, structured information about some of the entities (*e.g.*, entity types, relationships to other entities) can be found in external KBs. Can we align the corpus with external KBs to automatically generate training data for extracting entity and relation structures at a large scale? Such retrieved information supports the automated annotation of entities and relations in text and labeling of their categories, yielding (possibly noisy) corpus-specific training data. Although the overlaps between external KBs and the corpus at hand might involve only a small proportion of the corpus, the scale of the automatically labeled training data could still be much larger than that of manually annotated data by domain experts.

- **Type propagation via co-occurring text features**. Text units (*e.g.*, word, phrase) co-occur frequently with entities and other text units in a massive corpus. We can exploit the textual co-occurrence patterns to characterize the semantics of text units, entities, and entity relations. As such patterns become more apparent in a massive corpus with rich data redundancy, big text data leads to big opportunities in representing semantics of text unit without complex feature generation. This is a principle that go through all the chapters, mainly illustrated in Chapter 4.

- **Model semantic similarity by exploiting co-occurrence patterns**. Text units used as features in type propagation framework are highly variable – one string can have multiple semantic meanings and one object can be expressed using different strings. We propose to learn the low-dimensional representations to model the semantic meaning of text units based on their surrounding context (*i.e.*, distributional assumption). With effective semantic representation, we are able to group similar text units together to facilitate type propagation (*i.e.*, overcome sparsity issue for infrequent text units). This is a principle that go through all the chapters, mainly illustrated in Chapter 4 and Chapter 6.

2. We study **different structure extraction tasks** for mining typed entity and relation structures from corpora, which include entity recognition and typing (Chapter 4), fine-grained entity typing (Chapter 5), and entity relationship extraction (Chapter 6). In particular, we investigate human effort-light solutions for these several tasks using distant

supervision in conjunction with external knowledge bases. This yields different problem settings as compared to the fully-supervised learning problem setup in most existing studies on information extraction. A key challenge in dealing with distant supervision is on designing effective typing models that are robust to the noisy labels in the automatically generated training data.

3. We have proposed **models** and **algorithms** to solve the above tasks.

- We studied distantly-supervised entity recognition and typing, and proposed a novel relation phrase-based entity recognition framework, ClusType (Chapter 4). A domain-agnostic phrase mining algorithm is developed for generating candidate entity mentions and relation phrases. By integrating relation phrase clustering with type propagation, the proposed method is effective in minimizing name ambiguity and context problems, and thus predicts each mention's type based on type distribution of its string name and type signatures of its surrounding relation phrases. We formulate a joint optimization problem to learn object type indicators/signatures and cluster memberships simultaneously.

- For fine-grain entity typing, we propose hierarchical partial-label embedding methods, AFET and PLE , that models "clean" and "noisy" mentions separately and incorporates a given type hierarchy to induce loss functions (Chapter 5). Both models build on a joint optimization framework, learns embeddings for mentions and type-paths, and iteratively refines the model.

- Our work on extracting typed relationships studies domain-independent, joint extraction of typed entities and relationships with distant supervision (Chapter 6). The proposed CoType framework runs domain-agnostic segmentation algorithm to mine entity mentions, and formulates the joint entity and relation mention typing problem as a global embedding problem. We design a noise-robust objective to faithfully model noisy type label from distant supervision, and capture the mutual dependencies between entity and relation based on the translation embedding assumption.

7.2 APPLICATIONS

Our work on effort-light StructMine is used in several downstream applications, has led to a few lines of follow-up research, and yield real-world impact. We start with discussing how to build on top of distant supervision to incorporate human supervision (*e.g.*, curated rules from domain experts) in the effort-light StructMine framework, followed by showing a

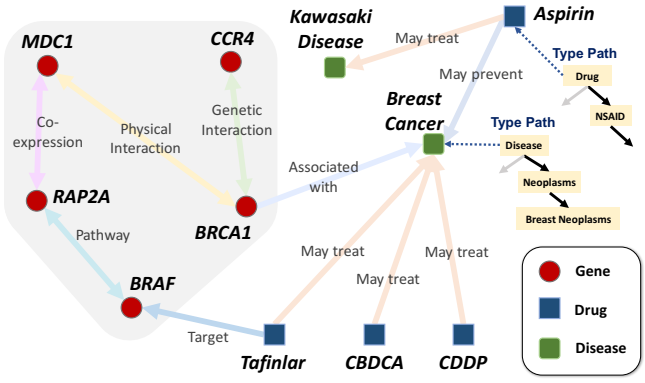
real application on life sciences domain that make use of the StructNet constructed by our methods, and introducing several other applications of our proposed work.

7.2.1 Structuring Life Science Papers: Life-iNet System

Biomedical literature is one of the major sources storing biomedical knowledge and new research findings. A lot of useful information, *e.g.*, new drugs discovered and new bio-molecular interactions, are deeply buried in the literatures. Currently, the most common way to dig out these information is by human curation. For example, bio-curators will manually read each paper and try to assign the most appropriate MeSH terms for each paper to facilitate further literature retrieval. Also, they will manually extract the major biomedical entities (*e.g.*, genes, proteins, drugs, diseases) and their related information from each paper and add the extracted information into human curated databases (*e.g.*, MeSH, UniProt, DrugBank, KEGG, GO) to facilitate further biomedical research. National Institute of Health has a big group of human annotators performing manual literature annotation. This process is relatively slow compared with the rapid growth of the number of published biomedical literatures each year, and costs a large amount of money. Therefore, developing an accurate and efficient way to automatically extract information from literatures has great significance in facilitating future biomedical research. For example, in the following sentence (taken from a PubMed publication with the PMID 383855) one can identify several biomedical entities and their relationships.

Example 7.1 (Biomedical Entity Relationships) *These murine models demonstrate that amikacin has in vivo activity against Nocardia and may be potentially useful in the treatment of human disease.*

The above sentence presents a fact that "*amikacin*" is a `chemical` entity, and claims the finding that "*amikacin*" can potentially treat '*Nocardia*", which is a `disease`. Without tools for mining entity and relation structures such as effort-light StructMine, human experts have to read through the whole sentence to identify the chemical and disease entities in the sentence, and then infer their relationship as a `treatment` relationship from the sentence. However, text mining tools, such as CoType [13], will be able to take the large document collection and some existing biomedical databases as input, and automatically recognize "*amikacin*" as a chemical and "*Nocardia*" as a disease and further infer that there is a treatment relation between them. This example shows that automatic techniques for mining entity and relation structures can greatly save time, human effort and costs for biomedical



Properties	Statistics
# of entity categories (as in MESH)	1,116
# of relation types (as in UMLS)	414
# nodes (entities) in Life-iNet	192M
# edges (relationships) in Life-iNet	1.29B
# of indexed sentences	266M

Figure 7.1: An illustrative example of the constructed Life-iNet, and its statistics.

information extraction from literatures, which serves as a primary step for many downstream applications such as new drug discovery, adverse event detection for drug combination, and biomedical knowledge base construction.

As a follow-up effort, we develop a novel system, called Life-iNet [15] on top of our entity recognition and relation extraction methods, which automatically turns an *unstructured* background corpus into a *structured* network of factual knowledge (see Figure 7.1), and supports multiple exploratory and analytic functions over the constructed network for knowledge discovery. To extract factual structures, Life-iNet automatically detects token spans of entities mentioned from text (*i.e.*, ClusType [8]), labels entity mentions with semantic categories (*i.e.*, PLE [11]), and identifies relationships of various relation types between the detected entities (*i.e.*, CoType [13]). These inter-related pieces of information are integrated to form a unified, structured network, where nodes represent different types of entities and edges denote relationships of different relation types between the entities. To address the issue of limited diversity and coverage, Life-iNet relies on the external knowledge bases to provide seed examples (*i.e.*, *distant supervision*), and identifies additional entities and relationships from the given corpus (*e.g.*, using multiple textual resources such as scientific literature and encyclopedia articles) to construct a structured network. By doing so, we integrate the factual information in the existing knowledge bases with those extracted from the given corpus. To support analytic functionality, the Life-iNet system implements link prediction functions over the construct network and integrates a distinctive summarization function to provide insight analysis (*e.g.*, answering questions such as "*which genes are distinctively related to the given disease type under* `GeneDiseaseAssociation` *relation?*")

