VIETNAM NATIONAL UNIVERSITY, HANOI
UNIVERSITY OF ENGINEERING AND TECHNOLOGY

CAN DUY CAT

ADVANCED DEEP LEARNING MODELS AND APPLICATIONS IN SEMANTIC RELATION EXTRACTION

MASTER THESIS

Major: Computer Science

HA NOI - 2019

VIETNAM NATIONAL UNIVERSITY, HANOI
UNIVERSITY OF ENGINEERING AND TECHNOLOGY

Can Duy Cat

ADVANCED DEEP LEARNING MODELS AND APPLICATIONS IN SEMANTIC RELATION EXTRACTION

MASTER THESIS

Major: Computer Science

Supervisor: **Assoc.Prof. Ha Quang Thuy**

Assoc.Prof. Chng Eng Siong

[illegible] Chng Eng Siong.

HA NOI - 2019

Abstract

Relation Extraction (RE) is one of the most fundamental task of Natural Language Processing (NLP) and Information Extraction (IE). To extract the relationship between two entities in a sentence, two common approaches are (1) using their shortest dependency path (SDP) and (2) using an attention model to capture a context-based representation of the sentence. Each approach suffers from its own disadvantage of either missing or redundant information. In this work, we propose a novel model that combines the advantages of these two approaches. This is based on the basic information in the SDP enhanced with information selected by several attention mechanisms with kernel filters, namely RbSP (Richer-but-Smarter SDP). To exploit the representation behind the RbSP structure effectively, we develop a combined Deep Neural Network (DNN) with a Long Short-Term Memory (LSTM) network on word sequences and a Convolutional Neural Network (CNN) on RbSP.

Furthermore, experiments on the task of RE proved that data representation is one of the most influential factors to the model's performance but still has many limitations. We propose (i) a compositional embedding that combines several dominant linguistic as well as architectural features and (ii) dependency tree normalization techniques for generating rich representations for both words and dependency relations in the SDP.

Experimental results on both general data (SemEval-2010 Task 8) and biomedical data (BioCreative V Track 3 CDR) demonstrate the out-performance of our proposed model over all compared models.

Keywords: *Relation Extraction, Shortest Dependency Path, Convolutional Neural Network, Long Short-Term Memory, Attention Mechanism.*

Acknowledgements

I would first like to thank my thesis supervisor Assoc.Prof. Ha Quang Thuy of the Data Science and Knowledge Technology Laboratory at University of Engineering and Technology. He consistently allowed this paper to be my own work, but steered me in the right the direction whenever he thought I needed it.

I also want to acknowledge my co-supervisor Assoc.Prof Chng Eng Siong from Nanyang Technological University, Singapore for offering me the internship opportunities at NTU, Singapore and leading me working on diverse exciting projects.

Furthermore, I am very grateful to my external advisor MSc. Le Hoang Quynh, for insightful comments both in my work and in this thesis, for her support, and for many motivating discussions.

In addition, I have been very privileged to get to know and to collaborate with many other great collaborators. I would like to thank BSc. Nguyen Minh Trang and BSc. Nguyen Duc Canh for inspiring discussion, and for all the fun we have had over the last two years. I thank to MSc. Ho Thi Nga and MSc. Vu Thi Ly for continuous support during the time in Singapore.

Finally, I must express my very profound gratitude to my family for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them.

Declaration

I declare that the thesis has been composed by myself and that the work has not be submitted for any other degree or professional qualification. I confirm that the work submitted is my own, except where work which has formed part of jointly-authored publications has been included. My contribution and those of the other authors to this work have been explicitly indicated below. I confirm that appropriate credit has been given within this thesis where reference has been made to the work of others.

The model presented in Chapter 3 and the results presented in Chapter 4 was previously published in the Proceedings of ACIIDS 2019 as *"Improving Semantic Relation Extraction System with Compositional Dependency Unit on Enriched Shortest Dependency Path"* and NAACL-HTL 2019 as *"A Richer-but-Smarter Shortest Dependency Path with Attentive Augmentation for Relation Extraction"* by myself et al. This study was conceived by all of the authors. I carried out the main idea(s) and implemented all the model(s) and material(s).

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have fully authorship to improve these materials.

Master student

Can Duy Cat

Table of Contents

Acronyms

Adam	Adaptive Moment Estimation
ANN	Artificial Neural Network
BiLSTM	Bidirectional Long Short-Term Memory
CBOW	Continuous Bag-Of-Words
CDR	Chemical Disease Relation
CID	Chemical-Induced Disease
CNN	Convolutional Neural Network
DNN	Deep Neural Network
DU	Dependency Unit
GD	Gradient Descent
IE	Information Extraction
LSTM	Long Short-Term Memory
MLP	Multilayer Perceptron
NE	Named Entity
NER	Named Entity Recognition
NLP	Natural Language Processing
POS	Part-Of-Speech

RbSP	Richer-but-Smarter Shortest Dependency Path
RC	Relation Classification
RE	Relation Extraction
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SDP	Shortest Dependency Path
SVM	Suport Vector Machine

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

With the advent of the Internet, we are stepping in to a new era, the era of information and technology where the growth and development of each individual, organization, and society is relied on the main strategic resource - information. There exists a large amount of unstructured digital data that are created and maintained within an enterprise or across the Web, including news articles, blogs, papers, research publications, emails, reports, governmental documents, etc. Lot of important information is hidden within these documents that we need to extract to make them more accessible for further processing.

Many tasks of Natural Language Processing (NLP) would benefit from extracted information in large text corpora, such as Question Answering, Textual Entailment, Text Understanding, etc. For example, getting a paperwork procedure from a large collection of administrative documents is a complicated problem; it is far easier to get it from a structural database such as that shown above. Similarly, searching for the side effects of a chemical in the bio-medical literature will be much easier if these relations have been extracted from biomedical text.

We, therefore, have urge to turn unstructured text into structured by annotating semantic information. Normally, we are interested in relations between entities, such as person, organization, and location. However, it is impossible for human annotation because of sheer volume and heterogeneity of data. Instead, we would like to have a Relation Extraction (RE) system that annotate all data with the structure of our interest. In this thesis, we will focus on the task of recognizing relations between entities in unstructured text.

1.2 Problem Statement

Relation Extraction task includes of *detecting* and *classifying* relationship between entities within a set of artifacts, typically from text or XML documents. Figure 1.1 shows an overview of a typical pipeline for RE system. Here we have to sub-tasks: Named Entity Recognition (NER) task and Relation Classification (RC) task.

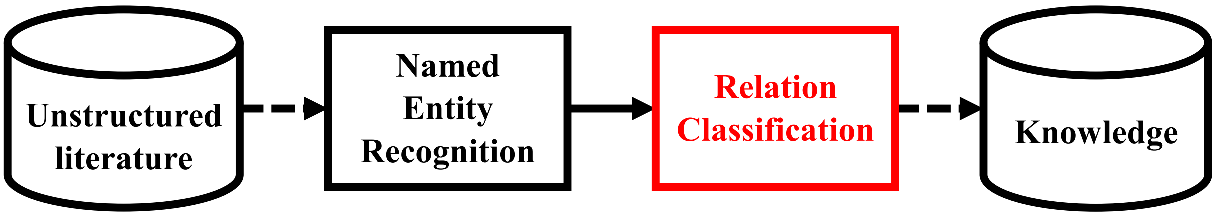


Figure 1.1: A typical pipeline of Relation Extraction system.

A **Named Entity** (NE) is a specific real-world object that is often represented by a word or phrase. It can be abstract or have a physical existence such as a person, a location, a organization, a product, a brand name, etc. For example, "*Hanoi*" and "*Vietnam*" are two named entities, and they are specific mentions in the following sentence: "*Hanoi city is the capital of Vietnam*". Named entities can simply be viewed as entity instances (e.g., *Hanoi* is an instance of a *city*). A named entity mention in a particular sentence can be using the name itself (*Hanoi*), nominal (*capital of Vietnam*), or pronominal (*it*). **Named Entity Recognition** is the task of seeking to locate and classify named entity mentions in unstructured text into pre-defined categories.

A **relation** usually denotes a well-defined (having a specific meaning) relationship between two or more NEs. It can be defined as a labeled tuple $R(e_1, e_2, ..., e_n)$ where the e_i are entities in a predefined relation R within document D. Most relation extraction systems focus on extracting binary relations. Some examples of relations are the relation `capital-of` between a `CITY` and a `COUNTRY`, the relation `author-of` between a `PERSON` and a `BOOK`, the relation `side-effect-of` between `DISEASEs` and a `CHEMICAL`, etc. It is also possible be the n-ary relation as well. For example, the relation `diagnose` between a `DOCTOR`, a `PATIENT` and a `DISEASE`. In short, **Relation classification** is the task of labeling each tuple of entities $(e_1, e_2, ..., e_n)$ a relation R from a pre-defined set. The main focus of this thesis is on classifying relation between two entities (or nominals).

1.2.1 Formal Definition

There have been many definitions for Relation Extraction problem. According to the definition in the study of Bach and Badaskar [5], we first model the relation extraction task as a classification problem (binary, or multi-class). There are many existing machine learning techniques which can be useful to train classifiers for relation extraction task. To keep it simple and clarified, we restrict our focus on relations between two entities.

Given a sentence $S = w_1 w_2 ... e_1 ... w_i ... e_2 ... w_{n-1} w_n$, where e_1 and e_2 are the entities, a mapping function $f(.)$ can be defined as:

$$f_R(T(S)) = \begin{cases} +1 & \text{If } e_1 \text{ and } e_2 \text{ are related according to relation } R \\ -1 & \text{Otherwise} \end{cases} \tag{1.1}$$

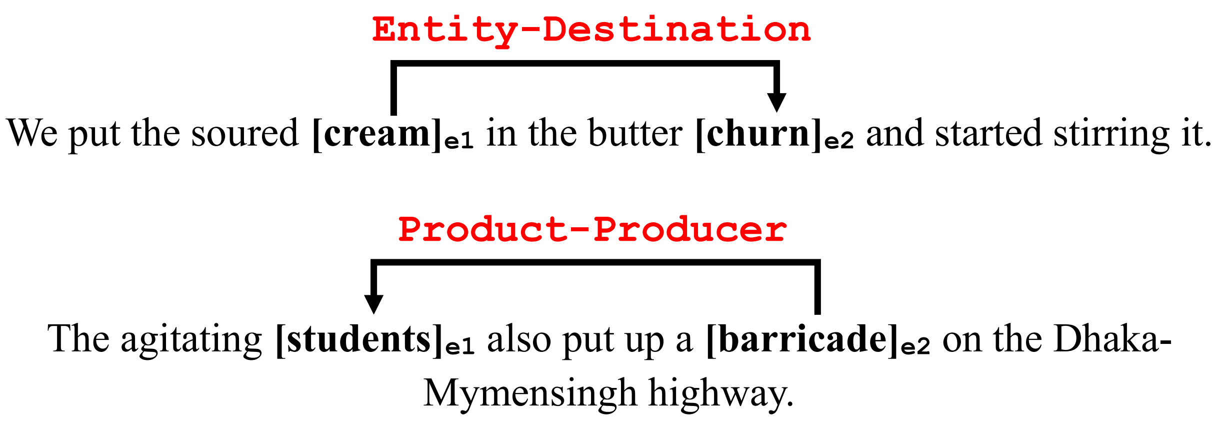
Where $T(S)$ is the set of features extracted for entity pair e_1 and e_2 from S. These features can be linguistic features from the sentence where these entities are mentioned or a structured representation of the sentence (labeled sequence, parse trees), etc. The mapping function $f(.)$ defines the existence of relation R between entities in the sentence. The discriminative classifier like Support Vector Machines (SVMs), Perceptron or Voted Perceptron are some examples for function $f(.)$ which can be used to train as a binary relation classifier. These classifiers can be trained using a set of features like linguistic features (Part-Of-Speech tags, corresponding entities, Bag-Of-Word, etc.) or syntactic features (dependency parse tree, shortest dependency path, etc.), which we discuss in Section 2.2.1. These features require a careful designed by experts and this takes huge time and effort, however cannot generalize data well enough.

Apart from these methods, Artificial Neural Network (ANN) based approaches are capable of reducing the effort to design a rich feature set. The input of a neural network can be words represented by word embedding and positional features based on the relative distance from the mentioned entities, etc and will be generalized to extract the relevant features automatically. With the feed-forward and back-propagation algorithm, the ANN can learn its parameters itself from data as well. The only things we need to concern are the way we design the network and how we feed data to it. Most recently, two dominant Deep Neural Networks (DNNs) are Convolutional Neural Network (CNN) [40] and Long Short-Term Memory (LSTM) [32]. We will discuss more on this topic in Section 2.2.2.

1.2.2 Examples

In this section, we shows some examples of semantic relations that annotated in text from many domains.

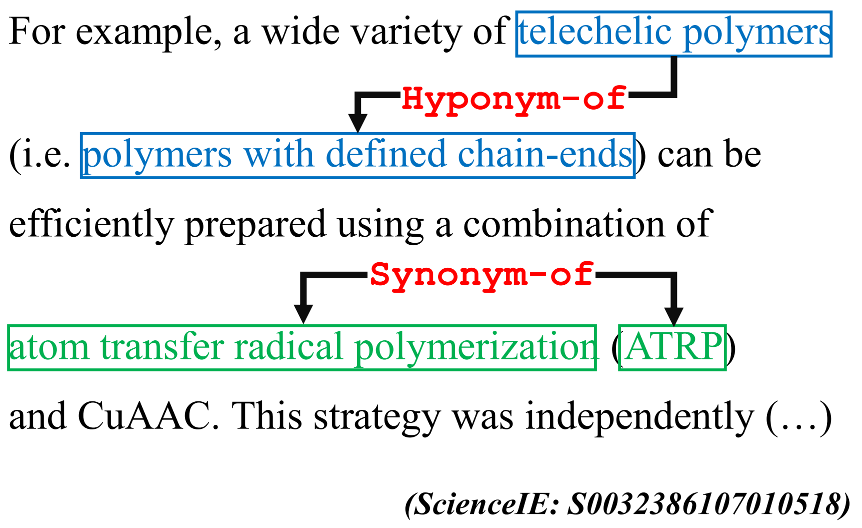
Figure 1.2 are two exemples from SemEval-2010 Task 8 dataset [30]. In these examples, the direction of relation is well-defined. Here nominals "*cream*" and "*churn*" in sentence (i) are of relation `Entity-Destination(e1,e2)` while nominals "*students*" and "*barricade*" are of relation `Product-Producer(e2,e1)`.



Entity-Destination
We put the soured [cream]e1 in the butter [churn]e2 and started stirring it.
Product-Producer
The agitating [students]e1 also put up a [barricade]e2 on the Dhaka-Mymensingh highway.

Figure 1.2: Two examples from SemEval 2010 Task 8 dataset.

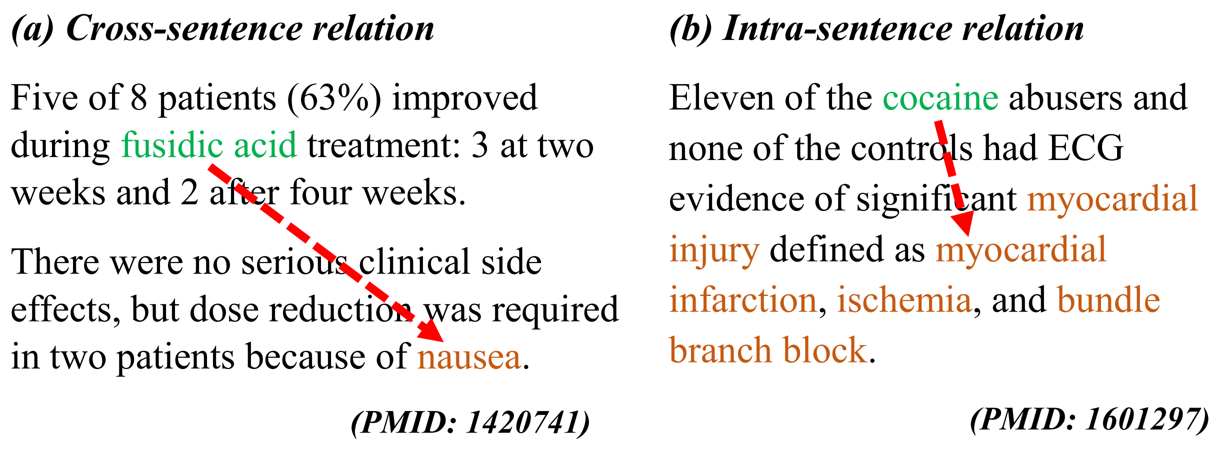
Figure 1.3 is an example form SemEval 2017 ScienceIE dataset [4]. In this sentence, we have two relations: `Hyponym-of` represented by an explanation pattern and `Synonym-of` relation represented by an abbreviation pattern. These patterns are different from semantic patterns in Figure 1.2. It require the adaptability of proposed model to perform well on both datasets.



For example, a wide variety of telechelic polymers
Hyponym-of
(i.e. polymers with defined chain-ends) can be
efficiently prepared using a combination of
Synonym-of
atom transfer radical polymerization (ATRP)
and CuAAC. This strategy was independently (…)
(ScienceIE: S0032386107010518)

Figure 1.3: Example from SemEval 2017 ScienceIE dataset.

Figure 1.4 includes examples form BioCreative 5 CDR corpus [65]. These examples show two `CID` relations between a chemical (in green) and a disease (in orange). However, example (a) is a cross-sentence relation (i.e., two corresponding entities belongs to two separate sentences) while example (b) is an intra-sentence relation (i.e., two corresponding entities belongs to the same sentence).



(a) Cross-sentence relation

Five of 8 patients (63%) improved during fusidic acid treatment: 3 at two weeks and 2 after four weeks.

There were no serious clinical side effects, but dose reduction was required in two patients because of nausea.

(PMID: 1420741)

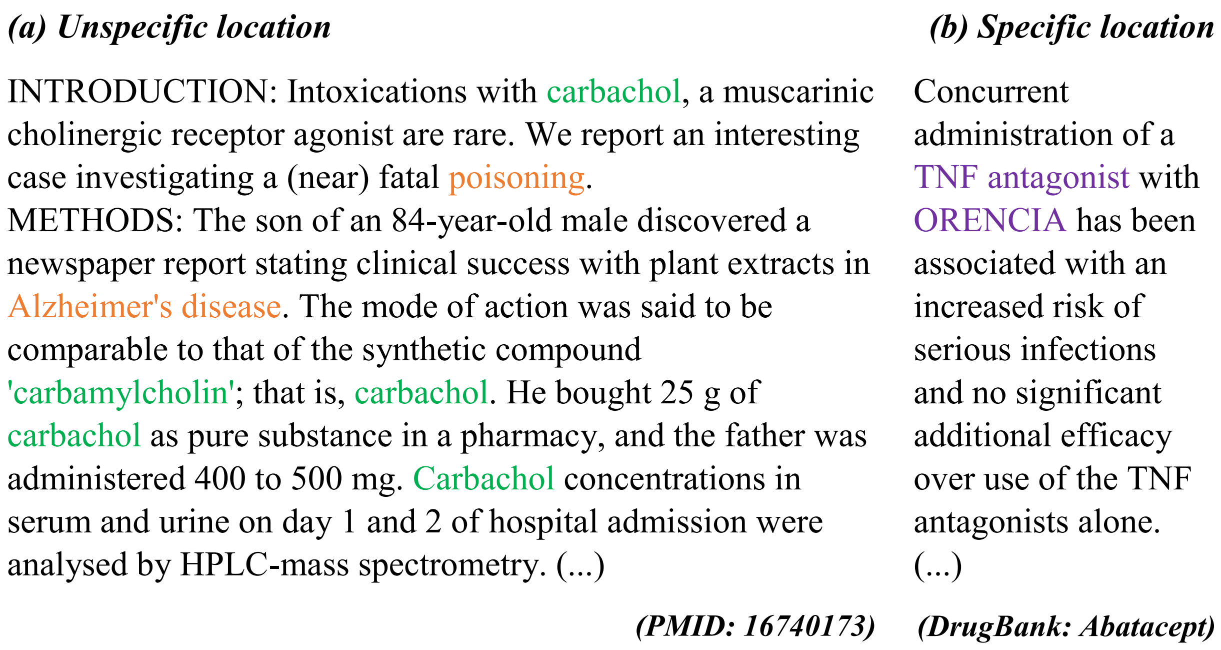
(b) Intra-sentence relation

Eleven of the cocaine abusers and none of the controls had ECG evidence of significant myocardial injury defined as myocardial infarction, ischemia, and bundle branch block.

(PMID: 1601297)

Figure 1.4: Examples of (a) cross-sentence relation and (b) intra-sentence relation.

