UNSUPERVISED GRAPH-BASED RELATION EXTRACTION AND VALIDATION FOR KNOWLEDGE BASE POPULATION

By

Dian Yu

A Dissertation Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

DOCTOR OF PHILOSOPHY

Major Subject: COMPUTER SCIENCE

Examining Committee:

_____

Dr. Heng Ji, Dissertation Adviser

_____

Dr. Deborah L. McGuinness, Member

_____

Dr. Peter Fox, Member

_____

Dr. Ralph Grishman, Member

Rensselaer Polytechnic Institute
Troy, New York

October 2017
(For Graduation December 2017)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Knowledge bases (KBs), which store millions of facts about the world, have been widely applied to a broad range of applications such as semantic search and question answering. Each relational fact contains two entities (*e.g.*, person and location) and the relation between them. However, existing KBs are far from complete. Manually updated Wikipedia Infoboxes still serve as the important structured input for many large-scale KBs. Furthermore, completing KBs by inferring missing relations from existing structured data cannot completely solve this problem since KBs mainly focus on famous entities.

To populate KBs, researchers have made significant progress in relation extraction from unstructured text corpora. However, it remains very challenging since a relation can be expressed in numerous ways through a sophisticated long-range linguistic structure. Previous successful methods require sufficient clean training data, external knowledge bases, or high-quality patterns, which result in extensive human involvement and poor portability to a new relation type or a different language. The consolidation of relations extracted by multiple relation extraction systems from multiple information sources may also generate erroneous, conflicting, redundant or complement results, which are caused by the differences in source trustability and the significant differences in performance among multiple systems. In many cases, certain facts can only be discovered by a minority of advanced systems from a few trustworthy sources. Therefore, it poses a challenge but also an opportunity for KB fact validation.

In this thesis, we aim to improve multilingual knowledge base population by designing unsupervised graph-based methods to extract and validate relations from unstructured textual data. We develop language/relation independent methods which can be adapted to a new language/relation with less effort. We want to further improve the performance of a single relation extraction system by incorporating evidence from multiple information sources and multiple systems. To evaluate the effectiveness of our approach, we choose Slot Filling as our evaluation platform, which aims to extract the values of a variety of predefined attributes for a given entity from a large-scale corpus and provide justification sentences to support these values.

CHAPTER 1
Introduction

1.1 Challenges in Knowledge Base Population

Knowledge bases (KBs), such as Wikipedia Infoboxes, YAGO [2], Freebase [3], DBpedia [4], and Google's Knowledge Graph store millions of facts about the world. Each fact contains two entities (*e.g.*, person and location) and the relations between entities. They are typically represented as triples (head entity, relation type, tail entity) (*e.g.*, (*Rensselaer Polytechnic Institute, Established, 1824*)).

Recently almost all major search engines have adopted KBs for semantic search to answer some questions from users by leveraging the rich structured data from KB [5]. For example, when we type *"when was Rensselaer Polytechnic Institute founded?"*, advanced search engines such as Google return a structured response *"Rensselaer Polytechnic Institute, founded, 1824"* based on existing KBs. Intelligent assistants such as Siri, Alexa, and Cortana also exploit structured KBs to respond to users. Besides question answering, KBs are also useful resources for other Natural Language Processing applications such as document summarization, automated translation, and information retrieval. Compared to unstructured texts, structured information is more *accessible* for further processing [6].

However, we still face three major challenges related to KB population. First, ***manually*** updated Wikipedia Infoboxes still serve as important input for many existing large-scale KBs [2]–[4], [7]. It is time-consuming and labor-intensive to manually create and maintain KBs. For example, the Wikipedia entry "*Rensselaer Polytechnic Institute*" has been revised more than $2,000$ times by 744 volunteer editors since March, 2003. In addition, we can hardly have access to up-to-date KBs because of the delay in updating KBs when a change occurs.

Large-scale KBs contain billions of relation triples. The latest version of Freebase contains 1.9 billion triples. However, they still suffer from ***incompleteness*** [8] which would negatively affect their usefulness in downstream applications. One importance research direction is to automatically infer missing triples by making inferences from existing ones [9]. For instance, given (A, *spouse*, B) and (C, *sibling*, B), we can infer

the missing triple (C, *otherFamily*, A). Unfortunately, KBs are still far from complete even if we could fill in all the missing triples in KBs correctly since existing KBs place more emphasis on famous entities. A majority of the involved entities in unstructured data, which accounts for more than $70\% - 80\%$ of all data in the world [10], can not be linked to existing KB entries. Therefore, we turn to the other direction to populate KBs from unstructured corpora automatically.

Besides, most KBs recently constructed are in English [11]. Existing KBs in other languages (even majority languages such as Chinese and Spanish) are insufficient (Figure 1.1). Considering linguistic differences such as word segmentation and grammatical structure [12], most of the English KB population techniques cannot be easily applied to build KBs in other languages from different families.

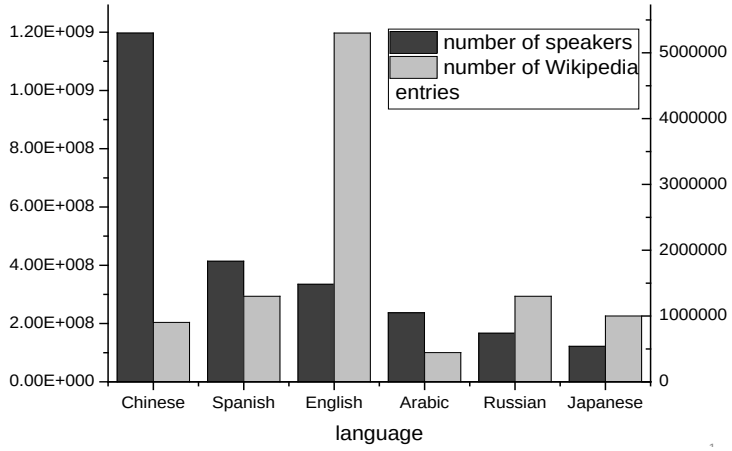

Figure 1.1: Size of Wikipedia entries for different popular languages.

KB population raises the third important issue: it requires relation extraction systems to deal with redundant, complementary or conflicting answers extracted from multiple information sources based on multiple extraction algorithms [13]. For example, if an author claims "*RPI was founded in **1823** by Stephen Van Rensselaer*" in his book and then *"1823"* is extracted as a candidate answer in conflict with the existing answer *"1824"*. How can we identify those trustworthy claims automatically?

Considering these challenges, in this thesis, we focus on language-independent relation extraction and validation from large-scale unstructured textual copora (*e.g.*, news, discussion forum, and web documents) to populate multilingual knowledge bases.

1.2 Motivations and Solutions

1.2.1 Data-Driven Relation Extraction

With the development of relation extraction techniques, we are requiring less and less human-labeled data. Rule-based methods [14]–[16] are based on hundreds of high-quality hand-crafted patterns. However, it is very expensive and laborious to manually create and maintain large sets of patterns. Instead, researchers developed supervised relation-specific classifiers from annotated relation instances or applied semi-supervised methods which require fewer labeled relation instances or patterns as seeds to do bootstrapping learning from a large corpus. The resulting patterns often suffer from low precision and semantic drift [17], [18].

Recent successful relation extraction methods applied distant supervision without the need of labeled data. Instead, they use large-scale knowledge bases which contain thousands of relation types. For each pair of entities in a certain relation category, they find all the sentences containing those entities in a large unlabeled corpus and extract syntactic and lexical features to train relation classifiers [19]–[21]. However, the resulting training data is extremely noisy since the co-occurence of two entities does not necessarily indicate the existence of a specific KB relation between them. For example, the following two sentences containing the same pair of entities actually describe two different relations.

"***Jennifer Aniston*** *and* ***Brad Pitt's*** *divorce totaled to* 60 *million, with her taking ownership of their Beverly Hills Home*"

"***Brad Pitt*** *and* ***Jennifer Aniston*** *are keeping up with their friendship.*"

All these data-driven methods attempt to imitate how human writers express a certain relation repeatedly, and train from redundant relation instances based on lexical/syntactic features. Given sufficient labeled data, data-driven methods can achieve promising performance for a given relation type, while suffering from poor portability to low-resource languages or unknown relation types.

1.2.2 Importance-Based Relation Extraction

Patterns has already played an important role in relation extraction since the 1990s. They are extremely useful in low-resource settings. Latest successful relation extraction systems still benefit from pattern matching [20]. Thus, we think we should turn back to a

relatively neglected topic: why patterns can effectively help us extract relations?

Among all the previous methods mentioned above, we think patterns are closest to the nature of relation expression by generalizing how people express a relation. Involved arguments and the relative words or phrases which indicate the relation between arguments can be regarded as essential elements of a relation. Patterns are effective since they accurately capture the essential elements of a relation and provide lexical/syntactic forms to express the connections among them. For example, in the following sentence:

"The other workers declined to touch the water bowl because ***Asia Bibi****, a* ***Christian****, had carried the container, according to her husband,* ***Ashiq Masih****, and other accounts."*

the relation type between *"Asia Bibi"* and *"Ashiq"* is clearly indicated by *"husband"*. Formally, we call such an indicative word as ***trigger***.

Based on the existence of triggers, we roughly divide relation types into two categories: ***trigger-driven*** and ***value-driven***. For value-driven types, there is relatively a shorter distance between two entities and there is usually no clear indicator. For trigger-driven relation types, a relation is indicated by a relation-specific word. In this thesis, we mainly focus on trigger-driven relation extraction.

For a trigger-driven relation type, an argument pair and its corresponding trigger are the essential elements of a relation. Existing patterns can extract relations accurately and effectively since they contain these key elements in their representations. Specifically, trigger-driven relation types usually contain relation-specific triggers.

However, methods based on lexical patterns suffer from the fact that the limited number of simple representations cannot cover a wide variety of natural language forms in which a given relation may be expressed [6]. Lexical patterns fail to capture long-distance and complex relations. Besides, it is also useful to know that *"her"* and *"BiBi"* refer to the same person, which can be handled by entity co-reference resolution [22]. If "*her*" can be resolved correctly, we can extract the relation *spouse* from *"her husband, Ashiq"* easily. However, it is a challenging task [1] and co-reference failure was the one of the most frequent source of the errors in relation extraction [25]. Considering the unsatisfying quality of state-of-the-art co-reference resolution and long-distance relations,

[1] State-of-the-art system achieves an average F-score of 54.62% on the development set [23], [24].

sentence-level dependency parsing trees are widely used to construct patterns by previous methods. However, to reduce the resulting large amount of noisy candidates, we will heavily rely on linguistic constraints and heuristics.

Compared to flat syntactic structures, in this thesis we use graph-based methods to analyze the global structure of a dependency tree which can be used to mine the triggers given an argument pair for traditional relation extraction tasks such as Slot Filling (Chapter 3). We can also take advantage of this global structure to identify important open domain relational triples without requiring a predefined relation schema (Chapter 4). We follow two assumptions in this thesis to mine trigger and argument pairs and we will introduce them in the following chapters.

Heuristic 1 (Trigger Identification) *A trigger is usually an important node relative to the query and filler nodes in the dependency graph of a context sentence.*

Heuristic 2 (Argument Pair Identification) *A candidate relational triple is likely to be salient if its two arguments are strongly connected in a dependency tree.*

In summary, we propose to evaluate the relative importance of each word in the dependency tree of the context sentence, given a candidate argument pair, to identify its corresponding trigger word following the first heuristic. We evaluate the relative importance of each candidate entity pair within a sentence to identify the key points stressed by the author. Our experiments demonstrate that our method achieves comparable results to the state-of-the-art supervised models and achieves 12.9% higher recall than state-of-the-art open extraction methods by keeping those relative importance units within a sentence.

1.2.3 The Wisdom of Crowds in Truth Finding

Multi-source/system consolidation offers both a challenge and an opportunity to a single relation extraction system due to the resulting erroneous, conflicting, redundant or complements results. The "*truth finding*" problem has been studied in the data mining and database communities. Researchers observed that majority voting is a competitive baseline and they seldom applied linguistic analysis to validate facts.

1.2.4 Multidimensional Evidence in Relation Validation

Unfortunately, it is easy for majority voting to go wrong without deep analysis of corresponding evidence text. We aim to validate facts by judging both the correctness of the fact and pieces of supporting evidence which are automatically generated from unstructured texts by systems, which has been ignored before. We propose to integrate relation extraction and truth-finding research and present a novel unsupervised *multi-dimensional truth finding* framework which incorporates signals from multiple information sources, multiple relation extraction systems, and multiple pieces of evidence by extended dependency graph construction and multi-layer deep linguistic analysis in Chapter 5.

1.3 Case Studies

Knowledge Base Population (KBP) is an evaluation track of Text Analysis Conference (TAC), a workshop organized by the National Institute of Standards and Technology (NIST) since 2009. The main goal of KBP is to gather information about an entity that is scattered among the documents in a large collection, and then use the extracted information to populate an existing knowledge base (KB) [26]. Slot Filling (SF) and Slot Filling Validation (SFV) are two important KBP tasks targeting at relation extraction and question answering technologies [26]–[29].

To evaluate the relation extraction and validation performance of our methods, we choose SF and SFV as our case studies, for which data sets are publicly available. In general, we can regard SF as filling the incomplete Wikipedia infoboxes based on reading a large text corpus of blogs and newswire automatically. SFV can be considered as validating the content of Wikipedia infoboxes simultaneously filled by multiple systems from multiple information sources.

1.3.1 Slot Filling

Slot Filling aims to extract the values (***slot fillers***) of specific attributes (***slot types***) for a given entity (***query***) from a large-scale corpus and provide justification sentences to support these slot fillers.

For regular Slot Filling, a query can be either a PERSON (PER) or an ORGANIZA-

TION (ORG) entity. A filler can be an entity (*e.g.*, person, organization, geo-political), a value (*e.g.*, a number or a date), or a string (*e.g.*, disease, crime). Moreover, single-value slots can only have a single slot filler (*e.g.*, city of birth), while list-value slots can take multiple slot fillers (*e.g.*, cities of residence).

SF defined 41 slot types (Table A.1): 25 slot types for person queries and 16 slot types for organization queries. For instance, given a relevant document of the person query *Lucille Clifton*:

> A Poet's Voice Silenced
> By Matt Schudel WASHINGTON POST-BLOOMBERG–02-21-10 1430ET
> . . .
> **Lucille Clifton**, who died of a **bacterial infection Feb. 13** at Johns Hopkins Hospital in **Baltimore** at **73**, was one of the most important and most popular **poets** of our time. As prizes and honorary degrees came her way, she lived quietly in **Columbia, Md.**, commuting for years to teach at **St. Mary's College** on Maryland's Eastern Shore.
> . . .
> "In the choice between things and people," she said, "I choose people."

SF aims to discover the following slots (Table 1.1):

Table 1.1: Slots extracted from the example document.

Query	Slot Type	Filler	Filler Type
Lucille Clifton	per:date_of_death	2010-02-13	Value
	per:age	73	Value
	per:cause_of_death	bacterial infection	String
	per:title	poet	String
	per:city_of_death	Baltimore	Name
	per:cities_of_residence	Columbia	Name
	per:statesorprovinces_of_residence	Md.	Name
	per:employee_or_member_of	St. Mary's College	Name

Compared with the knowledge base completion based on inference, Slot Filling aims to extract attributes for a given, query which might not have a KB entry in existing

KBs, from a large-scale corpus. For example, in 2013, 18 SF teams extract 730 correct relational triples in total given 100 predefined queries. However, the number of relational triples founded in existing reference KBs is only 91.

1.3.2 Slot Filling Validation

Similar, complementary, or conflicting information may exist in large-scale multiple heterogeneous sources. Slot Filling Validation [26] focuses on refinement of the output from multiple SF systems by either combining information, or applying more extensive linguistic processing to validate individual candidate slot fillers [30]. Specifically, given a set of SF system submissions, a SFV system classifies each candidate slot filler as correct or incorrect, by considering the correctness of the facts and their associated context sentences.

1.4 Hypotheses

We propose the following hypotheses for relation extraction and validation.

- **Hypothesis 1**: Effective Slot Filling and Validation approaches leads to the trustworthy knowledge base population from unstructured texts automatically.
- **Hypothesis 2**: The inter-dependencies among relation components (*i.e.*, argument pairs and triggers) are language-independent and they are useful to enhance both the quality and portability of relation extraction.
- **Hypothesis 3**: A candidate relational triple is more likely to be salient if its two arguments are strongly connected in a dependency tree of its evidence sentence.
- **Hypothesis 4**: The capability of relative importance evaluation enables one open relation extraction to extract more diverse and qualified relational triples.
- **Hypothesis 5**: A relational triple is more likely to be truth if it is extracted by many trustworthy relation extraction systems and derived from many trustworthy information sources.

1.5 Contributions of the Thesis

1. To the best of our knowledge, this is the first open Relation Extraction method which exploits the global structure of a dependency tree to extract salient relational triples. This is also the first unsupervised relation grounding method to name relation types for open RE based on KB triples and intra-sentence context information. Experiments on the English Slot Filling (SF) [26], [28] 2013 dataset demonstrate that our approach outperforms state-of-the-art open RE approaches.

2. We proposed the first unsupervised graph mining approach to identify triggers which does not require any annotated data or external knowledge bases for supervision. Our proposed framework is language-independent. These features ensure easy portability to new languages or adaptability to any new relation types and to specialized domains with less effort as long as a few trigger seeds, name tagging and dependency parsing capabilities exist. Experiments on the English Slot Filling (SF) [26], [28] 2013 dataset demonstrate that our approach outperforms state-of-the-art supervised approaches.

3. We proposed a novel unsupervised multi-dimensional truth-finding model incorporating signals both from multiple sources, multiple systems, and multiple pieces of evidence based on extended dependency graph with multi-layer linguistic analysis. Experiments demonstrate that our approach can find truths accurately. Our truth finding model can also be applied to expedite the human annotation process.

1.6 Related Publications

Some of the proposed research work has been published in the following peer-reviewed conferences.

- Pattern-based Relation Extraction: [31] developed a pattern-based method based on dependency paths. [32] further improved pattern-based method by considering the trigger scopes.

- Unsupervised Relation Extraction: [33] explored the sentence-level global dependency graph to identify the trigger automatically as well as the connection among

trigger, query, and the slot filler candidate.

- Open Relation Extraction and Grounding: [34] proposed a novel importance-based open RE approach by exploiting the global structure of a dependency tree to extract salient triples. We also designed an unsupervised method to name relation types by grounding relational triples to a large-scale KB schema.

- Truth Finding in Relation Validation: [35] presented a multi-dimensional truth finding framework for knowledge refinement.

1.7 Thesis Structure

In this thesis, we overview related work in the literature (Chapter 2). Based on Hypothesis **1** and **2**, we propose an unsupervised language-independent trigger-driven relation extraction method (Chapter 3). Based on Hypothesis **3** and **4**, we introduce an open RE and grounding approach (Chapter 4). Following Hypothesis **5**, we propose an unsupervised multi-dimensional Truth-Finding framework for relation validation (Chapter 5). Finally we discuss the remaining challenges (Chapter 6).

CHAPTER 2
Related Work

2.1 Successful Relation Extraction Methods Used for Slot Filling

Supervised classification is one of the most successful state-of-the-art relation extraction techniques. Considering any pair of query and candidate slot filler as an instance, classifiers are trained from manually labeled data through active learning [21] or noisy labeled data through distant supervision [19], [36] to predict the existence of a specific relation between them.

Pattern-based methods have also been proven to be effective in SF in the past years [15], [16], [31], [37]. Dependency-based patterns achieve better performance since they can capture long-distance relations. Most of these approaches assume that a relation exists between a query and a filler if there is a dependency path connecting them and all the words on the path are equally regarded as trigger candidates. We explore the complete graph structure of a sentence rather than chains/subgraphs as in previous work. Our previous research focused on identifying the relation between filler and trigger by extracting filler candidates from the identified scope of a trigger (*e.g.*, [32]). We found that each slot-specific trigger has its own scope, and corresponding fillers seldom appear outside its scope. We did not compare with results from previous approaches which did not consider redundancy removal required in the official evaluations.