To systematically incorporate these ideas, Life-iNet leverages the novel entity and relation structure mining techniques [13, 11, 8] developed in effort-light StructMine to implement an *effort-light network construction framework*. Specially, it relies on distant supervision in

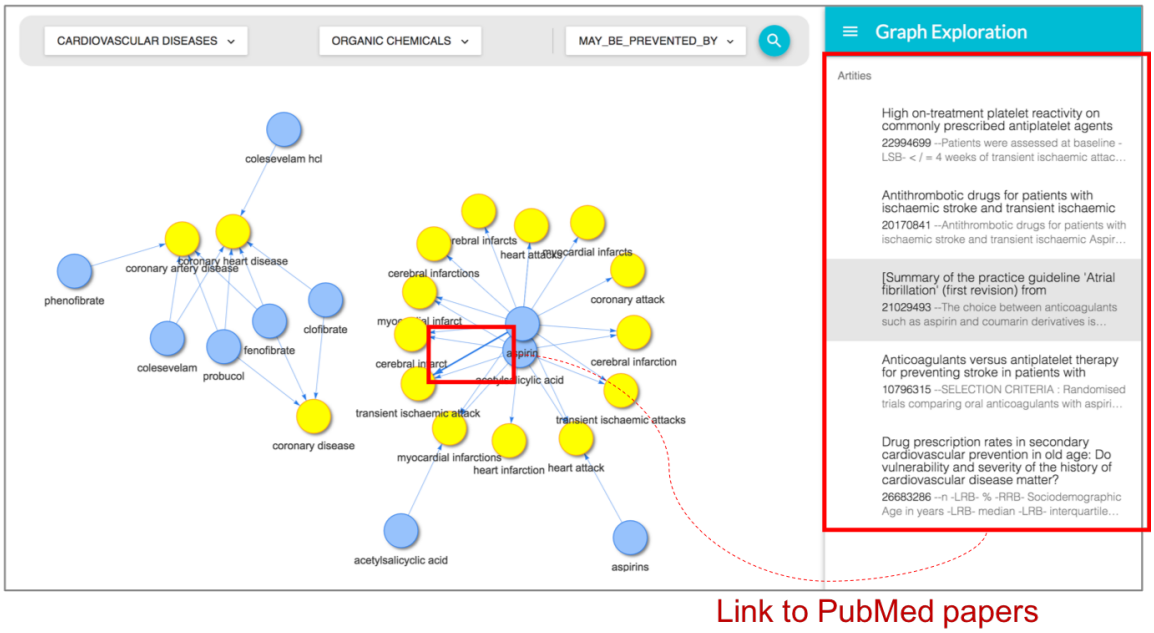


Figure 7.2: A screenshot of the graph exploration interface of Life-iNet system. By specifying the types of two entity arguments and the relation type between them, Life-iNet system returns a graph which visualize the typed entities and relationship, and allows users to explore the graph to find relevant research papers.

conjunction with external knowledge bases to (1) detect quality entity mentions [8], (2) label entity mentions with fine-grained entity types in a given type hierarchy [11], and (3) identify relationships of different types between entities [13]. In particular, we design specialized loss functions to faithfully model "*appropriate*" labels and remove "*false positive*" labels for the training instances (heuristically generated by distant supervision), regarding the specific context where an instance is mentioned [13, 11]. By doing so, we can construct *corpus-specific* information extraction models by using distant supervision in a noise-robust way (see Figure 7.1). The proposed network construction framework is domain-independent—it can be quickly ported to other disciplines and sciences without additional human labeling effort. With the constructed network, Life-iNet further applies link prediction algorithms [60, 63] to infer new entity relationships, and distinctive summarization algorithm [130] to find other entities that are distinctively related to the query entity (or the given entity types).

Impact of Life-iNet:

- A biomedical knowledge graph constructed by our Life-iNet system is used by researchers at Stanford Medical school to facilitate drug re-purposing. It yields significant improvement of performance on new drugs and rare diseases.
- Life-iNet system is adopted by veterinarians at Veterinary Information Network Inc.

Technique	**Application**
conditional random field; unsupervised learning; support vector machine; hidden markov model	Document summarization; sequence labeling; statistical classification
Evaluation Metric	**Dataset**
F1; Rouge-2	DUC

Table 7.1: Example of extracted facets for a research publication.

(VIN) to construct the first veterinary knowledge graph from multiple sources of information including research articles, books, guidelines, drug handbooks and message board posts.

- Technologies developed in Life-iNet system have been transferred to Mayo Clinic, UCLA Medical School, and NIH Big Data to Knowledge Center to facilitate construction of domain knowledge bases from massive scientific literature.

7.2.2 Extracting Document Facets from Technical Corpora

With the ever-increasing number of technical documents being generated every day, including, but not limited to, patent folios, legal cases, real-estate agreements, historical archives, and scientific literature, there is a crucial need to develop automation that can identify the *concepts* for *key facets* for each document, so that readers can quickly get a sense for what the document is about, or search and retrieve documents based on these facets. Consider the domain of scientific publications, one we are all intimately familiar with. Given a new scientific paper, it is impossible for a reader to instantly understand the *techniques* being used, the kinds of *applications* that are addressed, or the *metrics* that are used to ascertain whether the techniques have good performance. Thus, we pose the following question: *Can we develop algorithms that can efficiently and automatically identify the key facets of each document in a large technical document corpora, with little manual supervision?*

Therefore, we identify a novel research problem, called Facet Extraction, in making sense of a large corpus of technical documents. Given a collection of technical documents, the goal of facet extraction is to automatically label each document with a set of concepts for the key facets (*e.g.*, application, technique, evaluation metrics, and dataset) that people may be interested in. The result of Facet Extraction is a summary of the major information of each document into a structured, multi-dimensional representation format, where the target

facets serve as different attributes, and extracted concepts correspond to the attribute values (see Table 7.1).

Extracted facets largely enrich the original structured bibliographic meta information (*e.g.*, authors, venues, keywords), and thus enables a wide range of interesting applications. For example, in a literature search, facets can be used to answer questions such as "which techniques are used in this paper?" and "what are the applications of this work?" (see Table 7.1), which require a deeper understanding of the paper semantics than analyzing the author-generated keyword list. One can also answer questions like "what are the popular applications in the Natural Language Processing or the Database Systems community?" and "how does the facet of *entity recognition* vary across different communities?", by aggregating the facets statistics across the database. Such results enable the discovery of ideas and the dynamics of a research topic or community in an effective and efficient way.

Our ClusType method [8] leverages relation phrase as the bridge to propagate type information. The proposed relation-based framework is general, and can be applied to different kinds of classification task. Therefore, we propose to extract document facets by doing type propagation on corpus-induced graphs. The major challenge in performing facet extraction arises from multiple sources: concept extraction, concept to facet matching, and facet disambiguation. To tackle these challenges, we extend ClusType approach and develop FacetGist, a framework for facet extraction. Facet extraction involves constructing a graph-based heterogeneous network to capture information available across multiple *local* sentence-level features, as well as *global* context features. We then formulate a joint optimization problem, and propose an efficient algorithm for graph-based label propagation to estimate the facet of each concept mention. Experimental results on technical corpora from two domains demonstrate that FacetGist can lead to an improvement of over 25% in both precision and recall over competing schemes [131].

