UMBC

DISSERTATION APPROVAL SHEET

Title of Dissertation: Combining Text Embedding with Additional Knowledge for Information Extraction

Name of Candidate: Arpita Roy

Doctor of Philosophy, 2021

Graduate Program: Information Systems

Dissertation and Abstract Approved:

Shimei Pan

Shimei Pan

Associate Professor

Department of Information Systems

7/27/2021 | 12:54:53 AM EDT

NOTE: *The Approval Sheet with the original signature must accompany the thesis or dissertation. No terminal punctuation is to be used.

Arpita Roy

+1-702-445-9989 arpita.r00y@gmail.com

Education

- University of Maryland, Baltimore County, Baltimore, MD, USA
  Ph.D. in Information Systems, August 2021
  Advisor – Dr. Shimei Pan
  Dissertation: Combining Text Embedding with Additional Knowledge for Information Extraction
- University of Maryland, Baltimore County, Baltimore, MD, USA
  M.S. in Information Systems, May 2018, GPA: 4.00/4.00
- Bangladesh University of Engineering and Technology, Dhaka, Bangladesh.
  B.Sc. in Computer Science and Engineering, February 2013, GPA: 3.55/4.00

Research Interests

- Natural Language Processing, Machine Learning, Deep Learning, Data Science

Publications

- **Arpita Roy**, Shimei Pan. "Incorporating medical knowledge in BERT for clinical relation extraction" *(under EMNLP 2021 review*)
- **Arpita Roy** and Shimei Pan. "Incorporating Extra Knowledge to Enhance Word Embedding", *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2020
- Fatema Hasan, **Arpita Roy**, and Shimei Pan. "Integrating Text Embedding with Traditional NLP Features for Clinical Relation Extraction", *IEEE International Conference on Tools with Artificial Intelligence (ICTAI), 2020*
- **Arpita Roy**, Youngja Park, Taesung Lee, and Shimei Pan. "Supervising Unsupervised Open Information Extraction Models", *Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language (EMNLP-IJCNLP)*, 2019
- **Arpita Roy**, Youngja Park and Shimei Pan. "Predicting Malware Attributes from Cybersecurity Texts", *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 2019*
- **Arpita Roy**, Youngja Park and Shimei Pan. "Incorporating Domain Knowledge in Learning Word Embedding", *IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, 2019
- Ankur Padia, **Arpita Roy**, Taneeya W. Satyapanich, Francis Ferraro, Shimei Pan, Youngja Park, Anupam Joshi, and Tim Finin "UMBC at SemEval-2018 Task 8: Understanding Text about Malware"**,** *International Workshop on Semantic Evaluation (SemEval)*, 2018
- **Arpita Roy**, Anamika Paul, Hamed Pirsiavash and Shimei Pan. "Automated Detection of Substance Use-Related Social Media Posts Based on Image and Text Analysis", *Tools with Artificial Intelligence (ICTAI), 2017*
- Tao Ding, **Arpita Roy**, Zhiyuan Chen, Qian Zhu and Shimei Pan. "Analyzing and Retrieving Illicit Drug-Related Posts from Social Media", *Bioinformatics and Biomedicine (BIBM), 2016*
- **Arpita Roy** and Shimei Pan. "Predicting Gene Functional Interactions with Semantic Word Embedding", *IEEE International Conference on Tools with Artificial Intelligence (ICTAI), 2016*
- Shaheena Sultana, Md Saidur Rahman, **Arpita Roy**, and Suraiya Tairin. "Bar 1-visibility drawings of 1-planar graphs", *International Conference on Applied Algorithms (ICAA)*, 2014

Technical Skills

- Machine Learning Tools: Keras, Tensorflow, Matlab, Weka
- Programming Language: Python, Java, C, C++
- Database Management System: MySQL, Oracle, SQlite, PostgreSQL

Professional Experience

- **IBM**
  Research Intern May 2018 - August 2018
- **University of Maryland, Baltimore County**
  Graduate Research Assistant in Information Systems Department August 2015 – December 2020
- **Primeasia University, Dhaka, Bangladesh**
  Lecturer in Computer Science and Engineering Department June 2014 - July 2015

- **Samsung R&D Institute Bangladesh, Dhaka**
  Software Engineer October 2013 - June 2014

Additional Experience

- *Reviewer* for 30th International Joint Conference on Artificial Intelligence - Survey Track (IJCAI 2021)
- *Reviewer* for ACM Transactions on Interactive Intelligent Systems (ACM TiiS 2021)
- *Program committee member* of IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2020)
- *Program committee member* of the Multidisciplinary Track of 2019 IEEE/ACM international Conference on Advances in Social Networks Analysis and Mining (ASONAM 2019)
- *Student organizer* of Mid-Atlantic Student Colloquium on Speech, Language and Learning (MASC-SLL-2018)

Awards & Honors

- Best Student Paper Award – ICTAI 2017
- Grace Hopper Scholar - 2017
- Dean List Award – 2010

ABSTRACT

Title of dissertation: COMBINING TEXT EMBEDDING WITH ADDITIONAL KNOWLEDGE FOR INFORMATION EXTRACTION

Arpita Roy, Doctor of Philosophy, 2021

Dissertation directed by: Dr. Shimei Pan
Department of Information Systems
University of Maryland, Baltimore County (UMBC)

Information Extraction (IE) is an essential field of natural language processing (NLP). Over the years, researchers have studied numerous approaches and techniques to meet the challenges of different IE tasks. This dissertation explores various knowledge fusion techniques to combine diverse domain-independent and domain-specific knowledge with multiple text embedding techniques for effective IE. Specifically, this work presents a systematic investigation to combine different types of knowledge (e.g., lexical, syntactic, semantic, and domain knowledge) with different text embedding techniques (e.g., static and contextual embeddings) to achieve the state of the art performance in several IE tasks (e.g., Open IE, malware attribute identification and clinical relation extraction).

COMBINING TEXT EMBEDDING WITH ADDITIONAL KNOWLEDGE FOR INFORMATION EXTRACTION

by

Arpita Roy

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, Baltimore County in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2021

Advisory Committee:
Dr. Shimei Pan, Chair/Advisor
Dr. Zhiyuan Chen
Dr. James Foulds
Dr. Tim Finin
Dr. Youngja Park

Dedication

To my parents.

Acknowledgments

I owe my gratitude to all the people who have made this thesis possible.

First and foremost, I would like to thank my advisor, Dr. Shimei Pan, for guidance, encouragement and support. I cannot begin to express how grateful I am to have her as my advisor. It has been a pleasure to work with and learn from such an extraordinary researcher. I want to thank Dr. Zhiyuan Chen, Dr. Tim Finin, Dr. James Foulds and Dr. Youngja Park for taking time out of their busy schedule to be on my thesis committee. Many thanks to my mentors Dr. Youngja Park and Dr. Taesung Lee at IBM Research, for giving me a wonderful opportunity to work with them during my internship at IBM. I would like to thank Dr. Anupam Joshi, Dr. Karuna Joshi, Dr. Tim Finin and Dr. Frank Ferraro for their valuable feedback while working on the IBM Accelerated Cognitive Cybersecurity project. I would like to acknowledge the IBM Accelerated Cognitive Cybersecurity Laboratory Grant for supporting my research.

Special thanks to Fatema Hasan, Tao Ding and Philip Feldman for all the collaborations, discussions and suggestions.

I want to thank my friends for lifting my spirit in stressful times.

Finally, I would like to thank my family. I am incredibly thankful to my mother, Namita Roy, and my father, Sanjoy Roy, for their endless love, support and encouragement. I owe all my success to them. I want to mention my dearest younger brother, Sudipta. Last but not least, Shawoon, my partner, thank you so much for being there for me.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Overview of Information Extraction

As a significant part of human knowledge is captured in text, we must process and extract the knowledge in texts efficiently. As the amount of text available in electronic form is overwhelming and increasing every day, manually managing all the text and making sense of it is time-consuming and unscalable. The noisiness and flexibility of natural language also pose a significant challenge to analyzing the information in text using traditional statistical analysis and data mining methods. To access information from a vast amount of unstructured text effectively, we need efficient and accurate methods that can extract structured information from unstructured text automatically. This explosion of machine-readable text and the need to access information expressed in unstructured text gives rise to Information Extraction (IE), a natural language processing (NLP) technique that automatically extracts structured information from unstructured text. IE systems are frequently used to extract entities, attributes, relations, and events from text, which can then be stored in a structured format to facilitate downstream data query and analysis with traditional statistical or data-mining techniques. Typical IE applications range from business intelligence [1], financial investigation [2, 3], scientific research [4, 5],

healthcare record management [6–8], resume harvesting [9], social media analysis [10], sentiment analysis [11], email scanning [12] and many more.

1.2 Text Embedding for IE

As in many NLP tasks, one of the major challenges for IE is input text representation. As words can not be directly processed or understood by a computer, obtaining effective representations of text that capture important lexical, syntactic, semantic and contextual information in input text has long been a research focus in NLP. Like many NLP tasks, IE needs meaningful word, phrase and sentence representations. In many early NLP systems, a word is often represented as a one-hot vector. Since the construction of such a vector is simple, this method and other similar techniques like tf-idf and n-gram vectors have been widely used in many NLP tasks. However, these representations have obvious shortcomings like the curse of dimensionality, poor scalability, feature sparsity and being incapable of representing syntactic, semantic and contextual information in the text. For instance, in typical NLP tasks, the number of unique words can be millions. As a result, the size of corresponding one-hot vectors becomes extremely large, leading to an inefficient representation of words. In addition, these discrete vectors are unable to capture the syntactic, semantic and contextual relations between words. To resolve these issues, recent NLP research focuses on learning low-dimensional continuous vector representations of words, also known as word embedding. Word embedding is a process to automatically map the words in a vocabulary into dense vectors of real numbers

in a continuous embedding space. Word embedding is learned based on word co-occurrences so that semantically and contextually related words have similar vector representations. Figure 1.1 shows an example of word embedding visualization. Word embedding is mostly derived from large text corpora in an unsupervised/self-supervised manner. It has gained much attention and popularity as the derived word vectors often encode essential syntactic and semantic information that is useful for NLP. Research has shown that word embedding boost the performance of diverse downstream NLP tasks such as sentiment analysis [13], text classification [14] and question answering [15]. Significant improvement has also been achieved with word embedding for various IE tasks such as named entity recognition [16] and relation extraction [17].

According to the methodologies used, word embedding can be divided into two main categories: (1) static embedding and (2) contextual embedding.

Static embedding only learns a single context-independent representation for each word. This implies that each word has only one meaning regardless of its context. In reality, however, the meaning of a word may change with its context. For example, the word "bank" has two meanings; one is related to a "financial institution" and the other is related to the "land alongside rivers." However, in static embedding, "bank" has only one embedding representation regardless of its sentence and application context. Commonly used static embedding methods include Word2Vec [18], Glove [19] and FastText [20].

Contextual embedding models generate a word representation that is a function of its context (e.g., the sentence in which it appears). Using the same

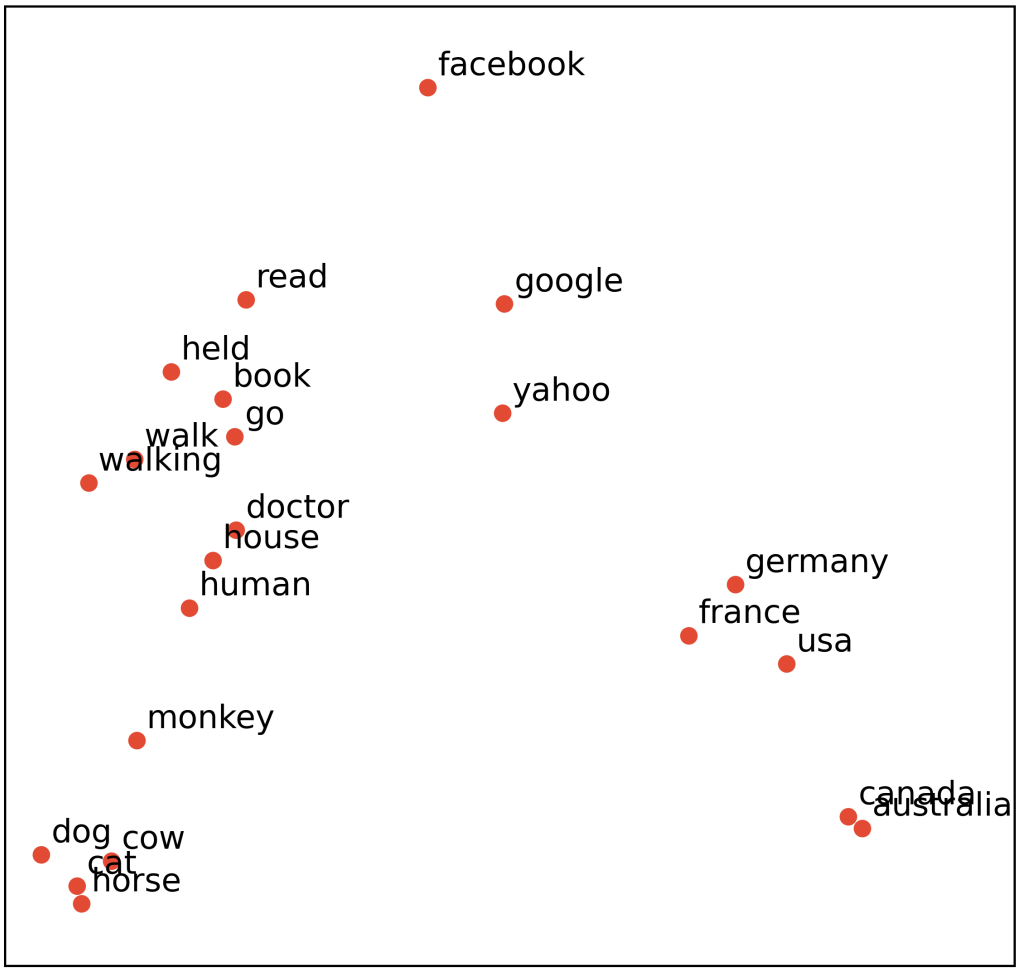


Figure 1.1: Word Embedding Visualization

example, the word "bank" may have many dynamic embedding representations; each is tailored to its sentence and domain contexts. Recent research has shifted chiefly towards contextual embedding. CoVe [21], context2vec [22] and [23] are some of the earlier efforts to learn context dependent representation. These researches were followed by ELMo (Embeddings from Language Models) [24], GPT (Generative Pre-trained Transformer Models) [25] and BERT(Bidirectional Encoder Representations from Transformers) [26]. BERT is the most popular contextual embedding technique to generate text embeddings for input texts.

The main differences between static and contextual embedding include:

- Static word embedding models cannot represent word polysemy. Contextual word embedding models can address this issue by generating context-dependent word representations.

- Static embedding is frequently trained with shallow neural networks, while contextual embedding is trained using deep neural networks.

- Frequently downstream NLP tasks utilize pre-stored word embedding vectors learned by a static embedding model because the number of unique words in a vocabulary is limited. In contrast, with contextual embedding, since the representation of a word varies with its context, each word may be represented by a large number of word vectors. As a result, downstream NLP tasks often rely on a pre-trained contextual embedding model to dynamically generate context-sensitive word vectors.

- Contextual embedding models are more computationally expensive to train than static embedding models because the number of model parameters in a contextual embedding model is frequently much larger than that of a static embedding model.

1.3 Additional Knowledge for IE

Types of knowledge explored in IE systems are domain-independent knowledge and domain-specific knowledge.

Domain-independent knowledge that can be automatically inferred by standard NLP tools such as part-of-speech tags [27], dependency-tree paths between entities [28], constituent-tree paths between entities [29] and syntactic roles [30]. Domain-independent features are more likely to be either surface or syntactic features as existing NLP tools can more reliably infer them. They are also more generalizable across different domains.

Domain-specific knowledge such as domain lexicons [31] and domain ontology [32] has been used in various IE tasks. Domain knowledge may provide new information about the semantic types of words, explain the meaning of domain terms and indicate the semantic relationships between domain concepts. For example, based on the information in a biomedical knowledge base UMLS [33] an IE system may know that "fever" has a concept definition of "abnormal elevation of body temperature, usually as a result of a pathologic process"; "fever" may be treated by "ibuprofen"; "fever" is a "sign" and "symptom"; "hyperpyrexia" is a type of "fever"; and "fever" is interpreted as "abnormal body temperature". Thus, incorporating additional knowledge from such a biomedical knowledge base may help an IE system better understand the entities and relations expressed in clinical texts. Domain-specific knowledge is also unlikely to be generalizable across different domains.

1.4 Combining Text Embedding with Additional Knowledge

While word embeddings are powerful tools for representing input texts, using only word embeddings might be insufficient to extract all the necessary information. As a result, in this research, I explore, besides text embeddings, whether additional knowledge can be incorporated to help improve the performance of IE systems.

Static embedding models such as Word2Vec capture context similarity between words. That is, words that appear in similar contexts often have similar meanings. Prior research suggests that static embeddings may capture some morphological (e.g., walk and walking) and syntactic relations between words [34]. However, the syntactic information captured in static word embeddings may not be as powerful as those uncovered via the state-of-the-art syntactic parsers trained on a large number of supervised examples [34]. Since dependency tree paths are often used in different IE tasks like named entity recognition [35] and relation extraction [28], it is possible that additional information from a parse tree may help improve the performance of an IE system than those that only use static word embeddings.

Recently, the efforts on text embeddings have been mostly shifted to contextual embedding models that are trained on massive amounts of texts. As their context heavily influences the dynamic embeddings of words, it is even harder to decipher the type of information captured in dynamic embeddings. For example, it is unclear whether syntactic information is adequately captured in BERT embeddings and whether additional information from a syntactic parser may help improve IE performance when they are combined with BERT embeddings. Due to their

superb performance in diverse NLP tasks, it has been speculated that given enough parameters and training data, contextual embedding models such as BERT might be able to make external knowledge augmentation unnecessary [36].

IE in specialized domains (e.g., cyber-security and bio-medical) presents new challenges as text in these domains often has unique characteristics. Usually, domain text contains domain-specific terms, which may not occur frequently in general domain text corpora. For example, the term "Malignant Neoplasms" which means "Cancer" in bio-medical text, may not be easily found in general domain text. Moreover, there may exist domain-specific relation among domain concepts. For example, the word "Virus" is related to "disease" in the biomedical domain and "software" in the cyber-security domain. Often there are many synonyms referring to the same concept in domain-specific texts. For example in biomedical text, "Malignant Neoplasms", "Neoplasm, Malignant","Malignant tumour", "Malignancy", "Cancer", "Cancers", and "CA" all refer to same the concept. Additionally, abbreviations, acronyms, shorthand lexical units and spelling variations are heavily used in clinical texts. Additional domain knowledge such as the knowledge encoded in a domain ontology may help an IE system correctly identify domain concepts and relations.

Typical techniques to combine additional knowledge with text embeddings for IE can be categorized into two classes: (1) feature concatenation and (2) direct knowledge injection into text embeddings. Feature concatenation is a simple yet effective way to combine various syntactic, semantic and lexical features with text embeddings. Concatenation provides the flexibility to combine multiple types

of features all within one framework without requiring feature-specific handling. Knowledge injection refers to directly incorporating knowledge into the same text embedding space. This technique enables to generate text embedding based on both text and knowledge. Injecting knowledge into text embedding may help to encode domain semantics effectively. Knowledge injection can be implemented during the training of embedding models or as a post-processing step.

1.5 IE Applications

IE has a wide range of applications such as merge and acquisition event extraction from financial reports [2,3], medical information extraction from healthcare records [6–8] and terrorist event extraction and tracking from news [37,38]. In the following, I briefly introduce the main IE applications studied in this thesis.

Open IE: Open IE focuses on extracting relations from text [39]. Typically, Open IE systems read one sentence and extract tuples with one relation phrase and two or more arguments. Open IE is necessary when the number of extracted relations is large and the relations themselves are unknown. As Open IE does not require the specification of target relations in advance, it can facilitate domain-independent knowledge discovery from the text.

IE for Malware Attribute Identification: Cybersecurity is one of the highly specialized domains that require focused effort and specialized techniques for effective IE. There is a critical need to aid cybersecurity analysts in processing a large amount of online text and extracting targeted information such as mal-

ware attributes, exploits, and security vulnerabilities. In the cybersecurity domain, automated malware attribute identification based on texts is vital for immediate awareness of security flaws. For example, extracting attributes like malware names, capabilities, and operating strategies can help cybersecurity specialists monitor malware's appearance and spread.

IE for Clinical Relation Extraction: The clinical text contains valuable information about a patient's conditions such as symptoms, diagnoses and treatments. Hence identifying, extracting, and mining this information is of great importance to managing and improving patient care. One of the fundamental tasks of clinical NLP is extracting entities and semantic relations between medical concepts from clinical notes (e.g., extracting relations between diseases and treatments). Identifying these relations is important for caregivers to understand how patients respond to treatments and facilitate clinical decision-making.

1.6 Thesis Statement

Since the word embedding features are latent, it is hard to know exactly what knowledge has been captured in the embedding representations and whether the information needed for IE has been encoded properly and adequately. The main research question explored in this thesis is:

Given the power of text embeddings, do we still need additional knowledge for IE?

1.7 Thesis Contributions

The main contributions of this thesis include:

- I develop a novel Open IE system employing static word embedding and domain-independent knowledge, including part-of-speech tags, syntactic roles, dependency tree information and output of existing Open IE systems. In this paradigm, the proposed model can take advantage of the semantics captured in word embedding, additional knowledge about sentence structure, and the outputs of existing Open IE models. Evaluations conducted on several benchmark datasets show that the proposed method outperforms the baseline systems by a large margin.

- I develop a novel *Word Annotation Embedding (WAE)* algorithm to incorporate domain-specific semantic information into static word embeddings. Knowledge utilized here includes diverse heterogeneous knowledge sources such as human annotations, raw texts and specifications from domain experts. Then the enriched word embeddings are used to build a *malware attribute identification system* with the state of the art performance.

- I perform an experimental study to investigate whether domain independent syntactic features can be combined with contextual embeddings generated by BERT to improve relation extraction from clinical text.

- I conduct a comprehensive examination of different techniques to add biomedical knowledge into contextual embedding models. I explore how domain-

specific semantic knowledge from UMLS, a biomedical knowledge base, can be added to contextual word embeddings generated by BERT to facilitate clinical relation extraction. The proposed knowledge fusion methods outperformed the state-of-the-art systems on a benchmark clinical relation extraction dataset.

1.8 Thesis Organization

Based on the type of embedding techniques and additional knowledge utilized, the thesis can be organized into four parts: (1) static word embedding with domain-independent knowledge, (2) static word embedding with domain-specific knowledge, (3) contextual embedding with domain-independent knowledge and (4) contextual embedding with domain-specific knowledge. As shown in figure 1.2, I investigate novel methods that systematically combine different types of knowledge with different types of embedding techniques. Additionally, I verify the effectiveness of proposed methods in various IE applications such as OpenIE, malware attribution identification and clinical relation extraction.