[38] built their SF system based on Open Information Extraction (IE) technology. Our method achieves much higher recall since dependency trees can capture the relations

---

Portions of this chapter previously appeared as: D. Yu, H. Huang, T. Cassidy, H. Ji, C. Wang, S. Zhi, J. Han, C. Voss, and M. Magdon-Ismail, "The wisdom of minority: unsupervised slot filling validation based on multi-dimensional truth-finding," in *Proc. 25th Conf. Computational Linguistics,* Dublin, Ireland, 2014, pp. 1567-1578.

Portions of this chapter previously appeared as: D. Yu, H. Ji, S. L and C. Lin "Why read if you can scan? trigger scoping strategy for biographical fact extraction," in *Proc. Conf. North Amer. Chapter of the Assoc. Computational Linguistics,* Denver, CO, 2015, pp. 1203-1208.

Portions of this chapter previously appeared as: D. Yu and H. Ji, "Unsupervised Person Slot Filling based on graph mining," in *Proc. 54th Annu. Meeting Assoc. Computational Linguistics,* Berlin, German, 2016, pp. 44-53.

Portions of this chapter previously appeared as: D. Yu, L. Huang, and H. Ji, "Open relation extraction and grounding," in *Proc. 8th Int. Conf. Natural Language Process.,* Taipei, Taiwan, 2017, pp. 1-11.

among query, slot filler and trigger in more complicated long sentences. In addition, our triggers are automatically labeled so that we do not need to design manual rules to classify relation phrases as in Open IE.

2.2 Open Information Extraction

Lexical or syntactic features and patterns have been widely used to extract relational triples [39]–[48]. Our work explores the global structure of a dependency tree to identify salient triples within a sentence. Some open IE approaches have the capability to extract relations between concepts or phrases [49]–[51]. Currently we focus on relations between two entities.

Given the SF schema, [52] manually design rules to map relational triples to slot types within hours. Researchers also use distantly labeled corpora to compute the PMI^2 value between open IE and SF relation pairs [47]. Instead, we propose a novel grounding approach which facilitates building a mapping table between KB relations and slot types. We do not compare with RE methods specifically designed for SF [16], [37], [53] since these methods actively search for candidate fillers of the given queries based on slot-specific training resources while ignoring the salient relations which are irrelevant to the queries or the predefined slot types.

2.3 Relation Grounding

Besides textual features, large-scale knowledge bases are widely used for distant supervised relation extraction [18], [54] to deal with the challenges caused by insufficient training data. [55] combine two relation representations trained from KB triples and context words independently for relation extraction. Recent studies such as [56] train relation representations of KB and textual relations jointly. Another kind of representations combining matrix factorization [57] with first-order logic information is learned by [58]. Compared with these previous efforts, our unsupervised grounding method does not need the aligned training corpus or relation mentions for KB tuples. [59] introduce an approach to map words to KB relations based on web text, but they only focus on verb phrases. Relation extraction methods based on distant supervision also take advantage of the large-scale KBs and mined sentences which contains at least one KB entity/argument

pair from external corpora to train relation-specific supervised models. KB triples and their aligned resources are mainly used to identify relations. In comparison, our method is fully unsupervised and we only use the KB triples and sentence-level context words associated with each candidate relational triple for relation typing.

2.4 Graph-Based Node Importance Computation

Graph-based algorithms such as PageRank [60] and TextRank [61] are useful in keyword extraction. The way we rank nodes is most similar to the work of [62] and [33] which generate the relative importance score of each node toward a set of preferred nodes. However, they only deal with unweighted undirected graphs.

2.5 Relation Validation and Truth Finding

Most previous SFV work (*e.g.*, [63],[64]) focused on filtering incorrect claims from multiple systems by simple heuristic rules, weighted voting, or costly supervised learning to rank algorithms. We are the first to introduce the truth finding concept to this task.

The "truth finding" problem has been studied in the data mining and database communities (*e.g.*, [65]–[77]). Compared with the previous work, our truth finding problem is defined under a unique setting: each *response* consists of a claim and supporting evidence, automatically generated from unstructured natural language texts by a SF *system*. The judgement of a *response* concerns both the truth of the claim and whether the *evidence* supports the claim. This has never been modeled before. We mine and exploit rich linguistic knowledge from multiple lexical, syntactic and semantic levels from evidence sentences for truth finding. In addition, previous truth finding work assumed most claims are likely to be true. However, most SF systems have hit a performance ceiling of 35% F-measure, and false responses constitute the majority class (72.02%) due to the imperfect algorithms as well as the inconsistencies of information sources. Furthermore, certain truths might only be discovered by a minority of good systems or from a few good sources. For example, 62% of the true responses are produced only by 1 or 2 of the 18 SF systems.

CHAPTER 3
Trigger-Driven Traditional Relation Extraction

In this chapter, we first discuss why we put special emphasis on triggers compared with a set of patterns for relation extraction. We can use existing gazetteers to identify triggers for relation extraction which might generate irrelevant trigger mentions. In this chapter, we propose a simple yet effective unsupervised Slot Filling approach based on the following two observations: (1) a trigger is usually a salient node relative to the query and filler nodes in the dependency graph of a context sentence; (2) a relation is likely to exist if the query and candidate filler nodes are strongly connected by a relation-specific trigger. Thus we design a graph-based trigger-driven algorithm to automatically identify triggers based on personalized PageRank and Affinity Propagation for a given (*query*, *filler*) pair and then label the slot type based on the identified triggers. Slot Filling is a traditional relation extraction task since it requires a predefined relation schema (*e.g.*, the 41 slot types in Slot Filling).

3.1 From Patterns to Triggers

Patterns have played an important role in extracting relations and they are especially useful for relations within very short distances. Two kinds of patterns are commonly used: lexical patterns and syntactic patterns.

Such patterns often fail to capture the diverse and complex ways to express a certain relation type. Therefore, high-quality patterns yield quite high precision, but relatively low recall. In addition, it is very time-consuming and expensive to expand an existing pattern set and keep it clean. Previously researchers applied bootstrapping algorithms to generate more relation-specific patterns [78], [79] based on a number of relation seeds and limited patterns. However, these methods suffer from low precision and semantic

Portions of this chapter previously appeared as: D. Yu and H. Ji, "Unsupervised Person Slot Filling based on graph mining," in *Proc. 54th Annu. Meeting Assoc. Computational Linguistics,* Berlin, German, 2016, pp. 44-53.

Portions of this chapter previously appeared as: D. Yu, H. Ji, S. L and C. Lin "Why read if you can scan? trigger scoping strategy for biographical fact extraction," in *Proc. Conf. North Amer. Chapter Assoc. Computational Linguistics,* Denver, CO, 2015, pp. 1203-1208.

drift [18].

In addition, we have to rebuilt patterns for a new language since they are designed as language-specific. For example, the following English lexical pattern to extract birth dates can not be applied to Chinese text directly.

Query, **born** in **Filler**

Trigger is defined as the smallest extent of a text which most clearly expresses an event occurrence or indicates a relation type. In this proposal, we define a trigger as the smallest extent of a text which clearly indicates a slot type. Many slot types, especially when the queries are person entities, are indicated by triggers. We call these slots *trigger-driven* slots. When multiple facts about a person entity are presented in a sentence, the author (e.g., a news reporter or a discussion forum poster) often uses explicit *trigger* words or phrases to indicate their relations with the entity. As a result, these inter-dependent facts and query entities are strongly connected via syntactic or semantic relations.

We notice that existing patterns (*e.g.*, [16], [80]) designed for trigger-driven slots always include slot-specific triggers. For example in Table 3.1, we can clearly see that the importance of triggers (*e.g.*, *brother*, *sister*) in patterns. It is almost impossible to summarize all the expressions for a slot type. However, the number of commonly used slot-specific trigger words is limited.

Table 3.1: Syntactic patterns used for extracting sibling filler [1] (Q: Query. F: Filler).

PER:SIBLING
[Q] *poss*$^{-1}$ **brother** *appos* **[F]**
[Q] *appos*$^{-1}$ **brother** *appos* **[F]**
[Q] *appos* **brother** *appos*-1 **[F]**
[Q] nsubjpass^{-1} survived *agent* **brother** *appos* **[F]**
[Q] *poss*$^{-1}$ **sister** *appos* **[F]**
[Q] *appos*$^{-1}$ **sister** *appos* **[F]**
[Q] *appos* **sister** *appos*$^{-1}$ **[F]**
[Q] *nsubjpass*$^{-1}$ survived *agent* **sister** *appos* **[F]**

Therefore, we made an attempt to extract relations without replying on existing patterns. We assume there might exist a relation between a query and a candidate filler which appear in the same sentence, if there is a dependency path connecting them and the

path contains a slot-specific trigger. Compared to lexical patterns, dependency paths can help us capture long-distance relations. However, this method yielded very high recall but relatively low precision since the above conditions can not strongly indicate the existence of relations.

For example, given the following sentence:

Penner *is survived by his brother,* ***John****, a copy editor at the Times, and his former* ***wife****, Times sportswriter* ***Lisa Dillman****.*

we can generate two dependency paths connecting the query *Penner* and two candidate fillers *John* and *Lisa Dillman* respectively as shown in Table 3.2. Based on the above assumption [31], both *Lisa Dillman* and *John* will be regarded as candidate sibling fillers since both of the paths contain the trigger *brother*.

Table 3.2: Dependency paths connecting Penner, John and Lisa.

Penner *nsubjpass*$^{-1}$ survived *nmod* **brother** *appos* **John**
Penner *nsubjpass*$^{-1}$ survived *nmod* **brother** *conj* **wife** *amod* **Lisa Dillman**

In the above example, the correct trigger indicating the relation between *Penner* and *Lisa* should be *wife* which also appears in the dependency path connection them. In this chapter, we aim to identify the trigger automatically rather than assuming all the words on the dependency path as triggers.

3.2 Connection Among Trigger, Query, and Filler

Previous successful SF methods have poor portability to a new language or a new slot type and they focus on the flat relation representation between the query and the candidate slot filler, while ignoring the global graph structure among them and other facts in the context.

In this chapter, we keep focusing on the *trigger-driven* person slot types in order to gain a better understanding of the connection among query, trigger and filler in a relation. Considering the limitations of previous flat representations for the relations between a query (Q) and a candidate slot filler (F), we focus on analyzing the whole dependency tree structure that connects Q, F and other semantically related words or phrases in each context sentence. Our main observation is that there often exists a trigger word (T) which

plays an important role in connecting Q and F in the dependency tree for trigger-driven slots. From the extended dependency tree shown in Figure 3.1, we can clearly see that "*divorced*" is most strongly connected to the query mention ("*he*") and the slot filler ("*Ellen Griffin Dunne*"). Therefore we can consider it as a trigger word which explicitly indicates a particular slot type.

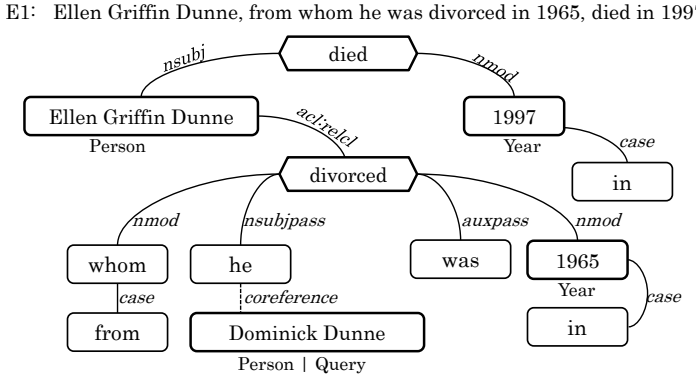

Figure 3.1: Extended dependency tree for E1.

Based on these observations, we propose a novel and effective **unsupervised graph mining** approach for person Slot Filling by deeply exploring the structures of dependency trees. It consists of the following three steps:

- **Step 1 - Candidate Relation Identification:** Construct an extended dependency tree for each sentence including any mention referring to the query entity. Identify candidate slot fillers based on slot type constraints (*e.g.*, the *spouse* fillers are limited to person entities) (Section 3.3).
- **Step 2 - Trigger Identification:** Measure the importance of each node in the extended dependency tree relative to Q and F, rank them and select the most important ones as the trigger set (Section 3.4).
- **Step 3 - Slot Typing:** For any given new slot type, automatically expand a few trigger seeds using the Paraphrase Database [81]. Then we use the expanded trigger set to label the slot types of identified triggers (Section 3.5).

This framework only requires name tagging and dependency parsing as pre-processing, and a few trigger seeds as input, and thus it can be easily adapted to a new language or a new slot type. Experiments on English and Chinese demonstrate that our approach dra-

matically advances state-of-the-art results for both pre-defined KBP slot types and new slot types.

3.3 Candidate Relation Identification

We first present how to build an extended dependency graph for each evidence sentence (Section 3.3.1) and generate query and filler candidate mentions (Section 3.3.2).

3.3.1 Extended Dependency Tree Construction

Given a sentence containing N words, we construct an undirected graph $G = (V, E)$, where $V = \{v_1, \ldots, v_N\}$ represents the words in a sentence, E is an edge set, associated with each edge e_{ij} representing a dependency relation between v_i and v_j. We first apply a dependency parser to generate basic uncollapsed dependencies by ignoring the direction of edges. Figure 3.1 shows the dependency tree built from the example sentence. In addition, we annotate an entity, time or value mention node with its type. For example, in Figure 3.1, "*Ellen Griffin Dunne*" is annotated as a person, and "*1997*" is annotated as a year. Finally we perform co-reference resolution, which introduces implicit links between nodes that refer to the same entity. We replace any nominal or pronominal entity mention with its coreferential name mention. For example, "*he*" is replaced by "*Dominick Dunne*" in Figure 3.1. Formally, an extended dependency tree is an annotated tree of entity mentions, phrases and their links.

3.3.2 Relation Arguments Identification

Given a query q and a set of relevant documents, we construct a dependency tree for each sentence. We identify a person entity e as a query mention if e matches the last name of q or e shares two or more tokens with q. For example, "*he/Dominick Dunne*" in Figure 3.1 is identified as a mention referring to the query *Dominick Dunne*. For each sentence which contains at least one query mention, we regard all other entities, values and time expressions as candidate fillers and generate a set of entity pairs (q, f), where q is a query mention, and f is a candidate filler. In Example $E1$, we can extract three entity pairs (*i.e.*, $\{$*Dominick Dunne*$\} \times \{$*Ellen Griffin Dunne*, 1997, 1965$\}$). For each entity pair, we represent the query mention and the filler candidate as two sets of nodes Q and

F respectively, where $Q, F \subseteq V$.

3.4 Trigger Identification

In this section, we proceed to introduce an unsupervised graph-based method to identify triggers for each query and candidate filler pair. We rank all trigger candidates (Section 3.4.1) and then keep the top ones as the trigger set (Section 3.4.2).

3.4.1 Trigger Candidate Ranking

As we have discussed in Section 3.2, we can consider trigger identification problem as finding the important nodes relative to Q and F in G. Algorithms such as Pagerank [60] are designed to compute the global importance of each node relative to all other nodes in a graph. By redefining the importance according to our preference toward F and Q, we can extend PageRank to generate relative importance scores.

We use the random surfer model [60] to explain our motivation. Suppose a random surfer keeps visiting adjacent nodes in G at random. The expected percentage of surfers visiting each node converges to the PageRank score. We extend PageRank by introducing a "back probability" β to determine how often surfers jump back to the ***preferred nodes*** (*i.e.*, Q or F) so that the converged score can be used to estimate the relative probability of visiting these preferred nodes.

Given G and a set of preferred nodes R where $R \subseteq V$, we denote the relative importance for all $v \in V$ with respect to R as $I(v \mid R)$, following the work of [62].

For a node v_k, we denote $N(k)$ as the set of neighbors of v_k. We use $\pi(k)$, the k-th component of the vector $\boldsymbol{\pi}$, to denote the stationary distribution of v_k where $1 \leq k \leq |V|$. We define a preference vector $\boldsymbol{p_R} = \{p_1, ..., p_{|V|}\}$ such that the probabilities sum to 1, and p_k denotes the relative importance attached to v_k. p_k is set to $1/|R|$ for $v_k \in R$, otherwise 0. Let $\boldsymbol{A}$ be the matrix corresponding to the graph G where $A_{jk} = 1/|N(k)|$ and $A_{jk} = 0$ otherwise.

For a given $\boldsymbol{p_R}$, we can obtain the personalized PageRank equation [82]:

$$\boldsymbol{\pi} = (1 - \beta)\boldsymbol{A}\boldsymbol{\pi} + \beta \boldsymbol{p_R} \tag{3.1}$$

where $\beta \in [0, 1]$ determines how often surfers jump back to the nodes in R. We set $\beta = 0.3$ in our experiment. The solution π to Equation 3.1 is a steady-state importance distribution induced by $\boldsymbol{p_R}$. Based on a theorem of Markov Theory, a solution π with $\sum_{k=1}^{|V|} \pi(k) = 1$ always exists and is unique [83].

We define relative importance scores based on the personalized ranks described above, *i.e.*, $I(v \mid R) = \pi(v)$ after convergence, and we compute the importance scores for all the nodes in V relative to Q and F respectively.

A query mention in a sentence is more likely to be involved in multiple relations while a filler is usually associated with only one slot type. Therefore we combine two relative importance scores by assigning a higher priority to $I(v \mid F)$ as follows.

$$\mathcal{I}(v \mid \{Q, F\}) = I(v \mid F) + I(v \mid F) \cdot I(v \mid Q) \tag{3.2}$$

We discard a trigger candidate if it is (or part of) an entity which can only act as a query or a slot filler. We assume a trigger can only be a noun, verb, adjective, adverb or preposition. In addition, verbs, nouns and adjectives are more informative to be triggers. Thus, we remove any trigger candidate v if it has a higher $\mathcal{I}(v \mid \{Q, F\})$ than the first top-ranked verb/noun/adjective trigger candidate.

For example, we rank the candidate triggers based on the query and slot filler pair ("*Dominick Dunne*", "*Ellen Griffin Dunne*") as shown in Figure 3.2.

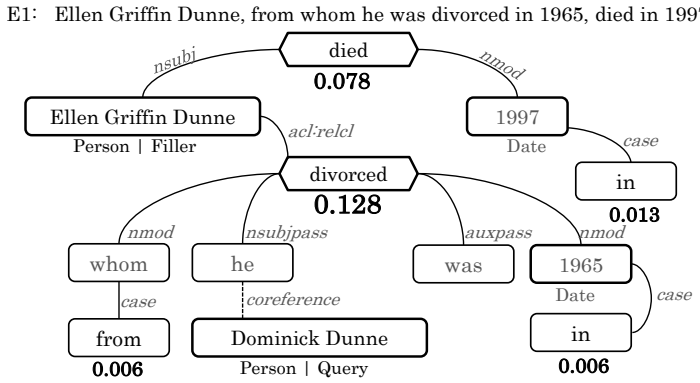

Figure 3.2: Importance scores of trigger candidates relative to query and filler in E1.

3.4.2 Trigger Candidate Selection

Given Q and F, we can obtain a relative importance score $\mathcal{I}(v \mid \{Q, F\})$ for each candidate trigger node v in V as shown in Section 3.4.1. We denote the set of trigger candidates as $T = \{t_1, \cdots, t_n\}$ where $n \leq |V|$.

Since a relation can be indicated by a single trigger word, a trigger phrase or even multiple non-adjacent trigger words, it is difficult to set a single threshold even for one slot type. Instead, we aim to automatically classify top ranked candidates into one group (*i.e.*, a trigger set) so that they all have similar higher scores compared to other candidates.

Therefore, we define this problem as a clustering task. We mainly consider clustering algorithms which do not require pre-specified number of clusters.