7.2.3 Comparative Document Analysis

In many use cases, people want to have a concise yet informative summary to describe the common and different places between two documents or two set of documents. One of our recent work presents a novel research problem, *Comparative Document Analysis* (*CDA*), that is, *joint* discovery of commonalities and differences between two individual documents (or two sets of documents) in a large text corpus. Given any pair of documents from a (background) document collection, CDA aims to automatically identify sets of quality phrases (entities) to summarize the commonalities of *both* documents and highlight the distinctions of each *with respect to the other* informatively and concisely. It makes use of the output from our

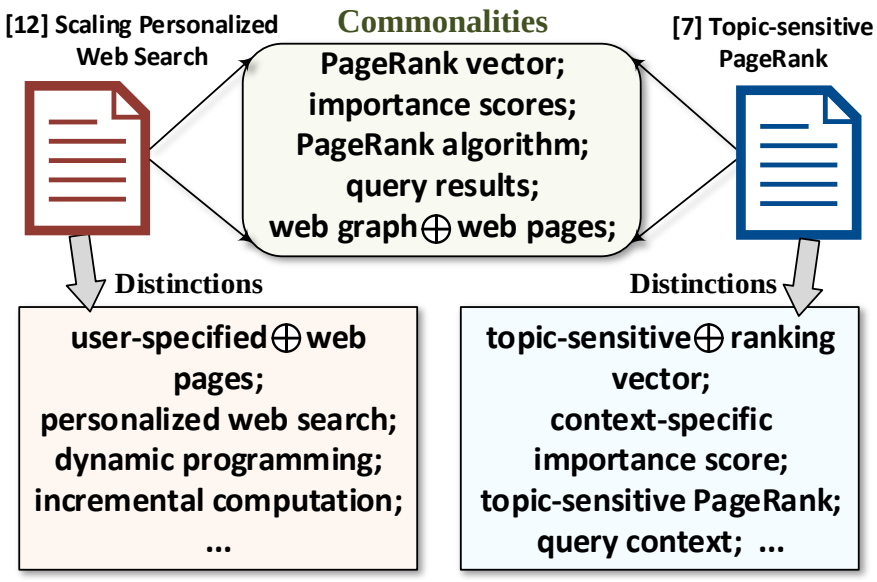


Figure 7.3: Example output of comparative document analysis (CDA) for papers [132] and [133]. CDA combines two proper names which frequently co-occur in the documents into a *name pair* using the symbol "⊕".

entity recognition and typing method to generate candidate phrases for each document.

While there has been some research in comparative text mining, most of these focus on generating word-based or sentence-based summarization for sets of documents. Word-based summarization [134, 135] suffers from limited readability as single words are usually non-informative and bag-of-words representation does not capture the semantics of the original document well—it may not be easy for users to interpret the combined meaning of the words. Sentence-based summarization [136, 137, 138], on the other hand, may be too verbose to accurately highlight the *general* commonalities and differences—users may be distracted by the irrelevant information contained there (as later shown in our case study). Furthermore, previous work compares two sets of documents using redundant contents (*e.g.*, word overlap) but the task becomes much more challenging when comparing two *individual* documents, as there exist a limited number of common content units between the two documents.

We study a novel comparative text mining problem which leverages *multi-word noun phrases* (*i.e.*, proper names) to represent the common and distinct information between *two individual documents* (or two sets of documents), by referring to a massive background corpus for measuring semantic relevance between documents and phrases. We refer the task as Comparative Document Analysis (CDA): Given a pair of documents from a document collection, the task is to (1) extract from each document salient phrases and phrase pairs which cover its major content; (2) discover the commonalities between the document pair by selecting salient phrases which are semantically relevant to *both* of them; and (3) find the

distinctions for each document by selecting salient phrases that are *exclusively* relevant to the document. CDA can benefit a variety of applications including related item recommendation and document retrieval. For example, as shown in Fig. 6.1, a citation recommendation system can show users the common and distinct concepts produced by CDA to help them understand the connections and differences between a query paper [132] and a recommended paper [133]. In a similar way, CDA can reduce the efforts on patentbility searching [139].

Our solution uses a general graph-based framework to derive novel measures on phrase *semantic commonality* and *pairwise distinction*, where the background corpus is used for computing phrase-document semantic relevance. We use the measures to guide the selection of sets of phrases by solving two joint optimization problems. A scalable iterative algorithm is developed to integrate the maximization of phrase commonality or distinction measure with the learning of phrase-document semantic relevance. Experiments on large text corpora from two different domains—scientific papers and news—demonstrate the effectiveness and robustness of the proposed framework on comparing documents. Analysis on a 10GB+ text corpus demonstrates the scalability of our method, whose computation time grows linearly as the corpus size increases. Our case study on comparing news articles published at different dates shows the power of the proposed method on comparing sets of documents.

7.2.4 Mining Meta Patterns for Attribute-Value Extraction

Mining textual patterns in news, tweets, papers, and many other kinds of text corpora has been an active theme in text mining and NLP research. Previous studies adopt a dependency parsing-based pattern discovery approach. However, the parsing results lose rich context around entities in the patterns, and the process is costly for a corpus of large scale. In this follow-up work, we study a novel *typed textual pattern structure*, called *meta pattern*, which is extended to a frequent, informative, and precise subsequence pattern in certain context. We propose an efficient framework, called MetaPAD, which discovers meta patterns from massive corpora with three techniques: (1) it develops a context-aware segmentation method to carefully determine the boundaries of patterns with a learned pattern quality assessment function, which avoids costly dependency parsing and generates high-quality patterns; (2) it identifies and groups synonymous meta patterns from multiple facets—their types, contexts, and extractions; and (3) it examines type distributions of entities in the instances extracted by each group of patterns, and looks for appropriate type levels to make discovered patterns precise.

We propose *meta-pattern mining*, a data-driven method that mines and uses meta patterns to discover attribute names and values of entities. A **meta pattern** refers to a sequence of

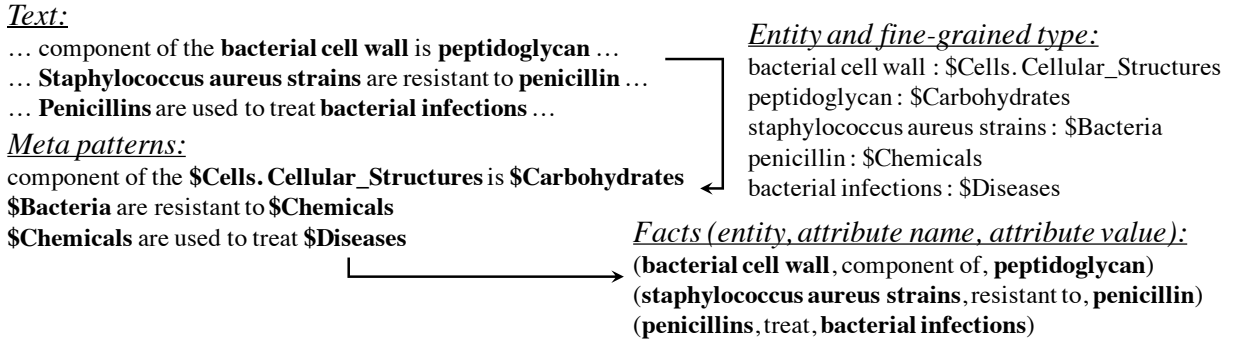


Figure 7.4: Discovering structured facts from the text by mining meta patterns: after replacing entities with their class names, the meta patterns (*i.e.*, segments of class symbols, words, phrases and possibly marks) become apparent, suggesting attribute names and values of the entities.

class symbols (*e.g.*, $BACTERIA, $CHEMICALS), words (*e.g.*, "treat"), phrases (*e.g.*, "component of", "resistant to") and possibly punctuation marks that appear contiguously and frequently in the text and serves as an integral semantic unit of the classes in certain context. We develop a framework called Meta Pattern-driven Attribute Discovery (METAPAD). Figure 7.4 illustrates how METAPAD automatically extracts entities and their attributes from the PubMed corpus. Suppose we can replace "penicillins" with the type "$CHEMICALS" and "bacterial infections" with the type $DISEASES, the pattern "$CHEMICALS are used to treat $DISEASES" becomes apparent in the corpus. Taking the meta pattern back to the text, we are able to find the facts that are (entity, attribute name, attribute value) triplets, *e.g.*, (penicillins, treat, bacterial infections). We introduce the details of METAPAD as follows.

First, METAPAD integrates data-driven text mining techniques to "translate" the documents into long sequences of the four basic kinds of elements of the meta pattern.

- *Quality phrase mining:* SEGPHRASE [126] is a data-driven method that explores automated quality phrase extraction and phrasal segmentation. In Figure 7.4, SEGPHRASE extracted quality phrases such as "bacterial cell wall", "bacterial infections" and "resistant to".

- *Entity recognition and typing:* CLUSTYPE [8] finds entity and their types with distant supervision from Knowledge Bases (e.g., MeSH databases, Freebase, Wikipedia). It integrates relation phrase clustering with type propagation for entity type prediction. With CLUSTYPE, we type "bacterial infections" as $DISEASES and type "penicillins" as $CHEMICALS.

- *Fine-grained typing:* PLE [11] uses embedding to model hierarchical type dependency that reduces label noise in distant supervision for fine-grained entity typing. Thus, we

are able to type "bacterial cell wall" as $Cells.Cellular_Structures and type "staphylococcus aureus strains" as $Organisms.Bacteria.