The rest of the dissertation is organized as follows. Chapter 2 provides a comprehensive overview of recent research work on different IE tasks. Following chapters explore various methodologies to combine additional knowledge with different text embedding techniques. Chapter 3 details how static word embedding can be combined with domain-independent knowledge extracted by existing NLP tools to improve Open IE performance. Chapter 4 presents a solution to overcome the challenges of domain-specific IE with additional domain-specific semantic

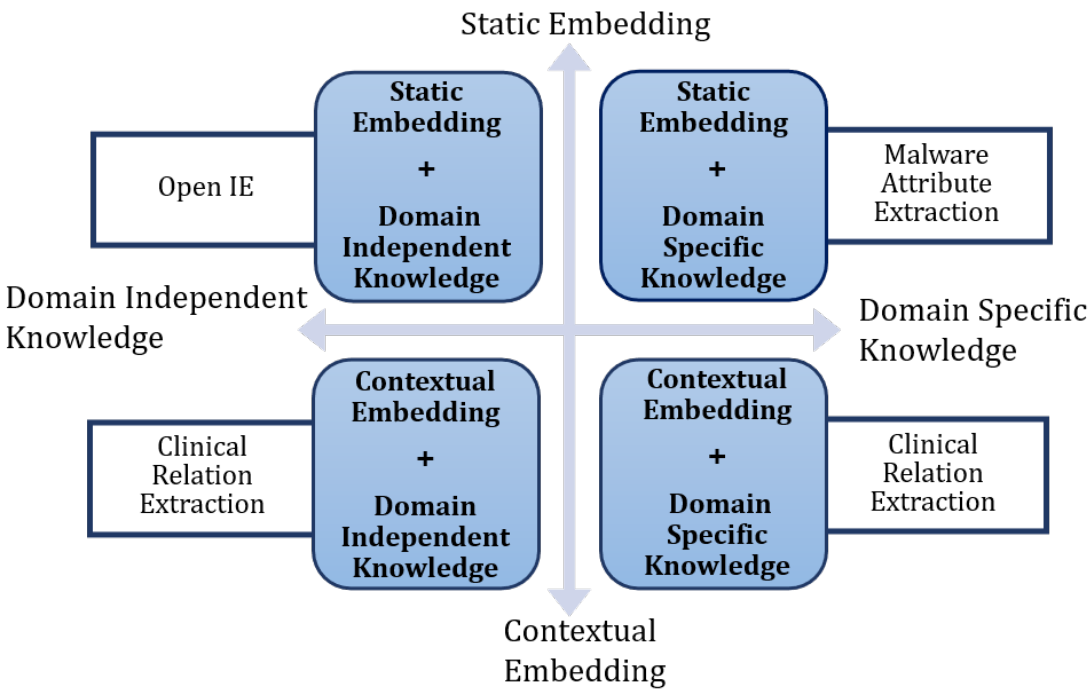


Figure 1.2: Thesis Organization

knowledge and static word embedding for malware attribute identification. Chapter 5 and chapter 6 illustrate novel IE techniques that combine contextual word embedding with additional knowledge for clinical relation extraction. While chapter 5 focuses on incorporating domain-independent knowledge, chapter 6 leverages domain-specific knowledge from an existing bio-medical knowledge base to improve clinical relation extraction. Finally, chapter 7 summarizes the main findings. It also identifies the main limitations of the current research and points out a few future research directions aiming at addressing these issues.

Chapter 2: Related Work

IE is one of the fastest growing fields of NLP. Research in IE has explored a variety of approaches and techniques. The complex task of IE can be divided into several sub-tasks: Named Entity Recognition [40,41], Coreference Resolution [42,43], Relation Extraction [40, 44–46] and Event Detection [47, 48].

To address the needs of diverse applications, the techniques of IE have evolved over the past years. Early systems were Knowledge-based and used crafted rules based on language syntax, grammar, lexical resources and domain specific knowledge [49–52]. As manual coding of rules was tedious, machine learning-based methods were developed [53–57]. Hybrid models [58, 59] attempted to take advantage of both supervised learning methods and rule-based methods. Overall supervised learning remains the dominant approach to solve IE problems. However, annotating training data for each new domain is challenging and time-consuming. This has led to semi-supervised learning [60, 61] and distant supervised approaches [62]. As these learning based models mostly relied on feature engineering, they still required human effort to hand craft features. Recently research has shifted towords feature-inferring neural network systems that use automatically learned text embedding. Studies have shown the importance of text embedding for neural network based

IE systems [63–66]. In the following sections, I survey prior work on topics that are most relevant to this research: (a) knowledge fusion techniques that combine text embeddings with additional knowledge and (b) the state-of-the-art techniques developed for relevant IE applications used in the thesis (i.e., Open IE, IE for cybersecurity and clinical relation extraction).

2.1 Combining Text Embeddings with Additional Knowledge

In recent years text embedding has become the most popular input text representation for IE approaches. Since traditional word embeddings mainly capture the semantic relatedness between co-occurring words in a predefined context, they suffer from certain limitations. For example, it may not capture syntactic or domain-specific semantic information very well. Additional knowledge can help bridge this gap when combined with word embedding and boost the performance of IE systems. Consequently a large portion of recent research has presented approaches to incorporate extra morphological, syntactic, semantic and domain knowledge with word embedding to facilitate IE systems. In this section, I summarize the recent advances in this research area. I include representative work on enhancing both static word embedding (e.g., Word2Vec) where a fixed embedding vector is learned for each word and dynamic/contextual word embedding (e.g., BERT), where the embedding vector for each word varies with its context. I discuss both domain-independent and domain-specific knowledge incorporation using feature concatenation and di-

rect knowledge injection techniques.

2.1.1 Adding Knowledge into Static Embedding

Feature concatenation has been used in studies [67–69] to combine various kinds of knowledge with Word2Vec embedding. To directly incorporate knowledge into static word embedding (Word2Vec), either joint optimization or post-processing is used. In joint optimization, word embedding is trained with both text and extra knowledge simultaneously. These methods frequently modify the training objective to include both the native word embedding objective and a new objective related to the knowledge. The methods can be further categorized into: (1) adding prior or regularization to the original distributed representation learning objective [70, 71]; (2) extending the original distributed representation learning objective to learn additional embedding [15, 72]; (3) adding rank or hierarchical structure constraints [14, 73]; (4) combining training objective with knowledge graph embedding objective [16, 17]; and (5) augmenting the input text with extra knowledge [74–78]. While [16, 17, 71, 72] focused on combining semantic knowledge, [74, 75, 77, 78] aimed to incorporate syntactic knowledge such parts-of-speech tag and dependency parse tree. These models are tied to the distributional objective and any change of the underlying distributional model induces a change in the entire joint model. Post-processing-based methods integrate extra knowledge into pre-trained word embedding. Popular post-processing methods such as retrofitting [79] fine-tuned the original word embeddings so that they satisfy additional constraints generated from

the extra knowledge. The most significant benefit of post-processing over joint optimization is that it can be applied to any pre-trained word embedding models without expensive retraining. Post-processing methods often only locally update word vectors involved in the external constraints, whereas vectors of the other words remain intact. In contrast, joint optimization propagates the influence of external knowledge to all the words via the joint objective.

2.1.2 Adding Knowledge into Contextual Embedding

To integrate knowledge into contextual word embedding (BERT), researchers have followed two methods: (1) fine-tune with domain text and (2) combine knowledge graph into BERT.

Incorporating domain text in BERT: There are quite a few BERT models which have been trained (or fine-tuned) with bio-medical text: BioBERT [80] is pre-trained on PubMed abstracts and PMC full-text articles; ClinicalBERT [81] is pre-trained on the clinic notes in the MIMIC-III database [82]; BlueBERT [83] is pre-trained on the PubMed abstracts and the clinical notes in MIMIC-III; PubMedBERT [84] is pre-trained using abstracts from PubMed and full-text articles from PMC; Scibert [85] is pre-trained on biomedical papers. Among them, BioBERT and BlueBERT are initialized with weights from a general-domain BERT model, ClinicalBERT is initialized from BioBERT, and PubMedBERT is trained from scratch.

Combining knowledge graph information with BERT: The simplest method to combine knowledge graph information with BERT is concatenation. For example, [86] combined knowledge graph embedding trained using Graph Convolution Networks (GCN) with BERT embeddings for citation recommendation. Efforts to directly inject knowledge graph information into BERT can be further categorized into the following categories.

- Joint optimization with knowledge graph objectives: For example, Clinical KB-BERT [87] pre-trained BERT with a knowledge graph objective. In addition to predicting masked words, a triplet classification objective was added, where given a triplet of two concepts and a relation in UMLS, the model aimed to predict if the relationship exists between the two concepts.

- Fusing entity embeddings from knowledge graphs with BERT: For example, [88] first retrieved pre-trained entity embeddings from a knowledge graph, then used them to update BERT word embeddings via word-to-entity attention. [89] incorporated entity embeddings learned from a UMLS knowledge graph into BERT using adversarial learning.

- Augmenting BERT input with knowledge graph information: For example, [90] presented K-BERT in which triples from knowledge graphs were added into the input sentences before being sent to BERT. In [91], relevant knowledge

statements were assigned to each training instance and BERT was fine-tuned on the modified training data to facilitate question answering tasks.

2.2 Relevant IE Applications

In this section, I discuss IE applications that are relevant to this research.

2.2.1 Open IE

Domain-independent Open IE system was first introduced in *TextRunner* [39]. This system generated candidate tuples by first identifying pairs of noun phrase arguments and then classified if the tuple holds valid relation. A large body of work on the task of Open IE has been carried out since [39]. *ReVerb* [92] extracted verbal propositions from part-of-speech tags using a logistic regression classifier. OLLIE [93] was built on *ReVerb* and extracted relations from syntactic and lexical dependency patterns. *ClausIE* [94] first classified clauses into clause types and extracted tuples based on the clause type using predefined rules. Followed by these systems, Stanford OpenIE [95] and OpenIE5 were developed by combining several different approaches such as CALMIE [96], BONIE [97], RelNoun [98] and SRLIE [99]. Further, several systems focusing on a specialized constructs were developed, including noun-mediated relations [100], n-ary relations [101], nested propositions [102] and numerical Open IE [103].

Recently, there have been efforts to apply deep learning methods to Open IE. *RnnOIE* [67] was the first attempt to apply a supervised learning approach for

Open IE using the labeled data set from [104]. One study [105] formulated Open IE as a sequence-to-sequence generation problem and presented an encoder-decoder framework with an attention-based copying mechanism to extract binary relation tuples. Instead of relying on manually labeled data, [105] trained the model using the results of OpenIE4 [106] as labeled training data and evaluated the model using the human-labeled data from [104] as *RnnOIE* [67]. [107] presented a supervised neural Open IE model for Chinese information extraction and applied attention-based sequence-to-sequence learning similarly to [105]. However, [107] used the gated dependency attention mechanism based on the shortest path between a pair of words in the sentence's dependency tree.

2.2.2 IE for cybersecurity

Cybersecurity researchers have recently recognized the benefits of leveraging information extracted from security documents. Most prior work utilizing NLP for cybersecurity has focused on privacy policy analysis and Android security [108–112]. These systems aimed to map written policies and the actual permission requests from Android applications and assessed the risk level of these applications. IE in cybersecurity is needed to assist security analysts with crucial information about malware, vulnerability and security patches. Researchers have explored various methods to extract cybersecurity entities and their relations in the cybersecurity domain. [113] described a semi-supervised technique for entity and relation extraction that incorporates active learning. [114] extracted cybersecurity-relevant enti-

ties, terms and concepts from the text and then mapped and linked to related resources on the Web. [115] used maximum entropy models with history-based features for cyber entity extraction. [116] developed a semi-supervised bootstrapping algorithm for identifying cyber entities from large unannotated text corpora. [117] used LSTM for named entity recognition and relation extraction on text collected national vulnerability database. [118] represented the first major effort to apply NLP techniques for general text-based malware behavior analysis. This work processed reports by cybersecurity companies (e.g., FireEye, IBM X-Force, Symantec and Trend Micro) on malware or campaigns associated with Advanced Persistent Threat (APT) groups [119] and assigned attributes to identified malware actions. This research used word unigrams as predicting features. SVM and Naive Bayes were used to build classifiers for attribute label prediction. SemEval organized a shared task (called SecureNLP) on semantic analysis for cybersecurity texts to extend this effort. It adopted the same dataset and task definitions as [118]. There are four subtasks in SemEval SecureNLP: (1) identifying sentences containing malware actions from APT reports; (2) identifying "Malware Action", "Subject of Action", "Object of Action" and "Modifier of Action" in the identified sentences; (3) identifying four relations, "Subject-Action", "Action-Object", "Modifier-Action" and "Action-Modifier" in identified sentences; (4) assigning attribute labels to each identified action based on the MAEC specification.

2.2.3 Clinical Relation Extraction

Early work on clinical relation extraction employed supervised machine learning with a wide range of hand-crafted features such as lexical features, syntactic features, and semantic features extracted from external knowledge resources such as UMLS, cTAKES, and Medline. Feature engineering and SVM were commonly used in [120–128]. In addition to supervised approach, hybrid approach [129] and bootstrapping [130, 131] also were applied. Some of the works used semantic features obtained from external knowledge sources. [130, 132] derived concept mapping and concept types based on UMLS. [128, 130] employed Medline to calculate Pointwise Mutual Information (PMI) between two concepts. Later, pre-trained word embeddings (e.g. Word2Vec) became the most popular input feature for relation extraction. A neural network-based classifier (e.g., CNN, LSTM, GCN) was often used to predict clinical relations based on word embeddings. [133] used convolution neural network (CNN), [134] proposed a Segment Convolutional Neural Network (Seg-CNN) and [135, 136] used LSTM to solve clinical relation extraction task. Word embedding features were frequently combined with additional features such as word types, POS tags, IOB encoding of semantic concepts, relative distance, and dependency relations to further improve performance [133–136]. Recently, BERT-based text embedding has gained dominance due to its superior performance [137]. However, [137] did not explore how contextual embedding features can be combined with traditional NLP features to improve performance. [138] is the first to combine BERT embeddings with standard IOB tags. However, [138] did not explore whether

BERT embeddings can be effectively combined with other syntactic and semantic features commonly used in relation extraction. Despite a substantial body of research on relation extraction, it is still an open question regarding the best method to integrate bio-medical knowledge graphs (e.g., UMLS) into BERT for clinical relation extraction. To the best of my knowledge, no existing study systematically investigates the effectiveness of combining a comprehensive set of domain-independent and domain-specific features with BERT contextual embedding in clinical relation extraction.

Chapter 3: Combining Static Embedding with Domain-independent Knowledge for Open Information Extraction

3.1 Research Overview

In recent years static word embedding has become popular over previous text representation techniques like bag of words, one-hot vector and tf-idf. Static Word embedding follows the basic principle of distributed hypothesis, which states that words occurring in similar contexts tend to have similar semantics. A study has also shown that in addition to semantic information, static word embeddings contain some useful syntactic information [34]. However, captured syntactic information is often insufficient for IE tasks. As IE systems often rely on sentence syntax and grammatical patterns for extracting desired information, syntactic features are crucial. For example, syntactic features like part-of-speech tag and dependency parse tree are beneficial for different IE tasks like named entity recognition [35, 139] and relation extraction [28, 140]. This research builds an IE system that uses semantically meaningful text representation from static embedding and domain-independent knowledge. In particular, this work combines static word embedding from the Word2Vec model with part-of-speech tag, dependency parse tree information, syntactic role

label and output of existing IE systems for Open IE application. Similar to [67], in this research, Open IE is defined as a sequence tagging problem and classifies each word if it is a part of a relation, arguments or none. In this paradigm, the proposed model can enjoy the advantages of semantics captured in word embedding, syntactic knowledge about sentence structure and grammar, and the wisdom of existing Open IE models. Evaluations with several benchmark datasets and Open IE systems show that the proposed method outperforms the baseline systems by a large margin.

3.2 Static Embedding

Word2Vec [141] is the most popular static word embedding model that employs a feed-forward shallow neural network trained with unlabeled text corpora. Two Word2Vec models have been proposed: CBOW and skipgram. Figure 3.1 and Figure 3.2 shows architecture of these two models. In the CBOW model, context words are used to predict a target word and in the Skipgram model, context words are predicted based on the target word. Both models are focused on learning about words given their local usage context, where a window of neighboring words defines the context. The key benefit of the approach is that high-quality word embeddings can be learned efficiently with low space and time complexity.

3.3 Open IE

Open IE extracts textual tuples consisting of a relation phrase and argument phrases from a sentence [39]. Open IE was introduced as an alternative to the

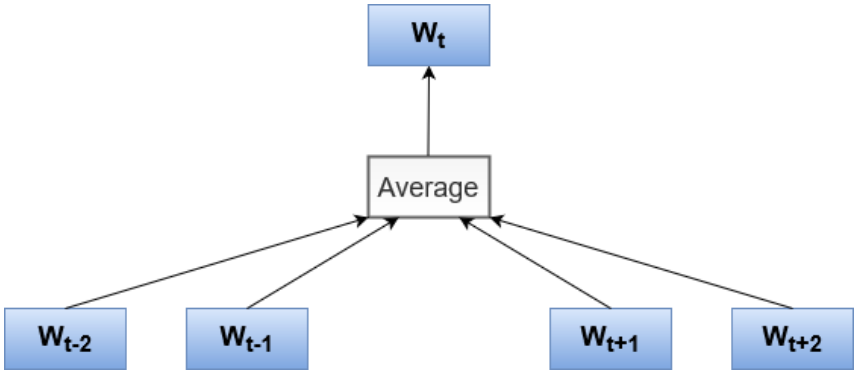


Figure 3.1: Word2Vec CBOW Architecture

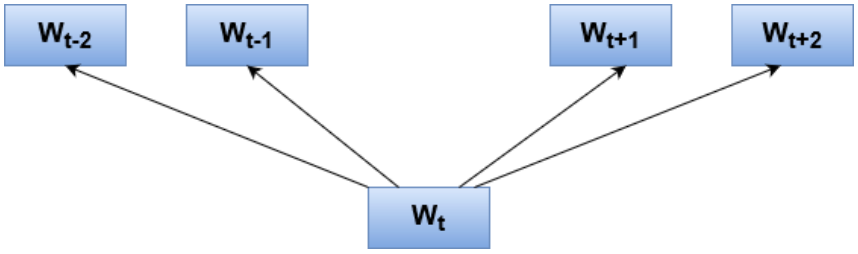


Figure 3.2: Word2Vec Skipgram Architecture

traditional supervised IE method to address two major limitations of supervised approaches. First, supervised IE relies heavily on labeled training data. Since manual relation annotation is very expensive, this method does not scale to a large number of relations and is very difficult to adapt to new domains. Second, supervised IE systems require the target relations to be predetermined and learn to extract only the predefined relations. Therefore, they miss new and potentially meaningful domain relations that are prominent in a given dataset. In contrast, Open IE operates in a completely domain-independent manner and is suitable when the target relations are not known in advance. Recently, Open IE has gained much attention, and various Open IE tools have been developed [67, 92–94, 100, 101, 142–144]. Typically, these systems read in one sentence and extract tuples with a relation phrase and

one or more arguments. However, while most existing Open IE systems extract verbal relations, each of the systems focuses on different relational structures and extraction rules, resulting in heterogeneous results. Table 3.1 shows the different extraction results from the same sentence by three different Open IE systems. Similar to other lexical and syntactic features, the outputs of these Open IE systems are domain-independent and can be obtained readily. These outputs can be used as features for a new Open IE model and help the model learn from the wisdom of multiple existing Open IE systems. This is especially attractive as no retraining or customization is needed to apply multiple existing Open IE systems, and they can be used as any other NLP tools.

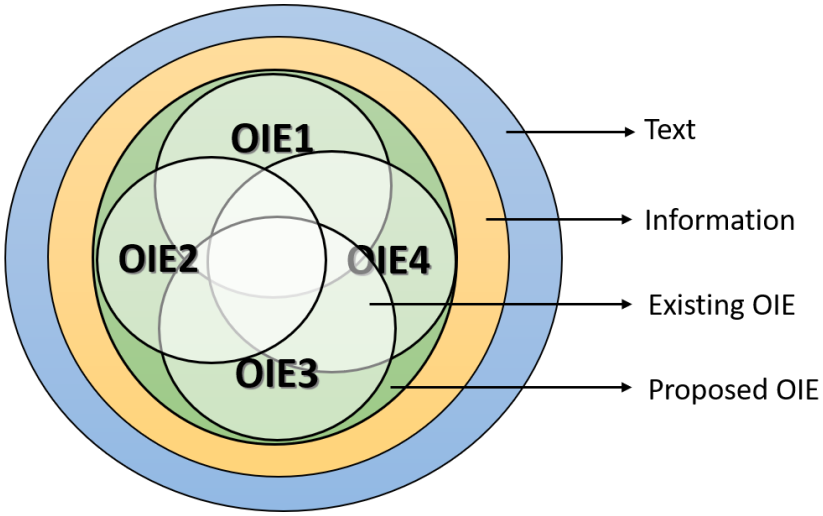


Figure 3.3: An Open IE System with Combined Knowledge of Multiple Existing Open IE Systems

Input	Evil Corp has released a new improved variant of the Dridex trojan that was spread through Andromeda botnet.
OpenIE5	1. (the Dridex trojan; was spread;) 2. (Evil Corp ; has released; new variant of the Dridex trojan)
Stanford	1. (Evil Corp; has released; variant) 2. (Evil Corp; has released; variant of Dridex trojan) 3. (Evil Corp; has released; new variant) 4. (Evil Corp; has released; new variant of Dridex trojan)
UKG	1. (Evil Corp; has released; a new improved variant of the Dridex trojan) 2. (new improved variant of the Dridex trojan; was spread through; Andromeda botnet)

Table 3.1: Extracted tuples by different Open IE systems for an input sentence

3.4 System Overview

In this work, I consider supervised open IE system (*SupervisedOIE*) that extracts of binary relations from sentences. Let us consider an input sentence S. The goal of this system is to extract a set of relation tuples T from S, where $T = \{T_1, T_2, \ldots, T_n\}$ and the i-th tuple T_i consists of $< e_{i1}, r_i, e_{i2} >$, where r_i is the relation phrase of T_i and e_{i1} and e_{i2} are the first and second arguments of r_i. This task is framed as a sequence tagging, and the model annotates each word in the sentence to $E1$, $E2$, R or O (EOR tags). $E1$ and $E2$ denote the first and the

second arguments, R is the relation, and O represents all other words. Figure 3.4 shows a system overview of the model.

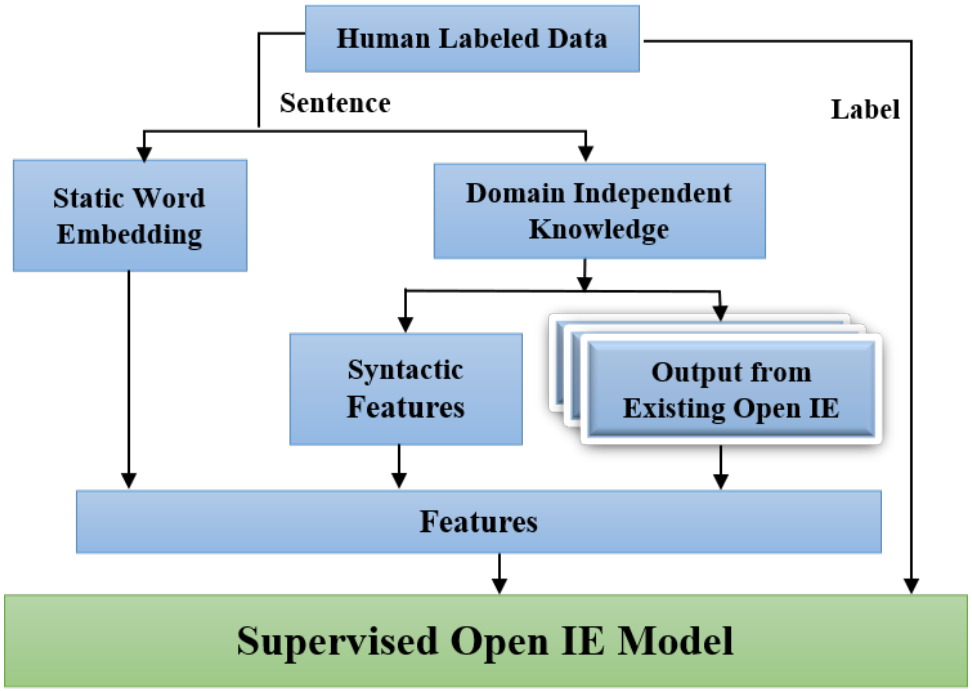


Figure 3.4: System Overview of *SupervisedOIE*

3.5 Input Features

3.5.1 Static Word Embedding:

Given an input sentence $\mathcal{S} = \langle w_1, w_2, w_3, ..., ..., w_n \rangle$, the static word embedding of each word $w_i \in \mathcal{S}$ is defined as $emb(w_i)$. Pre-trained Word2Vec trained on Google news is used for word embedding.

3.5.2 Domain-independent Knowledge

Domain-independent knowledge comes from syntactic features and output from existing Open IE systems.