We apply the affinity propagation approach to take as input a collection of real-valued similarity scores between pairs of candidate triggers. Real-valued ***messages*** are exchanged between candidate triggers until a high-quality set of ***exemplars*** (centers of clusters), and corresponding clusters gradually emerges [84].

There are two kinds of messages exchanged between candidate triggers: one is called ***responsibility*** $\gamma(i, j)$, sent from t_i to a candidate exemplar t_j; the other is ***availability*** $\alpha(i, j)$, sent from the candidate exemplar t_j to t_i.

The calculation of each procedure iterates until convergence. To begin with, the availabilities are initialized to zero: $\alpha(i, j) = 0$. Then the responsibilities are computed using the following rule:

$$\gamma(i, j) \leftarrow s(i, j) - \max_{j' s.t. j' \neq j} \{\alpha(i, j') + s(i, j')\} \tag{3.3}$$

where the similarity score $s(i, j)$ indicates how well t_j is suited to be the exemplar for t_i. Whereas the above responsibility update lets all candidate exemplars compete for the ownership of a trigger candidate t_i, the following availability update gathers evidence from trigger candidates as to whether each candidate exemplar would make a good exemplar:

$$\alpha(i,j) \leftarrow \min\Big\{0, \gamma(j,j) + \sum_{i's.t.i' \notin \{i,j\}} \max\{0, \gamma(i',j)\}\Big\} \tag{3.4}$$

Given T, we can generate an $n \times n$ affinity matrix $\boldsymbol{M}$ which serves as the input of the affinity propagation. M_{ij} represents the negative squared difference in relative importance score between t_i and t_j (Equation 3.5).

$$M_{ij} = -(\mathcal{I}(i \mid \{Q, F\}) - \mathcal{I}(j \mid \{Q, F\}))^2 \tag{3.5}$$

We compute the average importance score for all the clusters after convergence and keep the one with the highest average score as the trigger set. For example, given the query and slot filler pair in Figure 3.3, we obtain trigger candidates $T = \{died, divorced, from, in, in\}$ and their corresponding relative importance scores. After the above clustering, we obtain three clusters and choose the cluster $\{divorced\}$ with the highest average relative importance score (0.128) as the trigger set.

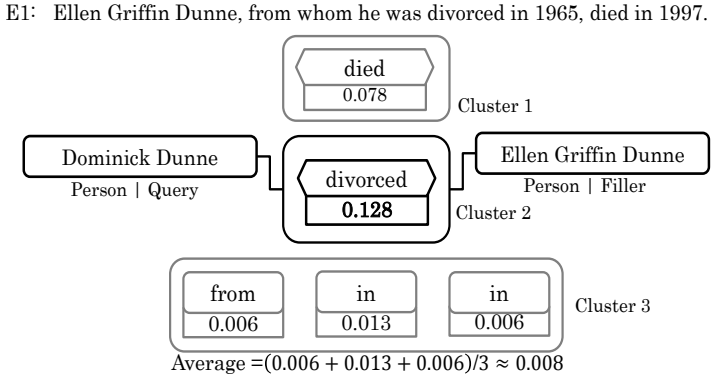

Figure 3.3: Trigger candidate filtering for E1.

3.5 Relation Type Labeling

In this section, we will introduce how to label the slot type for an identified relation tuple (Q, T, F). The simplest solution is to match T against existing trigger gazetteers

for certain types of slots. For example, Figure 3.4 shows how we label the relation as a *spouse* slot type.

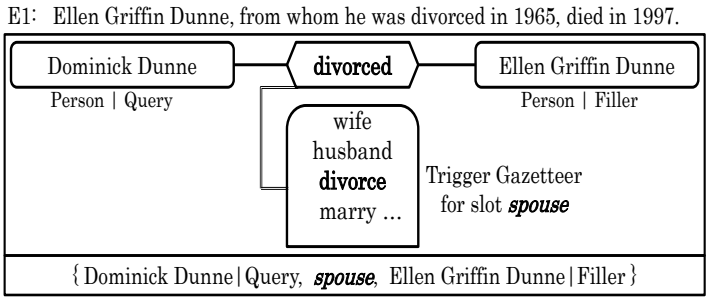

Figure 3.4: Example of slot type labeling.

In fact, some trigger gazetteers have already been constructed by previous work such as [32]. However, manual construction of these triggers heavily rely upon labeled training data and high-quality patterns, which would be unavailable for a new language or a new slot type.

Inspired by the trigger-based event extraction work [85], we propose to extract trigger seeds from the Slot Filling annotation guideline [86] and then expand them by paraphrasing techniques. For each slot type we manually select two trigger seeds from the guideline and then use the Paraphrase Database (PPDB) [81], [87] to expand these seeds. Specifically, we select top-20 lexical paraphrases based on similarity scores as our new triggers for each slot type. Some examples are shown in Table 3.3.

Table 3.3: PPDB-based trigger expansion examples.

Seeds	Slot Types	Expanded Triggers
assassinate	*death*	kill, die, slay, murder
graduate	*schools*	PhD, supervisor, diploma
sister	*siblings*	twin, half-brother, sibling
marriage	*spouse*	married, spouse, matrimony
dead, assassinate	*death*	kill, die, slay, murder
graduate, doctorate	*school*	PhD, supervisor, diploma
brother, sister	*sibling*	twin, half-brother, sibling
wife, marriage	*spouse*	married, spouse, matrimony

3.6 Filler Validation

After we label each relation tuple, we perform the following validation steps to filter noise and remove redundancy. For many slot types, there are some specific constraints on entity types of slot fillers defined in the task specification. For example, *employee_or_member_of* fillers should be either organizations or geopolitical entities, while family slots (*e.g.*, *spouse* and *children*) expect person entities. We apply these constraints to further validate all relation tuples.

Moreover, ***single-value*** slots can only have a single filler (*e.g.*, *date_of_birth*), while ***list-value*** slots can take multiple fillers (*e.g.*, *cities_of_residence*). However, we might extract conflicting relation tuples from multiple sentences and sources. For each relation tuple, it can also be extracted from multiple sentences, and thus it may receive multiple relative importance scores. We aim to keep the most reliable relation tuple for a single-value slot.

For a single-value slot, suppose we have a collection of relation tuples R which share the same query. Given $r \in R$ with a set of relative importance scores $\mathcal{I} = \{i_1, i_2, \cdots, i_n\}$, we can regard the average score of $\mathcal{I}$ as the credibility score of r. The reason is that the higher the relative importance score, the more likely the tuple is to be correct. In our experiments, we use the weighted arithmetic mean as follows so that higher scores can contribute more to the final average:

$$\bar{i} = \frac{\sum_{k=1}^{n} w_k \cdot i_k}{\sum_{k=1}^{n} w_k} \tag{3.6}$$

where w_k denotes the non-negative weight of i_k. When we regard the weight w_k equal to the score i_k, Equation 3.6 can be simplified as:

$$\bar{i} = \frac{\sum_{k=1}^{n} w_k^2}{\sum_{k=1}^{n} w_k} \tag{3.7}$$

We calculate the weighted mean $\bar{i}$ for each $r \in R$ and keep the relation tuple with

the highest $\bar{i}$.

3.7 Experiments

3.7.1 Data and Scoring Metric

In order to evaluate the quality of our proposed framework and its portability to a new language, we use TAC-KBP2013 English Slot Filling (ESF), TAC-KBP 2015 English Cold Start Slot Filling (CSSF) and TAC-KBP2015 Chinese Slot Filling (CSF) data sets for which we can compare with the ground truth and state-of-the-art results reported in previous work. The source collection includes news documents, web blogs and discussion forum posts. In ESF there are 50 person queries and on average 20 relevant documents per query; while in CSF there are 51 person queries, and on average 5 relevant documents per query. We use the official TAC-KBP Slot Filling evaluation scoring metrics: *Precision* (P), *Recall* (R) and *F-score* (F_1) [28] to evaluate our results.

We only test our method on 18 trigger-driven person slot types shown in Table 4.4. Some other slot types (*e.g.*, *age*, *origin*, *religion* and *title*) do not rely on lexical triggers in most cases; instead the query mention and the filler are usually adjacent or seperated by a comma. In addition, we do not deal with the two remaining trigger-driven person slot types (*i.e.*, *cause_of_death* and *charges*) since these slots often expect other types of concepts (*e.g.*, a disease or a crime phrase).

3.7.2 English Slot Filling

We apply Stanford CoreNLP [88] for English part-of-speech (POS) tagging, name tagging, time expression extraction, dependency parsing and coreference resolution. In Table 4.4 we compare our approach with two state-of-the-art English Slot Filling methods: a distant supervision method [20] and a hybrid method that combines distant and partial supervision [21]. Our method outperforms both methods dramatically. KBP2015 English Cold Start Slot Filling is a task which combines entity mention extraction and slot filing [29]. Based on the released evaluation queries from KBP2015 Cold Start Slot Filling, our approach achieves 39.2% overall F-score on 18 person trigger-driven slot types, which is significantly better than state-of-the-art [89] on the same set of news documents (Table 3.5).

Table 3.4: English Slot Filling F_1 (%) (KBP2013 SF data set).

Slot Type	Our Approach	Roth'13	Angeli'14
siblings	**62.9**	48.0	40
other_family	**42.4**	11.8	0
spouse	58.7	40.0	**66**
children	**66.7**	27.3	27
parents	43.1	**47.8**	39
schools_attended	**81.4**	30.2	60
date_of_birth	87.0	60.0	**92**
date_of_death	**73.2**	3.2	48
state_of_birth	**55.6**	30.8	17
state_of_death	**88.2**	53.3	0
city_of_birth	**70.0**	64.0	25
city_of_death	72.7	**73.7**	30
country_of_birth	**75.0**	0.0	0
country_of_death	**70.0**	46.2	18
states_of_residence	**57.1**	25.6	12
cities_of_res.	**61.4**	38.8	38
countries_of_res.	**45.7**	20.0	41
employee_of	**43.8**	18.5	38
Overall	**57.4**	32.3	–

Compared to the previous work, our method discards a trigger-driven relation tuple if it is not supported by triggers. For example, "*Poland*" is mistakenly extracted as the country of residence of "*Mandelbrot*" by distant supervision [20] from the following sentence:

A professor emeritus at Yale University, ***Mandelbrot*** *was born in* ***Poland*** *but as a child moved with his family to* ***France*** *where he was educated.*

maybe because the relation tuple (*Mandelbrot*, *live_in*, *Poland*) indeed exists in external knowledge bases. Given the same entity pair, our method identifies "*born*" as the trigger word and labels the slot type as *country_of_birth*.

When there are several triggers indicating different slot types in a sentence, our approach performs better in associating each trigger with the filler it dominates by analyzing the whole dependency tree. For example, given a sentence:

Haig *is survived by his wife of 60 years, Patricia; his children Alexander, Brian and* ***Barbara****; eight grandchildren; and his brother, the Rev. Francis R. Haig.*

(*Haig*, *sibling*, *Barbara*) is the only relation tuple extracted from the above sentence

Table 3.5: English Cold Start Slot Filling F_1 (%) (KBP2015 CSSF data set).

Slot Type	Our Approach	Angeli'15
siblings	**48.0**	26.1
other_family	0.0	**33.3**
spouse	14.3	**15.4**
children	**72.8**	0.0
parents	**25.0**	14.3
schools_attended	**63.6**	42.1
date_of_birth	0.0	**80.0**
date_of_death	**44.0**	0.0
state_of_birth	0.0	**33.3**
state_of_death	0.0	**15.4**
city_of_birth	0.0	**85.7**
city_of_death	0.0	0.0
country_of_birth	0.0	**66.7**
country_of_death	**100.0**	0.0
states_of_residence	0.0	0.0
cities_of_res.	0.0	**50.0**
countries_of_res.	0.0	0.0
employee_of	**60.0**	26.7
Overall	**39.2**	27.6

by the previous method. Given the entity pair (*Haig*, *Barbara*), the relative importance score of "*children*" (0.1) is higher than the score of "*brother*" (0.003), and "*children*" is kept as the only trigger candidate after clustering. Therefore, we extract the tuple (*Haig*, *children*, *Barbara*) instead. In addition, we successfully identify the missing fillers for other slot types: *spouse* (*Patricia*), *children* (*Alexander, Brian and Barbara*) and *siblings* (*Francis R. Haig*) by identifying their corresponding triggers.

In addition, flat relation representations fail to extract the correct relation (*i.e.*, *alternate_names*) between "*Dandy Don*" and "*Meredith*" since "*brother*" is close to both of them in the following sentence:

In high school and at Southern Methodist University, where, already known as ***Dandy Don*** *(a nickname bestowed on him by his brother) ,* ***Meredith*** *became an all-American.*

Table 3.6: Examples for new slot types.

Evidence Sentence	Slot Type	Query	Extracted Fillers
Many of ***his*** subjects were friends from his previous life , such as ***Elizabeth Taylor*** and ***Gloria Vanderbilt*** .	*friends*	Dominick Dunne	Gloria Vanderbilt; Elizabeth Taylor
Toby Keith hit an emotional note with a performance of "Cryin' For Me (Wayman's Song)," dedicated to ***his*** late friend, jazz artist and former basketball star ***Wayman Tisdale***, who died last May.	*friends*	Wayman Tisdale	Toby Keith
"I think all of her writing came from ***her*** heart," ***Michael Glaser***, a longtime colleague at St. Mary's and former Maryland poet laureate, said last week.	*colleagues*	Lucille Clifton	Michael Glaser
Cunningham has collaborated on two books: "Changes: Notes on Choreography," with Frances Starr, and "The Dancer and the Dance," with ***Jacqueline Lesschaeve***.	*collaborators*	Merce Cunningham	Jacqueline Lesschaeve

3.7.3 Adapting to New Slot Types

Our framework can also be easily adapted to new slot types. We evaluate it on three new person list-value slot types: *friends*, *colleagues* and *collaborators*.

We use "*friend*" as the slot-specific trigger for the slot *friends* and "*colleague*" for the slot *colleagues*. "*collaborate*", "*cooperate*" and "*partner*" are used to type the slot *collaborators*.

We manually annotate ground truth for evaluation. It is difficult to find all the correct fillers for a given query from millions of documents. Therefore, we only calculate precision. Experiments show we can achieve 56.3% for *friends*, 100% for *colleagues* and 60% for *collaborators* (examples shown in Table 3.6).

3.7.4 Impact of Trigger Mining

In Section 3.4.2, we keep top-ranked trigger candidates based on clustering rather than threshold tuning. We explore a range of thresholds for comparison, as shown in Figure 3.5. Our approach achieves 57.4% F-score, which is comparable to the highest F-score 58.1% obtained by threshold tuning.

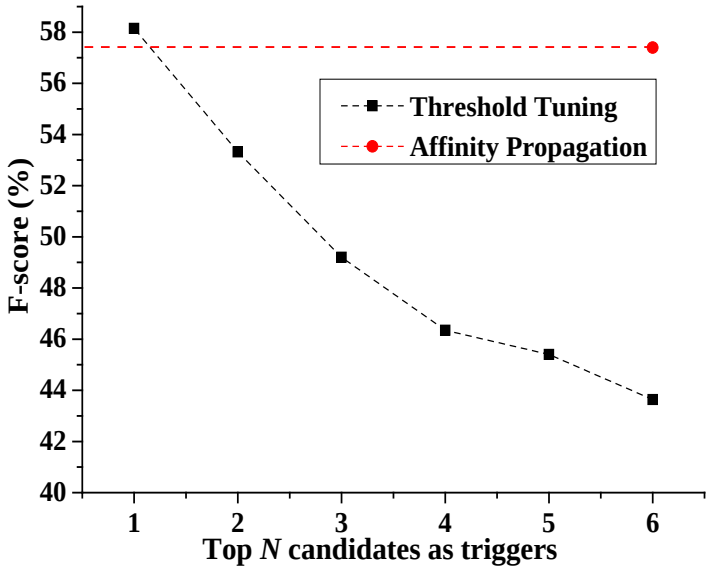

Figure 3.5: The effect of the number of trigger candidates on ESF.

We also measure the impact of the size of the trigger gazetteer. We already outperform state-of-the-art by using PPDB to expand triggers mined from guidelines as shown in Table 3.7. As the size of the trigger gazetteer increases, our method (marked with a $\star$) achieves better performance.

Table 3.7: The effect of trigger gazetteers on ESF (size: the number of triggers).

Method	Size	F_1 (%)
State-of-the-art [20]	–	32.3
Guideline seeds*	20	27.3
Guideline seeds + PPDB expansion*	220	**38.9**
Manually Constructed Trigger Gazetteers*	7,463	**57.4**

3.7.5 Chinese Slot Filling

As long as we have the following resources: (1) a POS tagger, (2) a name tagger, (3) a dependency parser and (4) slot-specific trigger gazetteers, we can apply the framework to a new language. Coreference resolution is optional.

We demonstrate the portability of our framework to Chinese since all the resources mentioned above are available. We apply Stanford CoreNLP [88] for Chinese POS tag-

ging, name tagging [90] and dependency parsing [91]. To explore the impact of the quality of annotation resources, we also use a Chinese language analysis tool: Language Technology Platform (LTP) [92]. We use the full set of Chinese trigger gazetteers [32]. Experimental results (Table 3.8) demonstrate that our approach can serve as a new and promising benchmark. As far as we know, there are no results available for comparison.

However, the performance of Chinese SF is heavily influenced by the relatively low performance of name tagging since our method returns an empty result if it fails to find any query metnion. About 20% and 16% queries cannot be recognized by CoreNLP and LTP respectively. One reason is that many Chinese names are also common words. For example, a buddhist monk's name "觉醒" (*wake*) is identified as a verb rather than a person entity.

A dependency parser is indispensable to produce reliable rankings of trigger candidates. Unfortunately, a high-quality parser for a new language is often not available because of language-specific features. For example, in Chinese a single sentence about a person's biography often contains more than five co-ordinated clauses, each of which

Table 3.8: Chinese Slot Filling F_1 (%) (KBP2015 CSF data set).

Slot Type	CoreNLP-based	LTP-based
siblings	40.0	**57.1**
other_family	**40.0**	0.0
spouse	40.0	**48.0**
children	19.0	**21.4**
parents	0.0	**25.0**
schools_attended	11.1	**17.1**
date_of_birth	**42.4**	0.0
date_of_death	**48.5**	0.0
state_of_birth	38.1	**52.2**
state_of_death	55.6	**70.0**
city_of_birth	**28.6**	26.7
city_of_death	33.3	**42.9**
country_of_birth	11.8	11.8
country_of_death	0.0	0.0
states_of_residence	**30.8**	29.6
cities_of_residence	27.3	**34.8**
country_of_residence	**6.5**	0.0
employee_of	31.0	**31.2**
Overall	**29.6**	28.3

includes a trigger. Therefore a dependency parser adapted from English often mistakenly identifies one of the triggers as a main predicate of the sentence.

3.8 Summary

In this chapter, we demonstrate the importance of deep mining of dependency structures for Slot Filling. Our approach achieves 11.6%-25% higher F-score over state-of-the-art English Slot Filling methods. Our experiments also demonstrate that as long as a few trigger seeds, name tagging, POS tagging, and dependency parsing capabilities exist, this approach can be quickly adapted to any language and new slot types. Our promising results on Chinese Slot Filling can serve as a new benchmark.

CHAPTER 4
Importance-Based Open Relation Extraction and Grounding

In the previous chapter, we introduce an unsupervised method which relies on importance-based trigger identification for a traditional relation extraction task. There are still some limitations. First, we heavily rely on human-crafted or automatically constructed gazetteers to type a relation. We focus on query-based attribute extraction while ignoring other important facts in the same sentence. To handle these limitations, in this chapter, we focus on open Relation Extraction (open RE) which aims to extract relational triples from large-scale corpora.