Second, MetaPAD develops three modules to address the issues of quality, rarity and granularity in Meta Pattern Mining: (1) quality meta pattern classifier, (2) synonym meta pattern detection, and (3) typing in meta patterns for appropriate granularity.

We implement a preliminary version of MetaPAD as well as Google's Biperpedia [140]. Unfortunately, we do not have real query log data. We compare F1 scores of the two methods on extracting entities' attributes from 4 general datasets including 3 from news and 1 from tweets. We use Freebase data for distant supervision. MetaPAD consistently gives high precision as well as recall, and outperforms the baseline method. In addition, the results demonstrate that each part that has been integrated into the framework has contributed to significant increase on the F1 score.

7.2.5 Open Information Extraction with Global Structure Constraints

Our work on typed entity and relation extraction focus on a pre-defined set of entity or relation types. ReMine is a novel open-domain information extraction(Open IE) system that integrates local context signal and global structural signal in a unified framework with distant supervision. Prior work on Open IE can be summarized as sharing two common characteristics: (1) conducting extraction based on local context information; and (2) adopting an incremental system pipeline.

Current Open IE systems focus on analyzing the local context within individual sentences to extract entity and their relationships, while ignoring the redundant information that can be collectively referenced across different sentences and documents in the corpus. Previously, ClusType has reduced local segmentation results via type propagation on text co-occurrence graph constructed from corpus. Having the same intuition, ReMine designs an effective way to measure quality of candidate relation tuple from the rich information redundancy in the massive corpus. For example, entity phrases London and Paris frequently co-occur with similar relation phrase and tail entities in the corpus. One can infer that they have close semantics (same for Great Britain and France). On one hand, it enhances that (Paris, is in, France) is a quality tuple if knowing (London, is in, Great Britain) is a good tuple. On the other, in sentence "[Louvre-Lens], a museum approximately 200 kilometers northwest of [Paris], is building striking [new satellites] to display parts of their collection.", this helps rule out the tuple (Paris, build, new satellites) as Louvre-Lens is semantically distant from Paris.

Most existing Open IE systems are composed of entity detection tools (e.g., named entity recognizer (NER), noun phrase (NP) chunker) relation tuple extraction module. The NERs and NP chunkers are often pre-trained for general domain and may not work well on a domain-specific corpus (e.g., biomedical papers, social media posts). Error propagation are inevitable in such two-step pipelines. To address this problem, CoType adopted distant supervision to minimize to minimize language gap between different domains. Furthermore, CoType demonstrated that low-dimensional vector representations of entity and relations help to reduce noise introduced by distant supervision. More generally, ReMine managed to extract general relation tuples rather than factual knowledge with specific relation types. Inspired by successful application of translative objective on knowledge base completion, ReMine measures quality of extracted tuples via a translative-based objective and incorporate it into an effective sentence segmentation framework. Overall, ReMine is an End-to-End pipeline that jointly optimizes both the extraction of entity and relation phrases and the global cohesiveness across the corpus. Experiments on massive real-world corpus demonstrate effectiveness and robustness of ReMine when compared with other open IE systems. Global statistics prunes wrong extractions from local context and lead to more valuable tuples.

7.3 INCORPORATING RULES FROM DOMAIN EXPERTS: HETEROGENEOUS SUPERVISION

In the thesis, we have been focusing on how to make use of external knowledge bases as distant supervision to automatically generate large amounts of (potentially noisy) training data for the task at hand, and propose noise-robust methods to learn effective typing models over the noisy labeled data. We have seen great successes on leveraging distant supervision to conduct effective and scalable extraction of typed entities and relationships. Such a distant supervision setting, however, may lead to two issues in some use cases: (1) in some domains, there may not exist any externally available knowledge bases for generating distant supervision, or the existing knowledge base is at very small scalable, resulting in insufficient amounts of training data; and (2) in some cases it is easy to collect certain amount of human-annotated data for the extraction tasks in addition to external knowledge bases, but there is no principled way to integrate these two sources of supervision.

In a recent follow-up work of CoType [13], we propose a general framework, *heterogeneous supervision* [141], which unifies various weak supervision sources for relation extraction (*e.g.*, knowledge base and domain-specific patterns). As shown in Figure 7.5, these supervisions often conflict with each other [142]. To address these conflicts, data programming [142]

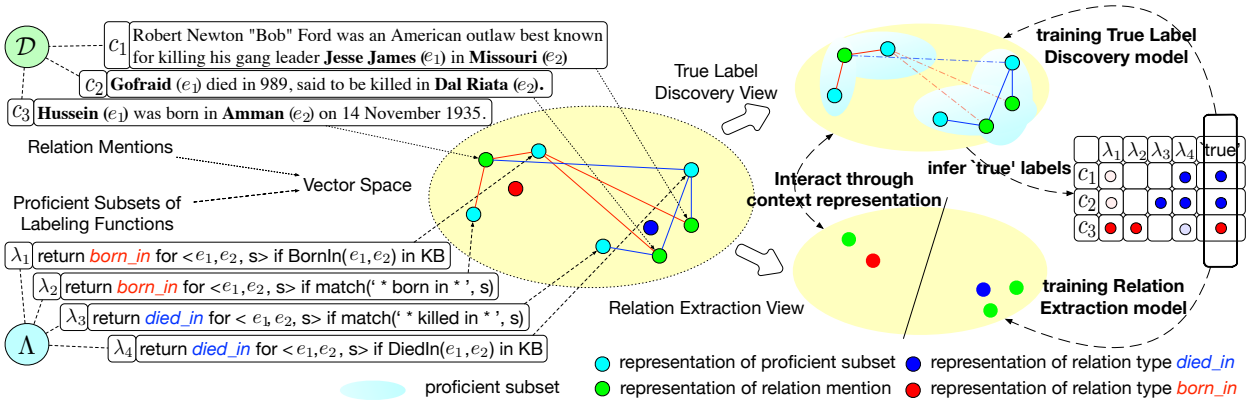


Figure 7.5: Illustration of relation extraction with heterogeneous supervision from both knowledge bases and domain rules.

employs a generative model, which encodes supervisions as labeling functions, and adopts the source consistency assumption: *a source is likely to provide true information with the same probability for all instances*. This assumption is widely used in true label discovery literature [143] to model reliabilities of information sources like crowd-sourcing and infer the true label from noisy labels. Accordingly, most true label discovery methods would trust a human annotator on all instances to the same level. However, labeling functions, unlike human annotators, do not make casual mistakes but follow certain "error routine". Thus, the reliability of a labeling function is not consistent among different pieces of instances. In particular, a labeling function could be more reliable for a certain subset (also known as its *proficient subset*) comparing to the rest. We identify these proficient subsets based on context information, only trust labeling functions on these subsets and avoid assuming global source consistency.

In heterogeneous supervision framework, we capture context's semantic meaning through representation learning, and conduct both relation extraction and true label discovery in a context-aware manner. Specifically, as depicted in Figure 7.5, we embed relation mentions in a low-dimension vector space, where similar relation mentions tend to have similar relation types and annotations. 'True' labels are further inferred based on reliabilities of labeling functions, which are calculated with their proficient subsets' representations. Then, these inferred true labels would serve as supervision for all components, including context representation, true label discovery and relation extraction. Besides, the context representation bridges relation extraction with true label discovery, and allows them to enhance each other. Such representations bridges all components with mutual enhancement in an iterative fashion. The resulting model achieves the state-of-the-art performance on two relation extraction benchmark datasets. We demonstrate that: (1) with additional domain-specific patterns we

can further improve the model performance in the presence of distant supervision; and (2) our proposed method can robust unify the two sources of supervision by resolving label conflicts – providing a principled approach to build on top of our effort-light StructMine to achieve even better results given rules from domain experts. This work has been accepted by the EMNLP 2017 conference as oral presentation [141].

