# Knowledge Base Population based on Entity Graph Analysis

Md Rashedur Rahman

université PARIS-SACLAY

UNIVERSITÉ PARIS SUD
Comprendre le monde, construire l'avenir

NNT : 2018SACLS092

# Knowledge Base Population based on Entity Graph Analysis

Thèse de doctorat de l'Université Paris-Saclay
préparée à l'Université Paris-Sud

École doctorale n°580
Sciences et Technologies de l'Information et de la Communication

Spécialité de doctorat : Informatique

Thèse présentée et soutenue à Orsay, le 17 avril 2018, par

**Md Rashedur RAHMAN**

Composition du Jury :

| | |
|---|---|
| Chantal Reynaud<br>Professeur, IUT d'Orsay | Président |
| Patrice Bellot<br>Professeur, Université Aix-Marseille | Rapporteur |
| Marc Spaniol<br>Professeur, Université de Caen Basse-Normandie | Rapporteur |
| Florian Boudin<br>Maître de conférences, Université de Nantes | Examinateur |
| Sophie Rosset<br>Directrice de recherche, LIMSI, CNRS | Co-encadrant de thèse |
| Brigitte Grau<br>Professeur, ENSIIE, LIMSI, CNRS | Directeur de thèse |

Thèse de doctorat

Université Paris-Saclay

# *Abstract*

**Knowledge Base Population based on Entity Graph Analysis**

by Md Rashedur RAHMAN

Knowledge Base Population (KBP) is an important and challenging task specially when it has to be done automatically. The objective of KBP task is to make a collection of facts of the world. A Knowledge Base (KB) contains different entities, relationships among them and various properties of the entities.

Relation extraction (RE) between a pair of entity mentions from text plays a vital role in KBP task. RE is also a challenging task specially for open domain relations. Generally, relations are extracted based on the lexical and syntactical information at the sentence level. However, global information about known entities has not been explored yet for RE task. We propose to extract a graph of entities from the overall corpus and to compute features on this graph that are able to capture some evidence of holding relationships between a pair of entities.

In order to evaluate the relevance of the proposed features, we tested them on a task of relation validation which examines the correctness of relations that are extracted by different RE systems. Experimental results show that the proposed features lead to outperforming the state-of-the-art system.

**Résumé:** Le peuplement de base de connaissance (KBP) est une tâche importante qui présente de nombreux défis pour le traitement automatique des langues. L'objectif de cette tâche est d'extraire des connaissances de textes et de les structurer afin de compléter une base de connaissances. Nous nous sommes intéressé à la reconnaissance de relations entre entités.

L'extraction de relations (RE) entre une paire de mentions d'entités est une tâche difficile en particulier pour les relations en domaine ouvert. Généralement, ces relations sont extraites en fonction des informations lexicales et syntaxiques au niveau de la phrase. Cependant, l'exploitation d'informations globales sur les entités n'a pas encore été explorée. Nous proposons d'extraire un graphe d'entités du corpus global et de calculer des caractéristiques sur ce graphe afin de capturer des indices des relations entre paires d'entités.

Pour évaluer la pertinence des fonctionnalités proposées, nous les avons testées sur une tâche de validation de relation dont le but est de décider l'exactitude de relations extraites par différents systèmes.

Les résultats expérimentaux montrent que les caractéristiques proposées conduisent à améliorer les résultats de l'état de l'art.

# Synthèse en français

Aujourd'hui, à l'ère du World Wide Web (WWW), une énorme quantité d'informations sont disponibles dans des formats lisibles par machine. Le texte est l'un des formats les plus courants de publication de contenus dans différents médias et de communication entre les êtres humains. Dans notre vie quotidienne, nous recherchons différentes informations et attendons qu'elles soient disponibles sur demande. Les sources d'information existantes lisibles par machine sont pour la plupart non structurées (par exemple un portail de nouvelles) et semi-structurées (par exemple Wikipédia). Par conséquent, il devient très difficile de trouver des informations utiles immédiatement. Ce problème peut-être résolu en stockant l'information d'une manière structurée.

Une base de connaissances (KB) est une collection structurée d'informations. Elle représente les faits du monde. Un fait dans un KB se réfère à une relation sémantique binaire entre deux entités du monde réel. Les types d'entités du monde réel sont la personne, l'organisation, la localisation, etc. Une KB nous fournit des informations précises sur différentes entités. Ainsi, il facilite la réponse aux questions (Demner-Fushman and Lin, 2007; Fader, Zettlemoyer, and Etzioni, 2014) et le service à la clientèle (Khodakarami and Chan, 2014;Cheung et al., 2003) dans divers domaines. Ces dernières années, l'extraction d'informations factuelles à partir de textes et leur stockage dans une base de données ont été un sujet de recherche très étudié.

Il existe de nombreux KBs où Freebase, DBpedia, YAGO, Wikidata qui sont très connus. Aucune base de connaissances existante n'est complète. En outre, sur le web, de nombreuses informations sont publiées chaque jour dans des textes. Les faits manquants peuvent être collectés à partir des textes libres. Le peuplement de base de connaissances (KBP) est la tâche de recueillir des informations factuelles à partir de textes. C'est une tâche importante et stimulante surtout quand elle doit être faite automatiquement. L'extraction de relations (RE) entre une paire de mentions d'entité à partir du texte joue un rôle essentiel dans la tâche KBP. L'extraction automatique de relation dansun texte est une tâche difficile. Elle est particulièrement complexe lors de la recherche d'un grand nombre de relations sémantiques qui décrivent des entités dans le domaine ouvert. Les systèmes d'extraction de relation existants génèrent un grand nombre de fausses relations entre différentes entités qui rendent une base de connaissances automatiquement remplie moins précise. Par conséquent, nous avons été motivés pour valider les hypothèses de relation qui sont générées par différents systèmes d'extraction de relation. L'objectif est de rejeter un

grand nombre de fausses relations sans affecter les bonnes pour que le score global dans la tâche KBP puisse être amélioré.

Dans la littérature sur la caractérisation des relations, des analyses fiables telles que le vote majoritaire (Sammons et al., 2014) ont été étudiées spécialement lorsque les hypothèses de relation sont générées par plusieurs systèmes. Le vote à la majorité lui-même ne peut pas atteindre un bon score. Par conséquent, des informations supplémentaires telles que les caractéristiques linguistiques sont prises en compte. Les caractéristiques linguistiques traditionnelles (par exemple, lexicales et syntaxiques) se sont avérées utiles pour la tâche de caractérisation des relations, mais elles n'obtiennent pas un très haut score.

Les informations globales sur les entités, c'est-à-dire la manière dont une entité est associée à d'autres entités dans une ressource partagée, peuvent être efficaces pour trouver des indices permettant de valider une relation entre deux entités. Deux entités apparentées partagent des voisins communs dans leurs activités quotidiennes. Dans les activités d'une entité particulière, les autres entités qui participent à ces activités sont considérées comme les voisins de cette entité particulière. Ainsi, une communauté d'une entité est construite par les entités voisines. Les informations globales sur les entités n'ont pas été utilisées pour la tâche de caractérisation des relations. Nous étudions un graphe d'entités qui nous permet d'explorer les informations globales sur les entités et de fournir des informations utiles si une paire d'entités est dans une vraie relation. Nous proposons de calculer plusieurs caractéristiques telles que la similarité du réseau, la densité du réseau, la centralité du vecteur propre et l'information mutuelle pour valider une relation basée sur l'analyse du graphe d'entité. De plus, nous utilisons certaines caractéristiques de la littérature sur la caractérisation des relations et proposons de nouvelles fonctionnalités basées sur l'analyse linguistique.

Nous considérons la validation de relation comme une tâche de classification binaire. Le but est de déterminer si une hypothèse de relation est vraie ou fausse lorsque on donne une paire d'entités, une phrase justifiant la relation ainsi qu'un graphe d'entités. De plus, nous proposons un modèle pour la tâche KBP basé sur notre modèle de validation de relation.

Afin d'évaluer la pertinence des fonctionnalités proposées, nous les avons testées sur une tâche de validation de relations. Nos résultats expérimentaux montrent que les caractéristiques des graphes proposées améliorent significativement la performance de la validation de la relation lorsqu'elles sont combinées avec certaines caractéristiques linguistiques de base. Dans l'ensemble, nous obtenons un score F plus élevé d'environ 10 points par rapport à une combinaison de quatre caractéristiques

linguistiques. De plus, notre méthode de validation de relation est utilisée pour remplir une base de connaissances en validant les hypothèses de relation générées par différents systèmes d'une manière globale. Notre système de KBP basé sur la validation des relations améloire le système KBP de référence la plus performante d'environ $1,27$ point spécialement pour certaines relations sémantiques sélectionnées.

# *Acknowledgements*

First and foremost I would like to express many thanks to my supervisors Brigitte Grau and Sophie Rosset. Their insightful guidelines directed me toward a successful PhD candidate. Both of them spent their precious time for many discussion sessions. Their intellectual suggestions helped me to overcome the hurdles during my research work. I am also thankful to the jury members of my PhD defense. Their appreciation and intellectual criticism are very important for me to become a researcher.

I would also like to express my gratefulness to IRT SystemX for financing my PhD. For the first two years of my PhD, I worked at SystemX as a team member of IMM project. All of the members involved in IMM project deserve my thanks for their cordial association. I offer many thanks to the IMM project manager Olivier Mesnard and other members Jérémy Guillemot, Pierre Ternay, Christian Lautier, Wilson Fred and so on for their support.

In the last year of my PhD, I worked at LIMSI where I found another group of beautiful minds who always supported me with a wonderful research environment. My thanks go to all the members of ILES group. I specially thank Arthur, Sanjay, Zheng, Julien, Arnaud, Swen, Christopher and Charlotte who offered me a nice company by playing Perudo after lunch.

Additionally, all of my friends in Bangladesh and abroad deserve my cordial thanks for their love, well-wishing and support. My special thanks to Zahurul Islam, Sohrab Hossain, Talebull Islam, Shohel Mahmud and Junayed Mahmud who are always beside me in my overseas life.

I am also grateful to all of my teachers at every level of my academic life. I learned many things from them and the acquired knowledge from them lead me to pursue the highest academic degree.

Finally, I am so thankful to all of my family members for their invaluable affection, love, encouragement and support. My endless love to my beloved wife who always inspires me to do something very special.

# Contents

# List of Figures

# List of Tables

*To my family*

# Chapter 1

# Introduction

Today, in the age of information technology, an enormous amount of information is available in machine-readable formats. The text is one of the very common formats of publishing information in different media and communicating among human beings. In our daily life, we search for various information and expect them to be available on demand. Therefore, extracting information automatically has become a very important task. Information extraction from texts has been a very interesting topic in Natural Language Processing (NLP) research since the last couple of decades (Grishman and Sundheim, 1996). The existing machine readable information sources are mostly unstructured (e.g. news portal) and semi-structured (e.g. Wikipedia). Usually, information is extracted automatically from these sources and stored in a structured fashion for further usage. Structured information basically refers to factual information between different types of entities. Such information is very useful in several tasks specially for question answering (Demner-Fushman and Lin, 2007; Fader, Zettlemoyer, and Etzioni, 2014) and customer service (Khodakarami and Chan, 2014;Cheung et al., 2003) in various domains. In recent years, extracting factual information from texts and storing them in a database have been much studied research topic.

Increasing of huge information on the web in different formats opens various dimensions of research works related to information extraction. In the same context, IRT SystemX[1], an Institute for Technological Research (IRT) comes with a project named Multimedia Multilingual Integration (IMM[2]). IMM project aims at developing tools for monitoring user extracted knowledge from unstructured information sources (specially text and audio) in order to make some reports or decisions. One important part of this project is extracting advanced semantic information such as factual information from raw texts. This research represents a part of IMM project aiming at populating a knowledge base from free texts.

A Knowledge Base (KB) is a special-purpose structured collection of information which is readable and manageable by machine. It provides useful information on demand. Usually, a knowledge base comprises factual information in triplets. A fact is a triple *(relation, subject, object)* where the *subject* and *object* refer to entities and *relation* refers to the relation name between the *subject* and *object*. For example, a triplet of *(spouse, Barack Obama, Michelle Obama)* indicates *spouse* relationship between *Barack Obama* and *Michelle Obama*. A fact may also represent a property of an entity. For instance, *(age, Barack Obama, 56)* denotes the age of *Barack*

[1] http://www.irt-systemx.fr/
[2] http://www.irt-systemx.fr/en/project/imm/

*Obama*. The facts we are interested in are basically binary relations between two entities where the types of the entities can be *person*, *organization*, *location* etc.

| Knowledge Base | # Entities | # Relation Types | Facts |
|---|---|---|---|
| Wikidata | 18 M | 1,632 | 66 M |
| YAGO2 | 9.8 M | 114 | 447 M |
| DBpedia | 4.6 M | 1,367 | 539 M |
| Freebase | 40 M | 35,000 | 637 M |
| Yahoo! Knowledge Graph | 3.4 M | 800 | 1,391 M |
| Google Knowledge Graph | 570 M | 35,000 | 18,000 M |

TABLE 1.1: Size of some existing knowledge bases (Li, 2016)

In recent years, several knowledge bases have been constructed where DBpedia (Auer et al., 2007), Freebase (Bollacker et al., 2008), YAGO (Hoffart et al., 2013; Biega, Kuzey, and Suchanek, 2013; Mahdisoltani, Biega, and Suchanek, 2014), DeepDive (Niu et al., 2012), Google Knowledge Graph (Singhal, 2012) etc. are very well-known. These knowledge bases contain millions of facts among different types of entities such as *person*, *organization*, *location* etc. Table 1.1 represents statistics of some existing knowledge bases. Most of the existing knowledge bases represent facts by following the standard of Resource Description Framework (RDF) or similar format. A knowledge base can be represented also graphically where the entities and relation types are denoted by nodes and edges accordingly.

A knowledge base provides us various information that we search for in our daily life. Moreover, knowledge bases are used in the back-end of search engines and some virtual personal assistant applications such as Google Assistant, Apple's Siri, Micorisoft's Cortana, Amazon's Alexa etc.

Knowledge Base Population (KBP) is the task of constructing a knowledge base. KBP has become a highly explored research topic in NLP field since last decade (Auer et al., 2007 Suchanek, Kasneci, and Weikum, 2007; Bollacker et al., 2008). In the existing knowledge bases such as DBpedia[3], Freebase[4], YAGO[5] etc, different properties of entities and relationships between entities have been collected from structured and semi-structured information sources. Different methods have been used for building these knowledge bases. DBpedia was built by crowd-sourced community. In DBpedia project, structured information has been extracted from Wikipedia infoboxes (Lehmann et al., 2015) automatically. However, these infoboxes

[3]http://wiki.dbpedia.org/
[4]https://developers.google.com/freebase/
[5]http://www.yago-knowledge.org/

followed different templates that prone to error in automatic information extraction. These errors were solved by crowd-sourced mapping. However, in Freebase, the data was mainly composed of its community members. Freebase allowed end-users to edit the existing structured information. It also integrated data from other structured information sources which are: Wikipedia, NNDB[6], FMD[7] and MusicBrainz[8]. On the other hand, YAGO was built automatically from Wikipedia, GeoNames and WordNet (Hoffart et al., 2013). Several rules were defined for extracting information automatically from Wikipedia articles.

No existing knowledge base is complete. A lot of facts are missing in the existing knowledge bases. Missing facts can be collected from free texts. However, free texts exist in a large volume and it has been increasing day by day. Therefore, populating a knowledge base has to be done automatically or semi-automatically leading to human curation. In order to construct a knowledge base automatically from free texts, extraction of relation is an important component. Relation Extraction (RE) is a kind of information extraction task which extracts related pairs of entities (subject and object) from texts and characterizes semantic types of the relations. It requires natural language understanding (NLU) of pieces of texts. For instance, a relation extraction system should be able to extract six relationships of three semantic types *spouse*, *children* and *parent* among four entities *Chelsea Victoria*, *Bill Clinton*, *Hillary Clinton* and *Marc Mezvinsky* from the sentence *Chelsea Victoria, the daughter of Bill Clinton and Hillary Clinton got married to Marc Mezvinsky.* as illustrated in Ex 1.1. The expression of a semantic relation mostly depends on trigger words. A trigger word is a content word which is able to characterize the semantic type of a relation when it takes place between two entities. For example, *married* is a trigger word for *spouse* relation.

Automatically extraction of relations from texts is a difficult task. It is particularly complex when searching for a large number of semantic relations that describe entities in the open domain. The semantic relation extraction methods are mostly supervised and a supervised method requires a lot of annotated training examples. Unfortunately, sufficient annotated examples of different types of relations are not available. Moreover, in many cases, relations are expressed in multiple sentences where the entities are mentioned by coreferences. In such cases, resolving coreferences becomes very important. Unfortunately, existing coreference resolution systems do not achieve very high scores and generate a lot of noise. As a consequence, automatic KBP becomes a very challenging task.

[6]http://www.nndb.com/
[7]http://www.fashionmodeldirectory.com/
[8]https://musicbrainz.org/

**Ex 1.1: Relation extraction from text**

***Chelsea Victoria***, *the daughter of* ***Bill Clinton*** *and* ***Hillary Clinton*** *got married to* ***Marc Mezvinsky***.

*Relations between the pairs of entities*

1. *(spouse, Chelsea Victoria, Marc Mezvinsky)*
2. *(spouse, Bill Clinton, Hillary Clinton)*
3. *(children, Bill Clinton, Chelsea Victoria)*
4. *(children, Hillary Clinton, Chelsea Victoria)*
5. *(parent, Chelsea Victoria, Bill Clinton)*
6. *(parent, Chelsea Victoria, Hillary Clinton)*

United States national institute of standards and technology (NIST[9]) has been conducting text analysis conference (TAC[10]) for evaluating automatic KBP task since 2009. In KBP task, a set of queries is given where each query is defined by a subject entity and a relation name. A KBP system has to extract the object entity of a query relation from raw texts. Additionally, it has to provide a relation justifying text to support the query relation between the pair of entities. KBP systems usually employ relation extraction systems to produce relation hypotheses between entities. Most of the automatic KBP systems obtain a very low score (Surdeanu and Ji, 2014). One of the most important reasons for resulting a low score is the poor performance of the relation extraction systems. Relation extraction systems generate a large number of false relationships among different entities that lead KBP systems to result in lower precision (Surdeanu et al., 2012).

## 1.1 Research Objective

KBP systems generate many false responses to the given queries. Some sample responses to a query are illustrated in Ex 1.2. The given subject entity and relation name are *Barack Obama* and *spouse* accordingly. There are three candidates *Michelle Obama*, *Michelle Robinson* and *Hillary Clinton* to be the object of the query relation *(spouse, Barack Obama, ?)*. According to the justification texts, *Michelle Obama* (Response 1) is the correct object since the word *married* expresses the *spouse* relationship between *Barack Obama* and *Michelle Obama*. In Response 2, the text does not justify *spouse* relationship between them. Moreover, in Response

[9]https://www.nist.gov/
[10]https://tac.nist.gov/

3, the relation justifying text expresses *spouse* relationship between *Barack Obama* and *Michelle Obama* but not between *Barack Obama* and *Hillary Clinton*. In KBP evaluation task, Response 2 and 3 are considered as wrong. As a result, these wrong responses cause a system getting a lower score. The score of a system can be improved by discarding the wrong responses. Moreover, several relation extraction systems can be employed to improve the KBP scores.

**Ex 1.2: Responses to a query of spouse relation**

*relation name:* ***spouse***

*subject entity:* ***Barack Obama***

*Response 1:* ***Michelle Obama***

***Michelle Obama*** *is married to* ***Barack Obama****, the 44th president of the United States...*

*Response 2:* ***Michelle Robinson***

*In June 1989,* ***Barack Obama*** *met* ***Michelle Robinson*** *when he was employed as a summer associate...*

*Response 3:* ***Hillary Clinton***

***Barack Obama*** *visited the family of* ***Hillary Clinton*** *with his wife Michelle Obama...*

Our research focuses on validating relation hypotheses which are generated by different relation extraction systems. The objective is to discard a large number of false relations without affecting the correct ones so that overall score in KBP task can be improved. Suppose, we are given a set of queries ($Q$) and a text corpus ($TC$) where each of the queries is defined by a subject entity ($E_q$) and a relation name ($R$), and different relation extraction systems generate a set of candidate objects $\{E_{c_1}, E_{c_2},...,E_{c_n}\} \in E_c$ from $TC$ with justifying text excerpts for each of the candidates. We aim at deciding whether a claimed relation $(R, E_q, E_{ci})$ is correct or wrong for improving KBP scores.

Trustworthy measurements have been introduced for relation validation task when relation hypotheses are generated by multiple systems. As the trustworthy measurements, Sammons et al. (2014) used majority voting and Rodrıguez and Wang (2016) combined several trustworthy signals come from systems, source documents and user beliefs for validating relation in KBP task. However, using such features has not proven to be sufficient.

Linguistic analysis (semantic and syntactic) provides useful information to justify

either a claimed relation is correct or not. However, in some cases, specially when a relation is expressed in a long or complex sentence, linguistic analyzer fails to capture the useful information. Sometimes, natural language is ambiguous and hence, linguistic analysis cannot identify the semantics of a word. As a consequence, it fails to detect the true relationship between a pair of entity mentions in a sentence.

Relation extraction and relation validation can be considered as the opposite sides of the same coin. In relation validation task, several linguistic features have been inherited from the task of relation extraction. Semantic information such as words between the related pair of entity mentions have been inspected by comparing them to a set of pre-collected trigger words for justifying a claimed relation (Yu et al., 2014). Syntactic dependency path has been examined as well. Dependency patterns of relation expression can be studied for validating a claimed relation.

Linguistic analysis captures local information for justifying a relation. Apart from linguistic information, global information about entities, i.e. how an entity is associated with other entities in a shared resource can be effective for finding some clues to validate a relationship between two entities. Two related entities share some common entities in their daily life activities. In the activities of a particular entity, other entities which participate in that activities are considered as the neighbors of that particular entity. Thus a community of an entity is built by the neighbor entities. A graph of entities allows us to explore global information about the entities and it can provide some information if they are in a relationship.

We summarize our research objective by two questions from two different perspectives: entity graph and linguistic analysis which are stated below.

- Q1: Can entity graph analysis provide some clues to validate a relationship between two entities?
- Q2: Can some linguistic features be improved to capture useful information for validating a claimed relation?

## 1.2 Contributions

Important and influential nodes in a graph can be identified by measuring node centralities (Friedl and Heidemann, 2010). Such measurement ranks the nodes in a graph. Graph based ranking Method has been proven effective for validating relation (Jain and Pantel, 2010; Jean-Louis, Besançon, and Ferret, 2011; Singh-Blom et al., 2013). These methods mainly employed PageRank (Brin and Page, 1998) algorithm and Katz measure (Katz, 1953) on a graph for re-ranking the candidates. A graph

also facilitates to perform information-theoretic measurements. Such measurement has been successfully used in knowledge discovery task (Holzinger et al., 2013).

In order to study the impact of the surroundings of entities, we construct a graph of entities from a given corpus. The link between two entities in a graph indicates that the pair of entities are mentioned in the same sentence so that they are somehow related in the real world. We define a community of an entity by the directly connected neighbors in the graph. Fig. 1.1 shows an example of such type of graph where the communities of *Barack Obama*, *Michelle Obama* and *Hilary Clinton* are denoted by the green rectangle, purple circle and orange ellipse accordingly. We analyze the community graphs of entities for identifying some clues of having a relationship between a pair of entities.

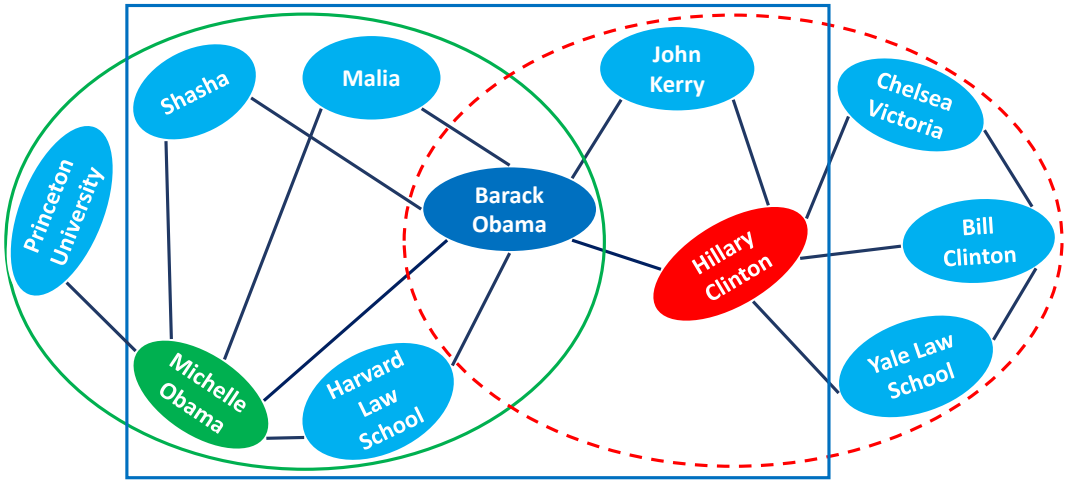


FIGURE 1.1: Community graph

We compute *network density*, *network similarity*, *mutual information* between two communities and *eigenvector centrality* of entity nodes in a community graph. These measurements quantify how two entities are related to each other based on their global association. For instance, *network similarity* quantifies the similarity between the communities of a query entity and of a candidate object based on the number of community members they share. According to Fig. 1.1, the similarity between the communities of *Barack Obama* and *Michelle Obama* is higher than that between the communities of *Barack Obama* and *Hillary Clinton* because the pair (*Barack Obama*, *Michelle Obama*) share three entities while (*Barack Obama*, *Hillary Clinton*) pair has only one entity in common between their communities.

The expression of relation at the sentence level follows some syntactic patterns which can be captured from annotated examples. However, all the patterns of a relation may not be collected due to lack of annotated data. Therefore, we propose to compute edit distance between a dependency pattern under analysis and a list of

learned dependency patterns for justifying the relation under validation. For characterizing the semantic type of a relation, we depend on some state-of-the-art linguistic features that have been successfully used in semantic relation extraction task.

Inspecting the existence of any trigger word between entity mentions plays an important role in relation validation task. However, it may not be possible to collect all the trigger words of a relation due to lack of annotated positive examples. In order to identify the unknown triggers of a relation, we propose to utilize word embeddings for computing similarity between the vectors of a content word and a known trigger word of a relation. Thus our relation validation model takes into account entity level global information as well as linguistic information.

We consider relation validation as a binary classification task. Therefore, we train a binary classifier with the graph features computed on the community graphs and features based on linguistic analysis. Our experimental results show that the proposed graph features improve the performance of relation validation significantly when they are combined with some baseline linguistic features. Overall we gain around $10$ points higher F-score by this combination compared to a baseline of four linguistic features. Furthermore, our relation validation method is employed to populate a knowledge base by validating the relation hypotheses generated by different systems in an ensemble fashion. Our relation validation based ensemble system outperforms the best scoring baseline KBP system by around $1.27$ point on trigger-dependent relations.

## 1.3 Outline

Relation extraction and relation validation are very active research domain. In particular, different linguistic and graph based methods have been proposed to handle these problems. Therefore, we dedicate Chapter 2 to the literature review on knowledge base population, relation extraction, relation validation.

- In Chapter 3, we present how community graphs can be used for validating relations. Firstly, we focus on the definition of the graph of entities and the graph construction. Then different measurements on the community graphs are described. Finally, we describe how a claimed relation between two entities can be validated based on community graph analysis.

- In Chapter 4, we describe the linguistic aspects of expressing and validating semantic relations. In this chapter, firstly, different syntactic modeling are studied. Then we illustrate the lexical semantics for characterizing the type

of a relation. We also study the use of word embeddings for analyzing lexical semantics. Finally, different word embeddings are evaluated in a setting of relation validation task.

- In Chapter 5, we describe our relation validation framework. This chapter summarizes all the features that we use for validating relations. We also describe the corpora used in our research and preprocessing of the corpora. Finally, evaluation metrics of relation validation and KBP tasks are defined.

- In Chapter 6, we present different experiments on relation validation and knowledge base population. Firstly, we build some relation validation models and employ them on KBP SFV task. Then we observe the performance and investigate the limitations of the relation validation models and how we improve the models. Finally, this chapter evaluates the performances of the improved models on both relation validation and KBP tasks.

- Finally, in Chapter 7, we present the conclusion and a summary how we answered our research question. Then this chapter concludes with some possible future research directions.

## Publications

During this research work, we have published our works in some workshops and conferences. The list of publications are given below:

1. Rashedur Rahman, Brigitte Grau and Sophie Rosset. "Impact of Entity Graphs on Extracting Semantic Relations." In: *Lossio-Ventura J., Alatrista-Salas H. (eds) Information Management and Big Data. SIMBig 2017. Communications in Computer and Information Science, vol 795. Springer, Cham.*

2. Rashedur Rahman, Brigitte Grau and Sophie Rosset. "Community graph and linguistic analysis to validate relationships for knowledge base population." In: *4th International Symposium on Information Management and Big Data (SIMBig 2017).*

3. Rashedur Rahman, Brigitte Grau and Sophie Rosset. "Graphe de communauté pour la validation de relations dans le cadre de la population de bases de connaissances." In: *COnférence en Recherche d'Information et Applications (CORIA 2017).*

4. Rashedur Rahman, Brigitte Grau, Sophie Rosset, Yoann Dupont, Jérémy Guillemot, Christian Lautier and Wilson Fred. "TAC KBP 2016 Cold Start Slot

Filling and Slot Filler Validation Systems by IRT SystemX." In: *TAC KBP Workshop 2016*.

5. Rashedur Rahman, Brigitte Grau, and Sophie Rosset. "Graph-Based Relation Validation Method". (Poster) In: *European Knowledge Acquisition Workshop (EKAW 2016)*

6. Olivier Mesnard, Yoann Dupont, Jérémy Guillemot and Rashedur Rahman. "Construction automatisée d'une base de connaissances." (Demo) In: *Traitement Automatique des Langues Naturelles (TALN 2016)*.

# Chapter 2

# Literature Review

Automatic knowledge base population (KBP) from texts is a special kind of information extraction (IE) task in the field of NLP research. It requires extracting factual information which refers to a binary relation between two entities. A binary relation is defined by a tuple of three elements: *relation name*, *subject* and *object*. For example, the *spouse* relationship between *Barack Obama* and *Michelle Obama* is referred by *(spouse, Barack Obama, Michelle Obama)* tuple.

KBP task is simply defined by some slots (or attributes) of different types of entities that have to be filled up by extracting the corresponding values from texts automatically. *Slot filling* task was first introduced in the 6th Message Understanding Conference (MUC-6) (Grishman and Sundheim, 1996). A slot is defined by a relation name and the subject of that relation. In slot filling task, the objective is to find the object of that relation. For example, *(spouse, Barack Obama, ?)* refers to a slot for slot filling task which has to be filled with the correct object value, *Michelle Obama*. U.S. National Institute of Standards and Technology (NIST) has been conducting a workshop named Text Analysis Conference (TAC) for evaluating KBP task since 2009 (McNamee and Dang, 2009; Ji et al., 2010; Surdeanu, 2013; Surdeanu and Ji, 2014). TAC defines slots of *person* and *organization* typed entities that count more slots compared to the number of slots defined in MUC-6.

Relation extraction (RE) plays a vital role in slot filling task. RE is the task of identifying the semantic type of relation between the subject and object entities from a text where the entities are mentioned. It can be formally defined as *(subject, object, relation?)* when a subject-object pair and a text are given. Suppose, given a sentence *Barack Obama is married to Michelle Obama* and two entities *Barack Obama* (subject) and *Michelle Obama* (object), RE task requires to identify the semantic type of the relation between these entities and a RE system should return the relation name, spouse.

A slot filling system employs one or more relation extractors in order to find out the filler values from texts. It is a very challenging task. Performances of the existing slot filling systems are affected by a large number of wrong filler values due to false relation extraction. Therefore, relation validation, a task of validating extracted relations becomes important for improving slot filling scores in KBP task. Relation validation can be defined as a binary classification task to say correct or wrong for provided a relation name, a pair of subject and object and a sentence. For example, given a sentence *Barack Obama is married to Michelle Obama* and a claimed relation *(spouse, Barack Obama, Michelle Obama)*, a relation validation system has to justify the claimed relation as correct if the sentence really expresses

the relation, otherwise wrong.

In this research, we focus on relation validation for improving slot filling scores in KBP task. Slot filling and relation validation tasks are different forms of relation extraction task. Therefore, in this chapter, we present the backgrounds and state-of-the-arts of slot filling, relation extraction and relation validation tasks. Since relation extraction task requires different kinds of linguistic analysis we study several linguistic features of relation extraction based on syntactic and semantic analysis that can be used for relation validation task. Moreover, we explore global information about entities based on their community graphs for validating relations. Therefore, in this chapter, we also study state-of-the-arts of some graph based relation validation methods and some features computed on a graph. Finally, this chapter is concluded by adopting some features from the related tasks and proposing some new features for relation validation task.

## 2.1 Slot Filling Task

A *slot* basically refers to an attribute of an entity. For example, the slot *per:employee_of* denotes an organization where a person works as an employee. In this case, the organization name is the value of the *employment* attribute of that person.

Slot filling task requires filling the defined slots of different entities in a required format. Usually, the filler type (or types) of a slot is predefined. For example, the filler type of *per:employee_of* is *organization* which means *per:employee_of* has to be filled by the name of an organization or a company.