Previous open RE approaches mainly rely on linguistic patterns and constraints to extract important relational triples from large-scale corpora. However, they lack of abilities to cover diverse relation expressions or measure the relative importance of candidate triples within a sentence. It is also challenging to name the relation type of a relational triple merely based on context words, which could limit the usefulness of open RE in downstream applications. We propose a novel importance-based open RE approach by exploiting the global structure of a dependency tree to extract salient triples. We design an unsupervised method to name relation types by grounding relational triples to a large-scale Knowledge Base (KB) schema, leveraging KB triples and weighted context words associated with relational triples. Experiments on the English Slot Filling 2013 dataset demonstrate that our approach achieves 8.1% higher F-score over state-of-the-art open RE methods.

4.1 Motivations

open RE [93] aims at extracting relational triples from a open-domain corpus. Each triple contains two arguments and a phrase which denotes the relation between them. In this chapter, we focus on discovering relations between *entities*.

Most successful open RE approaches [43], [46], [94], [95] extract salient relational

Portions of this chapter previously appeared as: D. Yu, L. Huang, and H. Ji, "Open Relation Extraction and Grounding," in *Proc. 8th Int. Conf. Natural Language Process.,* Taipei, Taiwan, 2017, pp. 1-11.

triples based on lexical or syntactic patterns. However, such handcrafted or automatically learned patterns are incapable of covering diverse relation expressions [52]. Subsequently, the shortest path between arguments derived from a dependency tree has been widely applied to generate patterns to capture long-distance and complex relations. However, we usually require additional heuristic rules to filter out the resulting large number of meaningless patterns [41], [45], [46]. Besides, these flat syntactic structures lack the ability to measure the relative importance of candidate triples in a sentence. For example, the sentence in $E1$ places particular emphasis on the relation between *"she"*(*"Lucille Clifton"*) and *"Fred Clifton"* which therefore should be retained.

E1 *"In 1958 she married Fred Clifton, who taught philosophy at the University at Buffalo, eventually setting with him in Maryland"*

We notice that a candidate relational triple is likely to be salient if its two arguments are strongly connected in a dependency tree. Instead of relying on patterns to capture important triples, we use an importance-based strategy by exploring the entire dependency tree structure to automatically measure the connection strength of candidate argument pairs. Specifically, we assume that a relational triple is important if there is a relatively short random walk-based distance between two relatively important arguments, measured against the entire dependency tree of a given sentence. For each argument pair, we apply an effective random-walk based method to assign weights to context words in the sentence (Section 4.2).

How to assign a meaningful relation type name to a relational triple is also a primary challenge for open RE. Previous methods use relevant context words in the associated sentence as relation phrases [51], [95]. However, there is still no generally accepted guideline for relation phrase extraction. Multiple relation phrases can correspond to the same relation type. Besides, overly-specific or implicit relation phrases are incapable of providing adequate information for downstream applications. For example, the relation between *"Patricia"* and *"Gary Cooper"* cannot be clearly demonstrated by a set of words in the following sentence $E2$. Therefore, previous studies heavily rely on resources such as patterns [52], training data [55], or distantly-labeled corpora [47] to map open RE triples to a known relation schema.

E2 *"Patricia later described her relation with Gary Cooper as one of the most beautiful*

things that ever happed to her in her life."

Compared with a small number of predefined relation types such as those defined in Automatic Content Extraction [96], the relation schema in a large-scale Knowledge Base (KB) such as DBpedia [4] covers a much wider range of informative relations along with their type signatures. Considering the open-domain nature shared by open RE and a large-scale KB, we propose an unsupervised grounding method to name the relation type between two arguments as either a KB relation or none, by leveraging KB triples and weighted context information associated with each argument pair based on pre-trained word embeddings (Section 4.3). Compared with previous methods (*e.g.*, [55], [57]), we regard intra-sentence context words as intermediate results for the subsequent grounding process, and we do not require any aligned training corpora or relation phrases for KB triples. The proposed framework is illustrated in Figure 4.1.

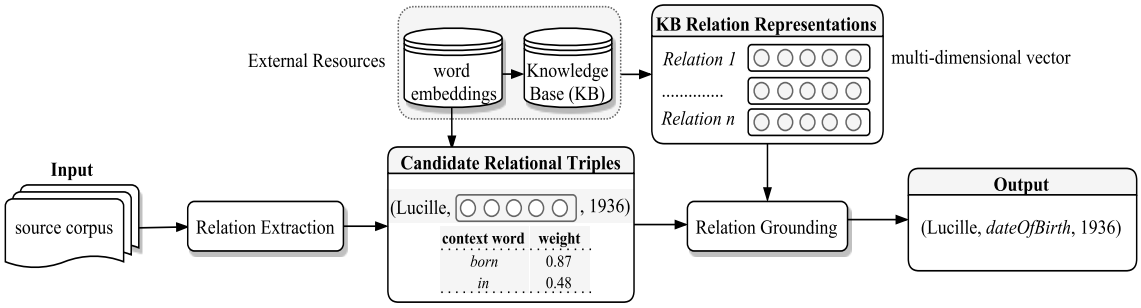

Figure 4.1: Framework overview.

To the best of our knowledge, this is the first open RE method which exploits the global structure of a dependency tree to extract salient relational triples. This is also the first unsupervised relation grounding method to name relation types for open RE based on KB triples and intra-sentence context information. Experiments on the English Slot Filling (SF) [26], [28] 2013 dataset demonstrate that our approach outperforms state-of-the-art open RE approaches.

4.2 Relation Extraction

In this section, we introduce a graph-based method to extract argument pairs of salient relational triples. We first present the extended dependency tree construction for each sentence (Section 4.2.1). Then we show the computation of the relation strength be-

tween two arguments (Section 4.2.4) considering both their random-walk based distance (Section 4.2.2) and the relative importance of each argument in the tree (Section 4.2.3).

4.2.1 Extended Dependency Tree Construction

Given a sentence containing N words, we construct a weighted directed graph $G = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \ldots, v_N\}$ represents words, and $\mathcal{E}$ is a directed edge set, associated with each directed edge $v_i \rightarrow v_j$ representing a dependency relation originating from v_i to v_j. We assign a weight $w_{ij} = 1$ to $v_i \rightarrow v_j$ and add its reverse edge $v_j \rightarrow v_i$ with $w_{ji} = 0.5$. By adding lower-weighted reverse edges, we can analyze the relation between two nodes which are not connected by directed dependency links while maintaining our preferences toward the original directions.

We first apply a dependency parser to generate basic uncollapsed dependencies.[2] We annotate an entity or time mention node with its type. For example in $E1$, "*Fred Clifton*" is annotated as a person, and "*1958*" is annotated as a date. Finally we perform coreference resolution which introduces coreference links between nodes that refer to the same entity within a document. We replace any nominal or pronominal entity mention with its coreferential name mention. For example, "*she*" is replaced by its full mention"*Lucile Clifton*". Formally, an extended dependency tree is an annotated tree of entity mentions and their links. By adding the reverse edges, we generate the final extended dependency tree in Figure 4.2. We regard any two entities as a candidate argument pair. $E1$ contains 5 entities and therefore we can extract $\binom{5}{2} = 10$ argument pairs (*e.g.*, (*"Fred Clifton"*, *"University of Buffalo"*)).

4.2.2 Distance Computation

As mentioned previously, a shorter distance between two strongly connected nodes is more likely to indicate the existence of an important relation. We compute the distance between two nodes based on a Markov-chain model of random walk. We define a random walk through G by assigning a transition probability to each directed edge. Thus, a random walker can jump from node v_i to v_j and represent a state of the Markov chain. For a node v_i, we denote $\mathcal{N}(i)$ as the set of its neighbors. The probability of transitioning from

[2] All the tools we used are introduced in Section 4.4.

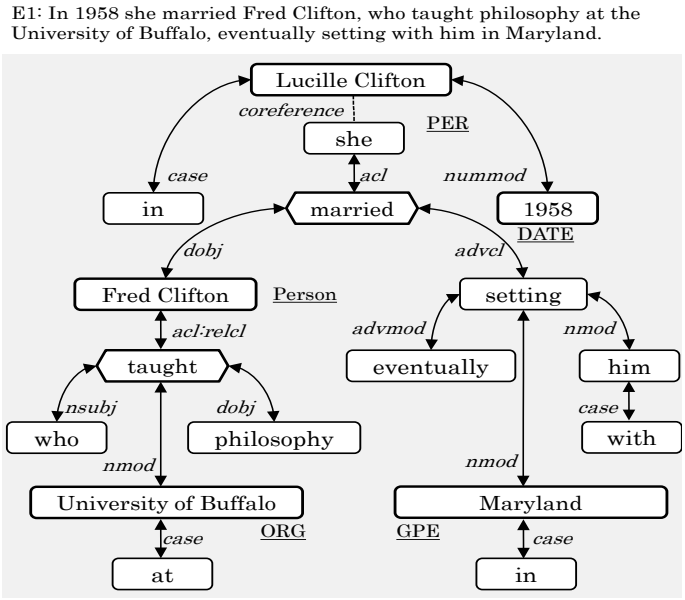

Figure 4.2: Extended dependency tree of E1.

node v_j to node v_i is defined as $p_{ji} = w_{ji} / \sum_{k \in \mathcal{N}(j)} w_{jk}$ for nodes v_i that have an edge from v_j to v_i, and 0 otherwise. We define the transition probability matrix of the Markov chain associated with random walks on G as $\boldsymbol{P}$.

The ***mean first-passage time*** m_{ji} [97] is the average number of steps needed by a random walker for reaching state i for the first time, when starting from state j. We call $c_{ij} = m_{ij} + m_{ji}$ as the ***average commute time*** [98]. The fact that c_{ij} can be regarded as a distance in G between nodes v_i and v_j is proven by [99]. Compared with the shortest path between v_i and v_j, the value of c_{ij} will decrease when the number of paths connecting v_i and v_j increases and when the length of any path decreases [100].

The fundamental matrix $\boldsymbol{Z}$ plays an essential role in computing various quantities related to random walks. For a weighted and directed graph, [101] demonstrate that $\boldsymbol{Z}$ can be computed directly using the following equation:

$$\boldsymbol{Z} = (\boldsymbol{I} - \boldsymbol{P} + \boldsymbol{E}\boldsymbol{D})^{-1} - \boldsymbol{E}\boldsymbol{D} \tag{4.1}$$

where $\boldsymbol{I}$ is the identity matrix, $\boldsymbol{E}$ is a matrix containing all 1s, and $\boldsymbol{D}$ is the diagonal

matrix with elements $d_{kk} = \pi(k)$ where $\pi(k)$ is the stationary distribution of node v_k in the Markov chain.

We can directly compute a mean first-passage $|\mathcal{V}| \times |\mathcal{V}|$ matrix and a symmetric average commute time matrix $\boldsymbol{C}$ based on $\boldsymbol{Z}$ as follows:

$$m_{ij} = \frac{z_{jj} - z_{ij}}{\pi_j} \tag{4.2}$$

$$c_{ij} = \frac{z_{jj} - z_{ij}}{\pi_j} + \frac{z_{ii} - z_{ji}}{\pi_i} \tag{4.3}$$

Using the example in Figure 4.2, we can obtain a 16×16 matrix $\boldsymbol{M}$ based on the above steps (16 nodes in total). We list the result involving only entity nodes in Table 4.1.

Table 4.1: Mean first-passage time matrix M for E1 ($\star$ refers to Lucille Clifton).

$m(row, col)$	she *	1958	Fred Clifton	University of Buffalo	Maryland
she *	0.0	22.9	35.8	259.0	182.8
1958	0.3	0.0	36.1	261.4	183.7
Fred Clifton	125.2	54.2	0.0	156.2	**282.3**
University of Buffalo	148.8	60.1	46.6	0.0	353.1
Maryland	100.1	47.9	**43.9**	323.5	0.0

We notice that argument roles can be identified based on the mean first-passage time. In a weighted directed graph, m_{ij} and m_{ji} are not necessarily similar. Actually in many cases nodes that lie on the boundaries have shorter mean first-passage time to the central nodes in the graph while there exists longer mean first-passage time from a central node to a node close to the boundary. A central node is more likely to be the central argument. We define the first argument as the more important argument. Therefore, we can regard v_i as the first argument of the argument pair (v_i, v_j) if m_{ij} is larger than m_{ji}. If m_{ij} and m_{ji} are equal, v_i and v_j have similar argument roles. For example, the boundary node *"Maryland"* in Figure 4.2 has shorter first-passage time to the central node *"Fred Clifton"* (*i.e.*, the first argument) compared with the reverse direction.

4.2.3 Node Importance Computation

As we mentioned earlier, a candidate relational triple is more likely to be salient if it involves important entities of the sentence. In this section, we illustrate the node importance computation based on the extended dependency tree of a sentence.

TextRank [61] can be used to compute the importance of each node within G. Similarly, suppose a random walker keeps visiting adjacent nodes in G at random. The expected percentage of walkers visiting each node converges to the TextRank score.

We define a set of preferred nodes $\mathcal{R}$ which correspond to entities in a sentence. We assign higher preferences toward these nodes when computing the importance scores since entities are more informative for relation extraction [102]. We extend TextRank by introducing a new measure called "back probability" $d \in [0, 1]$ to determine how often walkers jump back to the nodes in $\mathcal{R}$ so that the converged score can be used to estimate the relative probability of visiting these preferred nodes. We define a preference vector $\boldsymbol{p}_{\mathcal{R}} = \{p_1, ..., p_{|\mathcal{V}|}\}$ such that the probabilities sum to 1, and p_k denotes the relative importance attached to v_k. p_k is set to $1/|\mathcal{R}|$ for $v_k \in \mathcal{R}$, otherwise 0. Let I be the $1 \times |\mathcal{V}|$ importance vector to be computed over all nodes as follows.

$$I(i) = (1 - d) \sum_{j \in \mathcal{N}(i)} \frac{w_{ji}}{\sum_{k \in \mathcal{N}(j)} w_{jk}} I(j) + d \cdot p_i \tag{4.4}$$

Table 4.2: Importance score of each entity in $E1$.

ENTITY i	University of Buffalo	Fred Clifton	Lucille	1958	Maryland
$I(i)$	0.165	0.129	0.079	0.015	0.004

4.2.4 Combination and Filtering

Given the average commute time c_{ij} between nodes v_i and v_j (Section 4.2.2) and their relative importance scores $I(i)$ and $I(j)$ in G (Section 4.2.3), we will discuss how to combine them and generate the final score which can be used to measure the relation strength between two nodes. Intuitively, there exists a strong relation when there is a shorter distance between two relatively important nodes.

Previous approaches [103], [104] consider the distance between two nodes and the influence of each node modeled by its weighted frequency to measure the strength of links in networks. Similarly, in our setting we can regard c_{ij} as the distance between v_i and v_j and use the relative importance score to measure the influence of each node in G. Therefore, we obtain Equation 4.5 to compute the relation strength $F(i,j)$ between nodes v_i and v_j. We are more confident in predicting the existence of a salient relation with stronger relation strength.

$$F(i,j) = \frac{I(i) \times I(j)}{c_{ij}^2} \tag{4.5}$$

We get a complete entity graph since we analyze the connection between any two entities in a sentence. In this work, we mainly aim to identify the most significant structures among entities based on the connection strength we have obtained. Since the entity graph is undirected, we can simply apply the maximum spanning tree algorithm to keep those relatively important pairs. For $E1$, we get 4 argument pairs resulting after filtering: (*"Lucille"*, *"1958"*), (*"Fred Clifton"*, *"University of Buffalo"*), (*"Lucille"*, *"Fred Clifton"*), and (*"Fred Clifton"*, *"Maryland"*). In comparison, the relations between argument pairs such as (*"University of Buffalo"*, *"1958"*) and (*"Maryland"*, *"1958"*) are less important.

4.3 Relation Grounding

We have presented how to extract candidate argument pairs in Section 4.2. In this section, we first introduce how to rank the context words given a pair of arguments (Section 4.3.1). Then we describe the KB relation representation learning from existing KB triples based on pretrained word embeddings. Finally we ground each relational triple to a KB relation or none (Section 4.3.2).

4.3.1 Context Word Selection and Weighting

In this section, we introduce how to extract informative context words and their associated weights given an argument pair (v_i, v_j) in a sentence based on the average

commute time matrix $\mathbf{C}$ introduced in Section 4.2.2. Previous work [33] regards this problem as finding important nodes in G relative to given arguments. However, they need to run the algorithm repeatedly to analyze the same graph for each argument pair. Here we discuss an efficient method to extract weighted context words.

We only keep nouns, verbs, adjectives, prepositions, and particles as indicative context words $\mathcal{X}$. We assume that a context word $v_k \in \mathcal{X}$ is more important relative to (v_i, v_j) if $c_{ik} + c_{kj}$ is close to c_{ij}. Actually if the relation between v_i and v_j does not rely on any indicative words, c_{ij} will be much smaller than $c_{ik} + c_{kj}$ considering other nodes in the same sentence. We denote Λ as the weight set for all the context words of a given argument pair (v_i, v_j) as follows. The higher λ_k is, the more important the context word v_k is relative to (v_i, v_j).

$$\lambda_k = \frac{c_{ij}}{c_{ik} + c_{kj}} \tag{4.6}$$

In $E1$, given the argument pair (*Lucille Clifton*, *Fred James Clifton*), we generate the following weighted context words: $\{$*married* : 0.60, *in*^{1} : 0.36, *born* : 0.29, *in*^{2} : $0.24\}$.

4.3.2 Grounding

The associated weighted context words of each candidate argument pair are not sufficiently informative and flexible to clearly express the relation between two arguments. Thus, we aim to name the relation between a pair of arguments as one of the KB relations or none by comparing the semantic representations of context words and KB relations based on word embeddings. We also learn argument type signatures from KB triples.

For each word we obtain its pretrained word embedding $\boldsymbol{e} \in \mathbb{R}^k$ where k is the embedding dimensionality. For a phrase which contains multiple words, we simply average the vectors of all the single words in the phrase as its embedding.

Given a KB triple (h, l, t) composed of two entities h, t and a KB relation $l \in \mathcal{L}$ (the set of KB relations), we leverage a large-scale KB to learn the representation for each KB relation motivated by the basic idea behind previous studies [105], [106] that relation patterns can be represented as linear translations. We use $\mathcal{S}_l = \{(h_i, l, t_i), i = 1, \ldots, |\mathcal{S}_l|\}$

to represent all the KB triples with the KB relation l.

KB relation names can also provide important semantic information for relation representation and disambiguation especially for those relations which tend to co-occur in the same sentence, such as family relations (*e.g.*, *spouse*, *parents*, and *other family*). We segment a compound name of a KB relation into a set of words. For example, we separate a DBpedia relation *politicalGroups* into $\{$*political*, *groups*$\}$. Similarly, we average the vectors of all the words in the relation name as its embedding $\widetilde{\boldsymbol{e}}_l \in \mathbb{R}^k$. Incorporating both the KB tuples and KB relation names, we represent the relation embedding of l as follows.

$$\boldsymbol{e}_l = \frac{1}{|\mathcal{S}_l|} \sum_{i=1}^{|\mathcal{S}_l|} (\boldsymbol{e}_{h_i} - \boldsymbol{e}_{t_i} + \widetilde{\boldsymbol{e}}_l) \tag{4.7}$$

Given a single KB relation l, an argument pair (v_i, v_j) and a single context word x, we can compute the cosine similarity between any candidate open RE triple and any KB relation. We calculate the absolute value since we have already dealt with the direction of arguments in Section 4.2.2. Therefore, we can regard similarity scores -1 and 1 equally and 0 as the lowest score.

$$S(l, (i, j, x)) = \frac{|\boldsymbol{e}_x \cdot \boldsymbol{e}_l|}{\|\boldsymbol{e}_x\| \|\boldsymbol{e}_l\|} \tag{4.8}$$

When there are multiple context words $x \in \mathcal{X}$, we can compute the weighted cosine similarity between them as follows based on the squared weights of context words described in Section 4.3.1.

$$S(l, (i, j, \mathcal{X})) = \max_{x \in \mathcal{X}} S(l, (i, j, x)) \times \lambda_x^2 \tag{4.9}$$

Since we have multiple KB relations $l \in \mathcal{L}$, we can ground a candidate relational triple $(i, j, \mathcal{X})$ and obtain its relation $\widehat{l}_{i,j,\mathcal{X}}$ considering all the possible relations. The predicted relation can either be assigned a valid KB relation or None. We use a marker to

denote the relation between v_i and v_j which cannot be grounded to any KB relation.

$$\widehat{l}_{i,j,\mathcal{X}} = \arg\max_{l \in \mathcal{L}} S(l, (i, j, \mathcal{X})) \tag{4.10}$$

4.3.3 Relation Argument Type Constraints

For each KB relation, we can obtain its type constraints for its two arguments. Take the relation *birthPlace* as an example: the entity types of the two arguments should be a person name and a location name.