7.4 CONCLUSION

The contributions of this thesis work are in the area of text mining and information extraction, within which we focus on domain-independent and noise-robust approaches using distant supervision (in conjunction with publicly-available knowledge bases). The work has broad impact on a variety of applications: knowledge base construction, question-answering systems, structured search and exploration of text data, recommender systems, network analysis, and many other text mining tasks. Finally, our work has been used in the following settings:

- **Introduced in classes and conference tutorials**: Our methods on entity recognition and typing (ClusType), fine-grained entity typing (PLE [11], AFET [12]), and relation extraction (CoType [13]) are being taught in graduate courses, *e.g.*, University of Illinois at Urbana-Champaign (CS 512), and are introduced as major parts of the conference tutorial in top data mining and database conferences such as SIGKDD, WWW, CIKM and SIGMOD.
- **Real-world, cross-disciplinary use cases**:
  - Our entity recognition and typing technique (ClusType [14]) has been transferred to U.S. Army Research Lab, Microsoft Bing Ads and NIH Big Data to Knowledge Center to identify typed entities of different kinds from low-resource, domain-specific text corpora. ClusType is also used by Stanford sociologists to identify scientific concepts from 37 millions of scientific publications in Web of Science database to study innovation and translation of scientific ideas.
  - A biomedical knowledge graph (*i.e.*, Life-iNet [15]) constructed automatically from millions of PubMed publications using our effort-light StructMine pipeline is used by researchers at Stanford Medical school to facilitate drug re-purposing. It yields significant improvement of performance on new drugs and rare diseases.
  - Our effort-light StructMine techniques (ClusType, PLE, CoType) is adopted by veterinarians at Veterinary Information Network Inc. (VIN) to construct the

first veterinary knowledge graph from multiple sources of information including research articles, books, guidelines, drug handbooks and message board posts.

- **Awards**: The thesis work on effort-light StructMine has been awarded a Google PhD fellowship in 2016 (sole winner in the category of Structured Data and Data Management in the world) and a Yahoo!-DAIS Research Excellence Award, and a C. W. Gear Outstanding Graduate Student Award from University of Illinois.

CHAPTER 8: VISION AND FUTURE WORK

This chapter discusses several potential directions for future work. Along this line of research, there are three exciting directions that could be pursued: (1) exploring more ways to reduce human annotation efforts other than leverage distant supervision from KBs; (2) extract implicit language patterns from massive, unlabeled corpora to facilitate supervised models; and (3) enrich the factual structures currently defined for the corpus-specif StructNet to enable more use cases.

8.1 INDIRECT SUPERVISION: LEVERAGING KNOWLEDGE FROM AUXILIARY TASKS

Relation extraction is an important task for understanding massive text corpora by turning unstructured text data into relation triples for further analysis. To alleviate the exhaustive human labeling process for generating training data, many recent efforts have been put to develop relation extraction (RE) models with training data automatically obtained by distant supervision (DS). DS replaces the manual training data generation with a pipeline that automatically links texts to a knowledge base (KB). However, the noise introduced to the automatically generated training data is not negligible. There are two major causes of error: incomplete KB and context-agnostic labeling process. If we treat unlinkable entity pairs as the pool of negative examples, false negatives can be commonly encountered as a result of the insufficiency of facts in KBs, where many true entity or relation mentions fail to be linked to KBs. On the other hand, context-agnostic labeling can engender false positive examples, due to the inaccuracy of the DS assumption that if a sentence contains any two entities holding a relation in the KB, the sentence must be expressing such relation between them.

Towards the goal of diminishing the negative effects by noisy DS training data, distantly supervised RE models that deal with training noise, as well as methods that directly improve the automatic training data generation process have been proposed. These methods mostly involve designing distinct assumptions to remove redundant training information. They do not have external trustworthy sources as guidance to uncover incorrectly labeled data. Without other separate information sources, the reliability of false label identification can be limited. Moreover, these noise reduction systems usually only address one type of error, either false positives or false negatives, although both types of error are observed to be significant. With the aim to overcome the above two issues derived from relation ex-

traction with distant supervision, we study the problem of relation extraction with indirect supervision from external sources. And due to the rapid emergence of QA systems as well as datasets of various QA tasks, we are motivated to leverage QA, a downstream application of relation extraction, to provide additional signals for learning RE models. In our recent work, ReQuest, the problem we try to solve is: given a domain-specific corpus and a set of target relation types from a KB, we aim to detect relation mentions from text and categorize each in context by target types or Non-TargetType (None) by leveraging an independent dataset of QA pairs in the same domain.

We can design a framework to address the problem and alleviate the two issues in existing models with the following key ideas: (1) We integrate indirect supervision from another same-domain data source in the format of QA sentence pairs, that is, each question sentence maps to several positive (where a true answer can be found) and negative (where no answer exists) answer sentences. We adopt the principle that for the same question, positive pairs of (question, answer) should be semantically similar while they should be dissimilar from negative pairs. (2) Instead of differentiating types of labeling errors at the instance level, we concentrate on how to better learn semantic representation of features. Wrongly labeled training examples essentially misguide the understanding of features. It increases the risk of having a non-representative feature learned to be close to a relation type and vice versa. Therefore, if the feature learning process is improved, potentially both types of error can be reduced.

8.2 PATTERN-ENHANCED EMBEDDING LEARNING FOR RELATION EXTRACTION

Weakly-supervised relation extraction is an important task in data mining and natural language processing. Given a text corpus and a target relation specified by a set of seed entity pairs, the task aims at extracting more entity pairs under the target relation from the corpus. The extracted entity pairs can benefit various downstream applications, such as knowledge base completion and corpus-level relation extraction.

Existing approaches to weakly-supervised relation extraction can be divided into two kinds, i.e., the distributional methods and the pattern-based methods. Given a pair of entities, the distributional methods infer their relations based on the corpus-level statistics of both entities. Specifically, these methods try to learn low-dimensional representations of entities to preserve such statistics, so that entities with similar semantic meanings tend to have similar representations. Then a relation classifier can be learned using the given seed entity pairs, which takes entity representations as features for relation prediction. Differ-

ently, the pattern-based methods predict the relation of two entities from several sentences mentioning both of them. Towards this goal, traditional approaches try to extract discriminative textual patterns from such sentences, while recent approaches leverage deep neural networks for prediction. However, all these methods rely on a large number of seed entity pairs for training, and thus their performance is usually poor in our setting, as the seed entity pairs are very limited.

To tackle this problem, we can follow recent studies on entity typing (ClusType) and relation extraction (CoType), which show that integrating the global statistics and local contexts can improve the performance of prediction. Inspired by the idea, in this work we propose a co-training framework to integrate the distributional methods and the pattern-based methods, so that they can mutually provide extra supervision to overcome the scarcity problem of the seed entity pairs. Specifically, the pattern module acts as a generator, as it can extract some candidate entity pairs based on the discovered patterns; whereas the distributional module is treated as a discriminator to evaluate the quality of each generated entity pair, that is, whether a pair has the target relation. To encourage the collaboration of both modules, we formulate a joint optimization process, in which we iterate between two sub-processes. In the first sub-process, the discriminator (distributional methods) will evaluate the entity pairs generated by the generator (pattern-based methods) and the results serve as extra signals to adjust the generator. In the second sub-process, the generator (pattern-based methods) will in turn generate a set of highly confident entity pairs, which serve as extra training seeds to improve the discriminator (distributional methods). During training, we keep iterating between the two sub-processes, so that both methods can be consistently improved. Once the training converges, both methods can be applied to relation extraction, which extract new entity pairs from different perspectives.

8.3 EXTRACTING IMPLICIT PATTERNS FROM MASSIVE UNLABELED CORPORA

As neural language models can be trained without human annotations but generate texts of a high quality, we further explore the possibility to extract the abundant and self-contained knowledge in natural language. So far, we've employed two strategies to incorporate such models with sequence labeling, a general framework in natural language processing which encompassing various of applications (e.g., Named Entity Recognition, POS Tagging, Event Detection). The first is to treat such information as additional supervision, and guide the training of the target task by the knowledge transfer.

Specifically, we leave the word-level knowledge to pre-trained word embedding and co-

train a character-level language model with the end task. The proposed method can conduct efficient training and inference, which has been accepted and presented at the AAAI 2017 conference. Alternatively, we further explore the potential of the extensive raw corpora. We pre-train very large language models on such corpora to capture abundant linguistic features. Moreover, we design a novel model pruning method to allow us conduct model compression for better inference efficiency. The resulting model can be incorporated in a plug-in-and-play manner and greatly improve the performance without much loss of efficiency. This work has been submitted to a reputed venue for the review.