In MUC-6, the main focus was on information extraction from text messages. There was a special task named *scenario template* where several templates of different types of objects *PERSON*, *ORGANIZATION*, *ARTIFACT* etc. were defined. *PERSON* and *ORGANIZATION* templates individually counted 5 and 9 slots accordingly as listed in Table 2.1. The objective was to fill up the templates by extracting relevant information from text messages. In most cases (i.e. PER_NAME, ORG_NAME etc), the type of a filler is not a named entity but a string. However, some slots hold values of specific types of named entities. For example, ORG_LOCALE slot required to be filled up by a location name {CITY, PROVINCE, COUNTRY, REGION, UNKNOWN}. Detail description of the templates and slots are available on the MUC-6 website[1]. In *scenario template* task, a relation between two entities was encoded as an attribute of a template. To solve this problem, MUC-7 (Chinchor and Marsh, 1998) introduced *template relation* task by including LOCATION_OF, EMPLOYEE_OF, and PRODUCT_OF slots to break out the relations or facts from the

[1] http://www.cs.nyu.edu/cs/faculty/grishman/muc6.html

| Object | Slot Name | Filler Description |
|---|---|---|
| PERSON | PER_NAME | "NAME" |
| | PER_ALIAS | "ALIAS" |
| | PER_TITLE | "TITLE" |
| | OBJ_STATUS | {OPTIONAL} |
| | COMMENT | " " |
| ORGANIZATION | ORG_NAME | "NAME" |
| | ORG_ALIAS | "ALIAS" |
| | ORG_DESCRIPTOR | "DESCRIPTOR" |
| | ORG_TYPE | GOVERNMENT, COMPANY, OTHER |
| | ORG_LOCALE | LOCALE-STRING {LOC_TYPE} |
| | ORG_COUNTRY | NORMALIZED-COUNTRY |
| | ORG_NATIONALITY | NORMALIZED-COUNTRY-or-REGION |
| | OBJ_STATUS | {OPTIONAL} |
| | COMMENT | " " |

TABLE 2.1: PERSON and ORGANIZATION templates of *scenario template* task of MUC-6

templates. Thus each slot represented a binary relation. A binary relation is defined by a tuple *(r, arg1, arg2)* where *r*, *arg1* and *arg2* refer to the relation name, subject and object accordingly. For example, *(spouse, Barack Obama, Michelle Obama)* indicates a *spouse* relationship between two persons *Barack Obama* and *Michelle Obama*.

In TAC KBP-2014 slot filling task (Surdeanu and Ji, 2014), several slots were defined for PERSON and ORGANIZATION entities. PERSON and ORGANIZATION individually counted 25 and 16 slots accordingly as shown in Table 2.2. In most cases, the filler types are named entities. For example, *per:spouse*, *per:employee_of* etc have to be filled by *person* and *organization* typed entities accordingly. TAC provides an evaluation corpus and slot filling queries. A query is defined by an entity name, a corresponding document and a slot name as shown in Ex. 2.1. A slot filling system has to respond with the filler values including relation justification texts (from the given corpus). Detail descriptions of the slots, filler types, query and response formats are available in KBP-2014 website[2]. Definition and number of slots of the KBP slot filling task may vary in different years. For example, in KBP-2015 slot filling task, inverse slots (such as *gpe:births_in_country* is the inverse slot of *per:country_of_birth*) had been introduced. More details about the inverse slot can

[2]http://surdeanu.info/kbp2014/def.php

| Person Slots | | | Organization Slots | | |
|---|---|---|---|---|---|
| **Name** | **Type** | **List?** | **Name** | **Type** | **List?** |
| per:alternate_names | Name | Yes | org:alternate_names | Name | Yes |
| per:date_of_birth | Value | | org:political_religious_affiliation | Name | Yes |
| per:age | Value | | org:top_members_employees | Name | Yes |
| per:country_of_birth | Name | | org:number_of_employees_members | Value | |
| per:stateorprovince_of_birth | Name | | org:members | Name | Yes |
| per:city_of_birth | Name | | org:member_of | Name | Yes |
| per:origin | Name | Yes | org:subsidiaries | Name | Yes |
| per:date_of_death | Value | | org:parents | Name | Yes |
| per:country_of_death | Name | | org:founded_by | Name | Yes |
| per:stateorprovince_of_death | Name | | org:date_founded | Value | |
| per:city_of_death | Name | | org:date_dissolved | Value | |
| per:cause_of_death | String | | org:country_of_headquarters | Name | |
| per:countries_of_residence | Name | Yes | org:stateorprovince_of_headquarters | Name | |
| per:statesorprovinces_of_residence | Name | Yes | org:city_of_headquarters | Name | |
| per:cities_of_residence | Name | Yes | org:shareholders | Name | Yes |
| per:schools_attended | Name | Yes | org:website | String | |
| per:title | String | Yes | | | |
| per:employee_or_member_of | Name | Yes | | | |
| per:religion | String | Yes | | | |
| per:spouse | Name | Yes | | | |
| per:children | Name | Yes | | | |
| per:parents | Name | Yes | | | |
| per:siblings | Name | Yes | | | |
| per:other_family | Name | Yes | | | |
| per:charges | String | Yes | | | |

TABLE 2.2: Slots of PERSON and ORGANIZATION for the task of KBP slot filling in 2014 (Surdeanu and Ji, 2014)

be learned from the TAC KBP website[3].

In most cases, slots of the KBP slot filling task represent relations between a pair of named entities. Therefore, extraction of relation between named entities plays an important role in KBP slot filling task.

**Ex 2.1: An example of KBP slot filling query**

```
<query id="SF_004">
    <name>Nelson Mandela</name>
    <docid>a69c5c79caa4c2b2869775fabcbabc7f</docid>
    <beg>174</beg>
    <end>187</end>
    <enttype>per</enttype>
    <slot>per:spouse</slot>
</query>
```

[3] https://tac.nist.gov//2016/KBP/ColdStart/guidelines.html

## 2.2 Slot Filling Systems

Slot filling task requires extraction of relation between entities. Relation extraction for slot filling differs to traditional relation extraction task like ACE to some extents (Aguilar et al., 2014). In ACE, relation extraction task required to detect and characterize the relation type between two entities for a given sentence and a pair of entity mentions. In slot filling task, the goal is to find the object entity or entities for a given relation name and a subject entity where the entity types are predefined. For example, in Ex. 2.1, *Nelson Mandela* refers to the subject entity and *per:spouse* indicates the relation name or slot. This slot has to be filled up by a person name who is the spouse of Nelson Mandela. In addition, slot filling requires a system to justify the claimed relation by providing a justification text along with the filler value as discussed in Section 2.1. Traditional evaluation of relation extraction tasks such as ACE and SemEval provide annotated training data which makes this task completely supervised. In KBP slot filling task, no annotated training data is provided that makes the task harder compared to the traditional relation extraction task.

Several methods of slot filling have been proposed during the last couple of years where most of them employed distant supervision (Craven and Kumlien, 1999; Bunescu and Mooney, 2007; Mintz et al., 2009) based relation extraction models (Wiegand and Klakow, 2013; Nguyen et al., 2014; Roth et al., 2014; Angeli et al., 2014; Angeli et al., 2015; Sterckx et al., 2015; Adel and Schütze, 2015; Zhang et al., 2016). A distant supervision method uses an existing knowledge base to collect facts. A fact is a tuple which consists of two entities and a relation name. Any sentence containing the pair of entities is considered as an example of that particular relation. Thus distant supervision facilitates to generate a large number training examples for extracting relations in a supervised fashion. Distant supervision suffers from inappropriate alignment of a sentence to a fact in an existing knowledge base (Riedel, Yao, and McCallum, 2010) and involving multiple relations between a pair of entities (Hoffmann et al., 2011; Surdeanu et al., 2012). Therefore, distant supervision based relation extraction methods are usually trained on noisy data. As a consequence, they generate a large number of false relations which result in lower score in slot filling task.

In addition to the distant supervision, hand-coded patterns have been used by Angeli et al. (2014), Nguyen et al. (2014), Angeli et al. (2015) and Sterckx et al. (2015) for extracting the object of a relation in slot filling task. A rule based inference learning method creates new knowledge from known facts by applying some reasoning. For example, if a person, $P$ lives in a city, $C$ and $C$ is a city of state $S$, it can be implied that $P$ lives in $S$. Bentor et al. (2013), Nguyen et al. (2014) and Zhang et al.

(2016) employed such method to extract relations implicitly for slot filling task.

Precision and recall of relation extraction are affected by distant supervision due to noisy training data and hand coded patterns accordingly (Angeli et al., 2015). Therefore, Angeli et al. (2015) proposed bootstrapped self training method for the relation extraction component in their slot filling system. In bootstrapped learning, a system is firstly trained on a small clean dataset and tested on a large noisy dataset. Then it is iteratively trained by the correctly predicted examples. Thus, it takes advantages of distant supervision and pattern based methods. Lin et al. (2014) incorporated a conditional random field (CRF) model with patterns for extracting slot fillers. A multi-dimensional truth finding model (MTM) (Yu et al., 2013) model has been proposed for slot filling that computes credibilities of the fillers, sources and relation extraction systems which has been continuously achieving a competitive score. This model is extended by a temporality-based clustering mode (TBCM) and active learning (Hong et al., 2014) in 2014. Yu et al. (2016) proposed a graph based trigger driven slot typing method which closely explored the dependency tree structures and ranked the candidate trigger words by PageRank algorithm for slot filling. A couple of systems (Adel and Schütze, 2015; Zhang et al., 2016) employed neural networks for relation extraction module which obtain better scores compared to some others.

However, still all of the KBP slot filling systems suffer from poor F-score (Surdeanu, 2013; Surdeanu and Ji, 2014). The performances of some top ranked slot filling systems in recent years are shown in Table 2.3. In 2013 and 2014, the highest F-score was around $37\%$ while it was between $26\%$ and $27\%$ in 2015 and 2016. Slot filling scores of the top ranked systems are decreasing over the years. It might be due to some special requirements added to the evaluation task each year, due to lack of annotated training data and due to the changes in evaluation corpus. All the top ranked slot filling systems fails to achieve a decent score even though by using effective relation extraction models of literature. All these scores indicate that extracting entity level relations for slot filling is still a challenging task in NLP research.

## 2.3 Relation Extraction

Relation extraction module is a very important component of a slot filling system. Relation extraction refers to finding semantic relationships among given arguments (or entities) and characterizing the relation types. A relation between two arguments is called a binary relation. In this section, only binary relation will be studied. A binary semantic relation can be trigger-dependent or trigger-independent. A *trigger* is a word which strongly represents the semantics of a particular relation. For instance,

| Slot Filling System | Relation Extraction Method | F-score |
|---|---|---|
| Wiegand and Klakow (2013) | distant supervision (DS) | 37.28 |
| Yu et al. (2013) | MTM | 33.89 |
| Li et al. (2013) | pattern bootstrapping+trigger words | 32.27 |
| Angeli et al. (2014) | DS+MIML+patterns | 36.77 |
| Hong et al. (2014) | MTM+TBCM+active learning | 34.11 |
| Lin et al. (2014) | CRF+patterns | 30.53 |
| Angeli et al. (2015) | DS+patterns+SVM+LSTM | 26.70 |
| Sterckx et al. (2015) | DS+patterns | 23.20 |
| Adel and Schütze (2015) | patterns+SVM+CNN+RNN | 21.21 |
| Zhang et al. (2016) | DS+rules+patterns+SVM+LSTM | 22.00 |
| Chang et al. (2016) | DS+LSTM | 17.40 |

TABLE 2.3: TAC official scores of some top ranked slot filling systems in recent years

*wife* is a trigger word of *spouse* relation. A relation (i.e. *spouse*) that cannot be expressed in the text without any trigger word (i.e. *wife, husband, married*), we call it trigger-dependent relation. However, a trigger-independent relation (i.e. *residency*) relationship between a *person* and a *city* can be expressed without any trigger word. Different methods and features have been explored for relation extraction task. Here we give an overview of relation extraction methods and then we will illustrate some relation characterization models that are important for relation validation task.

### 2.3.1 Relation Extraction Methods

During the last couple of decades, different methods of relation extraction have been studied. Relation extraction methods are basically classified into two types: unsupervised and supervised.

In unsupervised methods (Rosenfeld and Feldman, 2006; Banko et al., 2007; Rosenfeld and Feldman, 2007a; Fader, Soderland, and Etzioni, 2011), pairs of entities are collected based on their co-occurrences. Then, the pairs of entities are clustered by extracting features at the sentence level automatically. Each cluster represents a relation. Unsupervised methods do not require any prior knowledge about the relation types. Such methods are useful for open relations where the precise semantic type of a relation is not important.

However, in supervised relation extraction, a system learns expression of a relation and characterization of the relation type from an annotated dataset. Usually,

the annotated dataset contains sentences of different relation types. Each sentence consists of at least one pair of entities and expresses a particular relation between them.

The expression of a binary relation follows some lexical and structural patterns between two arguments. Such patterns are repeated to mention that relationship between another pair of arguments. Subject-verb-object (SVO) is the simplest pattern of expressing relation. Such pattern was used for extracting events (Yangarber et al., 2000) and to capture hypernym relations (Snow, Jurafsky, and Ng, 2005). Regular expression patterns have been used by Hearst (1992) for hyponym relation extraction. Mostly, the pattern based relation extraction methods use POS-tag patterns (Fader, Soderland, and Etzioni, 2011) and lexico-syntactic patterns (Alfonseca et al., 2012; Pershina et al., 2014). However, in natural language, relations are expressed by many diverse patterns and it is not possible to capture all of them. As a consequence, pattern based methods suffer from low recall even though they achieve very high precision. In order to solve this problem feature based methods have been explored.

In feature based method, relation extraction is considered as relation classification task. Different features are computed on the annotated examples. A feature based method predicts an instance whether it expresses a specific type of relation by one of two possible ways: by computing similarity between the instance and annotated examples (Zelenko, Aone, and Richardella, 2003; Culotta and Sorensen, 2004; Bunescu and Mooney, 2005; Bunescu and Pasca, 2006) or by training a classifier model with the feature vectors of annotated examples (Kambhatla, 2004; GuoDong et al., 2005; Jiang and Zhai, 2007).

Supervised relation extraction methods require a large number of annotated examples to obtain a decent score. Manual annotations, as being expensive and time consuming, cannot provide sufficient data for training models of diverse relation types. Distant supervision (Bunescu and Mooney, 2007; Mintz et al., 2009) comes into play to solve this problem. However, distant supervision suffers from noisy training data that generates a lot of false relations.

Traditional supervised relation extraction or classification methods require a lot of feature engineering. In recent years, neural network based supervised relation classification methods (Zeng et al., 2014; Xu et al., 2015; Vu et al., 2016; Zheng et al., 2016; Dligach et al., 2017) have been popular in that they do not require any feature engineering. Such methods take positive and negative examples of relations as input and use word embeddings (Mikolov et al., 2013; Pennington, Socher, and Manning, 2014) to learn lexical features automatically. Neural network based methods learn also the structural representation of words in a sentence to express a relation.

| | Method | Dataset | F-score |
|---|---|---|---|
| Zhang et al. (2006) | supervised (kernel) | ACE | 72.1 |
| GuoDong et al. (2005) | supervised (SVM) | ACE | 74.7 |
| Zeng et al. (2014) | supervised (CNN) | SemEval | 82.7 |
| Zheng et al. (2016) | supervised (CNN+LSTM) | SemEval | 83.8 |
| Xu et al. (2015) | supervised (CNN) | SemEval | 85.6 |
| Mintz et al. (2009) | distant supervision | Riedel et al. (2010) | 40.0 |
| Surdeanu et al. (2012) | distant supervision | Riedel et al. (2010) | 42.6 |

TABLE 2.4: Relation extraction performance by different methods on different datasets

Table 2.4 illustrates the performances of some relation extraction methods evaluated on different datasets. Supervised methods of relation extraction systems with manually annotated training dataset obtain very good scores. However, distant supervised methods cannot achieve a good score even though by using similar features used in supervised methods. Interestingly, neural network based relation extraction methods achieve better performance than other supervised methods on these datasets.

### 2.3.2 Linguistic Features for Relation Characterization

Almost, all the feature based relation extraction methods extract different features based on the syntactic and semantic analysis. Basically, these analyses are performed at the sentence level. The syntactic analysis focuses on the grammatical representation of a sentence. On the other hand, semantic analysis emphasizes on understanding the meaning of a sentence.

#### 2.3.2.1 Syntactic Analysis

Syntactic dependency expresses the grammatical relationship among the words in a sentence. Moreover, syntactic dependency path between two related words indicates the structure of expressing a relation.

Usually, a relation between two entities is expressed in a shorter context. Therefore, shortest dependency path has been proven effective for kernel based relation extraction (Bunescu and Mooney, 2005; Zhang et al., 2006). Neural network based relation classification methods (Cai, Zhang, and Wang, 2016; Liu et al., 2015) used syntactic dependency labels for capturing features in the shortest path automatically.

However, Zhou et al. (2007) argued that in many cases shortest path trees cannot capture enough information for extracting relations. They proposed a context-sensitive shortest path to include necessary information outside the shortest path. In order to capture useful context, Culotta and Sorensen (2004) proposed smallest common subtree and Chowdhury, Lavelli, and Moschitti (2011) proposed minimal subtree for extracting relations.

Consecutive dependency labels in the shortest path between two related entities make a pattern of a relation. Such patterns could be useful for trigger-independent relation extraction. Several patterns have been studied for extracting relation from texts. Pershina et al. (2014) extracted dependency patterns of different relations where maximum pattern length of 3 was found most effective. A SVO pattern has been used by Snow, Jurafsky, and Ng (2005) for extracting hypernym relations.

Expression of a relation often includes a verb between the subject and object. Parts-of-speech (POS) tags provide useful information to identify syntactic roles of the words in a sentence. Therefore, POS-tags have been widely used in pattern based (Hearst, 1992; Fader, Soderland, and Etzioni, 2011), kernel based (Nguyen, Moschitti, and Riccardi, 2009) and feature based (Mintz et al., 2009; Surdeanu et al., 2012) relation extraction. (Mintz et al., 2009) used windows of POS-tags between two arguments, before the first argument and after the second argument. Subgraphs of syntactic parse trees with POS-tags have been explored by (Jiang and Zhai, 2007) for extracting relation at the sentence level.

The syntactic analysis provides useful information for learning grammatical structures of relation expression. However, in most cases, characterizing the semantic type of a relation is not possible by such information.

#### 2.3.2.2 Semantic Analysis

Semantic analysis facilitates understanding the meaning of a text. Since a relation is usually expressed in a sentence, semantic analysis is important to interpret what type of relation it mentions.

Words between and around the mentions hold useful information to characterize the relation type specially for trigger-dependent relations. Therefore, such lexical information has been widely used for learning relation types. Kambhatla (2004), Jiang and Zhai (2007) and Mintz et al. (2009) used words between the arguments of a relation. Moreover, the first word before the first argument and the first word after the second argument have been taken into account in addition to the words between the arguments by GuoDong et al. (2005). Mintz et al. (2009) included windows of k-words before the first argument and after the second argument in their distant supervision based relation learning model. Such windows of words have been inherited in

some other studies (Riedel, Yao, and McCallum, 2010; Riedel, Yao, and McCallum, 2010; Hoffmann et al., 2011; Surdeanu et al., 2012).

Traditional systems learn which words are useful for identifying the type of a relation. These systems cannot handle unseen and sparse words in the training data. However, neural network based relation classification methods (Zeng et al., 2014; Nguyen and Grishman, 2015) take the sequence of words as input and perform semantic analysis based on word embeddings. Word embeddings facilitate characterizing the semantic type of a relation by computing semantic similarity between words.

### 2.3.3 Collective and Statistical Analysis for Relation Extraction

Linguistic analysis is important for extracting relation at the sentence level. Relationship between two entities also depends on their co-existence and common resources between them. Such information cannot be explored by linguistic analysis. In relation validation task, corpus level studies e.g. co-occurrences of two entities and their sharing resources can be taken into account which we call collective analysis.

Collection level information has been explored for improving the performance of relation extraction by learning the boundaries of relation arguments (Rosenfeld and Feldman, 2007b). Augenstein (2016) has taken into account global information about the object of a relation such as object occurrence, markup link with the object, title of the document containing the object etc. for web relation extraction.

The statistical analysis gets importance for extracting relation in a collective manner. Niu et al. (2012) performed statistical inference on diverse data for learning relation. A probabilistic model of inference has also been explored by (Fang and Chang, 2011). Such model counts co-occurrences of the subject-object pairs, frequencies of the relational tuples and patterns and their probabilities. Co-occurrence context has also been quantified by measuring mutual information for extracting relation between entities in the web (Xu et al., 2014).

### 2.3.4 Conclusion

Supervised relation extraction systems with manually labeled training data have shown good results. However, in some special kind of relation extraction task, such as slot filling, no annotated data is provided for training a system. In such cases, systems employ distant supervision for training the relation extraction module. Such relation extraction methods cannot result in a sound score. As a consequence, slot filling system becomes harder and suffers to achieve a decent score. Moreover, traditional relation extraction systems focus on the relation between mentions of entities at the

sentence level. In contrast, slot filling task for KBP requires extraction of a relation at the entity level with a justification of the relation at the sentence level. This is why slot filling task differs from traditional relation extraction task.

Different features have been studied in relation extraction task where lexical features such as words between and around entity mentions and POS-tags, and syntactic feature such as dependency labels are used very commonly. Different patterns are also used for extracting relations. Patterns or dependency paths are generally made of syntactic dependency and lexical information. It is difficult to generalize the patterns and capture similarities among them because of syntactic and lexical variations together.

## 2.4 Relation Validation

Slot filling systems generate a large number of false candidate objects for a query relation with given subject. Performance of the slot filling systems can be improved by discarding the false candidates and it can be done by validating the claimed relations.

Relation validation is the task of selecting the correct object(s) of a relation among several candidates which can be generated by single or multiple systems. Basically, the objective is to discard wrong relation hypotheses by performing further investigation on the already extracted relations. Different relation validation methods have been studied in the literature which are described in the following sub-sections.

### 2.4.1 Ensemble Learning for Relation Validation

The output of a single learning system is affected by statistical, computational and representational problems that can be partly overcome by an ensemble learning (Dietterich, 2002). Therefore, outputs of different relation extraction systems can be aggregated for further processing as an ensemble manner.

#### 2.4.1.1 Ensemble Method

An ensemble method makes some decisions or generates output by performing some analysis on the outcomes of multiple systems. Ensemble methods have been well studied in machine learning. Bagging (Breiman, 1996), stacking (Wolpert, 1992), boosting (Freund and Schapire, 1995) etc. are widely used ensemble machine learning algorithms where bagging and boosting basically count majority votes of the outcomes by different learning algorithms trained on different subsets (bootstrapped

samples) of the training data. However, in stacking outputs of the first round bootstrap-trained algorithms are fed to train an algorithm at the second round to learn the final outcome.

Voting is another kind of ensemble method which takes into account agreements among the outputs of different systems. Such methods have been studied in many decision making tasks (Polikar, 2006; Boroushaki and Malczewski, 2010; Morais and Almeida, 2012; Cao et al., 2012)

Ensemble methods have been successfully used in many information extraction tasks. Yang et al. (2010) explored using ensemble methods to solve various bioinformatics problems such as identifying the interaction between genes, predicting regulatory elements from DNA and protein sequences etc and for identifying effective features. In order to recognize spoken emotions, Morrison, Wang, and De Silva (2007) employed both stacking and voting based ensemble methods. Moreover, Jean-Louis, Besançon, and Ferret (2011) performed majority voting on the outputs of three different methods for template filling task. Their voting method obtained better result compared to any individual method.

#### 2.4.1.2 Slot Filler Validation

In the series of KBP workshop, every year TAC operates several tracks focusing on different problems. Slot filler validation (SFV)[4] is one of the tracks which emphasizes on validating relations for improving slot filling scores. Basically, a SFV system examines whether a response of a slot filling system holds any evidence to justify a claimed relation in the response. Thus SFV implies the task of relation validation.

In slot filling task, a system generates outputs regarding some given queries. When multiple systems respond to the same set of queries, the system outputs can be analyzed to select the correct responses by validating them in an ensemble fashion. Viswanathan et al. (2015) used stacking classifier to improve the KBP slot filling performance. They trained the classifier by the offsets and confidence scores of the responses of different slot filling systems. A comparative study of different models for slot filling was done by Adel, Roth, and Schütze (2016) which found a combination of state-of-the-art and neural network models achieves a higher score than any single model. Therefore, outcomes of different slot filling systems have been taken into account by Wang et al. (2013) and Sammons et al. (2014) for SFV task. Sammons et al. simply counted majority voting of the fillers for the same purpose and achieved F-score between $45.70$ and $48.0$. In contrast, Wang et al. used confidence scores of the responses by different slot filling systems to solve a constraint optimization problem for validating the responses. Moreover, contributions of the systems, referenced

[4] https://tac.nist.gov//2016/KBP/SFValidation/index.html

| SFV System | Dataset | F-score |
|---|---|---|
| Yu et al. (2014) | TAC KBP-2013 | 61.72 |
| Sammons et al. (2014) | TAC KBP-2014 | 48.00 |
| Rodriguez, Goldberg, and Wang (2015) | TAC KBP-2015 | 34.83 |
| Rodrıguez and Wang (2016) | TAC KBP-2016 | 32.42 |

TABLE 2.5: Some top ranked SFV systems in recent years

documents and filler values corresponding to a slot filling query have been taken into account by Yu et al. (2014). They employed a multi-dimensional truth finding model to compute the credibility of a system, document and filler value. In their method, some linguistic indications also have been used such as filler type and inspection of trigger words and dependency path length in the responded relation justifying text etc.

A bipartite graph-based consensus maximization (BGCM) method has been proposed by Rodriguez, Goldberg, and Wang (2015) that combines the outputs of supervised stacked ensemble methods and slot filling runs. This method outperforms all other ensemble methods and the best slot filling run on 2015 KBP slot filling dataset. This method has been extended in Rodrıguez and Wang (2016) where consensus maximization technique is employed over multiple knowledge bases. This two SFV methods did not use any linguistic information but achieved better score compared to the best SF systems.

The performances of some top ranked SFV ensemble systems are shown in Table 2.5. The highest F-score of 61.72 was achieved on KBP-2013 dataset. However, the best SFV F-scores on KBP-2014, KBP-2015 and KBP-2016 datasets were 48.00, 34.83 and 32.42 accordingly. On the other hand, the best F-scores of different SF systems on the same datasets (KBP-2013 to KBP-2016) were 37.28, 36.72, 28.75 and 27.03 accordingly. Although SFV systems are improving the scores over the SF systems, the scores of SFV systems are also decreasing over the years which indicates that SFV is also a very challenging task.

### 2.4.2 Graph based Methods for Relation Validation

Mentions of entities in a collection of texts can be represented in a graph according to their co-existences in texts and sharing resources between them. Such graph facilitates to explore how the related entities are associated and what common information they share.

#### 2.4.2.1 Candidate Ranking Model

Several studies explored graph based ranking model to extract keywords and key-phrases for document summarization task (Mihalcea and Tarau, 2004; Litvak and Last, 2008; Bougouin, Boudin, and Daille, 2013). These methods basically represented texts in a graph and ranked words and phrases by HITS (Kleinberg, 1999) and PageRank (Brin and Page, 1998) algorithms.

Ranking graph nodes also plays an important role in entity linking task (Shen, Wang, and Han, 2015). Hachey, Radford, and Curran (2011) ranked candidate entities by measuring degree centrality and PageRank scores in a graph constructed from Wikipedia articles. Rao, McNamee, and Dredze (2013) and Alhelbawy and Gaizauskas (2014) took into account popularity of the entity nodes for entity linking task. They ranked the candidates in terms of popularity by using PageRank.

Graph based ranking method has been proven effective when there exist several object candidates for a relation, and the objective is to re-rank the candidates. Jain and Pantel (2010) and Jean-Louis, Besançon, and Ferret (2011) re-ranked the filler values by using PageRank algorithm. Singh-Blom et al. (2013) employed Katz (Katz, 1953) measure on a graph for predicting and validating gene-disease relation.

#### 2.4.2.2 Measurements on a Graph

A graph represents different objects as nodes and a relationship between two nodes can be expressed by a link between them. Such representation facilitates performing different measurements.

Identifying influential nodes in a graph is a similar task to ranking the nodes. Eigenvector centrality (Bonacich and Lloyd, 2001) measures the influence of a node in a network. It basically hypothesizes that a node will be even more influential if it is connected to other influential nodes. The concept of eigenvector centrality has been used to rank sentences for text summarization task (Erkan and Radev, 2004). Han, Sun, and Zhao (2011) measured evidence propagation based on semantically related neighbor entities for entity linking task in a collective manner. Such concept of neighbor can be extended to community graphs of entities in the same sentence. These community graphs can be analyzed for characterizing a relation between two entities. Yang and Leskovec (2012) argued that the density of two overlapping communities would be higher than the density of two non-overlapping communities. Therefore, it could be applied to the communities of two entities in a true relationship.

Information theoretic measurements have been explored in complex networks for knowledge discovery (Holzinger et al., 2013) and for detecting community structures (Rosvall and Bergstrom, 2007). Rosvall and Bergstrom maximized the mutual information between a network and a descriptor to detect a community structure. Thus relatedness between two entities can be quantified by measuring mutual information between their communities.

### 2.4.3 Summary

Relation validation task refers to justifying a claimed relation. It becomes important for discarding the wrong fillers in KBP slot filling task. In SFV task, slot filling scores have been improved significantly by validating relations. Several methods of relation validation have been studied in the literature specially for KBP task such as ensemble method, constraint optimization, consensus maximization, multi-dimensional truth finding method etc. The linguistic indication such as the existence of trigger word has also been inspected for validating relations. Moreover, graph based methods such as candidate ranking and centrality measurement have been studied for relation validation task. When there exist several candidate objects of a relation, PageRank has been used for selecting the best object. Katz measurement on a graph has been explored for validating gene-disease relations. These graph based studies indicate that a graph holds effective evidence of justifying relation between two entities.

## 2.5 Conclusion

Slot filling systems for knowledge base population require semantic relation extraction. Supervised relation extraction systems lack sufficient labeled training data. Therefore, existing slot filling systems employ distant supervision for training the relation extraction component. Distant supervision labels sentences based on the facts in an existing knowledge base. Labeling sentences by distant supervision is often erroneous that creates noisy training data. A relation extractor trained with noisy data generates a large number of false relations. As a consequence, slot filling systems suffer to achieve a decent score.

Relation validation comes into play for discarding false relations. The objective of a relation validation system is to remove incorrect candidates by keeping correct ones so that the overall score can be improved in KBP task. An ensemble relation validation method takes outcomes of different slot filling systems as input and applies

some heuristic model to generate a better output. Such methods have shown performance improvement over individual slot filling systems for KBP task. Existing relation validation methods followed voting (Sammons et al., 2014), multidimensional truth finding model (Yu et al., 2014), constraint optimization (Wang et al., 2013) and consensus maximization (Rodriguez, Goldberg, and Wang, 2015) techniques. Moreover, ranking methods (Jain and Pantel, 2010; Jean-Louis, Besançon, and Ferret, 2011) have been explored for relation validation task where object candidates of a relation are re-ranked by a graph based ranking algorithm.

Linguistic analysis is also important for validating a relation which is claimed in a sentence. Syntactic and semantic characteristics of relation expression can be inherited from linguistic analysis of relation extraction task. Yu et al. (2014) inspected linguistic evidence such as the existence of trigger words in the syntactic dependency path for validating relations.

However, few works have explored yet collective analysis for relation extraction task. The collective analysis could be effective for validating relations when the information in a corpus is represented in a graph structure. For example, documents, sentences and entity mentions in a corpus could be denoted as graph nodes and relationship among them could be indicated by edges. A graph structure facilitates to examine the association between related entities based on community analysis and centrality measurements. Moreover, node ranking and information theoretic measurements could be possible on a graph.

In order to validate a claimed relation between two entities, we come with the following propositions:

- We inherit different linguistic indications of expressing and characterizing relation at the sentence level from the literature of relation extraction task. We propose to generalize the dependency patterns of relation expression by computing edit distance. We also propose to employ word embeddings for detecting unknown trigger words of a relation by computing similarity between two word vectors.

- We adopt voting and candidate ranking techniques from the literature of relation validation.

- We compute several new features on the graph of entities based on community analysis, centrality measurement and information theoretic measurement.

We consider relation validation as a binary classification task to decide where the candidate object in a claimed relation is correct or wrong. We adopt two models: supervised and unsupervised regarding this task. As a supervised method, we train a

binary classifier with several features computed on the instances of a labeled dataset. In contrast, we employ PageRank algorithm on a graph for raking the candidate objects of a relation to choose the correct ones in an unsupervised fashion.

# Chapter 3

# Entity Graph and Measurements for Relation Validation

GRaph based models for knowledge representation has been popular for the last couple of years. A graph facilitates to represent various objects as nodes and links among the nodes describe how the different objects are related among them. Graph databases are used to store and represent the information about the objects by maintaining the graph structures. Graph based methods have been proven effective for solving a variety of problems such as text summarization (Erkan and Radev, 2004), community detection (Fortunato, 2010), social network analysis (Girvan and Newman, 2002), entity linking (Guo et al., 2011) etc.

The mentions of different entities and their co-occurrences in texts can be depicted in a graph. The information and resources which are common between two entities can be observed in such presentation. Thus it can help to analyze the relationship between a pair of entities. Therefore, we explore graph of entities to find out some clues of holding true relationship between a pair of entities by analyzing their communities. This chapter focuses on the theoretical background and construction of a graph. Furthermore, computation of several features based on graph analysis for validating a claimed relation hypothesis between two entities.

## 3.1 Graph Definition

A graph is a structure to represent objects and relationship among them. Formally a graph is defined by a set of vertices and edges. Objects are represented by the vertices (nodes) and edges (links) describe the relationship among the objects. A graph can be directed or undirected where the links of a directed graph indicate the directions of the relations between the pairs of nodes. In contrast, a link in an undirected graph just refers to the existence of a relationship between two nodes. It does not emphasize on the direction of the relation. A graph is called *connected* when a node is reachable from all other nodes.