Figure 1.5 indicates the difference of unspecific and specific location relations. Example (a) is an unspecific location relation from BioCreative V CDR corpus [65] that points out `CID` relations between *carbachol* and diseases without the location of corresponding entities. Example (b) is a specific location relation from the DDI DrugBank corpus [31] that specifies `Effect` relation between two drugs at a specific location.



(a) Unspecific location

INTRODUCTION: Intoxications with carbachol, a muscarinic cholinergic receptor agonist are rare. We report an interesting case investigating a (near) fatal poisoning.
METHODS: The son of an 84-year-old male discovered a newspaper report stating clinical success with plant extracts in Alzheimer's disease. The mode of action was said to be comparable to that of the synthetic compound 'carbamylcholin'; that is, carbachol. He bought 25 g of carbachol as pure substance in a pharmacy, and the father was administered 400 to 500 mg. Carbachol concentrations in serum and urine on day 1 and 2 of hospital admission were analysed by HPLC-mass spectrometry. (...)

(PMID: 16740173)

(b) Specific location

Concurrent administration of a TNF antagonist with ORENCIA has been associated with an increased risk of serious infections and no significant additional efficacy over use of the TNF antagonists alone. (...)

(DrugBank: Abatacept)

Figure 1.5: Examples of relations with specific and unspecific location.

Figure 1.6 are examples of `Promotes` - a directed relation and `Associated` - an undirected relation taken from Phenebank corpus. In the directed relation, the order of entities in the relation annotation should be considered, vice versa, in the undirected relation, two entities have the same role

(a) Directed relation

Some patients carrying mutations in either the ATP6V0A4 or the ATP6V1B1 gene also suffer from hearing impairment of variable degree.

(PMC3491836)

Directed relations:
ATP6V0A4 `Promotes` **hearing impairment**
ATP6V1B1 `Promotes` **hearing impairment**

(b) Undirected relation

Finally, new insight into related musculoskeletal complications (such as myopathy and tendinopathy) has also been gained through the (...)

(PMC4432922)

Undirected relations:
musculoskeletal complications `Associated` **myopathy**
musculoskeletal complications `Associated` **tendinopathy**

Figure 1.6: Examples of directed and undirected relation from Phenebank corpus.

1.3 Difficulties and Challenges

Relation Extraction is one of the most challenging problem in Natural Language Processing. There exists plenty of difficulties and challenges, from basic issue of natural language to its various specific issues as below:

- **Lexical ambiguity**: Due to multi-definitions of a single word, we need to specify some criteria for system to distinguish the proper meaning at the early phase of analyzing. For instance, in "*Time flies like an arrow*", the first three word "*time*", "*flies*" and "*like*" have different roles and meaning, they can all be the main verb, "*time*" can also be a noun, and "*like*" could be considered as a preposition.
- **Syntactic ambiguity**: A popular kind of structural ambiguity is modifier placement. Consider this sentence: "*John saw the woman in the park with a telescope*". There are two preposition phases in the example, "*in the park*" and "*with the telescope*". They can modify either "*saw*" or "*woman*". Moreover, they can also modify the first noun "*park*". Another difficulty is about **negation**. Negation is a popular issue in language understanding because it can change the nature of a whole clause or sentence.

- **Semantic ambiguity**: Relations can be hidden in phrases or clauses. However, a relation can be encoded at many lexico-syntactic levels with many form of representations. For example: "*tea*" and "*cup*" has a relationship `Content-Container`, but it can be encoded in three different ways N1 N2 (tea cup), N2 prep N1 (cup of tea), N1's N2 (*tea's cup). Vice versa, one pattern of representation can perform different relations. For instance: "*Spoon handle*" presents the `whole-part` relation, and "*bread knife*" presents the `functional` relations, although they have the same representation by one noun phrase.

- **Semantic relation discovery may be knowledge intensive**: In order to extract relations, it is preferable to have a large enough knowledge domain. However, building big knowledge database could be costly. We could easily find out that "GM car" is a `product-producer` relation if we have good knowledge, instead of misunderstanding it as a feature of a random car brand.

- **Imbalanced data**: is considered as an extremely serious classification issue, in which we can expect poor accuracy for minor classes. Generally, only positive instances are annotated in most relation extraction corpora, so negative instances must be generated automatically by pairing all the entities appearing in the same sentence that have not been annotated as positives yet. Because of a big number in such entities, the number of possible negatives pairs is huge.

- **Low pre-processing performance**: Information extraction usually gets errors, which are consequences of relatively low performance of pre-processing steps. NER and relation classification require multiple pre-processing steps, including sentence segmentation, tokenization, abbreviation resolution, entity normalization, parsing and co-reference resolution. Every step has its own effect to the overall performance of relation extraction system. These pre-processing steps need to be based on the current information extraction framework.

- **Relation direction**: We not only need to detect the relations between two nominals, but also need to determine which nominal is argument and which one is predicate. Moreover, in the same dataset (for example: in Figure 1.6 as mentioned before), the relation could be either directional or unidirectional. It is hard for machines to distinguish which context is unidirectional, which context is directional, and it is in which directions?

- **Multitude of possible relation types**: The task of Relation Extraction is applied in various domain from general, scientific to biomedical domain. Many datasets are proposed to evaluate the quality of Relation Extraction system, such as SemEval 2010 Tack 8 [30], BioCreative V Track 3 CDR [65], SemEval 2017 ScienceIE [4], etc. In any dataset, relations have different ways to represent (as examples in Figure 1.2 and Figure 1.3).

- **Context dependent relation**: One of the toughest challenges in Relation Extraction is that the relation is not simply presented in one single sentence. To detect the relation, we need to understand of the sentence and entities context. For example, in the sentence in Figure 1.4-(a), it is a cross-sentence relation, two entities are in two separate sentences.

There are many other difficulties in applying in various domains. For example, in relation extraction from biomedical literature:

- **Out-Of-Vocabulary (OOV)**: there are an extreme use of unknown words in biomedical literature such as acronyms, abbreviations, or words containing hyphens, digits, and Greek letters. These unknown words not only cause ambiguities, but also lead to many errors in pre-processing steps, i.e., tokenization, segmentation, parsing, etc.
- **Lack of training data**: In general NLP problems, it is possible to download training dataset for machine learning model online with good quality and quantity. However, data for biomedical is quite little. In addition, it is time and money consuming for labeling because it requires special experts with domain knowledge.
- **Domain specific data**: In general NLP problems, the data is familiar and similar to daily conversation, but in biomedical domain, data consists of uncommon terms and they appear maybe only once or several times in the whole corpus. It leads to mistakes in calculating distribution probabilities or connections between these terms. There are a lot of differences between detecting entities names in medicines or diseases and detecting ordinary entities such as a person's name or location. In fact, the name of a chemical can be super long (such as: "*N-[4-(5-nitro-2-furyl)-2-thiazolyl]-formamide*"), or different names for one chemical, such as: "*10-Ethyl-5-methyl-5,10-dideazaaminopterin*" and "*10-EMDDA*". However, none of current approaches can solve these problems. Furthermore, while normal entities usually come with a capital first letter for easier detection, entities in diseases and chemicals usually do not have this rule in common documents, for example: *nephrolithiasis* disease, *triamterene* medicine. Therefore, special approaches are required to archive good result.

1.4 Common Approaches

The research history of RE has witnessed the development as well as the competition of a variety of RE methodologies. Several studies make use of the dependency tree and the **Shortest Dependency Path** (SDP) between two nominals to take advantage of the syntactic information. Other conventional approaches are based on the **entire word sequence** of the sentence to obtain semantic information, sequence features, and local features. All of them are proven to be effective and have different strengths by leveraging different types of linguistic knowledge, however, also suffer from their own limitations.

Many deep neural network (DNN) architectures are introduced to learn a robust feature set from unstructured data [60], which have been proved effective, but, often suffer from irrelevant information, especially when the distance between two entities is too long. Another study to extract the relation between two entities is using whole sentence in which both are mentioned. This approach seems to be slightly weaker than using the SDP since not all words in a sentence contribute equally to classify relations and this leads to unexpected noises [49]. However, the emergence and development of attention mechanism [6] has re-vitalized this approach. For RE, the attention mechanism is capable of picking out the relevant words concerning target entities/relations, and then we can find critical words which determine primary useful semantic information [63, 77]. We therefore need to determine the object of attention, i.e., nominals themselves, their entity types or relation label. However, conventional attention mechanism on sequence of words cannot make use of structural information on dependency tree. Moreover, it is hard for machines to learn the attention weights from a long sequence of input text.

Some early studies stated that the shortest dependency path (SDP) in dependency tree is usually concise and contains essential information for RE [12, 22]. Many other researches have also illustrated the effectiveness of the shortest dependency path between entities for relation extraction [18]. By 2016, this approach became dominant with many studies demonstrating that using SDP brings better experimental results than previous approaches that used the whole sentence [14, 39, 47, 67, 68]. However, using the SDP may lead to the omission of useful information (i.e., negation, adverbs, prepositions, etc.). Recognizing this disadvantage, some studies have sought to improve SDP approaches, such as adding the information from the sub-tree attached to each node in the SDP [44] or applying a graph convolution over pruned dependency trees [74]. The detail and overview of related work will be stated in Section 2.

1.5 Contributions and Structure of the Thesis

Up to now, enriching word representation is still attracting the interest of the research community; in most cases, sophisticated design is required [39]. Meanwhile, the problem of representing the dependency between words is still an open problem. In our knowledge, most previous researches often used a simple way to represent them, or even ignore them in the SDP [67]. Considering these problems as motivation to improve, in this paper, we present a compositional embedding that takes advantage of several dominant linguistic and architectural features. These compositional embedding then are processed within a dependency unit manner to represent the SDPs.

In this work, we focus on condensed semantic and syntactic information on the SDP. Compensating for the limitations of the SDP may still lead to missing information so we enhance this with syntactic information from the full dependency parse tree. Our idea is based on fundamental notion that the syntactic structure of a sentence consists of binary asymmetrical relations between words. Since these dependency relations hold between a head word (parent, predicate) and a dependent word (children, argument), we try to use all child nodes of a word in the dependency tree to augment its information. Depending on a specific set of relations, it will turn out that not all children are useful to enhance the parent node; we select relevant children by applying several attention mechanisms with kernel filters.

The main contributions of our work can be concluded as:

- We introduce a enriched representation of SDP that utilizes a major part of linguistic and architectural features by using compositional embedding and investigated the effectiveness of dependency tree normalizing before generating the SDP.
- We proposed a novel representation of relation based on attentive augmented SDP that overcomes the disadvantages of traditional SDP and improved the attention mechanism with kernel filters to capture the features from context vectors.
- We proposed an advanced DNN architecture that utilizes the proposed Richer-but-Smarter Shortest Dependency Path (RbSP) and showed that CNN model is effective and adaptable in extracting semantic relations for different types of data without any architecture change.
- We also investigated the contributions of model components and features to the final performance that provide a useful insight into some aspects of our approach for future research.

My thesis includes four main Chapters and one Conclusions, as follow:

Chapter 1: Introduction. This Chapter is an introduction to Relation Extraction problem, the overview of RE system and some examples from different datasets. We present the motivations and the difficulties and challenges of Relation Extraction as well.

Chapter 2: Related Work. We introduces relevant related work shared among all the methods in this thesis. This chapter introduces the history and development of Relation Extraction research from traditional Rule-based approaches to advanced Statistic-based methods, including Supervised methods, Unsupervised methods, Distant and Semi-supervised methods and Hybrid Approaches, Joint Extraction. We mainly focus on two categories of supervised approaches: Feature-based Machine Learning and Deep Learning methods.

Chapter 3: Materials and Methods. Chapter 3 begins by providing an overview of our novel Richer-but-Smarter Shortest Dependency Path representation of a sentence. Next, we will introduce how we use the Deep Neural Network to exploit the relation between two nominals using RbSP representation. Furthermore, we present the Multilayer attention with Kernel filters architecture to extract augmented information from children nodes. Finally, we conclude the chapter by providing a brief introduction to how we improve our model's performance by several techniques.

Chapter 4: Experiments and Results. We provide an insight to the implementation of the models and discuss about the hyper-parameter settings. Next, we evaluate our model on two datasets in different domains. The method introduced in Chapter 3 substantially outperforms prior methods for extracting relation. Furthermore, provide an investigation on the contribution of each proposed components. Finally, we analyze the output and the error for better insight into our models.

Conclusions. This Chapter concludes the thesis by summarizing the important contributions and results. Also, we highlight the limitations of our models and point out some further extensions in the future work.

Chapter 2

Related Work

Since RE serves as an intermediate step in a variety of natural language processing applications, especially in knowledge extraction from unstructured texts, it has been widely studied in the NLP community for decades. In this chapter, we will discuss on some mainstream RE approaches. We categorize approaches to relation extraction as two main categories: Rule-Based Approaches (Section 2.1) and Statistic-Based Approaches. The Statistic-Based Approaches can further classified into several logical categories: (i) supervised techniques (Section 2.2), (ii) unsupervised nethods (Section 2.3), and (iii) distant supervision based and semi-supervised techniques (Section 2.4). Finally, we conclude in the Section 2.5 by discussing special class of techniques which are combination of previous techniques.

2.1 Rule-Based Approaches

One of the most fundamental approaches for RE is based on rules. Rule-based approaches need to generalize the structure of the entity mentions by pre-defined rules or patterns. Because there is always a requirement that rules builder needs to deeply understand the field background and characteristics, the big demand of human activity and low portability are the main difficulties of this approaches.

The simplest approaches for detecting potential relationships is co-occurrence statistic [51]. Based on the hypothesis that if two entities are frequently mentioned together, it is likely that they are somehow related, this method reveals binary relationship through counting their co-existence in single sentences. Examples of relation extraction researches that used co-occurrence approach for biomedical data includes [17, 41].

More accurate alternatives are based on manual-crafted rules and patterns to perform information extraction task. The previous study of Hearst [28] used this technique for identifying relation `hyponym-of` and archived a performance of 66% accuracy. More recently, the SemRep approach of Rosemblat et al. [57], which follows many of such rules, achieved the result of 0.69 Precision and 0.88 Recall for the task of relation extraction between medical entities [55]. One of the most recent research that based on pattern-based approach in our knowledge is a system called iXtractR [52], it is a generalizable NLP framework that uses some novel designates to develop the patterns for biomedical relation extraction.

These methods do not require any annotated data to train a system but typically meet two disadvantages (i) the dependence on manually-crafted rules, which are time consuming and often require domain experts knowledge. (ii) they are limited at extracting specific relation types.

2.2 Supervised Methods

The unsupervised [27, 53, 69], semi-supervised [7, 11, 21, 21] and distant supervision [38, 63] methods have been proven effective for the task of detecting relation from unstructured text. However, in this paper, we mainly focus on supervised approaches, which usually have higher accuracy. Generally, these methods can be divided into two categories: feature engineering-based methods and deep learning-based methods.

2.2.1 Feature-Based Machine Learning

In earlier RE studies, researchers focused on extracting various kinds of features representing each annotated data instance, i.e., each sample data is presented as a feature vector f $= f_1, f_2, ..., f_n$ in an n-dimensional space, in which f_i is the extracted features that follow a pre-defined feature set. This feature set are designed by domain experts. For relation extraction task, sentences or paragraphs that contain the target entities are used to construct feature vector through feature extraction process. Various feature types have been proposed to use, the commonly used feature for are categorized into three types as described below.

- **Lexical Features**: In this feature set, lexical features such as position of mentioned pair of entities, number of words between mentioned pair, word before or after mentioned pair, etc. are used to capture context of the sentence. With this, bag-of-words approach can be useful to represent represent sentence and words as a feature in our feature vector.

- **Syntactic Features**: In this feature set, there are two types of tree that are commonly used: Syntax Tree and Dependency Tree. The grammatical structure of the sentence and mentioned pair are used for feature creation. For example, part of speech tags for each mentioned pair, chunk head, etc. With this we can also use dependency tree path path between mentioned pair, path labels, distance between mentioned pair in dependency tree, etc., in our feature set.

- **Entity Features**: A relation can exist between certain type of entities, for example `CID` relations can only exist between a chemical and a disease. So, type of mentioned pair of entities are also important feature values for classification purpose. Entity features also includes the content of mentioned pair.

Kambhatla [36] gave the first report on the use of this feature-based approach to the ACE relation task. This paper employs Maximum Entropy models to combine a large number of features, including lexical features, syntactic features, and semantic features derived from the text. On the other hand, Zhao and Grishman [75] and GuoDong et al. [26] use SVMs trained on these features using polynomial and linear kernels respectively for classifying different types of entity relations. Studies of Le et al. [38] and Rink and Harabagiu [56] tried adapting this approach to other domain by using variety of handcrafted features that capture the semantic and syntactic information and then feed to an SVM classifier to extract the relations of the nominals from biomedical text.

To improve the performance of feature-based system, proposing and taking advantages of multiple richer feature types is a widely used strategy. Example of such additional rich features includes dictionary features that obtained using external resources or other domain knowledge [16, 19], co-occurrence word-pairs [3], richer linguistic information extracted from parse trees [46], statistical features, document features, distributional features, and even word embedding [65].

However, all the feature-based methods depend strongly on the quality of the designed features from an explicit linguistic pre-processing step. Since NLP applications in general and relation extraction in particular involve structured representations of the input data, it can be difficult to arrive at an optimal subset of relevant features. Therefor, these methods suffer from the problem of selecting a suitable feature set for each particular data.

2.2.2 Deep Learning Methods

In the last decade, deep learning methods have made significant improvement and produced the state-of-the-art result in relation extraction. The introduction of Word embeddings [9, 48], which are a form of distributional representations for the words in a low dimensional space, has made an evolutionary of deep learning. These methods usually utilize the word embeddings with various DNN architectures to learn the features without prior knowledge.

Approaches using whole sentence with position feature:

One of the very first studies is of Socher et al. [60] which combines matrix-vector representations with a recursive neural network to learn compositional vector representations for phrases and sentences from a syntactically plausible parse tree. The vector captures the inherent meaning of the constituent, while the matrix captures how it changes the meaning of neighboring words or phrases. The model obtains promising results on the task of classifying semantic relationships between nouns in a sentence. Some other studies use all words in sentence with position feature to extract the relations. For example, Zeng et al. [70] proposed position features to specify the pairs of nominals that are expected to assign relation labels and integrate this feature to a convolutional deep neural network for relation classification. Nguyen and Grishman [49] also used position embeddings for encoding relative distances in a convolutional neural network with multiple window sizes of filters.

Many studies also tried applying this model to other types of data. Panyam et al. [50] exploited the whole dependency graph for relation extraction in biomedical text. Study of Zhou et al. [76] presented an ensemble model using DNN with syntactic and semantic information. Gu et al. [25] extracted `CID` relations by using contextual of whole sentence with a Maximum Entropy (ME) model and a convolutional neural network model.

Approaches using Shortest Dependency Path:

In another view, the study of Bunescu and Mooney [12] provided an essential insight into identifying the relation between two entities on dependency parse tree. In recent years, many studies attempt other possibilities by using the concentrated information on the shortest dependency path. Convolutional neural network models (Xu et al. [67]) are among the earliest approaches applied on the SDP. On the same year, Xu et al. [68] rebuilt an Recurrent Neural Network (RNN) with Long Short-Term Memory

(LSTM) unit on the dependency path between two marked entities to utilize sequential information of sentences.

Various of improvements have been suggested to boost the performance of RE models on SDP, such as negative sampling [67], modeling the directed shortest path [14], voting schema and combining several deep neural networks [39], etc. Notice the lack of supported information on the SDP, study of Liu et al. [44] suggested incorporating additional network architectures to further improve the performance of SDP-based methods, which is using a recursive neural network to model the sub-tree.

Approaches using attention mechanism:

Recently, with the introduction and development of attention mechanism, many works tend to use the whole sentence or paragraph and focus on the most relevant information using attention technique. Some studies apply a single attention layer, that focus on word itself [59, 72]; word position [73] and global relation embedding [62]. The others apply several attention layers, such as word, relation and pooling attentions [64]; multi-head attentions [63]; word and entity based attentions [34].

The study of Shen and Huang [59] solved the semantic relation extraction task by an attention-based convolutional neural network architecture that makes full use of word embedding, part-of-speech tag embedding and position embedding information. Word level attention mechanism is used to determine which parts of the sentence are most influential with respect to the two entities of interest.

Wang et al. [64] further improved attention mechanism with a new form that is applied at two different levels to capture both entity-specific attention and relation-specific pooling attention. This architecture allows it to detect more subtle cues despite the heterogeneous structure of the input sentences, enabling it to automatically learn which parts are relevant for a given classification. Zhang et al. [71] used a bidirectional Long Short-Term Memory architecture with an attention layer and a tensor layer for organizing the context information and detecting the connections between two nominals.