Given all the KB triples, we can estimate the probability of one of the arguments belonging to a certain entity type $z \in \mathcal{Z}$, where $\mathcal{Z}$ represents the set of all the KB concept types. For a given KB relation l, we define $c(k \rightsquigarrow z \,|\, l)$ to be the number of times the k_{th} argument is seen paired with the entity type z where $k \in \{1, 2\}$ since there are two arguments. Given these definitions, the maximum likelihood estimate is as follows.

$$p(k, z \,|\, l) = \frac{c(k \rightsquigarrow z \,|\, l)}{\sum_{z \in \mathcal{Z}} c(k \rightsquigarrow z \,|\, l)} \tag{4.11}$$

Therefore, given a candidate argument pair (v_i, v_j) and their entity types z_i and z_j, we can compute the probability of its being labeled as the relation l by considering both $p(1, z_i \,|\, l)$ and $p(2, z_j \,|\, l)$. We set $S(l, (i, j, \mathcal{X}))$ to 0 if the harmonic mean of $p(1, z_i \,|\, l)$ and $p(2, z_j \,|\, l)$ is smaller than a given threshold which will be introduced later in Section 4.4.4. We will not consider a candidate KB relation for comparison if the argument type of i or j fails to satisfy its type constraints. In this way, we can filter out some candidate triples and reduce the number of similarity computations. For example, given a KB relation *placeOfBurial*, the concept type *Species* is less likely to be the correct second argument type compared with other entity types such as *City* and *Location*. Remind that the order of arguments in the candidate triple has been introduced in Section 4.2.2.

4.4 Experiments

4.4.1 Knowledge Base and Word Embeddings

We use the April 2016 dump of DBpedia as our KB which contains 2,060 relations and 30,024,093 relation triples in total. We use the 300-dimensional GloVe vectors [107] pretrained on 6 billion tokens from the English Gigaword Fifth Edition and a 2014 Wikipedia dump.

4.4.2 Evaluation Based on Slot Filling

There are several benchmarks developed for open RE (*e.g.*, [43], [108]). However, we mainly focus on relations between entities and therefore we cannot directly compare with state-of-the-art open RE methods on those datasets. To evaluate the effectiveness of our approach, we choose the TAC-KBP SF [26]–[29] task as our evaluation platform which has been widely used by open RE methods [47], [52] since 2009.

We use the SF 2013 dataset for which we can compare with the ground truth and state-of-the-art open RE results reported in SF. We obtain 1,701 relevant documents from the official evaluation assessment for 50 person queries and 50 organization queries. We manually map KB relations to slot types based on TAC-KBP slot descriptions. Note that a single KB relation can be mapped to multiple slot types. For example, *birthPlace* can be mapped to *per:city_of_birth*, *per:stateofprovince_of_birth*, and *per:country_of_birth*. We assign the subtype (*i.e.*, country, province, or city) to a location entity based on collected geographical gazetteers.

Table 4.3: Example mappings from DBpedia relations to slot types.

DBpedia Relations	**Slot Types**
founder	org:founded_by
keyPeople	org:top_members_employees
education	per:schools_attended
workInstitution	per:employee_or_member_of
birthDate	per:date_of_birth

We ignore all the slot types which require nominal phrases as fillers (*e.g.*, *per:cause_of_death*) and slot types *per/org:alternate_names* which depend on cross-document coreference resolution. We apply Stanford CoreNLP [88] for English part-of-speech tagging, name tag-

ging, time expression extraction, dependency parsing, and coreference resolution. We use the official Slot Filling evaluation scoring metrics: Precision (P), Recall (R), and F-score (F_1).

Table 4.4: Performance (%) on KBP2013 English SF based on different relation representations.

Method		**P**	**R**	**F_1**
UW Official [52]		**69.9**	12.2	20.8
UMass Official [109]		10.6	19.5	13.7
Our Approach	[1] KB Tuples	17.3	21.1	19.0
	[2] Relation Names	24.3	30.9	27.2
	[1]+[2] Joint	26.2	**32.4**	**28.9**

As shown in Table 4.4, our method outperforms the KBP2013 SF submission from the University of Washington [52] which applies Open IE V4.0, which is an extension of SRL-based IE [110] and noun phrase processing [111], to generate relation triples. This is their latest published approach which uses Open IE for Regular Slot Filling. Their approach achieves very high precision but comparatively low recall (12.2%). In our experiments, we keep all the candidate triples which could be mapped to a slot type without tuning thresholds. On the same dataset, We also compare with a approach [109] which extracts relations with matrix factorization and *universal schemas* [57], which are made up of textual patterns and all the slot types. We do not directly compare with the work of [47] because of the lack of access to their SF output.[3]

The importance-based strategy is useful for us to extract more salient information. For example, previous methods only extract one argument pair (*"the top Egyptian cleric"*, *"Wednesday"*) from the sentence *"**Sheikh Tantawi**, the top **Egyptian** cleric who died on **Wednesday** on a visit to . . . "* while omitting the person name. Our method extracts both (*"Sheikh Tantawi"*, *"Egyptian"*) and (*"Sheikh Tantawi"*, *"Wednesday"*) with their associated top-weighted context words *"cleric"* and *"died"* respectively, since the strength between *"Egyptian"* and *"Wednesday"* is much weaker.

Compared with relation phrases, the word embeddings of weighted context words are more flexible for comparison when we map relational triples to a known schema.

[3]The highest recall they achieve is around 13% on all the slot types including nominal relations on the same dataset.

For example, it is impossible for previous methods [52] to summarize all the related mentions (*e.g.*, *"appointed"* and *"CEO"*) and manually map them to the relation *employment*. Therefore previous approaches missed the slot filler *"Al-Azhar University"* of the query *"Mohammed Sayed Tantawi"* from the following sentence *"Tayeb, the president of* ***Al-Azhar University*** *since 2003, succeeds* ***Mohammed Sayed Tantawi****"* as *"succeeds"* was not included into the related terms. Our approach extracts it based on their semantic representations.

In addition, we obtain more generalized relation names based from the KB schema. For example, we ground the relation in $E2$ between *"Patricia"* and *"Gary"* to *influencedBy*. Similarly, in the sentence *"****Ginzburg*** *shared the Nobel Physics Prize with US physicists* ***Alexei Abrikosov*** *and Anthony Leggett for their contributions to the theory of superconductors ..."*, the relation phrase *"shared the Nobel Physics Prize with"* between *"Ginzburg"* and *"Alexei"* is too specific compared with the grounded KB relation *alongside* by our approach for subsequent applications.

4.4.3 Impact of Relation Representations

In Section 4.3.2, we use KB tuples and their relation names to learn KB relation representations. As shown in Table 4.4, our approach can already achieve promising performance based on the relation representations learned from KB relation names. However, sometimes relations are implicitly expressed. It is likely that the context words of a relation triple and its corresponding KB relation name are not semantically similar. In this case, we need more general relation representations with the help of millions of KB tuples. For example, we can ground the relation *school* between *"McGregor"* and *"Colorado State University"* successfully by comparing the representation of context words *"tight"* and *"end"* with the joint relation representations from the following sentence: *"****McGregor*** *was a two-time All-America tight end at* ***Colorado State University****"* even though this relation is not explicitly described.

4.4.4 Impact of Argument Type Constraints

As mentioned in Section 4.3.2, we aim to filter out some candidate relation triples if the entity types of the arguments are not popular for a given KB relation. By tuning

thresholds, there are no significant differences in performance when the threshold falls in the range 0.05–0.2. On the other hand, if the threshold is set too high (*e.g.*, greater than 0.35%), we will mistakenly discard correct candidates which satisfy type constraints.

We implement Jenks optimization [112] to automatically split the frequency values of all entity types into two tiers given a certain argument position and a KB relation. This is done by minimizing each tier's average deviation from the tier mean, while maximizing each tier's deviation from the means of other groups [113]. We automatically set the threshold automatically using the obtained natural breaks for two arguments respectively to compute the harmonic mean of them. This approach achieves 28.9% F_1 which is comparable to the highest F_1 (29.2%) obtained by threshold tuning.

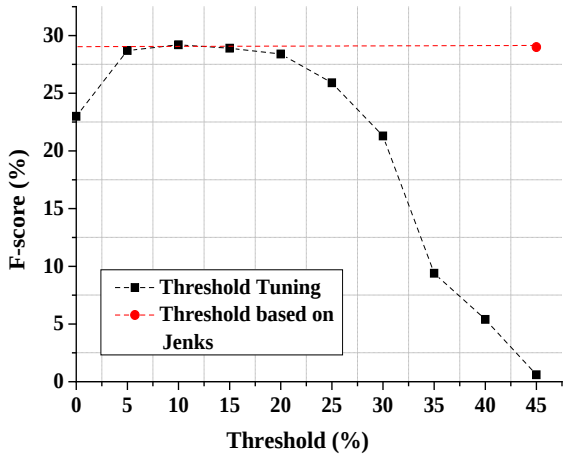

Figure 4.3: Performance (%) based on different thresholds for argument type constraints.

4.5 Summary

We propose an unsupervised open relation extraction method by exploring the global structure of dependency tree and show its effectiveness in extracting salient candidate relation triples. We also leverage the knowledge from the large-scale KB relation triples and weighted context words based on general embeddings to enhance the quality of our relation grounding technique. Experiments on English Slot Filling demonstrate that our approach outperforms state-of-the-art open RE approaches. In the future, we aim to extend our framework for multilingual open RE based on the KB schema. We are also interested in integrating fine-grained phrase typing techniques to improve the capability

of our methods.

CHAPTER 5
Unsupervised Relation Validation: Multi-Dimensional Truth-Finding Model

The consolidation of SF responses extracted by multiple SF systems from multiple information sources may generate erroneous, conflicting, redundant or complement results. Therefore, it poses a challenge but also an opportunity to Slot Filling Validation.

5.1 Approach Overview

In this chapter, we call a combination of query entity, slot type, and slot filler a ***claim*** for brevity. Along with each claim, each system must provide the ID of a source document and one or more detailed context sentences as ***evidence*** which supports the claim. A ***response*** (*i.e.*, a claim, evidence pair) is *correct* if and only if the claim is true *and* the evidence supports it.

Given the responses produced by multiple systems from multiple sources in the SF task, the goal of the SFV task is to determine whether each response is true or false. Though it's a promising line of research, it raises two complications: (1) different information ***sources*** may generate claims that vary in trustability; and (2) a large-scale number of SF ***systems*** using different resources and algorithms may generate erroneous, conflicting, redundant, complementary, ambiguously worded, or inter-dependent claims from the same set of documents. Table 5.1 presents responses from four SF systems for the query entity ***Ronnie James Dio*** and the slot type ***per:city_of_death***. Systems A, B and D return *Los Angeles* with different pieces of evidence [4] extracted from different information

Portions of this chapter previously appeared as: D. Yu, H. Li, T. Cassidy, Q. Li, H. Huang, Z. Chen, H. Ji, Y. Zhang, and D. Roth, "RPI-BLENDER TAC-KBP2013 knowledge base population system," in *Proc. Text Anal. Conf.,* Gaithersburg, MD, 2013.

Portions of this chapter previously appeared as: D. Yu, H. Huang, T. Cassidy, H. Ji, C. Wang, S. Zhi, J. Han, C. Voss, and M. Magdon-Ismail, "The wisdom of minority: unsupervised slot filling validation based on multi-dimensional truth-finding," in *Proc. 25th Conf. Computational Linguistics,* Dublin, Ireland, 2014, pp. 1567-1578.

Portions of this chapter previously appeared as: D. Yu and H. Ji, "Unsupervised Person Slot Filling based on graph mining," in *Proc. 54th Annu. Meeting Assoc. Computational Linguistics,* Berlin, German, 2016, pp. 44-53.

[4]Hereafter, we refer to "pieces of evidence" with the shorthand "evidences". Note that SF systems may

sources, though the evidence of System D does not decisively support the claim. System C returns *Atlantic City*, which is neither true nor supported by the corresponding evidence.

Such complications call for "*truth finding*": determining the *veracity* of multiple conflicting claims from various sources and systems. We propose a novel unsupervised multi-dimensional truth finding framework to study credibility perceptions in rich and wide contexts. It incorporates signals from multiple sources and systems, using linguistic indicators derived from extended dependency graphs constructed from multiple evidences using multi-layer deep linguistic analysis. Experiments demonstrate that our approach can find truths accurately (9.4% higher F-score than supervised methods) and efficiently (find 90% truths with only one half cost of a baseline without credibility estimation).

Table 5.1: Conflicting responses across different SF systems and different sources (query entity = Ronnie James Dio, slot type = per:city_of_death).

System	Source	Slot Filler	Evidence
A	Agence France-Presse, News	Los Angeles	The statement was confirmed by publicist Maureen O'Connor, who said ***Dio*** died in ***Los Angeles***.
B	New York Times, News	Los Angeles	***Ronnie James Dio***, a singer with the heavy-metal bands Rainbow, Black Sabbath and Dio, whose semioperatic vocal style and attachment to demonic imagery made him a mainstay of the genre, died on Sunday in ***Los Angeles***.
C	Discussion Forum	Atlantic City	***Dio*** revealed last summer that he was suffering from stomach cancer shortly after wrapping up a tour in ***Atlantic City***.
D	Associated Press Worldstream, News	Los Angeles	***LOS ANGELES*** 2010-05-16 20:31:18 UTC ***Ronnie James Dio***, the metal god who replaced Ozzy Osbourne in Black Sabbath and later piloted the bands Heaven, Hell and Dio, has died, according to his wife and manager.

5.2 MTM: A Multi-Dimensional Truth-Finding Model

5.2.1 MTM Construction

A response is trustworthy if its claim is true and its evidence supports the claim. A trusted source always supports true claims by giving convincing evidence, and a good

include multiple sentences as "evidence" within their responses.

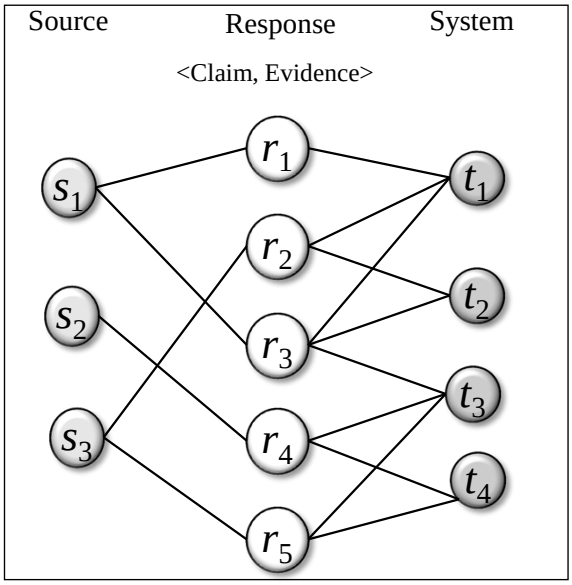

Figure 5.1: Heterogeneous networks for MTM.

system tends to extract trustworthy responses from trusted sources. We propose a *multi-dimensional truth-finding model (MTM)* to incorporate and compute multi-dimensional credibility scores.

Consider a set of responses $R = \{r_1, \ldots, r_m\}$ extracted from a set of sources $S = \{s_1, \ldots, s_n\}$ and provided by a set of systems $T = \{t_1, \ldots, t_l\}$. A heterogeneous network is constructed as shown in Fig. 5.1. Let weight matrices be $W^{rs}_{m\times n} = \{w^{rs}_{ij}\}$ and $W^{rt}_{m\times l} = \{w^{rt}_{ik}\}$. A link $w^{rs}_{ij} = 1$ is generated between r_i and s_j when response r_i is extracted from source s_j, and a link $w^{rt}_{ik} = 1$ is generated between r_i and t_k when response r_i is provided by system t_k.

5.2.2 Credibility Initialization

Each source is represented as a combination of publication venue and genre. The credibility scores of sources S are initialized uniformly as $\frac{1}{n}$, where n is the number of sources. Given the set of systems $T = \{t_1, \ldots, t_l\}$, we initialize their credibility scores $c^0(t)$ based on their interactions on the predicted responses. Suppose each system t_i generates a set of responses R_{t_i}. The similarity between two systems t_i and t_j is defined as $similarity(t_i, t_j) = \frac{|R_{t_i} \cap R_{t_j}|}{\log(|R_{t_i}|)+\log(|R_{t_j}|)}$ [114]. Then we construct a weighted undirected graph $G = \langle T, E\rangle$, where $T(G) = \{t_1, \ldots, t_l\}$ and $E(G) = \{\langle t_i, t_j\rangle\}$,

$\langle t_i, t_j \rangle = similarity(t_i, t_j)$, and apply the TextRank algorithm [114] on G to obtain $c^0(t)$.

We got negative results by initializing system credibility scores uniformly. We also got negative results by initializing system credibility scores using system metadata, such as the algorithms and resources the system used at each step, its previous performance in benchmark tests, and the confidence values it produced for its responses. We found the quality of an SF system depends on many different resources instead of any dominant one. For example, an SF system using a better dependency parser does not necessarily produce more truths. In addition, many systems are actively being improved, rendering previous benchmark results unreliable. Furthermore, most SF systems still lack reliable confidence estimation.

The initialization of the credibility scores for responses relies on deep linguistic analysis of the evidence sentences and the exploitation of semantic clues, which will be described in Section 5.3.

5.2.3 Credibility Propagation

We explore the following heuristics in MTM.

HEURISTIC 1: *A response is more likely to be true if derived from many trustworthy sources. A source is more likely to be trustworthy if many responses derived from it are true.*

HEURISTIC 2: *A response is more likely to be true if it is extracted by many trustworthy systems. A system is more likely to be trustworthy if many responses generated by it are true.*

Based on these two heuristics we design the following credibility propagation approach to mutually reinforce the trustworthiness of linked objects in MTM.

By extension of Co-HITS [115], designed for bipartite graphs, we develop a propagation method to handle heterogeneous networks with three types of objects: *source, response* and *system*. Let the weight matrices be W^{rs} (between responses and sources) and W^{rt} (between responses and systems), and their transposes be W^{sr} and W^{tr}. We

can obtain the transition probability that vertex s_i in S reaches vertex r_j in R at the next iteration, which can be formally defined as a normalized weight $p_{ij}^{sr} = \frac{w_{ij}^{sr}}{\sum_k w_{ik}^{sr}}$ such that $\sum_{r_j \in R} p_{ij}^{sr} = 1$. We compute the transition probabilities p_{ji}^{rs}, p_{jk}^{rt} and p_{kj}^{tr} in an analogous fashion.