A. **Syntactic features:** Following syntactic features are used in the model.

- Part-of-speech tag : POS embedding for each word w_i as $emb(pos(w_i))$ is learned during model training.
- Syntactic role label : $emb(role(w_i)$ is the syntactic role embedding for word w_i. This is also learned during model training.
- Dependency parse tree : In particular, I consider dependency parse tree based on the one-hop neighbors (*i.e.*, parent and children) of a word in the dependency tree. I use parent(w_i), the parent of word w_i, and left-child(w_i) and right-child(w_i), the closest left and right children of w_i.

B. **Output from existing Open IE systems:** Output from three different Open IE systems are used. These systems are Stanford Open IE [95], OpenIE 5[1] and UKG (a private Open IE tool). Stanford Open IE is a dependency parser-based system that uses handcrafted patterns to extract a predicate-argument triple from a sentence. On the other hand, OpenIE 5 can extract verbal relation, nominal relation, relation with numeric argument and relation from consecutive sentences. UKG extracts binary verbal relations based on noun phrase detection, named entity recognition and dependency parsing. All

[1] http://openie.allenai.org

Input Sentence	Apple released new iPhone in September.
Output	OIE1 : (Apple<E1>, released<R>, iPhone<E2>)
	OIE2 : (Apple<E1>, released<R>, September <E2>)
	OIE3 : (Apple<E1>, new <R>, iPhone<E2>)

Word	E1			R			E2		
	OIE1	OIE2	OIE3	OIE1	OIE2	OIE3	OIE1	OIE2	OIE3
Apple	1	1	1	0	0	0	0	0	0
released	0	0	0	1	1	0	0	0	0
new	0	0	0	0	0	1	0	0	0
iphone	0	0	0	0	0	0	1	0	1
on	0	0	0	0	0	0	0	0	0
September	0	0	0	0	0	0	0	1	0

17

Figure 3.5: Features Extracted from Existing Open IE Systems

three of these Open IE systems have different extraction rules and patterns focusing on extracting different relation tuples. Stanford Open IE, OpenIE 5 and UKG can complement each other when combined. To use the output of existing Open IE systems as input features, I first collect the Open IE system outputs by 1) running the existing Open IE systems as a black-box on the labeled corpus, 2) mapping the extracted tuples back to the original input sentence, and 3) assigning the EOR tags to each word based on the outputs of each Open IE system. This gives k EOR tags for each word from k Open IE tools (*e.g.*, 'E, E, O' by three Open IE systems). Figure 3.5 shows an example of how features are created from existing Open IE systems output.

For input to the system, I extract the all features for each word in the corpus. Formally, given an input sentence S, I extract a feature vector $\mathcal{F}(w_i)$ for each word

$w_i \in S$ defined as follows:

$$\mathcal{F}(w_i) = emb(w_i) \oplus \mathcal{F}_B(w_i) \oplus \mathcal{F}_B(\text{parent}(w_i))$$

$$\oplus \mathcal{F}_B(\text{left-child}(w_i)) \oplus \mathcal{F}_B(\text{right-child}(w_i))$$

where $\oplus$ denotes concatenation, $\mathcal{F}_B(w_i) = emb(pos(w_i)) \oplus emb(role(w_i)) \oplus EOR_{1,\ldots,k}(w_i)$; $pos(w_i)$ is the part-of-speech of w_i; $role(w_i)$ is the syntactic role of w_i; $emb(\cdot)$ is the respective embedding for the categorical input that can be trained as part of the model, or pre-trained; and $EOR_{1,\ldots,k}(w_i)$ represent the k EOR tags for w_i assigned by the k Open IE tools.

3.6 Model Architecture

The proposed system uses bidirectional long short term memory (Bi-LSTM) [145] to aggregate features and classify the labels of a sequence of words. The advantage of using Bi-LSTM is that it can leverage the information from neighboring words from both sides. The outputs are used in softmax for each word, producing independent probability distributions over possible EOR tags.

3.7 Experiments

3.7.1 Baseline Systems

I validate *SupervisedOIE* with several benchmark data sets and compare it with the state-of-the-art Open IE systems, including (1) a supervised Open IE system by [67]; (2) three unsupervised Open IE systems, OpenIE5 [2], Standford

[2] http://openie.allenai.org

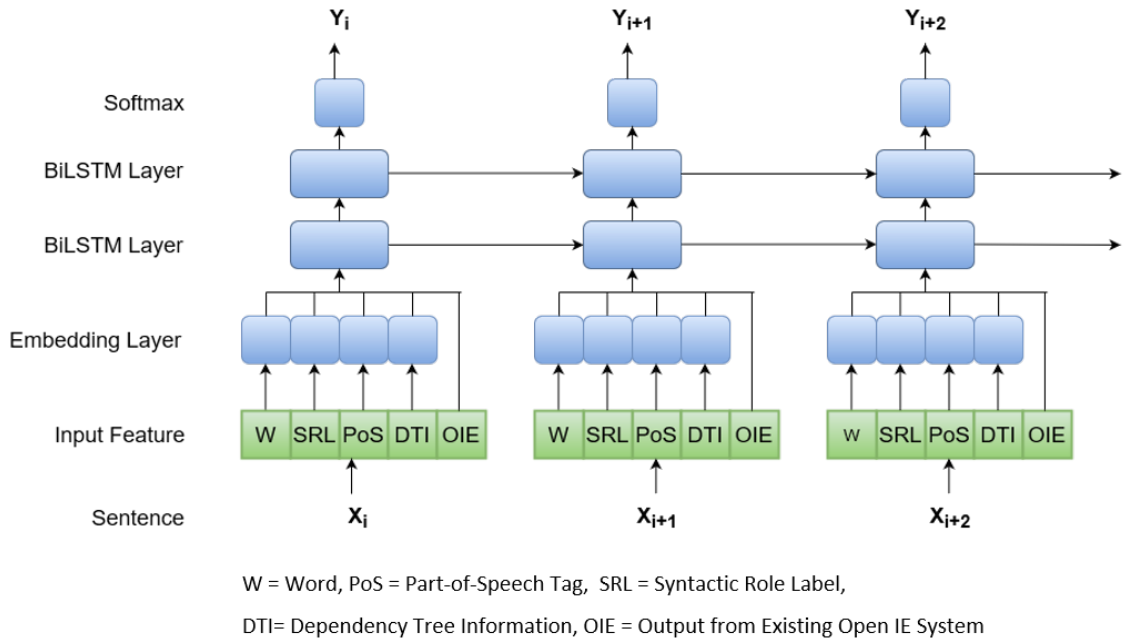


Figure 3.6: Model Architecture of *SupervisedOIE*

OpenIE [3] [95] and UKG which is a proprietary Open IE tool.

RnnOIE is the first supervised model built for Open IE [67]. The model is based on a Bi-LSTM transducer and is trained using the annotated corpus built by the same research team [104]. It takes a sentence and the word index of the predicate's syntactic head as input, and generates a feature vector for each word in the sentence by concatenating the word embeddings and POS tag embeddings of the word and the predicate head. Given these input features, the model learns whether the current word is part of an argument of the particular predicate. At inference time, they first identify verbs and verb nominalization as candidate predicates and generate an input instance with each candidate predicate head.

Stanford Open IE is heavily based on dependency parsers. The system

[3] https://stanfordnlp.github.io/CoreNLP/openie.html

first splits a sentence into a set of logically entailed shorter clauses by recursively traversing its dependency tree and predicts whether an edge should yield an independent clause or not. To increase the usefulness of the extracted propositions, each self-contained clause is then maximally shortened by running natural logic inference over it. In the end, a set of 14 handcrafted patterns are used to extract a predicate-argument triple from each utterance.

OpenIE 5 is a combination of four Open IE systems CALMIE [96], BONIE [97], RelNoun [98] and SRLIE [99]. SRLIE converts the output of a SRL system into an Open IE extraction by treating the verb as the relational phrase, and taking its role-labeled arguments as the arguments of the relation. On the other hand, RelNoun is a nominal Open IE system that extracts relations from compound noun phrases. BONIE focuses on extracting tuples where one of the arguments is a number or a quantity-unit phrase. CALMIE extracts information from conjunctive sentences by using language model-based scoring and several linguistic constraints to search over hierarchical conjunct boundaries.

UKG is a proprietary tool that was developed to construct a knowledge graph for the cybersecurity domain, which contains information about cyber-incidents involving malware, campaign, and IoCs (Indicators of Compromise). It extracts binary verbal relations based on noun phrase detection, named entity recognition, dependency parsing. UKG currently extracts binary verbal relations from three dependency structures, 'NP-VP-NP', 'NP-VP-PP' and 'VP-NP-PP'. Named entity extraction is performed to detect cybersecurity-specific entities (e.g., malware names) and constrain the extractions to only cybersecurity-related relations (those with

at least one argument being a cybersecurity entity). Furthermore, UKG employs a coreference resolution, and coordination and apposition analysis to increase the recall of the extraction. To make UKG similar to other OpenIE systems for the evaluation, I did not run the cybersecurity named entity extraction but used all noun phrases as candidate arguments.

Majority Votes is another baseline system that I compared with *SupervisedOIE*. In this system majority votes were taken from three different IE systems that were used to generate input feature for *SupervisedOIE*.

3.7.2 Experiment Data

I use four different benchmark datasets to train and test the models. The datasets are AW-OIE [67], WEB and NYT [146] and PENN [147]. Table 3.2 presents more details on these datasets.

Data Set	# of Sentences	# of Tuples
AW-OIE	3,300	17,165
AW-OIE-C	3,300	13,056
WEB	500	461
NYT	222	222
PENN	100	51

Table 3.2: Dataset Statistics

AW-OIE corpus was created by extending the OIE2016 corpus released by [104]. OIE2016 [104] was created by an automatic translation from question-answering

driven semantic role labeling annotations [148]. [67] extended these techniques and apply them to the QAMR corpus [149] to create AW-OIE. This dataset is the largest dataset available for supervised open information extraction.

However, when I observe the information extracted from this dataset, I notice that the dataset is not accurate enough to be considered as a benchmark dataset. I often find missing relations and noise introduced during the automatic generation process. To solve this problem, I manually inspect the dataset and find several patterns causing this noise in the dataset. I use these patterns to filter out noisy and missing relations from the dataset and call the cleaned data set 'AW-OIE-C'.

The WEB dataset represents the challenges of dealing with web text. This contains many incomplete and grammatically unsound sentences. NYT contains formal, well written news stories from the New York Times Corpus. The PENN dataset was created from PENN Tree Bank. I use AW-OIE-C for training and testing purpose and other three datasets only for testing. I use 8,000 instances from AW-OIE-C to train *SenseOIE* and 1,456 instances to test all the models. I set aside 3,600 instances for a new experiment described in Section 3.7.8.

3.7.3 Implementation Details

SupervisedOIE is implemented using the Keras framework [150] with TensorFlow backend [4]. Two layers of stacked bidirectional LSTM are used, each with 100 neurons with tanh activation. RMSprop optimizer is used as it is often recommended for recurrent neural networks. The model is trained using early stopping

[4] https://www.tensorflow.org/

to prevent overfitting. A batch size of 32 samples with 10% word-level dropout is used. The word embeddings are initialized using the Word2Vec [18] Google News 300-dimensions pre-trained embeddings. The part of speech and syntactic role embeddings are 25 dimensional and randomly initialized and updated during training.

3.7.4 Performance Evaluation

In this section, I report the utility of the proposed model by comparing its performance with the baseline systems on the four datasets (AW-OIE-C, WEB, NYT and PENN).

3.7.5 Evaluation Metric and Matching Function

I compare the systems using precision, recall and F1-score. To compute the measures, I need to match the automated extractions by the systems and the ground truth extractions. In this work, I compute the measures based on tuple-level matching and word-level matching. Word-level matching has been used for the evaluation metric for many NER systems. For each word, I match the tag generated by the system with the word's label.

Tuple-level matching is used in other Open IE systems [67, 105]. It is done by mapping extracted tuples with their corresponding benchmark tuples. One strategy for tuple matching would be to enforce an exact match by matching the boundaries of the extracted and benchmark tuples in text. However, as noted in earlier works [67, 151], this method penalizes different but equally valid arguments, which are resulted

from different annotation styles employed by different Open IE systems. Therefore, dealing with multiple OIE systems requires a less restrictive matching strategy. [151] introduced *relaxed containment strategy*. With this strategy, extractions are counted correct as long as they contain all gold standard arguments. [67] used a partial matching strategy allowing some variability (e.g., omissions of prepositions or auxiliaries) in the predicted tuples.

Following these works, I also use a partial matching strategy that accommodates all sorts of variabilities. I consider each argument or predicate correct if it partially matches with the benchmark data over a certain threshold. This threshold can control the leniency or strictness of the matching function. This metric allows a more balanced and fair comparison between systems that can extract potentially correct arguments beyond benchmark extraction.

3.7.6 Comparison with the Baseline Systems

Table 3.3 shows the tuple-level F1-score of *SupervisedOIE* and the benchmark systems. *SupervisedOIE* outperforms all baseline systems with a large difference. On the AE-OIE-C dataset, *SupervisedOIE* achieves the highest F1-score of 0.79. In comparison with the unsupervised Open IE methods, the performance gain of *SupervisedOIE* ranges from 66% to 315%. *SupervisedOIE* outperforms OpenIE5 by 36% to 56%. When compared to UKG, *SupervisedOIE*'s performance gain ranges from 92% to 186%. In terms of *SupervisedOIE*'s performance over the different datasets, it's worth noting the differences in annotations in the different datasets.

	AW-OIE-C	Web	NYT	PENN	AW-OIE
SupervisedOIE	**0.79**	0.66	**0.41**	**0.52**	**0.72**
RnnOIE	-	**0.67**	0.35	0.44	0.62
OpenIE5	0.58	0.46	0.29	0.34	-
Stanford OpenIE	0.19	0.24	0.21	0.31	-
UKG	0.41	0.23	0.15	0.21	-
Majority Votes	0.40	0.42	0.24	0.27	-

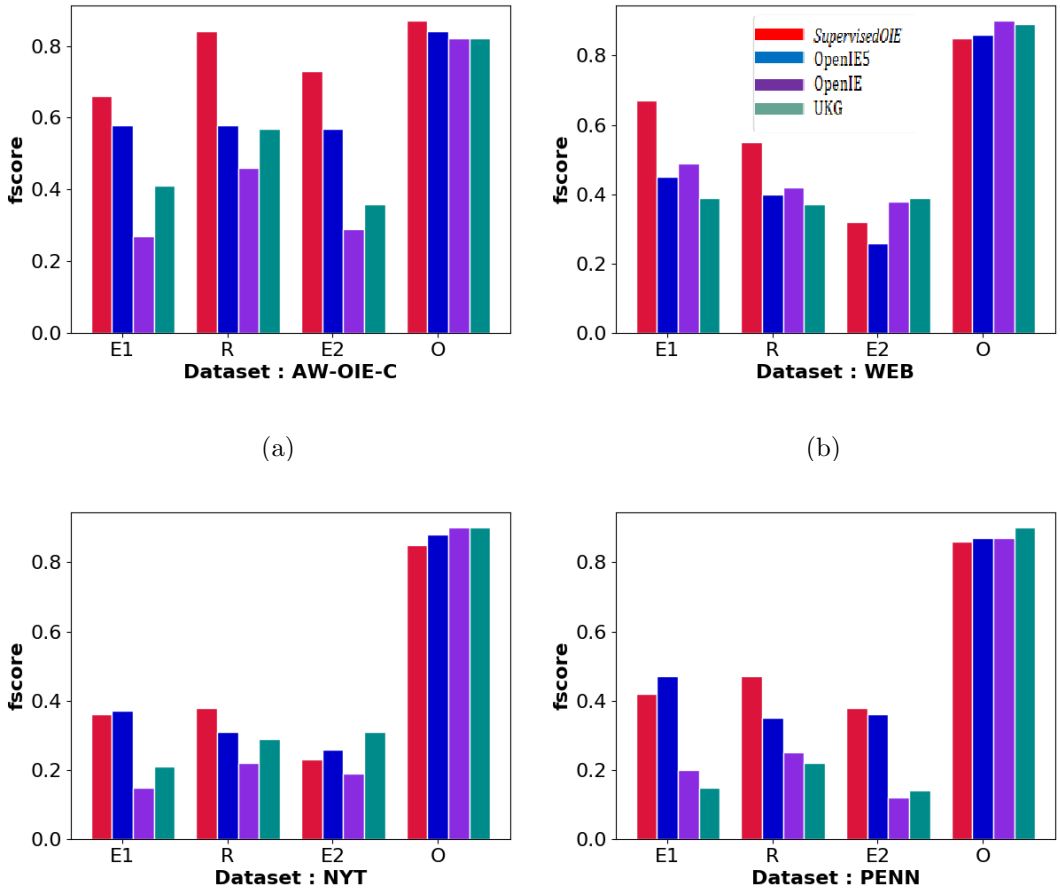
Table 3.3: Performance (F1-score) comparison of *SupervisedOIE* and the baseline systems

As the test data from AW-OIE-C follows the same annotation style as the training data, the performance of *SupervisedOIE* is much higher on this dataset compared to other datasets.

Figure 3.7 shows the comparison results based on the word level F1-scores. The results also demonstrate that *SupervisedOIE* works better than the other systems. Especially, *SupervisedOIE* shows much higher accuracy in detecting words belonging to the arguments and the relation, but a slightly lower accuracy for other words.

3.7.7 Feature Ablation Study

To investigate the contribution of each feature type on *SupervisedOIE*, I conducted a feature ablation study. Table 3.4 shows the F1-scores of several variations



(a) (b)

(c) (d)

Figure 3.7: Word level F1-score comparison

built with a different subset of features. I note that the performance of *SupervisedOIE* is, on average, 109% higher than the model using the only word embedding features. Overall the performance gain over word embedding gain ranges from 37% to 156%. This performance boost proves that using domain independent knowledge with static embedding for OpenIE systems is very effective. Surprisingly model without features from the dependency parse tree outperforms *SupervisedOIE* in 2 out of 4 datasets. This might be an indicator that simple concatenation is not the best way to include features from the dependency tree.

	AW-OIE-C	Web	NYT	PENN
Word Embedding	0.38	0.48	0.16	0.22
(Word + PoS + SRL) Embedding	0.42	0.58	0.16	0.27
Features from Existing Open IE	0.70	0.54	0.38	0.47
All Features - DTI	0.78	**0.69**	**0.45**	0.51
All Features	**0.79**	0.66	0.41	**0.52**
Improvement over Word Embedding	107%	37%	156%	136%

Table 3.4: Performance (F1-score) comparison of different feature sets

3.7.8 *SupervisedOIE* as Annotator

Since *SupervisedOIE* outperforms the baseline systems by a large margin, it is worth investigating if *SupervisedOIE* can be used to bootstrap a supervised Open IE model for new domains by automatically producing annotated data. Previously, [105] used OpenIE4 [152], an earlier version of OpenIE5, to automatically create a training dataset. The limitation of their approach is that using only one OpenIE system's extraction as ground truth will result in biased and low coverage of extracted relations. As each unsupervised OpenIE system has its own rules to extract different relations, applying only one system might miss other potential relations that other Open IE systems can extract. However, since *SupervisedOIE* learns from multiple existing Open IE systems, it can extract many different relation types.

For this purpose, I run *SupervisedOIE* on the 3,600 instances from AW-OIE-

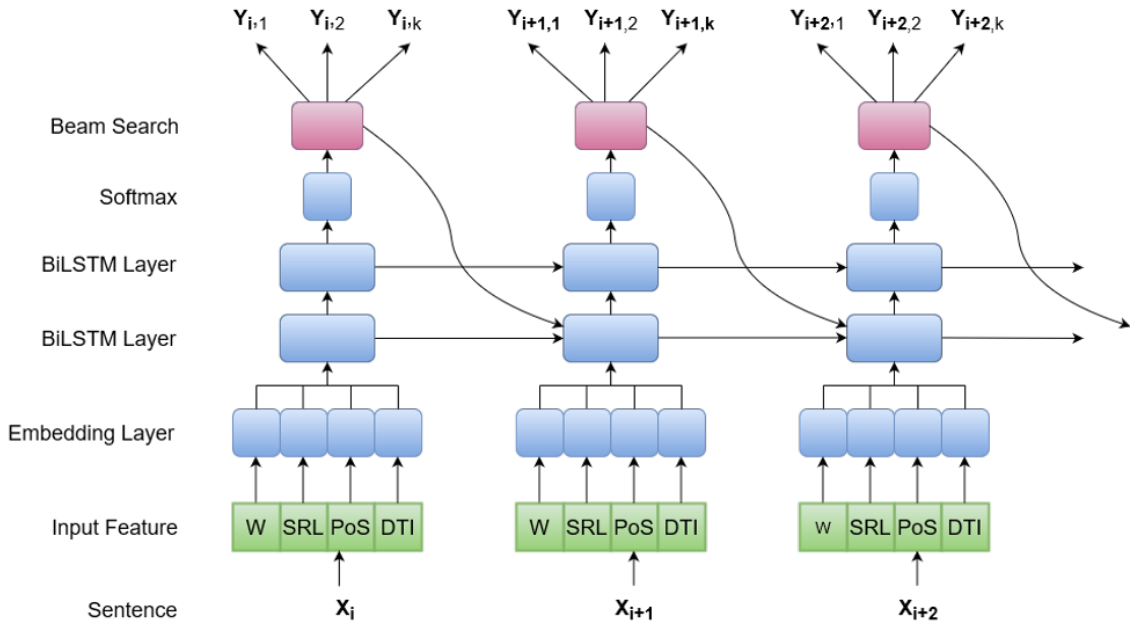
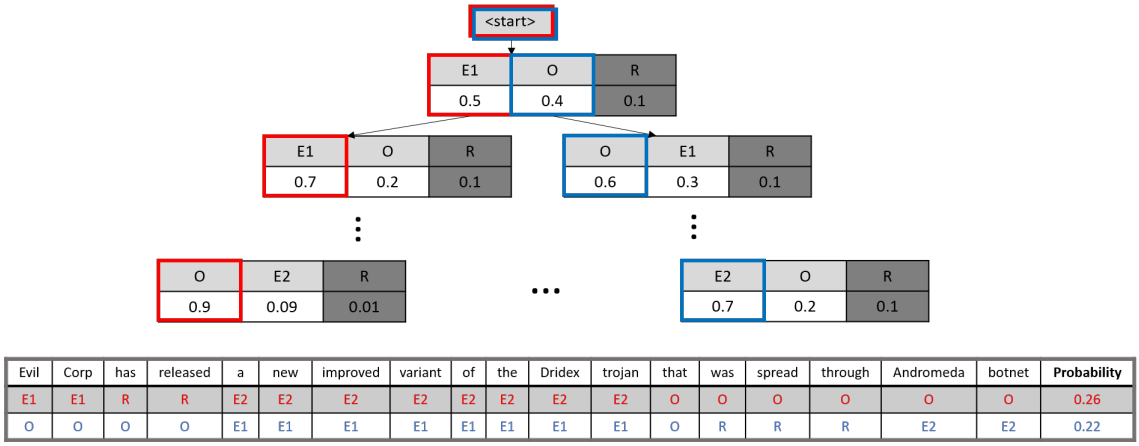


Figure 3.8: Model Architecture of *SupervisedOIE_V2*

C and use its extracted results as the ground truth to train a supervised model. This new model is named *SupervisedOIE_V2* to differentiate it from *SupervisedOIE*. The model is quite similar to *SupervisedOIE* using LSTM to aggregate features and classify the labels of a sequence of words. The input features for each word are word embedding, POS embedding, syntactic role embedding, dependency tree information and label of the previous word. During training, label of the previous word comes from ground truth and during testing, this value is predicted by the model. This feature is helpful to generate multiple sequences of extractions from a single sentence. However, note that this model does not use the results of unsupervised systems as features. Figure 3.8 shows the architecture and features of *SupervisedOIE_V2*.