In another study, Verga et al. [63] tried to predict Chemical-Induced Disease relation from entire biomedical paper abstracts using a multi-head attention architecture. This model out-performed the previous state-of-the-art on the BioCreative V CDR dataset despite using no additional linguistic resources or mention pair-specific features.

2.3 Unsupervised Methods

In this section, we discuss some of the important unsupervised RE approaches which do not require any labelled data. Without demand of annotated data, unsupervised methods can use very large amounts of data as well as extract very large amount of relations. However, since there is no standard form of relations, the output resulting may not be easy to map to relations which is necessary for a particular knowledge base.

One of earliest approaches for completely unsupervised Relation Extraction was proposed by Hasegawa et al. [27]. They only require a NER tagger to identify named entities in the text so that the system focuses only on those named entity mentions. The approach can be summarized in three steps:

- **Named Entity pairs and context**: Pairs of all such co-occurring named entities, that there are at most N intermediate words in between them, are formed. The words occurring to the left of first NE and the words occurring to the right of second NE are not considered to be part of the context.
- **Context similarity computation**: For each NE pair, a word vector is formed using all the words occurring in its context where each word is weighted by TF-IDF. The similarity of the contexts of two NE pairs is computed as the Cosine Similarity between their word vectors.
- **Clustering and Labelling**: Using the similarity values, the NE pairs are clustered using hierarchical clustering with complete linkage. The resultant clusters are also labelled automatically using the high frequency words in the contexts of all the NE pairs in the cluster.

Another similar approach for unsupervised RE from Wikipedia texts, was proposed by Yan et al. [69]. Another interesting line of research, is based on inducing relation types by generalizing dependency paths [43]. One of the major non clustering based approach for unsupervised relation extraction is the URES (Unsupervised RE System) by Rosenfeld and Feldman [58]. The only input required by the URES system is the definitions of the relation types of interest. A relation type is defined as a small set of keywords indicative of that relation type and entity types of its arguments.

Unsupervised-machine learning methods also applied successfully to biomedical relation extraction problem. For example, Quan et al. [53] proposed an unsupervised method based on pattern clustering and sentence parsing to deal with biomedical relation extraction.

2.4 Distant and Semi-Supervised Methods

Supervised machine learning methods relied on the availability of handcrafted annotated data, which are often expensive and time-consuming to obtain, especially in the biomedical domain. Therefore, alternatives such that learning from both labeled and unlabeled data have been proposed. These approaches for relation classification can be categorized as: Semi-supervised learning methods and Distant-supervision methods.

Semi-supervised learning methods use a small set of annotated relations as the training "seed" to iteratively extract new relation instances. The semi-supervised learning methods for relation extraction often use the idea of pattern-based method. I.e., they try to derive patterns from the textual contexts of "seed", then use these patterns to detect more relations. Finding new patterns and predicting new relations are processed alternately and repeatedly in the iterative architecture. This was first introduced in DIPRE [11], and then extended in Snowball [2], KnowItAll [21] and TextRunner [7].

Distant-supervision methods (weak supervision methods) typically makes use of weakly labeled data derived from a knowledge base of known relationship to automatically collect large amounts of training data from unlabeled data. A supervised classifier then be used with the collected training data. Therefore, it often does not require any labeled data. This approach has attracted some attention in the research community since it can take advantages of available resources in a very in a flexible manner.

Distant Supervision is widely used in biomedical data due to the lack of training data. An example includes study of Le et al. [38] that used silver corpus derived form the Comparative Toxicogenomics Database[1] (CTD) to improve the chemical-induced disease relation extraction system.

Crowd-sourcing is demonstrated as an inexpensive, fast and practical approach for collecting high-quality annotations for different biomedical NLP tasks. This approach is used in some biomedical relation extraction system [13, 42].

2.5 Hybrid Approaches

Hybrid approaches which combine different techniques into a single model, have also been successfully applied to many relation classification system. One of the most recent

[1] `http://ctdbase.org/`: CTD is open site and scientific data management research tool that describes the relationship between chemicals, diseases, proteins, genes, phenotypes, etc.

hybrid model is proposed in the research of Le et al. [39]. Authors proposed a "Man for All Seasons" model that the combination of multi-channel BiLSTM with two directional Convolutional Neural Networks. MASS model is capable of adapting to many domains from general data, scientific data to biomedical data. They also applied many post-processing rules to have better results. MASS model achieved state-of-the-art result on SemEval 2017 ScienceIE dataset by using regular expression `hyponym` and `synonym` rules.

The architecture of hybrid models for relation classification are very abundant, such as combining rule-based model with SVM classifiers in the study of Wei et al. [66]. Many other researches focus on integrating pattern recognition into supervised machine learning. This method is used on research of Abacha and Zweigenbaum [1] to extract semantic relations from MEDLINE abstracts or Javed et al. [35] to investigate Drug-Drug interaction. Another attempt to combine finite state automata and random forest using weighted fusion is of Mavropoulos et al. [45].

To strengthen the performance of Deep Neural Network, Cai et al. [14] and Zhang et al. [72] combine several deep learning model. Gu et al. [25] and Zhou et al. [76] tried other possibilities by incorporate CNN with maximum entropy model or LSTM with SVM model respectively.

Chapter 3

Materials and Methods

In this chapter, we will discuss on the materials and methods this thesis is focused on. Firstly, **Section 3.1** will provide an overall picture of theoretical basis, including distributed representation, convolutional neural network, long short-term memory, and attention mechanism. Secondly, in **Section 3.2**, we will introduce the overview of our relation classification system. **Section 3.3** is about materials and techniques that I proposed to model input sentences to extract relations. The proposed materials include dependency parse tree (or dependency tree) and dependency tree normalization; shortest dependency path (SDP) and dependency unit. I further present a novel representation of a sentence; namely Richer-but-Smarter Shortest Dependency Path (RbSP); that overcome the disadvantages of traditional SDP and take advantages of other useful information on dependency tree. Subsequently, in **Section 3.4**, we will introduce the design of our multi-layer attention architecture with kernel filters to extract the SDP's augmented information from the Richer-but-Smarter Shortest Dependency Path. Finally, in **Section 3.5** we will describe how we use the deep neural networks to explore the semantic relation between two nominals.

3.1 Theoretical Basis

In recent years, deep learning has been extensively studied in natural language processing, a large number of related materials have emerged. In this section, we briefly review some theoretical basis that are used in our model: distributed representation (Sub-section 3.1.1), convolutional neural network (Sub-section 3.1.2), long short-term memory (Sub-section 3.1.3), and attention mechanism (Sub-section 3.1.4).

3.1.1 Distributed Representation

The input of NLP task is usually a sequence of words that cannot be processed by deep learning models directly. One of the very first ideas to represent input words is using one-hot vector. However, this approach seems to be ineffective because of large vocabulary size (millions or tens of millions words). This resulted in the motivation to learn distributed representations of words in low-dimensional spaces [8].

3.1.1.1 Word-level Context-based Embeddings

Word-level embeddings are distributional vectors that follow the distributional hypothesis, which similar words tend to occur in the same context. The main advantage of distributed representation is that it encapsulates similarities between words. Word-level embedding is typically pre-trained through a large unlabeled corpus by solving a "fake" task, such as predicting a word based in context, where the learned word vectors can capture general semantic and syntactic information.

Word2Vec is one of the most popular methods for learning word embedding with a shallow neural network [48]. Representations of words can be obtained using two methods, Skip-gram and Continuous Bag of Words (CBOW). In view of k context words surrounding the target word, CBOW computes its conditional probability. On the other hand, the skip-gram model predicts the context words around the main target word. Both of them have their own advantages and disadvantages. Skip-gram works very well with small quantities of data, in which rare words are well represented [48].

3.1.1.2 Character-level Context-based Embeddings

Word-level embeddings are capable of capturing semantic and syntactic information. However, for many tasks, sub-word information like shape or word morphological can also be very useful. The unknown term or out-of-vocabulary word (OOV) is a common phenomenon for languages with large vocabulary. Naturally, character embedding deals with it as every word is only considered a composition of individual characters. The study of Bojanowski et al. [9] attempted to improve word representation by using rich morphological information at character level. They utilized the skip-gram model to learn the words representations using as bag-of-characters n-grams. Their work was thus effective in addressing certain persistent word embedding issues in conjunction with the skip-gram models.

3.1.2 Convolutional Neural Network

By using distributed representation, a sequence of words is turned into a sequence of word embeddings in a distributed space. There was a need for an effective feature extraction method which extracts higher-level features from the constituent words or n-grams. These abstract features would be applied in numerous NLP tasks such as machine translation, sentiment analysis, text summarization, question answering and relation classification. Convolutional Neural Network turned out to be the one of the most effective methods to capture local high-level features. CNNs have proved to be effective and produced state-of-the-art results in many tasks of computer vision. In following section, we will describe how CNNs are applied to NLP tasks.

3.1.2.1 Sequence Modeling

Figure 3.1 depicts the overall architecture of CNN that used to model a text sequence. Given a sequence has n words[1], we use a distributed representation method to represent the ith word as an embedding vector $\mathbf{w}_i \in \mathbb{R}^d$ where d is the dimension of the embedding. The input sequence can now be represented as an embedding matrix.

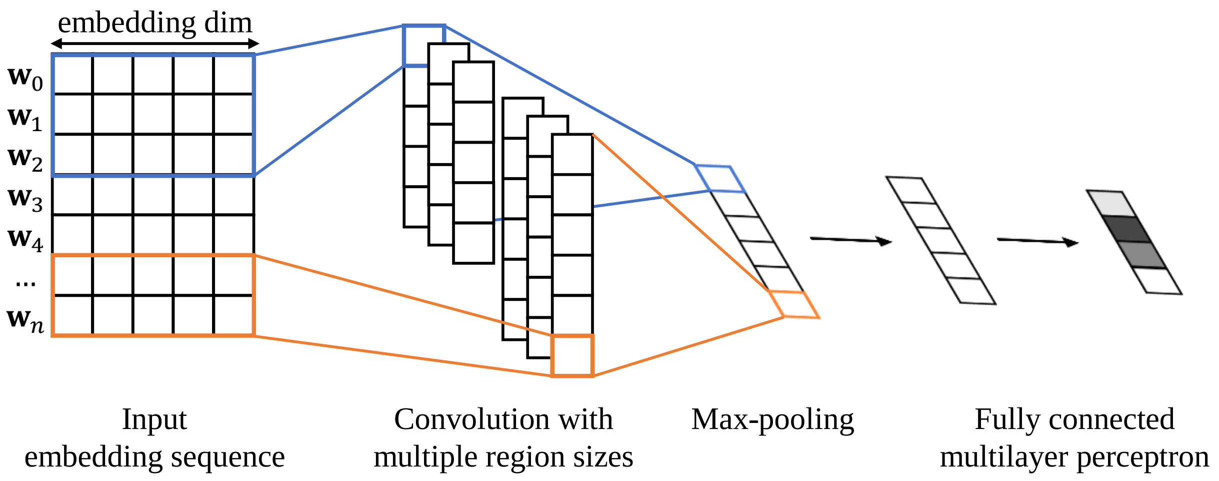


Figure 3.1: Sentence modeling using Convolutional Neural Network.

Here we define a slice $\mathbf{w}_{i:i+j}$ as the concatenation of j consecutive word vectors from i to $i+j-1$ as follow:

$$\mathbf{w}_{i:i+j} = \mathbf{w}_i \oplus \mathbf{w}_{i+1} \oplus ... \oplus \mathbf{w}_{i+j-1} \tag{3.1}$$

[1]words can be replaced by phrases, tokens, characters, n-grams, etc.

Convolution operation is performed on this input embedding slice. A filter $\mathbf{f} \in \mathbb{R}^{kd}$ is applied to a window of k successive word vectors to capture a local feature. For example, a feature c_i is generated from the window of k words $\mathbf{w}_{i:i+k}$ as follow:

$$c_i = f(\mathbf{w}_{i:i+k}\mathbf{f} + b) \tag{3.2}$$

where $b \in \mathbb{R}$ is the bias term and f is a non-linear activation function. The common used activation functions are hyperbolic tangent and ReLU. This operation is applied to all possible windows using the same weights to produce convolved feature map. I.e.,

$$\mathbf{c} = [c_1, c_2, c_3, ..., c_{n-h+1}] \tag{3.3}$$

We then gather the most important feature by a pooling layer. A max-pooling layer usually follows a convolution layer, which usually sub-samples the input using a max operation on each filter $\hat{c} = \max(\mathbf{c})$. There are two main reasons for this strategy:

- Max pooling always maps the input to a certain output dimension, regardless of the filter size that offers a fixed-length output, generally necessary for fully connected dense.
- The output dimension is reduced while maintaining the most important n-gram characteristics features across the whole sentence.

In a CNN model, a certain number of convolutional filters, also known as kernels, of various widths, slide through the whole word embedding matrix to capture different pattern of n-gram.

3.1.2.2 Convolutional Character Embeddings

In order to tackle with word-level embeddings problem, another possibility is using convolutional approach. As illustrated in Figure 3.2, convolutional model captures the local features on the successive characters and then combines them using a max pooling layer to produce the final fixed-sized character embedding vector of the word.

Given a n-character word w that is composed of $\{c_1, c_2, ..., c_n\}$, we first transform every character c_i into corresponding embedding $\mathbf{c}_i$ using a look-up table $\mathbf{W}^{char} \in \mathbb{R}^{\mathrm{dim}_{char} \times |V_{char}|}$. Here V_{char} is the set of all characters we used. This sequence of character embeddings is then used as input for the convolutional layer.

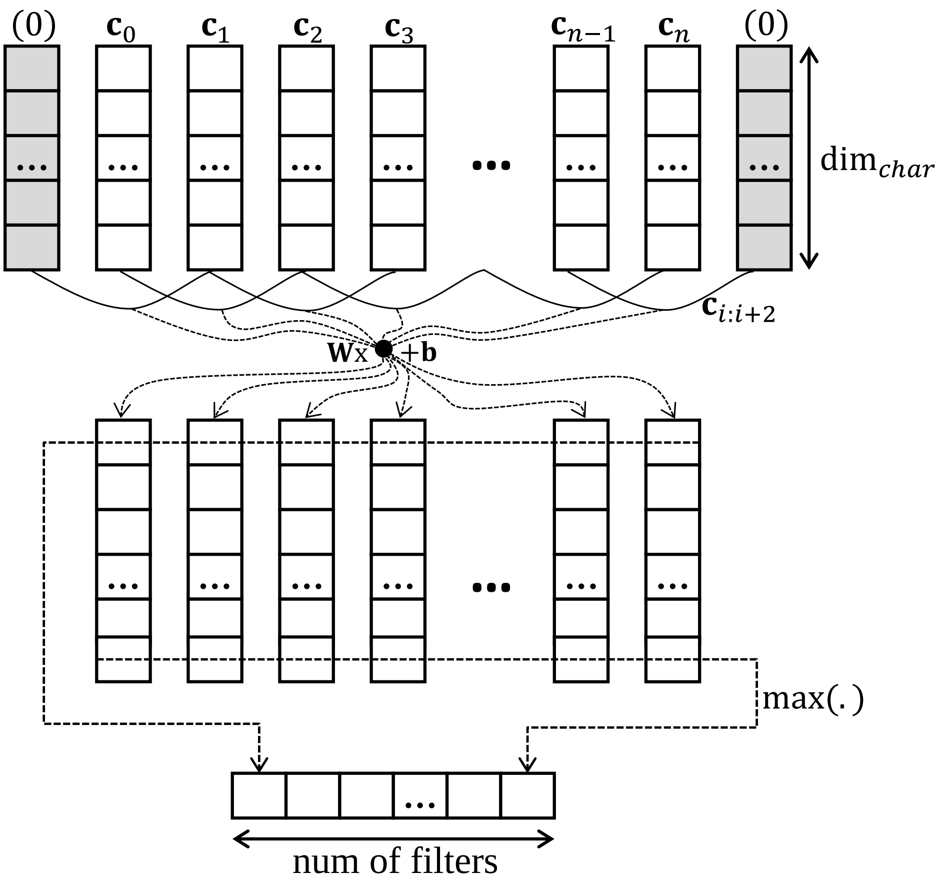


Figure 3.2: Convolutional approach to character-level feature extraction.

In general, let the vector $\mathbf{c}_{i:i+j}$ refer to the concatenation of $[\mathbf{c}_i, \mathbf{c}_{i+1}, ..., \mathbf{c}_{i+j}]$. A convolution operation with region size r applies a filter $\mathbf{w}^c \in \mathbb{R}^{r.\text{dim}_{char}}$ on a window of r successive characters to capture a local feature. We apply this filter to all possible window $[\mathbf{c}_{1:r}, \mathbf{c}_{2:r+1}, ..., \mathbf{c}_{n-r+1:n}]$ to produce convolved feature map. For example, a feature map $\mathbf{c}^r \in \mathbb{R}^{n-r+1}$ is generated from a word of n characters by:

$$\mathbf{c}^r = \left\{ \tanh(\mathbf{c}_{i:i+r-1}\mathbf{w}^c + \mathbf{b}^c) \right\}_{i=1}^{n-r+1} \tag{3.4}$$

The most important feature is then gathered from the feature map, which have the highest values, by a max pooling [10] layer. This idea of pooling can naturally deal with variable sentence lengths since we take only the maximum value $\hat{c} = \max(\mathbf{c}^r)$ as the feature to this particular filter.

The described process extracts one feature from one filter with region r. To take advantage from wide ranges of n-gram features, multiple filters with varying region sizes $(1-3)$ are used to obtain a convolutional character embedding $\mathbf{e}^c$.

3.1.3 Long Short-Term Memory

3.1.3.1 Simple Recurrent Neural Networks

CNN model are capable of capturing local features on the sequence of input words. However, the long-term dependencies play the vital role in many NLP tasks. The most dominant approach to learn the long-term dependencies is Recurrent Neural Network (RNN). The term "recurrent" applies as each token of the sequence is processed in the same manner and every step depends on the previous calculations and results. This feedback loop distinguishes recurrent networks from feed-forward networks, which ingest their own outputs as their input moment after moment. Recurrent networks are often said to have "memory" since the input sequence has information itself and recurrent networks can use it to perform tasks that feed-forward networks cannot.

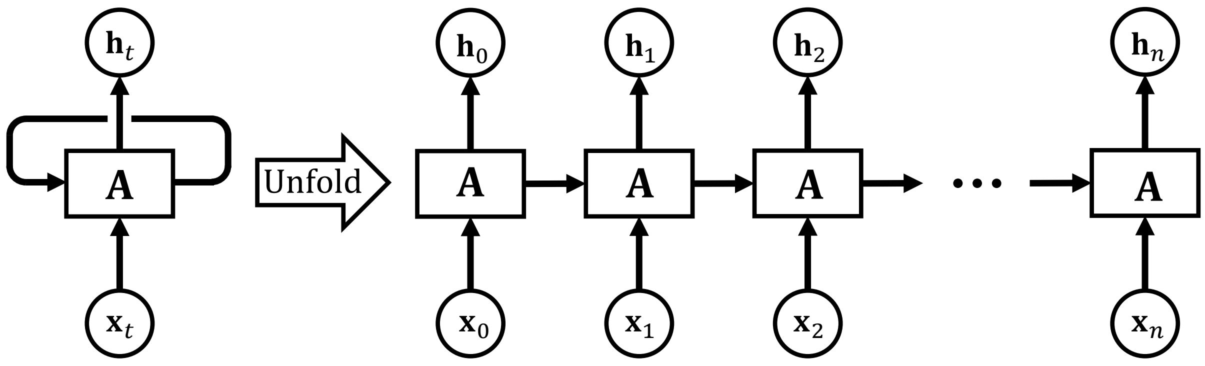


Figure 3.3: Traditional Recurrent Neural Network.

Figure 3.3 illustrates a recurrent neural network and the unfolding across time in its forward computation. This chain-like nature shows that recurrent neural networks closely associated with sequences and lists. The hidden state $\mathbf{h}_t$ at time step t is calculated based on input at the same time step $\mathbf{x}_t$ and hidden state of the previous time step $\mathbf{h}_{t-1}$ as follow:

$$\mathbf{h}_t = f\left(\mathbf{x}_t\mathbf{W} + \mathbf{h}_{t-1}\mathbf{U} + \mathbf{b}\right) \tag{3.5}$$

where $\mathbf{W}$, $\mathbf{U}$, and $\mathbf{b}$ account for weights and bias that are shared across time; the function f is taken to be a non-linear activation such as ReLU, $\mathrm{sigmoid}$, or $\tanh$. The weight matrices are filters that determine how significant the current input and the past hidden state are. The error that they generate will be returned through back-propagation and used to update their weight during the training phase.