An example of a connected undirected graph is shown in Fig. 3.1 which consists of 6 nodes $V = \{v_1, v_2, ..., v_6\}$ and 9 links $E = \{e_1, e_2, ..., e_9\}$. The link $e_1$ between nodes $v_1$ and $v_2$ indicates that these two nodes are in a relationship. This graph constraints that there can be at most one link between a pair of nodes. The properties of the nodes and relationships (e.g. node types and weights of the relations) can be described in the nodes and links accordingly. This graph can be represented as a two dimensional matrix as shown in Eq. 3.1. Here the rows and columns refer to the indices of the nodes and a cell of the index pair $(i, j)$ of $G$ indicates the relationship between the nodes $v_i$ and $v_j$. Moreover, an adjacency matrix $A$ (Eq. 3.2) of the graph

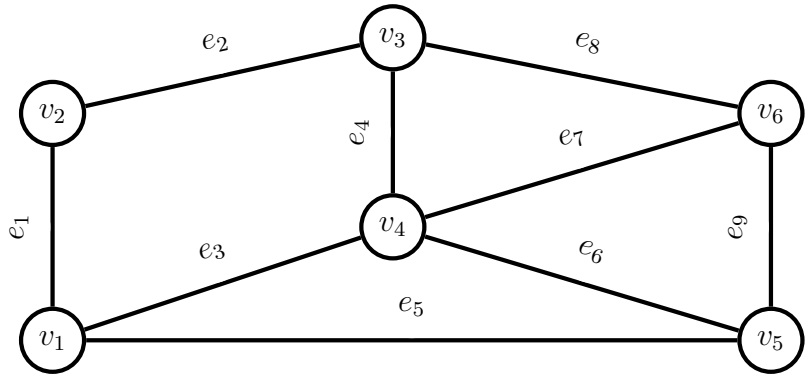


FIGURE 3.1: A graph with nodes and links

$G$ represents the connectivity information among the nodes where $A(i,j) = 1$ means that the nodes $v_i$ and $v_j$ are connected, and $A(i,j) = 0$ otherwise. Adjacency matrix can also be constructed by the number of links of each node, weights of the links and so on. Such representation facilitates performing mathematical operations on the graph.

$$G_{i,j} = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{pmatrix} - & e_1 & - & e_3 & e_5 & - \\ e_1 & - & e_2 & - & - & - \\ - & e_2 & - & e_4 & - & e_8 \\ e_3 & - & e_4 & - & e_6 & e_7 \\ e_5 & - & - & e_6 & - & e_9 \\ - & - & e_8 & e_7 & e_9 & - \end{pmatrix} \end{array} \tag{3.1}$$

$$A_{i,j} = \begin{array}{c} \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \end{array} \begin{array}{c} \begin{array}{cccccc} v_1 & v_2 & v_3 & v_4 & v_5 & v_6 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \end{array} \tag{3.2}$$

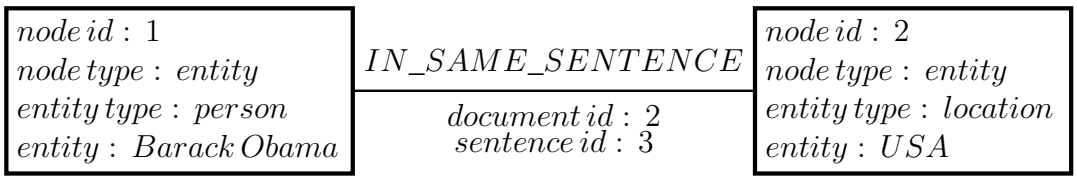


FIGURE 3.2: A graph with two entity nodes and their relationship

## 3.2 Entity Graph and Graph Database

We use the graph structure for representing different types of entities and relationships among them. The node of a graph describes an entity with its properties and a link between two entities refers to the relation and properties of the relationship between them.

Fig. 3.2 shows two entity nodes of types *person* and *location* and their relationship in term of co-occurrence in text. For example, the entities *Barack Obama* and *USA* are found in the same sentence (IN_SAME_SENTENCE) of a document where document id and sentence id are 2 and 3 accordingly. The information about the entities and their relationships extracted from a collection of texts are stored in a graph database that visualizes the information in a graph structure.

Neo4j[1] is a schemaless graph database which facilitates storing information in the form of a node, edge and attribute, visualizing in a graph structure and performing different queries and operations on the graph based on different criteria. Cypher Query Language is used to perform queries on this database. A sample Neo4j graph is shown in Fig. 3.3 which consists of three types of entity nodes (person, organization and location) and a document type node. It also contains two types of relational links IN_SAME_SENTENCE and FOUND_IN where IN_SAME_SENTENCE connects two entities that are found in the same sentence and FOUND_IN connects a document node to an entity node.

**Ex 3.1: Cypher syntax for querying on Neo4j**

```
MATCH (m:PERSON) MATCH (n:ENTITY) MATCH (m)-[r:IN_SAME_SENTENCE]-(n) WHERE m.name = "Barack Obama" RETURN r
```

A query (Ex. 3.1) performed on this graph asking the subgraph of the entities having IN_SAME_SENTENCE links to *Barack Obama* returns the subgraph shown in Fig. 3.4 which consists of seven entity nodes where five are *person*, one is *organization* and one is *location* typed. This subgraph can be considered as the community of *Barack Obama* at level 1 based on IN_SAME_SENTENCE relation. Thus it is

[1] https://neo4j.com/

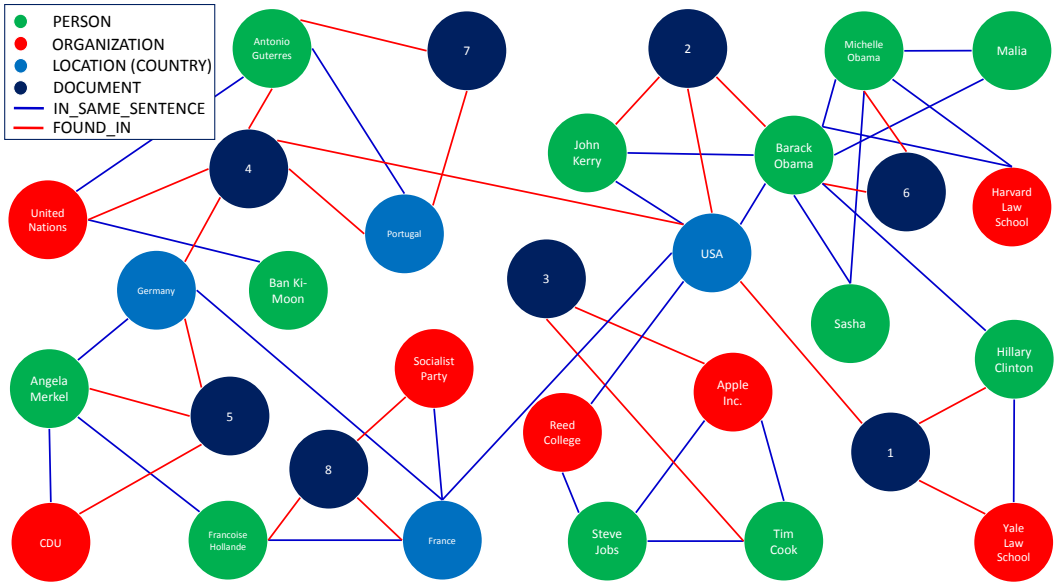


FIGURE 3.3: Neo4j graph with different types of nodes and links

possible to extract the community of an entity based on different relations among the community members.

## 3.3 Graph Construction

The graph of entities or community graph (as shown in Fig. 1.1 on Page 9) is constructed from a graph representing the association among documents, sentences and entity-mentions in a collection of texts (lower part of Fig. 3.5) called association graph. The association graph represents documents, sentences, mentions and entities as nodes and the edges between these nodes represent relationships between these elements. The association graph is generated by applying systems of named entity recognition and sentence splitting. Named entity recognition (NER) is done by *Luxid*[2], which is able to decompose the entities into components, such as *first name* and *title*, and by the Stanford CoreNLP (Manning et al., 2014b). When the two systems disagree, we choose the annotation produced by Luxid that gets better precision. This entity-graph basically contains three types of named entities which are *person*, *geopolitical location* and *organization*. Geopolitical locations are classified into three subtypes: country, region (state/province) and city. Dates are also included in the graph as nodes after normalizing the date string. For example, *7th March 2017* and *August 2018* are normalized as *2017-03-07* and *2018-08-XX* accordingly.

We assume that two entities mentioned in the same sentence share the common meaning since a sentence as being the basic unit of a language states a complete

[2]http://www.expertsystem.com/fr/

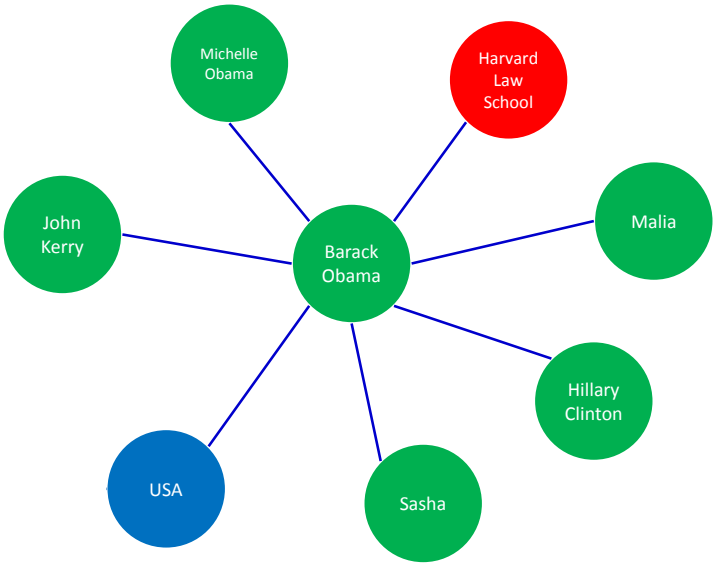


FIGURE 3.4: A subgraph extracted from the Neo4j graph (Fig. 3.3) by performing the query of Ex. 3.1

sense. We hypothesize that the co-occurrences of pairs of entity mentions could provide some evidence of justifying relationships. Therefore, we aim at connecting two entities which co-exist in the same sentence by IN_SAME_SENTENCE relational link.

Multiple mentions of the same entity found in the same document are connected to the same entity node in the association graph, based on the textual similarity of the mentions and their possible components, which corresponds to the first step of entity creation on local criteria. This operation is performed by Luxid. However, an entity can be mentioned in different documents, in the same or different forms (e.g. *Barack Obama*, *President Barack Obama*, *President Obama* etc.) which create redundant nodes in the association graph.

Then, the entities are clustered based on the similarities in their names. We assume that different mentions of an entity can be grouped together by matching their mentioning strings or components of the mentions. However, if two different entities have similar mention, they can be distinguished by their neighbor entities. The task of clustering *person* type entities is done in two sub-steps. Firstly, we inspect whether the mention of an entity includes three basic components such as *title*, *first name* and *last name*. Then, a pair of entities (i.e. $E_1$ and $E_2$) are compared in terms of their components. We define a set of constraints (Constraints 3.1) to decide if a pair of entities belongs to the same cluster. When two entities satisfy any one of the constraints (based on conditional OR operation) we keep them in the same cluster. We select a seed entity of a cluster which is mentioned with a comparatively higher

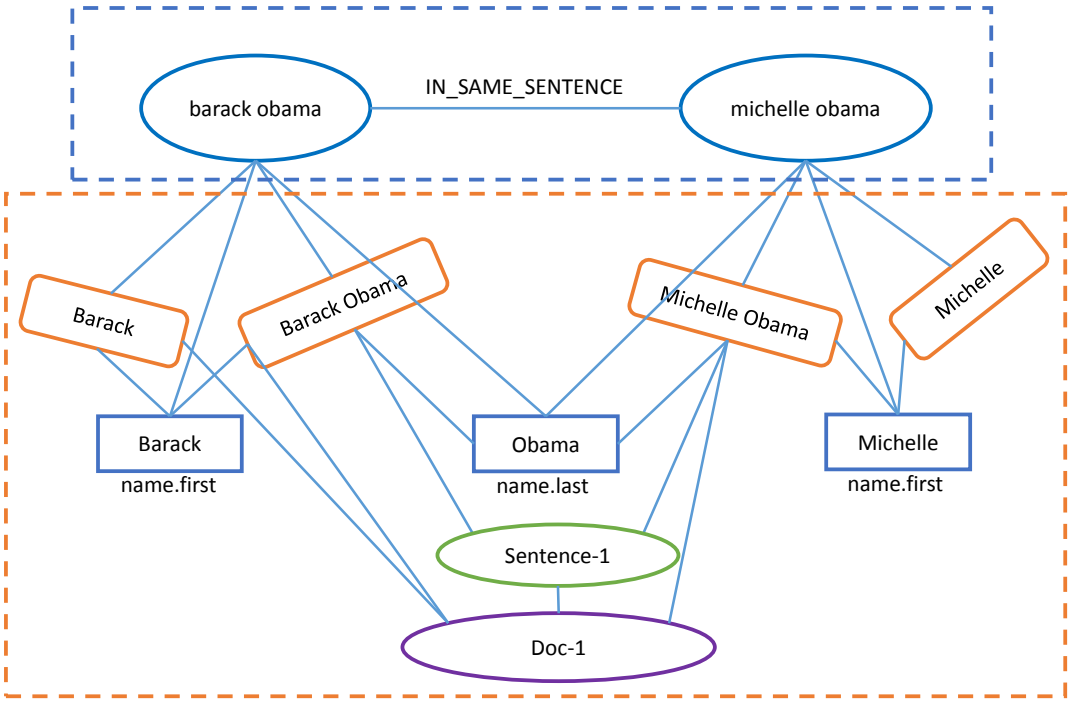


FIGURE 3.5: Association graph (in the bottom part) that enables to build the community graph (in the top part)

number of components.

**Constraints 3.1: Person Type Entity Clustering**

1. *Title*($E_1$) == *Title*($E_2$) AND *FirstName*($E_1$) == *FirstName*($E_2$) AND *LastName*($E_1$) == *LastName*($E_2$)
2. *Title*($E_1$) == *Title*($E_2$) AND NOT *HasFirstName*($E_2$) AND *LastName*($E_1$) == *LastName*($E_2$)
3. *Title*($E_1$) == *Title*($E_2$) AND *FirstName*($E_1$) == *FirstName*($E_2$) AND NOT *HasLastName*($E_2$)
4. NOT *HasTitle*($E_2$) AND *FirstName*($E_1$) == *FirstName*($E_2$) AND *LastName*($E_1$) == *LastName*($E_2$)
5. NOT *HasTitle*($E_2$) AND *FirstName*($E_1$) == *FirstName*($E_2$) AND NOT *HasLastName*($E_2$)
6. NOT *HasTitle*($E_2$) AND NOT *HasFirstName*($E_2$) AND *LastName*($E_1$) == *LastName*($E_2$)

However, an organization type entity cannot be splitted into the components like a person type entity. Therefore, we compute cosine similarity between the character tri-grams of a pair of entities for clustering organization type entities. Two entities are considered to belong to the same cluster if the similarity score satisfies a threshold of $0.5$. Unlike person and organization type entities, a location type entity maintains

almost the same form in its different mentions. Therefore, we do not perform any clustering of the location type entities.

In order to decide if two entities with a similar name in the same cluster truly refer to the same person or organization, we compute the similarity of their neighboring entities by Eq. 3.10 on Page 45. We empirically define two different thresholds for the person and organization type entities which are $0.1$ and $0.5$ accordingly. If the similarity score satisfies the predefined threshold we merge two entity nodes into a single node.

However, an entity may belong to two or more clusters according to our clustering method. In such case, we select the cluster of that particular entity based on the highest similarity score between the communities of that entity and other entities in the different clusters. When several entities in a cluster refer to the identical person or organization, we merge the corresponding nodes into a single node and choose the name of seed entity for the merged node in the community graph as shown in the upper part of the association graph illustrated in Fig. 3.5. In the association graph, the links between entities and documents are always maintained via their mention nodes. It is thus possible to know the number of mentions of each entity and the number of coexistences of a pair of entities.

We do not either use any existing entity graph such as DBpedia or construct the graph from Wikipedia articles because this database or articles mostly contain information about popular entities. The entities and relations we study in this thesis may not exist there. Therefore, the association graph is built from a given corpus where entities and relations have to be extracted. We store the graph in a Neo4j database which makes it possible to extract the subgraphs linked to an entity by queries as illustrated in Section 3.2.

## 3.4 Measurements on Graph

Several measurements on a graph have been explored for solving different problems like entity linking by measuring the degree of entity nodes (Guo et al., 2011), knowledge discovery by measuring entropy in a publication network (Holzinger et al., 2013) etc. Centralities of the graph nodes have been computed for finding the most important and influential node and clustering the nodes into several groups. In order to quantify the degree of connectivity among the nodes of a cluster, density is measured. Moreover, the similarity between two clusters can be calculated based on the common members.

### 3.4.1 Node Centrality

Centrality measurement in a graph or network characterizes the nodes of that particular network. The characterization of the nodes differs based on different measurements of centrality. Here we focus on three centrality measurements related to our task.

#### 3.4.1.1 Degree Centrality

Degree centrality (Freeman, 1977) is used to find the highly connected nodes in a network by counting the number of direct neighbors of each node. The popular nodes in a network are highly connected and more information flow through these nodes. Thus degree centrality facilitates to detect important nodes. The degree centrality $C_D(v)$ of a node in a network *G:=(V, E)* with a set of nodes $V = \{v_1, v_1, ..., v_n\}$ and a set of links $E = \{e_1, e_1, ..., e_m\}$ is measured by Eq. 3.3.

$$C_D(v) = \sum_{v':v' \neq v} e(v, v') = degree\, of\, node\, v \tag{3.3}$$

*where,* $e(v, v')$ *refers to a direct link between* $v$ *and a neighbor node* $v'$*.*

#### 3.4.1.2 Betweenness Centrality

Betweenness centrality (Freeman, 1977) is measured based on the shortest paths between each pair of nodes. In a connected graph, every pair of nodes has at least one shortest path. Betweenness centrality of a node counts the number of shortest paths passing through that particular node. It characterizes a node to become a bridge between two other nodes. Higher betweenness centrality of a node refers to be the bridge among a large number of nodes. The betweenness centrality $C_B(v)$ of a node $v$ in a network is computed by the Eq. 3.4.

$$C_B(v) = \sum_{s \neq v \neq t} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{3.4}$$

*where,* $\sigma_{st}$ *refers to the total number of shortest path between nodes* $s$ *and* $t$*, while* $\sigma_{st}(v)$ *is the number of those shortest path passing through node* $v$*.*
Edge betweenness (Girvan and Newman, 2002) has been studied for community detection in a network which implies the similar concept of betweenness centrality. When a network contains some interconnected communities or groups and these are loosely connected by some edges, the shortest paths among the communities pass through at least one of the loosely connected edges. Thus the edges connecting

different communities get high edge betweenness score and the communities can be separated by removing these edges.

#### 3.4.1.3 Eigenvector Centrality

Eigenvector centrality (Bonacich and Lloyd, 2001) measures the influence of a node in a graph. A node will be even more influential if it is connected to other influential nodes. Unlike degree centrality, a node having high eigenvector centrality does not mean the node is highly connected. Moreover, two nodes having the same degree centrality may have different eigenvector centrality. Degree centrality does not account the neighbor nodes at the further levels apart from the first level to compute the importance of a node. However, eigenvector centrality accounts the contribution of the neighbor nodes recursively to compute the centrality of a node that facilitates to find out the nodes having the similar influence in a network.

$$C_E(v) = \frac{1}{\lambda} \sum_{t \epsilon G} a_{v,t} C_E(t) \tag{3.5}$$

*where,* $\lambda \neq 0$ *is a constant and the equation can be expressed in the matrix form:* $\lambda C_E = AC_E$.
Suppose, a network *G:=(V,E)* and $A = (a_{v,t})$ is the adjacency matrix of the network $G$ where $a_{v,t} = 1$ if node $v$ is linked to node $t$, and $a_{v,t} = 0$ otherwise. The eigenvector centrality $C_E(v)$ score of a node $v$ can be computed by Eq. 3.5.

Eigenvector centrality can be compared to PageRank (Brin and Page, 1998) centrality. PageRank vector is also an eigenvector but a damping factor is added in the measurement of PageRank centrality to restrict extending community to a certain level by continuously reducing the weight at each level.

### 3.4.2 Mutual Information

Information flows in a network node to node through the links. The flow of information differs among different nodes in a network based on their connectivity. Shannon entropy (Shannon, 1948) measures the rate of information flow of a random variable in a transmission medium. It basically counts the number of bits to encode a message or information. Thus the flow of information in a network can be quantified by measuring the entropy of the network based on the degree of connectivity of each node. Suppose, $G := (V, E)$ is a network with vertices $V = \{v_1, v_2, ..., v_n\}$ then the

entropy of the network $G$ can be measured by Eq. 3.6.

$$H(G) = -\sum_{i=1}^{n} p(v_i)\ log_2(p(v_i)) \tag{3.6}$$
$$\textit{where, } p(v) = \frac{degree\ of\ node\ v}{total\ number\ of\ links\ (|E|)\ in\ G}$$

Mutual information quantifies the amount of information gained by a random variable compared to another one. The mutual information between two random variables $X$ and $Y$ can be measured by Eq. 3.7.

$$MI(X,Y) = H(X) + H(Y) - H(X,Y) \tag{3.7}$$

If a network consists of two communities, it is possible to quantify the mutual information between them based on the degree of connectivity of the nodes by using Eq. 3.6 and by modifying Eq. 3.7 a little. Suppose, a network $G$ is composed of two community networks $G_x$ and $G_y$. We compute the mutual information between $G_x$ and $G_y$ by using Eq. 3.8.

$$MI(G_x, G_y) = H(G_x) + H(G_y) - H(G_x, G_y) \tag{3.8}$$
$$\text{where, } H(G_x, G_y) = H(G)$$

### 3.4.3 Network Density

Network density measures the degree of connectivity in a network. Mathematically it calculates the ratio of the number of existing links to the number of potential links in a network. The density of a network of $n$ nodes and $|E|$ existing links can be calculated by Eq. 3.9 where $L'$ refers to the highest number of possible links.

$$\rho = \frac{total\ number\ of\ links\ (|E|)}{number\ of\ potential\ links\ (L')} \tag{3.9}$$
$$\text{where, } L' = \frac{n * (n-1)}{2}$$

According to this definition the density of the network shown in Fig. 3.1 is $0.60$ where the numbers of nodes, existing links and potential links are $6$, $9$ and $15$ accordingly. However, the density of the network presented in Fig. 3.4 is $0.25$. These two scores of network density help to realize that the network of Fig. 3.1 on Page 36 is highly interconnected compared to the network of Fig. 3.4 on Page 39. If some nodes in a network are highly interconnected among them these particular nodes can be considered as the members of the same community.

### 3.4.4 Network Similarity

Network similarity measures how much similar two networks are. Typically it is calculated based on the cosine similarity measurement. The similarity between two networks can be easily measured on the basis of the common nodes shared by both networks. Suppose, $V_x = \{v_{x1}, v_{x1}, ..., v_{xm}\}$ and $V_y = \{v_{y1}, v_{y1}, ..., v_{yn}\}$ are the node sets of two different networks accordingly. The similarity between these networks can be measured by Eq. 3.10.

$$network\ similarity = \frac{|V_x \cap V_y|}{\sqrt{|V_x||V_y|}} \tag{3.10}$$

Network similarity gives a realization similar to the network density. If two networks share a large number of members, the similarity score would be very high. Thus two communities in a network having high similarity score to each other can be considered as a single community by merging them.

## 3.5 Relation validation by Graph Analysis

A relation can be defined between a pair of entities by a tuple as shown in Def. 3.1 where $R$, $E_q$ and $E_c$ refer to the relation name, subject entity and object entity accordingly. For the task of relation validation $E_q$ and $E_c$ become a subject and a candidate object accordingly for the relation $(R, E_q, ?)$ and our objective is to predict whether the candidate object $(E_c)$ is correct or not for the claimed relation.

**Def 3.1: A relation between two entities**

$$(R, E_q, E_c)$$

*(spouse, Barack Obama, Michelle Obama)*

We propose to incorporate graph based analysis (as presented in Section 3.4) for validating a claimed relation between two entities in addition to the linguistic analysis. Such studies on graph analysis were not done before for relation validation task. Basically, we analyze the communities of the subject and object entities of a relation hypothesis to get some clues for validating it. Here, we define a community of an entity by the neighbor entities which are mentioned in the same sentences where that particular entity is mentioned.

A network as shown in Fig. 3.6 constructed based on the co-occurrences of the entities in texts does not hold any semantic evidence of holding a specific type of relationship between two entities but gives some realizations how they are associated to each other and with other entities which are common between the neighbors of this

two particular entities. We compute several features on the networks of two entities in a relationship to realize their relatedness.

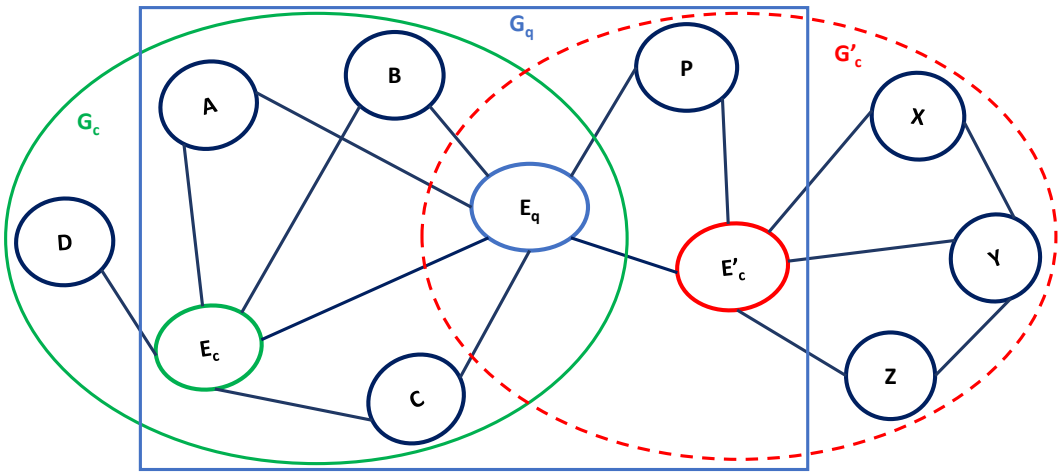


FIGURE 3.6: Community graph for realizing different measurements

Suppose, the communities of a subject entity ($E_q$) and a candidate object ($E_c$) are extracted from the constructed network which are $G_q$ and $G_c$ accordingly. A community can be expanded up to several levels from the entity node under observation. These two communities are analyzed to predict whether $R(E_q, E_c)$ is true.

**Network similarity:** we compute the similarity score (Eq. 3.10) between $G_q$ and $G_c$ and expect higher similarity score if $G_c$ is the community of the correct candidate than the score between $G_q$ and the community of an incorrect candidate, $G'_c$. In Fig. 3.6, the similarity score between $G_q$ and $G_c$ is $0.46$ while the score between $G_q$ and $G'_c$ is $0.15$.

**Network density:** we merge the community of $G_q$ to $G_c$ and calculate the density (Eq 3.9) of the merged network. We hypothesize that the density would be higher for a correct candidate than the density of the merged network of $G_q$ and $G'_c$. The density scores of the merged networks $G_q + G_c$ and $G_q + G'_c$ are $0.20$ and $0.17$ accordingly.

**Eigenvector centrality:** we also compute eigenvector centrality for each node after merging the communities of $G_q$ and $G_c$. We expect similar centrality score for the nodes of $E_q$ and $E_c$ if $E_c$ is the correct object because $E_q$ and $E_c$ should be highly influential to each other. Therefore, we calculate the absolute difference of the eigenvector centrality scores of $E_q$ and $E_c$. We hypothesize that $|C_E(E_q) - C_E(E_c)| < |C_E(E_q) - C_E(E'_c)|$ where $E'_c$ refers to an incorrect candidate. In Fig. 3.6, eigenvector centrality is measured for each node of the full network. The normalized scores of nodes $E_q$, $E_c$ and $E'_c$ are $1.0$, $0.78$ and $0.74$ accordingly where the scores of $|C_E(E_q) - C_E(E_c)|$ and $|C_E(E_q) - C_E(E'_c)|$ are $0.22$ and $0.26$ respectively.

**Mutual information:** furthermore, we compute mutual information between $G_q$ and $G_c$ and expect higher mutual information if $G_c$ is the correct candidate than the mutual information measured between $G_q$ and $G'_c$.

The community of an entity is constructed by the neighbors of level one for calculating the network similarity and network density scores. However, the eigenvector centrality and mutual information are computed by expanding the community up to third level to include more information about the entities.

As early mentioned, we consider relation validation as a binary classification task. A feature vector is generated including the four graph based features: network similarity, network density, mutual information and eigenvector centrality.

## 3.6 Conclusion

We expect that the features computed on the network as we discussed so far are able to characterize the existence of a strong relationship between two entities. However they are not able to characterize the semantic type of a relation. Characterization of the type of a semantic relation mostly depends on the linguistic information. Therefore, we incorporate linguistic features to the graph based features which are discussed in this chapter in order to fully represent a relation. In the next chapter, we will focus on the linguistic analysis of expressing relations. The contribution of the proposed features in addition to the linguistic features of validating relations will be discussed in Chapter 6.

# Chapter 4

# Linguistic Characteristics of Expressing and Validating Relations

The expression of a relationship between two entities in text follows some linguistic properties. Studies of relation extraction from texts basically focus on syntactic and lexical semantic analysis (Kambhatla, 2004; GuoDong et al., 2005; Mintz et al., 2009; Surdeanu et al., 2012). Syntactic analysis inspects the grammatical structure of a relation when it is expressed between a pair of entities at the sentence level. However, Semantic type of a relation is characterized by the words between and/or around the entity mentions.

Our objective is to validate already extracted relations. We assume that the tasks of relation extraction and relation validation are the opposite side of the same coin. Similar to the relation extraction, validating a relation requires natural language understanding of pieces of text. Therefore, linguistic analysis comes into play to justify whether a claimed relation between a pair of entities in text is true or false.

In this chapter, we focus on dependency patterns of relations based on syntactic analysis, identifying semantic type of a relation based on trigger words and word-embeddings, and finally make a fusion to take advantages of both syntactic and semantic analysis.

## 4.1 Linguistically Motivated Classification of Relation

Relations are expressed between different types of entities such as *person*, *organization*, *location* etc. Semantic type of a relation depends on the types of the pair or entities and words between and/or around their mentions. For example, a *spouse* relationship always occurs between two persons while a *residence* relationship is mentioned between a person and a location. However, the types of the entities do not hold sufficient evidence to explicitly characterize the relations. Because different pairs of entities with similar types may express different relations. For example, the types of the entity pairs (*person-person*) in both spouse and children relationships (in Ex. 4.1) are the same. Therefore, words around the mentions are analyzed to collect more evidence for identifying the specific type of a relation. The words *married* and *daughter* characterize two different relations *spouse* and *children* accordingly.

**Ex 4.1: Trigger-dependent and trigger-independent relation**

*spouse (John, Julie):* John **married** Julie in December.

*children (Julie, Jesi):* Julie's **daughter** Jesi likes to play piano.

*residence (John, France):* John was in France for three years.

*residence (John, France):* John is currently **living** in France.

A content word which is explicitly able to characterize the semantics of a relation between two entities is called a trigger word. For example, *married, husband, wife* are the trigger words for *spouse* relation. Semantic relations can be classified into two types based on triggers words: *trigger-dependent* and *trigger-independent* (Yu and Ji, 2016). A trigger-dependent relation requires at least one trigger word to express the relation between a pair of entities. *Spouse*, *children*, *bornIn* etc. are examples of trigger-dependent relations. In contrast, a trigger-independent relation can be expressed with or without a trigger word. For example, *residence, subsidiaries* etc are trigger-independent relations. The first sentence in Ex. 4.1 states *spouse* relationship between John and Julie where the word *married* explicitly indicates the semantic type of the relationship. The third one expresses *residence* relationship between John and France where no word explicitly holds the semantic of *residence* relation. However, the *residence* relationship in the third sentence is understandable by humans based on the grammatical orientation of the words. In contrast, the word *living* in the last sentence explicitly refers to the *residence* relation. Thus validating a relation either trigger-dependent or trigger-independent requires semantic and syntactic analysis. Sometimes validating a trigger-independent relation depends only on the syntactic analysis when it is expressed in a sentence having no trigger word.

We studied annotated sentences of different relations to classify the relations into trigger-dependent and trigger-independent. We collected the words between the subject and object pairs of positive examples of each kind of relation and ranked them by counting their frequencies. We observed that for some relations (i.e. *spouse*, *parents*, etc) at least one of the top five words is discriminating (i.e. *married*) for characterizing the semantics of the relation. We call these relations trigger-dependent relations. In contrast, we notice that for other relations (i.e. *city_of_residence*, *subsidiary* etc) top five words are either prepositions or other words that are not able to distinguish any semantic relation. Such relations are called trigger-independent relations. A trigger-independent relation can also be expressed by using a trigger word but it is not mandatory. Table 4.1 shows some trigger-dependent and trigger-independent

relation names which have been grouped based on the trigger word analysis. Section 4.3 describes in detail how we collect the trigger words and identify unknown triggers of a relation.

| **Trigger-dependent** | **Trigger-independent** |
|---|---|
| per:spouse | per:city_of_residence |
| per:parents | per:country_of_residence |
| per:children | per:employee_or_member_of |
| per:city_of_birth | per:age |
| per:country_of_birth | org:parents |
| per:city_of_death | org:shareholders |
| per:country_of_death | org:subsidiaries |
| per:date_of_birth | org:number_of_employee |
| per:date_of_death | |
| org:top_members_employess | |
| org:founded_by | |
| org:member_of | |
| org:city_of_headquarter | |
| org:country_of_headquarter | |

TABLE 4.1: Trigger-dependent and trigger-independent relations

In order to validate a relation, we inspect the existence of any trigger word of the claimed relation and analyze syntactic dependencies among different words in a relation justifying sentence.