During the inference time, to extract multiple relations from a single sentence,



Evil	Corp	has	released	a	new	improved	variant	of	the	Dridex	trojan	that	was	spread	through	Andromeda	botnet	**Probability**
E1	E1	R	R	E2	E2	E2	E2	E2	E2	E2	E2	O	O	O	O	O	O	0.26
O	O	O	O	E1	E1	E1	E1	E1	E1	E1	E1	O	R	R	R	E2	E2	0.22

Figure 3.9: Example of Beam Search Predicting Multiple Relation Extraction Sequences from One Sentence

beam search is used to find multiple possible labels for each word. Instead of greedily choosing the most likely next step as the sequence is constructed, the beam search expands all possible next steps and keeps the k most likely results, where k is a user-specified parameter and controls the number of beams or parallel searches through the sequence of probabilities. Figure 3.9 shows an example of beam search predicting multiple relation extraction sequences from one sentence.

To validate the effectiveness of *SupervisedOIE* as an annotator model, I compare *SupervisedOIE_V2*'s performance when trained with the human-labeled data and *SupervisedOIE*'s extractions. As with *SupervisedOIE*, both models are initialized with the pre-trained word embedding and randomly initialize the part-of-speech and syntactic role embeddings. In this experiment, the beam size is set to 3, which gives an overall best performance. Table 3.5 shows the results from these two models. Both models achieve similar F1-scores on the four test dataset. These results

support *SupervisedOIE*'s role as a digital annotator for an unlabeled dataset.

	AW-OIE-C	**Web**	**NYT**	**PENN**
Human Labels	0.55	0.51	0.23	0.27
SupervisedOIE Labels	0.54	0.50	0.23	0.23

Table 3.5: Performance (F1-score) of *SupervisedOIE_V2* trained with the human-labeled data vs. labeled data generated by *SupervisedOIE*

3.8 Summary

I develop a new supervised Open IE system that uses static word embedding and domain independent knowledge including parts of speech tag, syntactic role label, dependency tree information and results of existing Open IE systems as features. Validation using several benchmark data sets generated for the Open IE task shows that the proposed method is very effective in outperforming both other supervised and unsupervised Open IE systems.

Further, I investigate if proposed model can be applied to automatically generate annotated data to train a new supervised model for a new task. The experiment shows that a supervised model trained with the model-generated data performs similarly to the model trained with human labeled data. This result shows that the proposed approach can overcome the cold-start problem in machine learning by leveraging existing unsupervised systems.

Chapter 4: Combining Static Embedding with Domain-specific Knowledge for Malware Attribute Extraction

4.1 Research Overview

IE in a specific domain focuses on satisfying precise, narrow, pre-specified target information from domain-specific text. Domain-specific IE is challenging because domain-specific text usually has a highly specialized vocabulary and domain-specific relations between domain concepts. Domain knowledge can be very beneficial to overcome the challenges of the domain-specific IE approach. A specialized domain like cybersecurity is ideal for illustrating the benefit of utilizing domain knowledge with word embedding for IE tasks.

In the cyber-security domain, automated malware identification based on text is crucial for immediate awareness of security flaws. Detection of malware often relies on an understanding of the characteristics of malware behavior. To establish a standard for unambiguously characterizing malware, MAEC (Malware Attribute Enumeration and Characterization), a community-based project organized by MITRE, has specified a set of standard malware attributes [153]. Based on MAEC, the actions of a malware can be categorized by four attributes: *ActionName*, *Capability*, *StrategicObjectives* and *TacticalObjectives*. *ActionName* specifies the actions taken by a malware. For example, "delete file" is a malware action that

deletes existing files from affected systems. *Capability* defines the general capabilities of a malware. For example, "anti-removal" is a malware capability that prevents itself from being removed from a system. *StrategicObjectives* and *TacticalObjectives* are subcategories of *Capability* to capture more details. For example, a malware can have a *StrategicObjective* of "staging data for exfiltration" and a *TacticalObjective* of "moving data to a staging server". In total, MAEC specified 211 *ActionNames*, 20 *Capabilities*, 65 *StrategicObjectives*, and 148 *TacticalObjectives*.

The IE task of automated malware attribute identification based on cybersecurity text is very challenging for three reasons: 1. domain-specific text, 2. a large number (444) of malware attribute labels and 3. a small number of training instances. To solve this domain-specific IE task, I incorporate domain knowledge with static word embedding. Domain knowledge that details what type of information is to be obtained from domain-specific text can boost the performance of the IE system. In this research, I develop a technique to enrich static word embedding with diverse domain knowledge, including domain text, MAEC attribute specification and human annotation. Domain knowledge enriched embeddings are used as features to train a supervised IE model that achieves high performance. Evaluation has demonstrated the effectiveness of the proposed method over the state-of-the-art malware attribute prediction systems.

4.2 Background

SemEval organized a shared task (called SecureNLP) on semantic analysis for cybersecurity texts. It adopted the same dataset and task definitions as [118].

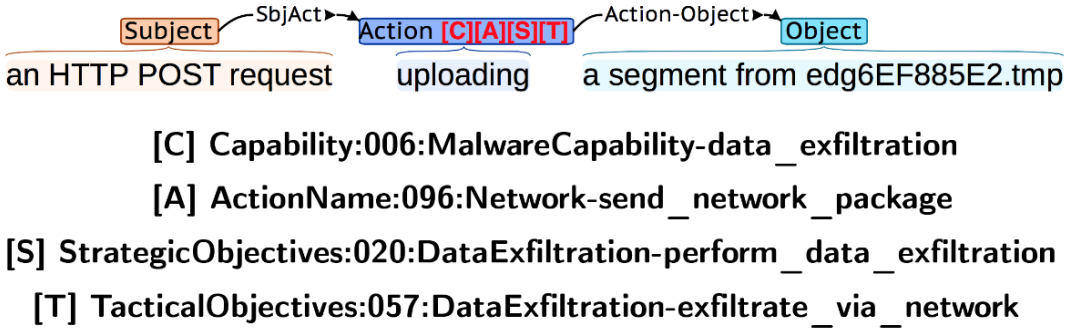


Figure 4.1: Annotated Sentence Fragment for SemEval Shared Task.

There are four subtasks in SemEval SecureNLP: (1) identifying sentences containing malware actions from APT reports; (2) identifying "Malware Action", "Subject of Action", "Object of Action" and "Modifier of Action"in the identified sentences; (3) identifying four relations,"Subject-Action", "Action-Object", "Modifier-Action" and "Action-Modifier" in identified sentences; (4) assigning attribute labels to each identified action based on the MAEC specification. Figure 4.1 shows an annotated example for these tasks.

This research describes my approach to solve subtask 4. The input to the proposed system includes all the sentences identified in subtask 1 with additional labels for the entities identified in subtask 2. Each training and testing instance used in SemEval SecureNLP only contains a single malware action.

4.3 System Overview

Figure 4.2 shows the high-level system architecture. First, all the raw APT reports are augmented with annotations to encode the knowledge from both the training data and MAEC. This allows designing a unified representation for both

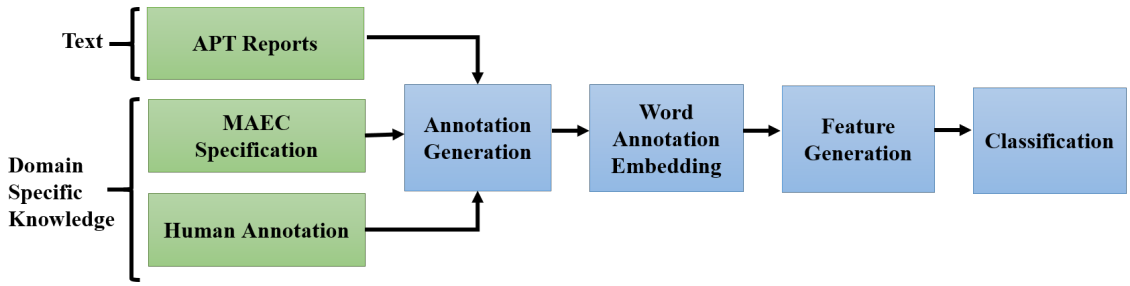


Figure 4.2: System Architecture

types of knowledge. The annotated texts are then used by WAE to learn embeddings for both words and malware attribute labels simultaneously. The learned word and attribute label embeddings are then used to construct high qualify prediction features. Finally, supervised machine learning is employed to predict malware attribute labels. There are four classifiers, one for each malware attribute. Each classifier performs n+1-way classification, where n is the number of possible labels for each attribute and 1 is for 'no value' when the value of an attribute is not conveyed in the text.

4.4 Annotation Generation

To annotate text with additional knowledge, the first step is to map each attribute label to a set of keywords based on both MAEC and the human annotations in the training data. Then attribute labels are used to annotate corresponding keywords in text. Details of these steps are described below.

Identify keywords from MAEC: Figure 4.3 shows a snippet of the MAEC specification. Each malware attribute label in MAEC includes a description and a few keywords. The malware action 004 in Figure 4.3 has a name 'emulate driver', a description 'specify the defined action of emulating an existing driver on a system'

No.	Name	Description	Relevant Words
000	send dns query	Specifies the defined Action of sending a DNS query.	[DNS][query][send]
001	send reverse dns lookup	Specifies the defined Action of sending a reverse DNS lookup.	[DNS][reverse lookup][send]
002	check for kernel debugger	Specifies the defined Action of checking for the presence of a kernel debugger.	[debug][kernel debugger]
003	check for remote debugger	Specifies the defined Action of checking for the presence of a remote debugger.	[debug][remote debugger]
004	emulate driver	Specifies the defined Action of emulating an existing driver on a system.	[driver][emulate]
005	load and call driver	Specifies the defined Action of loading a driver into a system and then calling the loaded driver.	[driver][load][call]
006	load driver	Specifies the defined Action of loading a driver into a system.	[driver][load]
007	unload driver	Specifies the defined Action of unloading a driver from a system.	[driver][unload]

Figure 4.3: A Snippet of the MAEC Specification

and two keywords: 'driver' and 'emulate'. Since the keywords carry the most essential information about a malware attribute label, each label is linked with these keywords (e.g., ActionName004: 'driver', 'emulate').

Identify keywords from training data: Since a malware attribute label in the training data is at the sentence level, to extract keywords for each attribute label, all the sentences associated with the same label are extracted and considered as one document. To select the most relevant keywords, I only keep those conveying "Malware Action", "Subject of Action", or "Object of Action" (which were identified in subtask 2).

Keywords ranking: For the same attribute label, the keywords from MAEC and those from the training data are merged to form a single document. Then tf-idf scores are used to select the most informative keywords to differentiate these labels. In the experiments, the top 25 keywords based on their tf-idf scores are used.

Text annotation generation: Finally, for all the APT documents, the text is annotated with malware attribute labels. Specifically, for any word in the APT documents, if it is a keyword associated with k different labels, the word is annotated with k attribute labels.

4.5 Word Annotation Embedding (WAE)

To combine domain knowledge with static word embedding, I develop a novel Word Annotation Embedding (WAE) method that learns word embeddings with text annotations. Similar to static word embedding Word2Vec [141], the goal here is to learn features that capture the semantic relations between words. In addition, to facilitate attribute classification, WAE aims to capture the semantic relations between words and their labels. Specifically, WAE brings words and their attribute labels closer in the embedding space and pushes the embeddings of different labels far away from each other. The inputs to this system are texts as well as annotations. In addition to producing vector representations of words, WAE also produces vector representations of the annotations in the same embedding space. Similar to the CBOW and the Skip-gram models used in Word2Vec, there are five variants of the WAE model with different objectives.

WAE Model 1: The learning objective of model 1 is to predict a word given its annotations plus the other words in its context. A sliding window on the input text stream is employed to generate the training samples. In each sliding window, the model uses the surrounding words and the annotations of the target word as the input to predict the target word. More formally, assume a word W_t has a set of M_t annotations ($A_{t,1}$, $A_{t,2}$, ...,A_{t,M_t}). Given a sequence of T training words, $\{W_1,...,W_t,...,W_T\}$, the objective of WAE Model 1 is to maximize the average log probability shown in Equation 4.1.

$$\frac{1}{T}\sum_{t=1}^{T}\Big(\sum_{-C\leq j\leq C, j\neq 0}\log P(W_t|W_{t+j}) + \sum_{0\leq k\leq M_t}\log P(W_t|A_{t,k})\Big) \tag{4.1}$$

where C is the window size, W_t is the target word, W_{t+j} is a context word, $A_{t,k}$ is the k-th annotation of W_t.

WAE Model 2: The objective of this model is to predict the context words and their annotations based on a target word. More formally, given a sequence of T training words, $\{W_1,...,W_t,...,W_T\}$, and their annotations $((A_{1,1},...A_{1,M_1}),...,(A_{t,1},...,A_{t,M_t}),...,(A_{T,1},...A_{T,M_T}))$, the objective of WAE Model 2 is to maximize the average log probability shown in Equation 4.2.

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-C\leq j\leq C, j\neq 0}\Big(\log P(W_{t+j}|W_t)+\sum_{0\leq k\leq M_{t+j}}\log P(A_{t+j,k}|W_t)\Big) \tag{4.2}$$

where $A_{t+j,k}$ is the k-th annotation of the context word W_{t+j}, and M_{t+j} is the number of annotations associated with the context word W_{t+j}.

WAE Model 3: The objective of model 3 is to use the target word to predict it's own annotation as well as the context words and to maximize the average log probability shown in Equation 4.3.

$$\frac{1}{T}\sum_{t=1}^{T}\Big(\sum_{-C\leq j\leq C, j\neq 0}\log P(W_{t+j}|W_t)+\sum_{0\leq k\leq M_t}\log P(A_{t,k}|W_t)\Big) \tag{4.3}$$

WAE Model 4: The objective of model 4 is to use the target word to predict the context words and the annotations of both the target word and the context words and to maximize the average log probability shown in Equation 4.4.

$$\frac{1}{T}\sum_{t=1}^{T}\Big(\sum_{-C\leq j\leq C, j\neq 0}\Big(\log P(W_{t+j}|W_t)+\sum_{0\leq k\leq M_{t+j}}\log P(A_{t+j,k}|W_t)\Big)+\sum_{0\leq l\leq M_t}\log P(A_{t,l}|W_t)\Big) \tag{4.4}$$

WAE Model 5: The target word is used to predict not only its context words but also its labels. To further strengthen their relations, the labels of the target word are also used to predict the target word. Specifically, given a sequence of T

words $(W_1,..,W_t,...,W_T)$ and their annotations $((A_{1,1},...,A_{1,M_1}),...,(A_{T,1},...,A_{T,M_T}))$, the objective of this model is to maximize the average log probability shown in Equation 4.5, where C is the size of the context window, W_t is the target word, W_{t+j} is a context word, M_t is number of annotations W_t has and $A_{t,k}$ is the k-th annotation of W_t.

$$\frac{1}{T}\sum_{t=1}^{T}\Big(\sum_{-C\leq j\leq C, j\neq 0}\log P(W_{t+j}|W_t)+\sum_{0\leq k\leq M_t}(\log P(A_{t,k}|W_t)+\log P(W_t|A_{t,k}))\Big) \quad (4.5)$$

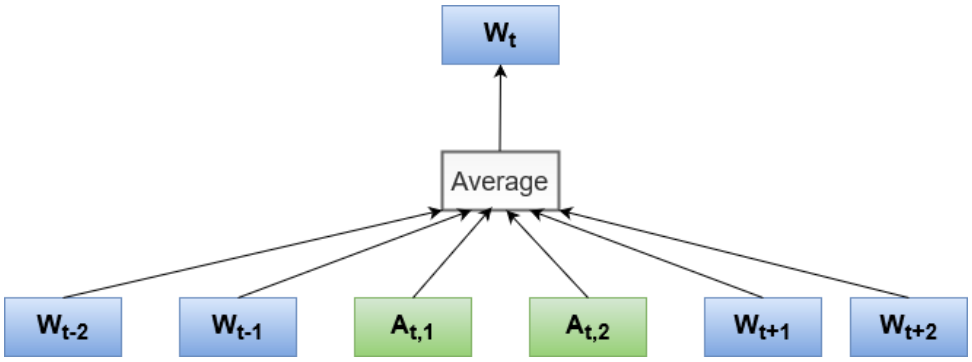


Figure 4.4: Architecture of WAE Model 1

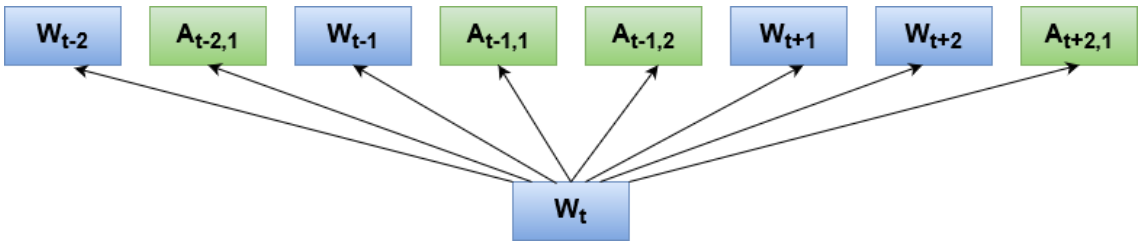


Figure 4.5: Architecture of WAE Model 2

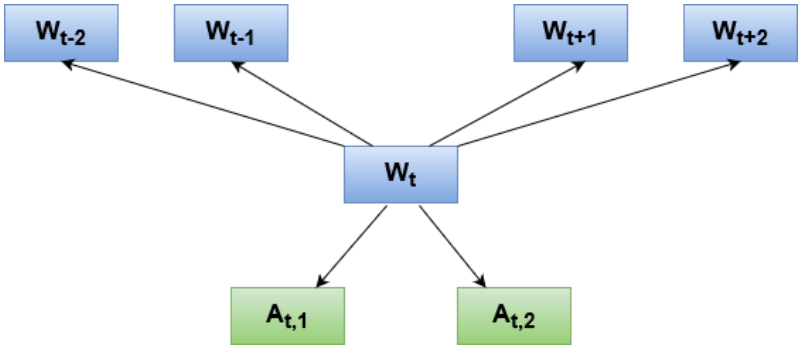


Figure 4.6: Architecture of WAE Model 3

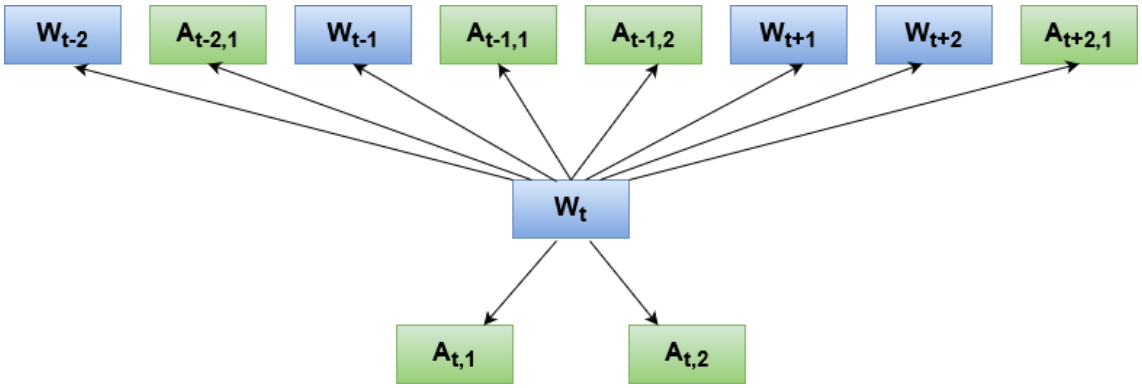


Figure 4.7: Architecture of WAE Model 4

Label-aware Negative Sampling: Negative sampling [141] was introduced as an approximation method in Word2Vec to improve the efficiency of model training. Previously, negative samples were selected either randomly or based on popularity. The method, however, is insensitive to class labels. Here, I develop a new annotation-aware negative sampling method to (1) keep different annotations apart in the embedding space and (2) to keep words associated with different labels apart, in addition to bringing words and their associated labels closer to each other via positive samples. For example, in the figure 4.9, word $W1$, $W2$ are close to their annotation $A1$ and $W3$ is close to its annotation $A2$ in the embedding space. $W1$ and $W2$ are nearby as they share the same annotation $A1$. And $W3$ is far away from

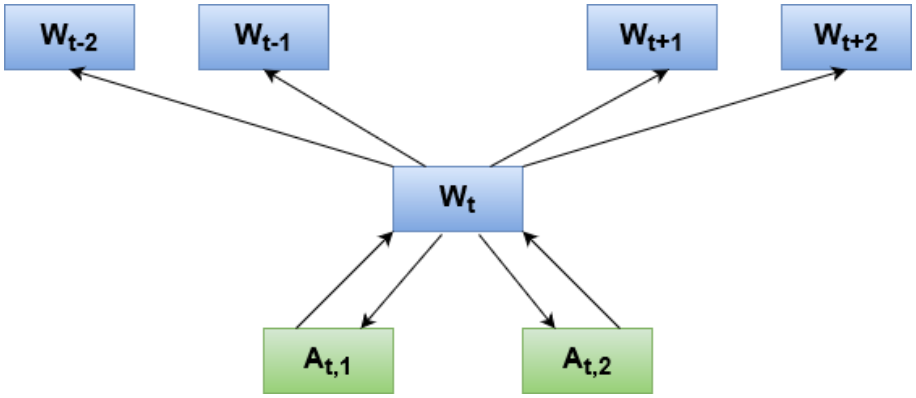


Figure 4.8: Architecture of WAE Model 5

W1 and $W2$ as annotation of $W3$ is different from them. And finally, annotation $A1$ and $A2$ are also distant from each other. To achieve this, WAE randomly selects (1) a word as a negative sample if it does not share the same annotations as the target word; (2) an attribute label as a negative sample if it is not the same as the labels of the target word.

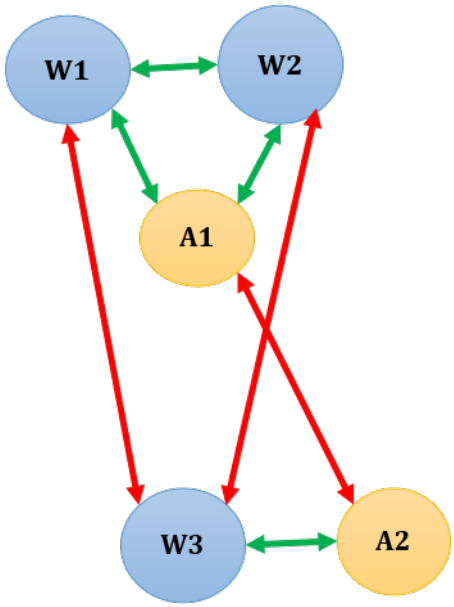


Figure 4.9: Impact of Label-aware Negative Sampling

4.6 Feature Generation and Classification

Six sets of features are constructed to train the classifiers.

A. *(S1)* WAE_W*+Sim:* Assume the average word embeddings for a given data instance generated by WAE is I_{wae}, and the malware attribute label embedding learned by WAE is $LabelE_i$ for each label i. For each $LabelE_i$, SIM_i, is the cosine similarity between WE_{wae} and $LabelE_i$. WAE_W*+Sim* is the concatenation of WE_{wae} and all the SIM_i. For example, to predict *ActionName*, the model will include 100 word embedding features learned by WAE plus 211 similarity features, one for each *ActionNames*.

B. *(S2)* $w.WAE_W$*+Sim:* This feature set is similar to WAE_W*+Sim* except when computing WE_{wae}, a word with a label is assigned twice as much weight as one without a label. The intuition is words with labels are important keywords based on either MAEC or the training data.

C. *(S3)* WAE_W*+*WAE_L*+Sim:* It is similar to WAE_W*+Sim* except the average embeddings of attribute labels associated with the instance is also included.

D. *(S4)* $w.WAE_W$*+*WAE_L*+Sim:* This is the weighted version of (S3).

E. *(S5)* WAE_W*+*WAE_L*+Sim+Cap:* Since the label of *StrategicObjective* and *TacticalObjective* depends on the label of *Capability*, the *capability* label is added in the feature set. A 1-hot vector with 20 elements is used to encode a *Capability* label. The ground truth and the predicted label of *Capability* is used during training and testing respectively.