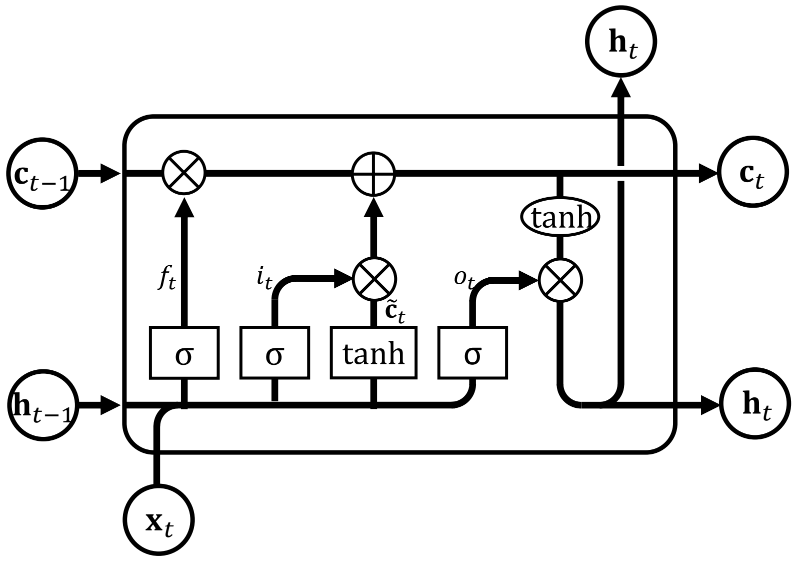


Figure 3.4: Architecture of a Long Short-Term Memory unit.

3.1.3.2 Long Short-Term Memory Unit

In practice, the *vanishing gradient* and *exploding gradient* problem emerged as a major impediment to RNNs performance. Long Short-Term Memory (LSTM) networks are an extension for RNNs, which basically extends their memory. By adding three gates and a memory cell, LSTM (Figure 3.4) calculates the hidden state at time step t as below:

$$\mathbf{i}_t = \sigma\left(\mathbf{x}_t\mathbf{W}^i + \mathbf{h}_{t-1}\mathbf{U}^i + \mathbf{b}^i\right) \tag{3.6}$$

$$\mathbf{f}_t = \sigma\left(\mathbf{x}_t\mathbf{W}^f + \mathbf{h}_{t-1}\mathbf{U}^f + \mathbf{b}^f\right) \tag{3.7}$$

$$\tilde{\mathbf{c}}_t = \tanh\left(\mathbf{x}_t\mathbf{W}^c + \mathbf{h}_{t-1}\mathbf{U}^c + \mathbf{b}^c\right) \tag{3.8}$$

$$\mathbf{c}_t = \mathbf{i}_t \odot \tilde{\mathbf{c}}_t + \mathbf{f}_t \odot \mathbf{c}_{t-1} \tag{3.9}$$

$$\mathbf{o}_t = \sigma\left(\mathbf{x}_t\mathbf{W}^o + \mathbf{h}_{t-1}\mathbf{U}^o + \mathbf{b}^o\right) \tag{3.10}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh\left(\mathbf{c}_t\right) \tag{3.11}$$

In which, $\mathbf{W}^i$, $\mathbf{U}^i$, $\mathbf{W}^f$, $\mathbf{U}^f$, $\mathbf{W}^c$, $\mathbf{U}^c$, $\mathbf{W}^o$, and $\mathbf{U}^o$ are model's trainable parameters; $\mathbf{b}^i$, $\mathbf{b}^f$, $\mathbf{b}^c$, and $\mathbf{b}^o$ are bias terms; σ denotes the $\mathrm{sigmoid}$ function, and $\odot$ denotes the element-wise product.

To simplify the upcoming explanation, we encapsulate the output of bidirectional Long Short-Term Memory network on a sequence $\mathbf{S} = \{\mathbf{x}_i\}_{i=1}^n$ as follows:

$$\mathbf{H} = \overleftrightarrow{\mathrm{biLSTM}}(\mathbf{S}) = \{\mathbf{h}_i\}_{i=1}^n \tag{3.12}$$

3.1.4 Attention Mechanism

In recent years, attention emerge as one of the most influential ideas in the Deep Learning community. This mechanism is now used in several problems, such as image captioning, text summarization, etc. The mechanisms for attention in neural networks are relatively based on the human visual attention mechanism. Human visual attention is well-studied resulted in different models, all of them are essentially able to focus on the "high resolution" certain region of an image while the surrounding image is perceived in "low resolution" and the focus over time is adjusted.

Attention mechanism was originally developed using Seq2Seq models in connection with neural machine translation. Prior to the Attention Mechanism, translation is based on reading the whole sentence before condensing all information into a fixed-long vector. As the result, a sentence with hundreds of words represented by several words will surely lead to the loss of information or insufficient translation, etc. This problem is partly addressed by attention mechanism. The machine translator can perceive all the information contained alongside original sentence and then generate the proper word in accordance with the current word it works on and with the context. It can even allow translator to "zoom in or out" (focus on local or global features).

Attention is nevertheless not mysterious or complicated. It is simply a vector formulated by parameters and delicate math, often the outputs of dense layer using softmax function. It could also be plugged anywhere that is suitable, and potentially, the result may be enhanced.

This fancy mechanism, similar to the fundamental encoder-decoder architecture, plugs a context vector into the encoder-decoder gap. And it is quite simple to build context vector. First of all, to compare target and source state, we loop through all encoder states and generate scores for each state in encoders. Then softmax could be used to normalize all scores, resulting in the probability distribution based on target states. I.e.,

$$\alpha_{ts} = \frac{\exp\big(\text{score}(\mathbf{h}_t, \bar{\mathbf{h}}_s)\big)}{\sum_{s'=1}^{S} \exp\big(\text{score}(\mathbf{h}_t, \bar{\mathbf{h}}_{s'})\big)} \quad \text{[Attention weight]} \tag{3.13}$$

$$\mathbf{c}_t = \sum_s \alpha_{ts} \bar{\mathbf{h}}_s \quad \text{[Context vector]} \tag{3.14}$$

$$\mathbf{a}_t = f(\mathbf{c}_t, \mathbf{h}_t) = \tanh\left([\mathbf{c}_t; \mathbf{h}_t]\mathbf{W}^c\right) \quad \text{[Attention vector]} \tag{3.15}$$

3.2 Overview of Proposed System

We developed an end-to-end Relation Classification (RC) system that receive the raw input data and export the corresponding relations result. This system is a small framework that contains plug-and-play modules. The overall architecture of RC framework is illustrated in the Figure 3.5.

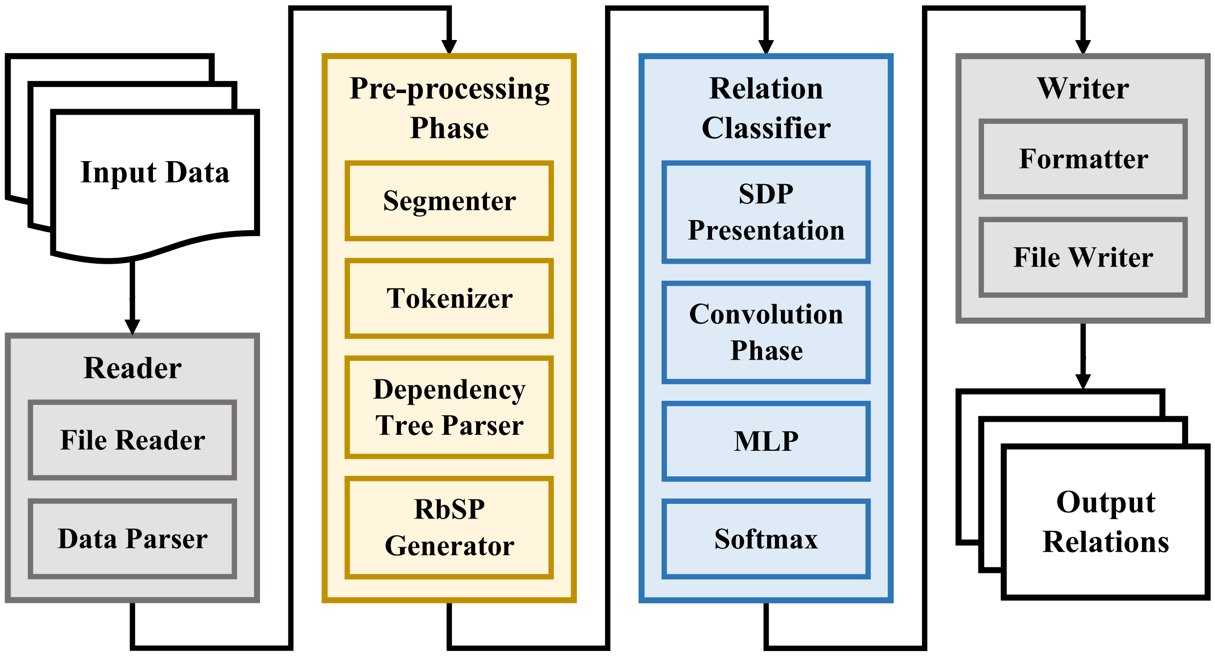


Figure 3.5: The overview of end-to-end Relation Classification system.

Proposed system comprises of three main components: IO-Module (Reader and Writer), Pre-processing module, and Relation Classifier. The Reader receives raw input data in many formats (*e.g.*, SemEval 2010 task 8 [29], BioCreative V CDR [65]) and parse them into an unified document format. These document objects are then passed to Pre-processing phase. In this phase, a document is segmented into sentences, and tokenized into tokens (or words). Sentences that contain at least two entities or nominals are processed by dependency parser to generate a dependency tree and a list of corresponding POS tags. A RbSP generator is followed to extract the Shortest Dependency Path and relevant information. In this work, we use spaCy[2] to segment documents, to tokenize sentences and to generate dependency trees. Subsequently, the SDP is classified by a deep neural network to predict a relation label from the pre-defined label set. The architecture of DNN model will be discussed in the following sections. Finally, output relations are converted to standard format and exported to output file.

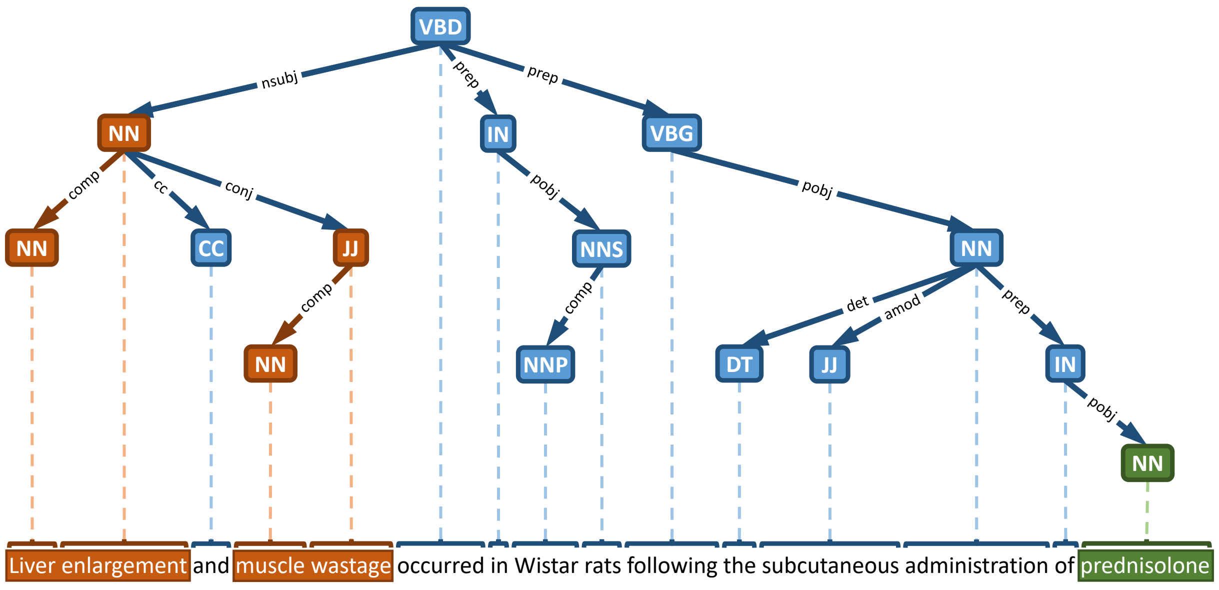
[2]spaCy: An industrial-strength NLP system in Python: `https://spacy.io`

3.3 Richer-but-Smarter Shortest Dependency Path

3.3.1 Dependency Tree and Dependency Tree Normalization

3.3.1.1 Dependency Tree

The dependency tree of a given sentence is a tree-structural representation, in which: Each token (or word) is represented as a node of the tree; Each token-token dependency is represented as a directed edge from parent node to a child node.



VBD
nsubj
prep
prep
NN
IN
VBG
comp
cc
conj
pobj
pobj
NN
CC
JJ
NNS
NN
comp
comp
det
amod
prep
NN
NNP
DT
JJ
IN
pobj
NN
Liver enlargement and muscle wastage occurred in Wistar rats following the subcutaneous administration of prednisolone

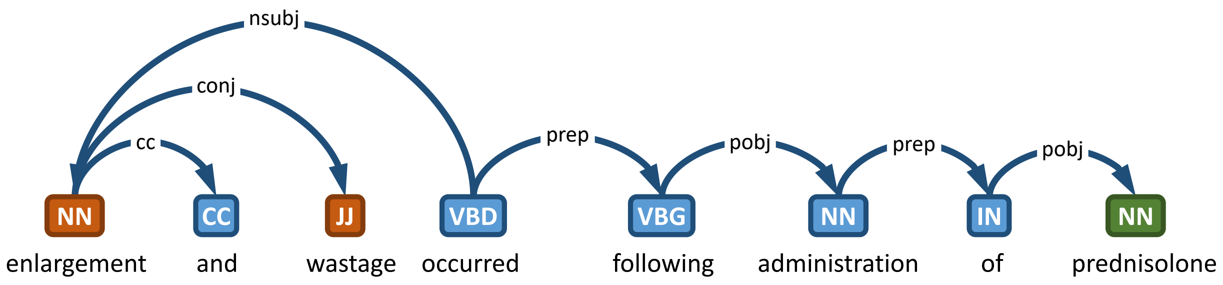
Notes: Diseases are annotated in orange (two first framed entities) and Chemical is annotated in green (last framed entity).

Figure 3.6: An example of dependency tree generated by spaCy.

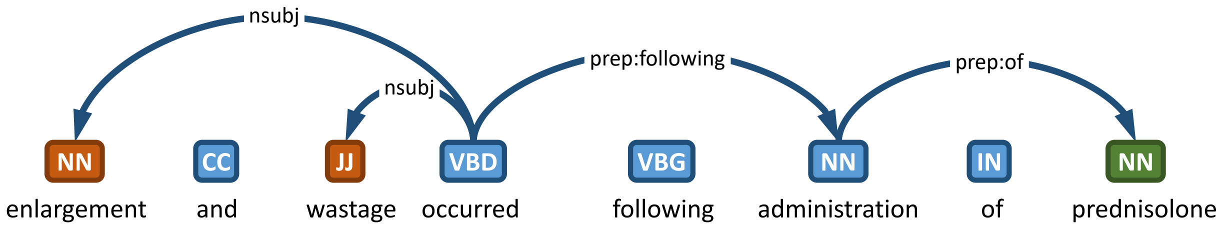
For example, Figure 3.6 shows the dependency parsing result of an example sentence from BioCreative V Track 3 CDR corpus (PMID:2425813). The labels inside node (*e.g.*, NN, JJ, CC, DT, etc.) are corresponding POS tags of token. Tags such as "*pobj*", "*amod*", "*nsubj*" are called the dependency relations between two tokens.

3.3.1.2 Dependency Tree Normalization

The dependency parsing of a sentence was designed to provide a brief depiction of the grammatical relationships in sentence that people without linguistic knowledge could easily understand and effectively use. The original dependence tree gives the full grammatical information of sentence, as well as that of the SDP, but some of this information might not help the relation extraction, even with noise.



(a) Subtree from original dependency tree.



(b) Subtree from normalized dependency tree.

Figure 3.7: Example of normalized dependency tree.

In this work, we applied three techniques to normalize the dependency tree, in order to reduce noise as well as enrich information in the SDP extracted from the dependency tree (see Figure 3.7 for example).

- **Preposition normalization**: Following De Marneffe and Manning [20], we collapse the *"pobj"* dependency (object of preposition) with the predecessor dependency (*e.g.*, *"prep"*, *"acl"*) into a single dependency, and cut the preposition off from the SDP.
- **Conjunction normalization**: Base on the assumption that two tokens that linked by a conjunction dependency *"conj"* should have the same semantic and grammatical roles; we therefor add a skip-edges to ensure that these conjuncted tokens have same dependencies with other tokens.
- ***Dependency cut-off***: Above normalization techniques would prevent the alteration of the semantics, but also lead to a non-uniform treatment of prepositions. Therefore, we cut-off unnecessary information to produce a representation which sacrifices the exact semantics of the original sentence by producing a sentence roughly equivalent, but which ensures uniformity across relations.

  For example, cut-off *"prep:of"*, *"prep:for"*, *"prep:regarding"*, etc. into *"prep"*; and cut-off *"conj:for"*, *"conj:in"*, *"conj:with"*, etc. into *"conj"*.

Figure 3.7 is an example of normalized dependency tree. We can see that both diseases "**Liver enlargement**" and "**muscle wastage**" have Chemical-Induced Disease (CID) relation with the chemical "**prednisolone**"; and they play the same role in the sentence. However, the SDP from "**Liver enlargement**" to "**prednisolone**" is different from the SDP from "**muscle wastage**" to "**prednisolone**". In addition, the SDP in this example goes through a sequence of nodes "**occurred following administration of**". This sequence contains some preposition nodes, such as "**following**" and "**of**", are not important for relation extraction. On the other hand, from the normalized dependency tree, the SDPs only contain the important node. Two old "$prep$" relations are changed to "$prep{:}following$" and "$prep{:}of$" relation. Moreover, the SDPs from two diseases to the chemical go through the identical nodes "**occurred administration**".

3.3.2 Shortest Dependency Path and Dependency Unit

3.3.2.1 Shortest Dependency Path

The Shortest Dependency Path (SDP) is the shortest sequence go from a starting token to the ending token in the dependency tree, containing tokens interspersed with the dependency between two adjacent tokens. Because the SDP offers a very condensed representation of the information needed to assess the relation between two entities [12], we suppose that the SDP contain necessary information to shows their relationship. Therefore, classifying the SDP becomes one of the most direct approaches to extract the relation between two entities which appear in the same sentence.

3.3.2.2 Dependency Unit on the SDP

According to the study of Le et al. [39], a pair of a token and its ancestor has the difference in meaning when they are linked by a different dependency relation. We make use of this structure and represent the SDP as a sequence of substructures like "$t_a \xleftarrow{r_{ab}} t_b$", in which t_a and t_b are token and its ancestor respectively; r_{ab} is the dependency relation between them. This substructure refers to the Dependency Unit (DU) as described in Figure 3.8. This example includes Dependency Unit of two nodes and conjuncted dependency relation. To capture wider range of dependencies, we make use of bigger Dependency Unit that contains 3, 4 or 5 nodes with 2, 3 or 4 conjuncted dependency relations respectively.