Given the initial credibility scores $c^0(r)$, $c^0(s)$ and $c^0(t)$, we aim to obtain the refined credibility scores $c(r)$, $c(s)$ and $c(t)$ for responses, sources, and systems, respectively. Starting with sources, the update process considers both the initial score $c^0(s)$ and the propagation from connected responses, which we formulated as:

$$c(s_i) = (1 - \lambda_{rs})c^0(s_i) + \lambda_{rs} \sum_{r_j \in R} p_{ji}^{rs} c(r_j) \tag{5.1}$$

Similarly, the propagation from responses to systems is formulated as:

$$c(t_k) = (1 - \lambda_{rt})c^0(t_k) + \lambda_{rt} \sum_{r_j \in R} p_{jk}^{rt} c(r_j) \tag{5.2}$$

Each response's score $c(r_j)$ is influenced by both linked sources and systems:

$$c(r_j) = (1 - \lambda_{sr} - \lambda_{tr})c^0(r_j) + \lambda_{sr} \sum_{s_i \in S} p_{ij}^{sr} c(s_i) + \lambda_{tr} \sum_{t_k \in T} p_{kj}^{tr} c(t_k) \tag{5.3}$$

where $\lambda_{rs}, \lambda_{rt}, \lambda_{sr}$ and $\lambda_{tr} \in [0, 1]$. These parameters control the preference for the propagated over initial score for every type of random walk link. The larger they are, the more we rely on link structure[5]. The propagation algorithm converges (10 iterations in our experimental settings) and a similar theoretical proof to HITS [116] can be constructed. Algorithm 1 summarizes MTM.

5.3 Response Credibility Initialization

Each evidence along with a claim is expressed as a few natural language sentences that include the query entity and the slot filler, along with semantic content to support the claim. We analyze the evidence of each response in order to initialize that response's

[5]We set $\lambda_{rs} = 0.9$, $\lambda_{sr} = 0.1$, $\lambda_{rt} = 0.3$ and $\lambda_{tr} = 0.2$, optimized from a development set. See Section 5.4.1.

Input: A set of responses (R), sources (S) and systems (T).
Output: Credibility scores ($c(r)$) for R.

Initialize the credibility scores $c^0(s)$ for S as $c^0(s_i) = \frac{1}{|S|}$;
Use TextRank to compute initial credibility scores $c^0(t)$ for T;
Initialize the credibility scores $c^0(r)$ using linguistic indicators (Section 5.3);
Construct heterogeneous networks across R, S and T;
$k \leftarrow 0$, diff $\leftarrow 10e6$;
while $k <$ MaxIteration and diff $>$ MinThreshold **do**
    Use Eq. (5.1) to compute $c^{k+1}(s)$;
    Use Eq. (5.2) to compute $c^{k+1}(t)$;
    Use Eq. (5.3) to compute $c^{k+1}(r)$;
    Normalize $c^{k+1}(s)$, $c^{k+1}(t)$, and $c^{k+1}(r)$;
    diff $\leftarrow \sum(|c^{k+1}(r) - c^k(r)|)$;
    $k \leftarrow k + 1$
end while

Algorithm 1: Multi-dimensional Truth-Finding.

credibility score. This is done using heuristic rules defined in terms of the binary outputs of various *linguistic indicators* (Section 5.3.1).

5.3.1 Linguistic Indicators

We encode linguistic indicators based on deep linguistic knowledge acquisition and use them to determine whether responses provide supporting clues or carry negative indications (Section 5.3.3). These indicators make use of linguistic features on varying levels - surface form, sentential syntax, semantics, and pragmatics - and are defined in terms of extended dependency graphs (Section 5.3.2). We define a heuristic rule for each slot type in terms of the binary-valued linguistic indicator outputs to yield a single binary value (1 or 0) for each response. If a response's linguistic indicator value is 1, the credibility score of a response is initialized at 1.0, and 0.5 otherwise.

5.3.2 Extended Dependency Graph Construction

A semantically rich dependency graph is constructed that links a query entity, all of its relevant slot filler nodes, and nodes for other intermediate elements excerpted from evidence sentences. There is one graph per sentence.

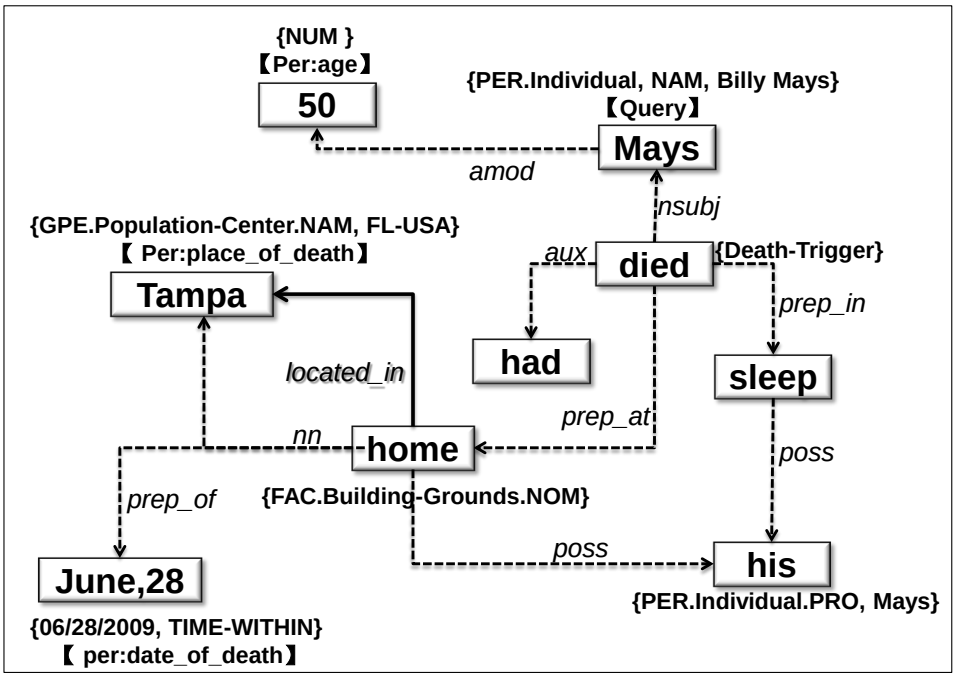

Figure 5.2: Extended dependency graph example.

Fig. 5.2 shows a subregion of the graph built from the sentence: "***Mays***, ***50***, *died in his sleep at his* ***Tampa*** *home the morning of* ***June 28***.". It supports 3 claims: [*Mays, per: city_of_death, Tampa*], [*Mays, per: date_of_death, 06/28/2009*] and [*Mays, per: age, 50*].

Formally, an extended dependency graph is an annotated graph of entity mentions, phrases and their links. It must contain one query entity node and one or more slot filler nodes. The annotation of a node includes its entity type, subtype, mention type, referent entities, and semantic category (though not every node has each type of annotation). The annotation of a link includes a dependency label and/or a semantic relation between the two linked nodes.

The extended dependency graph is constructed using the following procedure. First, we annotate the evidence text using dependency parsing [117] and Information Extraction (entity, relation and event) [118], [119]. Two nodes are linked if they are deemed related by one of the annotation methods (*e.g.*, [*Mays, 50*] is labeled with the dependency type *amod*, and [*home, Tampa*] is labeled with the semantic relation *located_in*). The annotation output is often in terms of syntactic heads. Thus, we extend the boundaries of entity, time, and value mentions (*e.g.*, people's titles) to include an entire phrase where possible.

We then enrich each node with annotation for entity type, subtype and mention type. Entity type and subtype refer to the role played by the entity in the world, the latter being more fine-grained, whereas mention type is syntactic in nature (it may be pronoun, nominal, or proper name). For example, "*Tampa*" in Fig. 5.2 is annotated as a *Geopolitical (entity type) Population-Center (subtype) Name (mention type)* mention. Every time expression node is annotated with its normalized reference date (*e.g.*, "*June, 28*" in Fig. 5.2 is normalized as "*06/28/2009*").

Second, we perform co-reference resolution, which introduces implicit links between nodes that refer to the same entity. Thus, an entity mention that is a nominal or pronoun will often be co-referentially linked to a mention of a proper name. This is important because many queries and slot fillers are expressed only as nominal mentions or pronouns in evidence sentences, their canonical form appearing elsewhere in the document.

Finally, we address the fact that a given relation type may be expressed in a variety of ways. For example, "*the face of*" indicates the membership relation in the following sentence: "*Jennifer Dunn was the face of the Washington state Republican Party for more than two decades*." We mined a large number of trigger phrases for each slot type by mapping various knowledge bases, including Wikipedia Infoboxes, Freebase [120], DBPedia [4] and YAGO [2], into the Gigaword corpus [121] and Wikipedia articles via distant supervision [18][6]. Each intermediate node in the extended dependency graph that matches a trigger phrase is then assigned a corresponding semantic category. For example, "*died*" in Fig. 5.2 is labeled a *Death-Trigger*.

5.3.3 Graph-Based Verification

We design linguistic indicators in terms of the properties of nodes and paths that are likely to be bear on the response's veracity. Formally, a *path* consists of the list of nodes and links that must be traversed along a route from a query node to a slot filler node.

Node indicators contribute information about a query entity or slot filler node in isolation, that may bear on the trustworthiness of the containing evidence sentence. For instance, a slot filler for the ***per:date_of_birth*** slot type must be a time expression.

[6]Under the distant supervision assumption, sentences that appear to mention both entities in a binary relation contained in the knowledge base were assumed to express that relation.

Node indicators are shown as follows.

1. *Surface*: Whether the slot filler includes stop words; whether it is lower cased but appears in news. These serve as negative indicators.

2. *Entity type, subtype and mention type*: For example, the slot fillers for "*org:top_employees*" must be person names; and fillers for "*org:website*" must match the url format. Besides the entity extraction system, we also exploited the entity attributes mined by the NELL system [122] from the KBP source corpus.

Each path contains syntactic and/or semantic relational information that may shed light on the manner in which the query entity and slot filler are related, based on dependency parser output, IE output, and trigger phrase labeling. Path indicators are used to define properties of the context in which which query-entity and slot-filler are related in an evidence sentence. For example, whether the path associated with a claim about an organization's top employee includes a title commonly associated with decision-making power can be roughly represented using the *trigger phrases* indicator.

Path indicators are shown as follows.

1. *Trigger phrases*: Whether the path includes any trigger phrases as described in Section 5.3.2.

2. *Relations and events*: Whether the path includes semantic relations or events indicative of the slot type. For example, a "*Start-Position*" event indicates a person becomes a "*member*" or "*employee*" of an organization.

3. *Path length*: Usually the length of the dependency path connecting a query node and a slot filler node is within a certain range for a given slot type. For example, the path for "*per:title*" is usually no longer than 1. A long dependency path between the query entity and slot filler indicates a lack of a relationship. In the following evidence sentence, which does not entail the "*per:religion*" relation between "*His*" and the religion "*Muslim*", there is a long path ("*his-poss-moment-nsubj-came-advcl-seized-militant-acmod-Muslim*"): "*His most noticeable moment in the*

public eye came in 1979, when Muslim militants in Iran seized the U.S. Embassy and took the Americans stationed there hostage.".

Detecting and making use of interdependencies among various claims is another unique challenge in SFV. After initial response credibility scores are calculated by combining linguistic indicator values, we identify responses that have potentially conflicting or potentially supporting slot-filler candidates. For such responses, their credibility scores are changed in accordance with the binary values returned by the following indicators.

Interdependent claims indicators are shown as follows.

1. *Conflicting slot fillers*: When fillers for two claims with the same query entity and slot type appear in the same evidence sentence, we apply an additional heuristic rule designed for the slot type in question. For example, the following evidence sentence indicates that compared to "*Cathleen P. Black*", "*Susan K. Reed*" is more likely to be in a "*org:top_employees/members*" relation with "*The Oprah Magazine*" due to the latter pair's shorter dependency path: "*Hearst Magazine's President Cathleen P. Black has appointed Susan K. Reed as editor-in-chief of the U.S. edition of The Oprah Magazine.*". The credibility scores are accordingly changed (or kept at) 0.5 for responses associated with the former claim, and 1.0 for those associated with the latter.

2. *Inter-dependent slot types*: Many slot types are inter-dependent, such as "*per:title*" and "*per:employee_of*", and various family slots. After determining initial credibility scores for each response, we check whether evidence exists for any implied claims. For example, given initial credibility scores of 1.0 for two responses supporting the claims that (1)"*David*" is "*per:children*" of "*Carolyn Goodman*" and (2)"*Andrew*" is "*per:sibling*" of "*David*", we check for any responses supporting the claim that (3)"*Andrew*" is "*per:children*" of "*Carolyn Goodman*", and set their credibility scores to 1.0. For example, a response supporting this claim included the evidence sentence, "*Dr. Carolyn Goodman, her husband, Robert, and their son, David, said goodbye to David's brother, Andrew.*".

5.4 Experiments

This section presents the experiment results and analysis of our approach.

5.4.1 Data

The data set we use is from the TAC-KBP2013 Slot Filling Validation (SFV) task, which consists of the merged responses returned by 52 runs (regarded as systems in MTM) from 18 teams submitted to the Slot Filling (SF) task. The source collection has 1,000,257 newswire documents, 999,999 web documents and 99,063 discussion forum posts, which results in 10 different sources (combinations of publication venues and genres) in our experiment. There are 100 queries: 50 person and 50 organization entities. After removing redundant responses within each single system run, we use 45,950 unique responses as the input to truth-finding. Linguistic Data Consortium (LDC) human annotators manually assessed all of these responses and produced 12,844 unique responses as ground truth. In order to compare with state-of-the-art supervised learning methods for SFV [63], [64], we trained a SVMs classifier [7] as a counterpart, incorporating the same set of linguistic indicators, sources and systems as features. We picked 10% (every 10th line) to compose the development set for MTM and the training set for the SVMs. The rest is used for blind test.

5.4.2 Overall Performance

Table 5.2 shows the overall performance of various truth finding methods on judging each response as true or false. MTM achieves promising results and even outperforms supervised learning approach. Table 5.3 presents some examples ranked at the top and the bottom based on the credibility scores produced by MTM. The last column represents the system rank.

We can see that majority voting across systems performs much better than random assessment, but its accuracy is still low. For example, the true claim *T5* was extracted by only one system because most systems mistakenly identified "*Briton Stuart Rose*" as a person name. In comparison, MTM obtained much better accuracy by also incorporating multiple dimensions of source and evidence information.

[7] We used the LIBSVM toolkit [123] with Gaussian radial basis function kernel.

Table 5.2: Overall performance comparison ($\star$ mean average precision).

Methods	Precision	Recall	F-measure	Accuracy	MAP*
1.Random	28.64%	50.48%	36.54%	50.54%	34%
2.Voting	42.16%	70.18%	52.68%	62.54%	62%
3.Linguistic Indicators	50.24%	70.69%	58.73%	72.29%	60%
4.SVM (3 + System + Source)	56.59%	48.72%	52.36%	75.86%	56%
5.MTM (3 + System + Source)	53.94%	**72.11**%	**61.72**%	**81.57**%	**70**%

Method 3 using linguistic indicators alone, already achieved promising results. For example, many claims are judged as truths through trigger phrases (*T1* and *T5*), event extraction (*T2*), coreference (*T4*), and node type indicators (*T3*). On the other hand, many claims are correctly judged as false because their evidence sentences did not include the slot filler (*F1*, *F4*, *F5*) or valid knowledge paths to connect the query entity and the slot filler (*F2*, *F3*). The performance gain (2.99% F-score) from Method 3 to Method 5 shows the need for incorporating system and source dimensions. For example, most truths are from news while many false claims are from newsgroups and discussion forum posts (*F1*, *F2*, *F5*).

The SVMs model got very low recall because of the following two reasons: (1) It ignored the inter-dependency between multiple dimensions; (2) the negative instances are dominant in the training data, so the model is biased towards labeling responses as false.

5.4.3 Truth Finding Efficiency

Table 5.3 shows that some truths (*T1*) are produced from low-ranked systems whereas some false responses from high-ranked systems (*F1*, *F2*). Note that systems are ranked by their performance in KBP SF task. In order to find all the truths, human assessors need to go through all the responses returned by multiple systems. This process was proven very tedious and costly [28], [63].

Our MTM approach can expedite this process by ranking responses based on their credibility scores and asking human to assess the responses with high credibility first. Traditionally, when human assess responses, they follow an alphabetical order or system IDs in a "passive learning" style. This is set as our baseline. For comparison, we also present the results using only linguistic indicators, using voting in which the responses

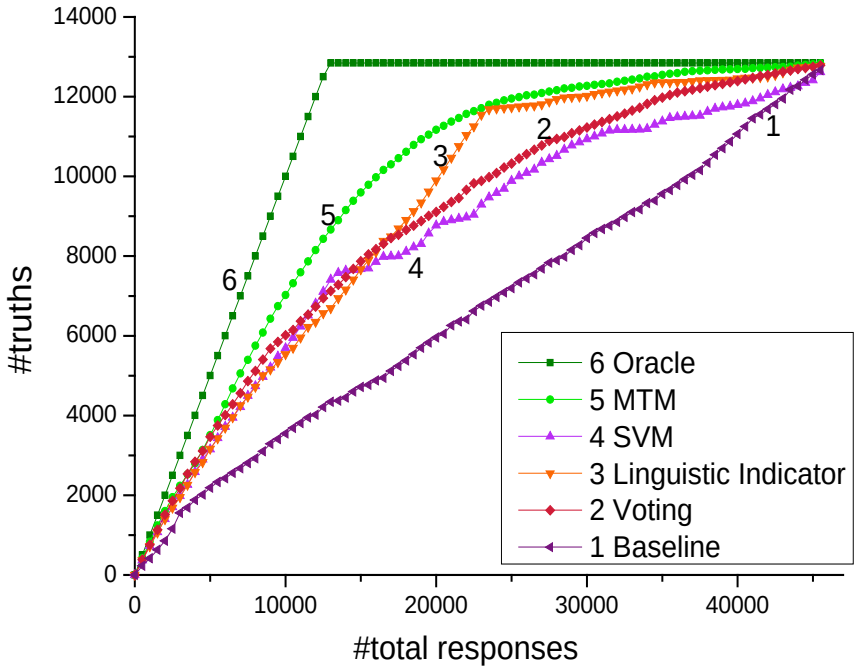

Figure 5.3: Truth finding efficiency.

which get more votes across systems are assessed first, and the oracle method assessing all correct responses first. Table 5.2 shows our model can successfully rank trustworthy responses at high positions compared with other approaches.

Fig. 5.3 summarizes the results from the above 6 approaches. The common end point of all curves represents the cost and benefit of assessing all system responses. We can see that the baseline is very inefficient at finding the truths. If we employ linguistic indicators, the process can be dramatically expedited. MTM provides further significant gains, with performance close to the Oracle. With only half the cost of the baseline, MTM can already find 90% truths.

5.4.4 Enhance Individual SF Systems

Finally, as a by-product, our MTM approach can also be exploited to validate the responses from each individual SF system based on their credibility scores. For fair comparison with the official KBP evaluation, we use the same ground-truth in KBP2013 and standard precision, recall and F-measure metrics as defined in [26]. To increase the chance of including truths which may be particularly difficult for a system to find, LDC prepared a manual key which was assessed and included in the final ground truth. According to the

SF evaluation setting, F-measure is computed based on the number of unique true claims. After removing redundancy across multiple systems, there are 1,468 unique true claims. The cutoff criteria for determining whether a response is true or not was optimized from the development set.

Fig. 5.4 presents the F-measure scores of the best run from each individual SF system. We can see that our MTM approach consistently improves the performance of almost all SF systems, in an absolute gain range of [-1.22%, 5.70%]. It promotes state-of-the-art SF performance from 33.51% to 35.70%. Our MTM approach provides more gains to SF systems which mainly rely on lexical or syntactic patterns than other systems using distant supervision or logic rules.