8.4 ENRICHING FACTUAL STRUCTURE REPRESENTATION

In the current definition of StructNet, edges between two entities are weighted by the frequency of the facts mentioned in the text corpus. Such a representation has several limitations: (1) raw frequency cannot indicate *uncertainty* of the fact (*e.g.*, drug A treats drug B with 75% success rate), (2) conditions of a relation are ignored in the modeling (*e.g.*, if the patient is under 50 years old), and (3) complex relations involve more than two entities (*e.g.*, protein `localization` relation). To address these challenges, I am interested in **collaborating with NLP researchers and linguists** to work on domain-independent sentiment analysis and syntax parsing for large text corpora, and incorporate the sophisticated linguistic features in StructNet construction. In particular, to measure fact uncertainty, it is critical to mine from a sentence words/phrases that indicate uncertainty (*e.g.*, "*unlikely*", "*probably*", "*with 50% chance*"), negation (*e.g.*, "*no*", "*barely*"), sentiments (*e.g.*, "*efficiently*", "*nicely*"), or their enhancers (*e.g.*, "very", "*extremely*"), and design systematic measures to quantify these units into weights of the edges in StructNets. To mine conditions for relations, I aim to extend the meta pattern-based attribute mining algorithm to identify patterns for "condition descriptions" (*e.g.*, "*...[with age _]...*") and attach the mined conditions to edges in StructNet for further analysis. To extract complex relations, I plan to design scalable candidate generation process (*e.g.*, different pruning strategy) to avoid producing exponential number of candidate relations, and extend the CoType embedding approach to model types for n-ary relations, while preserving the mutual constraints between relations and their entity arguments.

REFERENCES

[1] X. L. Dong, T. Strohmann, S. Sun, and W. Zhang, "Knowledge vault: A web-scale approach to probabilistic knowledge fusion," in *SIGKDD*, 2014.

[2] A. Carlson, J. Betteridge, B. Kisiel, B. Settles, E. R. Hruschka Jr, and T. M. Mitchell, "Toward an architecture for never-ending language learning." in *AAAI*, 2010.

[3] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates, "Web-scale information extraction in knowitall:(preliminary results)," in *WWW*, 2004.

[4] J. Shin, S. Wu, F. Wang, C. De Sa, C. Zhang, and C. Ré, "Incremental knowledge base construction using deepdive," *Proceedings of the VLDB Endowment*, vol. 8, no. 11, pp. 1310–1321, 2015.

[5] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open information extraction from the web," in *IJCAI*, 2007.

[6] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *SIGMOD*, 2008.

[7] N. F. Noy, N. H. Shah, P. L. Whetzel, B. Dai, M. Dorf, N. Griffith, C. Jonquet, D. L. Rubin, M.-A. Storey, C. G. Chute, and M. a. Musen, "Bioportal: ontologies and integrated data resources at the click of a mouse," *Nucleic acids research*, vol. 37, pp. W170–3, 2009.

[8] X. Ren, A. El-Kishky, C. Wang, F. Tao, C. R. Voss, and J. Han, "ClusType: Effective entity recognition and typing by relation phrase-based clustering," in *Proceedings of the 21th ACM SIGKDD Intl. Conference on Knowledge Discovery and Data Mining (KDD)*, 2015.

[9] X. Ren, A. El-Kishky, C. Wang, and J. Han, "Automatic entity recognition and typing from massive text corpora: A phrase and network mining approach (conference tutorial)," in *Proceedings of the 21th ACM SIGKDD Intl. Conference on Knowledge Discovery and Data Mining (KDD)*, 2015.

[10] X. Ren, A. El-Kishky, H. Ji, and J. Han, "Automatic entity recognition and typing in massive text data (conference tutorial)," in *Proceedings of the 2016 Intl. Conf. on Management of Data (SIGMOD)*, 2016.

[11] X. Ren, W. He, M. Qu, H. Ji, C. R. Voss, and J. Han, "Label noise reduction in entity typing by heterogeneous partial-label embedding," in *Proceedings of the 22nd ACM SIGKDD Intl. Conference on Knowledge Discovery and Data Mining (KDD)*, 2016.

[12] X. Ren, W. He, M. Qu, H. Ji, and J. Han, "AFET: Automatic fine-grained entity typing by hierarchical partial-label embedding," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

[13] X. Ren, Z. Wu, W. He, M. Qu, C. R. Voss, H. Ji, T. F. Abdelzaher, and J. Han, "CoType: Joint extraction of typed entities and relations with knowledge bases," in *Proceedings of the 27th Intl. Conference on World Wide Web (WWW)*, 2017.

[14] X. Ren, J. Liu, X. Yu, U. Khandelwal, Q. Gu, L. Wang, and J. Han, "ClusCite: effective citation recommendation by information network-based clustering," in *Proceedings of the 20th ACM SIGKDD Intl. Conference on Knowledge Discovery and Data Mining (KDD)*, 2014.

[15] X. Ren, J. Shen, M. Qu, X. Wang, Z. Wu, Q. Zhu, M. Jiang, F. Tao, S. Sinha, D. Liem et al., "Life-inet: A structured network-based knowledge exploration and analytics system for life sciences," *Proceedings of ACL 2017, System Demonstrations*, 2017.

[16] G. R. Doddington, A. Mitchell, M. A. Przybocki, L. A. Ramshaw, S. Strassel, and R. M. Weischedel, "The automatic content extraction (ace) program-tasks, data, and evaluation." in *LREC*, vol. 2, 2004, p. 1.

[17] D. Vrandečić and M. Krötzsch, "Wikidata: a free collaborative knowledgebase," *Communications of the ACM*, vol. 57, no. 10, pp. 78–85, 2014.

[18] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 697–706.

[19] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "Dbpedia: A nucleus for a web of open data," in *The semantic web*. Springer, 2007, pp. 722–735.

[20] D. B. Lenat, "Cyc: A large-scale investment in knowledge infrastructure," *Communications of the ACM*, vol. 38, no. 11, pp. 33–38, 1995.

[21] E. Agichtein and L. Gravano, "Snowball: Extracting relations from large plain-text collections," in *Proceedings of the fifth ACM conference on Digital libraries*, 2000.

[22] S. Brin, "Extracting patterns and relations from the world wide web," in *International Workshop on The World Wide Web and Databases*, 1998.

[23] D. E. Appelt, J. R. Hobbs, J. Bear, D. Israel, and M. Tyson, "Fastus: A finite-state processor for information extraction from real-world text," in *IJCAI*, 1992.

[24] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *Proceedings of the 14th conference on Computational linguistics (ACL)*, 1992.

[25] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *ACL*, 2005.

[26] L. Ratinov and D. Roth, "Design challenges and misconceptions in named entity recognition," in *ACL*, 2009.

[27] R. C. Bunescu and R. J. Mooney, "A shortest path dependency kernel for relation extraction," in *HLT-EMNLP*, 2005.

[28] R. J. Mooney and R. C. Bunescu, "Subsequence kernels for relation extraction," in *NIPS*, 2005.

[29] O. Etzioni, M. Cafarella, D. Downey, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates, "Unsupervised named-entity extraction from the web: An experimental study," *Artificial Intelligence*, vol. 165, no. 1, pp. 91–134, 2005.

[30] Y. He and D. Xin, "Seisa: set expansion by iterative similarity aggregation," in *WWW*, 2011.

[31] S. Gupta and C. D. Manning, "Improved pattern learning for bootstrapped entity extraction," in *CONLL*, 2014.

[32] S. Sarawagi and W. W. Cohen, "Semi-markov conditional random fields for information extraction," in *NIPS*, 2004.

[33] A. Carlson, J. Betteridge, R. C. Wang, E. R. Hruschka Jr, and T. M. Mitchell, "Coupled semi-supervised learning for information extraction," in *WSDM*, 2010.

[34] N. Nakashole, T. Tylenda, and G. Weikum, "Fine-grained semantic typing of emerging entities." in *ACL*, 2013.

[35] T. Lin, O. Etzioni et al., "No noun phrase left behind: detecting and typing unlinkable entities," in *EMNLP*, 2012.

[36] X. Ling and D. S. Weld, "Fine-grained entity recognition," in *AAAI*, 2012.

[37] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, "Distant supervision for relation extraction without labeled data," in *ACL*, 2009.

[38] M. Surdeanu, J. Tibshirani, R. Nallapati, and C. D. Manning, "Multi-instance multi-label learning for relation extraction," in *EMNLP*, 2012.