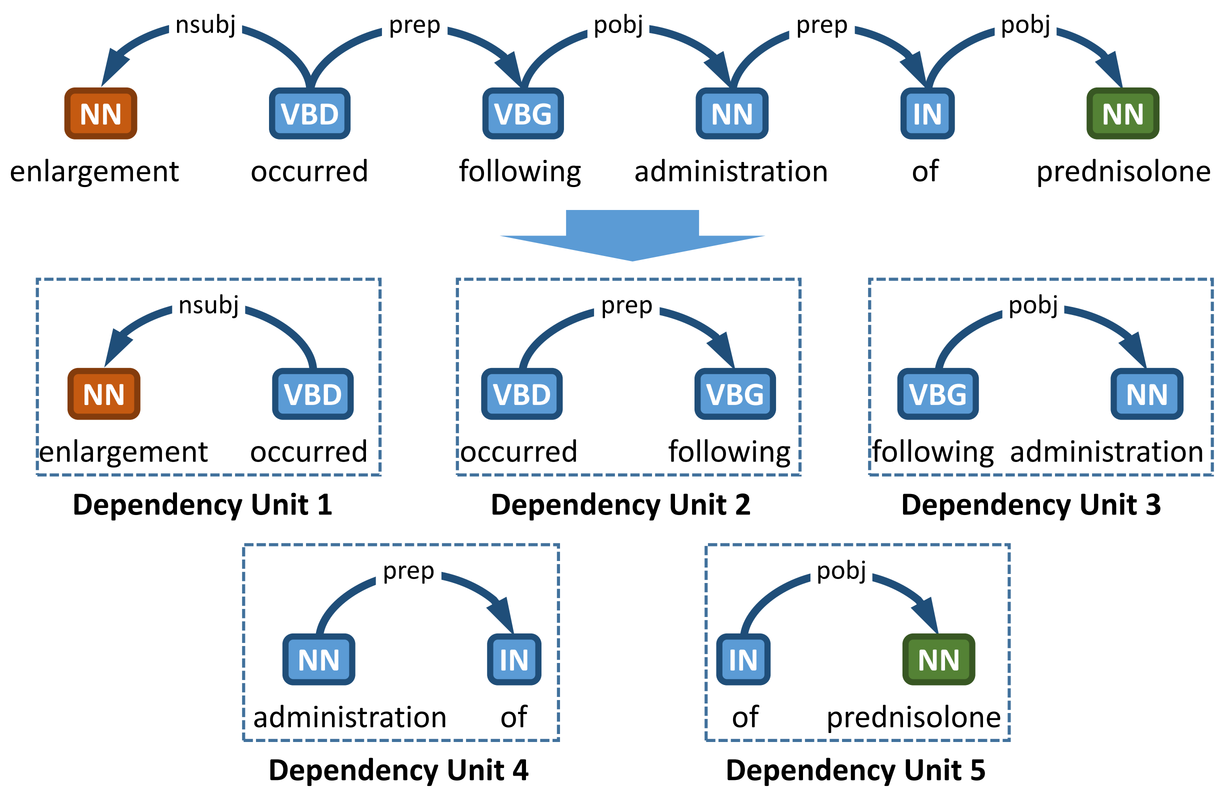


Figure 3.8: Dependency units on the SDP.

3.3.3 Richer-but-Smarter Shortest Dependency Path

As previously mentioned, we utilize the condensed information in the SDP to learn the relation between two nominals. The simple structure of the SDP is one of its weaknesses since there exists some useful information in dependency tree that does not appear in the SDP. This information can be leveraged to represent the relation more precisely. Two examples in Figure 3.9 belong to different relation types, but the paths between two nominals in these examples contain only one token ("**put**"). However, the meaning of token "**put**" in two SDPs are completely different. In this situation, it is difficult for the machine to distinguish the two shortest dependency paths from these instances.

We notice that the child nodes attached to the shortest dependency paths and their dependency relation from their parent can provide supplemental information for relation classification. In the previous examples, the sub-structure "$-\text{prt}\rightarrow$ **up**" provides semantic information about token "**put**" in the specific sentence to make it discriminated from the stand-alone one. Based on similar observations, we propose the idea of combining sub-tree information with original SDP to form a more precise structure for classifying relations. In this RbSP structure each token t is represented by itself and its attached children on the dependency tree.

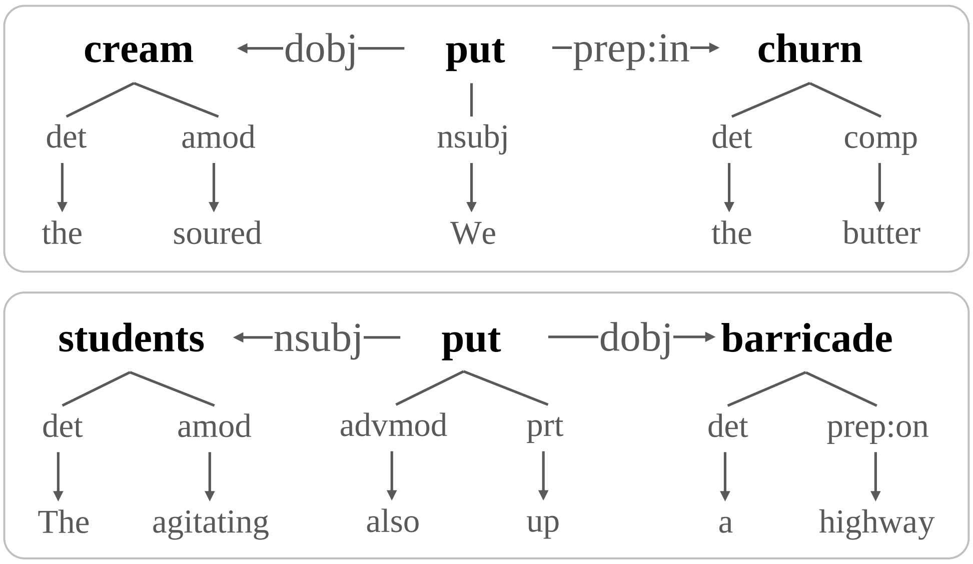


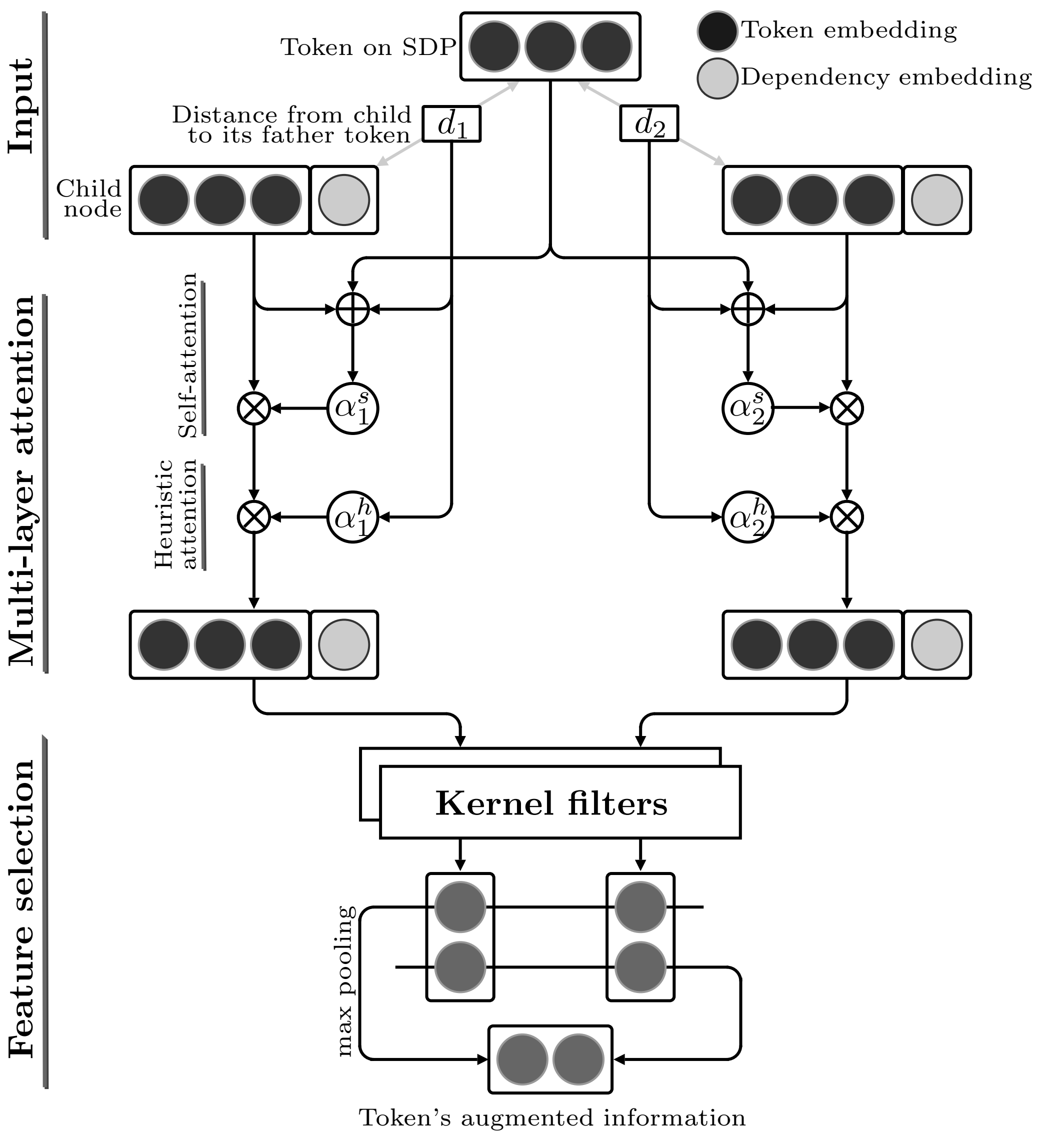
Figure 3.9: Examples of SDPs and attached child nodes.

3.4 Multi-layer Attention with Kernel Filters

To capture the appropriate augmented information from the child nodes of each token, we propose a novel multi-layer attention with kernel filters architecture. As illustrated in Figure 3.10, we employ two sequential attention layers on the children of a token to produce children context vectors. Afterward, to utilize all informative child nodes and preserve the integrity of the word information, we capture the token's augmented information using kernel filters instead of using the average of context vectors weighted by multi-layer attention.

3.4.1 Augmentation Input

Given a token t and their child nodes, we first represent every token by a real-valued vector to provide lexical semantic features. Token t is transformed into a token embedding vector $\mathbf{t} \in \mathbb{R}^{\dim_t}$ which is the concatenation of its word embedding and part-of-speech (POS) tag embedding. To utilize all the information in the sub-structure of token's children, we form a child node not only by its token embedding as in parent node but also by the dependency relation from its direct ancestor on the sentence's parse tree. Suppose t has a set C of M children, i.e., $C = \{c_1, c_2, ..., c_M\}$. Our model represents each child in C with a real-valued vector $\mathbf{c}_i \in \mathbb{R}^{\dim_t + \dim_{dep}}$. To additionally capture information about the child node to the target token, we incorporate the position embeddings d_i to reflect the relative distances between the ith child's token to the target parent token on the original sentence.



Notes: $\bigoplus$ denotes the concatenation of components. $\bigotimes$ denotes the scalar multiplication.

Figure 3.10: The multi-layer attention architecture to extract the augmented information.

3.4.2 Multi-layer Attention

We then apply a simple self-attentive network to child nodes $\{\mathbf{c_i}\}_{i=1}^{M}$ where the attention weights are calculated based on the concatenation of themselves with parent information and distance from parent, as follow:

$$\bar{\mathbf{C}} = \left\{\mathbf{c}_i \oplus \mathbf{t} \oplus d_i\mathbf{w}^d\right\}_{i=1}^{M} = \left\{\bar{\mathbf{c}}_i\right\}_{i=1}^{M} \tag{3.16}$$

$$\mathbf{e} = \left\{\bar{\mathbf{c}}_i\mathbf{W}^e + b_e\right\}_{i=1}^{M} = \left\{e_i\right\}_{i=1}^{M} \tag{3.17}$$

$$\alpha_i^s = \frac{\exp(e_i)}{\sum_{k=1}^{M}\exp(e_k)} \tag{3.18}$$

where $\oplus$ denotes concatenation operation; $\mathbf{w}^d \in \mathbb{R}^{\text{dim}_d}$ is base distance embedding; $\mathbf{W}^e$ and $b_e \in \mathbb{R}$ are weight and bias term. The self-attentive context vector $\mathbf{a}^s$ of the target token is the weighted sum of the self-attentive children context vectors as follows:

$$\mathbf{c}_i^s = \alpha_i^s \mathbf{c}_i \tag{3.19}$$

$$\mathbf{a}^s = \sum_i \mathbf{c}_i^s \tag{3.20}$$

We observe that the importance of a child node to the parent node depends on the distance between them on the original sentence. Therefore, we apply a heuristic attentive layer on the self-attentive children context vectors based on the distances $d_1, d_2, ..., d_M$ to keep track of how close each child is to the target token. We heuristically choose the activation function for the distances $d_1, d_2, ..., d_M$ as $f(d) = \beta d^2$ with $\beta = -0.03$, and a softmax layer is followed to calculate the heuristic attention weight. I.e.,

$$\alpha_i^h = \frac{\exp(\beta d_i^2)}{\sum_{k=1}^{N} \exp(\beta d_k^2)} \tag{3.21}$$

$$\mathbf{c}_i^h = \alpha_i^h \mathbf{c}_i \tag{3.22}$$

$$\mathbf{a}^h = \sum_i \mathbf{c}_i^h \tag{3.23}$$

3.4.3 Kernel Filters

The multi-attentive context vector $\mathbf{a}^h$ is a synthetic representation of all child nodes with the target token node taken into account. Since the child nodes are usually distinct from each other, an average vector is not suitable to represent the children information. We propose to use the kernel filters to capture the relevant and important information from the output of the multi-attention layer. K kernel filters are applied to each child's attentive vector to produce K features from each child. I.e.,

$$\mathbf{F} = \left\{ \text{ReLU} \left(\mathbf{c}_i^h \mathbf{W}^f + \mathbf{b}^f \right) \right\}_{i=1}^{M} \tag{3.24}$$

where $\mathbf{W}^f \in \mathbb{R}^{(2\text{dim}+\text{dim}_{dep}+\text{dim}_d)\times K}$ is the weight of K kernel filters; and $\mathbf{b}^f \in \mathbb{R}^K$ is bias term. Finally, to produce the final augmented information $\mathbf{a}$, we apply a max-pooling [10] layer to the feature matrix $\mathbf{F}$ and select the most important features as:

$$\mathbf{a} = \left\{ \max \left(\mathbf{F}_k^\intercal \right) \right\}_{k=1}^{K} \tag{3.25}$$

3.5 Deep Learning Model for Relation Classification

The extracted SDPs are then fed to a Deep Neural Network to predict the relation labels. The overall architecture of our proposed model is shown in Figure 3.11. Given a sentence and its dependency tree, we build our model on the SDP between two nominals and its attached children on the tree. Here, we mainly focus on the SDP representation using compositional embedding which is composed of dependency embeddings, token embeddings (will be discussed in Section 3.5.1), and token's augmented information (as illustrated in Section 3.4). After the SDP representation phase, each token and dependency relation is transformed into a vector. These sequence of vectors are then fed to a convolutional neural network to capture the convolved features that can be used to determine which relation two nominals are of (will be discussed in Section 3.5.2).

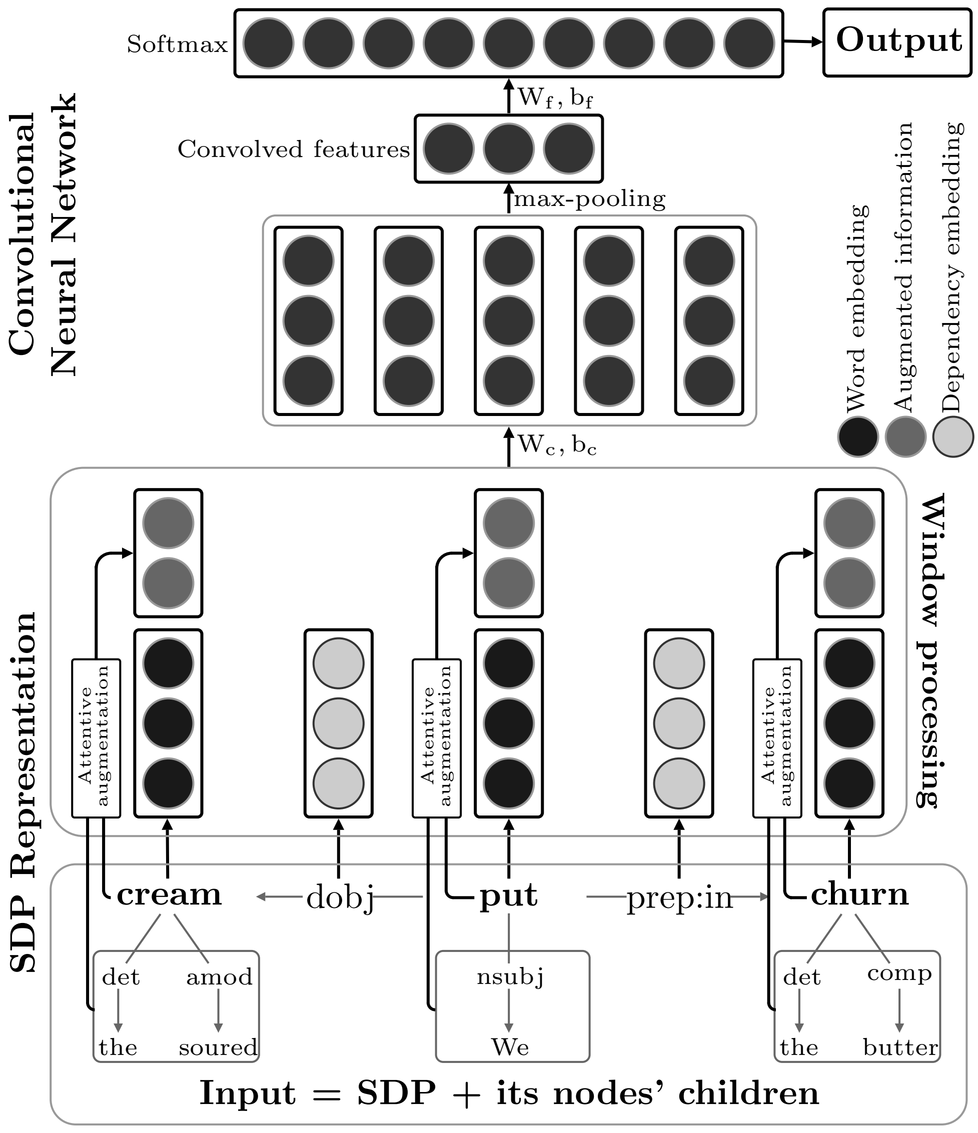


Figure 3.11: The architecture of RbSP model for relation classification.

3.5.1 Compositional Embeddings

In the embeddings layer, each component of the SDP (*i.e.*, token or dependency) are transformed into a vector $\mathbf{e} \in \mathbb{R}^d$, where d is the final dimension of an embedding vector. In order to capture more dependency features and linguistic features along the SDP, we compositionally represent the token and dependency on SDP.

3.5.1.1 Dependency Relation Embeddings

The dependency relations with directions are proven more effective than the dependency relations without directions for the relation extraction task [67]. However, treated the dependency relations with opposite direction as two separated relations can induce that two vectors of a same relation are disparate. We represent the dependency relations with two discrete components. A dependency relation dep_i is represented as a vector $\bar{\mathbf{d}}_i$ that is the concatenation of two vectors as follow:

$$\bar{\mathbf{d}}_i = \mathbf{d}_i^{typ} \oplus \mathbf{d}_i^{dir} \tag{3.26}$$

where $\mathbf{d}^{typ} \in \mathbb{R}^{\dim_{typ}}$ represents the dependency relation type among 62 labels; and $\mathbf{d}^{dir} \in \mathbb{R}^{\dim_{dir}}$ is the direction of the dependency relation, *i.e.* from left-to-right or vice versa on the SDP. The $\mathbf{d}^{typ}$ and $\mathbf{d}^{dir}$ vectors are generated by looking up embedding matrices $\mathbf{W}^e_{typ} \in \mathbb{R}^{\dim_{typ} \times 62}$ and $\mathbf{W}^e_{dir} \in \mathbb{R}^{\dim_{dir} \times 2}$ respectively.

The concatenated vector $\bar{\mathbf{d}}_i$ is then transform into final D-dimensional representation $\mathbf{d}_i$ of dependency relation as follow:

$$\mathbf{d}_i = \tanh\left(\bar{\mathbf{d}}_i \mathbf{W}^d + \mathbf{b}^d\right) \tag{3.27}$$

where $\mathbf{W}^d \in \mathbb{R}^{(\dim_{typ}+\dim_{dir}) \times D}$ and $\mathbf{b}^d \in \mathbb{R}^D$ are trainable parameters.

3.5.1.2 Token Embeddings

For token representation, as mentioned above, we assume that each token should be interpreted by itself and its children. Then, each token on SDP is represented by three types of information: (i) the word information $\mathbf{t}_i$ of token itself on the SDP; (ii) the dependencies information in the original sentence $\mathbf{h}_i$; (iii) the attentive augmented information $\mathbf{a}_i$ based on the attached children, as illustrated in Section 3.4.

Token Information Representation

Token information includes clues about semantic meaning, grammatical word-category, word morphology, shape of the token itself. In this work, we utilize five types of information, including:

1. ***Pre-trained word embeddings***: As previously mentioned in Section 3.1.1, word embeddings allow tokens that often appear in similar context to have similar representations, so that help to capture their meaning. Each token in the input SDP is transformed into a vector $\mathbf{t}_i^w$ by looking up the embedding matrix $\mathbf{W}_w^e \in \mathbb{R}^{\mathrm{dim}_{we} \times |V|}$, where dim_{we} is the word embedding dimension, and V is a vocabulary of all words we consider.