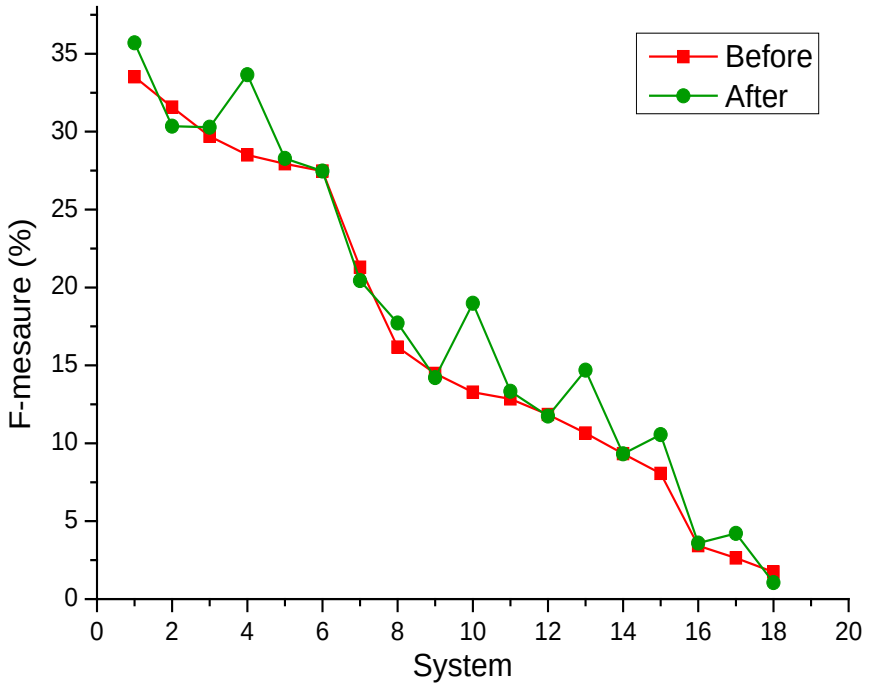

Figure 5.4: Impact on individual SF systems.

5.5 Summary

Truth finding has received attention from both Natural Language Processing (NLP) and Data Mining communities. NLP work has mostly explored linguistic analysis of the content, while Data Mining work proposed advanced models in resolving conflict information from multiple sources. They have relative strengths and weaknesses. In this chap-

ter we leverage the strengths of these two distinct, but complementary research paradigms and propose a novel unsupervised multi-dimensional truth-finding framework incorporating signals both from multiple sources, multiple systems and multiple evidences based on extended dependency graph construction with multi-layer linguistic analysis.

Experiments on the case study of Slot Filling Validation demonstrate that our approach can find truths accurately (9.4% higher F-score than supervised methods) and efficiently (finding 90% truths with only one half the cost of a baseline without credibility estimation).

Table 5.3: Top and bottom response examples ranked by MTM (T: top truths, F: bottom false claims).

	Response Ranked by MTM						
	Claim			Evidence	Source		Rank
	Query Entity	Slot Type	Slot Filler				
T1	China Banking Regulatory Commission	org:top members-employees	Liu Mingkang	**Liu Mingkang**, the chairman of the China Banking Regulatory Commission	Central News Agency of Taiwan	News	15
T2	Galleon Group	org:founded by	Raj Rajaratnam	Galleon Group, founded by billionaire **Raj Rajaratnam**	New York Times	News	9
T3	Mike Penner	per:age	52	Mike Penner, **52**	New York Times	News	1
T4	China Banking Regulatory Commission	org:alternate names	CBRC	...China Banking Regulatory Commission said in the notice. The five banks ... according to **CBRC**.	Xinhua, News	News	5
T5	Stuart Rose	per:origin	Briton	Bolland will replace **Briton** Stuart Rose at the start of 2010.	Agence France-Presse	News	3
F1	American Association for the Advancement of Science	org:top members employees	Freedman	American Library Association, President: Maurice **Freedman** < http://www.aft.org	Google	News Group	4
F2	Jade Goody	per:origin	Britain	because Jade Goody's the only person to ever I love **Britain**	Discussion Forum		3
F3	Don Hewitt	per:spouse	Swap	...whether "Wife **Swap**" on ABC or "Jon & Kate" on TLC	New York Times	News	7
F4	Council of Mortgage Lenders	org:website	cml.org.uk	me purchases in the U.K. jumped by 16 percent in April, suggesting the property market slump may have bottomed out	Associated Press Worldstream	News	18
F5	Don Hewitt	per:alternate names	Hewitt Mchen	US DoMIna THOMPson LACtaTe haVeD [3866 words]	Google	News Group	13

CHAPTER 6
Conclusions and Future Directions

6.1 Limitations

- **Dependency Parsing**

  To obtain the rich syntactic information besides shallow surface features for deeper language analysis, our method rely on dependency parsing. The majority of work on dependency parsing has been dedicated to resource-rich languages [124] such as English, Chinese, Spanish and French. However, for an Incident Language with very few resources (*e.g.*, Hausa, Thai, Yoruba and Bengali), there are fewer or even no labeled training data for parsing and it is labor-intensive and time-consuming to manually build treebanks for these languages. Therefore, currently our method cannot be adapted to these languages lacking of available dependency parsers.

- **Limited Argument Types** Our current framework is limited to extracting relations based on entities by applying name tagging techniques [88]. However, there are limited number of entity types such as Person, Location, Organization and so on. We rely on gazetteers to roughly identify diseases, crimes, or regions. One possible direction is to combine our framework with fine-grained entity and phrase typing techniques (*e.g.*, [125],[126]) which can be quickly adapted to a new domain, genre, and language.

6.2 Remaining Challenges

6.2.1 Value-Driven Relation Types

We placed more emphasis on the trigger-driven slot types in our previous work. As future work, we aim to work on the value-driven slot types (*e.g.*, *per:title*, *per:age*, *per:origin*, *per:religion*). It is interesting to note the difference between the two kinds of slot types. For value-driven slots, there is a stronger connection between query and filler and usually there is no lexical cues to indicate such a relation.

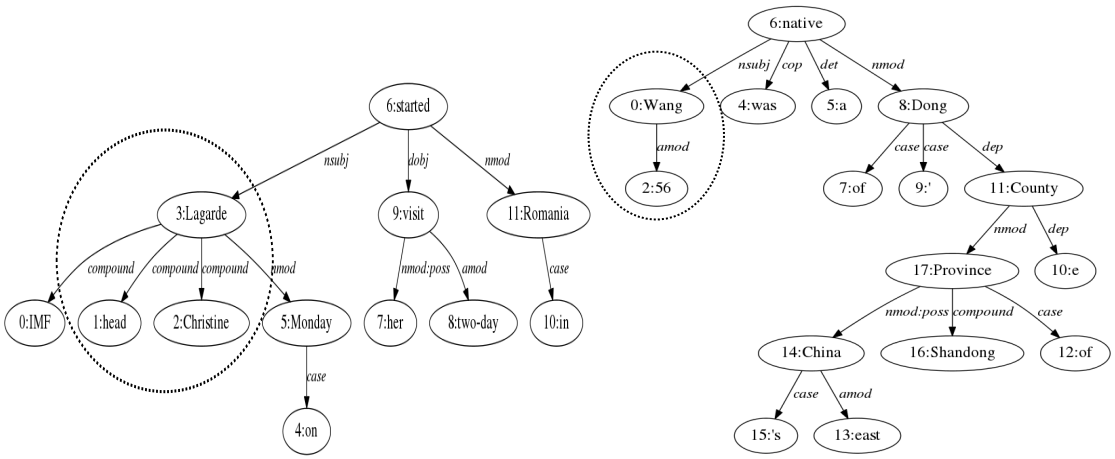

Figure 6.1: Dependency tree for value-driven slots per:title (left) and per:age (right).

Therefore, fine-grained slot filler types become more important for extracting value fillers. For example, a correct *per:origin* slot filler should be the nationality and/or ethnicity of the given query and a *per:religion* slot filler should be a religion name. Besides applying existing manually cleaned gazetteers for typing filler candidates, we plan to extract gazetteers automatically based on the NELL system output [122]. Nell provides more than $1,000$ categories for noun phrases (*e.g.*, person, disease, fruit, emotion, origin, religion).

For value-driven slot types, there is a relatively short distance between query and filler which can be captured by lexical or syntactic patterns easily. However, these patterns have poor potability to a new language and a "surprise" slot type. We notice that filler is strongly connected to query and has relatively weak connection with other entities in the same dependency graph. For example, given two sentences

*"**Wang**, **56**, was a native of Dong'e County of east China's Shandong Province."*

*"IMF **head Christine Lagarde** on Monday started her two-day visit in Romania."*

we can see the age filler "*56*" only serves to modify the person query "*Wang*" and the title filler "*head*" plays the same role in modifying the query "*Christine Lagarde*". Therefore, one possible direction is to compute the connection strength between a filler candidate and the query compared with other entities in the same dependency tree.

For trigger-driven slot types, our unsupervised graph-based approach has a good performance in discovering the connection between filler and trigger. However, the ap-

Table 6.1: Gender in Spanish triggers.

slot type	Spanish Trigger		English Translation
	Masculine	Feminine	
birth-related	nacido	nacida	*born*
per:spouse	esposo	esposa	*spouse*

proach will attach the correct (trigger, filler) pair to an irrelevant entity when there are multiple entities competing with the query mention for the filler. For example, *"Jack McEdwards"* is mistakenly extracted as the *per:spouse* filler for the query *"Blake McEdwards"* in the following sentence. One reason is that our model places extra emphasis on the connection between filler and trigger.

*"He became **Blake McEdwards** when he was 4, after his mother, Lillian, had **married Jack McEdwards**, an assistant director and movie production manager."*

To tackle the above problems, we will keep working on applying graphical models to gain a deeper understanding of the relationship between query and filler regardless of the existence of triggers so that our method can be applied to both trigger-driven and value-driven slot types.

6.2.2 Multilingual Relation Extraction

Our framework only requires name tagging and dependency parsing as pre-processing and a few trigger seeds as input, and thus it can be easily adapted to a new language. We will keep our method language-independent and test it on Spanish besides English and Chinese.

Corenlp [88] provides the language support for Spanish including tokenize, Part of Speech tagging, and name tagging. Spanish dependency parsers (*e.g.*, DepPattern [127]) are also available. Compared to English and Chinese, the main problem comes from the word gender. In Spanish, the following word classes have gender: determiners, nouns, pronouns, adjectives and participle verbs [128] (examples in Table 6.1).

In addition, Chinese is a very concise language. For example, a "[Person Name - Organization Suffix]" structure can indicate various different types of relations between the person name and the organization: "杨明牙医诊所" (*Yang Ming Clinic*) indicates ownership, "邵逸夫图书馆" (*Shao Yifu Library*) indicates sponsorship, "丰子恺研究

中心" (*Feng Zikai Research Center*) indicates research theme, and "罗京治丧委员会" (*Luojing Commemoration Committee*) indicates commemoration. None of them includes an explicit trigger nor indicates employment relation. It requires more fine-grained dependency relation types to distinguish them.

Compared to English, Chinese tends to have more variants for some types of triggers (e.g., there are at least 31 different titles for "*wife*" in Chinese). Some of them are implicit and require shallow inference. For example, "投奔" (*to seek shelter or asylum*) indicates a residence relation in most cases. It is an interesting and meaningful problem to combine language-specific features with language-independent open relation extraction techniques automatically.

REFERENCES

[1] Y. Li *et al.*, "PRIS at TAC2013 KBP Track," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2013.

[2] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *Proc. Int. Conf. World Wide Web*, Banff, Canada, 2007, pp. 697–706.

[3] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Vancouver, Canada, 2008, pp. 1247–1250.

[4] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, "DBpedia: A nucleus for a web of open data," in *Proc. 6th Int. Semantic Web Conf.*, Busan, Korea, 2007, pp. 722–735.

[5] D. Savenkov, W.-L. Lu, J. Dalton, and E. Agichtein, "Relation extraction from community generated question-answer pairs," in *Proc. NAACL-HLT 2015 Student Res. Workshop*, Denver, CO, 2015, pp. 96–102.

[6] R. Grishman, "Information Extraction: Techniques and Challenges," in *Information Extraction: a Multidisciplinary Approach to an Emerging Information Technology*, M. T. Pazienza, Ed. New York, NY: Springer, 1977, vol. 1299, pp. 10–27.

[7] J. Hoffart, F. M. Suchanek, K. Berberich, and G. Weikum, "YAGO2: a Spatially and Temporally Enhanced Knowledge Base from Wikipedia," *Artif. Intell.*, vol. 194, no. 1, pp. 28–61, Jan. 2013.

[8] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Advances Neural Inform. Process. Syst. 26*, Lake Tahoe, NV, 2013, pp. 926–934.

[9] M. Gardner, P. Talukdar, J. Krishnamurthy, and T. Mitchell, "Incorporating vector space similarity in random walk inference over knowledge bases," in *Proc. Conf. Empirical Methods Natural Language Process.*, Doha, Qatar, 2014, pp. 397–406.

[10] Wikipedia, "Unstructured data — Wikipedia, the free encyclopedia." [Online]. Available: https://en.wikipedia.org/wiki/Unstructured_data (Date Last Accessed October 9, 2017).

[11] L. Chen, Y. Feng, Y. Chen, L. Zou, and D. Zhao, "Towards automatic construction of knowledge bases from chinese online resources," in *Proc. ACL 2012 Student Res. Workshop*, Jeju, South Korea, 2012, pp. 67–72.

[12] F. Esposito, S. Ferilli, N. Fanizzi, and G. Semeraro, "Learning from parsed sentences with inthelex," in *Proc. 2nd Workshop Learning Language Logic and 4th Conf. Computational Natural Language Learning*, Lisbon, Portugal, 2000, pp. 194–198.

[13] H. Ji and R. Grishman, "Knowledge base population: successful approaches and challenges," in *Proc. 49th Annu. Meeting Assoc. Computational Linguistics*, Portland, OR, 2011, pp. 1148–1158.

[14] Z. Chen *et al.*, "CUNY-BLENDER TAC-KBP2010 Entity Linking and Slot Filling system description," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2010.

[15] R. Grishman and B. Min, "New York University KBP 2010 Slot-Filling system," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2010.

[16] Y. Li *et al.*, "PRIS at TAC2012 KBP Track," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2012.

[17] D. Ravichandran and E. Hovy, "Learning surface text patterns for a question answering system," in *Proc. 40th Annu. Meeting Assoc. Computational Linguistics*, Philadelphia, PA, 2002, pp. 41–47.

[18] M. Mintz, S. Bills, R. Snow, and D. Jurafsky, "Distant supervision for relation extraction without labeled data," in *Proc. Joint Conf. 47th Ann. Meeting ACL and 4th IJCNLP AFNLP*, Suntec, Singapore, 2009, pp. 1003–1011.

[19] M. Surdeanu *et al.*, "A simple distant supervision approach for the TAC-KBP Slot Filling task," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2010.

[20] B. Roth, T. Barth, M. Wiegand, M. Singh, and D. Klakow, "Effective Slot Filling based on shallow distant supervision methods," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2013.

[21] G. Angeli, J. Tibshirani, J. Wu, and C. Manning, "Combining distant and partial supervision for relation extraction," in *Proc. Conf. Empirical Methods Natural Language Process.*, Doha, Qatar, 2014, pp. 1556–1567.

[22] V. Ng, "Supervised noun phrase coreference research: The first fifteen years," in *Proc. 48th Annu. Meeting Assoc. Computational linguistics*, Uppsala, Sweden, 2010, pp. 1396–1411.

[23] M. Recasens, M.-C. de Marneffe, and C. Potts, "The life and death of discourse entities: Identifying singleton mentions," in *Proc. NAACL-HLT 2013*, Atlanta, GA, 2013, pp. 627–633.

[24] H. Lee, A. Chang, Y. Peirsman, N. Chambers, M. Surdeanu, and D. Jurafsky, "Deterministic Coreference Resolution Based on Entity-Centric, Precision-Ranked Rules," *Computational Linguistics*, vol. 39, no. 4, pp. 885–916, Dec. 2013.

[25] B. Min and R. Grishman, "Challenges in the knowledge base population Slot Filling task," in *Proc. Int. Conf. Language Resources and Evaluation*, Istanbul, Turkey, 2012, pp. 1137–1142.

[26] H. Ji, R. Grishman, and H. Dang, "An overview of the TAC2011 Knowledge Base Population Track," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2011.

[27] P. McNamee and H. Dang, "Overview of the TAC 2009 Knowledge Base Population Track," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2009.

[28] H. Ji, R. Grishman, H. Dang, K. Griffitt, and J. Ellis, "An overview of the TAC2010 Knowledge Base Population Track," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2010.

[29] M. Surdeanu and H. Ji, "Overview of the English Slot Filling Track at the TAC2014 Knowledge Base Population Evaluation," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2014.

[30] NIST, "TAC KBP 2016 validation/ensembling track." [Online]. Available: http://tac.nist.gov/2016/KBP/SFValidation/ (Date Last Accessed October 9, 2017).

[31] D. Yu *et al.*, "RPI-BLENDER TAC-KBP2013 knowledge base population system," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2013.

[32] D. Yu, H. Ji, S. Li, and C. Lin, "Why read if you can scan: Scoping strategy for biographical fact extraction," in *Proc. Conf. North Amer. Chapter Assoc. Computational Linguistics - Human Language Technol.*, Denver, CO, 2015, pp. 1203–1208.

[33] D. Yu and H. Ji, "Unsupervised Person Slot Filling based on graph mining," in *Proc. 54th Annu. Meeting Assoc. Computational Linguistics*, Berlin, Germany, 2016, pp. 44–53.

[34] D. Yu, H. Ji, and L. Huang, "Open relation extraction and grounding," in *Proc. 8th Int. Joint Conf. Natural Language Process.*, Taipei, Taiwan, 2017, pp. 1–11.

[35] D. Yu *et al.*, "The wisdom of minority: Unsupervised Slot Filling Validation based on multi-dimensional truth-finding." in *Proc. 25th Int. Conf. Computational Linguistics*, Dublin, Ireland, 2014, pp. 1567–1578.

[36] G. Angeli *et al.*, "Stanford's 2014 Slot Filling systems," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2014.

[37] A. Sun, R. Grishman, B. Min, and W. Xu, "NYU 2011 system for KBP Slot Filling," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2011.

[38] S. Soderland, J. Gilmer, R. Bart, O. Etzioni, and D. Weld, "Open IE to KBP relations in 3 hours," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2013.

[39] F. Suchanek, M. Sozio, and G. Weikum, "SOFIE: a self-organizing framework for information extraction," in *Proc. Int. Conf. World Wide Web*, Madrid, Spain, 2009, pp. 631–640.

[40] H. Poon and P. Domingos, "Unsupervised semantic parsing," in *Proc. Conf. Empirical Methods Natural Language Process.*, Suntec, Singapore, 2009, pp. 1–10.

[41] F. Wu and D. S. Weld, "Open information extraction using wikipedia," in *Proc. 48th Annu. Meeting Assoc. Computational Linguistics*, Uppsala, Sweden, 2010, pp. 118–127.

[42] N. Nakashole, M. Theobald, and G. Weikum, "Scalable knowledge harvesting with high precision and high recall," in *Proc. ACM Int. Conf. Web Search and Data Mining*, Hong Kong, China, 2011, pp. 227–236.

[43] A. Fader, S. Soderland, and O. Etzioni, "Identifying relations for open information extraction," in *Proc. Conf. Empirical Methods Natural Language Process*, Edinburgh, UK, 2011, pp. 1535–1545.

[44] N. Nakashole, G. Weikum, and F. Suchanek, "PATTY: a taxonomy of relational patterns with semantic types," in *Proc. 2012 Joint Conf. Empirical Methods Natural Language Process. and Computational Natural Language Learning*, Jeju, South Korea, 2012, pp. 1135–1145.

[45] Mausam, M. Schmitz, S. Soderland, R. Bart, and O. Etzioni, "Open language learning for information extraction," in *Proc. 2012 Joint Conf. Empirical Methods Natural Language Process. and Computational Natural Language Learning*, Jeju, South Korea, 2012, pp. 523–534.

[46] C. Bovi, L. Telesca, and R. Navigli, "Large-Scale Information Extraction from Textual Definitions Through Deep Syntactic and Semantic Analysis," *Transactions Assoc. for Computational Linguistics*, vol. 3, no. 1, pp. 529–543, Oct. 2015.