## 4.2 Syntactic Modeling

Syntactic analysis facilitates to understand the meaning of a sentence by dividing the words in a sentence into different parts and by characterizing their grammatical relations. The syntactic analysis covers a set of grammatical annotations at the sentence level such as *parts-of-speech tagging* (POS-tagging), *segmentation*, *dependency parsing* etc (Jurafsky and Martin, 2014).

POS tagging is a kind of annotation of the words in a sentence based on the *parts of speech* such as *noun, verb, adjective* etc. Such annotation indicates the role of a word and formal relationship between the adjacent or related words in a sentence. For example, *John* and *likes* in the sentence of Ex. 4.2 refer to a *proper noun* and a *verb* accordingly in terms of parts-of-speech.

**Ex 4.2: POS tagging and segmentation of words in a sentence**

| *John* | *likes* | *the* | ***coffee*** | ***shop*** | *where* | *he* | *first* | *met* | *Julie.* |
|---|---|---|---|---|---|---|---|---|---|
| *NNP$* | *VBZ* | *DT* | *NN* | *NN* | *WRB* | *PRP* | *RB* | *VBD* | *NNP* |

The boundaries of a compound word (two or more consecutive words together expresses a single meaning) and a clause (consists of a subject and a predicate) in a sentence are identified by segmentation. For instance, *coffee shop* in Ex. 4.2 refers to a place of selling coffee. Moreover, the sentence contains two clauses (purple colored) separated by *where* which individually expresses a complete sense.

In information extraction and relation extraction related tasks, POS tagging and segmentation play important roles to realize the inter-word relationships in a sentence (Fader, Soderland, and Etzioni, 2011; Li et al., 2011). *Syntactic dependency parsing* gives a better explanation of the relationships among words based on the directed grammatical relations as found in some relation extraction tasks (Mintz et al., 2009).

### 4.2.1 Syntactic Dependency Analysis

Syntactic dependency refers to the directed grammatical relation between a pair of words where the relation is defined by two arguments *head* and *dependent* (Jurafsky and Martin, 2014). A relation between head and dependent indicates a grammatical function such as subject, object etc.

**Ex 4.3: Syntactic dependency**

| *John* | *likes* | *the* | *coffee* | *shop* | *where* | *he* | *first* | *met* | *Julie.* |
|---|---|---|---|---|---|---|---|---|---|
| *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *10* |

***function(head, dependent)***
*nsubj(likes-2, John-1)*
*root(ROOT-0, likes-2)*
*det(shop-5, the-3)*
*compound(shop-5, coffee-4)*
*dobj(likes-2, shop-5)*
*advmod(met-9, where-6)*
*nsubj(met-9, he-7)*
*advmod(met-9, first-8)*
*acl:relcl(shop-5, met-9)*
*dobj(met-9, Julie-10)*

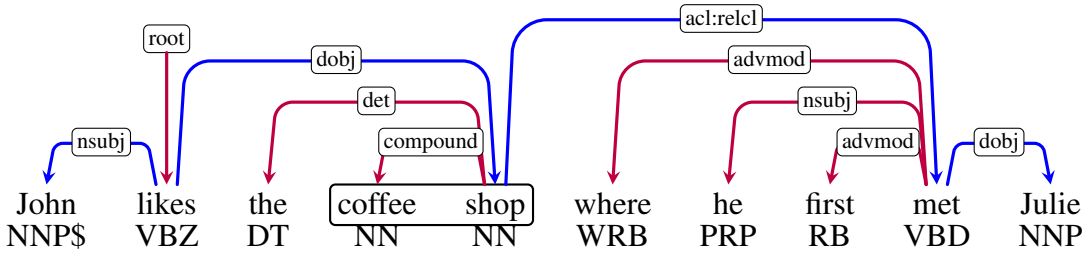


FIGURE 4.1: Syntactic dependency graph

The dependency relations between the pairs of words are shown in Ex. 4.3. Here *nsubj(likes-2, John-1)* indicates the grammatical relationship between the words *John* and *likes* where *nsubj* refers *John* to be a nominal subject. In addition, *dobj(likes-2, shop-5)* refers *shop* to be a direct object. Moreover, *compound(shop-5, coffee-4)* means *coffee* and *shop* together to be a compound word, *coffee shop*. On the basis of these grammatical relations, it makes possible to detect the clause with the words *John likes the coffee shop*. Thus syntactic dependency analysis facilitates to identify the related words and the syntactic relationships among them in a sentence.

A graph representation of the syntactic dependencies as shown in Fig. 4.1 incorporates to find the dependency path between two words. For example, the blue colored connections indicate the path between *John* and *Julie*. Such path provides some information such as dependency labels (function) and path length to justify if any potential relationship exists between two words.

### 4.2.2 Dependency Patterns and Edit Distance

Binary relations (i.e. *spouse*) follow some structures (i.e. *X married Y*) by placing the words between two arguments (X and Y). Such structures are repeated to mention relationships between different pairs of arguments which are called *patterns* of relations. Pattern based relation extraction was first studied by Hearst, 1992. Pattern based methods usually use POS-tags (Fader, Soderland, and Etzioni, 2011) and lexico-syntactic information ( Alfonseca et al., 2012; Pershina et al., 2014). Since lexical information is important to characterize the semantic type of a relation, lexical information is added to the POS-tag based patterns to map the open information to semantic relations. However, lexico-syntactic patterns include both lexical and structural information for semantic relation extraction where dependency labels are used for capturing the grammatical structure.

A relation between two arguments in a short or long dependency path can be captured by using syntactic dependency tree. The dependency labels between a pair of words in a relationship make a pattern of that particular relation. Such pattern can be repeated between the same or different pair of words in another sentence.

For example, a spouse relationship is mentioned between *John* and *Julie* in Fig. 4.2a where the dependency labels (blue colored solid lines) make a pattern *(nsub, dobj)*. Therefore, we have been motivated to define patterns of relations by only syntactic information i.e. dependency labels between a pair of arguments.

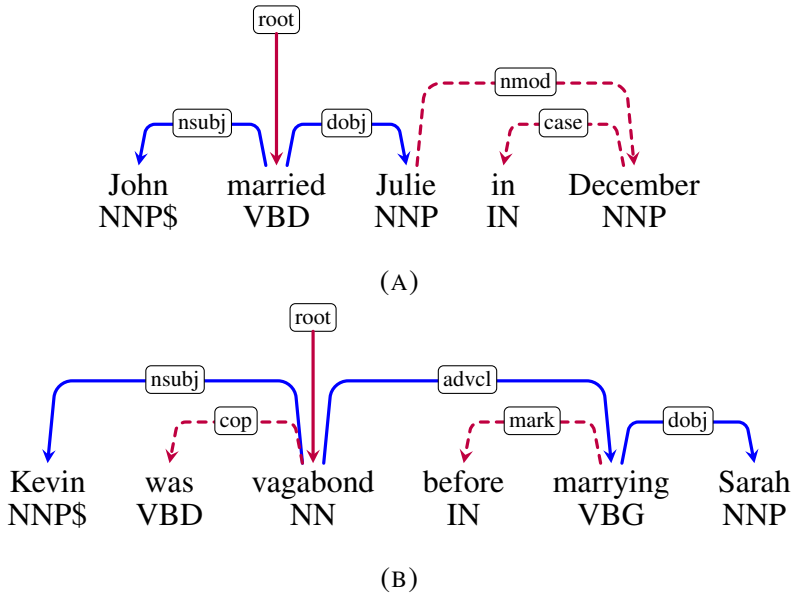


FIGURE 4.2: Syntactic dependency graph of two sentences mentioning spouse relationship between two pairs of persons

We extract a list of dependency patterns for each relation by using a set of positive examples. For example, in the sentence *Paola, Queen of the Belgians is the wife of King Albert of Belgium.* the dependency pattern between *Paola* and *King Albert* is *[nn, nsubj, prep_of]*. We simplify the pattern *[nn, nsubj, prep_of]* to *[nsubj, prep_of]* by removing leading and following *nn*. We notice that sometimes the dependency patterns contain consecutive labels like *[nsubj, dobj, prep_of, prep_of, poss]*. In such cases, we simplify the pattern by substituting the consecutive labels with a single label that means *[nsubj, dobj, prep_of, prep_of, poss]* is simplified as *[nsubj, dobj, prep_of, poss]*. This simplification generalizes the dependency patterns.

In natural language, a relation can be expressed in several ways and it may not be possible to capture all the patterns of that relation due to lack of positive examples. For example, we obtain 21 different patterns of spouse relation from 37 annotated sentences. The statistics on the collected patterns of some semantic relations is given in Table 4.2. Additionally, some relation expressing snippets with trigger words are shown in Table 4.3. We propose to match the pattern of an unlabeled sentence that claims to hold a specific relation to the existing patterns of that relation by an approximation.

Levenshtein distance or edit distance (Levenshtein, 1966) measures the dissimilarity between two strings by counting the minimum number of operations required to

| Relation Name | # Patterns | # Triggers | # Sentences |
|---|---|---|---|
| per:spouse | 21 | 3 | 37 |
| per:parents | 25 | 8 | 47 |
| per:children | 16 | 8 | 19 |
| per:country_of_death | 8 | 6 | 9 |
| per:country_of_birth | 4 | 4 | 6 |
| per:city_of_death | 14 | 6 | 15 |
| per:city_of_birth | 9 | 4 | 12 |
| per:employee_or_member_of | 78 | 40 | 166 |
| org:top_members_employees | 49 | 20 | 155 |
| org:member_of | 7 | 5 | 7 |
| org:country_of_headquarters | 21 | 6 | 46 |
| org:city_of_headquarters | 26 | 6 | 42 |

TABLE 4.2: Statistics of the collected dependency patterns and trigger words

transform one string to another. An example of edit distance calculation is shown in Ex. 4.4 where there are two strings *PHOTOGRAPH* and *AUTOGRAPH* with lengths 10 and 9 accordingly. The string *AUTOGRAPH* can be transformed into *PHOTOGRAPH* by minimum three modifications which are (i) inserting *P* at index 1 (ii) replacing *A* by *H* at index 2 (iii) replacing *U* by *O* at index 3. Edit distance calculation is widely used for different tasks like automatic spelling correction (Brill and Moore, 2000), comparing genetic sequences (Kim et al., 2013) etc. Since a relation is expressed in different sequences of dependency relations, we can compute the edit distance between a pair of dependency patterns to capture the deviation among different patterns.

**Ex 4.4: Realization of edit distance**

| *1* | *2* | *3* | *4* | *5* | *6* | *7* | *8* | *9* | *10* |
|---|---|---|---|---|---|---|---|---|---|
| *P* | *H* | *O* | *T* | *O* | *G* | *R* | *A* | *P* | *H* |
| | *A* | *U* | *T* | *O* | *G* | *R* | *A* | *P* | *H* |
| *P* | *H* | *O* | *T* | *O* | *G* | *R* | *A* | *P* | *H* |

Suppose, a list of existing dependency patterns are *(a,b,c), (a,c,d), (b,c,d)* for a relation $R$ and a dependency pattern *(a,c,b)* between two words or entity mentions is captured in a sentence that claims $R$ relationship between the words. We calculate

| Relation Name | Relation Expressing Snippet |
|---|---|
| per:spouse | Alan Gross' **wife**, Judy, attended the event |
| per:parents | Amanda Knox's **father**, Curt Knox, said he hopes |
| per:children | Assaf Ramon, the **son** of Israel's first astronaut, Col. Ilan Ramon |
| per:country_of_birth | Raul Castro was **born** on June 3, 1931 in the town of Biran in northern Cuba |
| per:country_of_death | Vladimir Ladyzhenskiy **died** late Saturday in southern Finland |
| per:employee_or_member_of | Alan Gross was **working** as a contractor for the U.S. Agency |
| org:top_members_employees | Tim Carpenter, national **director** of the Progressive Democrats of America |
| org:member_of | Taiwan is a senior **member** of the Pacific Asia Travel Association |
| org:city_of_headquarters | Qatalys, Inc. founded in 1995 and **headquartered** in Dallas |

TABLE 4.3: Relation expressing snippets with trigger words (words in bold font indicate trigger words)

**Algorithm 1** Calculate dependency pattern edit distance

1: $P$ : *a dependency pattern claiming to hold relation R*
2: $Q$ : *a list of pre-annotated patterns of R*
3: **procedure** GETDEPENDENCYPATTERNEDITDISTANCE($P, Q$)
4: $Score \leftarrow 9999$ ▷ initializes edit distance score with a large number
5: **for** *each pattern* $q$ *in* $Q$ **do**
6: $d \leftarrow$ *CalculateEditDistance(q, P)* ▷ calculates Levenshtein distance
7: **if** $d < Score$ **then**
8: $Score \leftarrow d$
9: **return** $Score$

the edit distance between each pair of *[(a,c,b), (a,b,c)], [(a,c,b), (a,c,d)],[(a,c,b), (b,c,d)]* and take the minimum edit distance as a feature for relation validation. The algorithmic pseudo code of dependency pattern edit distance shown in Alg. 1. We expect lower edit distance score of a pattern $P$ with reference to the existing patterns $P'$ of a relation $R$ if $P$ supports $R$. For example, in Fig. 4.2a, *(nsub, dobj)* is a known pattern of expressing a spouse relation. Now, another sentence in Fig. 4.2b mentions spouse relation between a different pair of words which gives a different pattern *(nsub, advcl, dobj)*. We compute the edit distance between *(nsub, dobj)* and *(nsub, advcl, dobj)* when the second sentence claims to hold spouse relationship between *Kevin* and *Sarah*. The resulted edit distance score between these pair of patterns is 1. We use this score as a feature of binary classification task and a classifier automatically learns whether an unlabeled pattern belongs to the claimed relation. Thus edit distance calculation facilitates to generalize diverse patterns of the same relation. we use this edit distance feature for validating both trigger-dependent and trigger-independent relations.

## 4.3 Lexical Analysis

Lexis refers to the complete list of words used in a language and word is the basic unit of a sentence. According to the linguistic point of view, semantic analysis refers to understanding the meaning of a word, phrase, sentence etc (Jurafsky and Martin, 2014).

Every word has its own meaning and it represents an emotion, concept, object etc. A word may express different meanings based on the context of the text where it takes place. For example, *mouse* refers to an animal in terms of zoology while it indicates an electric device in terms of computer accessories. In contrast, different words may express the same meaning and they can be used alternatively in texts to express a concept such as *dinner* and *supper* both mean the evening meal. Moreover, several words collectively may form a single meaning such as *coffee shop* refers to a place of serving and drinking coffee. Thus a variety of words are used for expressing different things in natural language.

**Ex 4.5: Sentence categorization based on lexical semantics**

*Julie got **married** to John and two years later she became a **mother**.*

*John started **working** in a bank as a junior **officer**.*

Different words that are related to the same concept can be grouped together. For example, *father*, *mother*, *husband*, *wife*, *children* etc. denote the concept of family. Similarly, a sentence or paragraph can be categorized into a semantic type based on the words in it. For instance, the first sentence in Ex. 4.5 can be cast as the type of *family* relationship since it contains the words *married* and *mother*. However, the words *working* and *officer* refer the second to the semantic of *employment* or *profession*.

A relation between a pair of entities can be mentioned in a sentence by several words that are specific to characterize the relation. Such words are called trigger words. It requires interpreting the meaning of these words for understanding the mentioned relation and for justifying it linguistically. Thus the semantic type of a relation between two entity mentions in a sentence depends on the words between and/or around them. For example, the first sentence in Ex. 4.5 denotes *spouse* relationship between *Julie* and *John* where the word *married* and its position characterize the semantic type of the relationship.

### 4.3.1 Trigger Word Collection

A trigger word holds the semantics of a relation type as defined in section 4.1. Collecting the trigger words of different relations is required for validating a claimed relation hypothesis.

A set of very common trigger words (i.e. husband, wife etc) of a relation (i.e. spouse) can be easily collected as shown in section 4.1. In order to improve the coverage, synonyms and similar words have to be included in the list of trigger words. These words can be found in lexical resources, as WordNet[1] for instance. However, for some relations (i.e. shareholders, subsidiaries etc.) it may not be possible to collect all the trigger words, because, these relations can be expressed by using many different words. Highly frequent words (except stop words) between the entity mentions in the annotated sentences of a relation type can be considered as the trigger words of that relation. Therefore, for each type of relation, we extract the words between the pairs of entities in relation. We manually examine which of the frequent words are semantically able to characterize the specific relation and enlist them in the set of trigger words. Table 4.2 represents the statistics on the collected trigger words of some semantic relations.

However, still it is not enough to identify and collect all the trigger words of a relation due to lack of annotated sentences. We need to identify similar words in new sentences. Word embeddings are known for finding semantically similar words. Therefore, we employ word embeddings to recognize the unknown triggers.

### 4.3.2 Word Embeddings

Word embeddings refers to the vector representation of the words of a corpus where each word is represented by a vector of real numbers (Bengio et al., 2003; Schwenk, Dchelotte, and Gauvain, 2006; Mikolov et al., 2013). Such representation of words basically is done by vector space model where semantically similar words are mapped to nearby points. Vector representation of the words is done based on the co-occurrences of the words in a corpus but the measurement of co-occurrences differs in different embedding models. There are basically two models of learning the vector encoding of the words: *count-based* and *predictive*.

In *count-based* models, a large matrix is constructed where row and column refer to the words and contexts accordingly or vice versa. Co-occurrences of the words are counted according to different contexts. Finally, the high-dimensional vectors of the words are reduced to low-dimensional vectors. An example of count-based vector representation of the words is shown in Ex. 4.6. GloVe (Pennington, Socher,

[1] https://wordnet.princeton.edu/

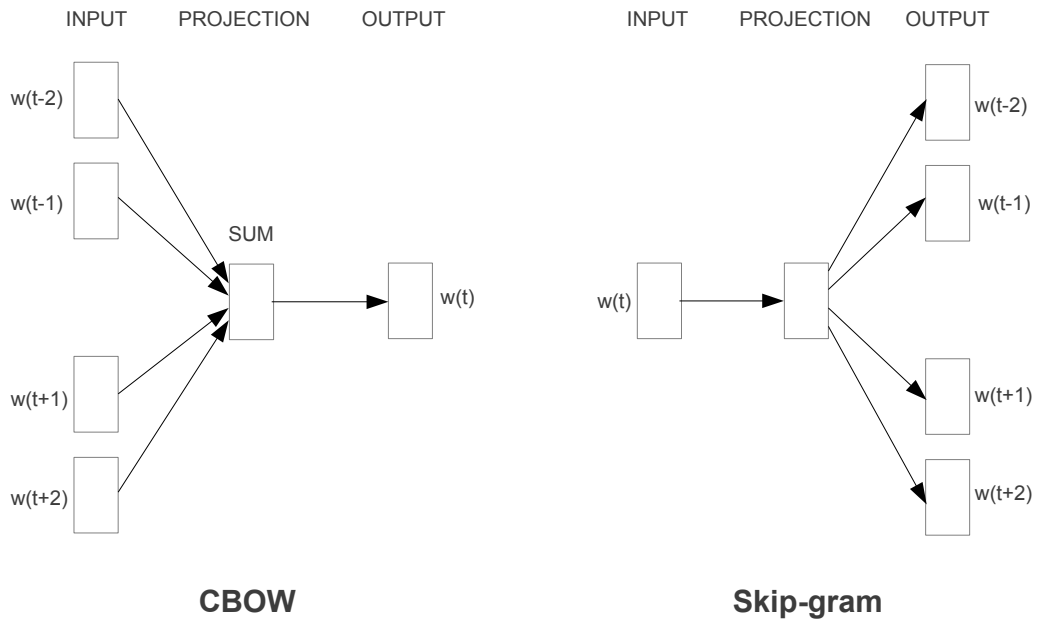


FIGURE 4.3: Predictive models of word embedding (Mikolov et al., 2013)

and Manning, 2014) embedding is trained by such model by normalizing the co-occurrence values of the words.

**Ex 4.6: Count-based vector representation**

| | |
|---|---|
| he goes to *school by* **car** | students go to *school by* **bus** |
| there *was a* **car** *accident* | *a* **bus** *accident injured* them |
| I *drive a* **car** *every weekend* | they *travel by* **bus** *every weekend* |

| | drive | by | a | every | accident | travel | school | weekend |
|---|---|---|---|---|---|---|---|---|
| car | 1 | 1 | 2 | 1 | 1 | 0 | 1 | 1 |
| bus | 0 | 2 | 1 | 1 | 1 | 1 | 1 | 1 |

In *predictive* models, either a word is predicted given the surrounding (context) words or the surrounding words are predicted for a given word. The method of training a predictive model for predicting a word based on the context words is called continuous bag-of-words (CBOW) model. In contrast, the method of predicting context words based on a given word called *continuous skip-gram* model. Concepts of the predictive models are illustrated in Fig. 4.3. The comparison between these two models made by Mikolov et al. (2013) shows that CBOW is faster than skip-gram for training the model and gets a little higher syntactic accuracy. However, Skip-


FIGURE 4.4: Encoding semantic differences in between the vectors (Mikolov, Yih, and Zweig, 2013)

gram model achieves higher semantic accuracy and represents better the rare words compared to CBOW. Word2Vec[2] is a widely used implementation to train predictive models based on both skip-gram and CBOW.

Word embeddings holds the syntactic and semantic regularities between words that seem to be encoded between the vectors of the words. For example, the differences between two words in terms of gender (male and female) and number (singular and plural) seem to be constant as shown in Fig. 4.4. Thus, two words can be examined whether these represent the same semantic type by calculating the cosine similarity of their vectors. For example, in a word embeddings, it is expected that *married* and *wife* would be in the same region of the semantic space. The cosine similarity between the vectors of these words is $0.86$ in GloVe embeddings. The score of cosine similarity ranges between $0$ and $1$ and a higher score means higher similarity. Thus *married* and *wife* are close to each other in terms of semantic similarity.

### 4.3.3 Recognition of Trigger Words

It can be possible to recognize an unknown trigger word of a relation based on a known trigger of that relation by calculating the cosine similarity between the word vectors as illustrated in Alg. 2. Suppose, *married, husband, wife* are the known trigger words of *spouse* relation and a sentence *There was a big party on wedding anniversary of John and Julie* claims to hold spouse relationship between John and Julie. Since the sentence does not contain any known trigger word of spouse relation, we compare each of the content words (except *John*, *Julie* and stopwords) in the sentence to the individual known trigger. We take the vectors of a content word and a known trigger from a pre-trained word embeddings. Then cosine similarity is computed between the pair of vectors.

[2]https://code.google.com/archive/p/word2vec/

**Algorithm 2** Calculate trigger word similarity

```
W : a context word from the sentence claiming to hold relation R
L : a list of trigger words of R
procedure GETTRIGGERWORDSIMILARITY(W, L)
    V_w ← GetVector(W)
    Score ← 0
    for each word X in L do
        V_x ← GetVector(X)
        d ← CosineSimilarity(V_w, V_x)
        if d > Score then
            Score ← d
    return Score
```

The best similarity score of $0.61$ is resulted between *wedding* and *married*. If any word completely matches to one of the known triggers the similarity score becomes $1$. High similarity score indicates that it is highly probable to consider the pair of words holding the same semantics. Therefore, we use the best similarity score as a feature for validating a relation hypothesis. A classifier automatically learns the threshold of the score to decide if the word is a trigger or not from the training data. Thus we incorporate word embeddings for recognizing unknown trigger words for relation validation task. The next section describes how we use the trigger words for relation validation task. Then in section 4.5, we will study different word embeddings for this task.

## 4.4 Syntactic-Semantic Fusion

Trigger words are useful for characterizing the types of the relations. It is important to inspect the position of the trigger words in a sentence carefully to validate a relation. For example, a system claims *parent* relationship between Julie and John in the first sentence of Ex. 4.5 on Page 59 where there is a trigger word *mother* to support the claimed relation. Though *mother* is a trigger for *parent* relation this sentence does not express that relation between Julie and John. Therefore, it requires to combine semantic and syntactic evidence for validating a relation hypothesis.

A sentence can be represented by a sequence of words and a dependency graph. The most common pattern of mentioning a relation is by placing a trigger word between the pair of entity mentions in a sentence as in the first sentence of Ex. 4.5 where *married* word contributes to express the *spouse* relation between Julie and John.

However, it cannot be always expected that a claimed relation would be true when a trigger of that relation places in between the entity mentions. For instance, the sentence in Fig. 4.5 contains *married* between Julie and Tim and a system claims spouse

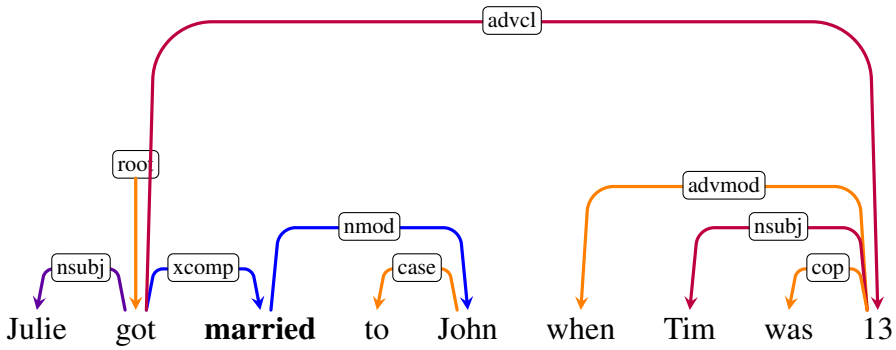


FIGURE 4.5: Finding trigger words in the dependency path

relationship between them. Though there is a trigger word *married* in between the mentions it does not mean the spouse relationship between them. However, the dependency path *Julie* → *got* → *13* → *Tim* (see Fig. 4.5) between the mentions does not contain any trigger to justify spouse relationship. In contrast, the hypothesis of *spouse* relationship between *Julie* and *John* satisfies the existence of any trigger word for both cases: between the mentions and in the dependency path. Therefore, we examine the existence of any trigger in the sequence of words when they are placed according to the dependency path. For example, in Fig. 4.5, according to the dependency path (*nsubj-xcomp-nmod*) between Julie and John, the word sequence is *got married* which contains a trigger *married*.

Moreover, sometimes, the trigger can be found outside the dependency path for justifying a relation. For example, in the sentence *John first met Julie three years before their marriage*, the trigger word *marriage* does neither lie between the mentions of John and Julie nor in between the dependency path as shown in Fig. 4.6. In such cases, we check if there is a trigger word in the minimum subtree. A minimum sub tree is a part of a dependency tree which mainly focuses on the target nodes and their direct neighbor nodes. Such trees have been explored by Chowdhury, Lavelli, and Moschitti (2011) for biomedical relation extraction. In order to find a trigger word in the minimum subtree, firstly we find out the common root node of the pair of mention nodes. Then we examine the directly connected child nodes of the root. In Fig. 4.6 the common root between John and Julie is *met* and the child nodes of *met* are *first*, *years* and *marriage* where *marriage* is a trigger word for *spouse* relation. We examine the directly connected child nodes in order to capture the trigger evidence in a shorter context. When the dependency tree of a long sentence contains several levels, including indirectly connected child nodes for inspection may lead to noisy context. We assume that the information in the shorter context is more reliable than in longer context.

In summary, for validating a claimed relation we inspect the existence of any

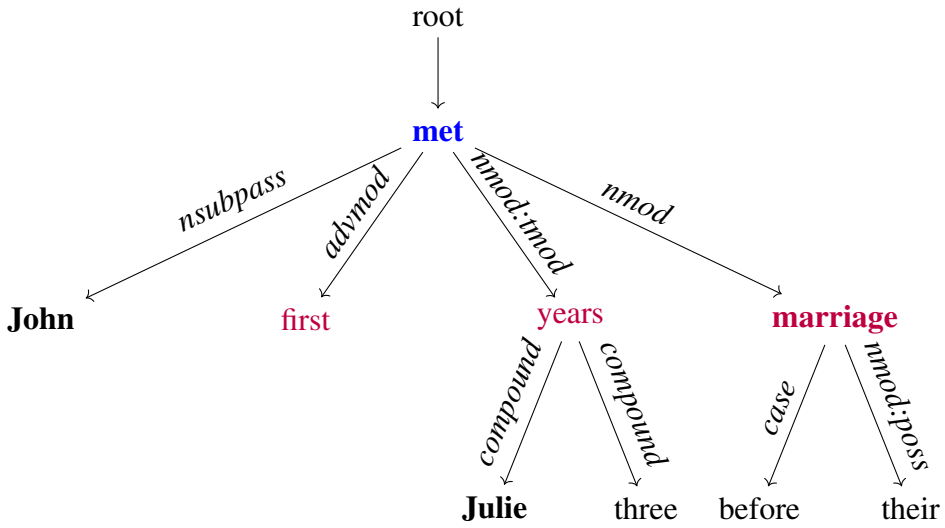


FIGURE 4.6: Finding trigger words in the minimum subtree

trigger word (i) between the entity mentions (ii) in between the dependency path and (iii) in the minimum subtree of a dependency parse tree.

## 4.5 Evaluation of Word-embeddings

We examine the quality of different word-embeddings in terms of trigger words similarity. In this evaluation, we use three pre-trained word-embeddings which are GloVe[3], GoogleNews-Vectors[4] and FastText[5]. Moreover, we build two more word-embeddings KBP-WordSG and KBP-CharSG by training the word2vec model with word skip-gram and character skip-gram accordingly on TAC KBP-2014 corpus. Table 4.4 represents the characteristics of these word-embeddings. In the pre-trained GloVe, GoogleNews-Vectors and FastText, each word is represented by a vector of 300 dimension. In contrast, KBP-WordSG and KBP-CharSG represent a word by a vector of 250 dimension. We reduce vector dimension for training the models rapidly. Our study finds that the reduction of vector dimension from 300 to 250 does not degrade the quality a word embeddings.

The trigger words of a relation denote the same semantic. Therefore, we expect that trigger words of the same relation would lie on the same vector space in a good word-embeddings. In order to evaluate the qualities of these word-embeddings, we compute the similarity between every pair of trigger words and standard deviation of the similarity scores. For example, *wife, husband, married* are trigger words of *spouse* relation. We compute similarity of *(wife, husband)*, *(husband, married)*

[3]https://nlp.stanford.edu/projects/glove/
[4]https://github.com/mmihaltz/word2vec-GoogleNews-vectors
[5]https://github.com/facebookresearch/fastText

| Word-embeddings | Model | Training Corpus (Token Count) | Vocabulary | Dimension |
|---|---|---|---|---|
| GloVe | word co-occurrence statistics | Wikipedia + Gigaword (6 B) | 400 K | 300 |
| GoogleNews-Vector | word2vec word skip-gram | Google News (3 B) | 3M | 300 |
| FastText | word2vec character skip-gram | Wikipedia (1.9 B) | 2.5 M | 300 |
| KBP-WordSG | word2vec word skip-gram | TAC KBP-2014 (11 B) | 3.5 M | 250 |
| KBP-CharSG | word2vec character skip-gram | TAC KBP-2014 (11 B) | 3.5 M | 250 |

TABLE 4.4: Characteristics of different word-embeddings

and *(wife, married)* pairs. We expect that similarity scores of these pairs would also be similar. That means, the standard deviation of these similarity scores would be very low. Table 4.5 shows standard deviations of trigger-word similarity scores of 11 trigger-dependent relations measured on different word-embeddings. In most cases, KBP-CharSG results in lower standard deviation compared to other embeddings. It gets the lowest standard deviation for 9 relations out of 11. In contrast, KBP-WordSG obtains the lowest standard deviation for only 2. Word-embeddings trained on TAC KBP-2014 result lower standard deviation compared to pre-trained embeddings on other corpora. An average standard deviation of 0.092 of 11 relations has been scored by KBP-CharSG. On the other hand, GloVe, GoogleNews-Vector, FastText and KBP-WordSG result average standard deviation of 0.120, 0.143, 0.128 and 0.126 respectively. According to these scores, it seems that a word-embeddings trained on TAC KBP-2014 corpus with word2vec character skip-gram model is more effective to find the trigger words of a relation. We can expect better performance from this word-embeddings to identify the unknown triggers of a relation.

| RelationName | GloVe | GoogleNews-Vector | FastText | KBP-WordSG | KBP-CharSG |
|---|---|---|---|---|---|
| parents | 0.109 | 0.118 | 0.106 | 0.132 | **0.071** |
| children | 0.109 | 0.118 | 0.106 | 0.132 | **0.071** |
| spouse | 0.082 | 0.131 | 0.081 | 0.118 | **0.060** |
| city_of_birth | 0.078 | 0.131 | 0.132 | **0.106** | 0.107 |
| country_of_birth | 0.078 | 0.131 | 0.132 | **0.106** | 0.107 |
| city_of_death | 0.139 | 0.123 | 0.112 | 0.131 | **0.072** |
| country_of_death | 0.139 | 0.123 | 0.112 | 0.131 | **0.072** |
| top_members_employees | 0.156 | 0.179 | 0.149 | 0.157 | **0.127** |
| member_of | 0.167 | 0.223 | 0.199 | 0.153 | **0.110** |
| city_of_headquarters | 0.127 | 0.150 | 0.142 | 0.110 | **0.106** |
| country_of_headquarters | 0.127 | 0.150 | 0.142 | 0.110 | **0.106** |
| Average | 0.120 | 0.143 | 0.128 | 0.126 | **0.092** |

TABLE 4.5: Standard deviation of trigger-word similarity scores for different relations measured on different word-embeddings

We perform another experiment which focuses on validating relations as a binary classification task. The objective of this experiment is to evaluate the quality of a word-embeddings on identifying unknown triggers of a relation. In this

experiment, we use the three features that inspect the existence of trigger-words (Section 4.4) based on trigger-word similarity scores (Section 4.3.3) measured on a word-embeddings. We expect better classification performance based on the similarity scores measured on a better word-embeddings. The evaluation dataset consists of $3,945$ training instances where positive and negative examples count $1,443$ and $2,502$ instances accordingly. In contrast, the test dataset counts of $8,423$ instances where the number of positive and negative examples are $1,805$ and $6,618$ respectively.

| **Word-embeddings** | **# Features** | **Precision** | **Recall** | **F-score** | **Acc.** |
|---|---|---|---|---|---|
| GloVe | 3 | 42.33 | 62.66 | 50.52 | 73.70 |
| GoogleNews-Vector | 3 | 39.82 | 64.27 | 49.17 | 71.53 |
| FastText | 3 | 39.79 | 66.48 | 49.78 | 71.26 |
| KBP-WordSG | 3 | 40.34 | 69.31 | 51.00 | 71.46 |
| KBP-CharSG | 3 | 39.15 | 66.04 | 49.15 | 70.72 |
| GoogleNews-Vector + FastText | 6 | 41.51 | 65.65 | 50.86 | 72.81 |
| GoogleNews-Vector + KBP-CharSG | 6 | 41.20 | 63.43 | 49.96 | 72.77 |
| GloVe + KBP-WordSG | 6 | 42.68 | 63.60 | 51.08 | 73.89 |
| FastText + KBP-CharSG | 6 | 42.42 | 65.37 | 51.45 | 73.56 |
| GoogleNews-Vector + KBP-WordSG | 6 | 43.97 | 64.21 | 52.20 | **74.80** |
| KBP-WordSG + KBP-CharSG | 6 | 42.47 | 67.20 | 52.05 | 73.47 |
| **GoogleNews-Vector + FastText + KBP-WordSG + KBP-CharSG** | **12** | **44.26** | **68.37** | **53.73** | 74.77 |

TABLE 4.6: Performance of relation validation by different word-embeddings (trained on KBP-2015 dataset and tested on KBP-2016 dataset)

The upper part of Table 4.6 represents the performance of relation validation by individual word-embeddings. We see that the best recall and F-score is obtained by KBP-WordSG. In contrast, KBP-CharSG results in the lowest precision and F-score which are $39.15$ and $49.15$ accordingly. The F-scores obtained by other embeddings ranges between $49.17$ to $50.52$. These scores indicate that no individual word-embeddings gets a significantly better score. Therefore, we combine the trigger-word similarity scores resulted by different word-embeddings. This combination method increases the number of features. For example, the combination of KBP-WordSG and KBP-CharSG counts $6$ features where each of them counts $3$ features by inspecting the existence of trigger words. These features differ in terms of similarity scores measured on two different embeddings. Thus a combination of $4$ word-embeddings counts $12$ features. The lower part of Table 4.6 shows the performance of relation validation by different combinations of word-embeddings. We observe that two combinations *GoogleNews-Vector + KBP-WordSG* and *KBP-WordSG + KBP-CharSG* outperform the best individual embeddings by around $1$ point in term of F-score.