F. *(S6)* $w.WAE_W$*+*WAE_L*+Sim+Cap:* This is the weighted version of (S5)

4.7 Experiments

4.7.1 Dataset

The dataset used in the experiments was provided as a part of the SemEval shared task. It contains 456 APT reports [119], 39 of them were annotated by humans. Among them, 2975 sentences contain malware actions, which are the data instances used in this study. The annotated data are very sparse. There are 982 sentences with *ActionName* label, 2524 sentences with *Capability* label, 2004 sentences with *StrategicObjective* label and 1592 sentences with *TacticalObjective* label. Out of the 444 attribute labels, 190 labels do not appear in the labeled data. For the remaining 254 attribute labels, 92 labels occur less than five times, and 50 labels occur only once. In the experiments, I use the raw text in all the APT reports to train WAE. There are 16423 unique tokens and a total of 2544645 tokens in the dataset. I train both Word2Vec and WAE with context window size 5 and 100 dimension vectors. The 39 annotated documents were divided into a training set (23 documents), a validation set (8 documents) and a test set (8 documents). Only the

training dataset is used to generate annotations for WAE. The validation dataset isused for parameter tuning.

4.7.2 Evaluation Results

For feature generation, I test all five versions of WAE model. In all the experiments, features learned from WAE model 5 consistently give the best performance over other four WAE models. In this section I only report results using features from WAE model 5. For classification, I try both SVM and neural network-based models such as multilayer perceptron. After experimenting with different model parameters, I find that the best SVM model with a linear kernel performed slightly better than the best neural network models. I speculate that this might be because SVM is less likely to overfit when the training data are sparse. Table 3.3 shows the average F-scores over 5 runs by the SVM models on the test data. I compare proposed models with three baseline systems: (B1) [118], (B2) Word2Vec and (B3) Word2Vec + cap. Among them, (B1) represents the best published results on the same dataset. (B2) and (B3) are all comparable models with embeddings learned by Word2Vec.

As shown in Table 4.1, all proposed models outperform all the baseline systems. For *ActionName*, best model achieve a F-score of 0.46 ($w.WAE_W$+*Sim*) and for *Capability*, best model achieve a F-score of 0.63 (e.g., WAE_W+WAE_L+*Sim*). For *StrategicObjective* and *TechnicalObjective*, the WAE_W+WAE_L+*Sim*+*Cap* model results in the highest F-score of 0.47 and 0.45 respectively. The improvement over the Word2Vec model is 15%, 11%, 15% and 25% respectively, and, the improvement

Models	**ActionName**	**Capability**	**StratObj**	**TactObj**
(B1) [118]	0.33	0.41	0.27	0.22
(B2) Word2Vec	0.40	0.57	0.41	0.36
(B3) Word2Vec+Cap	NA	NA	0.41	0.39
(S1) WAE_W+Sim	0.44	0.61	0.43	0.41
(S2) $w.WAE_W$+Sim	**0.46**	0.62	0.44	0.43
(S3) WAE_W+WAE_L+Sim	0.45	**0.63**	0.46	0.43
(S4) $w.WAE_W$+WAE_L+Sim	0.45	**0.63**	0.45	0.43
(s5) WAE_W+WAE_L+Sim+Cap	NA	NA	**0.47**	**0.45**
(S6) $w.WAE_W$+WAE_L+Sim+Cap	NA	NA	**0.47**	**0.45**
$\Delta 1$: over [118]	↑**39%**($p < 0.05$)	↑**54%**($p < 0.05$)	↑**74%**($p < 0.05$)	↑**105%**($p < 0.05$)
$\Delta 2$: over Word2Vec	↑**15%**($p < 0.05$)	↑**11%**($p < 0.05$)	↑**15%**($p < 0.05$)	↑**25%** ($p < 0.05$)
$\Delta 3$: over Word2Vec+Cap	NA	NA	↑**15%**($p < 0.05$)	↑**15%** ($p < 0.05$)

Table 4.1: Evaluation Results on Malware Attribute Extraction

over [118], a previous state of the art, is 39%, 54%, 74% and 105% respectively. The improvement over $Word2Vec + Cap$ is 15% and 15% respectively for *StrategicObjective* and *TechnicalObjective*. I also conduct t-tests to verify the significance of the improvements. The t-test results confirm that the proposed models significantly outperformed the baseline models with p<0.05. Moreover, the value of "Capability" seems to help the prediction of *StrategicObjective* and *TechnicalObjective*.

4.8 Summary

In this chapter, I combine domain specific knowledge with static word embedding to build IE system for cyber-security. I present a novel method to predict malware attribute labels from cybersecurity text. Given a large number of attribute labels and limited training data, I develop a new feature learning method to incorporate knowledge from diverse knowledge sources such as raw text, MAEC specifications and human annotations. I test the new system using the SemEval shared task data and evaluation demonstrates that the features learned by the proposed models are much more effective than an existing state of the art as well as embedding features learned by word2vec. My investigation highlights the importance of incorporating diverse domain knowledge sources in domain specific IE task.

Chapter 5: Combining Contextual Embedding with Domain-independent Knowledge for Clinical Relation Extraction

5.1 Research Overview

Electronic Medical Record (EMR) is an electronic version of a patient's medical history that healthcare providers maintain over time. Clinical text in EMR includes physicians' notes, surgical records, discharge summaries and laboratory reports. This contains valuable information about a patient's conditions such as symptoms, diagnoses and treatments. Hence identifying, extracting and mining this information is of great importance for managing and improving patient care. Manually inspecting a large amount of clinical text is labor-intensive and time-consuming. Therefore, IE system, which can automatically extract information of interest from text, is beneficial for clinical research and applications. One of the fundamental tasks of clinical NLP is extracting relations between medical concepts from texts (e.g., extracting relations between diseases and treatments). For example, in the sentence "Ibuprofen reduced inflammation but likely caused heartburn", "inflammation" was effectively treated by "ibuprofen" and "heartburn" is an adverse event caused by the same drug. Identifying these relations is essential to understand how patients respond to treatments. Therefore, automated extraction of relations between medical concepts is necessary for clinical decision making and patient care.

In this study, I develop neural network models that classify the relationship between two medical concepts in a given sentence (e.g., between a treatment such as "ibuprofen" and a medical problem such as "inflammation").

Text embedding, a technique to automatically learn dense vector representations for words and sentences, has proven to be effective for relation extraction [154, 155]. Recently contextual word embedding (e.g., BERT), where the embedding vector for each word varies with its context, has gained much popularity over static word embedding (e.g., Word2Vec), where a fixed embedding vector is learned for each word. In this research, I investigate the effectiveness of contextual word embedding for clinical relation extraction task. While contextual embedding has produced impressive results on diverse downstream NLP tasks, it is unclear whether contextual embedding can be combined effectively with typical surface and syntactic features commonly used in traditional relation extracting systems. In this research, I combine contextual embedding from BERT with domain-independent knowledge, including POS tag, dependency tree path, relative distance between entities and IOB encoding. These domain-independent syntactic and surface features are traditionally used for IE applications. These combined embeddings and knowledge are used as input features for the relation classifier model. I compare the effectiveness of BERT contextual embedding with Word2Vec static embedding, especially when combined with traditional domain-independent features. I investigate if these domain-independent features are beneficial for IE systems in a specialized domain like clinical text. I evaluate models on a benchmark clinical text dataset; the i2b2-2010 dataset [156] where the task is to classify relations between medical

concepts.

5.2 Contextual Embedding

BERT (Bidirectional Encoder Representations from Transformers) is the most popular contextual embedding model. BERT is a transformer-based language model that uses an attention mechanism to learn contextual relations between words in a text. Unlike directional models, which read the text input sequentially (left-to-right or right-to-left), the Transformer encoder reads the entire sequence of words at once. Therefore it is considered bidirectional. This characteristic allows the model to learn the context of a word based on its surroundings (left and right of the word). BERT is trained on large text corpora collected from the Books Corpus and Wikipedia. BERT uses a masked language model (MLM) and next sentence prediction objectives to learn text embedding. Before feeding word sequences into BERT, 15% of the words in each sequence are replaced with a [MASK] token. The model then attempts to predict the original value of the masked words based on the context provided by the other, non-masked words in the sequence. In the BERT training process, the model receives pairs of sentences as input and learns to predict if the second sentence in the pair is the subsequent sentence in the original document. The input is processed in the following way before entering the model. Input is tokenized using wordpiece tokenization. A [CLS] token is inserted at the beginning of the first sentence and a [SEP] token is inserted at the end of each sentence. Pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of NLP tasks.

5.3 System Overview

I develop a supervised relation classifier trained with contextual embedding and domain-independent knowledge. I also compare models using contextual embedding with baselines that use static embedding. Figure 5.1 shows system overview of the model.

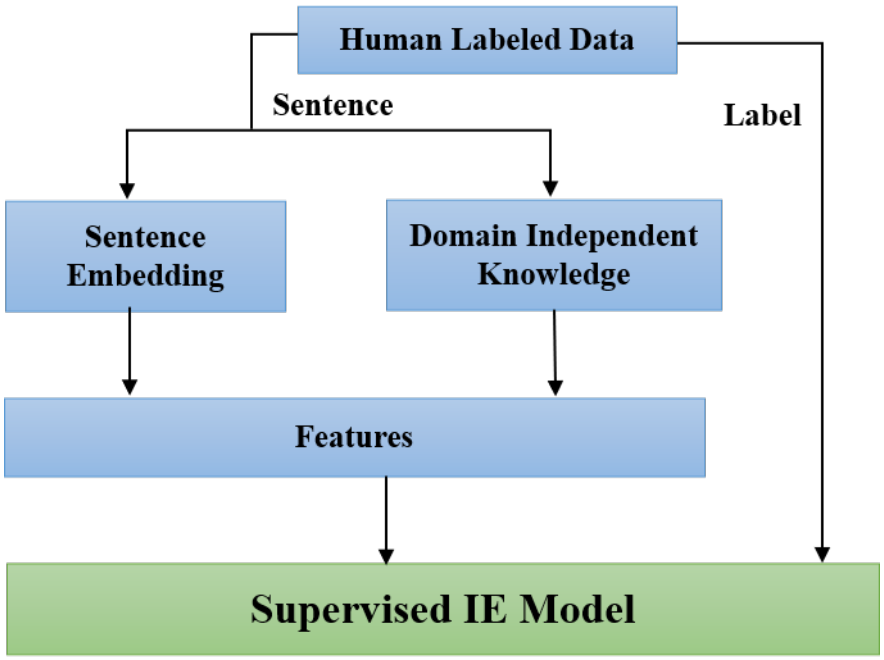


Figure 5.1: System Overview

5.4 Input Features

In this section, I describe the diverse feature sets used for clinical relation extraction.

5.4.1 Text embedding

Word embedding: To generate static word embedding a Word2Vec model is trained on the Medical Information Mart for Intensive Care (MIMIC)-III clinical

corpus [82] and the i2b2 dataset to ensure good quality word features for target application domain. Given an input sentence $\mathcal{S} = \langle w_1, w_2, w_3, ..., ..., w_n \rangle$, the word embedding of each word $w_i \in \mathcal{S}$ is defined as $E_w(w_i)$.

Sentence embedding: [CLS] token embedding from BERT is used as contextual sentence embedding in the relation extraction model. ClinicalBert [157], a pre-trained BERT model in the clinical domain is used to generate embedding. ClinicalBert is trained on the texts from BookCorpus, English Wikipedia, biomedical articles from PubMed and EMRs from the (MIMIC)-III dataset [82]. To get static sentence embedding, Doc2Vec (D2V) [158] which employs an embedding learning method similar to Word2Vec is used. Similar to Word2Vec, D2V model is trained on the Medical Information Mart for Intensive Care (MIMIC)-III clinical corpus [82] and the i2b2 dataset.

Entity embedding: Entity embedding is generated by taking the average of the embeddings of the words presented in a concept for both static and contextual embedding. For example, the embedding of entity e_2 in Table 5.1 is generated by averaging the word embeddings of "left", "carotid", "ophthalmic" and "aneurysm".

5.4.2 Domain-independent Features

POS embedding: Part of speech tag of each word, $\texttt{POS}(w_i)$ is derived using Stanford CoreNLP [159]. Then a POS embedding for each word w_i as $E_{pos}(w_i)$ is learned.

IOB encoding: To encode whether each word in $\mathcal{S}$ belongs to any of the target entity, IOB encoding $\texttt{IOB}(w_i)$ is assigned to each word. There are a total of

seven IOB tags in the system: $\langle \texttt{I}_{\texttt{treatment}}, \texttt{I}_{\texttt{test}}, \texttt{I}_{\texttt{problem}}, \texttt{B}_{\texttt{treatment}}, \texttt{B}_{\texttt{test}}, \texttt{B}_{\texttt{problem}}, \texttt{O} \rangle$, where the prefix B represents the beginning of a entity (E.g., $\texttt{B}_{\texttt{treatment}}$ represents the first word of a `treatment` entity), and I indicates a word inside a entity (e.g., $\texttt{I}_{\texttt{treatment}}$ represents an internal word of a `treatment`). The tag O represents other words not related to any of the target entity. Thereafter, each IOB tag is transformed into an IOB embedding during model trainining. Table 5.1 shows an example of the IOB encoding of the words in a sentence.

Relative distance: A relative distance encoding is employed for each of the two entities e_1, e_2 in a relation. This is done by marking the positions of all the words in a target concept as 0. Every word to its right is assigned an incrementally higher distance number and every word to its left is assigned an incrementally lower number. The embedding of the relative positions as $E_{r1}(w_i)$ and $E_{r2}(w_i)$ are learnt during model training. Table 5.1 shows an example of the relative distance encoding for the two concepts e_1, e_2 in a relation.

Dependency tree: Dependency tree is represented as a directed graph, with m nodes corresponding to each of the m words in a sentence. Figure 5.2 shows the dependency tree for an example sentence. Dependency tree is extracted using Stanford CoreNLP [159]. To create a fix sized data structure irrespective of the length of the sentence and the depth of the dependency tree, the tree is converted to an $n \times n$ adjacency matrix, $\mathcal{A}$, where n is a pre-determined fixed sentence length, If there is a edge between w_i and w_j in the dependency tree, then $\mathcal{A}_{ij} = \mathcal{A}_{ji} = 1$ and 0 otherwise. Following [160], self-loop and normalization are added into the adjacency matrix as these operations have shown to improve effectiveness. Self loop

Sentence, $\mathcal{S}$	She is status post [coiling]$_{treatment}$ with stent placement for [left carotid ophthalmic aneurysm]$_{problem}$.
e_1	coiling
e_2	left carotid ophthalmic aneurysm
Relative Distance (e_1)	$\langle -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \rangle$
Relative Distance (e_2)	$\langle -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 0, 0, 0, 1 \rangle$
IOB Encoding	$\langle$ O, O, O, O, $\mathtt{B}_{\mathtt{treatment}}$, O, O, O, O, $\mathtt{B}_{\mathtt{problem}}$, $\mathtt{I}_{\mathtt{problem}}$, $\mathtt{I}_{\mathtt{problem}}$, $\mathtt{I}_{\mathtt{problem}}$, O $\rangle$

Table 5.1: A Example of Relative Distance and IOB Encoding

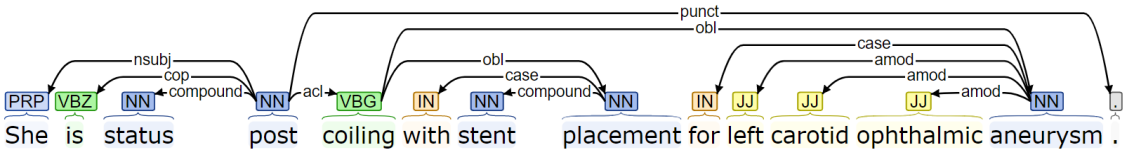


Figure 5.2: An Example of a Sentence Dependency Tree and POS Tags

is added by $\tilde{\mathcal{A}} = (\mathcal{A} + I)$ where I is an $n \times n$ identity matrix. Then normalization is performed for each row of $\tilde{\mathcal{A}}$ so that $\tilde{\mathcal{A}}_i = \tilde{\mathcal{A}}_i / d_i$ where $d_i = \sum_{j=1}^{n} A_{ij}$ is the degree of the word at the ith position in the graph.

5.5 Experiment

5.5.1 Dataset

For this study, I use the 2010 i2b2/VA dataset on Natural Language Processing Challenges for Clinical Records. This dataset contains discharge summary and progress report from different healthcare providers. My research focuses on the relation extraction task, which is to identify eight target relations among three medical concepts such as treatments, problems and tests. The dataset used for relation extraction during the 2010 i2b2/VA challenge includes 394 training reports, 477 test reports, and 877 un-annotated reports. After the challenge, however, only a part of the data was publicly released. The dataset I downloaded from the i2b2 website only

Relation Type	Train Report	Test Report	Total
Treatment improve or cure medical problem (**TrIP**)	51	152	203
Treatment worsen medical problem (**TrWP**)	24	109	133
Treatment caused medical problems (**TrCP**)	184	342	526
Treatment administered medical problem (**TrAP**)	885	1732	2617
Treatment was not administered because of medical problem (**TrNAP**)	62	112	174
Test reveal medical problem (**TeRP**)	993	2060	3053
Test conducted to investigate medical problem (**TeCP**)	166	338	504
Medical problem indicates medical problems (**PIP**)	755	1448	2203
No Relation (**None**)	7111	12821	19932

Table 5.2: Statistics of the relation extraction dataset from the 2010 i2b2/VA challenge

includes 170 documents for training and 256 documents for testing. Descriptions and statistics of the target relations can be found in table 5.2.

5.5.2 Model Architecture

To systematically investigate the advantage of contextual embedding over static embedding and the effectiveness of merging domain-independent knowledge with different types of embedding, I train several models. Here I discuss each model in detail.

Word2Vec-BiLSTM : In this model, static word embeddings are used as input to BiLSTM. BiLSTM is used to aggregate word-level features and generate sentence-level representations [161]. BiLSTM can learn the word order as well as the long-term dependency in a sentence. The advantage of using BiLSTM is that it can leverage the information from neighboring words on both sides. The output of

BiLSTM is used for classification.

Doc2Vec-Classifier : Doc2Vec sentence embedding is connected to a fully connected layer and finally to the classifier layer.

BERT-classifier : BERT is fine-tuned for the relation extraction task by adding a softmax layer on top of BERT final layer. This model only uses [CLS] token embedding as sentence embedding for classification.

Word2Vec-DIF-BiLSTM : In this model, static word embedding and independent domain features (DIF) are combined and passed through the BiLSTM model. In general, words in sentences, words in concepts, relative distances, POS tags and IOB tags are first input into an embedding layer to learn their embeddings. The embedding features are then concatenated and input to BiLSTM: $E(w_i) = E_w(w_i) \oplus E_{pos}(w_i) \oplus E_{iob}(w_i) \oplus E_{r1}(w_i) \oplus E_{r2}(w_i) \oplus \tilde{\mathcal{A}}_i$, where $\oplus$ operator denotes a concatenation operation. Therefore, $E_{BiLSTM} = \langle E(w_1), E(w_2), E(w_3), ..., ..., E(w_n) \rangle$ forms our input features for all words in a sentence. Output of BiLSTM is combined with concept embedding for relation classification. Figure 5.3 shows the architecture of Word2Vec-DIF-BiLSTM model.

BERT-DIF-BiLSTM: To combine BERT contextual sentence embedding with other features, there are two options: early fusion and late fusion. Early fusion combines BERT embedding with other features at the word/token level and then uses BiLSTM to learn a sequential representation of all the word features. With this strategy, feature fusion occurs at the word level. Late fusion combines the sentence representation learned by BERT with the sequential representation of all

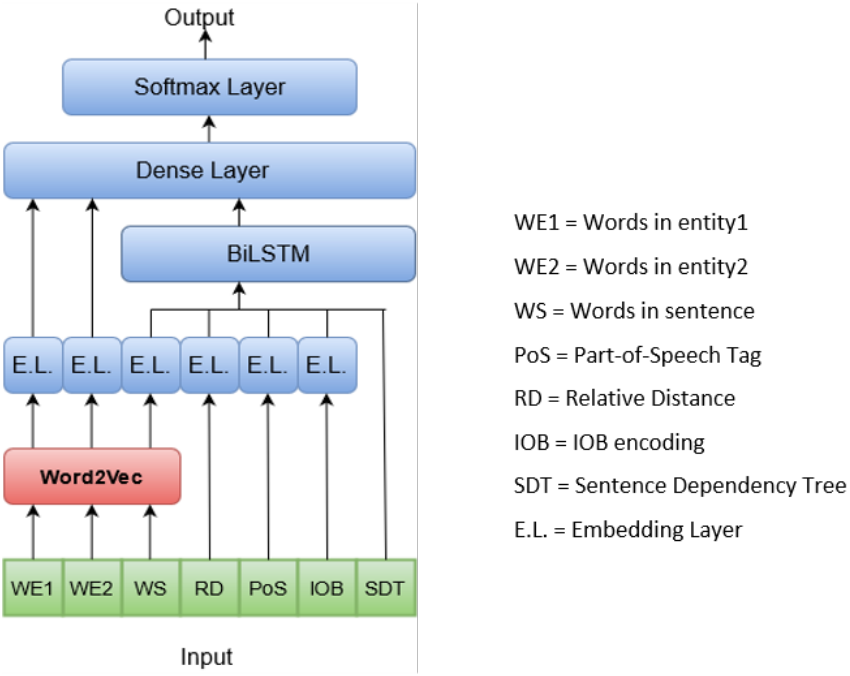


Figure 5.3: Model Architecture of Word2Vec-DIF-BiLSTM

the other features learned by BiLSTM. With this strategy, feature fusion occurs at the sentence level.

Previous research suggested that using BERT sentence embedding for classification is more effective than using BERT word/token embedding and then input them to an LSTM to generate a sentence embedding [138]. As BERT already captures long-term dependency between words in a sentence, it could be redundant to use LSTM to capture contextual information again. For this reason, in the model shown in Figure 5.4, a late fusion strategy is adapted where BERT sentence embedding is combined with a sequential representation of other word-level features (POS tag, relative distances, IOB encoding and dependency tree information) learned by LSTM. In addition, entity embeddings are learned by averaging the embeddings of the tokens in each entity. Finally, all the representations are merged and passed

through a fully connected layer for classification. In this model, all the existing BERT parameters and the new parameters in BiLSTM are fine-tuned during relation classification.

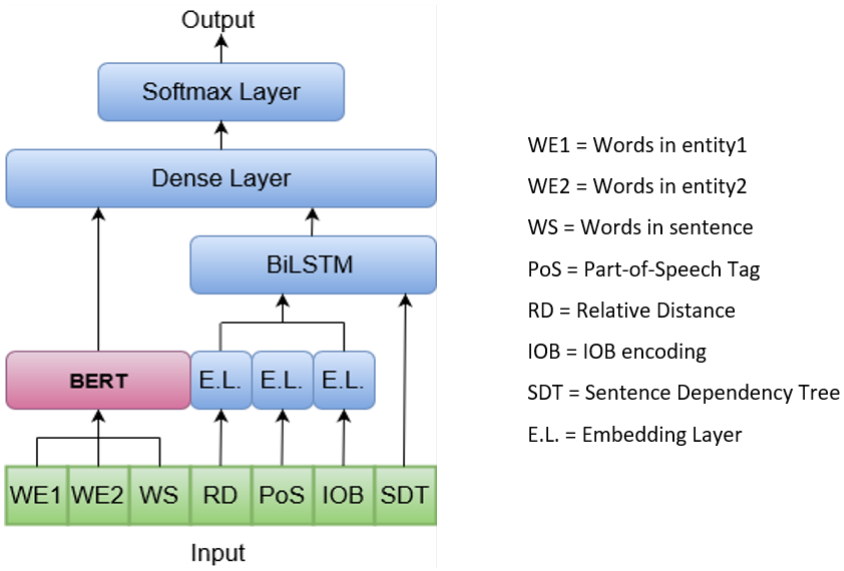


Figure 5.4: Model Architecture of BERT-DIF-BiLSTM

Doc2Vec-DIF-BiLSTM: Same late fusion is adopted to combine Doc2Vec-based static sentence embedding with the output of BiLSTM as shown figure 5.5.