2. ***Character-based embeddings***: The previous CNN-based character-level representation, that described in Section 3.1.2, offers the information about word morphology and shape (like the prefixes or suffixes of word) from the characters of words. However, by practical evaluation, we find out that CNN-based character embeddings do not bring significant improvement. We, therefor, leverage LSTM network to learn the character-based embeddings of words.

   Given a token composed of N characters $c_1, c_2, ..., c_N$, we first represent each character c_j by a character embedding $\mathbf{c}_j$ using a look-up table $\mathbf{W}_c^e \in \mathbb{R}^{\mathrm{dim}_{char} \times |V^{char}|}$, where V^{char} is character alphabet. A bi-directional LSTM model is applied on the sequence $\{\mathbf{c}_1, \mathbf{c}_2, ..., \mathbf{c}_N\}$ to produce character hidden states $\mathbf{H} = \overleftrightarrow{\mathrm{biLSTM}}\left(\{\mathbf{c_i}\}_{\mathbf{i=1}}^{\mathbf{N}}\right)$. The final character embedding vector $\mathbf{t}_i^c$ is the concatenation of last forward hidden state and last backward hidden state.

3. ***Position embeddings***: The SDP is lack of location in sentence information, in which the informative words are usually close to the target entities. We make use of position embeddings to keep track of how close each SDP token is to the target entities on the original sentence. We first create a $2-$dimensional vector $[d_i^{e_1}, d_i^{e_2}]$ that is combination of relative distances $d_i^{e_1}$ and $d_i^{e_2}$ from each token to two entities respectively. For example, "*The [**fire**]$_{e_1}$ inside WTC was caused by exploding [**fuel**]$_{e_2}$.*" In this sentence, the position feature of the token "caused" is $[4, -3]$. Then, we obtain the position embedding $\mathbf{t}_i^p$ by transforming the position feature as follow:

$$\mathbf{t}_i^p = \tanh\left([d_i^{e_1}, d_i^{e_2}]\mathbf{W}^p + \mathbf{b}^p\right) \tag{3.28}$$

4. ***POS tag embeddings***: A token may have more than one meaning representing by its grammatical tag such as noun, verb, adjective, adverb, etc. To take advantage of this grammatical information, we concatenate the part-of-speech (POS) tag information into the token representation vector. We randomly initialize the embeddings matrix $\mathbf{W}_t^e \in \mathbb{R}^{d^t \times 56}$ for 56 OntoNotes 5.0 version of the Penn Treebank POS tags. Each POS tag label is then represented as a corresponding vector $\mathbf{t}_i^t$.

5. ***WordNet embeddings***: English WordNet is a large lexical database containing the set of the cognitive synonyms (synsets). Synsets are interlinked by their conceptual-semantic and lexical meanings. For this paper, we heuristically select 45 $F1$-children of the WordNet root which can be represent the super-senses of all synsets. The WordNet embedding $\mathbf{t}_i^n$ of a token is in form of a sparse vector that figure out which sets in 45 super-senses the token belongs to.

Finally, we concatenate the word embedding, character-based embedding, position embedding, POS tag embedding, and WordNet embedding of each token into a vector, and transform it into the final token embedding as follow:

$$\bar{\mathbf{t}}_i = \mathbf{t}_i^w \oplus \mathbf{t}_i^c \oplus \mathbf{t}_i^p \oplus \mathbf{t}_i^t \oplus \mathbf{t}_i^n \tag{3.29}$$

$$\mathbf{t}_i = \tanh\left(\bar{\mathbf{t}}_i \mathbf{W}^t + \mathbf{b}^t\right) \tag{3.30}$$

Sequence Dependencies Representation

To take advantage of the original sentence sequence information, we use a recurrent neural network with LSTM units to pick up the information along the sentence $\mathrm{S} = \{\mathbf{t}_i\}_{i=1}^n$ as follow:

$$\mathbf{H} = \overleftrightarrow{\mathrm{biLSTM}}(\mathbf{S}) = \{\mathbf{h}_i\}_{i=1}^n \tag{3.31}$$

Each token t_i is then augmented by the corresponding hidden state $\mathbf{h}_i$ from $\mathbf{H}$.

Final Token Representation

Finally, this concatenation is transformed into an X-dimensional vector to form the representation $\mathbf{x}_i \in \mathbb{R}^X$ of the token. I.e.,

$$\mathbf{x}_i = \tanh\left(\left[\mathbf{t}_i \oplus \mathbf{a}_i \oplus \mathbf{h}_i\right] \mathbf{W}^x + \mathbf{b}^x\right) \tag{3.32}$$

where $\mathbf{W}^x$ and b^x are trainable parameters of the network.

3.5.2 CNN on Shortest Dependency Path

3.5.2.1 Feature Extraction

After SDP representation layer, the input SDP is transformed into:

$$\mathbf{SDP} = \left[\mathbf{x}_1, \overleftarrow{\mathbf{d}_1}, \mathbf{x}_2, ..., \mathbf{x}_{N-1}, \overrightarrow{\mathbf{d}_{N-1}}, \mathbf{x}_N\right] \tag{3.33}$$

in which the over-arrow on $\mathbf{d}_i$ denotes the direction of the dependency relation. In general, let us define the slice $\mathbf{x}_{i:i+j}$ as the concatenation of j tokens and $j-1$ dependency relation between them. This slice of j tokens is also known as a Dependency Unit of j tokens. I.e.,

$$\mathbf{x}_{i:i+j} = \mathbf{x}_i \oplus \mathbf{d}_i \oplus \mathbf{x}_{i+1} \oplus ... \oplus \mathbf{d}_{i+j-2} \oplus \mathbf{x}_{i+j-1} \tag{3.34}$$

To extract features from sequence of Dependency Units along the SDP, we build a CNN model on this SDP; our model is similar to the model of Xu et al. [67]. The convolution operation with region size r applies k filters to all possible window of r successive tokens to produce convolved feature map. We then gather the most important features by applying a max pooling [10] layer over the entire feature map. I.e., the convolutional layer computes the i-th element of the convolved feature vector $\mathbf{f}$ as follows:

$$\mathbf{f}_i = \max_{0 \le j \le N-r+1} \left[\mathbf{x}_{j:j+r}\mathbf{W}^c + \mathbf{b}^c\right]_i \tag{3.35}$$

where $\mathbf{W}^c \in \mathbb{R}^{(rX+(r-1)D)\times k}$ and $\mathbf{b}^c \in \mathbb{R}^k$ are the weight matrix and bias vector of the convolutional layer.

3.5.2.2 Fully connected MLP and Softmax layer

In this phase, we use the convolved features $\mathbf{f}$ from the convolutional layer which is a single feature vector representing the input. This feature vector has dimensionality that is equal to the number of filters we used. To extract more meaningful features, we use a fully connected MLP network with many hidden layers. We choose hyperbolic $\tanh$ as the non-linearity function in hidden layers, which has the advantage of being slightly cheaper to compute, while leaving the generalization performance unchanged.

The output vector of last hidden layer $\mathbf{h}_n$ can be considered as higher level features, which is then fed to a $\mathrm{softmax}$ classifier to predict a (K+1)-class distribution over labels $\hat{\mathbf{y}}$. I.e.

$$\hat{\mathbf{y}} = \mathrm{softmax}\left(\mathbf{h}_n \mathbf{W}^y + \mathbf{b}^y\right) \tag{3.36}$$

where $\mathbf{W}^y$ and $\mathbf{b}^y$ are parameter of the network to be learned.

3.5.3 Training objective and Learning method

The proposed model can be stated as a parameter tuple $\theta = (\mathbf{W}, \mathbf{b})$. To compute the model parameters θ, we define the training objective for a data sample as:

$$L(\theta) = -\sum_{i=0}^{K} \mathbf{y}_i \log \hat{\mathbf{y}}_i + \lambda \left\|\theta\right\|^2 \tag{3.37}$$

where $\mathbf{y} \in \{0,1\}^{(K+1)}$ indicating the one-hot vector represented ground truth; and λ is a regularization coefficient. By minimizing $L(\theta)$ using mini-batch gradient descent (GD) with Adam optimizer [37], θ is updated via back-propagation through neural network structures.

3.5.4 Model Improvement Techniques

For this paper, we directly utilize the pre-trained fastText word embeddings model [9], which is trained on Wikipedia data, to represent token on SDP for general domain. To adapt our model to biomedical data, we trained a new embeddings model with the vocabulary size of 1.52 million words on 25 million PubMed abstracts from MEDLINE[3] literature.

The look-up tables for dependency relation types, dependency directions, word characters, POS tags are randomly constructed using the Glorot normal initializer [23]. These look-up tables are then treated as the parameters of the model to be learned during the training phase.

Since the CNN model take the fixed size matrix as input, we need to trim the longer inputs and pad the shorters to the same length. Instead of global padding for all inputs, we pad the inputs in each batch of data dynamically to the longest input length

[3]MEDLINE® description at `https://www.nlm.nih.gov/bsd/medline.html`

of the batch. We further use the batch normalization [33] which is able to enable higher learning rates and reduces over-fitting.

During the training phase, we make use of several techniques, including: *clipping the gradients* if their norm exceeds a given threshold to mitigate gradient vanishing and gradient exploding [24]; adding *Gaussian noise* [54] with mean 0.001 to the input embeddings; applying *dropout* [61] with the probability of 0.5 on embeddings layer, CNN hidden states, and penultimate layer; using *early stopping* [15] by validation loss after 10 epochs without loss improvement.

Further, to reduce the impact of random effects on our model, we employ the ensemble mechanism and find it reduce the variance whilst yielding better performance than the average result. Two simple ensemble methods that are proven the effectiveness include strict majority vote [39, 47] and weighted sum [63] over the distinct distribution. For this study, we run the model for 20 times and uses the strict majority vote to obtain the final results.

Chapter 4

Experiments and Results

4.1 Implementation and Configurations

4.1.1 Model Implementation

Our model was implemented using Python version 3.5 and TensorFlow[1]. TensorFlow is a free and open-source platform designed by the Google Brain team for data-flow and differentiable programming across a number of machine learning tasks. It has a comprehensive, flexible ecosystem of tools, libraries and community resources that are used to bring out state-of-the-art in many tasks of ML. TensorFlow can be used in research and industrial environment as well.

Other Python package requirements include:

- `numpy`
- `scipy`
- `h5py`
- `Keras`
- `sklearn`

The sample data and source code are available on GitHub. Source code is public and is free to edit and upgrade.

- Compositional embedding project: `https://github.com/catcd/cduCNN`
- Richer-but-Smarter SDP project: `https://github.com/catcd/RbSP`

[1]Information about TensorFlow is at `https://www.tensorflow.org/about`

4.1.2 Training and Testing Environment

Our model's training and testing experiments are executed on FIT High Performance Computing (`FIT-HPC`) system. `FIT-HPC` runs Linux-based operating systems. The configurations of system are listed bellow:

- 1 central controller machine
  - · 2 x Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30GHz
  - · 4 x 16 GB DIMM ECC DDR4 @ 2400MHz
- 16 computing machines
  - 1 machine for database system
    - · 2 x Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30GHz
    - · 4 x 16 GB DIMM ECC DDR4 @ 2400MHz
  - 2 machines with powerful GPU
    - · 2 x Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30GHz
    - · 4 x 16 GB DIMM ECC DDR4 @ 2400MHz
    - · GPU: Tesla K40m
  - 12 machines with medium RAM capacity
    - · 2 x Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30GHz
    - · 8 x 16 GB DIMM ECC DDR4 @ 2400MHz
  - 1 machine with high RAM capacity
    - · 2 x Intel(R) Xeon(R) CPU E5-2697 v4 @ 2.30GHz
    - · 24 x 16 GB DIMM ECC DDR4 @ 2400MHz
- 1 machine for data storage and extended JBOD
  - · 2 x Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz
  - · 8 x 16 GB DIMM ECC DDR4 @ 2400MHz
  - · 24 x 6.0 TB HDD
  - · JBOD: 24 x 6.0 TB HDD

4.1.3 Model Settings

The tuned hyper-parameters are described in Table 4.1. These hyper-parameters and number of parameters are for reference on SemEval 2010 Task 8 dataset only, the real value may vary depended on testing dataset and some additional techniques.

Table 4.1: Configurations and parameters of proposed model.

Information		**Configuration**	**Parameters**
Dependency embeddings	Dependency type	LUT $\mathbf{W}^e_{typ}$ size 62×100	$6,300$
	Dependency direction	LUT $\mathbf{W}^e_{dir}$ size 2×50	100
	Dependency embedding transformation	$(\mathbf{W}^d, \mathbf{b}^d)$ transform from 150 dims to 100 dims	$15,100$
Token information	Character embedding	LUT $\mathbf{W}^e_c$ size 85×50	$4,250$
		biLSTM with 50 dims hidden state	$40,400$
	POS tag	LUT $\mathbf{W}^e_t$ size 57×50	$2,850$
	Position embedding transformation	$(\mathbf{W}^d, \mathbf{b}^d)$ transform from 2 dims to 50 dims	150
	Token information transformation	$(\mathbf{W}^t, \mathbf{b}^t)$ transform from 545 dims to 300 dims	$163,800$
Sentence dependencies modeling		biLSTM with 100 dims hidden state	$320,800$
Augmented information	Base distance embedding	vector $\mathbf{w}^d$ 32 dims	32
	Self attention score	$(\mathbf{W}^e, \mathbf{b}^e)$ transform from 832 dims to scalar	833
	Heuristic attention	Linear	-
	Kernel filters	100 filters size 832×1	$83,300$
Final token embedding transformation		$(\mathbf{W}^x, \mathbf{b}^x)$ transform from 500 dims to 300 dims	$150,300$
Convolutional Neural Network		256 filters size 300×1	$77,056$
		512 filters size 700×1	$358,912$
		512 filters size 1100×1	$563,712$
Classifier	Fully-connected MLP	1 hidden layers 128 nodes	$163,968$
	Softmax	19 classes	$2,451$
		Total number of parameters	$1,954,314$

4.2 Datasets and Evaluation methods

4.2.1 Datasets

Our model was evaluated on two different datasets: SemEval-2010 Task 8 dataset [29] for general domain relation extraction and BioCreative V Track 3 CDR dataset [65] for chemical-induced disease relation extraction in bio-medical data.

4.2.1.1 SemEval 2010 Task 8 dataset

The SemEval-2010 Task 8 contains 10,717 annotated relation classification examples, and is separated into two subsets: 8,000 instances for training and 2,717 for testing. We randomly split $1/10$ of the training data for validation dynamically. Table 4.2 shows the distributions of the relation types on this dataset. There are 9 directed relations and one undirected `Other` class.

Table 4.2: Statistics of SemEval-2010 Task 8 dataset.

Relation	**Frequency**	
	Train	***Test***
Cause-Effect	1003 (12.54%)	328 (12.07%)
Component-Whole	941 (11.76%)	312 (11.48%)
Entity-Destination	845 (10.56%)	292 (10.75%)
Product-Producer	717 (8.96%)	231 (8.50%)
Entity-Origin	716 (8.95%)	258 (9.50%)
Member-Collection	690 (8.63%)	233 (8.58%)
Message-Topic	634 (7.92%)	261 (9.61%)
Content-Container	540 (6.75%)	192 (7.07%)
Instrument-Agency	504 (6.30%)	156 (5.74%)
Other	1410 (17.63%)	454 (16.71%)
Total	**8000**	**2717**

We further investigated this dataset and found that there are 28% of Out-Of-Vocabulary (OOV) words that appear in testing set but do not apprear in training set. More interesting, the percentage of nominal pairs in the testing set that have never appeared in the training set is more than 93%.

4.2.1.2 BioCreative V Track 3 CDR dataset

The BioCreative V CDR copus consists of three datasets: training, development and testing set. Each dataset has 500 PubMed articles (the details are shown on Table 4.3), in which an abstract contains human annotated chemicals, diseases entities, and their abstract-level chemical-induced disease relations. According to the data survey of BioCreative [65], most abstracts were selected from an existing CTD dataset; there are 100 new curated abstracts and they are incorporated into the testing set.

Table 4.3: Summary of the BioCreative V CDR dataset

Data set	**Articles**	**Chemical**		**Disease**		**CID**
		Mention	**ID**	**Mention**	**ID**	**relations**
Training	500	5203	1467	4182	1965	1038
Development	500	5347	1507	4244	1865	1012
Testing	500	5385	1435	4424	1988	1066

The OOV ratio of this dataset is about 34% comparing with 28% of SemEval 2010 dataset. The percentage of new entity pairs is 79%. We also find out that there are about 30% cross-sentence relations in this dataset. Because our model only extract intra-sentence relation, the upper bound of our model's Recall is 70%.

4.2.2 Metrics and Evaluation

To evaluate the performance of a RE system, we use three standard metrics: Precision, Recall and $F1$ score. Suppose a dataset has G positive examples. A RE system predicts P "positive" relation instances, in which only some instances, denoted by TP, are actually positive. The system may also mistakenly extract some relation instances as positive that we call FP. In G, some relation instances are not extracted by the system which we denote by FN. Based on the above definitions, precision (P), recall (R) and $F1$ score can be defined as:

$$P = \frac{TP}{TP + FP} \qquad R = \frac{TP}{TP + FN} \qquad F1 = \frac{2 \times P \times R}{P + R} \tag{4.1}$$

In the experiments, we fine-tune our model on training (and development) set(s) and report the results on the testing set, which is kept secret with the model. We conduct the training and testing process 20 times and calculate the averaged results.

4.3 Performance of Proposed model

4.3.1 Comparative models

Our model is compared with many comparative models that have reported results for the SemEval 2010 Task 8 or BioCreative V CDR datasets. We categorize comparing model into four types: feature-based models, deep learning models using whole sentence, SDP-based models, and attention-based models. For a fair comparison with other researches, we implemented a **Baseline** model, in which we remove all the proposed components (compositional embeddings, dependency unit, augmented information using multi-layer attention with kernel filters, and LSTM on original sentence). This baseline model is similar to the model of Xu et al. [67], with some technical improvements and additional information sources.

For a comprehensive investigation, we implemented two proposed model separately: **cduCNN** and **RbSP**. The first model, **cduCNN**, is the **Baseline** model enhanced with compositional embedding and dependency unit. We also investigated the impact of dependency normalization techniques on this model. **RbSP** is upgrade version of **cduCNN** with augmented information, including children attentive augmentation and sequence dependencies on original sentence.

4.3.1.1 SemEval 2010 Task 8 models

On SemEval 2010 Task 8 dataset, we compared our model with 10 other comparative models. The information of these models are summarized bellow:

- One feature-based model:
  - **SVM** (Rink and Harabagiu [56]): This is the top performed feature-based system that takes place by means of SVM classifiers and various features capturing the context, semantic, and possible pre-existing nominal relations.
- Two deep learning models using whole sentence:
  - **CNN** (Zeng et al. [70]): This model predict relation label from all of the word tokens without complicated pre-processing. The lexical level and sentence level features are extracted using the given nouns and a CNN model.
  - **mvRNN** (Socher et al. [60]): Authors use a recursive neural network to learns the distributed representation from the path between two entities on constituent parse tree for relation classification.

- Four SDP-based models:
  - **SDP-LSTM** (Xu et al. [68]): SDP-LSTM picks up heterogeneous information along the SDP with a Long Short-Term Memory network.
  - **depLCNN**: Xu et al. [67] investigated negative sampling technique on CNN-based model to capture relation between two nominals using directed SDP.
  - **BRCNN** (Cai et al. [14]): By using two directed CNN model stacking on the output of two RNN with LSTM unit, BRCNN model can capture the direction of relations effectively.
  - **DepNN** (Liu et al. [44]): To overcome the lack of information on SDP problem, DepNN utilized a recursive neural network to model the token's subtree. This information is used to enrich the original SDP.
- Three attention-based models:
  - **Attention-CNN** (Shen and Huang [59]): This is one of the earliest attention-based model applied to RE task. Attention-CNN model uses attention mechanism to focus on the important information on input sentence before extracting features by a CNN model.
  - **AT-BLSTM** (Zhang et al. [71]): In this model, an attention layer organizes the word level context and a tensor layer identifies complicated links between two entities.
  - **Multi-Att-CNN** (Wang et al. [64]): This is a novel convolutional neural network architecture that relies on two levels of attention to better distinguish patterns in heterogeneous contexts.

4.3.1.2 BioCreative V CDR models

On BioCreative V Track 3 CDR dataset, we compared our model with 9 other approaches. **BioCreative** provided 3 benchmark results. First one is result of simple co-occurrence model. Two others results are first rank and average result from BioCreative V Track 3 competition. The information of 6 other models are summarized bellow:

- Two feature-based models:
  - **UET-CAM** (Le et al. [38]): UET-CAM is a pipeline with a co-reference resolution module and a SVM relation extraction model. This model uses a very rich set of features for the input of SVM.