[39] S. Riedel, L. Yao, and A. McCallum, "Modeling relations and their mentions without labeled text," in *ECML*, 2010.

[40] L. Huang, J. May, X. Pan, H. Ji, X. Ren, J. Han, L. Zhao, and J. A. Hendler, "Liberal entity extraction: Rapid construction of fine-grained entity typing systems," *Big data*, vol. 5, no. 1, pp. 19–31, 2017.

[41] A. Fader, S. Soderland, and O. Etzioni, "Identifying relations for open information extraction," in *EMNLP*, 2011.

[42] M. Schmitz, R. Bart, S. Soderland, O. Etzioni et al., "Open language learning for information extraction," in *EMNLP*, 2012.

[43] G. Angeli, M. J. J. Premkumar, and C. D. Manning, "Leveraging linguistic structure for open domain information extraction," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, vol. 1, 2015, pp. 344–354.

[44] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.

[45] N. Bach and S. Badaskar, "A review of relation extraction," *Literature review for Language and Statistics II*.

[46] A. Culotta and J. Sorensen, "Dependency tree kernels for relation extraction," in *ACL*, 2004.

[47] Z. GuoDong, S. Jian, Z. Jie, and Z. Min, "Exploring various knowledge in relation extraction," in *ACL*, 2005.

[48] Q. Li and H. Ji, "Incremental joint extraction of entity mentions and relations," in *ACL*, 2014.

[49] M. Miwa and Y. Sasaki, "Modeling joint entity and relation extraction with table representation," in *EMNLP*, 2014.

[50] D. Roth and W.-t. Yih, "Global inference for entity and relation identification via a linear programming formulation," *Introduction to statistical relational learning*, pp. 553–580, 2007.

[51] K. Nigam and R. Ghani, "Analyzing the effectiveness and applicability of co-training," in *CIKM*, 2000.

[52] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *COLT Workshop on Computational Learning Theory*, 1998.

[53] S. Shi, H. Zhang, X. Yuan, and J.-R. Wen, "Corpus-based semantic class mining: distributional vs. pattern-based approaches," in *COLING*, 2010.

[54] P. P. Talukdar and F. Pereira, "Experiments in graph-based semi-supervised learning methods for class-instance acquisition," in *ACL*, 2010.

[55] S. Takamatsu, I. Sato, and H. Nakagawa, "Reducing wrong labels in distant supervision for relation extraction," in *ACL*, 2012.

[56] X. Ren, W. He, M. Qu, C. R. Voss, H. Ji, and J. Han, "Label noise reduction in entity typing by heterogeneous partial-label embedding," in *KDD*, 2016.

[57] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *NIPS*, 2013.

[58] D. Yogatama, D. Gillick, and N. Lazic, "Embedding methods for fine grained entity type classification," in *ACL*, 2015.

[59] B. Salehi, P. Cook, and T. Baldwin, "A word embedding approach to predicting the compositionality of multiword expressions," in *NAACL-HLT*, 2015.

[60] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *WWW*, 2015.

[61] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *KDD*, 2014.

[62] M. R. Gormley, M. Yu, and M. Dredze, "Improved relation extraction with feature-rich compositional embedding models," *EMNLP*, 2015.

[63] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *NIPS*, 2013.

[64] K. Toutanova, D. Chen, P. Pantel, P. Choudhury, and M. Gamon, "Representing text for joint embedding of text and knowledge bases," in *EMNLP*, 2015.

[65] T. Cour, B. Sapp, and B. Taskar, "Learning from partial labels," *JMLR*, vol. 12, pp. 1501–1536, 2011.

[66] N. Nguyen and R. Caruana, "Classification with partial labels," in *KDD*, 2008.

[67] L. Galárraga, G. Heitz, K. Murphy, and F. M. Suchanek, "Canonicalizing open knowledge bases," in *CIKM*, 2014.

[68] R. C. Bunescu and R. Mooney, "Learning to extract relations from the web using minimal supervision," in *ACL*, 2007.

[69] A. Nagesh, G. Haffari, and G. Ramakrishnan, "Noisy or-based model for relation extraction using distant supervision," in *EMNLP*, 2014.

[70] W. Xu, R. Hoffmann, L. Zhao, and R. Grishman, "Filling knowledge base gaps for distant supervision of relation extraction," in *ACL*, 2013.

[71] R. Hoffmann, C. Zhang, X. Ling, L. Zettlemoyer, and D. S. Weld, "Knowledge-based weak supervision for information extraction of overlapping relations," in *ACL*, 2011.

[72] I. Augenstein, A. Vlachos, and D. Maynard, "Extracting relations between non-standard entities using distant supervision and imitation learning," in *EMNLP*, 2015.

[73] S. Singh, S. Riedel, B. Martin, J. Zheng, and A. McCallum, "Joint inference of entities, relations, and coreference," in *Workshop on Automated Knowledge Base Construction*, 2013.

[74] W. Shen, J. Wang, and J. Han, "Entity linking with a knowledge base: Issues, techniques, and solutions," *TKDE*, no. 99, pp. 1–20, 2014.

[75] R. Huang and E. Riloff, "Inducing domain-specific semantic class taggers from (almost) nothing," in *ACL*, 2010.

[76] Z. Kozareva and E. Hovy, "Not all seeds are equal: Measuring the quality of text mining seeds," in *NAACL*, 2010.

[77] W. Shen, J. Wang, P. Luo, and M. Wang, "A graph-based approach for ontology population with named entities," in *CIKM*, 2012.

[78] A. El-Kishky, Y. Song, C. Wang, C. R. Voss, and J. Han, "Scalable topical phrase mining from text corpora," *VLDB*, 2015.

[79] X. He and P. Niyogi, "Locality preserving projections," in *NIPS*, 2004.

[80] B. Min, S. Shi, R. Grishman, and C.-Y. Lin, "Ensemble semantics for large-scale unsupervised relation extraction," in *EMNLP*, 2012.

[81] J. Liu, C. Wang, J. Gao, and J. Han, "Multi-view clustering via joint nonnegative matrix factorization," in *Proceedings of the 2013 SIAM International Conference on Data Mining (SDM)*, 2013.

[82] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of optimization theory and applications*, vol. 109, no. 3, pp. 475–494, 2001.

[83] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *HLT-NAACL*, 2003.

[84] C. Whitelaw, A. Kehlenbeck, N. Petrovic, and L. Ungar, "Web-scale named entity recognition," in *CIKM*, 2008.

[85] F. Jiao, S. Wang, C.-H. Lee, R. Greiner, and D. Schuurmans, "Semi-supervised conditional random fields for improved sequence segmentation and labeling," in *ACL*, 2006.

[86] W. Lin, R. Yangarber, and R. Grishman, "Bootstrapped learning of semantic classes from positive and negative examples," in *ICML Workshop on The Continuum from Labeled to Unlabeled Data*, 2003.

[87] Z. Kozareva, K. Voevodski, and S.-H. Teng, "Class label enhancement via related instances," in *EMNLP*, 2011.

[88] D. S. Kim, K. Verma, and P. Z. Yeh, "Joint extraction and labeling via graph propagation for dictionary construction." in *AAAI*, 2013.

[89] B. B. Dalvi, W. W. Cohen, and J. Callan, "Websets: Extracting sets of entities from the web using unsupervised information extraction," in *WSDM*, 2012.

[90] V. Ganti, A. C. König, and R. Vernica, "Entity categorization over large document collections," in *SIGKDD*, 2008.

[91] C. Li, J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee, "Twiner: named entity recognition in targeted twitter stream," in *SIGIR*, 2012.

[92] H. Lin, Y. Jia, Y. Wang, X. Jin, X. Li, and X. Cheng, "Populating knowledge base with collective entity mentions: A graph-based approach," in *ASONAM*, 2014.

[93] F. Sha and F. Pereira, "Shallow parsing with conditional random fields," in *HLT-NAACL*, 2003.

[94] S. Sarawagi, "Information extraction," *Foundations and trends in databases*, vol. 1, no. 3, pp. 261–377, 2008.

[95] A. Yates and O. Etzioni, "Unsupervised methods for determining object and relation synonyms on the web," *Journal of Artificial Intelligence Research*, vol. 34, no. 1, p. 255, 2009.