These two combinations obtain the F-scores of $52.20$ and $52.06$ accordingly. Moreover, these combined embeddings perform better than some of the combinations of two embeddings. Interestingly, we notice that each embeddings of these two combinations has been trained by word2vec model. Moreover, they include at least one word-embeddings which is trained on TAC KBP-2014 corpus. However, we achieve the best result by the combination of four embeddings *GoogleNews-Vector* + *FastText* + *KBP-WordSG* + *KBP-CharSG* in terms of precision, recall and F-score. This combination results in an F-score of $53.73$ which beats the best individual embeddings (KBP-WordSG) by around $3$ points. Moreover, it outperforms the paired combination (GoogleNews-Vector + KBP-WordSG) by around $1.5$ points. These results depict that the combination of word skip-gram and character skip-gram based word2vec models and models trained on TAC KBP-2014 corpus improves the performance of relation validation by identifying the trigger words more precisely.

## 4.6 Conclusion

This chapter discussed on the linguistic aspects for relation validation task. Semantic and syntactic analysis facilitate extracting effective features regarding this task. Lexical semantics is able to characterize the types of trigger dependent relations. On the other hand, syntactic patterns denote grammatical structures of sentences expressing some relations. In this chapter, we focused on the state-of-the-art linguistic features and explored some new aspects for validating a claimed relation. We proposed dependency-pattern-edit-distance to generalize the syntactic patterns of the sentences expressing the same relation. Moreover, we presented a method of identifying unknown trigger words by using the known triggers and word-embeddings.

# Chapter 5

# Relation Validation Framework

This chapter presents our relation validation system. Here, we firstly focus on our relation validation model with a system overview. Then we describe the evaluation corpus. Finally, the evaluation metrics are defined for relation validation and KBP tasks.

## 5.1 Relation Validation Model

We want to validate a claimed relation whether it is correct or wrong. Given a relation hypothesis, (R, $E_q$, $E_c$) and a justifying sentence, $S$, the objective is to predict either the relation hypothesis is correct or wrong by analyzing the community graphs ($G_q$ and $G_c$) of $E_q$ and $E_c$ and by performing some linguistic analysis on $S$.

We consider relation validation as a binary classification task. Therefore, a binary classifier is trained with several features which are computed on the positive and negative examples. Then the classifier predicts a relation hypothesis under validation if it is correct or wrong.

Furthermore, we populate a knowledge base after validating the relation hypotheses which are generated by different systems. In KBP task, there are two types of slots, single-valued and multi-valued. A single-valued slot (e.g. per:country_of_birth) has to be filled up by only one filler value. In contrast, a multi-valued slot (e.g. per:country_of_residence) can be filled up by several non-repeating filler values. In order to select the correct filler values of a slot filling query for KBP, we apply the algorithm as shown in Alg. 3. The inputs to the model are a query relation name or slot name ($R$), subject entity of the query relation ($E_q$), list of candidate objects ($E_c$), list of relation justifying sentences ($S$), a graph of entities ($G$) and a pre-trained binary classifier ($M$). For each candidate ($E_{ci}$), a feature vector ($V_{ci}$) is generated and then a classifier predicts whether $(R, E_q, E_{ci})$ is true or false. All the candidates ($E'_c$) classified as true are selected as the filler values of $R$ if it refers to a multi-valued slot. However, if $R$ is a single-valued slot, we take the first element from the list $E'_c$.

### 5.1.1 Relation Validation Features

We first remind the features based on the graph and linguistic analysis for validating relations as discussed in Chapter 3 and Chapter 4. Additionally, we will present some trustworthy features based on the agreements among different system responses. Table 5.1 summarizes the features we used in our model.

**Algorithm 3** Slot filling by relation validation

1: $R$ : *a query relation name or slot name*
2: $E_q$ : *subject entity of the query relation, R*
3: $E_c = \{E_{c1}, E_{c2}, ..., E_{cn}\}$ : *list of candidate objects of the relation, R*
4: $S = \{S_{c1}, S_{c2}, ..., S_{cn}\}$ : *list of justifying sentences corresponding to each* $E_{ci} \epsilon E_c$
5: $G =$: *a graph of entities*
6: $M =$: *a pre-trained binary claasifier*
7: **procedure** GETSLOTFILLERVALUES($R, E_q, E_c, S, G, M$)
8: $E'_c \leftarrow$ *initialize an empty list*
9: **for** *each candidate* $E_{ci}$ *in* $E_c$ **do**
10: $H \leftarrow (R, E_q, E_{ci})$
11: $V_{ci} \leftarrow$ *generate feature vector for* $(R, E_q, E_{ci})$
12: $L \leftarrow$ *Classify*$(M, V_{ci})$
13: **if** $L = TRUE$ **then**
14: $E'_c \leftarrow E_{ci}$
15: **if** *R is a multi-valued slot* **then**
16: **return** $E'_c$
17: **else if** *R is a single-valued slot* **then**
18: **return** *top element from* $E'_c$

(a) **Graph Features**

We propose four graph features for validating a claimed relation between two entities as described in Section 3.5 on Page 45. Here we briefly remind these features.

(i) **Network Similarity:** we compute the similarity between the communities of a subject ($E_q$) and an object ($E_c$). We hypothesize that the community of a subject entity would be more similar to the community a true candidate than the community a false candidate.

(ii) **Network Density:** we combine the community networks of a subject ($E_q$) and an object ($E_c$) of a relation hypothesis and compute density of the merged network. We assume that the density would be higher for a correct object than a wrong one when the community of an object entity is merged with the community of the subject entity.

(iii) **Eigenvector centrality:** we hypothesize that a subject of a relation hypothesis would be more influenced by a true candidate object than a false candidate. Eigenvector centrality measures the influence of a node in a network. A node will be even more influential if it is connected to other influential nodes. Therefore, we quantify the influence of a candidate object ($E_c$) to a subject entity ($E_q$) by measuring Eigenvector centrality.

| Feature Group | Feature Name |
|---|---|
| Graph | Network similarity between the communities of $E_q$ and $E_c$ of a relation hypothesis, (R, $E_q$, $E_c$) |
| | Network density by merging the communities of $E_q$ and $E_c$ of a relation hypothesis, (R, $E_q$, $E_c$) |
| | Eigenvector centralities of $E_q$ and $E_c$ of a relation hypothesis, (R, $E_q$, $E_c$) |
| | Mutual information between the communities of $E_q$ and $E_c$ of a relation hypothesis, (R, $E_q$, $E_c$) |
| Linguistic | Dependency pattern minimum edit distance |
| | Dependency pattern length |
| | Existence of the subject and object entities in the same clause of $S$ |
| | Trigger word between entity mentions |
| | Trigger word in the dependency path between entity mentions |
| | Trigger word in the minimum subtree between entity mentions |
| | Is trigger-dependent relation |
| Trustworthy | Filler Credibility |
| | Document Credibility |
| | System Credibility |
| | Response confidence |

TABLE 5.1: Relation validation features

(iv) **Mutual Information:** the amount of information shared between two variables can be measured by mutual information. We assume that a subject entity shares more information with a true candidate object than a false one. Therefore, we quantify the shared information between two entities ($E_q$ and $E_c$) by measuring mutual information between them in their communities.

(b) **Linguistic Features**

Validating a claimed relation requires linguistic information for capturing syntactic patterns of a relation expression and characterizing the semantic type of a relation. Therefore, we use several linguistic features regarding this task.

(i) **Dependency Patterns:** expression of relations follow some syntactic dependency patterns. Matching these patterns is useful for relation extraction. However, a relation can be expressed by several different patterns. Some of these patterns may be unidentified. Therefore, we propose to calculate edit distance between a pattern under inspection and a set of pre-identified patterns. Moreover, length of the dependency path between a pair of entity-mentions is used as a feature for validating relation.

(ii) **Inspection of the Entity Pair in the Same Clause:** we hypothesize that the related pair of entities are mentioned in the same clause if the sentence of expressing relation is complex or its length is comparatively long. Therefore, we inspect whether the related pair of entities is mentioned in the same clause.

(iii) **Inspection of the Trigger Words:** relations are mostly expressed in a sentence by some trigger words. Therefore, we inspect existence of any trigger word between a pair of entity mentions, in the syntactic dependency path and in the minimum dependency tree (see Section 4.4).

(iv) **Indication of Relation Type:** the existence of trigger word is not mandatory for some relations such as *per:city_of_residence*. We use a boolean indication of relation type either trigger-dependent or trigger-independent as a feature for validating relation.

(c) **Trustworthy Features**

We define voting features to observe the trustworthy influence of multiple systems on validating claimed relations. Therefore, we calculate a credibility score (*voting*) of a candidate object based on all the responses given by different systems for a given subject and relation name.

Suppose, $E_c = \{E_{c1}, E_{c2}, E_{c3}\}$ is the list of candidate objects generated by three systems $S = \{S_1, S_2, S_3\}$ with respect to a query relation $(R, E_q, ?)$ with id $Q$ and $D = \{D_1, D_2, D_3, D_4\}$ be the documents where the candidates are mentioned. The relations among the candidates, systems and documents are as follows: $E_{c1}(S_1, S_2, D_1, D_3)$, $E_{c2}(S_2, D_2)$, $E_{c3}(S_1, S_3, D_1, D_4)$. We compute the Filler Credibility of a candidate by using Eq. 5.1.

$$Filler\,Credibility(E_{ci}, Q) = \frac{\#\,occurrences\,of\,a\,candidate\,E_{ci}\,for\,query\,Q}{\#\,occurrences\,of\,all\,the\,candidates\,for\,Q} \tag{5.1}$$

The Filler Credibility counts the relative votes for a candidate which indicates the degree of agreement by different systems to consider the candidate as correct. Since we can assume that systems already performed some analysis to make the responses, Filler Credibility holds strong evidence for a candidate to be correct. Such measurement has been used by Sammons et al. (2014) where candidate objects have been validated based on the majority voting.

We also compute the Document Credibility of a referenced document in a response to a query relation by Eq. 5.2. The Document Credibility measures the reliability of a source document where a candidate object is mentioned.

$$Document\,Credibility(D_i, Q) = \frac{\#\,occurrences\,of\,a\,document\,D_i\,for\,query\,Q}{\#\,occurrences\,of\,all\,the\,documents\,for\,Q} \tag{5.2}$$

Moreover, the System Credibility is measured to quantify the reliability of a system by counting the number of common candidates among different systems. We build a two-dimensional matrix where the rows and columns refer to the slot filling systems. At first, the matrix is initialized by zero. If two systems respond with the same filler to a query, we increase the credibility of both systems by 1. We repeat this process for all the queries. After completing this process with all the queries, we count total credibility of each system by adding the values of the corresponding row or column. Finally, the credibility values are normalized by the top score. We also include the confidence score of a response (which is given by a system) as a feature for validating relations.

### 5.1.2 Relation Validation System Overview

We build a system to integrate different components of our relation validation model. This system basically contains three components: input processing, feature extraction and binary classification. Figure 5.1 depicts the different components of the relation validation system.

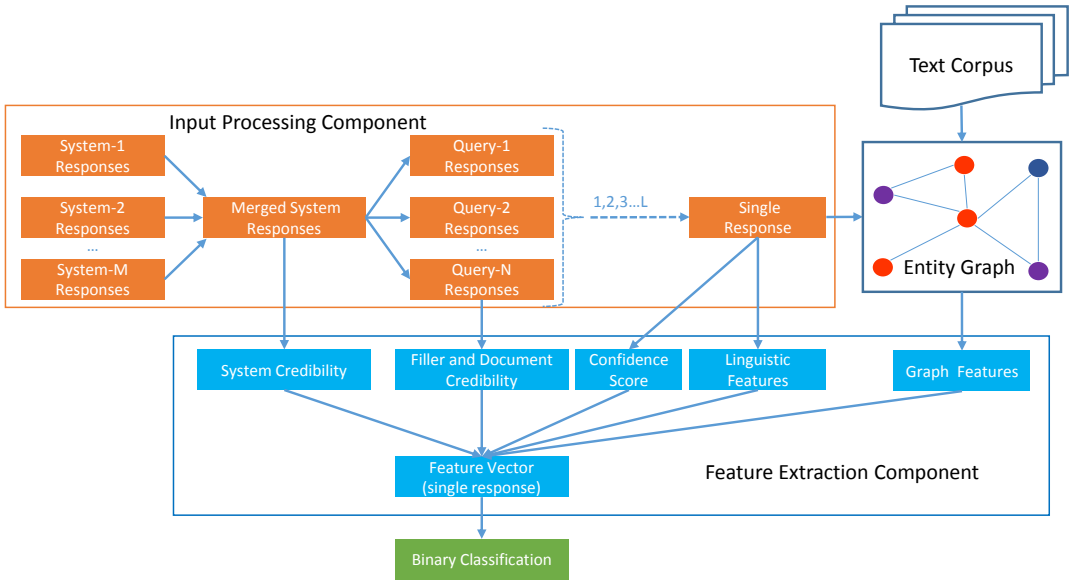


FIGURE 5.1: System overview of the relation validation model

- **Input processing:** all the responses of different slot filling systems are merged into a single file and responses are grouped into individual files regarding the query ids.
- **Feature extraction:** at this level, we generate a feature vector for each response of a query by analyzing the relation provenance text, system ids, document ids, filler values and community graphs.

| Dataset | newswire | discussion_forum | web | Total |
|---|---|---|---|---|
| KBP-2014 | 1,000,257 | 99,063 | 999,999 | 2,099,319 |
| KBP-2015 | 8,938 | 40,186 | 0 | 49,124 |
| KBP-2016 | 15,001 | 15,001 | 0 | 30,002 |

TABLE 5.2: Number of documents in the KBP corpus of three different years

- **Binary classification:** finally each response is classified as correct or wrong by using a pre-trained classifier.

## 5.2 Corpus and Preprocessing

In our research, we used the corpus of KBP Slot Filling (English) evaluation task in 2014, 2015 and 2016 which have been provided by NIST. This section illustrates these corpora and their preprocessing for constructing a graph of entities.

### 5.2.1 KBP Slot Filling Corpora

Each of the KBP document corpora consists of *newswire* and *discussion_forum* document. The corpus of KBP-2014 includes *web* data in addition to the newswire and discussion_forum. Each of the document in the corpus is XML formatted and contains an *id*. The web data are noisier compared to the newswire and discussion_forum. Among these three types of data, the increasing order of noisiness is newswire, discussion_forum and web data. Table 5.2 shows the statistic of KBP corpus in different years. In KBP-2014 corpus, there are $2,099,319$ documents in total where newswire, discussion_forum and web individually counts $1,000,257$, $99,063$ and $999,999$ accordingly. A total of $49,124$ documents are counted in KBP-2015 corpus where $8,938$ are newswire and $40,186$ are discussion_forum data. On the other hand, KBP-2016 corpus consists of $30,002$ documents where both of newswire and discussion_forum individually counts $15,001$.

A newswire document follows the markup as shown in Ex. 5.1 where the *id* refers to the unique identifier of the document and *type* (=*story*) refers to the type of a document if it is characterized as *story*. However, the *HEADLINE* and *DATELINE* tags are optional and the *TEXT* content may include *<P> ... </P>* tags if *doc_type_label* is *story*. A discussion_forum data may contain multiple posts as the markup described in Ex. 5.2. There may also be arbitrarily deep nesting of quote elements, and other elements e.g. *<a...>...</a>* tags in the discussion_forum documents.

Furthermore, Ex. 5.3 represents the markup for the web documents where most of the web documents contain <*QUOTE* ...> tag without having the end tag, </*QUOTE*>. A portion of text in the discussion_forum and web data may repeat under the tags <quote> and <QUOTE> accordingly when the same topic is shared among different users and comments are made on it.

**Ex 5.1: Markup framework of newswire data**

```
<DOC id="{doc_id_string}" type="{doc_type_label}">
        <HEADLINE>
                ...
        </HEADLINE>
        <DATELINE>
                ...
        </DATELINE>
        <TEXT>
                <P>
                        ...
                </P>
                ...
        </TEXT>
</DOC>
```

**Ex 5.2: Markup framework of discussion_forum data**

```
<doc id="{doc_id_string}">
        <headline>
                ...
        </headline>
        <post ...>
                ...
                <quote ...>
                        ...
                </quote>
                ...
        </post>
        ...
</doc>
```

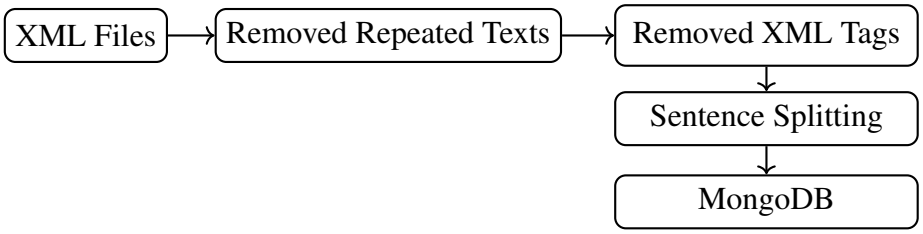


FIGURE 5.2: Pipeline of processing the XML formatted source files

**Ex 5.3: Markup framework of web data**

```
<DOC>
    <DOCID> {doc_id_string} </DOCID>
    <DOCTYPE> ... </DOCTYPE>
    <DATETIME> ... </DATETIME>
    <BODY>
        <HEADLINE>
            ...
        </HEADLINE>
        <TEXT>
            <POST>
                <POSTER> ... </POSTER>
                <POSTDATE> ... </POSTDATE>
                ...
            </POST>
            ...
        </TEXT>
    </BODY>
</DOC>
```

Preprocessing of the corpus was done by the team of IMM project at IRT SystemX with the IMM platform (Mesnard et al., 2016). Firstly, all the documents of these three categories of data were parsed and stored in a MongoDB database. The pipeline of processing the XML formatted source files is shown in Fig. 5.2 that makes the collection of text corpus. The markup tags were removed from the documents to make them clean and for better linguistic preprocessing. Moreover, duplicate instances of texts (according to the <quote> or <QUOTE> tag) were removed. While removing these texts, the original positions of the remaining texts were preserved. Basically, the repeated texts were replaced by blank spaces where the number of blank spaces was equal to the number of characters in a repeated text. IRT SystemX used KBP corpus for developing a slot filling system. If a mention of a relation between two entities is repeated by the same text-segment in the same document, counting more instances of the relation may bias a slot filling system. Therefore, removing the repeated texts was important to avoid the bias factor. Then each of the documents was splitted into sentences by using Stanford CoreNLP (Manning et al., 2014a) and both the original and cleaned documents were stored in MongoDB for further use.

| Entity Type | Entity Count | |
|---|---|---|
| | KBP-2015 | KBP-2016 |
| person | 62,267 | 20,971 |
| organization | 75,552 | 34,293 |
| location (country) | 292 | 271 |
| location (state/province) | 2,088 | 2,040 |
| location (city) | 12,384 | 7,814 |
| All Together | 152,583 | 65,389 |

TABLE 5.3: Statistics of the entities in the two graphs

We construct two Neo4j graph databases from KBP-2015 and KBP-2016 corpus as described in Section 3.3 on Page 38. Table 5.3 summarizes the statistics of these two graphs. There are $152,583$ and $65,389$ entities (person, organization and location) in the graphs constructed on the KBP-2015 and KBP-2016 corpus accordingly. Moreover, these graphs consist of $805,216$ and $488,198$ *IN_SAME_SENTENCE* links among different entity mentions respectively. We construct two different graphs because the query entities and the corresponding filler entities are different in the KBP tasks of 2015 and 2016. Thus constructing the two graphs helps to find the information regarding these entities.

### 5.2.2 KBP Slot Filling Responses and Snippet Assessments

The KBP task defines a set of slot filling queries by the entity name, reference document, offset of the entity mention, entity type and the slot name as depicted in Ex. 5.4.

**Ex 5.4: A slot filling query**

```
<query id="CSSF_ENG_0e7479c2b6">
    <name>Nelson Mandela</name>
    <docid>a69c5c79caa4c2b2869775fabcbabc7f</docid>
    <beg>174</beg>
    <end>187</end>
    <enttype>per</enttype>
    <slot>per:spouse</slot>
</query>
```

A slot filling system has to respond with the filler value of the query and some additional information according to a predefined format of 8 tab-separated columns

as listed in Ex. 5.5. TAC provides the assessments of a subset of the query-responses by different systems after the evaluation. The assessment file contains the queries, responses and assessments of the responses. Each response is assessed based on the filler value and its justification by the relation provenance snippets.

**Ex 5.5: Format of a slot filling response**

*column 1: Query Id*
*column 2: Slot Name*
*column 3: A unique run Id of a SF system*
*column 4: Provenance for the relation (doc ID and offsets)*
*column 5: A slot filler*
*column 6: A filler type (PER, ORG, GPE, STRING)*
*column 7: Provenance for the filler value (doc ID and offsets)*
*column 8: Confidence score*

The assessment of a relation provenance snippet can be either correct (C), wrong (W), inexact-short (S) or inexact-long (L) where inexact-short means that the snippet does not contain enough information to justify the relation and inexact-long indicates that the snippet is too long having some unnecessary information. There are 8 tab-separated columns in the assessment file as the format is illustrated in Ex. 5.6.

**Ex 5.6: Format of the assessment regarding a slot filling response**

*column 1: Response Id*
*column 2: Concatenation of Query Id and Slot Name*
*column 3: Provenance for the relation (doc ID and offsets)*
*column 4: A slot filler*
*column 5: Provenance for the filler value (doc ID and offsets)*
*column 6: Assessment of the filler (Correct(C) or Inexact(X) or Wrong(W))*
*column 7: Assessment of the relation provenance (Correct(C) or Inexact-Long(L) or Inexact-Short(S) or Wrong(W))*
*column 8: LDC equivalent class of column 4 (slot filler) if the filler is correct or inexact (in column 6)*

We compile the positive and negative instances of the different relations based on the queries, assessments of the responses and the document corpus. Fig. 5.3 shows the high level block diagram of compiling labeled snippets (positive and negative) of different relations from the slot filling responses assessed by TAC. The processor takes the queries, responses and assessment file as the inputs and extracts the snippets from the corpus based on the information provided in the responses and assessments. The format of an assessed relation instance is shown in Ex. 5.7 which consists of

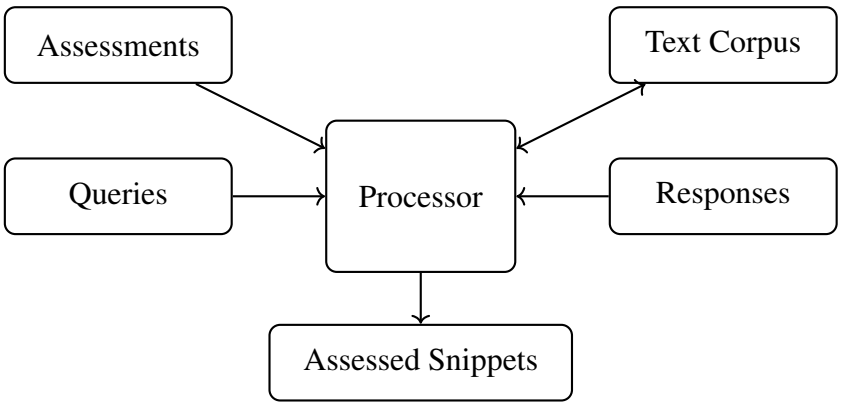


FIGURE 5.3: Compilation of assessed snippets from the queries, responses, assessments and corpus

| Dataset | # Query | # Positive Instance | # Negative Instance | # Total |
|---|---|---|---|---|
| KBP-2014 | 1,589 | 3,454 | 10,192 | 13,646 |
| KBP-2015 | 4,416 | 26,608 | 80,672 | 107,280 |
| KBP-2016 | 925 | 4,531 | 25,475 | 30,006 |

TABLE 5.4: Statistics of the assessed responses to the queries of the KBP tasks in three different years

different information in 8 tab separated columns. Since the slot filling response and assessment file refer to the relation provenance text by document Id and offset, we collect the text snippets regarding the responses from the document corpus based on the document Id and offsets. A snippet is considered as positive or negative if the relation provenance assessment is C or W accordingly. We do not take into account the snippets that are assessed as S or L because they are either incomplete or contain excess information accordingly.

**Ex 5.7: Format of an assessed snippet**

*column 1: Query Id*
*column 2: response Id*
*column 3: Unique run Id*
*column 4: Relation name*
*column 5: Query entity*
*column 6: Object entity*
*column 7: Relation provenance text*
*column 8: Assessment of the relation provenance text (C or W)*

Table 5.4 represents the statistics of the compiled assessed queries and responses

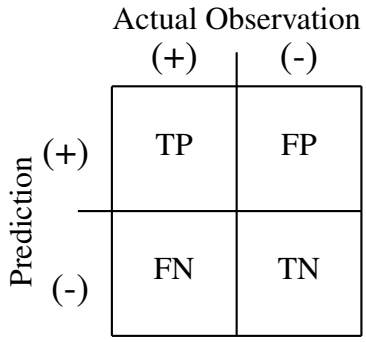


FIGURE 5.4: A model of confusion matrix for binary classification

from the datasets of three different years. It shows that the number of negative responses is much higher than the number of positive responses in all of the three datasets. Each of the assessed datasets covers 41 types of relations.

## 5.3 Evaluation Metrics

Evaluation is important for determining the performance of a method or a system. It quantifies the merit of a system by calculating some metrics governed by a set of standards. Thus quantitative measurements of the metrics facilitate to compare the performances of different systems that perform the similar task.

The evaluation metrics differ for different tasks. Information extraction related tasks such as Message Understanding Conference (MUC) (Grishman and Sundheim, 1996) and KBP (Surdeanu and Ji, 2014) are usually evaluated by precision, recall and F-score.

In the classification based tasks such as relation classification (Girju et al., 2007) precision and recall is calculated based on the confusion matrix (Sokolova and Lapalme, 2009). The accuracy also gives a good indication of performance measurement for classification task which calculates the ratio of correctly classified instances to the total number of instances. Fig. 5.4 shows a confusion matrix model of binary classification where the observations come from two classes *positive* (+) and *negative* (-). A confusion matrix is described by four terms *true positive* (TP), *false positive* (FP), *false negative* (FN) and *true negative* (TN) which are defined below:

**Terms of the confusion matrix**

**TP** counts the number of positive observations predicted as positive.
**FP** counts the number of negative observations predicted as positive.
**FN** counts the number of positive observations predicted as negative.
**TN** counts the number of negative observations predicted as negative.

In the classification task, the precision, recall and accuracy are calculated by the Eq. 5.3, Eq. 5.4 and Eq. 5.5 based on the TP, FP, FN and TN. However, the calculation of F-score remains the same as it is defined for information extraction task in Eq. 5.6.

$$Precision = \frac{TP}{TP + FP} \tag{5.3}$$

$$Recall = \frac{TP}{TP + FN} \tag{5.4}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{5.5}$$

$$F_1 = \frac{2 * precision * recall}{precision + recall} \tag{5.6}$$

In both information extraction and classification tasks the objective of a system is to obtain high precision and recall so that a high F-score can also be achieved. Sometimes, high recall is expected at the cost of precision and vice versa to get a high F-score. A high recall is expected when a system has to extract or predict a large number of positive observations. On the other hand, when a system is required to say confidently an extracted information or predicted observation is correct the precision gets more importance. We expect high precision at the cost of the recall to achieve a good F-score for our relation validation task. A high accuracy is desired as well.

A KBP system (slot filling and slot filler validation) is also evaluated by precision, recall and F-score but the definition of precision and recall are a little different than traditional definitions. The precision and recall for KBP task are measured by the Equations 5.7 and 5.8 respectively. For calculating the recall, ground truths of the queries have been defined base on the total number of possible non-redundant correct filler values.

$$precision = \frac{number\,of\,correct\,objects}{number\,of\,correct\,and\,false\,objects} \tag{5.7}$$

$$recall = \frac{number\,of\,correct\,objects}{total\,number\,of\,ground\,truth} \tag{5.8}$$

## 5.4 Conclusion

This chapter presented our relation validation framework, the different corpus and the evaluation metrics. We consider relation validation as a binary classification task. In

the next chapter, we will do the experiments related to the relation validation and KBP tasks.

# Chapter 6

# Experiments and Results

THis chapter illustrates the experiments related to the tasks of relation validation and knowledge base population (KBP). Firstly we build several supervised relation validation models based on different feature combinations and observe their contributions on the slot filler validation (SFV) task. Then we investigate drawbacks of these models and propose some improvements. Finally, we improve our supervised relation validation models and evaluate them on a larger corpus. We also evaluate an unsupervised relation validation model to realize its performance. Moreover, we employ these relation validation models for KBP task and compare KBP scores to different systems.

## 6.1 Participation to TAC KBP-2016 SFV Task

NIST conducts SFV task in the TAC KBP workshop which aims at ensembling the slot filling responses of different systems to construct a more precise knowledge base. We participated in the TAC KBP-2016 SFV ensemble task. NIST provided the assessments of the responses of slot filling queries used in TAC KBP-2014 and KBP-2015 tasks. We used them for learning purpose.

In the assessment of TAC KBP-2015 dataset, the responses of around $4,400$ slot filling queries were assessed. A lot of queries have been answered with only wrong responses. Therefore, we do not take into account these queries for building our training corpus. We select the queries that have been answered with at least one correct and one wrong response. This subset counts total $1,296$ slot filling queries. We have extracted linguistic features for around $55,276$ responses from $68,076$ responses that contain mentions of the subject and object entities of a claimed relation. Our system fails to detect many named entities from texts. Moreover, in many cases, entities are mentioned by pronouns. We do not use any tool of co-reference resolution due to lower performance of the existing tools. Therefore, in many cases, our system cannot detect both the subject and object entities of a relation which are mentioned in the same sentence. As a result, we are not able to extract linguistic and graph features for a large number of responses. Our system extracted both linguistic and graph features for around $4,321$ responses from $260$ queries.

### 6.1.1 Evaluation of Different Feature Groups

Here we observe the relation validation performances of three feature sets based on trustworthy, linguistic and graph analysis. We perform 10-fold cross-validation by

Random Forest classifier on both datasets which count $55,276$ and $4,321$ responses accordingly. We use Random Forest classifier because it gives better performance than other classifiers on our data.

| **Feature Set** | **Precision** | **Recall** | **F-score** | **Accuracy** |
|---|---|---|---|---|
| $55,276$-response corpus | | | | |
| Trustworthy (all) | 89.79 | 86.59 | 88.16 | 93.27 |
| Trustworthy (Filler Credibility) | 70.65 | 65.32 | 67.88 | 82.11 |
| Linguistic | 67.80 | 48.78 | 56.74 | 78.47 |
| Trustworthy+Linguistic | 94.03 | 93.27 | 93.64 | 96.34 |
| $4,321$-response corpus | | | | |
| Trustworthy (all) | 86.70 | 82.26 | 84.42 | 91.09 |
| Trustworthy (Filler Credibility) | 65.47 | 60.57 | 62.93 | 79.06 |
| Linguistic | 73.98 | 55.26 | 62.60 | 80.98 |
| Trustworthy+Linguistic | 92.36 | 89.59 | 90.95 | 94.77 |
| Trustworthy+Graph | 94.80 | 93.53 | 94.16 | 96.60 |
| Trustworthy+Linguistic+Graph | 95.09 | 93.22 | 94.15 | 96.60 |

TABLE 6.1: Relation validation evaluation by cross validation on KBP-2015 dataset

The upper part of Table 6.1 shows the cross-validation results on the responses having no graph feature. We observe that four trustworthy features collectively performs better than the linguistic features to classify correct and wrong responses. Trustworthy and linguistic features resulted in F-scores of $88.16$ and $56.74$ accordingly. Interestingly, Filler Credibility, as a single feature, obtains an F-score of $67.88$ which is around $11$ points higher than the linguistic features. This result indicates that trustworthy features are very effective for validating relations. However, the combination of trustworthy and linguistic features outperforms the trustworthy features by around $5$ points. This result signifies the contribution of linguistic analysis for validating relations.