    - **ASM** (Panyam et al. [50]): ASM stands for Approximate Subgraph Matching kernel that applied graph kernels for extracting relations expressed in multiple sentences.

- Two deep learning models using whole sentence:
    - **hybridDNN** (Zhou et al. [76]): This is a hybrid model that consists of a feature-based model, a tree kernel-based model and a neural network model. The feature-based model takes advantage of lexical features, the tree kernel-based model captures features of syntactic structure, and semantic representations are generated by the neural network model.
    - **ME+CNN** (Gu et al. [25]): This model handles cross-sentence relations and intra-sentence relations by two separate models: A maximum entropy model for relation extraction at inter-sentence relation and a convolutional neural network model for intra-sentence level.
- One SDP-based model:
    - **MASS**: Le et al. [39] proposed a multi-chanel bidirectional LSTM network in combination with two directed CNNs to extract relation from shortest dependency path between two entities.
- One attention-based model:
    - **BRAN** (Verga et al. [63]): This method solve the problem of relation at document level (rather than sentence level), which simultaneously predicts relationships between all mention pairs in a document using an efficient self-attention encoder.

4.3.2 System performance on General domain

4.3.2.1 Performance of Compositional embeddings

Table 4.4 summarizes the performances of our model and comparative models. Our baseline model, which we interleave the word embeddings and dependency type embeddings for the input of CNN, yields higher $F1$ than competitors which are feature-based or DNN-based with information from pre-trained Word embeddings only. With the improvement of 0.3% when applying DU on the baseline model, our model achieves the better result than the remaining comparative DNN approaches which utilized full sentence and position feature without the advanced information selection methods (*e.g.*, attention mechanism). This result is also equivalent to other SDP-based methods.

Table 4.4: The comparison of our model with other comparative models on SemEval 2010 Task 8 dataset.

Model	**Source of information**	**F1**
SVM Rink and Harabagiu [56]	Lexical features, dependency parse, hypernym, NGrams, PropBank, FanmeNet, NomLex-Plus, TextRunner	82.2
CNN	Word embeddings	69.7
Zeng et al. [70]	+ Lexical features, WordNet, position feature	82.7
mvRNN	Word embeddings	79.1
Socher et al. [60]	+ WordNet, NER, POS tag	82.4
SDP-LSTM	Word embeddings	82.4
Xu et al. [68]	+ WordNet, GR, POS tag	83.7
depLCNN Xu et al. [67]	Word embeddings, SDP, CNN	81.9
	+ WordNet, word around nominals	83.7
	+ Negative sampling	85.6
BRCNN	Word embeddings, SDP, LSTM, CNN	85.4
Cai et al. [14]	+ POS, NER, WordNet embeddings, inverse SDP	86.3
DepNN Liu et al. [44]	200-d Gigaword embeddings, SDP, CNN	81.8
	+ Augmented sub-tree, Recursive Neural Network	82.8
	+ NER	83.6
Attention-CNN	Sentence convolution, Attention-based context	84.3
Shen and Huang [59]	+ WordNet, Words around nominals	85.9
AT-BLSTM Zhang et al. [71]	Word embeddings, Sentence attention features, Tensor feature	86.3
Multi-Att-CNN Wang et al. [64]	Multi-Level Attention CNNs, Attention pooling	88.0†
		85.5‡
Baseline	Word embeddings	83.4
	+ Dependency Unit	83.7
cduCNN (our model)	Compositional Embedding, Dependency Unit	84.7
	+ Normalize conjunction	85.1
	+ Normalize object of a preposition	80.6
RbSP	**cduCNN** + Augmented Information	86.3
(our model)	+ Ensemble	**86.7**

Notes: The reported results are macro-averaged F1 scores of (9+1)-way evaluation with directionality taken into account. Since the comparative models did not report the precision (P) and recall (R), we also report the F1 score only. †: We failed to reproduce good result with the Multi-Att-CNN model, the performance of our implementation is just about 84.9. ‡: Another re-implemented result of Multi-Att-CNN model reported by Zhang et al. [71].

The results also demonstrate the effectiveness of using compositional embedding that brings an improvement of 1.0% in $F1$. Our **cduCNN** model yields an F1-score of 84.7%, outperforms other feature-based and CNN-based comparative models, except **depLCNN** with data augmented strategy, by a large margin. However, the ensemble strategy by majority voting on the results of 20 runs drives our model to achieve a better result than the augmented **depLCNN** model.

It is worth to note that we have also conducted two techniques to normalize the dependency tree. Unfortunately, the results did not meet our expectations, with only 0.4% improvement of conjunction normalization. Normalizing the object of preposition even degrades the performance of the model with 4.1% of F1 reduction. A possible reason is that the preposition itself represent the relation on SDP, such as "*scars from stitches*" shows `Cause-Effect` relation while "*clip about crime*" shows `Message-Topic` relation. With the cut-off of prepositions, the SDP is lack of information to predict the relation.

4.3.2.2 Performance of Multi-layer attention with Kernel filters

Table 4.4 also summarizes the performance of our multi-layer attention with kernel filters architecture. The base **cduCNN** model yields higher F1 than competitors which are based on SDP without any data augmentation methods. This result is also comparative when is placed next to the result of basic **Attention-CNN** model.

The results also demonstrate the effectiveness of our proposed methods that brings an improvement of 1.6% in $F1$, compared to the **cduCNN** base result at 84.7%. Our RbSP model yields an F1-score of 86.3%, outperforms other comparative models, except Multi-Att-CNN model of Wang et al. [64] with multi-level attention CNN. However, we have tried to re-implement the **Multi-Att-CNN**, but we failed to reproduce the positive result in the original paper. The performance of our re-implementation is about 84.9% of $F1$. This result has a high consensus with Zhang et al. [71] since they also tried to re-build this model, and their re-implemented result is not much different from us, as 85.5%.

It is worth to note that when comparing with another augmented method of Liu et al. [44], our multi-layer attention with kernel filters architecture brings more significant improvement. Relatively, in comparison of efficiency of augmented methods on the baseline model, the full-tree augmentation only brings 1% improvement of F1 while our attentive augmentation boosts up to 1.6%. Unlike the method of using the whole sub-tree to supplement information for the target node, our method only uses the most relevant nodes that are direct children to represent augmented information. In addition, our method further focuses on the most important children through two attention layers.

We also observe that during many training-testing processes, the results may vary. The standard deviation of 20 runs is about 0.27. We perform the ensemble strategy by majority voting on the results of 20 runs, and it drives our model to achieve a better result of 86.7%. This result is outperformed other comparative models.

4.3.3 System performance on Biomedical data

To demonstrate the robustness and ability to adapt to different data of our proposed model, we applied the **cduCNN** model and **RbSP** model on the biomedical literature to extract Chemical-Induced Disease relations. Table 4.5 shows our results on biomedical BioCreative V CDR corpus compared to some related researches.

The results also highlight out the limitation of our model about cross-sentence relation. Our model receives input as the dependency tree of a sentence containing at least two entities. Therefore, it is not capable to extract cross-sentence relations from abstract, which have the arguments of relations in different sentences. According to our statistics on the BioCreative V CDR dataset, about 30% of relations are inter-sentence relationships. The upper bound of our model's $Recall$ is 70%. We leave this issue for our future works.

However, our model yields very competitive results when compared to other models that have taken into account the inter-sentence relationships. Our model outperforms the traditional SVM model using rich feature set without additional data and the hybrid DNN model with position feature. The average result is lower than ASM model using dependency graph.

Different from the results on the SemEval 2010 Task 8 dataset, both dependency tree normalization techniques bring improvement to the model. Normalizing object of preposition only boosts 0.06% of $F1$, while the conjunction normalization brings 0.3% efficiency. With the improvement of 1.19% from attentive augmented information, our **RbSP** model model achieves positive results at 57.73%, outperforms other competitors that do not take into account inter-relations. The ensemble technique can boost our F1-score 0.6%.

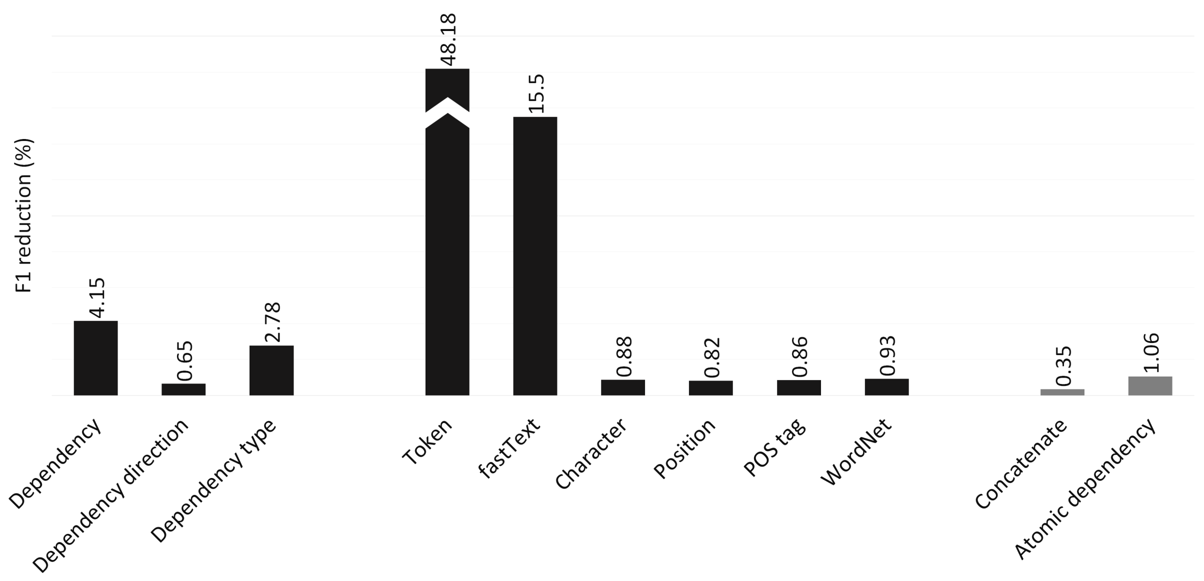
Table 4.5: The comparison of our model with other comparative models on BioCreative V CDR dataset

Model	Feature set	P	R	F1
BioCreative benchmarks[*]	Co-occurrence	16.43	76.45	27.05
	Average result	47.09	42.61	43.37
	Rank no.1 result	55.67	58.44	57.03
UET-CAM Le et al. [38]	SVM, rich feature set	53.41	49.91	51.60
	+ silverCID corpus	57.63	60.23	58.90
ASM[†] Panyam et al. [50]	Dependency graph	49.00	67.40	56.80
hybridDNN Zhou et al. [76]	Syntactic feature, word embeddings	62.15	47.28	53.70
	+ Context	62.39	47.47	53.92
	+ Position	62.86	47.47	54.09
ME+CNN[†] Gu et al. [25]	Contextual of whole sentence	59.70	57.50	57.20
	+ Cross-sentence	60.90	59.50	60.20
	+ Post processing	55.70	68.10	61.30
MASS[†] Le et al. [39]	SDP, LSTM, CNN, WordNet	58.90	54.90	56.90
	+ Ensemble	56.80	57.90	57.30
	+ Post processing	52.80	71.10	60.60
BRAN[†] Verga et al. [63]	Position, multi-head attention	55.60	70.80	62.10
	+ Data	64.00	69.20	66.20
	+ Ensemble	63.30	67.10	65.10
Baseline	Word embeddings	60.25	49.37	54.27
	+ Dependency Unit	60.33	50.36	54.90
cduCNN (our model)	Compositional Embedding, Dependency Unit	57.24	55.27	56.24
	+ Normalize conjunction	56.95	56.14	56.54
	+ Normalize object of a preposition	56.66	55.94	56.30
RbSP (our model)	**cduCNN** + Augmented Information	55.28	60.41	57.73
	+ Ensemble	56.08	60.77	58.33
	+ Post processing	52.38	74.19	61.40

[*] results are provided by the BioCreative V.

[†] reported result are only one decimal place.

We further apply the post processing rules, that have proposed in study of Gu et al. [25], on the predictions of our model to improve the recall. Improved **RbSP** model achieves the best result among competing models, except **BRAN** model [63] with 61.40% of $F1$.



Notes: The black columns indicate the removing of components. The grey columns indicate the alternative methods of embedding.

Figure 4.1: Contribution of each compositional embeddings component.

4.4 Contribution of each Proposed Component

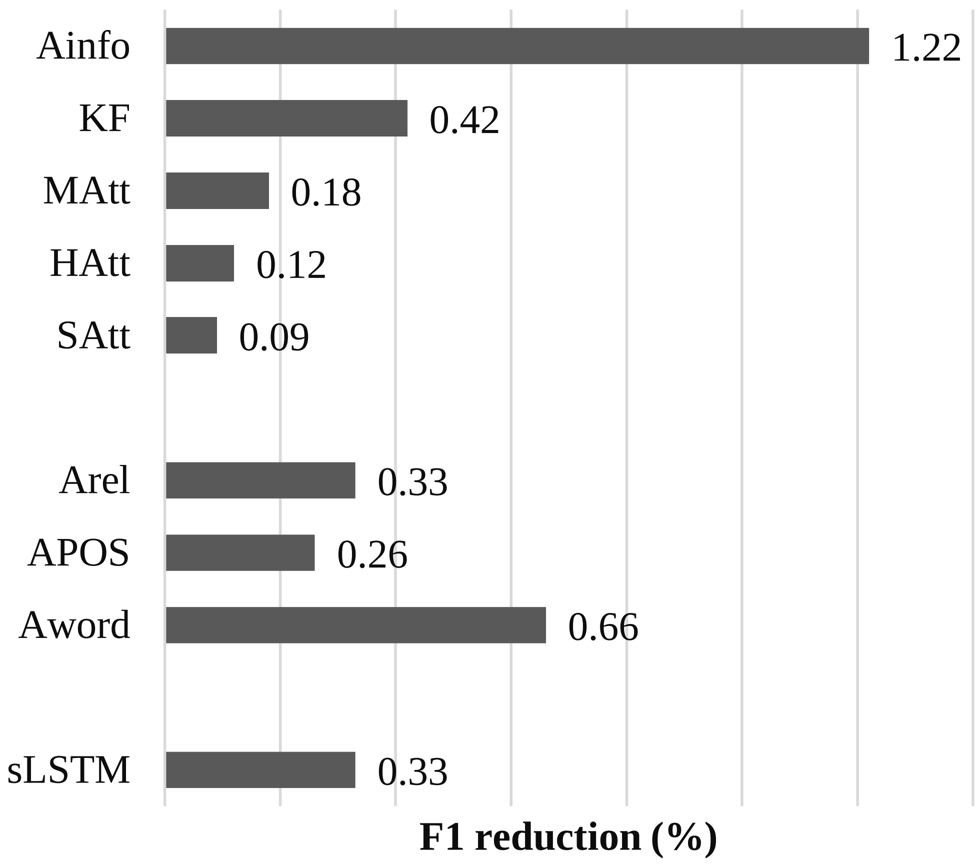
4.4.1 Compositional Embedding

We further investigate the contribution of each component and information source. Figure 4.1 shows the changes in $F1$ when ablating each component and information source from the **cduCNN** model. The $F1$ reductions illustrate the contributions of all proposals to the final result. However, the important levels are varied among different components and information sources.

Both dependency and token embeddings have a great influence on the model performance. Token embedding plays the leading role, eliminating it will reduce the $F1$ by 48.18%. However, dependency embedding is also an essential component to have the good results. Removing fastText embedding, dependency embedding and dependency type make significant changes of 15.5% 4.15% and 2.78% respectively. The use of other components brings a quite small improvement.

An interesting observation comes from the interior of dependency and token embeddings. The impact of kicking the whole component out is much higher than the total impact of kicking each minor component out. This proves that the combination of constituent parts is thoroughly utilized by our compositional embedding structure.

Another experiment on using alternative methods of embedding also proves the



Notes: self-attention (SAtt), heuristic attention (HAtt), multi-layer attention (MAtt), kernel filers (KF), augmented information (Ainfo), augmentation using word embedding (Aword), augmentation using POS tag (APOS), augmentation using dependency relation (Arel), and LSTM on original sentence (sLSTM).

Figure 4.2: Comparing the contribution of augmented information by removing these components from the model

minor improvement of compositional embedding. The result lightly reduces when we concatenate the embedding elements directly without transforming into a final vector or treat two divergent directional relations as to atomic relations.

4.4.2 Attentive Augmentation

Figure 4.2 shows the changes in F1 when removing each proposed component from the **RbSP** model. The F1 reductions illustrate the contributions of all proposals to the final result. However, the impact levels vary with different components. Between two proposed component, the multi-layer attention with kernel filters (augmented information) plays a vital role when contributing 1.22% to the final performance while the contribution of the LSTM on the original sentence is 0.33%.

An interesting observation comes from the interior of the multi-layer attention with kernel filters. The impact of removing the whole augmented information is much higher than the total impact of removing multi-layer attention or kernel filters (1.22 vs. $0.42 + 0.18 = 0.6$). These results demonstrate that the combination of constituent parts is thoroughly utilized by our sequential augmented architecture.

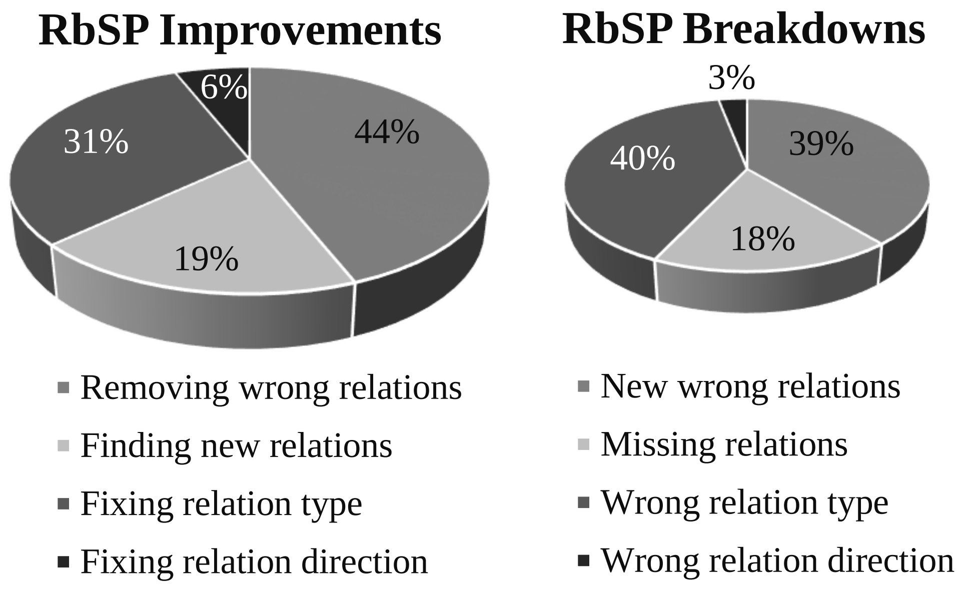
Another experiment is on investigating the meaning of each attention component. The result lightly reduces when we remove the self-attention or heuristic attention component. The results also prove that our proposed heuristic attention method is simple but effective. Its improvement is equivalent to the self-attention which is a complex attention mechanism. Among the input of multi-layer attention, the word embedding has a great influence on the model performance. However, children POS tag and relation to parent are also essential components to have the good results.

4.5 Error Analysis

We studied model outputs to analyze system errors in the cases of using the baseline model and using the proposed model with RbSP representation. In Figure 4.3, we considered four types of errors: If the model makes a wrong decision and labels an `Other` relation (negative) as an actual relation (positive), it indicates 1 `FP` (False Positive) error. Vice versa, if it labels an actual relation as `Other`, it brings 1 `FN` (False Negative). In the case that model confused between two types of relations, the model will be penalized twice, with 1 `FP` and 1 `FN`. Direction error, i.e., the model predicts the relation correctly but its direction wrongly, also brings 1 `FP` and 1 `FN`. The proportions of the left and the right of Figure 4.3 are quite consistent. In which, RbSP seems to have the most impact on determining whether an instance is positive or negative. RbSP also changes the decision of the relation type in quite many cases. It also influences the decision-making about relation's directionality, but not much.

Totally, the use of RbSP helps to correct more than 150 errors of the baseline model. However, it also yields some new errors (about 70 errors). Therefore, the difference of $F1$ between the baseline model and our RbSP model is only 1.5%, as stated in table 4.4.