[47] G. Angeli, M. Premkumar, and C. Manning, "Leveraging linguistic structure for open domain information extraction," in *Proc. 53rd Annu. Meeting Assoc. Computational Linguistics*, Beijing, China, 2015, pp. 344–354.

[48] A. Grycner and G. Weikum, "POLY: Mining relational paraphrases from multilingual sentences," in *Proc. Conf. Empirical Methods Natural Language Process.*, Austin, TX, 2016, pp. 2183–2192.

[49] S. Kok and P. Domingos, “Extracting semantic networks from text via relational clustering,” in *Proc. Joint Eur. Conf. Mach. Learning and Knowledge Discovery Databases*, Antwerp, Belgium, 2008, pp. 624–639.

[50] B. Min, S. Shi, R. Grishman, and C. Lin, “Ensemble semantics for large-scale unsupervised relation extraction,” in *Proc. 2012 Joint Conf. Empirical Methods Natural Language Process. and Computational Natural Language Learning*, Jeju, South Korea, 2012, pp. 1027–1037.

[51] L. Del Corro and R. Gemulla, “Clausie: clause-based open information extraction,” in *Proc. Int. Conf. World Wide Web*, Rio de Janeiro, Brazil, 2013, pp. 355–365.

[52] S. Soderland, J. Gilmer, R. Bart, E. Orenand, and D. Weld, “Open information extraction to KBP relations in 3 hours,” in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2013.

[53] G. Angeli *et al.*, “Stanford’s distantly supervised Slot Filling systems for KBP 2014,” in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2015.

[54] S. Riedel, L. Yao, and A. McCallum, “Modeling relations and their mentions without labeled text,” in *Proc. Joint Eur. Conf. Mach. Learning and Principles and Practice Knowledge Discovery Databases*, Barcelona, Spain, 2010, pp. 148–163.

[55] J. Weston, A. Bordes, O. Yakhnenko, and N. Usunier, “Connecting language and knowledge bases with embedding models for relation extraction,” in *Proc. Conf. Empirical Methods Natural Language Process.*, Seattle, WA, 2013, pp. 1366–1371.

[56] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon, “Representing text for joint embedding of text and knowledge bases,” in *Proc. Conf. Empirical Methods Natural Language Process.*, Lisbon, Portugal, 2015, pp. 1499–1509.

[57] S. Riedel, L. Yao, A. McCallum, and B. Marlin, “Relation extraction with matrix factorization and universal schemas,” in *Proc. Conf. North Amer. Chapter Assoc. Computational Linguistics - Human Language Technol.*, Atlanta, GA, 2013, pp. 74–84.

[58] T. Rocktäschel, S. Singh, and S. Riedel, “Injecting logical background knowledge into embeddings for relation extraction,” in *Proc. Conf. North Amer. Chapter Assoc. Computational Linguistics - Human Language Techn.*, Denver, CO, 2015, pp. 1119–1129.

[59] D. Wijaya and T. Mitchell, “Mapping verbs in different languages to knowledge base relations using web text as interlingua,” in *Proc. Conf. North Amer. Chapter Assoc. Computational Linguistics - Human Language Technol.*, San Diego, CA, 2016, pp. 818–827.

[60] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web," Stanford InfoLab, Stanford, CA, Tech. Rep. 1999-66, Nov. 1999. [Online]. Available: http://ilpubs.stanford.edu:8090/422/ (Date Last Accessed August 31, 2017).

[61] R. Mihalcea and P. Tarau, "TextRank: Bringing order into texts," in *Proc. Conf. Empirical Methods Natural Language Process.*, Barcelona, Spain, 2004, pp. 404–411.

[62] S. White and P. Smyth, "Algorithms for estimating relative importance in networks," in *Proceedings 9th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Washington, D.C., 2003, pp. 266–275.

[63] S. Tamang and H. Ji, "Adding smarter systems instead of human annotators: Re-ranking for system combination," in *Proc. 1st Int. Workshop Search and Mining Entity-relationship Data*, Glasgow, UK, 2011, pp. 3–8.

[64] X. Li and R. Grishman, "Confidence estimation for knowledge base population," in *Proc. Recent Advances Natural Language Process.*, Hissar, Bulgaria, 2013, pp. 396–401.

[65] X. Yin, J. Han, and S. Y. Philip, "Truth Discovery with Multiple Conflicting Information Providers on the Web," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 6, pp. 796–808, June 2008.

[66] X. L. Dong, L. Berti-Equille, and D. Srivastava, "Integrating Conflicting Data: the Role of Source Dependence," *VLDB Endowment*, vol. 2, no. 1, pp. 550–561, Aug. 2009.

[67] A. Galland, S. Abiteboul, A. Marian, and P. Senellart, "Corroborating information from disagreeing views," in *Proc. 3rd ACM Int. Conf. Web Search and Data Mining*, New York City, NY, 2010, pp. 131–140.

[68] X. L. Dong, L. Berti-Equille, and D. Srivastava, "Truth Discovery and Copying Detection in a Dynamic World," *VLDB Endowment*, vol. 2, no. 1, pp. 562–573, Aug. 2009.

[69] L. Blanco, V. Crescenzi, P. Merialdo, and P. Papotti, "Probabilistic models to reconcile complex data from inaccurate data sources," in *Proc. Int. Conf. Advanced Inform. Syst. Eng.*, Hammamet, Tunisia, 2010, pp. 83–97.

[70] J. Pasternack and D. Roth, "Knowing what to believe (when you already know something)," in *Proc. 23rd Int. Conf. Computational Linguistics*, Beijing, China, 2010, pp. 877–885.

[71] X. Yin and W. Tan, "Semi-supervised truth discovery," in *Proc. Int. Conf. World Wide Web*, Hyderabad, India, 2011, pp. 217–226.

[72] J. Pasternack and D. Roth, "Making better informed trust decisions with generalized fact-finding," in *Proc. 22nd Int. Joint Conf. Artificial Intell.*, Barcelona, Spain, 2011, pp. 2324–2329.

[73] V. Vydiswaran, C. Zhai, and D. Roth, "Content-driven trust propagation framework," in *Proc. 17th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, San Diego, CA, 2011, pp. 974–982.

[74] L. Ge, J. Gao, X. Yu, W. Fan, and A. Zhang, "Estimating local information trustworthiness via multi-source joint matrix factorization," in *Proc. 2012 IEEE 12th Int. Conf. on Data Mining*, Brussels, Belgium, 2012, pp. 876–881.

[75] B. Zhao, B. I. Rubinstein, J. Gemmell, and J. Han, "A Bayesian Approach to Discovering Truth from Conflicting Sources for Data Integration," *VLDB Endowment*, vol. 5, no. 6, pp. 550–561, Feb. 2012.

[76] D. Wang, L. Kaplan, H. Le, and T. Abdelzaher, "On truth discovery in social sensing: A maximum likelihood estimation approach," in *Proc. ACM/IEEE Int. Conf. Inform. Process. Sensor Networks*, Beijing, China, 2012, pp. 233–244.

[77] J. Pasternack and D. Roth, "Latent credibility analysis," in *Proc. Int. Conf. World Wide Web*, Rio de Janeiro, Brazil, 2013, pp. 1009–1020.

[78] E. Riloff and R. Jones, "Learning dictionaries for information extraction by multi-level bootstrapping," in *Proc. 16th Nat. Conf. Artificial Intell.*, Orlando, FL, 1999, pp. 474–479.

[79] E. Agichtein and L. Gravano, "Snowball: Extracting relations from large plain-text collections," in *Proc. 5th ACM Conf. Digital Libraries*, San Antonio, TX, 2000, pp. 85–94.

[80] A. Sun, R. Grishman, W. Xu, and B. Min, "New York University 2011 system for KBP Slot Filling," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2011.

[81] J. Ganitkevitch, B. Van Durme, and C. Callison-Burch, "PPDB: The paraphrase database," in *Proc. Conf. North Amer. Chapter Assoc. Computational Linguistics - Human Language Techn.*, Atlanta, GA, 2013, pp. 758–764.

[82] G. Jeh and J. Widom, "Scaling personalized web search," in *Proc. Int. Conf. World Wide Web*, Budapest, Hungary, 2003, pp. 271–279.

[83] R. Motwani and P. Raghavan, "Randomized Algorithms," *ACM Computing Surveys*, vol. 28, no. 1, pp. 33–37, Mar. 1996.

[84] B. J. Frey and D. Dueck, "Clustering by Passing Messages Between Data Points," *Science*, vol. 315, no. 5814, pp. 972–976, Feb. 2007.

[85] O. Bronstein, I. Dagan, Q. Li, H. Ji, and A. Frank, "Seed-based event trigger labeling: How far can event descriptions get us?" in *Proc. 53rd Annu. Meeting Assoc. Computational Linguistics and 7th Int. Joint Conf. Natural Language Process.*, Beijing, China, 2015, pp. 372–376.

[86] NIST, "TAC KBP 2015 slot descriptions." [Online]. Available: https://tac.nist.gov/2015/KBP/ColdStart/guidelines/TAC_KBP_2015_Slot_Descriptions_V1.0.pdf (Date Last Accessed October 9, 2017).

[87] E. Pavlick, P. Rastogi, J. Ganitkevitch, B. Van Durme, and C. Callison-Burch, "PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification," in *Proc. 53rd Annu. Meeting Assoc. Computational Linguistics and 7th Int. Joint Conf. Natural Language Process.*, Beijing, China, 2015, pp. 425–430.

[88] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Proc. 52nd Annu. Meeting Assoc. Computational Linguistics: Syst. Demonstrations*, Baltimore, MD, 2014, pp. 55–60.

[89] G. Angeli *et al.*, "Bootstrapped self training for knowledge base population," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2015.

[90] M. Wang, W. Che, and C. D. Manning, "Joint word alignment and bilingual named entity recognition using dual decomposition," in *Proc. 51st Annu. Meeting Assoc. Computational Linguistics*, Sofia, Bulgaria, 2013, pp. 1073–1082.

[91] R. Levy and C. Manning, "Is it harder to parse Chinese, or the Chinese Treebank?" in *Proc. 41st Annu. Meeting Assoc. Computational Linguistics*, Sapporo, Japan, 2003, pp. 439–446.

[92] W. Che, Z. Li, and T. Liu, "LTP: A Chinese language technology platform," in *Proc. 23rd Int. Conf. Computational Linguistics: Demonstrations*, Beijing, China, 2010, pp. 13–16.

[93] M. Banko and O. Etzioni, "The tradeoffs between open and traditional relation extraction," in *Proc. 46th Annu. Meeting Assoc. Computational Linguistics*, Columbus, OH, 2008, pp. 28–36.

[94] Y. Xu, M.-Y. Kim, K. Quinn, R. Goebel, and D. Barbosa, "Open information extraction with tree kernels," in *Proc. Conf. North Amer. Chapter Assoc. Computational Linguistics - Human Language Technol.*, Atlanta, GA, 2013, pp. 868–877.

[95] N. Bhutani, H. Jagadish, and D. R. Radev, "Nested propositions in open information extraction," in *Proc. Conf. Empirical Methods Natural Language Process.*, Austin, TA, 2016, pp. 55–64.

[96] Wikipedia, "Automatic content extraction — Wikipedia, the free encyclopedia." [Online]. Available: https://en.wikipedia.org/wiki/Automatic_Content_Extraction (Date Last Accessed October 9, 2017).

[97] D. Aldous and J. Fill, "Reversible Markov Chains and Random Walks on Graphs," 2002. [Online]. Available: https://www.stat.berkeley.edu/~aldous/RWG/book.pdf (Date Last Accessed September 5, 2017).

[98] L. Lovász, "Random Walks on Graphs," *Combinatorics, Paul Erdos Is Eighty*, vol. 2, no. 1, pp. 1–46, Jan. 1993.

[99] D. J. Klein and M. Randić, "Resistance Distance," *J. Math. Chemistry*, vol. 12, no. 1, pp. 81–95, Dec. 1993.

[100] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens, "Random-Walk Computation of Similarities Between Nodes of a Graph with Application to Collaborative Recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 355–369, Mar. 2007.

[101] Y. Li and Z.-L. Zhang, "Random walks on digraphs, the generalized digraph laplacian and the degree of asymmetry," in *Proc. Int. Workshop Algorithms and Models Web-Graph*, Stanford, CA, 2010, pp. 74–85.

[102] A. Björkelund and R. Farkas, "Data-driven multilingual coreference resolution using resolver stacking," in *Proc. 2012 Joint Conf. Empirical Methods Natural Language Process. and Computational Natural Language Learning*, Jeju, South Korea, 2012, pp. 49–55.

[103] S. Spagnola and C. Lagoze, "Edge dependent pathway scoring for calculating semantic similarity in conceptnet," in *Proc. 9th Int. Conf. Computational Semantics*, Oxford, UK, 2011, pp. 385–389.

[104] J. Guo, H. Guo, and Z. Wang, "An Activation Force-Based Affinity Measure for Analyzing Complex Networks," *Scientific Reports*, vol. 1, no. 113, pp. 1–9, Oct. 2011.

[105] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances Neural Inform. Process. Syst. 26*, Lake Tahoe, NV, 2013, pp. 2787–2795.

[106] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances Neural Inform. Process. Syst. 26*, Lake Tahoe, NV, 2013, pp. 3111–3119.

[107] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Language Process.*, Doha, Qatar, 2014, pp. 1532–1543.

[108] G. Stanovsky and I. Dagan, "Creating a large benchmark for open information extraction," in *Proc. Conf. Empirical Methods Natural Language Process.*, Austin, TX, 2016, pp. 2300–2305.

[109] S. Singh *et al.*, "Universal schema for Slot Filling and Cold Start: UMass IESL at TACKBP 2013," in *Proc. Text Anal. Conf.*, Gaithersburg, MD, 2013.

[110] J. Christensen, Mausam, S. Soderland, and O. Etzioni, "An analysis of open information extraction based on semantic role labeling," in *Proc. 6th Int. Conf. Knowledge Capture*, Banff, Canada, 2011, pp. 113–120.

[111] H. Pal and Mausam, "Demonyms and compound relational nouns in nominal open IE," in *Proc. 5th Workshop AKBC*, San Diego, CA, 2016, pp. 35–39.

[112] Wikipedia, "Jenks natural breaks optimization — Wikipedia, the free encyclopedia." [Online]. Available: https://en.wikipedia.org/w/index.php?title=Jenks_natural_breaks_optimization&oldid=791523957 (Date Last Accessed September 6, 2017).

[113] R. McMaster and S. McMaster, "A History of Twentieth-Century American Academic Cartography," *Cartography and Geographic Inform. Sci.*, vol. 29, no. 3, pp. 305–321, Mar. 2002.

[114] R. Mihalcea, "Graph-based ranking algorithms for sentence extraction, applied to text summarization," in *Proc. 42st Annu. Assoc. Computational Linguistics*, Barcelona, Spain, 2004, pp. 170–173.

[115] H. Deng, M. R. Lyu, and I. King, "A generalized co-hits algorithm and its application to bipartite graphs," in *Proc. 15th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Paris, France, 2009, pp. 239–248.

[116] E. Peserico and L. Pretto, "Score and rank convergence of HITS," in *Proc. 32nd Int. ACM SIGIR Conf. Res. and Develop. Inform. Retrieval*, Boston, MA, 2009, pp. 770–771.

[117] M.-C. De Marneffe, B. MacCartney, and C. D. Manning, "Generating typed dependency parses from phrase structure parses," in *Proc. 5th Int. Conf. Language Resources and Evaluation*, Genoa, Italy, 2006, pp. 449–454.

[118] Q. Li, H. Ji, and L. Huang, "Joint event extraction via structured prediction with global features," in *Proc. 51st Annual Meeting Assoc. Computational Linguistics*, Sofia, Bulgaria, 2013, pp. 73–82.

[119] Q. Li and H. Ji, "Incremental joint extraction of entity mentions and relations," in *Proc. 52nd Annu. Meeting Assoc. Computational Linguistics*, Baltimore, MD, 2014, pp. 402–412.

[120] K. Bollacker, R. Cook, and P. Tufts, "Freebase: A shared database of structured general human knowledge," in *Proc. 22nd AAAI Conf. Artificial Intell.*, Vancouver, Canada, 2007, pp. 1962–1963.

[121] LDC, "English gigaword fifth edition." [Online]. Available: https://catalog.ldc.upenn.edu/LDC2011T07 (Date Last Accessed October 9, 2017).

[122] 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 *Proc. 24th AAAI Conf. Artificial Intell.*, Atlanta, GA, 2010, pp. 1306–1313.

[123] C.-C. Chang and C.-J. Lin, "LIBSVM: a Library for Support Vector Machines," *ACM Trans. Intell. Syst. and Technol.*, vol. 2, no. 3, pp. 1–27, Apr. 2011.

[124] J. Guo, W. Che, D. Yarowsky, H. Wang, and T. Liu, "Cross-lingual dependency parsing based on distributed representations," in *Proc. 53rd Annu. Meeting Assoc. Computational Linguistics and 7th Int. Joint Conf. Natural Language Process.*, Beijing, China, 2015, pp. 1234–1244.

[125] L. d. Corro, A. Abujabal, R. Gemulla, and G. Weikum, "Finet: Context-aware fine-grained named entity typing," in *Proc. 2015 Conf. Empirical Methods Natural Language Process.*, Lisbon, Portugal, 2015, pp. 868–878.

[126] L. Huang *et al.*, "Liberal Entity Extraction: Rapid Construction of Fine-Grained Entity Typing Systems," *Big Data*, vol. 5, no. 1, pp. 19–31, Mar. 2017.

[127] P. Gamallo *et al.*, "Dependency syntactic parser and formal grammar for natural languages." [Online]. Available: https://github.com/citiususc/DepPattern (Date Last Accessed October 9, 2017).

[128] M. d. P. V. Ibanez and A. Ohtani, "Automatic detection of gender and number agreement errors in spanish texts written by japanese learners," in *Proc. 26th Pacific Asia Conf. Language, Inform. and Computation*, Bali, Indonesia, 2012, pp. 299–307.

APPENDIX A
Slots Table

Table A.1: 41 slot types and their categorization.

Query Type	Slot Name	Content	Quantity
PER	per:alternate_names	Name	List
PER	per:children	Name	List
PER	per:cities_of_residence	Name	List
PER	per:city_of_birth	Name	Single
PER	per:city_of_death	Name	Single
PER	per:countries_of_residence	Name	List
PER	per:country_of_birth	Name	Single
PER	per:country_of_death	Name	Single
PER	per:employee_or_member_of	Name	List
PER	per:origin	Name	List
PER	per:other_family	Name	List
PER	per:parents	Name	List
PER	per:schools_attended	Name	List
PER	per:siblings	Name	List
PER	per:spouse	Name	List
PER	per:stateorprovince_of_birth	Name	Single
PER	per:stateorprovince_of_death	Name	Single
PER	per:statesorprovinces_of_residence	Name	List
PER	per:age	Value	Single
PER	per:date_of_birth	Value	Single
PER	per:date_of_death	Value	Single
PER	per:cause_of_death	String	Single
PER	per:charges	String	List
PER	per:religion	String	Single
PER	per:title	String	List
ORG	org:alternate_names	Name	List
ORG	org:city_of_headquarters	Name	Single
ORG	org:country_of_headquarters	Name	Single
ORG	org:founded_by	Name	List
ORG	org:member_of	Name	List
ORG	org:members	Name	List
ORG	org:parents	Name	List
ORG	org:political_religious_affiliation	Name	List
ORG	org:shareholders	Name	List
ORG	org:stateorprovince_of_headquarters	Name	Single
ORG	org:subsidiaries	Name	List
ORG	org:top_members_employees	Name	List
ORG	org:date_dissolved	Value	Single
ORG	org:date_founded	Value	Single
ORG	org:number_of_employees_members	Value	Single
ORG	org:website	String	Single