Moreover, the lower part of Table 6.1 depicts the cross-validation results on the responses having graph features. Trustworthy feature set achieves better precision, recall, F-score and accuracy than the scores resulted by linguistic features. However, Filler Credibility and linguistic features obtain similar F-score although their precision and recall are different. Here we also notice better result by combining different feature sets. The combination of trustworthy and linguistic features results in F-score of $90.95$ which is around $6.5$ points higher than the trustworthy features.

Interestingly, graph features combined with trustworthy and linguistic features significantly improves the performance to classify correct and wrong responses. This combination achieves an F-score of $94.15$ which is around $10$ and $3$ points higher than the score of trustworthy and trustworthy+linguistic features accordingly. These results signify the effectiveness of graph features on validating relations although the experiment is performed on a small dataset.

### 6.1.2 Relation Validation Models for KBP-2016 SFV Task

We build different classifier models by merging the two datasets and combining different feature sets. The merged dataset counts in total $59,597$ responses. In order to handle the responses having no graph feature, we allow the classifier to assign some values to the graph features. In such cases, Random Forest classifier assigns the median value of a missing feature. Four relation validation models had been built as shown in Table 6.2. Cross-validation evaluation on these models show the performance order (descending) Trustworthy+Linguistic+Graph, Trustworthy+Linguistic, Trustworthy, Linguistic in terms of F-score. These models had been used in TAC KBP-2016 SFV evaluation task. We expected the best result from the model of Trustworthy+Linguistic+Graph combination because this model obtained the best score in cross-validation as shown in Table 6.1.

| **Relation Validation Model (Feature Set)** | **Precision** | **Recall** | **F-score** |
|---|---|---|---|
| Trustworthy | 83.85 | 84.41 | 84.13 |
| Linguistic | 74.76 | 63.68 | 68.77 |
| Trustworthy+Linguistic | 90.16 | 89.22 | 89.69 |
| Trustworthy+Linguistic+Graph | 90.61 | 89.17 | 89.88 |

TABLE 6.2: Relation validation models (by cross-validation) used in KBP-2016 SFV task

The upper part of Table 6.3 shows the official scores in KBP-2016 SFV ensemble task by employing four relation validation models. The relation validation models resulted very low scores in the official evaluation of SFV task. Moreover, Trustworthy+Linguistic+Graph did not get the best score among four relation validation models. We obtained the best F-score of $24.79$ by Trustworthy+Linguistic model which is around $5$ points higher than Trustworthy+Linguistic+Graph model. The model based on trustworthy features resulted in the second highest F-score of $23.90$ among four models. However, Linguistic and Trustworthy+Linguistic+Graph models obtained

similar F-score which is around 19.50. The lower part of this table presents the official scores obtained by three other participants. The best performance was achieved by Anonymous Participant-3 which scored F-score of 32.42. In contrast, two other Anonymous Participants obtained the F-scores of 27.34 and 28.64 accordingly. Our best model scored F-score 24.79 which is around 8 points lower than the score of Anonymous Participant-3. Our results were very frustrating as they did not meet our expectation.

| **Relation Validation Model** | **Precision** | **Recall** | **F-score** |
|---|---|---|---|
| Trustworthy | 22.02 | 26.13 | 23.90 |
| Linguistic | 17.50 | 21.91 | 19.46 |
| Trustworthy+Linguistic | **22.56** | 27.50 | **24.79** |
| Trustworthy+Linguistic+Graph | 14.48 | **29.76** | 19.49 |
| Anonymous Participant-1 | 24.19 | **31.43** | 27.34 |
| Anonymous Participant-2 | 33.08 | 25.25 | 28.64 |
| Anonymous Participant-3 | **37.78** | 28.39 | **32.42** |

TABLE 6.3: Official scores in KBP-2016 SFV ensemble task

### 6.1.3 Conclusion

Relation validation models evaluated by cross-validation and tested on different datasets resulted very different scores. Moreover, the relation validation model with graph features (Trustworthy+Linguistic+Graph) did not achieve the best score among four models. The performance of Trustworthy+Linguistic+Graph model on official evaluation indicated that the training data were not sufficient. Moreover, lower scores in the official evaluation of all the relation validation models (presented in the upper part of Table 6.3) required to investigate the datasets and features used for training and evaluating the models.

## 6.2 System Investigation

Our relation validation models are basically trained on TAC KBP-2015 dataset and evaluated on TAC KBP-2016 SFV task. Since the relation validation models do not perform well, we try to analyze both of the training and evaluation datasets. Moreover, all the features that we use in relation validation task may not be effective for different datasets. Therefore, we also analyze the effectiveness of the features.

### 6.2.1 Statistical Difference Between TAC KBP Evaluation Datasets in 2015 and 2016

In order to investigate the training and evaluation datasets, we compare these datasets statistically. In statistics, the measurement of *p*-value quantifies the significance of the difference between two samples. Student's t-test (Student, 1908) facilitates to measure the *p*-value. *p*-value ranges between $0$ and $1$. If the *p*-value is less than $0.05$ in a particular observation, the result of that observation is interpreted as significant. We measure the *p*-value of each feature score regarding the instances of training and evaluation datasets used in our relation validation task. Our objective is to observe whether these two datasets are significantly different.

| Feature Index | Feature Name |
|---|---|
| A | Has Trigger Word Between Entity Mentions |
| B | Has Trigger Word in Minimum Dependency Path |
| C | Has Trigger Word in Maximum Dependency Path |
| D | Has Trigger Word in Minimum Sub-tree |
| E | Minimum Dependency Pattern Length |
| F | Maximum Dependency Pattern Length |
| G | Positive Pattern Edit Distance |
| H | Are Object Pairs in a Single Clause |
| I | Filler Credibility (Trustworthy) |
| J | Document Credibility (Trustworthy) |
| K | System Credibility (Trustworthy) |
| L | Confidence Score (Trustworthy) |

TABLE 6.4: List of features used for analyzing differences between two datasets

In this investigation, we take into account the responses on $9$ relations which individually counts at least $50$ positive responses. Relations with fewer positive responses may not give us a good realization of the differences between two datasets. Table 6.5 shows the measured *p*-values of different features where a significant *p*-value is indicated with a star (*) mark. We see that almost all the feature values of different relations in the two datasets differ significantly. For example, in $6$ relations out of $9$, the feature values of column $A$ and $B$ (both are trigger word features as shown in Table 6.4) in the KBP-2015 and KBP-2016 datasets are significantly different. Only two syntactic features (column $F$ and $G$) have less difference between KBP-2015 and KBP-2016 datasets.

Since there are many negative responses compared to the number of positive ones, we measure *p*-value by balancing the positive and negative responses with a ratio of around $1:2$. That means each dataset counts two negative responses for a positive one for each query. Table 6.6 represents the *p*-values of balanced datasets. Here

| | A | B | C | D | E | F | G | H | I | J | K | L |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| per:employee_member_of | 0* | 0.023 | 0.081 | 0.018 | 0.636 | 0.816 | 0.003* | 0.024* | 0* | 0.029* | 0* | 0* |
| org:country_of_HQ | 0* | 0.441 | 0.681 | 0.003* | 0.048* | 0* | 0.002* | 0.004* | 0* | 0.400 | 0* | 0* |
| org:top_member_employee | 0.846 | 0* | 0* | 0.004* | 0.436 | 0.590 | 0.787 | 0.043* | 0.001* | 0* | 0* | 0.066 |
| per:parent | 0* | 0* | 0* | 0* | 0.011* | 0.651 | 0* | 0* | 0* | 0* | 0* | 0* |
| per:countries_of_residence | 0.820 | 0.676 | 0.118 | 0* | 0.008* | 0.094 | 0.087 | 0.026* | 0* | 0.015* | 0.685 | 0* |
| per:states_prov._of_residence | 0* | 0* | 0* | 0* | 0.168 | 0.558 | 0.949 | 0.523 | 0* | 0.033* | 0* | 0* |
| org:state_province_of_HQ | 0* | 0.093 | 0.074 | 0* | 0.061 | 0.092 | 0.086 | 0.739 | 0* | 0.001* | 0* | 0* |
| org:subsidiaries | 0* | 0* | 0* | 0* | 0.011* | 0.513 | 0* | 0* | 0* | 0.170 | 0* | 0* |
| org:city_of_HQ | 0.592 | 0* | 0* | 0* | 0.014* | 0.080 | 0.700 | 0.001* | 0.097 | 0* | 0.919 | 0.884 |
| count significant (*) | 6 | 6 | 5 | 9 | 5 | 1 | 4 | 7 | 8 | 7 | 7 | 7 |

TABLE 6.5: p-values measured on 9 relations including all of their positive and negative examples

we notice the similar behavior of the two datasets as shown in Table 6.5 in terms of *p*-value measurement.

| | A | B | C | D | E | F | G | H | I | J | K | L |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| per:employee_member_of | 0* | 0.140 | 0.595 | 0.077 | 0.340 | 0.790 | 0.041* | 0.337 | 0* | 0.001* | 0* | 0* |
| org:country_of_HQ | 0.041* | 0.852 | 0.265 | 0.027* | 0.235 | 0* | 0.050 | 0.038* | 0* | 0.524 | 0* | 0* |
| org:top_member_employee | 0.219 | 0* | 0* | 0* | 0.444 | 0.383 | 0.490 | 0.545 | 0.070 | 0* | 0* | 0.034* |
| per:parent | 0* | 0* | 0* | 0* | 0.700 | 0.053 | 0.006* | 0.010* | 0.001* | 0.074 | 0.026* | 0.001* |
| per:countries_of_residence | 0.358 | 0.437 | 0.831 | 0* | 0.200 | 0.394 | 0.529 | 0.070 | 0 | 0.038* | 0.951 | 0* |
| per:states_prov._of_residence | 0.013* | 0.004* | 0.034* | 0.279 | 0.111 | 0.256 | 0.792 | 0.295 | 0.048* | 0.861 | 0.001* | 0.040* |
| org:state_province_of_HQ | 0* | 0.400 | 0.281 | 0* | 0.614 | 0.544 | 0.735 | 0.942 | 0* | 0.003* | 0* | 0* |
| org:subsidiaries | 0* | 0* | 0* | 0* | 0.416 | 0.192 | 0.082 | 0* | 0* | 0.076 | 0* | 0* |
| org:city_of_HQ | 0.738 | 0* | 0* | 0* | 0.005* | 0.012* | 0.420 | 0.007* | 0.176 | 0* | 0.799 | 0.732 |
| count significant (*) | 6 | 5 | 5 | 7 | 1 | 2 | 3 | 4 | 7 | 5 | 7 | 8 |

TABLE 6.6: p-values measured on 9 relations by balancing the number of negative examples

According to the Table 6.5 and Table 6.6, in most cases, specially for the trigger word features (A-D) and trustworthy features (I-L) the datasets differ significantly. This difference might be one of the reasons of low score in the official evaluation of slot filler validation task.

### 6.2.2 Impact of the Trustworthy Features

We investigate the effects of trustworthy features on validating relations when the models are trained and tested on the subsets of KBP-2015 and KBP-2016 datasets accordingly. In this experiment, we balance the number of positive and negative examples of each dataset by a ratio of $1:2$. Before balancing, we remove the duplicate instances from both datasets. Thus, KBP-2015 dataset counts 24,624 instances (positive = 8,206 and negative = 16,418) and KBP-2016 dataset contains 4,882 instances (positive = 1,627 and negative = 3,255). Table 6.7 shows the performances

of different feature combinations when they are trained on the KBP-2015 dataset and tested on the KBP-2016 dataset. The set of four trustworthy features results in the precision, recall and F-score of 52.33, 49.78 and 51.02 accordingly. We notice that Filler Credibility gets better precision, recall and F-score than the combination of all trustworthy features by around 4, 11 and 7 points accordingly. Surprisingly, these two combinations do not exceed the scores obtained by Filler Credibility even though they are combined with linguistic features. The combination of Filler Credibility and linguistic features obtains the precision, recall and F-score of 54.98, 50.58 and 52.69 accordingly which are lower than the scores resulted by Filler Credibility alone. Moreover, the combination of four trustworthy and linguistic features obtains around 11 and 5 points lower recall and F-score than the scores resulted by Filler Credibility. However, this combination achieves the best precision of 58.90 which is around 2.65 points higher than the precision obtained by Filler Credibility.

| Feature Combination | Precision | Recall | F-score |
|---|---|---|---|
| Trustworthy (All) | 52.33 | 49.78 | 51.02 |
| Filler Credibility | 56.25 | **60.05** | **58.09** |
| Filler Credibility + Linguistic | 54.98 | 50.58 | 52.69 |
| Trustworthy (All) + Linguistic | **58.90** | 49.23 | 53.63 |

TABLE 6.7: Trustworthy feature investigation: models trained on KBP-2015 dataset and tested on on KBP-2016 dataset

However, we noticed different characteristics of the feature combinations by 10-fold cross validation on the subset of the KBP-2015 dataset as shown in Table 6.8. Here, the combination of four trustworthy features outperforms the Filler Credibility by around 19, 21 and 20 points in terms of precision, recall and accuracy accordingly. Moreover, Filler Credibility combined linguistic features results better scores than the Filler Credibility itself. The best precision, recall and F-score of 94.36, 94.03 and 94.20 have been achieved by combining all the trustworthy features and linguistic features.

All the scores observed in Table 6.7 and Table 6.8 indicate that the performances of trustworthy features which are dependent on systems and data in different years are not consistent. However, the Filler Credibility, which counts votes of agreement on the responses, seems consistent for validating relations.

| Feature Combination | Precision | Recall | F-score |
|---|---|---|---|
| Trustworthy (All) | 89.79 | 86.60 | 88.16 |
| Filler Credibility | 70.65 | 65.32 | 67.88 |
| Filler Credibility + Linguistic | 86.39 | 88.12 | 87.25 |
| Trustworthy (All) + Linguistic | **94.36** | **94.03** | **94.20** |

TABLE 6.8: Trustworthy feature investigation: 10-fold cross validation on a subset of KBP-2015 dataset

### 6.2.3 Impact of Trigger Words in the Slot Filling Responses

Relations are mostly expressed by trigger words and we use several features for validating relations based on the existence of any trigger word in a relation justifying sentence. We investigate the slot filling positive responses of KBP-2015 and KBP-2016 datasets whether there is any trigger word in a relation justifying sentence. The objective is to observe how two datasets differ in terms of trigger words. Here, we take into account only those slots that individually counts at least 30 positive responses.

| | **Dataset KBP-2015** | | | **Dataset KBP-2016** | | |
|---|---|---|---|---|---|---|
| **Relation Name** | **# Responses** | **# Resp. with Trigger** | **(%)** | **# Responses** | **# Resp. with Trigger** | **(%)** |
| per:children | 98 | 98 | 100% | 85 | 85 | 100% |
| per:city_of_birth | 673 | 673 | 100% | 163 | 163 | 100% |
| per:country_of_birth | 211 | 211 | 100% | 30 | 25 | 83.3% |
| per:country_of_death | 120 | 120 | 100% | 133 | 91 | 68.4% |
| per:city_of_death | 181 | 172 | 95.0% | 38 | 38 | 100% |
| org:top_member_employees | 582 | 533 | 91.6% | 228 | 215 | 94.3% |
| org:founded_by | 1,020 | 921 | 90.3% | 57 | 47 | 82.5% |
| per:parents | 277 | 244 | 88.1% | 114 | 114 | 100% |
| per:spouse | 243 | 211 | 86.8% | 75 | 57 | 76.0% |
| per:cities_of_residence | 242 | 178 | 73.6% | 71 | 9 | 12.7% |
| per:employee_member_of | 2,952 | 1,995 | 67.6% | 609 | 316 | 52.2% |
| org:member_of | 316 | 165 | 52.2% | 36 | 21 | 58.3% |
| org:city_of_headquarters | 781 | 331 | 42.4% | 267 | 104 | 39.0% |
| org:subsidiaries | 232 | 93 | 40.1% | 257 | 53 | 20.6% |
| per:countries_of_residence | 866 | 201 | 23.2% | 617 | 49 | 7.9% |
| org:parents | 231 | 42 | 18.2% | 87 | 2 | 2.3% |
| org:country_of_headquarters | 460 | 68 | 14.8% | 195 | 32 | 16.4% |
| org:shareholders | 41 | 2 | 4.9% | 41 | 0 | 0% |

TABLE 6.9: Comparison of the slot filling responses of KBP-2015 and KBP-2016 datasets in terms of trigger words

Table 6.9 represents the statistics of the number of positive responses and number (with percentage) of responses having trigger word (a content word to justify

the claimed relation) for several slots in KBP-2015 and KBP-2016 datasets. For example, there are 98 responses for *per:children* slot in the KBP-2015 dataset where each of them contains at least one trigger anywhere in the relation justifying snippet. We observe that each of the responses of *per:children* and *per:city_of_birth* slots in both datasets contains at least one trigger. Only 3 (*per:city_of_death*, *per:parents* and *org:member_of*) out of 18 slots in KBP-2016 dataset count higher percentage of positive responses having trigger words compared to the KBP-2015 dataset. However, 13 slots in KBP-2016 dataset count less percentage of responses with trigger words compared to the KBP-2015 dataset. For example, *per:country_of_birth*, *per:spouse* and *per:cities_of_residence* count 100%, 86.8% and 73.6% responses having triggers in KBP-2015 dataset while these numbers are 83.3%, 76% and 12.7% in KBP-2016 dataset respectively. These numbers indicate that KBP-2015 and KBP-2016 datasets differ to each other on the basis of trigger words as well.

We split the slot based on the percentage of responses having trigger words. Almost all the slots in the upper part of Table 6.9 individually counts more than 80% responses with trigger words in both datasets. Each of the slots in the lower part individually counts less than 60% responses with trigger words except *per:cities_of_residence* and *per:employee_member_of* in the KBP-2015 dataset. We can consider these slots in the lower part of Table 6.9 as trigger-independent. This list of trigger-independent slots slightly differs from the trigger-independent slots defined in Table 4.1 in Chapter 4. In Table 6.9, *org:city_of_headquarters* and *org:country_of_headquarters* are considered as trigger-independent while in Table 4.1 they have been considered as trigger-dependent. Moreover, *org:member_of* was included in the group of trigger-dependent relation while here we discover it as a trigger-independent relation. The relations in Table 4.1 were grouped into trigger-dependent and trigger-independent based on the slot filling responses of the KBP-2014 dataset which consisted of a small number of responses compared to KBP-2015 and KBP-2016 datasets. According to the statistics presented in Table 6.9, we assume that it would be better to focus on mostly trigger dependent relations (upper part of this table) for realizing the task of semantic relation validation.

### 6.2.4 Identifying the Reason of Failure to Compute Graph Features

For computing graph features on the two entities, we retrieve the subject and object entities of a relation hypothesis from the constructed association graph based on the given document id and offsets in a slot filling query and response. Since the construction of an association graph depends on named entity recognition (NER),

we do not compute graph features if the association graph does not contain any of the two entities of a claimed relation.

Our system was able to generate graph features for a very small number of responses. We were able to compute graph features for only $1,326$ responses (around $14\%$) out of $9,510$ potential responses. We suspect that redundant text filtering in preprocessing (see Section 5.2.1 on Page 76) and limitation of NER tool restricted to detect some named entities from texts. We investigate around $1,350$ queries of KBP-2016 whether our system can retrieve the subject entities mentioned in these queries.

#### 6.2.4.1 Repeated Text Filtering

In the corpus preprocessing step, we filtered out repeated texts in a document. In some queries, the subject entities are mentioned in the text segments which we filtered out. Therefore, because of filtering redundant text, we failed to detect the subject entities of $78$ queries which is around $5.8\%$ of the total $1,350$ queries.

#### 6.2.4.2 The Ambiguity of Entity Type in Slot Filling Query

There are several queries (e.g. gpe:member_of) where the named entity type of the subject is mentioned as GPE (Geo-Political Entity). GPE usually refers to a location typed entity such as city, state/province and country. However, in a query of GPE typed entity, the sub-type is not mentioned. Therefore, a GPE typed entity becomes ambiguous to identify either it is a country name, state name or city name. There are $181$ queries out of $1,350$ which mention ambiguous GPE typed entities. Thus, we failed to retrieve around $14\%$ query entities.

#### 6.2.4.3 Limitation of the NER Tools

Our fusion NER system was able to detect only $690$ entities as mentioned in $1,350$ queries. That means our NER system was unable to retrieve around $37\%$ of the query entities. We noticed that in several cases, type of the same entity mention differs in the slot filling query and in the association graph. Therefore, retrieving the entity which was mentioned in a query was not possible. By ignoring the type, our system retrieved additional $178$ query entities. Thus, lack of retrieving entities was reduced to around $20.5\%$. Still, this percentage is high but seems reasonable according the performances of existing NER tools as the highest F-score of entity mention extraction on the KBP-2016 corpus (English) is $76.0$ (Ji et al., 2016).

#### 6.2.4.4 Two Related Entities in the Same Sentence

We constrained to find the subject and object entities of a relation in the same sentence for computing graph features. Therefore, we also investigated how many entity pairs of the relation hypotheses have been mentioned in the same sentence. We found that only $1,326$ entity pairs out of $2,467$ were mentioned in the same sentence. That means, our system was not able to find around $46\%$ of the related entity pairs in the same sentences.

#### 6.2.4.5 Summary

Our investigation finds that due to redundant text filtering, ambiguous entity (GPE) type and NER limitation we lack to retrieve in total around $40\%$ query entities. We also noticed that only $2,467$ entity pairs out of $9,510$ slot filling responses were retrieved from our association graph which was around $26\%$. Moreover, due to the constraint of finding the subject and object entities of a relation in the same sentence for computing graph features, we missed around $46\%$ of all the responses. Thus, finally we were able to compute graph features for only around $14\%$ slot filling responses. In order to increase the dataset with graph features, two constraints can be relaxed: (1) relaxing the constraint of retrieving entities from the exact position as mentioned in the queries (2) relaxing the constraint to exist the subject and object entities of a relation in the same sentence.

### 6.2.5 Conclusion and Plans for Improving the System

In our experiments on the KBP-2015 dataset, graph based features have been proven effective to achieve a very good score for validating relations. However, our system lacked to generate graph features for a large number of relation instances. Moreover, slot filler validation system built on some relation validation models with different feature sets did not obtain a good score in the official evaluation of KBP-2016 slot filler validation task. We investigated different issues such as differences between the training and evaluation datasets, trustworthy features, relations with trigger words and lacking responses having graph features etc.

We want to increase the training and test datasets having graph features by relaxing some constraints and to redesign relation validation models with optimized feature sets. Moreover, we want to focus on mostly trigger-dependent relations because they are easy to understand and they are more suitable to evaluate both linguistic and graph features for relation validation task.

## 6.3 Supervised Relation Validation and Knowledge Base Population

Our objective is to employ the relation validation models for discarding wrong relation hypotheses which are generated by several slot filling systems. We aim at populating a knowledge base of trigger-dependent relations based on our supervised relation validation models. In this study, we select 12 relations where 8 are highly trigger-dependent, 2 are softly trigger-dependent and 2 are trigger-independent according to Table 6.9. The selected relations are listed in Table 6.10. We rebuild several relation validation models which are trained and tested on subsets KBP-2015 and KBP-2016 datasets accordingly. Then the best relation validation model is employed to populate a knowledge base.

| | **Dataset KBP-2015** | **Dataset KBP-2016** |
|---|---|---|
| **Relation Name** | **# Responses with Trigger (%)** | **# Responses with Trigger (%)** |
| per:children | 100% | 100% |
| per:city_of_birth | 100% | 100% |
| per:country_of_birth | 100% | 83.3% |
| per:country_of_death | 100% | 68.4% |
| per:city_of_death | 95.0% | 100% |
| org:top_member_employees | 91.6% | 94.3% |
| per:parents | 88.1% | 100% |
| per:spouse | 86.8% | 76.0% |
| per:employee_member_of | 67.6% | 52.2% |
| org:member_of | 52.2% | 58.3% |
| org:city_of_headquarters | 42.4% | 39.0% |
| org:country_of_headquarters | 14.8% | 16.4% |

TABLE 6.10: Selected relations (highly trigger-dependent, softly trigger-dependent and trigger-independent) for knowledge base population

### 6.3.1 Enlarging the Training and Testing Datasets

Since many entity pairs do not have IN_SAME_SENTENCE relationship between them in the association graphs of these datasets, we cannot compute graph features for these entity pairs. Our investigation proposed to relax IN_SAME_SENTENCE constraint between two entities for computing graph features. In this section, we enlarge the training and test datasets by relaxing the IN_SAME_SENTENCE relation constraint.

#### 6.3.1.1 Hard Constraint

Usually, a relation between two entities is expressed when both of the entities are mentioned in the same sentence. Therefore, in our preliminary study, we constrained the system to find an IN_SAME_SENTENCE link (in the association graph of entities) between the subject and object entities of a relationship under observation for computing graph features. Thus we could compute graph features for around $14\%$ of the responses. We obtained $2,274$ ($827$ positive and $1,447$ negative) instances for training from $130$ queries of the $12$ selected relations. In this setting, the test dataset counted $3,429$ ($262$ positive and $1,167$ negative) instances from $63$ queries for the same number of relations. The number of training instances (positive and negative) for different relations are very different. Moreover, some relations (i.e. *per:spouse, org:member_of*) count very small number of training instances and some (i.e. *per:children, country_of_death*) have no training example at all. Therefore, in the setting of hard constraint, we include training examples of some other relations (as shown in Table 4.1) to the instances of the $12$ selected relations. We obtained in total $3,481$ ($1,268$ positive and $2,213$ negative) instances from $260$ queries of $19$ relations. Our experiment on this dataset obtains poor result because of a small number of training examples. Therefore, we defined a strategy for increasing the number of examples.

#### 6.3.1.2 Relaxed Constraint

We relax the constraint of having IN_SAME_SENTENCE relation between the subject and object entities of a relation. If the entity pairs are not connected by an IN_SAME_SENTENCE link in the graph we forcefully connect them by creating the link before computing graph features and delete the link after completing the feature computation of the entity pair. We delete this link after computing the graph features because it does not exist in the original graph. We want to keep the original graph unchanged so that a forcefully created link cannot affect other entity pairs during their graph feature computation. Relaxed constraint significantly increases both training and test instances for all the relations as shown in Table 6.13. We obtain a training dataset that counts in total $14,804$ ($5,933$ positive and $8,871$ negative) instances from $411$ queries of the $12$ selected relations. In a similar way, our test dataset counts $1,109$ and $4,827$ positive and negative instances accordingly from $223$ queries.

| Feature Group | Feature Name |
|---|---|
| Graph | Network similarity between the communities of $E_q$ and $E_c$ of a relation hypothesis, (R, $E_q$, $E_c$) |
| | Network density by merging the communities of $E_q$ and $E_c$ of a relation hypothesis, (R, $E_q$, $E_c$) |
| | Eigenvector centralities of $E_q$ and $E_c$ of a relation hypothesis, (R, $E_q$, $E_c$) |
| | Mutual information between the communities of $E_q$ and $E_c$ of a relation hypothesis, (R, $E_q$, $E_c$) |
| Linguistic | Dependency pattern minimum edit distance |
| | Dependency pattern length |
| | Trigger word between entity mentions |
| | Trigger word in the dependency path between entity mentions |
| | Trigger word in the minimum subtree between entity mentions |
| Trustworthy | Filler Credibility |

TABLE 6.11: Best features for relation validation

### 6.3.2 Relation Validation Models

In this section, we optimize different features to achieve the best results and we observe the effectiveness of adding more training data on relation validation task. Moreover, we measure the performance of different classifiers for binary classification and impact of the proposed features on validating relations.

#### 6.3.2.1 Feature Optimization

In order to select the best features for validating relations, we compiled development training and test datasets from KBP-2015 and KBP-2016 datasets. The development training data consists of $1,407$ instances where the number of positive and negative instances are $605$ and $802$ accordingly. On the other hand, the development test data counts $408$ instances where the number of positive and negative instances are $162$ and $246$ respectively.

| Feature Combination | P | R | F | A |
|---|---|---|---|---|
| Linguistic (all) | 38.5 | 37.0 | 37.7 | 51.5 |
| Linguistic (selected) | 42.5 | 40.1 | 41.3 | 54.7 |
| Trustworthy (all) | 62.7 | 45.7 | 52.9 | 67.6 |
| Filler Credibility | 70.1 | 59.3 | 64.2 | 73.8 |
| Linguistic (selected) + Filler Credibility | 71.0 | 60.5 | 65.3 | 74.5 |
| Linguistic (selected) + Graph | 60.1 | 53.1 | 56.4 | 67.4 |
| Linguistic (selected) + Filler Credibility + Graph | 74.0 | 66.7 | 70.1 | 77.5 |

TABLE 6.12: Evaluation of relation validation features on development data (trained on a small part of KBP-2015 dataset and tested on a small part of KBP-2016 dataset)

| | Hard Constraint | | | | | | Relaxed Constraint | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Relation Name** | **# Train Data** | | **# Test Data** | | **F** | **Acc.** | **# Train Data** | | **# Test Data** | | **F** | **Acc.** |
| | **Pos.** | **Neg.** | **Pos.** | **Neg.** | | | **Pos.** | **Neg.** | **Pos.** | **Neg.** | | |
| per:parents | 35 | 12 | 17 | 15 | 66.7 | 65.6 | 148 | 229 | 94 | 386 | *50.0* | 77.1 |
| per:children | 0 | 0 | 0 | 221 | 0.0 | 99.1 | 67 | 93 | 37 | 630 | **65.9** | 95.7 |
| per:spouse | 2 | 1 | 0 | 0 | - | - | 155 | 298 | 25 | 106 | **49.1** | 79.4 |
| per:country_of_death | 0 | 0 | 6 | 114 | 53.3 | 94.2 | 77 | 148 | 72 | 189 | **88.3** | 93.1 |
| per:country_of_birth | 14 | 28 | 1 | 65 | 8.0 | 65.2 | 108 | 140 | 5 | 260 | **80.0** | 99.3 |
| per:city_of_death | 56 | 100 | 0 | 0 | - | - | 243 | 398 | 30 | 227 | **51.4** | 86.0 |
| per:city_of_birth | 141 | 281 | 70 | 22 | 86.8 | 77.2 | 485 | 814 | 139 | 90 | **91.8** | 90.4 |
| per:employee_or_member_of | 211 | 355 | 16 | 232 | 15.4 | 73.4 | 2,517 | 3,267 | 287 | 1,538 | **50.4** | 80.8 |
| org:top_members_employees | 68 | 78 | 29 | 30 | 96.7 | 96.6 | 461 | 743 | 61 | 277 | *63.4* | 79.9 |
| org:member_of | 8 | 14 | 0 | 0 | - | - | 571 | 917 | 27 | 389 | **58.1** | 91.4 |
| org:country_of_headquarters | 158 | 310 | 82 | 334 | 23.0 | 74.3 | 471 | 822 | 140 | 362 | **54.3** | 78.9 |
| org:city_of_headquarters | 134 | 268 | 41 | 134 | 44.6 | 58.9 | 630 | 1,002 | 192 | 373 | **74.7** | 83.0 |
| **All Together** | 827 | 1,447 | 262 | 1,167 | 51.5 | 78.2 | **5,933** | **8,871** | **1,109** | **4,827** | **63.3** | **84.8** |

TABLE 6.13: Comparison of relation validation performance between hard and relaxed constraints (in both cases, trained on KBP-2015 dataset and tested on KBP-2016 dataset)

We select the best feature set based on an empirical analysis which is listed in Table 6.11. The performances of different features for validating relations as a binary classification task are presented in Table 6.12. We observe that the set of linguistic features used in KBP-2016 SFV task obtains an F-score of 37.7. However, a set of selected linguistic features which removes the indications of relation type (trigger-dependent or trigger-independent) and related entity pairs in the same clause gets a better F-score of 41.3. Moreover, filler credibility as a single trustworthy feature gets an F-score of 64.2 which outperforms the set of four trustworthy features by around 11 points. The combination of selected linguistic features and filler credibility increases the F-score by around 1 point compared to the score of filler credibility. We achieved the best F-score of 70.1 by combining selected linguistic features, filler credibility and graph features. This combination results in the best precision, recall and accuracy as well.

#### 6.3.2.2 Impact of Adding More Training Data

We inspect the efficiency of the relaxed constraint of the IN_SAME_SENTENCE link between subject and object entities over hard constraint to improve the classification performance. We expect that the relation validation system would learn and perform better by training with and testing on more data accordingly.

Table 6.13 represents the statistics of training and test dataset, F-score (F) and accuracy (Acc.) regarding both hard and relaxed constraints. This table shows the scores obtained by Random Forest classifier which is trained by the best feature

combination, i.e. $Linguistic + FillerCredibility + Graph$. Here we use a single word embeddings, GloVe for computing trigger-word based features since it results in better scores compared to other individual embeddings as described in Section 4.5 on Page 65.