5.5.3 Experiment Setting

All models with BiLSTM are implemented with 2 stacked BiLSTM layers with 128 cells in each layer, each using the $\texttt{tanh}(\cdot)$ activation function. Two fully connected layers are used for all the models before the softmax layer with 128 and 64 neurons, respectively. Adam optimizer [162] with a 0.001 learning rate is used for all the models. Batch size of 256, with 10% word-level dropout and 10% recurrent dropout, are used in training BiLSTM. To avoid over-fitting, 10% dropout is employed in the fully connected layer. The word embeddings and concept embeddings

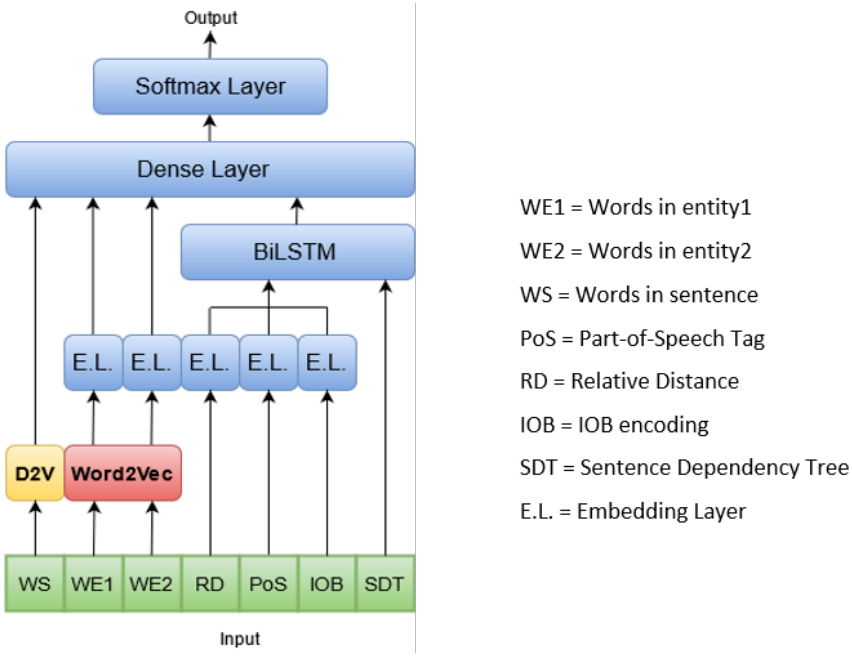


Figure 5.5: Model Architecture of Doc2Vec-DIF-BiLSTM

are initialized using 100-dimensional Word2Vec embedding and 768-dimensional BERT embedding. The embedding layers for POS, IOB encoding, Relative Positions are 20, 5, 50 respectively and randomly initialized. For relative position embedding, a specialized Position Embedding layer[1] that maps integers (negative, positive, zero) to an embedding space is used. All models are implemented using Keras[2] with a Tensorflow[3] backend. As this is a relatively small corpus, the training and testing data are combined, and 5-fold cross-validation is performed in all experiments.

Model Name	No Relation	PIP	TeCP	TeRP	TrAP	TrCP	TrIP	TrNAP	TrWP	9 Class F1 Score
Baseline[1]	N/A	0.6333	0.6117	0.8444	0.7974	0.6213	0.6159	0.4227	0.4457	0.7434
Word2Vec-biLSTM	0.8398	0.2924	0.2047	0.4536	0.5334	0.3298	0.1453	0.0621	0	0.6979
Word2Vec-DIF-BiLSTM	**0.9275**	**0.7896**	**0.6437**	**0.8685**	**0.8057**	**0.6320**	**0.5000**	0.4025	0.2262	**0.8808**
Doc2Vec	0.8057	0.0362	0	0.0165	0.0299	0	0	0	0	0.5585
Doc2Vec-DIF-BiLSTM	0.8985	0.7154	0.2931	0.7984	0.6967	0.4152	0.2222	0.1896	0.0925	0.8249
BERT	0.8438	0.4280	0.4593	0.5520	0.6068	0.5064	0.4946	**0.5372**	**0.3183**	0.7447
BERT-DIF-BiLSTM	0.8554	0.4634	0.4494	0.6788	0.6266	0.5096	0.4299	0.1860	0.0606	0.7646

Table 5.3: System Performance on Clinical Relation Extraction

5.5.4 Performance Evaluation

This section evaluates (i) the effectiveness of static versus contextual text embedding and (i) the efficacy of domain-independent knowledge combined with text embedding.

As shown in table 5.3, BERT sentence embedding on its own is the most useful feature for relation extraction. BERT-Classifier model only using contextual embedding outperforms model using both static word embedding and sentence embedding. BERT-classifier achieves F1=0.7447 over Word2Vec-BiLSTM with F1=0.6974 and Doc2Vec-Classifier with F1=0.5585. This result highlights the power of BERT contextual embedding in capturing the semantics of a sentence.

However, adding domain-independent knowledge to BERT embedding (BERT-DIF-BiLSTM) provides only 2.7% improvement of the F1 score. In contrast, adding the same features to static Word2Vec embedding (Word2Vec-DIF-BiLSTM) has resulted in a 26% increase in performance. The performance of BERT-DIF-BiLSTM

[1]https://github.com/CyberZHG/keras-pos-embd
[2]https://keras.io/
[3]https://www.tensorflow.org/

(F1=0.7646) is 15% lower than Word2Vec-DIF-BiLSTM (F1=0.8808). This result is quite surprising as it implies some incompatibility between BERT and other traditional NLP features, which prevents them from being combined effectively. In contrast, static word embeddings (e.g., Word2Vec) do not seem to suffer from the same problem. One possible explanation could be that the late fusion strategy adopted to combine BERT embeddings with other features differs from the early fusion strategy used to combine Word2Vec embeddings. To test this hypothesis, instead of merging word embedding with other word-level features before inputting to BiLSTM, I adopt late fusion to combine Doc2Vec-based sentence embedding with the output of BiLSTM. As shown in Table 5.3, Doc2Vec-Classifier on its own (F1=0.5622) is much worse than either BERT-Classifier (F1=0.7447) or Word2Vec-Classifier (F1=0.6974). However, when combining Doc2Vec with other features (Doc2Vec-DIF-BiLSTM), the performance improvement is 49%, which is the highest among all the text embedding models. Doc2Vec-DIF-BiLSTM (F1=0.8249) also outperformed BERT-DIF-BiLSTM (F1=0.7646) by (7.8%). Since the model with Doc2Vec also adopts a late fusion strategy, late fusion may not directly cause the poor performance of BERT models.

In summary, if used alone, BERT embedding is the best for relation extraction. However, Word2Vec-based static embedding works the best when combined with domain-independent knowledge. In contrast, combining domain-independent knowledge with contextual embedding has a limited positive impact on performance.

5.6 Summary

In this research, I investigate effectiveness of combining domain independent knowledge with contextual embedding for relation extraction from clinical texts. Experiment results show that although contextual embedding learned by BERT on its own is very effective, it performed poorly when combined with domain-independent knowledge. Combining domain independent knowledge with static embedding is very effective for such a task.

Chapter 6: Combining Contextual Embedding with Domain-specific Knowledge for Clinical Relation Extraction

6.1 Research Overview

In recent years contextual embedding from pre-trained language models (PLMs) such as ELMo [23], BERT [163], XLNet [164], and GPT [165] have become very popular as they can effectively boost the performance of diverse NLP tasks such as information extraction [166, 167], sentiment analysis [168], question answering [169] and language entailment [163]. These models are trained on large text corpora using self-supervised tasks such as masked language modeling (MLM) and next sentence prediction. These models can learn meaningful context-sensitive text embeddings, and they are frequently used to encode input text in many downstream text analysis tasks. However, PLMs trained on general-domain text (e.g., books, Wikipedia and web data) may not be ideal for domain-specific NLP applications (e.g., bio-medical NLP). In this research, I explore how domain-specific semantic knowledge can be added to contextual embedding. In particular, I investigate how medical knowledge can be combined with contextual embedding to facilitate clinical relation extraction.

Previously, significant effort has been made to add domain knowledge into contextual embedding. Based on the types of knowledge added, we can group the work into two categories: integrating domain text [80, 83, 84] and integrating domain-

specific knowledge [88, 170, 171]. In this research, I conduct a comprehensive investigation of these methods. I test their effectiveness in integrating knowledge from Unified Medical Language System (UMLS) into BERT for clinical relation extraction. I focus on UMLS because it is one of the most widely used bio-medical knowledge sources for clinical NLP.

The main contributions of this work include-

- I conduct a comprehensive empirical analysis of the effectiveness of applying diverse knowledge integration techniques to combine medical knowledge encoded in UMLS with contextual embeddings from pre-trained BERT models for clinical relation extraction.

- I develop several new knowledge fusion methods such as ClinicalBERT-EE-RI-CT/ST/SG, ClinicalBERT-EE-ED-CT and ClinicalBERT-EE-KB-MLM for clinical relation extraction.

- Proposed method ClinicalBERT-EE-RI-ST achieves the state of the art performance on a benchmark clinical relation extraction dataset.

6.2 UMLS

The Unified Medical Language System (UMLS) [33] is a repository of biomedical vocabularies developed by the National Library of Medicine of US. It has three Knowledge Sources: the Metathesaurus, Semantic Network and SPECIALIST Lexicon and Lexical Tools.

The Metathesaurus integrates millions of concepts from over 200 vocabular-

ies, including CPT, ICD-10-CM, LOINC, MeSH, RxNorm, and SNOMED CT. The Metathesaurus is organized by concepts. Each concept is characterized by a unique concept identifier (CUI), definition, attributes and relationships with other concepts. Additionally, concepts are linked to the corresponding concept names in the various source vocabularies. For example, the concept "Headache" has CUI C0018681, a definition of "The symptom of pain in the craninal region" and is related to the concept "Acetanilide" (CUI C0000973) with the relation of type "may_treat", and connected to "Ache" (CUI C0234238) with "is_a" relation. "Headache" has a total of 1165 relations with other concepts. The Metathesaurus mentions if different vocabularies use different names for the same concept or use the same name for multiple concepts. The Metathesaurus retains all hierarchical information from source vocabularies. UMLS 2018AB release has a total of 3.82 million concepts and 33.5 million relations.

Semantic Network provides consistent categorization of all concepts represented in the UMLS Metathesaurus. Each concept in the Metathesaurus is assigned one or more semantic types linked with each other through semantic relationships. The links among semantic types define the network structure and show relationships between the groupings and concepts. The primary link between semantic types is the "is_a" link, establishing a hierarchy of types. Semantic Network groups concepts according to their semantic types. It helps to reduce the complexity of the Metathesaurus by interpreting the meaning of the Metathesaurus concept. Each semantic type has an identifier, a definition, a few examples, and relationships. For example, the semantic type of "Headache" is "Sign or Symptom". The definition of "Sign or

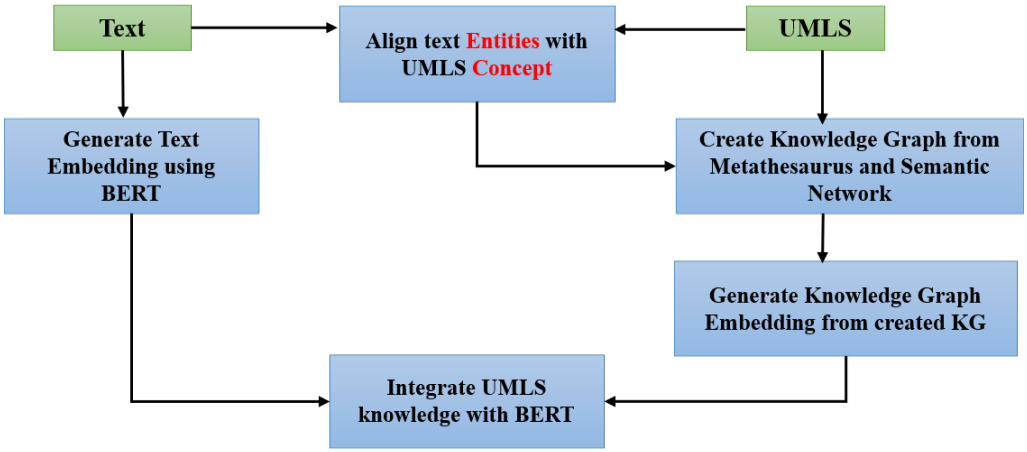


Figure 6.1: Methodology Overview

Symptom" is "An observable manifestation of a disease or condition based on clinical judgment or a manifestation of a disease or condition which is experienced by the patient and reported as a subjective observation". "Sign or Symptom" is a "finding" and has "associated with" relation with semantic type "Anatomical Abnormality" and "manifestation of" relation with semantic type "Pathologic Function". There are 127 semantic types and 54 relationships in total.

SPECIALIST Lexicon and Lexical Tools include a large syntactic lexicon and tools for normalizing strings, generating lexical variants, and creating indexes.

6.3 Methodology

The main steps in this research include (a) generating text embeddings using BERT, (b) aligning the entities in text with the concepts in UMLS, (c) creating knowledge graph from Metathesaurus and Semantic Network, (d) generating knowledge graph embeddings and (e) integrating UMLS knowledge with BERT. Figure 6.1 shows overview of this methodology.

6.3.1 Generating Text Embeddings Using BERT

Study [172] shows that incorporating information about the target entities along with BERT sentence representation greatly benefits relation classification. To implement this, given a sentence S, four markers e11, e12, e21 and e22 are inserted at the beginning and end of the two target entities (e1,e2) in a relation. In this study's i2b2 relation extraction dataset, the ground truth entity locations were provided as the input to the relation extraction model. After inserting these special tokens, for a sentence "The patient was given ibuprofen for high fever." with target entities "ibuprofen" and "fever" becomes: "[CLS] The patient was given e11 ibuprofen e12 for high e21 fever 22 . [SEP]". Based on the positions of the two target entities in the BERT embedding, entity embeddings (EE) can be calculated. Then sentence embedding derived from the [CLS] token embedding and the entity embeddings are concatenated and passed through a fully connected layer to generate a representation that contains both sentence and entity embeddings (BERT+EE). Figure 6.2 shows model architecture of BERT sentence classifier and Figure 6.3 shows model architecture of BERT-EE.

6.3.2 Text and UMLS Concept Alignment

To incorporate domain knowledge from UMLS into BERT, the first step is to identify UMLS concepts in clinical notes. Apache cTAKES [173] is used to extract named entity mentions in clinical notes and align them with the concepts in UMLS. cTAKES is a clinical Text Analysis and Knowledge Extraction System that extracts clinical information from unstructured text. It processes clinical notes and

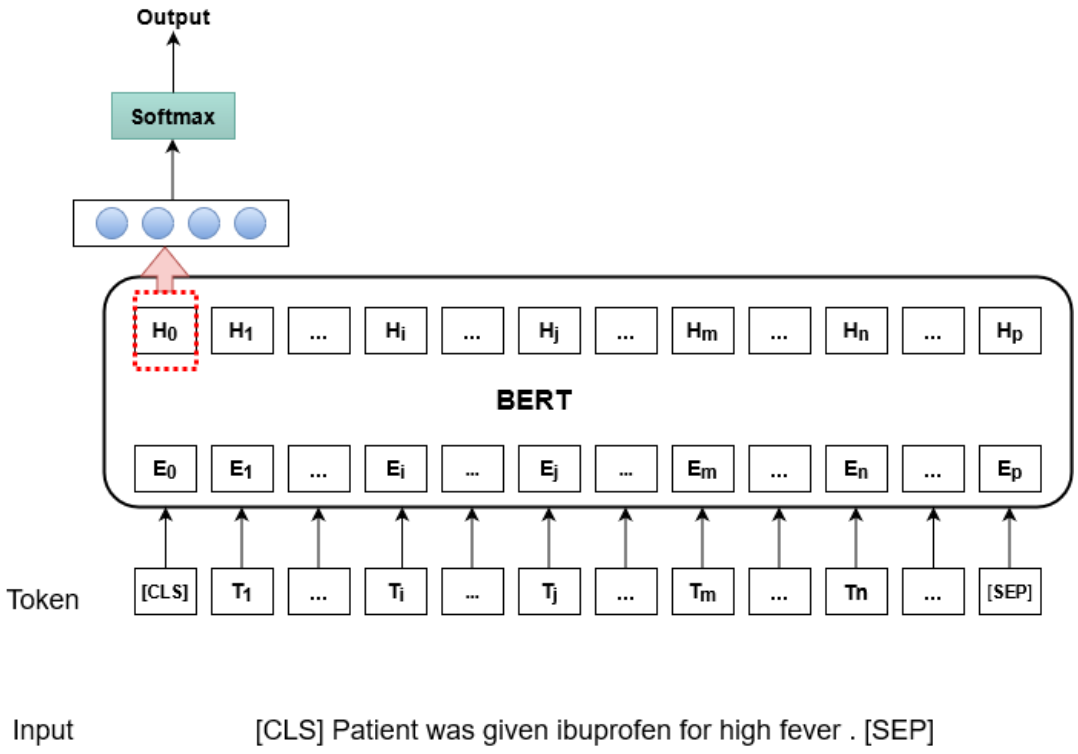


Figure 6.2: BERT Architecture for Sentence Classification

identifies clinical named entities such as drugs, diseases/disorders, signs/symptoms, anatomical sites and procedures, and maps them to UMLS concepts. Figure 6.4 shows an example of cTAKES output. Using cTAKES 46305 out of 58688 entities in the dataset can be mapped to UMLS concepts.

6.3.3 Creating Knowledge Graph from Metathesaurus and Semantic Network

The Metathesaurus and the Semantic Network can be considered multi-relational knowledge graphs with nodes corresponding to concepts or semantic types and edges to relations. Each relationship is represented as a triplet (h, r, t), indicating a relationship (r) between two nodes (h and t). A subset of the Metathesauras and the complete semantic network are used to create the knowledge graph (KG) for

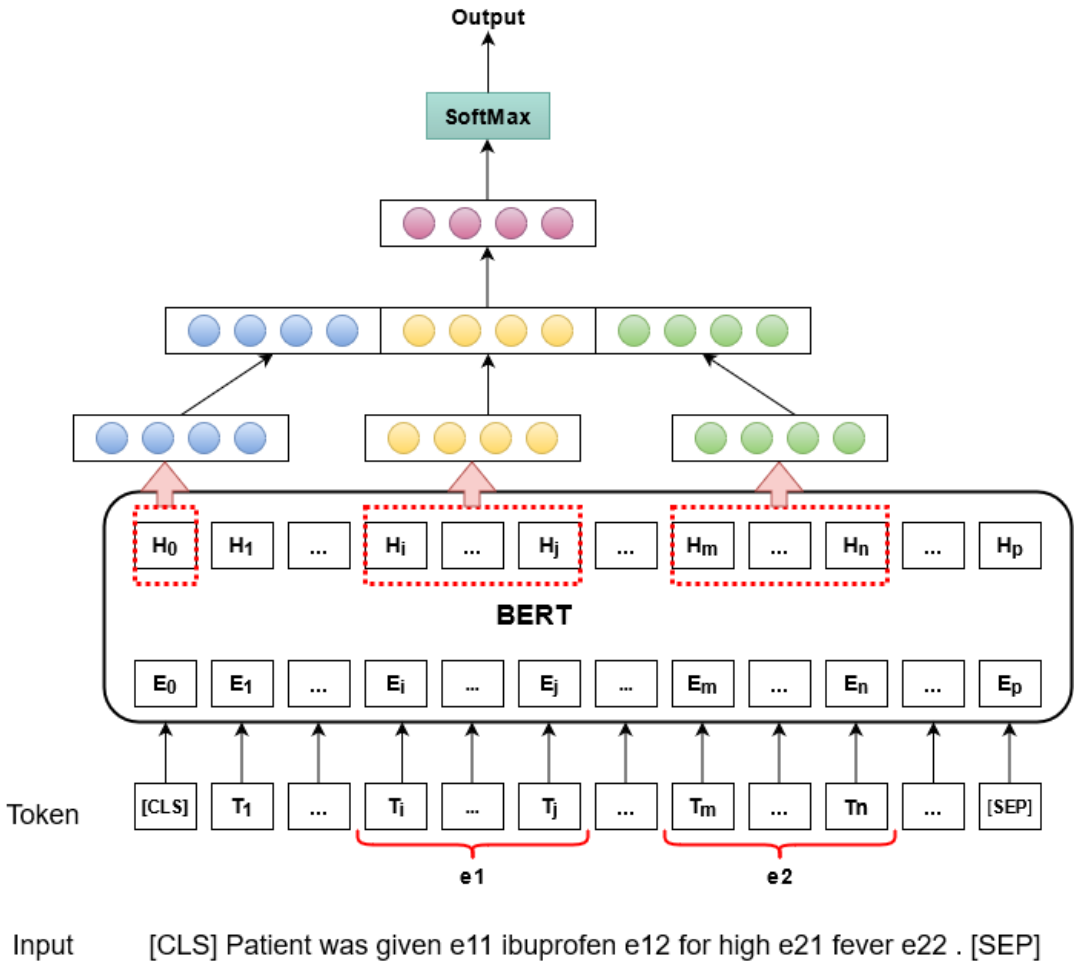


Figure 6.3: Model Architecture of BERT-EE

this research. Specifically, all the CUIs extracted from our dataset using cTAKES are selected. Then a subset of the Metathesauras is selected by collecting all CUIs and relations that are one hop away from the initial set of CUIs. This graph is connected with the semantic network by including CUI and their semantic type relationships. This created knowledge graph contains 312474 nodes and 1613019 relations. Figure 6.5 shows an example of the created knowledge graph.

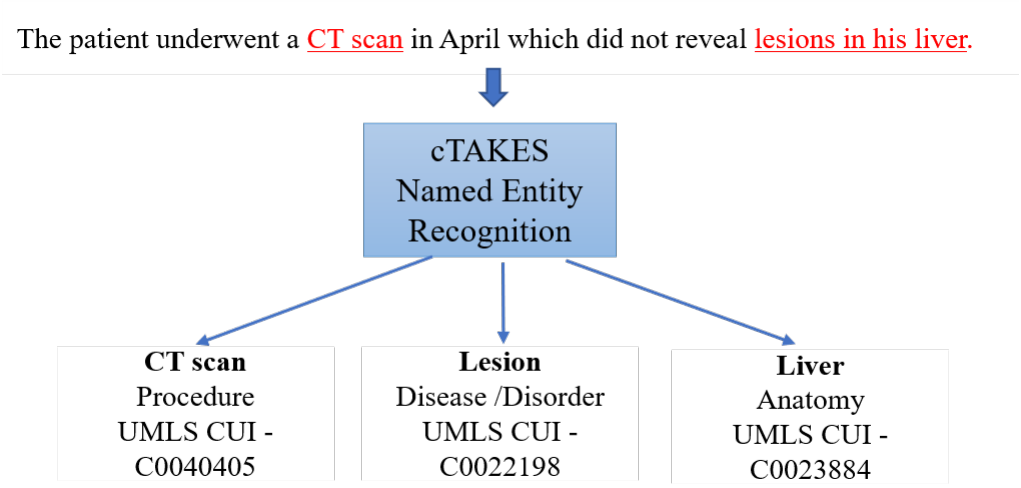


Figure 6.4: Example of cTAKES Output

6.3.4 Generating Knowledge Graph Embeddings

Once the knowledge graph is created, multiple popular KGE models such as TransE [174], DistMult [175], ComplEx [176], [177] and ConvKB [178] are employed to create knowledge graph embeddings. TransE is a translation-based model that uses a distance-based scoring function. DistMult and ComplEx are Semantic matching models that exploit similarity-based scoring functions. ComplEx is an extension of DistMult that uses complex-valued embedding vectors that contain complex numbers to handle asymmetric data. ConvE and ConveKB are neural network-based models that use convolution networks. The effectiveness of these methods is evaluated on a link prediction task, which predicts an entity that has a specific relation with a given entity, i.e., predicting h given (r, t) or t given (h, r). Among these KGE methods, ComplEx performs the best on the link prediction task. As a result, knowledge graph embeddings from ComplEx are used in the experiments. Table 6.1 shows the results of different KGE methods. From KGE, concept embedding, se-

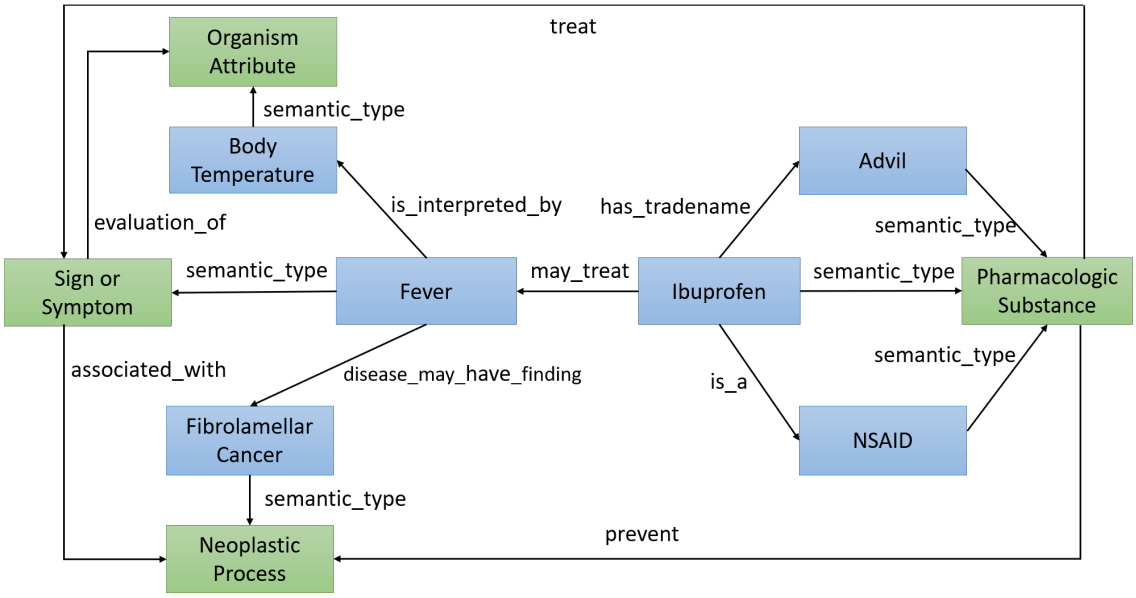


Figure 6.5: A Snippet of a Knowledge Graph Created from UMLS

mantic type embedding, semantic group embedding and relation embedding are collected.