Table 4.6 gives some realistic examples of different results when using the RbSP and not. We observed that the baseline model seems to be stuck in over-fitting problem, for examples, it classified all SDP with *prep:with* as `Instrument-Agency` and all SDP with *prep:in* as `Member-Collection` (examples $1-2$). RbSP is really useful for solving these cases partly since it uses attentive augmentation information to distinguish the same SDP or the same preposition with different meanings. RbSP is also proven to be stronger in examples $3-4$ to find new results and examples $5-7$



Notes: Four types of errors are analyzed, note that actual relations are considered as positive relations while `Other` is considered as negative: Labelling an `Other` relation as a positive relation; Labelling a positive relation as `Other`; Confusion between types of relations; Direction errors.

Figure 4.3: Comparing the effects of using RbSP in two aspects, (i) RbSP improved performance and (ii) RbSP yielded some additional wrong results.

to fix wrong results. In our statistic, the use of RbSP bring the big advantage for the relations `Component-Whole`, `Instrument-Agency`, `Entity-Destination`, `Message-Topic` and `Product-Producer`. The results are almost constant for `Member-Collection` relations. Vice versa, we regret to state that using RbSb brings some worse results (examples 8 – 11), especially for `Cause-Effect` and `Content-Container` relations.

Many errors seem attributable to the parser or our model's limitations that still cannot be overcome by using the RbSP (Examples 12 – 13). We listed here some highlight problems to prioritize future researches (a) information on the SDP and its child nodes is still insufficient or redundant to make the correct prediction, (b) the direction of relations is still challenging since some errors appeared because we predict the relation correctly but its direction wrongly (c) the over-fitting problem (leading to wrong prediction - `FP`) and (d) lacking in generality (cannot predict new relation - `FN`).

Table 4.6: The examples of error from RbSP and Baseline models.

#	SID[†]	SDP	Label[*] Golden	RbSP	Baseline
1	8652	**Heating** *prep:with* **wood**	**Other**	**Other**	IA-21
2	10402	**officer** *prep:of* **college**	**Other**	**Other**	MC-12
3	9728	**news** *acl* **crashed** *nsubj* **plane**	**MT-12**	**MT-12**	Other
4	8421	**lane** *prep:on* **road**	**CW-12**	**CW-12**	Other
5	9092	**hurts** *prep:from* **memories**	**EO-12**	**EO-12**	CE-21
6	8081	**bar** *prep:of* **seats**	**CW-12**	**CW-12**	MC-21
7	10457	**show** *nsubj* **offers** *dobj* **discussion**	**MT-12**	**MT-12**	MT-21
8	10567	**stand** *prep:against* **violence**	**Other**	MT-12	**Other**
9	10296	**fear** *prep:from* **robbers**	**CE-21**	Other	**CE-21**
10	9496	**casket** *nsubjpass* **placed** *prep:inside* **casket**	**CC-12**	ED-12	**CC-12**
11	9734	**documents** *acl* **discussed** *prep:at* **meeting**	**MT-21**	MT-12	**MT-21**
12	9692	**rhyme** *prep:by* **thing**	**PP-12**	Other	Other
13	10562	**profits** *prep:from* **inflation**	**Other**	CE-21	CE-21

Notes: The predicted labels are from the best runs. [†]SIDs are sentence IDs in the testing dataset. [*]Abbreviation of relations: `CC (Content-Container)`, `CE (Cause-Effect)`, `CW (Component-Whole)`, `ED (Entity-Destination)`, `EO (Entity-Origin)`, `IA (Instrument-Agency)`, `MC (Member-Collection)`, `MT (Message-Topic)`, `PP (Product-Producer)`. [*]Abbreviation of relation directions: `12 (e1,e2)`, `21 (e2,e1)`.

Conclusions

In this thesis, we have presented a neural relation extraction architecture with the compositional representation of the SDP. The proposed model is capable of utilizing the dominant linguistic and architectural features, such as word embeddings, character embeddings, position feature, WordNet and Part-Of-Speech tag. In addition, we have presented RbSP, a novel representation of relation between two nominals in a sentence that overcomes the disadvantages of traditional SDP. Our RbSP is created by using multi-layer attention to choose relevant information to augment a token in SDP from its child nodes. We also improved the attention mechanisms with kernel filters to capture the features on the context vector.

We evaluated our model on SemEval-2010 task 8 dataset and compared with recent state-of-the-art models. The experiments showed that our model outperforms other comparatives. Experiments were also constructed to verify the rationality and effectiveness of each of the model's components and information sources. The results demonstrated the advantage and robustness of our model, includes the LSTM on the original sentence, combination of self-attention and heuristic mechanisms and several augmentation inputs as well. Moreover, the results on BioCreative V Track 3 CDR also demonstrated the adaptability of our model on classifying many types of relation in different domains.

We also investigated and analyzed the results to find out some weaknesses of the model. Our limitation of cross-sentence relations extraction is highlighted since it resulted in low performance on the BioCreative V Track 3 CDR corpus compared to state-of-the-art results which handled this problem significantly. Although the lack of supported information in SDP is handled by attentive augmentation, the SDP and its child nodes are still insufficient or redundant to make the correct prediction. The direction of relation is still challenging since some errors appeared because we predict the relation correctly but its direction wrongly. We aim to address them and further extensions of our model in future work.

List of Publications

[1] **Duy-Cat Can**, Hoang-Quynh Le, Quang-Thuy Ha, and Nigel Collier. "A Richer-but-Smarter Shortest Dependency Path with Attentive Augmentation for Relation Extraction." In *The 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HTL)*, 2019, (In Press).

[2] **Duy-Cat Can**, Hoang-Quynh Le, and Quang-Thuy Ha. "Improving Semantic Relation Extraction System with Compositional Dependency Unit on Enriched Shortest Dependency Path." In *The 11th Asian Conference on Intelligent Information and Database Systems (ACIIDS)*, pp. 140-152, Springer, 2019.

[3] Trang M. Nguyen, Van-Lien Tran, **Duy-Cat Can**, Quang-Thuy Ha, Ly T. Vu, and Eng-Siong Chng. "QASA: Advanced Document Retriever for Open-Domain Question Answering by Learning to Rank Question-Aware Self-Attentive Document Representations." In *Proceedings of the 3rd International Conference on Machine Learning and Soft Computing*, pp. 221-225. ACM, 2019.

[4] **Duy-Cat Can**, Thi-Nga Ho, and Eng-Siong Chng. "A hybrid deep learning architecture for sentence unit detection." In *Proceedings of the 2018 International Conference on Asian Language Processing (IALP)*, pp. 129-132. IEEE, 2018.

[5] Thi-Nga Ho, **Duy-Cat Can**, and Eng-Siong Chng. "An investigation of word embeddings with deep bidirectional lstm for sentence unit detection in automatic speech transcription." In *Proceedings of the International Conference on Asian Language Processing (IALP)*, pp. 139-142. IEEE, 2018.

[6] Hoang-Quynh Le, **Duy-Cat Can**, Sinh T. Vu, Thanh Hai Dang, Mohammad Taher Pilehvar, and Nigel Collier. "Large-scale exploration of neural relation classification architectures." In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2266-2277. 2018.

References

[1] A. B. Abacha and P. Zweigenbaum, “A hybrid approach for the extraction of semantic relations from medline abstracts,” in *International conference on intelligent text processing and computational linguistics*. Springer, 2011, pp. 139–150.

[2] E. Agichtein and L. Gravano, “Snowball: Extracting relations from large plain-text collections,” in *Proceedings of the fifth ACM conference on Digital libraries*. ACM, 2000, pp. 85–94.

[3] A. Airola, S. Pyysalo, J. Björne, T. Pahikkala, F. Ginter, and T. Salakoski, “All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning,” *BMC bioinformatics*, vol. 9, no. 11, p. S2, 2008.

[4] I. Augenstein, M. Das, S. Riedel, L. Vikraman, and A. McCallum, “Semeval 2017 task 10: Scienceie-extracting keyphrases and relations from scientific publications,” in *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 2017, pp. 546–555.

[5] N. Bach and S. Badaskar, “A review of relation extraction,” *Literature review for Language and Statistics II*, vol. 2, 2007.

[6] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” in *Proceedings of the International Conference on Learning Representations*, 2015.

[7] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, “Open information extraction from the web.” in *IJCAI*, vol. 7, 2007, pp. 2670–2676.

[8] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, “A neural probabilistic language model,” *Journal of machine learning research*, vol. 3, no. Feb, pp. 1137–1155, 2003.

[9] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information,” *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.

[10] Y.-L. Boureau, J. Ponce, and Y. LeCun, “A theoretical analysis of feature pooling in visual recognition,” in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 111–118.

[11] S. Brin, “Extracting patterns and relations from the world wide web,” in *International workshop on the world wide web and databases*. Springer, 1998, pp. 172–183.

[12] R. C. Bunescu and R. J. Mooney, “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, 2005, pp. 724–731.

[13] J. D. Burger, E. Doughty, R. Khare, C.-H. Wei, R. Mishra, J. Aberdeen, D. Tresner-Kirsch, B. Wellner, M. G. Kann, Z. Lu *et al.*, “Hybrid curation of gene–mutation relations combining automated extraction and crowdsourcing,” *Database*, vol. 2014, 2014.

[14] R. Cai, X. Zhang, and H. Wang, “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)*, vol. 1, 2016, pp. 756–765.

[15] R. Caruana, S. Lawrence, and C. L. Giles, “Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping,” in *Advances in neural information processing systems*, 2001, pp. 402–408.

[16] Y. S. Chan and D. Roth, “Exploiting background knowledge for relation extraction,” in *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 152–160.

[17] E. S. Chen, G. Hripcsak, H. Xu, M. Markatou, and C. Friedman, “Automated acquisition of disease–drug knowledge from biomedical and clinical documents: an initial study,” *Journal of the American Medical Informatics Association*, vol. 15, no. 1, pp. 87–98, 2008.

[18] T. Ching, D. S. Himmelstein, B. K. Beaulieu-Jones, A. A. Kalinin, B. T. Do, G. P. Way, E. Ferrero, P.-M. Agapow, M. Zietz, M. M. Hoffman *et al.*, "Opportunities and obstacles for deep learning in biology and medicine," *Journal of The Royal Society Interface*, vol. 15, no. 141, p. 20170387, 2018.

[19] N. Collier, M.-V. Tran, H.-Q. Le, A. Oellrich, A. Kawazoe, M. Hall-May, and D. Rebholz-Schuhmann, "A hybrid approach to finding phenotype candidates in genetic texts," *Proceedings of COLING 2012*, pp. 647–662, 2012.

[20] M.-C. De Marneffe and C. D. Manning, "The stanford typed dependencies representation," in *Coling 2008: proceedings of the workshop on cross-framework and cross-domain parser evaluation*. Association for Computational Linguistics, 2008, pp. 1–8.

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

[22] K. Fundel, R. Küffner, and R. Zimmer, "Relex—relation extraction using dependency parse trees," *Bioinformatics*, vol. 23, no. 3, pp. 365–371, 2006.

[23] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[24] Y. Goldberg, "Neural network methods for natural language processing," *Synthesis Lectures on Human Language Technologies*, vol. 10, no. 1, pp. 1–309, 2017.

[25] J. Gu, F. Sun, L. Qian, and G. Zhou, "Chemical-induced disease relation extraction via convolutional neural network," *Database*, vol. 2017, 04 2017.

[26] Z. GuoDong, S. Jian, Z. Jie, and Z. Min, "Exploring various knowledge in relation extraction," in *Proceedings of the 43rd annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2005, pp. 427–434.

[27] T. Hasegawa, S. Sekine, and R. Grishman, "Discovering relations among named entities from large corpora," in *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2004, p. 415.

[28] M. A. Hearst, “Automatic acquisition of hyponyms from large text corpora,” in *Proceedings of the 14th conference on Computational linguistics-Volume 2*. Association for Computational Linguistics, 1992, pp. 539–545.

[29] I. Hendrickx, S. N. Kim, Z. Kozareva, P. Nakov, D. Ó Séaghdha, S. Padó, M. Pennacchiotti, L. Romano, and S. Szpakowicz, “Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals,” in *Proceedings of the Workshop on Semantic Evaluations*, 2009, pp. 94–99.

[30] I. Hendrickx, S. N. Kim, Z. Kozareva, P. Nakov, D. O. Séaghdha, S. Padó, M. Pennacchiotti, L. Romano, and S. Szpakowicz, “Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals,” in *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*. Association for Computational Linguistics, 2009, pp. 94–99.

[31] M. Herrero-Zazo, I. Segura-Bedmar, P. Martínez, and T. Declerck, “The ddi corpus: An annotated corpus with pharmacological substances and drug–drug interactions,” *Journal of biomedical informatics*, vol. 46, no. 5, pp. 914–920, 2013.

[32] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[33] S. Ioffe and C. Szegedy, “Batch normalization: accelerating deep network training by reducing internal covariate shift,” in *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*. JMLR.org, 2015, pp. 448–456.

[34] S. Jat, S. Khandelwal, and P. Talukdar, “Improving distantly supervised relation extraction using word and entity based attention,” in *6th Workshop on Automated Knowledge Base Construction (AKBC) at NIPS 2017*, 2017.

[35] R. Javed, S. Farhan, and S. Humdullah, “A hybrid approach based on pattern recognition and bionlp for investigating drug-drug interaction,” *Current Bioinformatics*, vol. 10, no. 3, pp. 315–322, 2015.

[36] N. Kambhatla, “Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction,” in *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, 2004.

[37] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[38] H.-Q. Le, M.-V. Tran, T. H. Dang, Q.-T. Ha, and N. Collier, "Sieve-based coreference resolution enhances semi-supervised learning model for chemical-induced disease relation extraction," *Database*, vol. 2016, 07 2016.

[39] H.-Q. Le, D.-C. Can, S. T. Vu, T. H. Dang, M. T. Pilehvar, and N. Collier, "Large-scale exploration of neural relation classification architectures," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 2266–2277.

[40] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[41] G. Leroy and H. Chen, "Genescene: An ontology-enhanced integration of linguistic and co-occurrence based relations in biomedical texts," *Journal of the American Society for Information Science and Technology*, vol. 56, no. 5, pp. 457–468, 2005.

[42] T. S. Li, À. Bravo, L. I. Furlong, B. M. Good, and A. I. Su, "A crowdsourcing workflow for extracting chemical-induced disease relations from free text," *Database*, vol. 2016, 2016.

[43] D. Lin and P. Pantel, "Dirt@ sbt@ discovery of inference rules from text," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 323–328.

[44] Y. Liu, F. Wei, S. Li, H. Ji, M. Zhou, and W. Houfeng, "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)*, vol. 2, 2015, pp. 285–290.

[45] T. Mavropoulos, D. Liparas, S. Symeonidis, S. Vrochidis, and I. Kompatsiaris, "A hybrid approach for biomedical relation extraction using finite state automata and random forest-weighted fusion," in *International Conference on Computational Linguistics and Intelligent Text Processing*. Springer, 2017, pp. 450–462.

[46] D. McClosky, M. Surdeanu, and C. D. Manning, "Event extraction as dependency parsing," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011, pp. 1626–1635.

[47] F. Mehryary, J. Björne, S. Pyysalo, T. Salakoski, and F. Ginter, "Deep learning with minimal training data: Turkunlp entry in the bionlp shared task 2016," in *Proceedings of the 4th BioNLP Shared Task Workshop*, 2016, pp. 73–81.

[48] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[49] T. H. Nguyen and R. Grishman, "Relation extraction: Perspective from convolutional neural networks," in *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, 2015, pp. 39–48.

[50] N. C. Panyam, K. Verspoor, T. Cohn, and K. Ramamohanarao, "Exploiting graph kernels for high performance biomedical relation extraction," *Journal of biomedical semantics*, vol. 9, no. 1, p. 7, 2018.

[51] G. A. Pavlopoulos, V. J. Promponas, C. A. Ouzounis, and I. Iliopoulos, "Biological information extraction and co-occurrence analysis," in *Biomedical Literature Mining*. Springer, 2014, pp. 77–92.

[52] Y. Peng, M. Torii, C. H. Wu, and K. Vijay-Shanker, "A generalizable nlp framework for fast development of pattern-based biomedical relation extraction systems," *BMC bioinformatics*, vol. 15, no. 1, p. 285, 2014.

[53] C. Quan, M. Wang, and F. Ren, "An unsupervised text mining method for relation extraction from biomedical literature," *PloS one*, vol. 9, no. 7, p. e102039, 2014.

[54] C. E. Rasmussen, "Gaussian processes in machine learning," in *Advanced lectures on machine learning*. Springer, 2004, pp. 63–71.

[55] T. C. Rindflesch and M. Fiszman, "The interaction of domain knowledge and linguistic structure in natural language processing: interpreting hypernymic propositions in biomedical text," *Journal of biomedical informatics*, vol. 36, no. 6, pp. 462–477, 2003.

[56] B. Rink and S. Harabagiu, "Utd: Classifying semantic relations by combining lexical and semantic resources," in *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, 2010, pp. 256–259.

[57] G. Rosemblat, D. Shin, H. Kilicoglu, C. Sneiderman, and T. C. Rindflesch, "A methodology for extending domain coverage in semrep," *Journal of biomedical informatics*, vol. 46, no. 6, pp. 1099–1107, 2013.

[58] B. Rosenfeld and R. Feldman, "Ures: an unsupervised web relation extraction system," in *Proceedings of the COLING/ACL on Main conference poster sessions*. Association for Computational Linguistics, 2006, pp. 667–674.

[59] Y. Shen and X. Huang, "Attention-based convolutional neural network for semantic relation extraction," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 2526–2536.

[60] R. Socher, B. Huval, C. D. Manning, and A. Y. Ng, "Semantic compositionality through recursive matrix-vector spaces," in *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*. ACL, 2012, pp. 1201–1211.

[61] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[62] Y. Su, H. Liu, S. Yavuz, I. Gur, H. Sun, and X. Yan, "Global relation embedding for relation extraction," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, vol. 1, 2018, pp. 820–830.

[63] P. Verga, E. Strubell, and A. McCallum, "Simultaneously self-attending to all mentions for full-abstract biological relation extraction," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, vol. 1, 2018, pp. 872–884.

[64] L. Wang, Z. Cao, G. de Melo, and Z. Liu, "Relation classification via multi-level attention cnns," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, vol. 1, 2016, pp. 1298–1307.

[65] C.-H. Wei, Y. Peng, R. Leaman, A. P. Davis, C. J. Mattingly, J. Li, T. C. Wiegers, and Z. Lu, "Assessing the state of the art in biomedical relation extraction: overview of the biocreative v chemical-disease relation (cdr) task," *Database*, vol. 2016, 2016.

[66] X. Wei, Q. Zhu, C. Lyu, K. Ren, and B. Chen, "A hybrid method to extract triggers in biomedical events," *Journal of Digital Information Management*, vol. 13, no. 4, p. 299, 2015.

[67] K. Xu, Y. Feng, S. Huang, and D. Zhao, "Semantic relation classification via convolutional neural networks with simple negative sampling," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 536–540.

[68] Y. Xu, L. Mou, G. Li, Y. Chen, H. Peng, and Z. Jin, "Classifying relations via long short term memory networks along shortest dependency paths," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 1785–1794.

[69] Y. Yan, N. Okazaki, Y. Matsuo, Z. Yang, and M. Ishizuka, "Unsupervised relation extraction by mining wikipedia texts using information from the web," 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, 2009, pp. 1021–1029.

[70] D. Zeng, K. Liu, S. Lai, G. Zhou, and J. Zhao, "Relation classification via convolutional deep neural network," in *Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers*, 2014, pp. 2335–2344.

[71] R. Zhang, F. Meng, Y. Zhou, and B. Liu, "Relation classification via recurrent neural network with attention and tensor layers," *Big Data Mining and Analytics*, vol. 1, no. 3, pp. 234–244, 2018.

[72] X. Zhang, F. Chen, and R. Huang, "A combination of rnn and cnn for attention-based relation classification," *Procedia computer science*, vol. 131, pp. 911–917, 2018.

[73] Y. Zhang, V. Zhong, D. Chen, G. Angeli, and C. D. Manning, "Position-aware attention and supervised data improve slot filling," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 35–45.

[74] Y. Zhang, P. Qi, and C. D. Manning, "Graph convolution over pruned dependency trees improves relation extraction," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

[75] S. Zhao and R. Grishman, “Extracting relations with integrated information using kernel methods,” in *Proceedings of the 43rd annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2005, pp. 419–426.

[76] H. Zhou, H. Deng, L. Chen, Y. Yang, C. Jia, and D. Huang, “Exploiting syntactic and semantics information for chemical–disease relation extraction,” *Database*, vol. 2016, 04 2016.

[77] P. Zhou, W. Shi, J. Tian, Z. Qi, B. Li, H. Hao, and B. Xu, “Attention-based bidirectional long short-term memory networks for relation classification,” in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, vol. 2, 2016, pp. 207–212.