Relaxed constraint significantly increases both training and test instances for all the relations as discussed in Section 6.3.1. The F-score and accuracy are gained over almost all the relations by relaxing the IN_SAME_SENTENCE constraint. In the results of the hard constraint dataset, we notice that several relations do not have any training examples (as *per:children* or *per:country_of_death*), or test data (as *per:spouse, per:city_of_death, org:member_of*). We see that relaxed constraint results in better F-score for all the relations except *per:parents* and *org:top_members_employees*. We obtain overall F-score and accuracy of $51.5$ and $78.2$ accordingly by hard constraint. In contrast, the relaxed constraint improves these scores by around $12$ and $6$ points accordingly. We achieve overall F-score and accuracy of $63.3$ and $84.8$ accordingly by relaxing the constraint. Thus relaxation of the constraint of IN_SAME_SENTENCE relation between subject and object entities facilitates to train a model with more data that significantly improves the performance to classify the correct and wrong relations.

We also observe the impact of multiple word embeddings for relation validation task on the relaxed constraint datasets of $12$ relations together. We use the best combination as described in Section 4.5 on Page 65 and include other features such as Filler Credibility, graph and syntactic dependency path length, dependency patterns and dependency pattern edit distance. A Random Forest classifier trained on KBP-2015 dataset and tested on KBP-2016 dataset results the precision, recall, F-score and accuracy of 54.5, 71.3, 61.8 and 83.3 accordingly. By employing multiple word embeddings we get around 1.2 points higher recall but the precision, F-score and accuracy are decreased by around 3, 1.5 and 1.5 points accordingly compared to the results presented in the relaxed constraint part of Table 6.13. Multiple word embeddings improve the recall by identifying trigger words of expressing relations but fails to achieve a better precision. As a consequence, it gets lower F-score and accuracy. However, in Section 4.5, we obtained better score by combining multiple word embeddings. The results would need further exploration to know why multiple word-embeddings provide a different behavior. Since trigger word features computed on GloVe and other features collectively gives better F-score, in the next experiments, we use only GloVe instead of multiple word embeddings.

| Classifier | Precision | Recall | F-measure | Accuracy |
|---|---|---|---|---|
| LibLinear | 45.6 | 73.9 | 56.1 | 79.1 |
| SVM | 49.8 | 73.5 | 59.4 | 81.6 |
| Vowpal | 48.4 | 66.7 | 56.8 | 81.2 |
| Naive Bayes | 48.0 | **76.5** | 59.0 | 80.6 |
| MaxEnt | 48.3 | 69.5 | 57.0 | 80.6 |
| Random Forest | **57.8** | 70.1 | **63.3** | **84.8** |

TABLE 6.14: Relation validation performances by different classifiers on relaxed constraint dataset (trained on KBP-2015 dataset and tested on KBP-2016 dataset)

#### 6.3.2.3 Performance Measure of Different Classifiers

We compare the classification performances of different classifiers e.g. LibLinear, SVM, Vowpal, Naive Bayes, MaxEnt and Random Forest based on the best feature combination (FillerCredibility+Linguistic+Graph) on the dataset of relaxed constraint (KBP-2015 dataset for training and KBP-2016 dataset for testing) as shown in Table 6.14. We achieve the best precision (57.8), F-score (63.3) and accuracy (84.8) by Random Forest classifier although it gets a lower recall (70.1) compared to other classifiers. The best recall of 76.5 is resulted by Naive Bayes which obtains the third highest F-score (59.0) and accuracy (80.6). The second highest precision (49.8), F-score (59.4) and accuracy (81.6) are resulted by SVM. In contrast, the second highest recall (73.9) is achieved by LibLiner which gets the lowest precision, F-score and accuracy.

#### 6.3.2.4 Impact of the Proposed Features on Relation Extraction

The tasks of relation validation and relation extraction are considered as the opposite side of the same coin. We evaluate the proposed features in terms of relation extraction. It may not be system-independent to use the trustworthy Filler Credibility feature to realize a task of relation extraction. Usually, a relation extractor does not employ multiple systems for generating relation hypothesis. Therefore, we discard the trustworthy Filler Credibility feature in evaluating relation validation to realize the contribution of proposed features for relation extraction. We define a baseline (BL) by four linguistic features as listed in Table 6.15. These are the most basic and commonly used features in relation extraction task. We compute the 3 features of inspecting trigger words by using GloVe embeddings. We observe relation validation performances through this linguistic baseline, the proposed linguistic (dependency

| |
|---|
| 1. Trigger word between entity mentions |
| 2. Trigger word in the dependency path between entity mentions |
| 3. Trigger word in the minimum subtree between entity mentions |
| 4. Dependency pattern length |

TABLE 6.15: Baseline linguistic features for evaluating relation validation models

pattern edit distance) and graph features and their combinations. Since Random Forest results in the best score over several classifiers, we observe the performances of different feature sets by this classifier.

| | **BL** | | | **BL + DPED** | | | **BL + Graph** | | | **BL + DPED + Graph** | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Relation Name** | **P** | **R** | **F** | **P** | **R** | **F** | **P** | **R** | **F** | **P** | **R** | **F** |
| per:parents | 37.30 | 73.40 | 49.46 | 40.12 | 69.15 | 50.78 | 51.75 | 78.72 | **62.45** | 45.65 | 67.02 | 54.31 |
| per:children | 62.22 | 75.68 | 68.29 | 60.87 | 75.68 | 67.47 | 70.00 | 75.68 | 72.73 | 93.33 | 75.68 | **83.58** |
| per:spouse | 36.23 | 100 | 53.19 | 73.53 | 100 | 84.75 | 65.00 | 52.00 | 57.78 | 68.42 | 52.00 | **59.09** |
| per:country_of_death | 98.55 | 94.44 | **96.45** | 98.55 | 94.44 | **96.45** | 97.50 | 54.17 | 69.64 | 98.55 | 94.44 | **96.45** |
| per:country_of_birth | 11.43 | 80.00 | 20.00 | 12.90 | 80.00 | 22.22 | 100 | 80.00 | **88.89** | 100 | 80.00 | **88.89** |
| per:city_of_death | 58.00 | 96.67 | 72.50 | 71.05 | 90.00 | 79.41 | 75.00 | 90.00 | **81.82** | 75.00 | 90.00 | **81.82** |
| per:city_of_birth | 97.20 | 100 | 98.58 | 97.18 | 99.28 | 98.22 | 100 | 99.28 | **99.64** | 100 | 99.28 | **99.64** |
| per:employee_or_member_of | 20.74 | 29.27 | 24.28 | 20.63 | 27.53 | 23.58 | 34.85 | 24.04 | 28.45 | 32.88 | 25.09 | **28.46** |
| org:top_members_employees | 39.39 | 63.93 | 48.75 | 35.78 | 63.93 | 45.88 | 52.38 | 90.16 | 66.27 | 59.14 | 90.16 | **71.43** |
| org:member_of | 28.57 | 44.44 | 34.78 | 38.71 | 44.44 | 41.38 | 48.00 | 44.44 | 46.15 | 50.00 | 44.44 | **47.06** |
| org:country_of_headquarters | 52.17 | 25.71 | 34.45 | 59.02 | 25.71 | 35.82 | 75.93 | 29.29 | **42.27** | 75.93 | 29.29 | **42.27** |
| org:city_of_headquarters | 50.00 | 44.27 | 46.96 | 60.14 | 43.23 | 50.30 | 86.67 | 47.4 | **61.28** | 89.69 | 45.31 | 60.21 |
| All Together | 44.75 | **55.73** | 49.64 | 48.55 | 54.46 | 51.34 | 65.09 | 53.29 | 58.60 | **66.02** | 54.82 | **59.90** |

TABLE 6.16: Classification performances by different feature sets (trained on KBP-2015 dataset and tested on KBP-2016 dataset)

Table 6.16 presents the classification scores where we observe that the combination of BL and proposed graph features outperforms the BL almost for all the relations except *per:country_of_death*. We obtain overall F-score of 58.60 by BL+Graph that is around 9 points higher than the BL. The experimental results also show that the combination of BL and dependency pattern edit distance (DPED) improves the overall F-score by 1.79 points over the BL. This combination achieves higher F-score for 7 relations (among 12) which indicates the effectiveness of DPED for relation validation task. Basically, we gain higher precision by allowing a slight drop of recall that results in better F-score over the BL. The best F-score is achieved by the combination of BL, DPED and graph (BL+DPED+Graph). This combination results overall F-score of 59.90 which is around 10 points higher than the BL. We observe that BL+DPED+Graph obtains higher F-score for 11 relations compare to the BL.

| | BL | | | | | BL + DPED + Graph | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Relation Name** | **TP** | **FN** | **FP** | **TN** | **Acc.** | **TP** | **FN** | **FP** | **TN** | **Acc.** |
| per:spouse | 25 | 0 | 44 | 62 | 66.41 | 13 | 12 | 6 | 100 | **86.28** |
| per:parents | 69 | 25 | 116 | 270 | 70.62 | 63 | 31 | 75 | 311 | **77.92** |
| per:children | 28 | 9 | 17 | 613 | 96.10 | 28 | 9 | 2 | 628 | **98.35** |
| per:country_of_death | 68 | 4 | 1 | 188 | 98.08 | 68 | 4 | 1 | 188 | 98.08 |
| per:country_of_birth | 4 | 1 | 31 | 229 | 87.92 | 4 | 1 | 0 | 260 | **99.62** |
| per:city_of_death | 29 | 1 | 21 | 206 | 91.44 | 27 | 3 | 9 | 218 | **95.33** |
| per:city_of_birth | 139 | 0 | 4 | 86 | 98.25 | 138 | 1 | 0 | 90 | **99.56** |
| org:top_members_employees | 39 | 22 | 60 | 217 | 75.74 | 55 | 6 | 38 | 239 | **86.98** |
| org:member_of | 12 | 15 | 30 | 359 | 89.18 | 12 | 15 | 12 | 377 | **93.51** |
| org:country_of_headquarters | 36 | 104 | 33 | 329 | 72.71 | 41 | 99 | 13 | 349 | **77.69** |
| org:city_of_headquarters | 85 | 107 | 85 | 288 | 66.02 | 87 | 105 | 10 | 363 | **79.65** |
| per:employee_or_member_of | 84 | 203 | 321 | 1217 | 71.29 | 72 | 215 | 147 | 1391 | **80.10** |
| All Together | 618 | 491 | 763 | 4,064 | 78.87 | 608 | 501 | 313 | 4,514 | **86.29** |

TABLE 6.17: Comparison of the confusion matrices resulted by BL and BL+DPED+Graph (trained on KBP-2015 dataset and tested on KBP-2016 dataset)

For only one relation (*per:country_of_death*) the classification performance remains same as the BL.

We notice in Table 6.16 that BL+Graph and BL+DPED+Graph obtain a very high score for *per:country_of_birth* over the BL. Both BL+Graph and BL+DPED+Graph achieve an F-score of $88.89$ which is around $69$ points higher than the BL. The reason behind this result is that we have a very small number of true instances (total 5) for this relation compare to the number of false instances (total 260) (as shown in Table 6.17) and a high precision is resulted by discarding $231$ false relations.

We achieve the highest precision almost for all the relations by BL+DPED+Graph. BL+DPED+Graph achieves an overall precision of $66.02$ that is around $21$ points higher than the BL that indicates the proposed features discard a large number of false relation instances correctly. A little drop of recall is caused by BL+DPED+Graph which is around $1$ point lower than the BL. The recall of $55.73$ and $54.82$ are resulted by the BL and BL+DPED+Graph accordingly. The drop of recall indicates the limitations of graph features to hold the semantic evidence of some relations.

Table 6.17 illustrates the confusion matrix resulted by BL and BL+DPED+Graph where we compare the number of true positive (TP), false negative (FN), false positive (FP), true negative (TN) and accuracy (Acc.). We see that the baseline and BL+DPED+Graph methods correctly classify overall $618$ and $608$ true relation instances accordingly among $1,109$. That means BL+DPED+Graph discards $501$ true relation instances which is around $1\%$ more than the BL. However, the BL and BL+DPED+Graph correctly discard overall $4,064$ and $4,514$ false relation instances respectively among $4,827$. The rate of discarding false relation instances by BL+DPED+Graph

| Claimed Relation | Justification Sentence | RV |
|---|---|---|
| spouse(Willem-Alexander, Maxima Zorreguieta Cerruti) | **Willem-Alexander** married **Maxima Zorreguieta Cerruti** from Argentina and they have three daughters: Princess Catharina-Amalia, Princess Alexia and Princess Ariane. | TP |
| children(Margaret Thatcher, Mark) | In a statement to the public, **Thatcher**'s son **Mark** Thatcher said his twin sister Carol and the rest of their family had been overwhelmed by messages of support they had received from around the globe. | TP |
| spouse(Willem-Alexander, Alexia) | **Willem-Alexander** married Maxima Zorreguieta Cerruti from Argentina and they have three daughters: Princess Catharina-Amalia, Princess **Alexia** and Princess Ariane. | TN |
| children(Margaret Thatcher, Carol) | In a statement to the public, **Thatcher**'s son Mark Thatcher said his twin sister **Carol** and the rest of their family had been overwhelmed by messages of support they had received from around the globe. | FN |

TABLE 6.18: True positive (TP), true negative (TN) and false negative (FN) examples after validating relations

is around $9\%$ higher than the BL which contributes to increasing the overall precision and finally achieves a high accuracy. The BL model achieves the overall accuracy of $78.87\%$ while BL+DPED+Graph achieves around $7.42\%$ higher accuracy which is $86.29\%$. While observing the accuracy relation-by-relation we see a significant improvement achieved by BL+DPED+Graph over the BL for all the relations.

#### 6.3.2.5 Error Analysis

We inspect some relation justifying snippets for error analysis. Table 6.18 presents classification results on some claimed relations from the test data that helps to realize the performance of our RV model. The first and second rows show two correctly classified true claims of *spouse* and *children* relation accordingly. Furthermore, a false claim of *spouse* relation has been detected as wrong as shown in the third row. In contrast, our system fails to correctly classify a true *children* relation as shown in the fourth row. However, our system achieves overall decent scores compared to the baseline. All the experimental results on relation validation task show that global information about the entities captured by the community-graph based features are significantly effective for relation extraction task.

### 6.3.3 Knowledge Base Population by Employing Relation Validation Models

Our objective is to populate a knowledge base by using the outputs of different slot filling systems. Since slot filling systems generate a lot of wrong relation hypotheses, we want to discard the false relations without affecting the true ones. Therefore, we are interested to know whether the validation step allows for building a better knowledge base. Here we employ our relation validation models for KBP task. We use the slot filling model presented in Section 3 on Page 72 regarding this task.

| | **Precision** | **Recall** | **F-score** |
|---|---|---|---|
| Top 3 TAC KBP systems (official score) | | | |
| System-1 | 36.69 | 20.51 | 26.31 |
| System-2 | 25.25 | 18.69 | 21.53 |
| System-3 | 28.80 | 13.15 | 18.06 |
| Top 3 TAC KBP systems (for the selected relations) | | | |
| System-1 | 36.73 | 22.78 | 28.12 |
| System-2 | 32.07 | 24.89 | 28.03 |
| System-3 | 37.50 | 21.52 | 27.35 |
| KBP by the Relation Validation models (for the selected relations) | | | |
| Filler Credibility + Linguistic + Graph | **38.51** | **24.05** | **29.61** |
| Linguistic + Graph | 29.53 | 18.57 | 22.80 |
| Filler Credibility | 24.88 | 21.10 | 22.83 |

TABLE 6.19: KBP performances of different systems and relation validation models on KBP-2016 dataset

For evaluating the KBP task, TAC defines a set of *ground truths* for all the query slots that contains different correct filler values for each of the queries. A filler value (object) is considered as correct if the excerpt containing the subject and object justifies their relation, otherwise wrong. The KBP system should not repeat an object for the same slot. If the system repeats an object for the same slot only one instance of that object would be considered as correct and others would be wrong.

We measure the KBP performances of different systems on our test dataset for comparison. Here, the results do not indicate the official KBP scores. Since we focus on some selected relations, the results in this study differ from the official scores even though we follow similar scoring method. The upper part of Table 6.19 shows the official scores of the top 3 TAC KBP systems where the best F-score of 26.31 is achieved by System-1. On the other hand, System-2 and System-3 obtain the F-score of 21.53 and 18.06 accordingly. However, these three systems obtain the F-scores of 28.12, 28.03 and 27.35 accordingly on our test dataset for the selected relations (see System-i in the middle part of Table 6.19).

Since different relation extraction systems can be employed for KBP task, we can use the trustworthy filler credibility feature to take advantage of the agreements on the outcomes by several relation extraction systems. Therefore, we built a relation validation model by using a single trustworthy feature. Since the best performance of validating relations is achieved by $Filler\,Credibility + Linguistic + Graph$

features, we use the relation validation model trained by this feature combination for KBP task.

In the lower part of Table 6.19, we see the voting (Filler Credibility) based KBP system obtains an F-score of 22.83 that indicates the importance of this feature. Interestingly, the $Filler\,Credibility + Linguistic + Graph$ based KBP system achieves an F-score of 29.61 which is higher than each individual KBP system. We also observe that $Filler\,Credibility + Linguistic + Graph$ based KBP system achieves the highest precision of 38.51 that is almost 2 points higher than the best KBP system. The precision improvement indicates that this model discards many wrong relations which are resulted by different relation extraction systems. Moreover, $Filler\,Credibility + Linguistic + Graph$ based KBP system obtains the recall of 24.05 that is around 1.27 point higher than the best relation extraction based KBP system and around 3 points higher than the voting based KBP system. These results justify that our system enables to fill more relations in knowledge base than the existing ones for the trigger-dependent relations.

## 6.4 An Experiment of Unsupervised Relation Validation and Knowledge Base Population

In some cases, unsupervised learning becomes very important, specially if no annotated data is available. We aim at exploring graph algorithm to exploit our graph representation for relation validation and KBP tasks in an unsupervised fashion.

We hypothesize that the graph of entities gives some clues on how two related entities are connected to each other via their common neighbors. A graph of entities facilitates to find such clues. Moreover, it helps to rank the entities by employing a ranking algorithm. We assume that for a query relation *(R, $E_q$, ?)*, a correct candidate-object will get higher ranking score than the score of a wrong one. If multiple correct objects are allowed for a query relation, we expect that the correct ones will get comparatively higher ranks than the wrong ones. Therefore, a threshold of rank has to be defined to choose the correct objects of a query relation.

We use a graph based candidate ranking model as shown in Fig. 6.1. This method takes the subject ($E_q$) and candidate entities ($E_{c1}$, $E_{c2}$, ..., $E_{cn}$) of a relation hypothesis as the input. Then it models a graph of entities with all the relation hypotheses and ranks the candidates. Finally, top-ranked candidates which satisfy a predefined threshold ($th$) are considered as the correct objects of that relation.

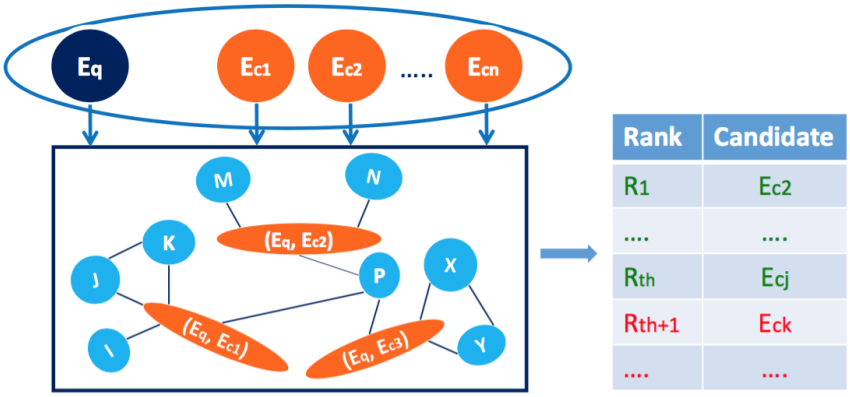


FIGURE 6.1: Unsupervised candidate ranking model

### 6.4.1 PageRank Algorithm

PageRank (Brin and Page, 1998) is a widely used algorithm for ranking graph nodes in an unsupervised fashion. The PageRank score of a node, $u$ in a graph can be computed recursively by Eq. 6.1. However, while computing PageRank, a damping factor, $d$ is used to avoid sink (i.e. nodes with no outgoing links). After adding the damping factor, the PageRank formula is generalized by Eq. 6.2. If the value of $d$ is 1, the PageRank score of a node always becomes 0. In common practice, the value of $d$ is 0.85. Sometimes, weights are given to graph nodes. By considering weights of the nodes, the original PageRank formula can be transformed into Eq. 6.3.

$$PR(u) = \sum_{v \epsilon N_u} \frac{PR(v)}{L(v)} \tag{6.1}$$

*where, $N_u$ is the list of nodes connected to $u$ and $L(v)$ is the number of links from an node $v$*

$$PR(u) = \frac{1-d}{N} + d \sum_{v \epsilon N_u} \frac{PR(v)}{L(v)} \tag{6.2}$$

*where, $N$ is the total number of nodes in a graph*

$$PR(u) = \frac{1-d}{N} + d \sum_{v \epsilon N_u} \frac{PR(v)W(v)}{L(v)} \tag{6.3}$$

*where, $W(v)$ is the weight given to node, $v$*

### 6.4.2 Graph Modeling

Our objective is to rank the candidate objects of a query relation by using PageRank algorithm. Such method has been successfully used in entity linking task (Pershina, He, and Grishman, 2015). We construct a special graph of entities for the ranking task.

We represent each candidate object as a relation hypothesis with the subject entity as shown in Fig. 6.2. For example, *($E_q$, $E_ci$)* refers to a claimed relation hypothesis where $E_q$ and $E_ci$ indicate the subject and a candidate respectively. The other entities linked to a relation hypothesis denote the community members of the corresponding candidate. For instance, $M$ and $N$ are the community members of candidate $E_c2$. We make some modifications in this graph which are reflected in Fig. 6.3. For example, any entity (e.g. $P$) which is not a community member of any candidate object but connected to the subject entity, is linked to all the candidate entities. In Fig. 6.3, $P$ is connected to the three candidates. Moreover, if there is a link between two candidate objects, the link is removed to avoid the influence of a candidate to another one. Finally, the subject entity is removed from the graph as Fig. 6.3 shows that it does not contain the subject entity and its links to the candidate entities. We finalize this graph by expanding the community of each candidate up to level 3.

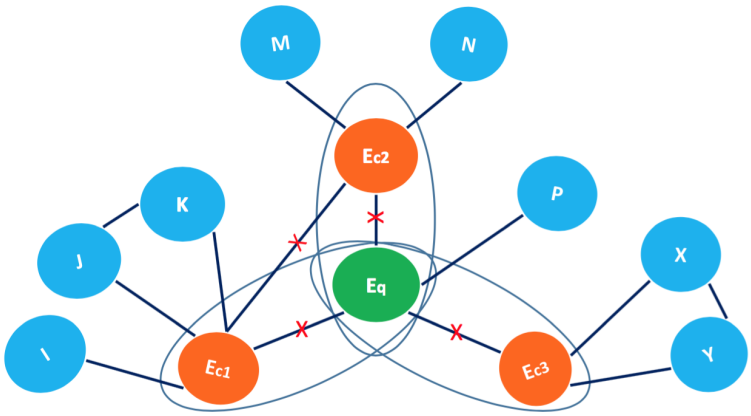


FIGURE 6.2: Entity graph with relation hypotheses

Since this graph does not hold any linguistic semantic for characterizing the relation type, we assign weights to the candidate objects based on their linguistic confidence scores. The linguistic confidence of a candidate refers to the trigger word similarity score (as described in Section 4.3.3 on Page 62) computed on the sentence which is provided for justifying relation between the subject and the candidate object.

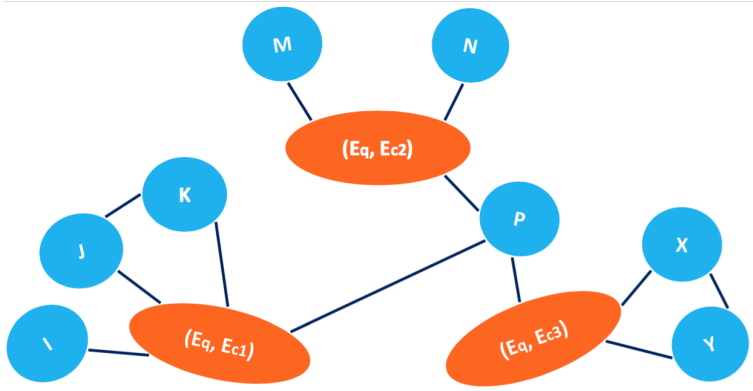


FIGURE 6.3: Modified graph for ranking the candidate entities by PageRank

We take the maximum score among the three options: trigger word between entity mentions, trigger word in the dependency path and trigger word in the minimum subtree (Section 4.4 on Page 63).

$$W(E_c i) = \frac{Sc_i}{\sum_{i=1}^{n} Sc_i} \tag{6.4}$$

The weights of the candidate nodes are normalized. For example, $E_c = \{E_{c1}, E_{c2}, ..., E_{cn}\}$ is the set of candidate objects of a relation and $S = \{Sc_1, Sc_2, ..., Sc_n\}$ refers to the set of linguistic confidence scores of the corresponding candidates. We define the weight, $W(E_c i)$ of a candidate, $E_c i$ by Eq. 6.4. Moreover, a small weight of $0.01$ is assigned to each of the non-candidate nodes in the graph in order to take them into account for scoring the candidates.

### 6.4.3 Evaluation

We evaluate the unsupervised relation validation model on the responses of $105$ queries of trigger-dependent relations from the KBP-2016 dataset. We select these queries because each of the queries has been responded by at least one correct and one wrong objects. If all the responses of a query are wrong, it does not make any sense to rank them.

In total there are $1,223$ correct and $3,864$ wrong responses to these queries by different systems. Firstly, we compute accuracy at $N = \{1, 2, 3, 4, 5\}$ when the total number of responses to a query is greater than $N$. For example, $A@1$ means the percentage of queries that get at least one correct response at the top ranked position when total number of responses is at least $2$. Similarly, $A@5$ counts the percentage of queries get at least one correct response anywhere in the top $5$ positions when

| N | # Query | # Correct | A@N |
|---|---|---|---|
| 1 | 105 | 25 | 23.81 |
| 2 | 104 | 48 | 46.15 |
| 3 | 102 | 59 | 57.84 |
| 4 | 102 | 68 | 66.67 |
| 5 | 100 | 78 | 78.00 |

TABLE 6.20: Accuracy@N (on KBP-2016 dataset) by ranking the candidate objects of a relation

| Ranking Threshold | TP | FN | FP | TN | P | R | F |
|---|---|---|---|---|---|---|---|
| 1 | 271 | 952 | 321 | 3,543 | 45.78 | 22.16 | 29.86 |
| 2 | 467 | 756 | 543 | 3,321 | **46.24** | 38.18 | 41.83 |
| 3 | 532 | 691 | 831 | 3,033 | 39.03 | 43.50 | 41.14 |
| 4 | 709 | 514 | 1,027 | 2,837 | 40.84 | 57.97 | 47.92 |
| 5 | 778 | 445 | 1,217 | 2,647 | 39.00 | **63.61** | **48.35** |

TABLE 6.21: Confusion matrix and evaluation scores (on KBP-2016 dataset) after filtering lower ranked candidate objects (TP = true positive, FN = false negative, FP = false positive, TN = true negative, P = precision, R = recall and F = F-score)

the total number of responses is at least 6. Table 6.20 shows the $A@N$ results. We observe that only 25 queries out of 105 get the correct response at the top ranked position. Thus the score of $A@1$ is 23.81. Similarly, the scores of $A@2$, $A@3$ and $A@4$ are 46.15, 57.84 and 66.67 accordingly. We get the best accuracy at $A@5$ which is 78.00.

We define different ranking thresholds ranging 1 to 5 (higher to lower) for selecting the correct responses. That means, for ranking threshold 5, if any candidate object gets a rank lower than 5, it would be considered as wrong. Then we observe the confusion matrix (true positive, false negative, false positive and true negative) and evaluation scores (precision, recall, F-score and accuracy) after filtering the lower ranked candidates in Table 6.21.

We want to discard a large number of false relation by keeping the correct ones. Therefore, we are interested in F-score because it makes a trade-off between the precision and recall. We notice that this ranking based relation validation method gets the best recall (63.81) and F-score (48.35) for a ranking threshold 5. This threshold results in a precision of 39.00. The ranking threshold of 2 gives the best precision of 46.24 but obtains F-score of 41.83 which is around 7 points lower than the best one.

| Relation Name | # Total Instances | # Positive | # Negative | TP | FN | FP | TN | P | R | F |
|---|---|---|---|---|---|---|---|---|---|---|
| per:spouse | 64 | 26 | 38 | 26 | 0 | 25 | 13 | 50.98 | 100 | 67.53 |
| per:parents | 365 | 94 | 271 | 38 | 56 | 110 | 161 | 25.68 | 40.43 | 31.40 |
| per:children | 514 | 38 | 476 | 14 | 24 | 89 | 387 | 13.59 | 36.84 | 19.86 |
| per:city_of_birth | 215 | 131 | 84 | 131 | 0 | 32 | 52 | 80.37 | 100 | 89.12 |
| per:country_of_birth | 112 | 5 | 107 | 5 | 0 | 42 | 65 | 10.64 | 100 | 19.23 |
| per:city_of_death | 120 | 31 | 89 | 30 | 1 | 36 | 53 | 45.45 | 96.77 | 61.86 |
| per:country_of_death | 176 | 76 | 100 | 58 | 18 | 37 | 63 | 61.05 | 76.32 | 67.84 |
| org:top_members_employees | 323 | 67 | 256 | 53 | 14 | 100 | 156 | 34.64 | 79.10 | 48.18 |
| org:member_of | 237 | 33 | 204 | 19 | 14 | 53 | 151 | 26.39 | 57.58 | 36.19 |
| org:country_of_headquarters | 619 | 150 | 469 | 56 | 94 | 196 | 273 | 22.22 | 37.33 | 27.86 |
| org:city_of_headquarters | 650 | 223 | 427 | 147 | 76 | 121 | 306 | 54.85 | 65.92 | 59.88 |
| per:employee_or_member_of | 1,692 | 349 | 1,343 | 201 | 148 | 376 | 967 | 34.84 | 57.59 | 43.41 |
| **All Together** | **5,087** | **1,223** | **3,864** | **778** | **445** | **1,217** | **2,647** | **39.00** | **63.61** | **48.35** |

TABLE 6.22: Evaluation scores relation-by-relation (on KBP-2016 dataset) after filtering lower ranked candidate objects for the ranking threshold of 5 (TP = true positive, FN = false negative, FP = false positive, TN = true negative, P = precision, R = recall and F = F-score)

The ranking threshold of $4$ obtains F-score of $47.92$ which is almost near to the best one. This threshold results in the precision and recall of $40.84$ and $57.98$ accordingly. On the other hand, the ranking threshold of $1$ and $3$ obtain the F-score of $29.86$ and $41.14$ accordingly.

We compare the confusion matrix resulted by the ranking threshold of $4$ and $5$ since they achieve similar and higher F-scores. The ranking threshold of $5$ discards $2,647$ wrong relations (out of $3,865$) by keeping $778$ correct relations (out of $1,223$) successfully. In contrast, the ranking threshold of $4$ discards $2,837$ wrong relations which counts $190$ more instances than that counted by a threshold of $5$. However, this threshold preserves $709$ correct relations which counts $69$ less instances than that resulted by the threshold of $5$.

In Table 6.22 we present the evaluation scores relation by relation for the ranking threshold of $5$. In some relations, per:spouse, per:city_of_birth and per:country_of_birth, many wrong relations ($13$, $52$ and $65$ accordingly) have been discarded by preserving all the correct ones. For these relations, resulted F-scores are $67.53$,$89.12$ and $19.23$ respectively. Moreover, in per:city_of_birth relation, $53$ wrong relations (out of $89$) have been removed where only one correct relation has been classified as wrong and it obtains the F-score of $61.86$. Also in per:city_of_birth relation, many wrong relations have been discarded by losing some correct ones which obtain the F-score of $67.84$. In per:country_of_birth relation, even though all the correct relations have been preserved, still the F-score is very low. The reason is that this relation counts very few positive examples ($5$) compared to the number of negative examples ($107$) and $42$ of the negative examples have been considered as correct.

| Ranking Threshold | Precision | Recall | F-score |
|---|---|---|---|
| 1 | 18.60 | 06.75 | 09.91 |
| 2 | **20.81** | 15.91 | 17.56 |
| 3 | 15.53 | 17.30 | 16.37 |
| 4 | 17.07 | 24.05 | 19.97 |
| 5 | 16.59 | **28.69** | **21.02** |
| KBP with the selected relations | | | |
| System-1 | 36.73 | 22.78 | 28.12 |
| System-2 | 32.07 | 24.89 | 28.03 |
| System-3 | 37.50 | 21.52 | 27.35 |

TABLE 6.23: KBP performance by the ranking based relation validation model (upper part) and top 3 individual KBP systems (lower part) on KBP-2016 dataset

However, for other relations (e.g. org:member_of, org:city_of_headquarters etc.), many wrong relations have been discarded but a large number of correct relations has not been preserved. As a result, they get very low F-scores. From these results, we observe that even though this unsupervised ranking based relation validation model discards a large number of wrong relations, it fails to preserve many correct ones.

Now, we construct a knowledge base (for the selected 12 relations) in an ensemble fashion by using the relations which have been classified as correct by the ranking model. The upper part of Table 6.23 shows the KBP scores resulted for different ranking thresholds. We see that the threshold of 5 achieves the best recall (28.69) and F-score (21.02). The best precision is obtained by the threshold of 2 which results around 3.5 points lower F-score that the best one. We notice that the increasing of ranking threshold results in better recall and F-score but causes to drop of precision. We have noticed similar characteristics in the evaluation of relation validation task (see Table 6.21). In both tasks of relation validation and KBP, the ranking threshold of 5 achieves the best score compared to other threshold values. Although the ranking based ensemble system obtains the F-score of 21.02 (for ranking threshold of 5), still it gets a lower score than the top ranked KBP systems. The top three individual KBP systems obtain the F-score of 28.12, 28.03 and 27.35 (in decreasing order) accordingly (see the lower part of Table 6.23).