6.3.5 Integrating UMLS Knowledge with BERT

In the experiments, I primarily use ClinicalBERT trained on clinical text corpora. I systematically investigate different techniques to infuse knowledge from UMLS with Pre-trained ClinicalBERT. The methods I have examined include:

ClinicalBERT-EE-KGE: The first technique is to combine knowledge graph embedding with the text embeddings from ClinicalBERT and feed them to the relation classifier. For two entities in an input sentence, I retrieve their respective concept embeddings (CT), semantic type embeddings (ST) and semantic group embeddings (SG) from KGE. In addition, for a pair of concepts mapped from two entities in a sentence, I use KGE to predict the UMLS relation between them. Then I retrieve the UMLS relation embedding from KGE. Finally, I concatenate all the

Method	MRR	Hits@10	Hits@3	Hits@1
TransE	0.3285	0.488	0.374	0.24
DistMult	0.68367	0.93	0.7995	0.544
ComplEx	**0.8778**	**0.941**	**0.914**	**0.8355**
ConvE	0.4159	0.636	0.4615	0.3055
ConvKB	0.3574	0.5545	0.395	0.254

Table 6.1: Result of link prediction task

KGE embeddings with the sentence and entity embeddings from ClinicalBERT for relation classification. Please note that text embeddings and knowledge graph embeddings are in two separate embedding spaces in this approach.

ClinicalBERT-EE-MLP: Effectively merging knowledge graph embeddings with BERT can be tricky. Because pre-trained language models, such as BERT, are often trained for 2 to 5 epochs with a smaller learning rate during fine-tuning, whereas graph embedding features extracted from KGE need to be trained for much longer with a higher learning rate. If BERT output is directly concatenated with the KGE features, the relation classifier might not benefit much from the KGE features. I first train a multi-layer perceptron (MLP) with knowledge graph embeddings for relation classification to solve this issue. The output of the hidden layer of the trained MLP is combined with BERT text embeddings for relation classification. The use of a trained MLP ensures that the KGE features do not underfit when

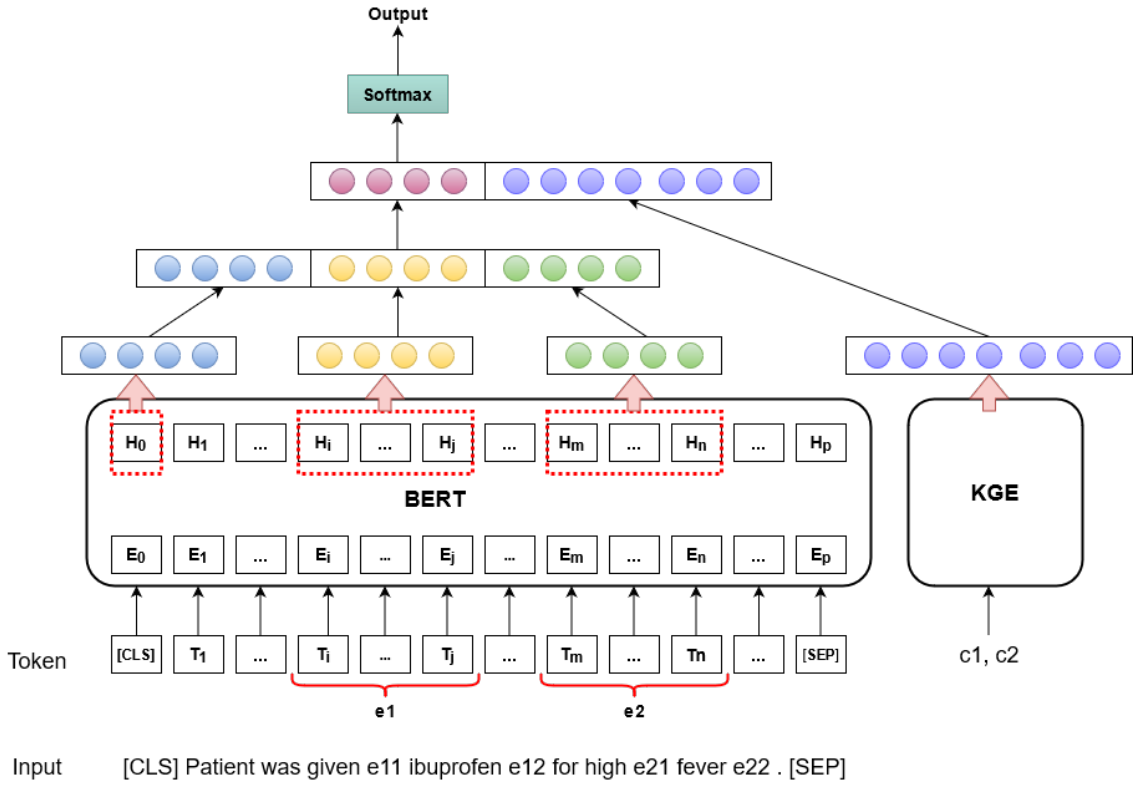


Figure 6.6: Model Architecture of ClinicalBERT-EE-KGE

trained in an ensemble with pre-trained BERT models for a small number of epochs.

ClinicalBERT with Relation Indicator: In each input sentence, relevant knowledge from a knowledge graph is injected into an input sentence to BERT, transforming the original sentence into a knowledge-enriched text input. I add knowledge from UMLS as the second sentence in the BERT input. Then both the original input sentence and the synthesized second sentence are sent to pre-trained ClinicalBERT. ClinicalBERT uses these knowledge enriched sentences to predict relation labels. With this method, UMLS knowledge is injected directly into the BERT embedding space. To construct this second sentence, first, I find corresponding CUIs for the two entities in a sentence using cTAKES. Then I use pre-trained KGE to predict

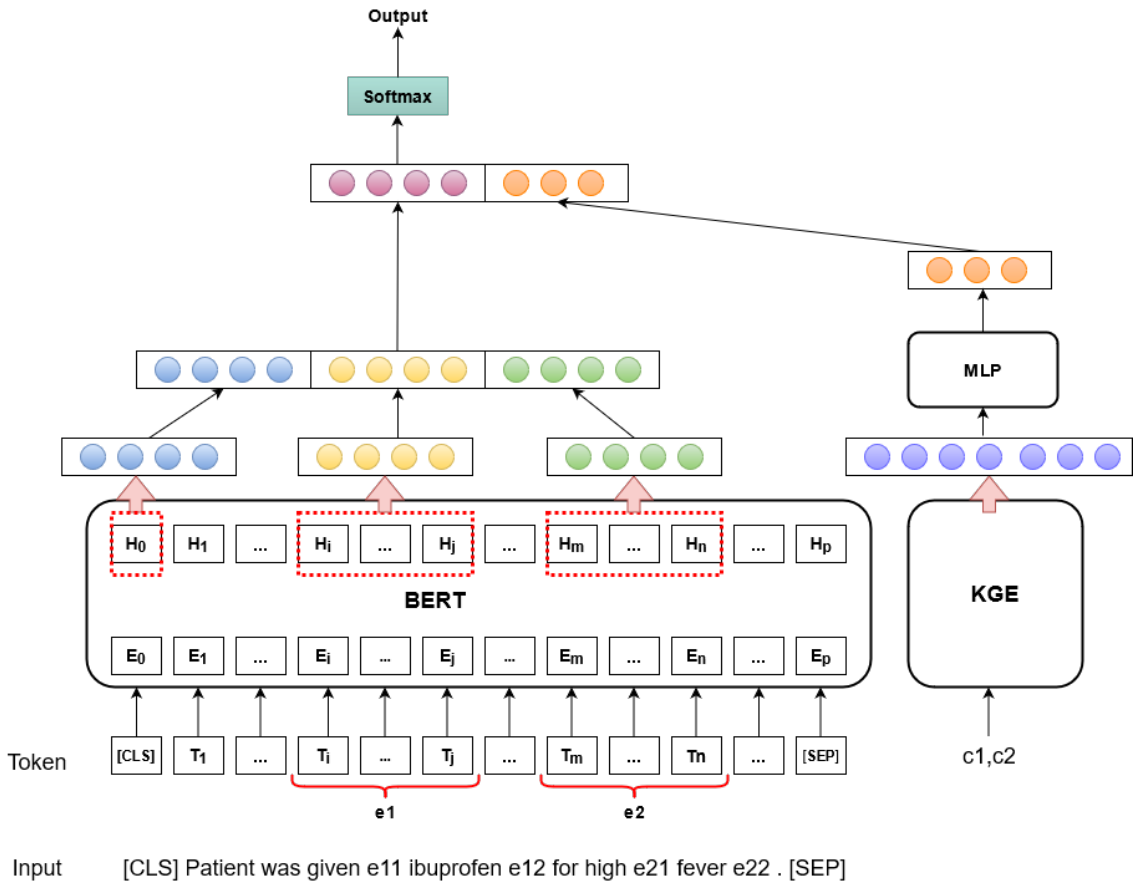


Figure 6.7: Model Architecture of ClinicalBERT-EE-MLP

the UMLS relation between them. Then I construct the second input sentence in the form of "concept1 relation concept2". For the input sentence, "The patient was given ibuprofen for high fever", I first map "ibuprofen" and "fever" to their UMLS CUIs. Pre-trained KGE predicts the UMLS relation between them is "may_treat". Then I construct the second sentence as "ibuprofen may treat fever". This KGE-predicted UMLS relation can potentially act as a relation indicator that may help to differentiate relation class labels. To pass this relation indicator information to BERT, I use special tokens before and after the relation indicator phrase. These tokens are used to extract the relation indicator embedding from BERT. Finally, the

combined sentence embedding, entity embedding and relation indicator embedding are used for relation classification. For example, the final input would be "[CLS] Patient was given e11 ibuprofen e12 for high e21 fever e22 . [SEP] ibuprofen r31 may treat r32 fever . [SEP]".

I also try a variety of the second sentence where its semantic type or semantic group replaces each entity. In the same example, the second sentence would be "pharmacologic substance r31 may treat r31 Sign or Symptom" or "drug and chemical r31 may treat r31 disease and disorder", where "pharmacologic substance" and "sign or symptom" are the semantic types of "ibuprofen" and "fever", and "drug and chemical" and "disease and disorder" are the semantic groups of "ibuprofen" and "fever". These models are called ClinicalBERT-EE-RI-CT, ClinicalBERT-EE-RI-ST and ClinicalBERT-EE-RI-SG where "RI" stands for "relation indicator", "CT" for "concept", "ST" for "semantic type", "SG" for "semantic group".

ClinicalBERT with Entity Definition: In this method, I fine-tune BERT with input sentences and the text descriptions of the two entities. For entities in an input sentence, I extract their corresponding concept definitions from UMLS. They are used as the input to BERT to get concept embeddings (ClinicalBERT-EE-ED-CT). I also generate semantic type embeddings using its definitions (ClinicalBERT-EE-ED-ST). These definitions are fed to a separate BERT model as input. Text representations are extracted based on the special entity markers I inserted. I use the [CLS] token embeddings related to the concept and semantic type definitions as the concept and semantic type embeddings. These embeddings are concatenated

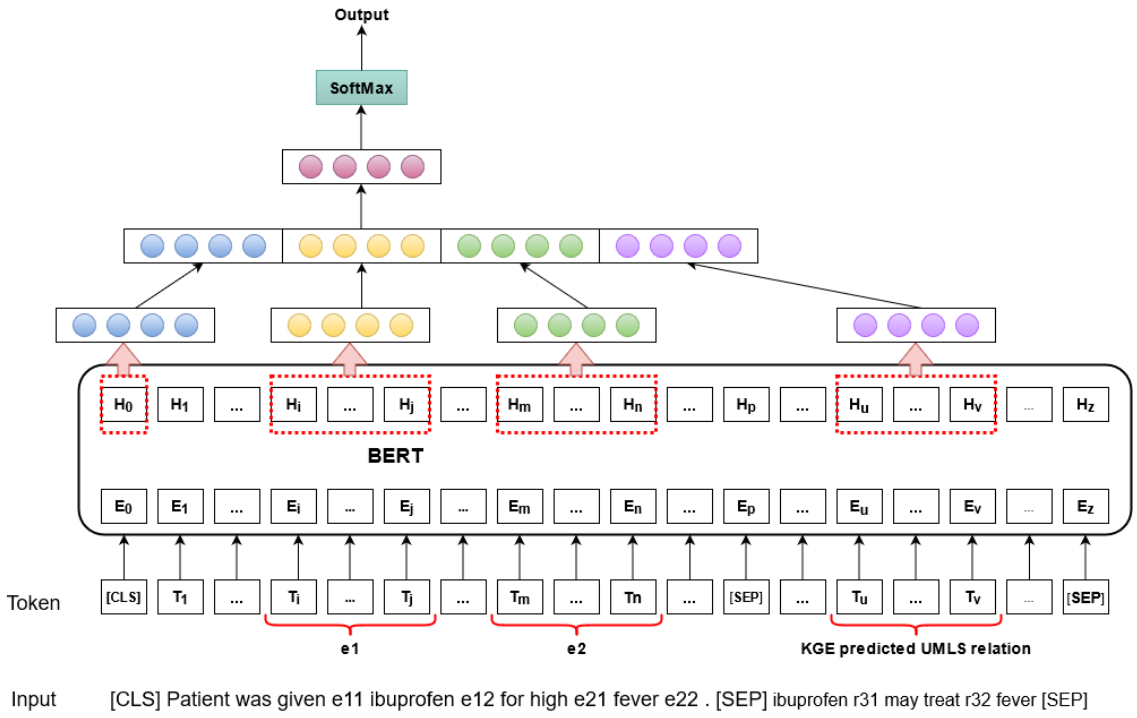


Figure 6.8: ClinicalBERT with Relation Indicator

with text embeddings of the input sentence before relation classification.

ClinicalBERT-EE-KB: UMLS knowledge is infused into BERT by jointly optimizing both a knowledge graph objective and a masked language model objective. Jointly optimizing the two objectives can implicitly integrate knowledge from external knowledge graphs into language models. Here I adopt the pre-trained Clinical KB-BERT [87] in my analysis.

ClinicalBERT-EE-KB-MLM: In this method, I pre-train BERT with UMLS information with only the masked language model (MLM) objective. I use the abbreviations provided by UMLS to map a triple into a natural language sentence (e.g., generating a sentence like "fever may be treated by ibuprofen" based on the triple (fever, may_be_treated_by, ibuprofen). In this way, I can get a set of

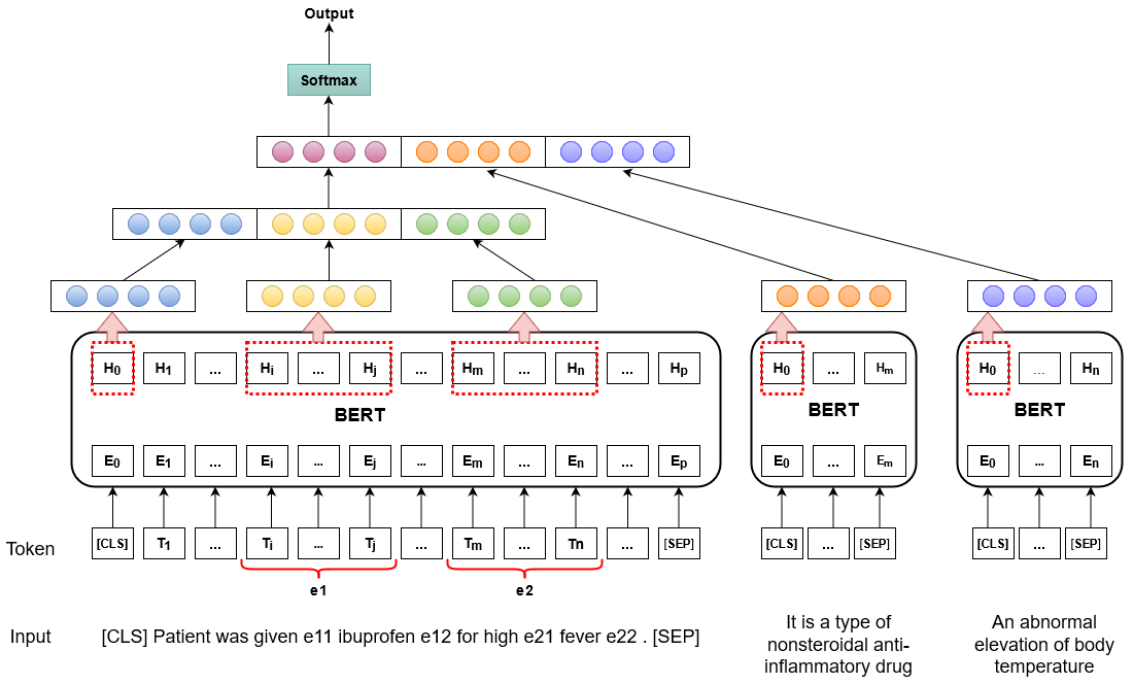


Figure 6.9: Model Architecture of ClinicalBERT with Entity Definition

sentences based on the triples in UMLS. I have created a total of 1613019 UMLS sentences. I then fine-tune ClinicalBERT with these UMLS sentences using only the MLM objective. By transferring the knowledge graph into natural language texts, I fuse UMLS knowledge with BERT in the same representation space.

Summary of Methods: Table 6.2 summarizes the methods I develop to add domain knowledge to BERT. I characterize them along multiple dimensions: infusion stage, type of domain knowledge added, the form of domain knowledge added and fusion methods.

Fusion stage: Domain knowledge fusion can happen (a) during BERT model training, which results in a BERT model that is aware of the domain information encoded in clinical notes or UMLS (BERT-train) and (b) during BERT prediction, where domain knowledge is combined with the input or output of BERT in relation

classification (BERT-PredIn or BERT-PredOut).

Knowledge type: Additional domain knowledge can be characterized into (a) domain text corpora such as clinical notes or PubMed publications (Text-Corpora), (b) UMLS concept, semantic type and semantic group, relation as well as UMLS triples with two entities and one relation (UMLS-CT, UMLS-ST, UMLS-SG , UMLS-RE, UMLS-triple) and (c) UMLS concept and semantic type definition (UMLS-CTD and UMLS-STD).

Knowledge form: Before fusion, domain knowledge is transformed into (a) embedding features extracted from KGE or KGE with MLP fine turning (embedding), (b) texts which are either synthetic sentences generated from UMLS or entity descriptions extracted from UMLS (text) and (c) training objective where knowledge graph training objective is combined with the BERT training objective to fuse knowledge in BERT(Training-Obj).

Fusion methods: Finally, in terms of fusion methods, we characterize them into (a) concatenation where BERT features are concatenated with knowledge graph features for relation classification, (b) joint optimization with both BERT and knowledge graph objectives (Joint-Opt), (c) BERT fine-tuning on sentences synthesized from UMLS using only the BERT training objective (BERT-Tune) and (d) BERT-fusion where additional domain knowledge is provided as the second sentence to BERT so that BERT itself becomes the fusion mechanism to combine domain knowledge with each input sentence (BERT-Fuse).

Method	Stage	Knowledge Type	Knowledge Form	Fusion Methods
ClinicalBERT-EE-KGE	BERT-PredOut	UMLS-(CT,ST,SG,RE)	Embedding	Concatenate
ClinicalBERT-EE-MLP	BERT-PredOut	UMLS-(CT,ST,SG,RE)	Embedding	Concatenate
ClinicalBERT-EE-RI-CT	BERT-PredIn	UMLS-(CT,RE)	Text	BERT-Fuse
ClinicalBERT-EE-RI-ST	BERT-PredIn	UMLS-(CT,RE)	Text	BERT-Fuse
ClinicalBERT-EE-RI-SG	BERT-PredIn	UMLS-(SG,RE)	Text	BERT-Fuse
ClinicalBERT-EE-ED-CT	BERT-PredIn	UMLS-CTD	Text	BERT-Fuse
ClinicalBERT-EE-ED-ST	BERT-PredIn	UMLS-STD	Text	BERT-Fuse
ClinicalBERT-EE-KB	BERT-train	UMLS-Triple	Training-Obj	Joint-Opt
ClinicalBERT-EE-KB-MLM	BERT-train	UMLS-Triple	Text	BERT-Tune

Table 6.2: Summary of the knowledge fusion methods

6.4 Experiments and Results

6.4.1 Dataset Description

For this study, I use the same clinical relation extraction dataset used in Chapter 5. This dataset contains discharge summaries and progress reports from different healthcare providers. The relation extraction task is to identify nine target relations between three types of medical concepts: treatments, problems and tests. Descriptions and statistics of the target relations can be found in table 5.2.

6.4.2 Experiment Details

In all the BERT-based classifiers, I use both the BERT sentence embedding and entity embedding (EE). I train all classifiers for five epochs with a learning rate of 0.00002 and a batch size of 8 with a softmax classification layer. In addition, for **ClinicalBERT-EE-KGE**, I concatenate 700 dimensional KGE with 768 dimensional BERT text representations. This 1468 dimensional vector is used as the

input to the softmax layer. To implement **ClinicalBERT-EE-MLP**, I first train a MLP with KGE as the input. I use a single hidden layer consisting of 128 hidden units, a tanh activation function and a final linear layer with a softmax function to make predictions. I train this model for 50 epochs with a learning rate of 0.001 and a batch size of 32. After the MLP model is trained, I combine the output of the hidden layer (a 128-dimensional vector) with the BERT text embeddings (a 768-dimensional vector). This combined vector is connected to a linear layer and a softmax function to make predictions. In **ClinicalBERT-EE-RI-CT/ST/SG**, I employ different variations of the second input sentence. Text embedding is created by combining sentence embedding, entity embedding and relation indicator embedding. I connect the combined embeddings to a fully connected layer. In addition, I combine sentence embedding (768-dimensional vector) with two concept embedding (each a 768-dimensional vector) to create a 2304 dimensional input vector in **ClinicalBERT-EE-ED-CT/ST)**. **ClinicalBERT-EE-KB** uses only the 768-dimensional text embedding. I pre-train **ClinicalBERT-EE-KB-MLM** with 4.4 million tokens from the text dataset created from the UMLS knowledge graph with 425K steps and a learning rate of 0.00002. I initialized this model with weights from ClinicalBERT.