[96] M. Ji, Y. Sun, M. Danilevsky, J. Han, and J. Gao, "Graph regularized transductive classification on heterogeneous information networks," in *ECMLPKDD*, 2010.

[97] H. Ji and R. Grishman, "Refining event extraction through cross-document inference." in *ACL*, 2008.

[98] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: a heterogeneous information network approach," in *Proceedings of the 7th ACM Intl. Conference on Web Search and Data Mining (WSDM)*, 2014.

[99] M. A. Yosef, S. Bauer, J. Hoffart, M. Spaniol, and G. Weikum, "Hyena: Hierarchical type classification for entity names," in *COLING*, 2012.

[100] X. Ren, A. El-Kishky, C. Wang, and J. Han, "Automatic entity recognition and typing in massive text corpora," in *WWW*, 2016.

[101] D. Gillick, N. Lazic, K. Ganchev, J. Kirchner, and D. Huynh, "Context-dependent fine-grained entity type tagging," *arXiv preprint arXiv:1412.1820*, 2014.

[102] J. Dunietz and D. Gillick, "A new entity salience task with millions of training examples," *EACL*, 2014.

[103] S. Singh, A. Subramanya, F. Pereira, and A. McCallum, "Wikilinks: A large-scale cross-document coreference corpus labeled via links to wikipedia," *UM-CS-2012-015*, 2012.

[104] E. Gabrilovich, M. Ringgaard, and A. Subramanya, "Facc1: Freebase annotation of clueweb corpora," 2013.

[105] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The stanford corenlp natural language processing toolkit," *ACL*, 2014.

[106] J. Weston, S. Bengio, and N. Usunier, "Wsabie: Scaling up to large vocabulary image annotation," in *IJCAI*, 2011.

[107] J.-Y. Jiang, C.-Y. Lin, and P.-J. Cheng, "Entity-driven type hierarchy construction for freebase," in *WWW*, 2015.

[108] R. Weischedel, E. Hovy, M. Marcus, M. Palmer, R. Belvin, S. Pradhan, L. Ramshaw, and N. Xue, "Ontonotes: A large training corpus for enhanced processing," 2011.

[109] R. Weischedel and A. Brunstein, "Bbn pronoun coreference and entity type corpus," *Linguistic Data Consortium*, vol. 112, 2005.

[110] L. Dong, F. Wei, H. Sun, M. Zhou, and K. Xu, "A hybrid neural model for type classification of entity mentions," in *IJCAI*, 2015.

[111] J. Tang, M. Qu, and Q. Mei, "Pte: Predictive text embedding through large-scale heterogeneous text networks," in *KDD*, 2015.

[112] L. Del Corro, A. Abujabal, R. Gemulla, and G. Weikum, "Finet: Context-aware fine-grained named entity typing," in *EMNLP*, 2015.

[113] M.-L. Zhang, "Disambiguation-free partial label learning," in *SDM*, 2014.

[114] L. Liu and T. G. Dietterich, "A conditional multinomial mixture model for superset label learning," in *NIPS*, 2012.

[115] M.-L. Zhang and F. Yu, "Solving the partial label learning problem: An instance-based approach," in *IJCAI*, 2015.

[116] Z. Hu, P. Huang, Y. Deng, Y. Gao, and E. P. Xing, "Entity hierarchy embedding," in *ACL*, 2015.

[117] R. West, E. Gabrilovich, K. Murphy, S. Sun, R. Gupta, and D. Lin, "Knowledge base completion via search-based question answering," in *WWW*, 2014.

[118] H. Sun, H. Ma, W.-t. Yih, C.-T. Tsai, J. Liu, and M.-W. Chang, "Open domain question answering via semantic enrichment," in *WWW*, 2015.

[119] J. Bian, Y. Liu, E. Agichtein, and H. Zha, "Finding the right facts in the crowd: factoid question answering over social media," in *WWW*, 2008.

[120] J. Ellis, J. Getman, J. Mott, X. Li, K. Griffitt, S. M. Strassel, and J. Wright, "Linguistic resources for 2013 knowledge base population evaluations," *Text Analysis Conference (TAC)*, 2014.

[121] S. Pyysalo, F. Ginter, J. Heimonen, J. Björne, J. Boberg, J. Järvinen, and T. Salakoski, "Bioinfer: a corpus for information extraction in the biomedical domain," *BMC bioinformatics*, vol. 8, no. 1, p. 1, 2007.

[122] D. Hovy, B. Plank, H. M. Alonso, and A. Søgaard, "Mining for unambiguous instances to adapt part-of-speech taggers to new domains," in *NAACL*, 2015.

[123] C. Lee, Y.-G. Hwang, and M.-G. Jang, "Fine-grained named entity recognition and relation extraction for question answering," in *SIGIR*, 2007.

[124] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer, "Dbpedia spotlight: shedding light on the web of documents," in *I-Semantics*, 2011.

[125] J. Hoffart, M. A. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum, "Robust disambiguation of named entities in text," in *EMNLP*, 2011.

[126] J. Liu, J. Shang, C. Wang, X. Ren, and J. Han, "Mining quality phrases from massive text corpora," in *Proceedings of the 2015 ACM SIGMOD Intl. Conf. on Management of Data (SIGMOD)*, 2015.

[127] A. El-Kishky, Y. Song, C. Wang, C. R. Voss, and J. Han, "Scalable topical phrase mining from text corpora," *VLDB*, vol. 8, no. 3, pp. 305–316, 2014.

[128] Y. S. Chan and D. Roth, "Exploiting background knowledge for relation extraction," in *COLING*, 2010.

[129] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal estimated sub-gradient solver for svm," *Mathematical programming*, vol. 127, no. 1, pp. 3–30, 2011.

[130] F. Tao, H. Zhuang, C. W. Yu, Q. Wang, T. Cassidy, L. Kaplan, C. Voss, and J. Han, "Multi-dimensional, phrase-based summarization in text cubes," *Data Engineering*, p. 74, 2016.

[131] T. Siddiqui, X. Ren, A. Parameswaran, and J. Han, "FacetGist: Collective extraction of document facets in large technical corpora," in *Proceedings of the 25th ACM Intl. Conference on Information and Knowledge Management (CIKM)*, 2016.

[132] G. Jeh and J. Widom, "Scaling personalized web search," in *WWW*, 2003.

[133] T. H. Haveliwala, "Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search," *TKDE*, vol. 15, no. 4, pp. 784–796, 2003.

[134] I. Mani and E. Bloedorn, "Multi-document summarization by graph search and matching," *AAAI*, 1997.

[135] C. Zhai, A. Velivelli, and B. Yu, "A cross-collection mixture model for comparative text mining," in *SIGKDD*, 2004.

[136] X. Huang, X. Wan, and J. Xiao, "Comparative news summarization using concept-based optimization," *Knowledge and information systems*, vol. 38, no. 3, pp. 691–716, 2014.

[137] D. Wang, S. Zhu, T. Li, and Y. Gong, "Comparative document summarization via discriminative sentence selection," *TKDD*, vol. 6, no. 3, p. 12, 2013.

[138] K. Lerman and R. McDonald, "Contrastive summarization: an experiment with consumer reviews," in *NAACL*, 2009.

[139] L. Zhang, L. Li, C. Shen, and T. Li, "Patentcom: A comparative view of patent document retrieval," *SDM*, 2015.

[140] R. Gupta, A. Halevy, X. Wang, S. E. Whang, and F. Wu, "Biperpedia: An ontology for search applications," *Proceedings of the VLDB Endowment*, vol. 7, no. 7, pp. 505–516, 2014.

[141] L. Liu, X. Ren, Q. Zhu, S. Zhi, H. Gui, H. Ji, and J. Han, "Heterogeneous supervision for relation extraction: A representation learning approach," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

[142] A. J. Ratner, C. M. De Sa, S. Wu, D. Selsam, and C. Ré, "Data programming: Creating large training sets, quickly," in *Advances in Neural Information Processing Systems*, 2016, pp. 3567–3575.

[143] Y. Li, J. Gao, C. Meng, Q. Li, L. Su, B. Zhao, W. Fan, and J. Han, "A survey on truth discovery," *SIGKDD Explor. Newsl.*, vol. 17, no. 2, pp. 1–16, Feb. 2016. [Online]. Available: http://doi.acm.org/10.1145/2897350.2897352