However, in our experiment of ranking based relation validation, we used the same threshold value for ranking the candidates of different relations. Different thresholds can be studied regarding different types of relations to observe whether

they can improve the relation validation performance.

## 6.5 Summary

We explored both supervised and unsupervised methods of relation validation task. Our experimental results show that the supervised model gets better score compared to the unsupervised model. Moreover, the supervised relation validation model contributes to populate a better knowledge base in an ensemble manner that outperforms the top ranked individual KBP systems. However, the ranking based unsupervised relation validation model gets a lower score than the individual KBP systems.

We used different types of features in our supervised relation validation model. Validating a semantic relation at the sentence level mostly depends on linguistic evidence such as the existence of trigger word between entity mention, in the syntactic dependency path, in the minimal subtree, length the syntactic dependency path, similarity of dependency patterns, clausal information, relation type regarding trigger dependency etc.

Trustworthy features such as confidence score, credibilities of the systems, documents and responded filler values collectively obtains a very good score specially when a classifier model trained and tested on the same year dataset (KBP-2015) and the relation hypotheses are generated by different systems. However, a classifier trained and tested on KBP-2015 and KBP-2016 datasets accordingly, does not result in good score by using confidence score and credibilities of the systems and documents as the features of relation validation. Our investigation finds that the KBP-2015 and KBP-2016 evaluation datasets differ significantly. Therefore, some the trustworthy features affect the relation validation performance badly. Moreover, our feature optimization method on a development dataset filters out two linguistic features which are clausal information and relation type regarding trigger dependency.

Trustworthy Filler Credibility as a voting feature obtains a good score for validating relations even though the relation validation model is trained and tested on different datasets. However, in order to be system independent, a relation validation method discards trustworthy filler credibility feature and relies on only linguistic features which are not sufficient to achieve a decent score. Thus, validating entity level relation requires global information at the entity level in addition to the linguistic evidence at the sentence level.

Global information about entities provides some clues how two related entities are associated in a collection when the entities in the collection are represented in a graph. A graph structure facilitates analyzing the communities of different entities. Moreover, it helps to compute several features such as network density, network

similarity, mutual information, the influence of a node on another one etc. on the community graphs for validating a relation hypothesis between two entities. Construction of a graph of entities requires NER and computing graph features requires having the IN_SAME_SENTENCE link between two entities under inspection. Our system was unable to generate graph features for a large number of entity pairs. Our investigation suggested relaxing some constraints to increase the dataset with graph features. By relaxing these constraints we significantly increase the training and test datasets with graph features.

Experimental results show that proposed graph features significantly contribute to outperform the baseline model of relation validation when they are combined with some baseline features. Furthermore, a relation validation model with the proposed graph features achieves a better score over the best individual KBP system for some trigger-dependent relations. All the experimental results signify the effectiveness of the proposed graph features in both tasks of relation validation and knowledge base population.

# Chapter 7

# Conclusion and Future Work

## 7.1 Conclusion

In the current age of World Wide Web (WWW), information is increasing very rapidly in text format which is readable by both human and machine. However, the huge source of information is mostly unstructured. Collecting important and useful information from the unstructured texts is a challenging and interesting task in NLP research. Knowledge Base Population is a special kind of information extraction task that aims at creating a collection of facts of the world. A Knowledge Base facilitates a user to retrieve necessary information on demand very easily. It also helps both human and machine to learn reasoning among the related facts. Manually populating a knowledge base from a large volume of unstructured information sources is a very difficult and time consuming task. Therefore, automatic knowledge base population comes into play. Automatic KBP task requires extraction of semantic relations from texts. Semantic relations can be classified into two distinct types: trigger-dependent and trigger-independent. In order to express a trigger-dependent relation between two entities in a sentence, there must exist a trigger word between or around the entities. In contrast, the existence of trigger word is not mandatory for expressing a trigger-independent relation. Existing relation extraction systems generate a large number of false relations that lead to a noisy and inaccurate knowledge base. Therefore, filtering false relations gets importance to construct a more accurate knowledge base.

In this research, we focused on validating trigger-dependent relation hypotheses which have been generated by different systems. We considered relation validation as a binary classification task. Several classifier models were trained by different combinations of features to classify a claimed relation as true or false.

We observed that there are some similarities between the tasks of relation validation and relation extraction specially in terms of linguistic analysis. Both the tasks inspect existence of trigger words between or around a pair of entity mentions. Therefore, we inherited a couple of linguistic features from the literature of relation

extraction. We analyzed the lexicons, syntactic dependency path and dependency parse tree of the relation justifying text of a claimed relation to validate it. The presence of trigger words had been inspected between the pair of entity mentions, in the shortest dependency path and in the minimum sub-tree. We also took into account length of the shortest dependency path between two entities.

In our primary study, we collected some trigger words of different relations from a small volume of labeled data. Some relations are expressed by a large number of trigger words that cannot be captured from a small dataset. Therefore, we employed a word embeddings to identify the unknown trigger words of a relation. Basically, we computed cosine similarity between the vectors of a known trigger and a word between a pair of entity mentions. A known trigger gets 1 as the similarity score. A high similarity score of a word to a known trigger of a relationship indicates that there is a high chance of that particular word to be a trigger of that particular relation. We defined the features of existence of trigger words by the trigger word similarity scores. We noticed that a classifier model trained with only linguistic features cannot result a good score for entity level relation validation task.

Global association among different entities provides useful information about their relatedness that cannot be captured by linguistic analysis. Two related entities share some common ideas and resources with their neighbors. Therefore, we proposed to explore the communities of two entities which were claimed to be in a relationship. We constructed a graph of entities by extracting different types of named entities from a given text corpus. Entities, documents and sentences were presented by graph nodes. Moreover, the coexistence of a pair of entities in a sentence was denoted by an IN_SAME_SENTENCE relational edge. The associations among the entities, mentions, sentences and documents were maintained by distinct relational edges as well. This graph facilitated us to find the community of an entity based on IN_SAME_SENTENCE relation. We proposed to compute several features on the community graphs which are network density, network similarity, eigenvector centrality and mutual information. These graph based features provided more clues in addition to the linguistic features to decide whether two entities are in a true relationship.

Our experimental results showed that the proposed features significantly improved the performance in relation validation over some baseline linguistic features. Our relation validation model discarded a large number of wrong relations. Furthermore, we employed our relation validation model to populate a knowledge base in an ensemble fashion. Our relation validation based ensemble system outperformed the best slot filling system in KBP task for some trigger-dependent relations.

## 7.2 Future Work

We aimed at discarding a large part of the wrong relation hypotheses by validating the outcomes of different relation extraction systems. Although our relation validation system outperforms some baselines, still it fails to achieve a very high precision. Moreover, we notice that the relation validation model discards some true relations that causes the drop of recall. Still there are many issues to improve our relation validation system that we could not focus due to various shortcomings. Both of linguistic and graph analysis can be further investigated to achieve a better performance.

- Our system learned trigger words and syntactic dependency patterns of various relations from a very small amount of labeled dataset. In supervised learning, a system requires to be trained with a large number of labeled data in order to achieve a very high performance.

- We did not focus on trigger-independent relations. Validating a trigger-independent relation becomes harder due to lack of semantic information. Such relations mostly rely on syntactic patterns. Therefore, syntactic patterns can be explored more for validating trigger-independent relations.

- Some relations such as *person* and *children* share common trigger words that cause difficulty in relation validation task. In such cases, learning directionality of relation is very important. We did not focus on this issue in this thesis that can be studied in future.

- A long sentence contains more information compared to a short one. If the distance between two related entity mentions is very long, validation of the relationship is affected by noise. Moreover, many relation expressing sentences are not grammatically correct specially for discussion_forum data. The linguistic analyzer such as dependency parser fails to analyze a long or grammatically incorrect sentences properly. Therefore, it can be investigated how to capture useful information from long and grammatically incorrect sentences.

- Construction of entity graph relies on recognition of named entity recognition from texts. We noticed that existing NER tools fail to detect many named entities. Sometimes, the type of a detected named entity is not correct. Moreover, co-references of the named entities are not resolved during construction of the entity graph. Named entity disambiguation is an important prerequisite for constructing an entity graph. We did not evaluate our entity disambiguation method. These issues can be further studied to construct a more precise and rich graph.

- In our system, time complexity is very high for extracting community of an entity and for computing features on the community graphs. Efficient algorithms can be explored to overcome this limitation.
- In recent years, neural networks have been very effective and popular for solving NLP related problems. Neural network based learning systems do not require explicit feature engineering. In this thesis, we did not use any neural network based technique for validating relations. Therefore, it could be a good idea to explore neural networks for relation validation task.
- We performed only one experiment for unsupervised relation validation. Due to time limitation, we could not explore in depth of unsupervised learning. It can be further studied for relation validation task.

Community graph based features have shown a great positive impact on relation validation task specially when they are combined with some baseline linguistic features. Moreover, these features have contributed to populating a more accurate knowledge base as well. The proposed features can also be used in relation extraction task to achieve a better precision. However, we have addressed some drawbacks that could be improved further to achieve a better performance in both relation validation and knowledge base population tasks.

# Bibliography

Adel, Heike, Benjamin Roth, and Hinrich Schütze (2016). "Comparing Convolutional Neural Networks to Traditional Models for Slot Filling". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, pp. 828–838. URL: http://www.aclweb.org/anthology/N16-1097.

Adel, Heike and Hinrich Schütze (2015). "Cis at tac cold start 2015: Neural networks and coreference resolution for slot filling". In: *Proc. TAC2015*.

Aguilar, Jacqueline et al. (2014). "A comparison of the events and relations across ace, ere, tac-kbp, and framenet annotation standards". In: *Proceedings of the Second Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pp. 45–53.

Alfonseca, Enrique et al. (2012). "Pattern learning for relation extraction with a hierarchical topic model". In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*. Association for Computational Linguistics, pp. 54–59.

Alhelbawy, Ayman and Robert J Gaizauskas (2014). "Graph Ranking for Collective Named Entity Disambiguation." In: *ACL (2)*, pp. 75–80.

Angeli, Gabor et al. (2014). "Stanford's 2014 slot filling systems". In: *TAC KBP*.

Angeli, Gabor et al. (2015). "Bootstrapped self training for knowledge base population". In: *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

Auer, Sören et al. (2007). "Dbpedia: A nucleus for a web of open data". In: *The semantic web*, pp. 722–735.

Augenstein, Isabelle (2016). "Web Relation Extraction with Distant Supervision". PhD thesis. University of Sheffield.

Banko, Michele et al. (2007). "Open Information Extraction from the Web." In: *IJCAI*. Vol. 7, pp. 2670–2676.

Bengio, Yoshua et al. (2003). "A neural probabilistic language model". In: *Journal of machine learning research* 3.Feb, pp. 1137–1155.

Bentor, Yinon et al. (2013). "University of Texas at Austin KBP 2013 Slot Filling System: Bayesian Logic Programs for Textual Inference." In: *TAC*.

Biega, Joanna, Erdal Kuzey, and Fabian M Suchanek (2013). "Inside YAGO2s: A transparent information extraction architecture". In: *Proceedings of the 22nd International Conference on World Wide Web*. ACM, pp. 325–328.

Bollacker, Kurt et al. (2008). "Freebase: a collaboratively created graph database for structuring human knowledge". In: *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. AcM, pp. 1247–1250.

Bonacich, Phillip and Paulette Lloyd (2001). "Eigenvector-like measures of centrality for asymmetric relations". In: *Social networks* 23.3, pp. 191–201.

Boroushaki, Soheil and Jacek Malczewski (2010). "Using the fuzzy majority approach for GIS-based multicriteria group decision-making". In: *Computers & Geosciences* 36.3, pp. 302–312.

Bougouin, Adrien, Florian Boudin, and Béatrice Daille (2013). "Topicrank: Graph-based topic ranking for keyphrase extraction". In: *International Joint Conference on Natural Language Processing (IJCNLP)*, pp. 543–551.

Breiman, Leo (1996). "Bagging predictors". In: *Machine learning* 24.2, pp. 123–140.

Brill, Eric and Robert C Moore (2000). "An improved error model for noisy channel spelling correction". In: *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pp. 286–293.

Brin, Sergey and Lawrence Page (1998). "The anatomy of a large-scale hypertextual Web search engine* 1". In: *Computer networks and ISDN systems* 30.1-7, pp. 107–117.

Bunescu, Razvan and Raymond Mooney (2007). "Learning to extract relations from the web using minimal supervision". In: *ACL*.

Bunescu, Razvan C and Raymond J Mooney (2005). "A shortest path dependency kernel for relation extraction". In: *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 724–731.

Bunescu, Razvan C and Marius Pasca (2006). "Using Encyclopedic Knowledge for Named entity Disambiguation." In: *EACL*. Vol. 6, pp. 9–16.

Cai, Rui, Xiaodong Zhang, and Houfeng Wang (2016). "Bidirectional Recurrent Convolutional Neural Network for Relation Classification". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 756–765. URL: http://www.aclweb.org/anthology/P16-1072.

Cao, Jiuwen et al. (2012). "Voting based extreme learning machine". In: *Information Sciences* 185.1, pp. 66–77.

Chang, H et al. (2016). "Extracting Multilingual Relations under Limited Resources: TAC 2016 Cold-Start KB construction and Slot-Filling using Compositional Universal Schema". In: *Proceedings of TAC*.

Cheung, Chi Fai et al. (2003). "A multi-perspective knowledge-based system for customer service management". In: *Expert systems with applications* 24.4, pp. 457–470.

Chinchor, Nancy and Elaine Marsh (1998). "Muc-7 information extraction task definition". In: *Proceeding of the seventh message understanding conference (MUC-7), Appendices*, pp. 359–367.

Chowdhury, Faisal Mahbub, Alberto Lavelli, and Alessandro Moschitti (2011). "A study on dependency tree kernels for automatic extraction of protein-protein interaction". In: *Proceedings of BioNLP 2011 Workshop*. Association for Computational Linguistics, pp. 124–133.

Craven, Mark, Johan Kumlien, et al. (1999). "Constructing biological knowledge bases by extracting information from text sources." In: *ISMB*. Vol. 1999, pp. 77–86.

Culotta, Aron and Jeffrey Sorensen (2004). "Dependency tree kernels for relation extraction". In: *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, p. 423.

Demner-Fushman, Dina and Jimmy Lin (2007). "Answering clinical questions with knowledge-based and statistical techniques". In: *Computational Linguistics* 33.1, pp. 63–103.

Dietterich, Thomas G (2002). "Ensemble learning". In: *The handbook of brain theory and neural networks* 2, pp. 110–125.

Dligach, Dmitriy et al. (2017). "Neural Temporal Relation Extraction". In: *EACL 2017*, p. 746.

Erkan, Günes and Dragomir R Radev (2004). "Lexrank: Graph-based lexical centrality as salience in text summarization". In: *Journal of Artificial Intelligence Research* 22, pp. 457–479.

Fader, Anthony, Stephen Soderland, and Oren Etzioni (2011). "Identifying relations for open information extraction". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pp. 1535–1545.

Fader, Anthony, Luke Zettlemoyer, and Oren Etzioni (2014). "Open question answering over curated and extracted knowledge bases". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, pp. 1156–1165.

Fang, Yuan and Kevin Chen-Chuan Chang (2011). "Searching patterns for relation extraction over the web: rediscovering the pattern-relation duality". In: *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, pp. 825–834.

Fortunato, Santo (2010). "Community detection in graphs". In: *Physics reports* 486.3, pp. 75–174.

Freeman, Linton C (1977). "A set of measures of centrality based on betweenness". In: *Sociometry*, pp. 35–41.

Freund, Yoav and Robert E Schapire (1995). "A desicion-theoretic generalization of on-line learning and an application to boosting". In: *European conference on computational learning theory*. Springer, pp. 23–37.

Friedl, Dipl-Math Bettina, Julia Heidemann, et al. (2010). "A critical review of centrality measures in social networks". In: *Business & Information Systems Engineering* 2.6, pp. 371–385.

Girju, Roxana et al. (2007). "Semeval-2007 task 04: Classification of semantic relations between nominals". In: *Proceedings of the 4th International Workshop on Semantic Evaluations*. Association for Computational Linguistics, pp. 13–18.

Girvan, Michelle and Mark EJ Newman (2002). "Community structure in social and biological networks". In: *Proceedings of the national academy of sciences* 99.12, pp. 7821–7826.

Grishman, Ralph and Beth Sundheim (1996). "Message understanding conference-6: A brief history". In: *Proceedings of the 16th conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, pp. 466–471.

Guo, Yuhang et al. (2011). "A Graph-based Method for Entity Linking." In: *IJCNLP*. Citeseer, pp. 1010–1018.

GuoDong, Zhou et al. (2005). "Exploring various knowledge in relation extraction". In: *Proceedings of the 43rd annual meeting on association for computational linguistics*. Association for Computational Linguistics, pp. 427–434.

Hachey, Ben, Will Radford, and James R Curran (2011). "Graph-based named entity linking with wikipedia". In: *Web Information System Engineering–WISE 2011*. Springer, pp. 213–226.

Han, Xianpei, Le Sun, and Jun Zhao (2011). "Collective entity linking in web text: a graph-based method". In: *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. ACM, pp. 765–774.

Hearst, Marti A (1992). "Automatic acquisition of hyponyms from large text corpora". In: *Proceedings of the 14th conference on Computational linguistics-Volume 2*. Association for Computational Linguistics, pp. 539–545.

Hoffart, Johannes et al. (2013). "YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia". In: *Artificial Intelligence* 194, pp. 28–61.

Hoffmann, Raphael et al. (2011). "Knowledge-based weak supervision for information extraction of overlapping relations". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, pp. 541–550.

Holzinger, Andreas et al. (2013). "On graph entropy measures for knowledge discovery from publication network data". In: *Availability, reliability, and security in information systems and HCI*. Springer, pp. 354–362.

Hong, Yu et al. (2014). "Rpi blender tac-kbp2014 knowledge base population system". In: *Proc. Text Analysis Conference (TAC 2014)*.

Jain, Alpa and Patrick Pantel (2010). "Factrank: Random walks on a web of facts". In: *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, pp. 501–509.

Jean-Louis, Ludovic, Romaric Besançon, and Olivier Ferret (2011). "Text Segmentation and Graph-based Method for Template Filling in Information Extraction." In: *IJCNLP*, pp. 723–731.

Ji, Heng et al. (2010). "Overview of the TAC 2010 knowledge base population track". In: *Third Text Analysis Conference (TAC 2010)*.

Ji, Heng et al. (2016). "Overview of TAC-KBP2016 Tri-lingual EDL and its impact on end-to-end Cold-Start KBP". In: *Proceedings of TAC*.

Jiang, Jing and ChengXiang Zhai (2007). "A Systematic Exploration of the Feature Space for Relation Extraction." In: *HLT-NAACL*, pp. 113–120.

Jurafsky, D. and J.H. Martin (2014). *Speech and Language Processing*. Always learning. Pearson. ISBN: 9781292025438. URL: https://books.google.fr/books?id=km-kngEACAAJ.

Kambhatla, Nanda (2004). "Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations". In: *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*. Association for Computational Linguistics, p. 22.

Katz, Leo (1953). "A new status index derived from sociometric analysis". In: *Psychometrika* 18.1, pp. 39–43.

Khodakarami, Farnoosh and Yolande E Chan (2014). "Exploring the role of customer relationship management (CRM) systems in customer knowledge creation". In: *Information & Management* 51.1, pp. 27–42.

Kim, Daehwan et al. (2013). "TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions". In: *Genome biology* 14.4, R36.

Kleinberg, Jon M (1999). "Authoritative sources in a hyperlinked environment". In: *Journal of the ACM (JACM)* 46.5, pp. 604–632.

Lehmann, Jens et al. (2015). "DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia". In: *Semantic Web* 6.2, pp. 167–195.

Levenshtein, Vladimir I (1966). "Binary codes capable of correcting deletions, insertions, and reversals". In: *Soviet physics doklady*. Vol. 10. 8, pp. 707–710.

Li, Xiang (2016). "IMPROVING KNOWLEDGE BASE POPULATION WITH INFORMATION EXTRACTION". PhD thesis. New York University.

Li, Yaliang et al. (2011). "Extracting Relation Descriptors with Conditional Random Fields." In: *IJCNLP*, pp. 392–400.

Li, Yan et al. (2013). "PRIS at Knowledge Base Population 2013". In: *Proc. TAC 2013 Workshop*.

Lin, Hailun et al. (2014). "OpenKN at TAC KBP 2014". In: *TAC KBP*.

Litvak, Marina and Mark Last (2008). "Graph-based keyword extraction for single-document summarization". In: *Proceedings of the workshop on Multi-source Multilingual Information Extraction and Summarization*. Association for Computational Linguistics, pp. 17–24.

Liu, Yang et al. (2015). "A Dependency-Based Neural Network for Relation Classification". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China: Association for Computational Linguistics, pp. 285–290. URL: http://www.aclweb.org/anthology/P15-2047.

Mahdisoltani, Farzaneh, Joanna Biega, and Fabian Suchanek (2014). "Yago3: A knowledge base from multilingual wikipedias". In: *7th Biennial Conference on Innovative Data Systems Research*. CIDR Conference.

Manning, Christopher et al. (2014a). "The Stanford CoreNLP natural language processing toolkit". In: *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pp. 55–60.

Manning, Christopher D. et al. (2014b). "The Stanford CoreNLP Natural Language Processing Toolkit". In: *Association for Computational Linguistics (ACL) System Demonstrations*, pp. 55–60. URL: http://www.aclweb.org/anthology/P/P14/P14-5010.

McNamee, Paul and Hoa Trang Dang (2009). "Overview of the TAC 2009 knowledge base population track". In: *Text Analysis Conference (TAC)*. Vol. 17, pp. 111–113.

Mesnard, Olivier et al. (2016). "Construction automatisée d'une base de connaissances". In: *PARIS Inalco du 4 au 8 juillet 2016*, p. 12.

Mihalcea, Rada and Paul Tarau (2004). "TextRank: Bringing Order into Text." In: *EMNLP*. Vol. 4, pp. 404–411.

Mikolov, Tomas, Wen-tau Yih, and Geoffrey Zweig (2013). "Linguistic regularities in continuous space word representations." In: *hlt-Naacl*. Vol. 13, pp. 746–751.

Mikolov, Tomas et al. (2013). "Efficient estimation of word representations in vector space". In:

Mintz, Mike et al. (2009). "Distant supervision for relation extraction without labeled data". In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*. Association for Computational Linguistics, pp. 1003–1011.

Morais, Danielle C and Adiel Teixeira de Almeida (2012). "Group decision making on water resources based on analysis of individual rankings". In: *Omega* 40.1, pp. 42–52.

Morrison, Donn, Ruili Wang, and Liyanage C De Silva (2007). "Ensemble methods for spoken emotion recognition in call-centres". In: *Speech communication* 49.2, pp. 98–112.

Nguyen, Thien Huu and Ralph Grishman (2015). "Relation Extraction: Perspective from Convolutional Neural Networks." In: *VS@ HLT-NAACL*, pp. 39–48.

Nguyen, Thien Huu et al. (2014). "New york university 2014 knowledge base population systems". In: *Proc. Text Analysis Conference (TAC2014)*.

Nguyen, Truc-Vien T, Alessandro Moschitti, and Giuseppe Riccardi (2009). "Convolution kernels on constituent, dependency and sequential structures for relation extraction". In: *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*. Association for Computational Linguistics, pp. 1378–1387.

Niu, Feng et al. (2012). "DeepDive: Web-scale Knowledge-base Construction using Statistical Learning and Inference." In: *VLDS* 12, pp. 25–28.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning (2014). "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543. URL: http://www.aclweb.org/anthology/D14-1162.

Pershina, Maria, Yifan He, and Ralph Grishman (2015). "Personalized Page Rank for Named Entity Disambiguation." In: *HLT-NAACL*, pp. 238–243.

Pershina, Maria et al. (2014). "Infusion of Labeled Data into Distant Supervision for Relation Extraction." In: *ACL (2)*, pp. 732–738.

Polikar, Robi (2006). "Ensemble based systems in decision making". In: *IEEE Circuits and systems magazine* 6.3, pp. 21–45.

Rao, Delip, Paul McNamee, and Mark Dredze (2013). "Entity linking: Finding extracted entities in a knowledge base". In: *Multi-source, multilingual information extraction and summarization*. Springer, pp. 93–115.

Riedel, Sebastian, Limin Yao, and Andrew McCallum (2010). "Modeling relations and their mentions without labeled text". In: *Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 148–163.

Rodriguez, Miguel, Sean Goldberg, and Daisy Zhe Wang (2015). "University of Florida DSR lab system for KBP slot filler validation 2015". In: *Proceedings of the Eighth Text Analysis Conference (TAC2015)*.

Rodrıguez, Miguel and Daisy Zhe Wang (2016). "University of Florida 2016 Slot Filler Validation system". In:

Rosenfeld, Benjamin and Ronen Feldman (2006). "Ures: an unsupervised web relation extraction system". In: *Proceedings of the COLING/ACL on Main conference poster sessions*. Association for Computational Linguistics, pp. 667–674.

— (2007a). "Clustering for unsupervised relation identification". In: *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*. ACM, pp. 411–418.

— (2007b). "Using corpus statistics on entities to improve semi-supervised relation extraction from the web". In: *ACL*.

Rosvall, Martin and Carl T Bergstrom (2007). "An information-theoretic framework for resolving community structure in complex networks". In: *Proceedings of the National Academy of Sciences* 104.18, pp. 7327–7331.

Roth, Benjamin et al. (2014). "Universal schema for slot-filling, cold-start KBP and event argument extraction: UMAss IESL at TAC KBP 2014". In: *Text Analysis Conference (Knowledge Base Population Track)'14 Workshop (TAC KBP), Gaithersburg, Maryland, USA*.

Sammons, Mark et al. (2014). "Overview of UI-CCG systems for event argument extraction, entity discovery and linking, and slot filler validation". In: *Urbana* 51, p. 61801.

Schwenk, Holger, Daniel Dchelotte, and Jean-Luc Gauvain (2006). "Continuous space language models for statistical machine translation". In: *Proceedings of the COLING/ACL on Main conference poster sessions*. Association for Computational Linguistics, pp. 723–730.

Shannon, Claude E (1948). "The Bell System Technical Journal". In: *A mathematical theory of communication* 27, pp. 379–423.

Shen, Wei, Jianyong Wang, and Jiawei Han (2015). "Entity linking with a knowledge base: Issues, techniques, and solutions". In: *IEEE Transactions on Knowledge and Data Engineering* 27.2, pp. 443–460.

Singh-Blom, U Martin et al. (2013). "Prediction and validation of gene-disease associations using methods inspired by social network analyses". In: *PloS one* 8.5, e58977.

Singhal, Amit (2012). "Introducing the knowledge graph: things, not strings, 2012". In: *Official Blog (of Google)*.

Snow, Rion, Daniel Jurafsky, and Andrew Y Ng (2005). "Learning syntactic patterns for automatic hypernym discovery". In: *Advances in neural information processing systems*, pp. 1297–1304.

Sokolova, Marina and Guy Lapalme (2009). "A systematic analysis of performance measures for classification tasks". In: *Information Processing & Management* 45.4, pp. 427–437.

Sterckx, Lucas et al. (2015). "Ghent University-IBCN participation in the TAC KBP 2015 cold start slot filling task". In: *8th Text Analysis Conference*.

Student (1908). "The probable error of a mean". In: *Biometrika*, pp. 1–25.

Suchanek, Fabian M, Gjergji Kasneci, and Gerhard Weikum (2007). "Yago: a core of semantic knowledge". In: *Proceedings of the 16th international conference on World Wide Web*. ACM, pp. 697–706.

Surdeanu, Mihai (2013). "Overview of the TAC2013 knowledge base population evaluation: English slot filling and temporal slot filling". In: *Proceedings of the Sixth Text Analysis Conference (TAC 2013)*.

Surdeanu, Mihai and Heng Ji (2014). "Overview of the english slot filling track at the tac2014 knowledge base population evaluation". In: *Proc. Text Analysis Conference (TAC2014)*.

Surdeanu, Mihai et al. (2012). "Multi-instance multi-label learning for relation extraction". In: *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, pp. 455–465.

Viswanathan, Vidhoon et al. (2015). "Stacked Ensembles of Information Extractors for Knowledge-Base Population". In: *Proceedings of ACL*.

Vu, Ngoc Thang et al. (2016). "Combining Recurrent and Convolutional Neural Networks for Relation Classification". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, pp. 534–539. URL: http://www.aclweb.org/anthology/N16-1065.

Wang, I-Jeng et al. (2013). "JHUAPL TAC-KBP2013 Slot Filler Validation System". In: *Proceedings of the Sixth Text Analysis Conference (TAC 2013)*. Vol. 24.

Wiegand, Benjamin Roth Tassilo Barth Michael and Mittul Singh Dietrich Klakow (2013). "Effective slot filling based on shallow distant supervision methods". In: *Proceedings of NIST KBP workshop*. Vol. 1.

Wolpert, David H (1992). "Stacked generalization". In: *Neural networks* 5.2, pp. 241–259.

Xu, Kun et al. (2015). "Semantic Relation Classification via Convolutional Neural Networks with Simple Negative Sampling". In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: Association for Computational Linguistics, pp. 536–540. URL: http://aclweb.org/anthology/D15-1062.

Xu, Zheng et al. (2014). "Mining temporal explicit and implicit semantic relations between entities using web search engines". In: *Future Generation Computer Systems* 37, pp. 468–477.

Yang, Jaewon and Jure Leskovec (2012). "Community-affiliation graph model for overlapping network community detection". In: *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, pp. 1170–1175.

Yang, Pengyi et al. (2010). "A review of ensemble methods in bioinformatics". In: *Current Bioinformatics* 5.4, pp. 296–308.

Yangarber, Roman et al. (2000). "Automatic acquisition of domain knowledge for information extraction". In: *Proceedings of the 18th conference on Computational linguistics-Volume 2*. Association for Computational Linguistics, pp. 940–946.

Yu, Dian and Heng Ji (2016). "Unsupervised Person Slot Filling based on Graph Mining". In: *ACL*.

Yu, Dian et al. (2013). "RPI-BLENDER TAC-KBP2013 Knowledge Base Population System." In: *TAC*.

Yu, Dian et al. (2014). "The Wisdom of Minority: Unsupervised Slot Filling Validation based on Multi-dimensional Truth-Finding." In: *COLING*, pp. 1567–1578.

Yu, Dian et al. (2016). "RPI BLENDER TAC-KBP2016 System Description". In: *Proceedings of TAC*.

Zelenko, Dmitry, Chinatsu Aone, and Anthony Richardella (2003). "Kernel methods for relation extraction". In: *The Journal of Machine Learning Research* 3, pp. 1083–1106.

Zeng, Daojian et al. (2014). "Relation Classification via Convolutional Deep Neural Network." In: *COLING*, pp. 2335–2344.

Zhang, Min et al. (2006). "A composite kernel to extract relations between entities with both flat and structured features". In: *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the*

*Association for Computational Linguistics*. Association for Computational Linguistics, pp. 825–832.

Zhang, Yuhao et al. (2016). "Stanford at TAC KBP 2016: Sealing Pipeline Leaks and Understanding Chinese". In: *Proceedings of TAC*.

Zheng, Suncong et al. (2016). "A neural network framework for relation extraction: Learning entity semantic and relation pattern". In: *Knowledge-Based Systems* 114, pp. 12–23.

Zhou, Guodong et al. (2007). "Tree kernel-based relation extraction with context-sensitive structured parse tree information". In: *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

**Titre :** Peuplement d'une base de connaissance fondé sur l'exploitation d'un graphe d'entités

**Mots clés :** traitement des langues naturelles, peuplement de base de connaissance, extraction de relation, validation de relation, graphe de communauté

**Résumé :** Le peuplement de base de connaissance (KBP) est une tâche importante qui présente de nombreux défis pour le traitement automatique des langues. L'objectif de cette tâche est d'extraire des connaissances de textes et de les structurer afin de compléter une base de connaissances. Nous nous sommes intéressé à la reconnaissance de relations entre entités.
L'extraction de relations (RE) entre une paire de mentions d'entités est une tâche difficile en particulier pour les relations en domaine ouvert. Généralement, ces relations sont extraites en fonction des informations lexicales et syntaxiques au niveau de la phrase. Cependant, l'exploitation d'informations globales sur les entités n'a pas encore été explorée.
Nous proposons d'extraire un graphe d'entités du corpus global et de calculer des caractéristiques sur ce graphe afin de capturer des indices des relations entre paires d'entités.
Pour évaluer la pertinence des fonctionnalités proposées, nous les avons testées sur une tâche de validation de relation dont le but est de décider l'exactitude de relations extraites par différents systèmes. Les résultats expérimentaux montrent que les caractéristiques proposées conduisent à améliorer les résultats de l'état de l'art.

**Title :** Knowledge Base Population based on Entity Graph Analysis

**Keywords :** natural language processing, knowledge base population, relation extraction, relation validation, community graph

**Abstract :** Knowledge Base Population (KBP) is an important and challenging task specially when it has to be done automatically. The objective of KBP task is to make a collection of facts of the world. A Knowledge Base (KB) contains different entities, relationships among them and various properties of the entities.
Relation extraction (RE) between a pair of entity mentions from text plays a vital role in KBP task. RE is also a challenging task specially for open domain relations. Generally, relations are extracted based on the lexical and syntactical information at the sentence level. However, global information about known entities has not been explored yet for RE task.
We propose to extract a graph of entities from the overall corpus and to compute features on this graph that are able to capture some evidence of holding relationships between a pair of entities.
In order to evaluate the relevance of the proposed features, we tested them on a task of relation validation which examines the correctness of relations that are extracted by different RE systems. Experimental results show that the proposed features lead to outperforming the state-of-the-art system.

**Université Paris-Saclay**
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France