6.4.3 Baseline Methods

To compare with the current state-of-the-art, I consider systems that employ the same number of training instances and define the classification task with the same granularity level. So far, I have found only one existing systems [135] meeting

these criteria. In addition, [179] and [89] also integrate UMLS knowledge into BERT for this task. Their reported weighted F1 scores are 0.747 and 0.782, respectively. But I did not include them in table 5.3 because either the total number of relations considered or the classification granularity (number of relation classes) are different.

To systematically investigate the effectiveness of different knowledge fusion methods, I consider multiple baselines. To evaluate the advantages of text embeddings generated from BERT over static embedding models (e.g. Word2Vec, Doc2Vec), I include baselines with static embeddings. I train a Word2Vec [18] and Doc2Vec [158] model using the MIMIC-III clinical corpus [82] plus the sentences in the i2b2 dataset. I implement the following baselines with static text embeddings.

- (Word2Vec+BiLSTM) : To generate sentence representations, I use a bidirectional LSTM to aggregate word embeddings in a sentence. This sentence representation is fed into a fully connected layer with ReLu activation and a linear layer with softmax activation.

- KGE+Word2Vec+BiLSTM : Here, I concatenate pre-trained knowledge graph embeddings with sentence embeddings generated by Word2Vec+BiLSTM. These combined embeddings are used as the input to a fully connected layer with ReLu activation and a linear layer with softmax activation.

- Doc2Vec : I use Doc2Vec generated sentence representations for classification.

- KGE+Doc2Vec : I combine Doc2Vec sentence representations with pre-trained knowledge graph embeddings.

To show the impact of domain text, I consider a baseline model where I use text embedding from BERT trained on general domain text (BERT-EE). I also consider a baseline where I only use sentence representations from BERT (BERT) to show the advantage of incorporating entity embeddings.

6.4.4 Performance Evaluation

Model Name	No Relation	PIP	TeCP	TeRP	TrAP	TrCP	TrIP	TrNAP	TrWP	9 Class F1 Score
[135]	N/A	0.6333	0.6117	0.8444	0.7974	0.6213	0.6159	0.4227	0.4457	0.7434
Word2Vec-biLSTM	0.8398	0.2924	0.2047	0.4536	0.5334	0.3298	0.1453	0.0621	0	0.6979
KGE-Word2Vec-biLSTM	0.8610	0.4554	0.4374	0.6939	0.64040	0.4604	0.4141	0.3537	0.0719	0.7693
Doc2Vec	0.8057	0.0362	0	0.0165	0.0299	0	0	0	0	0.5585
KGE-Doc2Vec	0.8277	0.0091	0	0.5434	0.2604	0.0689	0	0	0	0.6468
BERT	0.8529	0.3496	0.4508	0.4059	0.5487	0.5229	0.4827	0.6153	0.4878	0.7256
BERT-EE	0.94252	0.8005	0.6870	0.8940	0.8438	0.7109	0.7059	0.7288	0.5053	0.9039
ClinicalBERT-EE	0.9473	0.8085	0.7102	0.9045	0.8654	0.7559	0.6935	0.7469	0.5304	0.9127
ClinicalBERT-EE-KGE	0.9486	0.8149	0.7373	0.9058	0.8693	0.7671	0.7027	0.7945	0.51851	0.9162
ClinicalBERT-EE-MLP	0.9459	0.8209	0.7019	0.9008	0.8616	0.7823	0.7449	0.7177	0.5087	0.9124
ClinicalBERT-EE-RI-CT	0.9456	0.8277	0.7115	0.9003	0.8684	0.7745	0.7489	0.7490	0.4685	0.9133
ClinicalBERT-EE-RI-ST	**0.9490**	0.8270	0.7358	**0.9146**	0.8691	**0.7980**	0.7218	0.7955	0.4835	**0.9181**
ClinicalBERT-EE-RI-SG	0.9469	0.8290	0.7195	0.9018	0.8605	0.7740	0.7331	0.7404	0.5460	0.9140
ClinicalBERT-EE-ED-CT	0.9449	0.8205	0.7357	0.9074	**0.8721**	0.7081	**0.7567**	0.7415	**0.6037**	0.9137
ClinicalBERT-EE-ED-ST	0.9439	0.8128	0.7113	0.8994	0.8563	0.7692	0.7123	0.7536	0.5806	0.9107
ClinicalBERT-EE-KB	0.9473	**0.8301**	**0.7429**	0.9102	0.8792	0.7821	0.7435	**0.8195**	0.5094	0.9177
ClinicalBERT-EE-KB-MLM	0.9425	0.8211	0.6967	0.8947	0.8597	0.7112	0.7446	0.7491	0.5150	0.9078

Table 6.3: System Performance on Clinical Relation Extraction

In this section, I evaluate the effectiveness of different methods to incorporate knowledge into BERT. I use an 80%-20% train and test split and report the average result over multiple runs for each model. I calculate per class (9 class) and weighted F1 scores. Results of all models can be found in table 6.3.

Quality of text embedding: To investigate the quality of different types of sentence embedding techniques, I first compare the results from Word2Vec+biLSTM, Doc2Vec and BERT-base. Here sentence embedding from BERT-base (BERT) achieved significant improvement (3.9% and 29.9%) over sentence representations learned from Word2Vec and Doc2Vec (Word2Vec+biLSTM and Doc2Vec). Moreover, entity informed text representation from BERT-base (BERT-EE) achieved an impressive 24% performance boost over a model that use only the BERT sentence embedding (BERT).

Impact of domain text: I use ClinicalBERT to demonstrate the effect of domain text. Since entity-informed text representation achieves high performance, I continue to use that in all experiments. The result shows that ClinicalBERT-EE improves the performance by 0.97% over a general domain BERT-EE model. This demonstrates the utility of pre-training BERT using domain-specific text.

Impact of UMLS knowledge: To demonstrate the impact of additional UMLS knowledge, I combine knowledge graph embeddings with static text embeddings and ClinicalBERT text embeddings. While combined with word2Vec and Doc2Vec embeddings, UMLS knowledge results in a 10% and a 15% performance gain over Word2Vec-biLSTM and Doc2Vec. This performance gain indicates that UMLS knowledge is valuable for this task. As simple concatenation of KGE with static embedding works great, I continue to concatenate UMLS KGE with contextual embedding from ClinicalBERT. However result of adding UMLS knowledge with contextual embedding is not as impressive as adding it with static embedding. ClinicalBERT-EE-KGE (F1=0.9162) provides a 0.38% increase in performance over

ClinicalBERT-EE (F1=0.9127). To avoid underfitting on KGE embeddings, I train ClinicalBERT-EE-MLP. However, ClinicalBERT-EE-MLP (F1=0.9124) do not perform as well. I hypothesize that important information is lost when the 128-dimensional hidden layer vectors are used (versus the 700-dimensional knowledge graph embedding vectors). Next, I try to inject knowledge graph information directly into BERT input. Out of the three variations, ClinicalBERT-EE-RI-ST (with semantic type information and relation indicator in the second input) performed the best. This is overall best performing model, achieving an F1-score of 0.9181. The relation indicator predicted by KGE may play an important role in boosting performance. The results of ClinicalBERT-EE-ED-CT (F1=0.9137) and ClinicalBERT-EE-ED-ST (F1=0.9107) show that concept definitions and semantic type definitions did not help much. I hypothesize that the entity embeddings learned from BERT or the concept embeddings from KGE may be more precise than the embeddings learned from their text definitions. Next, I move on to pretrain BERT with knowledge graph information. Here I can see that ClinicalBERT-EE-KB is our second-best performing model with an F1-score of 0.9177. Finally, when I pre-train BERT with knowledge graphs using a Masked Language Model objective, I see that result (F1=0.9101) slightly goes down compared to ClinicalBERT-EE (F=0.9127). This indicates that incorporating knowledge into BERT using knowledge graph objective may be more efficient than training UMLS sentences with a language model objective.

6.5 Summary

In this research, I explore various ways to combine domain-specific knowledge into contextual word embedding. In particular, I investigate various techniques to incorporate the bio-medical knowledge base UMLS into BERT for clinical relation extraction. Results show that locating, extracting and adding entity embeddings from BERT is highly effective for relation extraction (24% improvement). Adding domain-specific information such as domain text (in ClinicalBERT) or UMLS domain knowledge (in ClinicalBERT-EE-RI-ST) only results in moderate performance gain (0.97% increase for adding domain text and an additional 0.59% increase for adding UMLS). The most effective method to fuse UMLS knowledge into BERT is BERT itself. The best performing model ClinicalBERT-EE-RI-ST transforms a corresponding triplet inferred from UMLS into a natural language sentence, which is added as the second sentence to BERT.

Chapter 7: Conclusion and Future Work

7.1 Conclusion

In this thesis, I investigate the effect of combining text embedding and additional knowledge for IE systems. I develop several novel IE systems for different applications. Experimental studies demonstrate the advantage of incorporating additional knowledge with text embedding and the superiority of proposed models over existing IE systems. The proposed *SupervisedOIE*, a supervised OpenIE model uses static word embedding and domain-independent knowledge, including parts of speech tag, syntactic role label, dependency tree information and results of existing Open IE systems as features. Validation using several benchmark datasets generated for the Open IE task shows that the proposed method is very effective in outperforming other supervised and unsupervised Open IE systems. To build an IE system for cyber-security, I combine domain-specific knowledge with static word embedding. I present a novel method WAE, to incorporate domain knowledge, including domain specification and human annotation, into static word embedding. Knowledge enriched embeddings are used to predict malware attribute labels from cybersecurity text. My investigation highlights the importance of incorporating diverse domain knowledge sources in domain-specific IE tasks. Next, I investigate the effectiveness of combining both domain-independent and domain-specific knowledge

with contextual embedding for relation extraction from clinical texts. While contextual embedding outperforms static embedding, it does not significantly improve the result after incorporating domain-independent knowledge. Experimental studies demonstrate that domain-independent knowledge works best while integrated with static embedding for clinical relation extraction task. I explore a wide range of techniques to incorporate the bio-medical knowledge base UMLS into contextual embedding model BERT for clinical relation extraction. Experimental studies illustrate that the most effective method to fuse domain knowledge into BERT is to transform a corresponding triplet inferred from UMLS into a natural language sentence and add it as the second sentence to BERT.

All the experimental studies in this thesis indicate that combining text embedding with additional knowledge is effective for IE systems. Both domain-independent and domain-specific knowledge combined with text embedding play a significant role in boosting the performance of IE systems. Both domain-independent and domain-specific knowledge are highly effective when combined with static embeddings. Static embedding encodes limited syntactic and domain-specific semantic information; thus, combining additional knowledge provides a significant boost. Overall it is more challenging to effectively combine additional knowledge with contextual embedding than static embedding. Both domain-independent and domain-specific knowledge have a positive but limited impact on performance when combined with contextual embeddings. Contextual embedding is comparatively better at encoding context-appropriate syntactic and semantic information. As a result, additional knowledge has minor effect on performance. Overall, knowledge injection is a better

technique to combine additional knowledge with BERT contextual embedding than knowledge concatenation.

7.2 Future Work

Currently, there is a limited understanding of the impact of adding new knowledge to word embedding. Especially for contextual embedding, it is unclear why additional knowledge provides only a minor improvement in IE tasks. In the future, I would like to focus on understanding the impact of knowledge integration on learned embeddings. I want to investigate how different types of knowledge and integration techniques may impact the embedding space. This issue can also be solved if we can interpret the meaning of word embeddings. Since the learned embeddings are in a high dimensional latent space that humans cannot intuitively understand, novel embedding visualization techniques may shed light on how different knowledge and injection methods may impact the embedding space.

Learning word embedding, primarily contextual embedding (e.g., BERT), requires massive data and computational power. Adding extra knowledge (e.g., an extensive knowledge base) to train a joint embedding only intensifies this problem. Due to resource limitations, I was able to explore only a specific type of knowledge incorporation method. In the future, I would like to explore other techniques like joint optimization of knowledge graph and text. I would also like to develop a unified framework to combine both domain-independent and domain-specific knowledge.

Recently, research shifted to contextual embedding because of its superiority; it is worth revisiting earlier works in this thesis with contextual embedding. It will

give a clear picture if we should always use contextual embedding instead of static embedding. Or if there are certain cases where static embedding is more beneficial when combined with additional knowledge.

I want to explore the ethical implication of additional knowledge. Specifically, I want to focus on how additional knowledge can be used to build fair IE systems. For example, human-generated text is biased, additional knowledge sources such as UMLS may have less bias.

Bibliography

[1] Horacio Saggion, Adam Funk, Diana Maynard, and Kalina Bontcheva. Ontology-based information extraction for business intelligence. In *The Semantic Web*, pages 843–856. Springer, 2007.

[2] Hassan H Malik, Vikas S Bhardwaj, and Huascar Fiorletta. Accurate information extraction for quantitative financial events. In *Proceedings of the 20th ACM international conference on information and knowledge management*, pages 2497–2500. ACM, 2011.

[3] Lisa F Rau. Conceptual information extraction from financial news. In *[1988] Proceedings of the Twenty-First Annual Hawaii International Conference on System Sciences. Volume III: Decision Support and Knowledge Based Systems Track*, volume 3, pages 501–509. IEEE, 1988.

[4] Kai Barkschat. Semantic information extraction on domain specific data sheets. In *European Semantic Web Conference*, pages 864–873. Springer, 2014.

[5] Hans-Michael Müller, Eimear E Kenny, and Paul W Sternberg. Textpresso: an ontology-based information retrieval and extraction system for biological literature. *PLoS biology*, 2(11):e309, 2004.

[6] Hui Yang, Irena Spasic, John A Keane, and Goran Nenadic. A text mining approach to the prediction of disease status from clinical discharge summaries. *Journal of the American Medical Informatics Association*, 16(4):596–600, 2009.

[7] Guergana K Savova, Jin Fan, Zi Ye, Sean P Murphy, Jiaping Zheng, Christopher G Chute, and Iftikhar J Kullo. Discovering peripheral arterial disease cases from radiology notes using natural language processing. In *AMIA Annual Symposium Proceedings*, volume 2010, page 722. American Medical Informatics Association, 2010.

[8] Peter L Elkin, David A Froehling, Dietlind L Wahner-Roedler, Steven H Brown, and Kent R Bailey. Comparison of natural language processing biosurveillance methods for identifying influenza from encounter notes. *Annals of Internal Medicine*, 156(1_Part_1):11–18, 2012.

[9] Kun Yu, Gang Guan, and Ming Zhou. Resume information extraction with cascaded hybrid model. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 499–506, 2005.

[10] Mena Badieh Habib Morgan and Maurice Van Keulen. Information extraction for social media. In *Proceedings of the Third Workshop on Semantic Web and Information Extraction*, pages 9–16, 2014.

[11] Rudy Prabowo and Mike Thelwall. Sentiment analysis: A combined approach. *Journal of Informetrics*, 3(2):143–157, 2009.

[12] Marcel Kvassay, Emil Gatial, et al. Email analysis and information extraction for enterprise benefit. *Computing and informatics*, 30(1):57–87, 2011.

[13] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. Learning sentiment-specific word embedding for twitter sentiment classification. In *ACL*, 2014.

[14] Qian Liu, Heyan Huang, Guangquan Zhang, Yang Gao, Junyu Xuan, and Jie Lu. Semantic structure-based word embedding by incorporating concept convergence and word divergence. In *AAAI*, 2018.

[15] Jianpeng Cheng, Zhongyuan Wang, Ji-Rong Wen, Jun Yan, and Zheng Chen. Contextual text understanding in distributional semantic space. In *CIKM*, 2015.

[16] Wei Fang, Jianwen Zhang, Dilin Wang, Zheng Chen, and Ming Li. Entity disambiguation by knowledge and text jointly embedding. In *CoNLL*, 2016.

[17] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph and text jointly embedding. In *EMNLP*, 2014.

[18] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

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

[20] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5, 2017.

[21] Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in translation: Contextualized word vectors. *arXiv preprint arXiv:1708.00107*, 2017.

[22] Oren Melamud, Jacob Goldberger, and Ido Dagan. context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of the 20th SIGNLL conference on computational natural language learning*, pages 51–61, 2016.

[23] Matthew E Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. Semi-supervised sequence tagging with bidirectional language models. *arXiv preprint arXiv:1705.00108*, 2017.

[24] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *CoRR*, abs/1802.05365, 2018.

[25] Alec Radford, Jeffrey Wu, Dario Amodei, Daniela Amodei, Jack Clark, Miles Brundage, and Ilya Sutskever. Better language models and their implications. *OpenAI Blog https://openai. com/blog/better-language-models*, 2019.

[26] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

[27] Jenny Rose Finkel and Christopher D Manning. Nested named entity recognition. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 141–150, 2009.

[28] Aron Culotta and Jeffrey Sorensen. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 423–429, 2004.

[29] Truc-Vien T Nguyen, Alessandro Moschitti, and Giuseppe Riccardi. Convolution kernels on constituent, dependency and sequential structures for relation extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1378–1387, 2009.

[30] Dat PT Nguyen, Yutaka Matsuo, and Mitsuru Ishizuka. Exploiting syntactic and semantic information for relation extraction from wikipedia. In *IJCAI Workshop on Text-Mining & Link-Analysis (TextLink 2007)*. Citeseer, 2007.

[31] Robert Leaman, Chih-Hsuan Wei, and Zhiyong Lu. tmchem: a high performance approach for chemical named entity recognition and normalization. *Journal of cheminformatics*, 7(1):1–10, 2015.

[32] Minlie Huang, Xiaoyan Zhu, Shilin Ding, Hao Yu, and Ming Li. Onbires: Ontology-based biological relation extraction system. In *Proceedings of the*

4th Asia-Pacific Bioinformatics Conference, pages 327–336. World Scientific, 2006.

[33] Olivier Bodenreider. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270, 2004.

[34] Jacob Andreas and Dan Klein. How much do word embeddings encode about syntax? In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 822–827, 2014.

[35] Zhanming Jie, Aldrian Muis, and Wei Lu. Efficient dependency-guided named entity recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[36] Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

[37] Jakub Piskorski, Hristo Tanev, Martin Atkinson, Eric Van Der Goot, and Vanni Zavarella. Online news event extraction for global crisis surveillance. In *Transactions on computational collective intelligence V*, pages 182–212. Springer, 2011.

[38] Christopher C Yang, Xiaodong Shi, and Chih-Ping Wei. Tracing the event evolution of terror attacks from on-line news. In *International Conference on Intelligence and Security Informatics*, pages 343–354. Springer, 2006.

[39] Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2670–2676, 2007.

[40] Ralph Grishman and Beth Sundheim. Message understanding conference-6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, 1996.

[41] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[42] Candace L Sidner. Focusing for interpretation of pronouns. *Computational Linguistics*, 7(4):217–231, 1981.

[43] Chinatsu Aone and Scott William Bennett. Evaluating automated and manual acquisition of anaphora resolution strategies. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 122–129. Association for Computational Linguistics, 1995.

[44] Zhou GuoDong, Su Jian, Zhang Jie, and Zhang Min. Exploring various knowledge in relation extraction. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 427–434. Association for Computational Linguistics, 2005.

[45] Yusuke Shinyama and Satoshi Sekine. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 304–311. Association for Computational Linguistics, 2006.

[46] Sergey Brin. Extracting patterns and relations from the world wide web. In *International Workshop on The World Wide Web and Databases*, pages 172–183. Springer, 1998.

[47] Qi Li, Heng Ji, and Liang Huang. Joint event extraction via structured prediction with global features. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 73–82, 2013.

[48] David Ahn. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8, 2006.

[49] Douglas E Appelt, Jerry R Hobbs, John Bear, David Israel, and Mabry Tyson. Fastus: A finite-state processor for information extraction from real-world text. In *IJCAI*, volume 93, pages 1172–1178, 1993.

[50] Ellen Riloff et al. Automatically constructing a dictionary for information extraction tasks. In *AAAI*, volume 1, pages 2–1. Citeseer, 1993.

[51] Luciano Del Corro and Rainer Gemulla. Clausie: clause-based open information extraction. In *Proceedings of the 22nd international conference on World Wide Web*, pages 355–366, 2013.

[52] Martin Atzmueller, Peter Kluegl, and Frank Puppe. Rule-based information extraction for structured data acquisition using textmarker. In *LWA*, volume 8, pages 1–7, 2008.

[53] GuoDong Zhou and Jian Su. Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 473–480, 2002.

[54] Xavier Carreras, Lluis Marquez, and Lluís Padró. Named entity extraction using adaboost. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002.

[55] Robert Malouf. Markov models for language-independent named entity recognition. In *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002.

[56] James S Aitken. Learning information extraction rules: An inductive logic programming approach. In *ECAI*, pages 355–359, 2002.

[57] R Mooney. Relational learning of pattern-match rules for information extraction. In *Proceedings of the sixteenth national conference on artificial intelligence*, volume 334, 1999.

[58] Yejin Choi, Claire Cardie, Ellen Riloff, and Siddharth Patwardhan. Identifying sources of opinions with conditional random fields and extraction patterns. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 355–362. Association for Computational Linguistics, 2005.

[59] Mary Elaine Califf and Raymond J Mooney. Bottom-up relational learning of pattern matching rules for information extraction. *Journal of Machine Learning Research*, 4(Jun):177–210, 2003.

[60] Tianhao Wu and William M Pottenger. A semi-supervised active learning algorithm for information extraction from textual data. *Journal of the American Society for Information Science and Technology*, 56(3):258–271, 2005.

[61] Andrew Carlson, Justin Betteridge, Richard C Wang, Estevam R Hruschka Jr, and Tom M Mitchell. Coupled semi-supervised learning for information extraction. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 101–110. ACM, 2010.

[62] Truc-Vien T Nguyen and Alessandro Moschitti. End-to-end relation extraction using distant supervision from external semantic repositories. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 277–282. Association for Computational Linguistics, 2011.

[63] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*, 2016.

[64] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344, 2014.

[65] Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 167–176, 2015.

[66] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133, 2016.

[67] Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. Supervised open information extraction. In *Proceedings of The 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT)*, pages 885–895, 2018.

[68] ACL. *Relation extraction from clinical texts using domain invariant convolutional neural network*, Berlin, Germany, 2016.

[69] Matthew R Gormley, Mo Yu, and Mark Dredze. Improved relation extraction with feature-rich compositional embedding models. *arXiv preprint arXiv:1505.02419*, 2015.

[70] Chang Xu, Yalong Bai, Jiang Bian, Bin Gao, Gang Wang, Xiaoguang Liu, and Tie-Yan Liu. Rc-net: A general framework for incorporating knowledge into word representations. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1219–1228. ACM, 2014.

[71] Jiang Bian, Bin Gao, and Tie-Yan Liu. Knowledge-powered deep learning for word embedding. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 132–148. Springer, 2014.

[72] Massimiliano Mancini, Jose Camacho-Collados, Ignacio Iacobacci, and Roberto Navigli. Embedding words and senses together via joint knowledge-enhanced training. In *CoNLL*, 2017.

[73] Quan Liu, Hui Jiang, Si Wei, Zhen-Hua Ling, and Yu Hu. Learning semantic word embeddings based on ordinal knowledge constraints. In *ACL-IJCNLP*, 2015.

[74] Omer Levy and Yoav Goldberg. Dependency-based word embeddings. In *ACL (2)*, pages 302–308, 2014.

[75] Andrew Trask, Phil Michalak, and John Liu. sense2vec-a fast and accurate method for word sense disambiguation in neural word embeddings. *arXiv preprint arXiv:1511.06388*, 2015.

[76] Julien Tissier, Christophe Gravier, and Amaury Habrard. Dict2vec : Learning word embeddings using lexical dictionaries. In *EMNLP*, 2017.

[77] Chen Li, Jianxin Li, Yangqiu Song, and Ziwei Lin. Training and evaluating improved dependency-based word embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[78] Dima Suleiman and Arafat A Awajan. Using part of speech tagging for improving word2vec model. In *2019 2nd International Conference on new Trends in Computing Sciences (ICTCS)*, pages 1–7. IEEE, 2019.

[79] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. Retrofitting word vectors to semantic lexicons. In *ACL*, 2015.

[80] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240, 2020.

[81] Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. Publicly available clinical BERT embeddings. In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, Minneapolis, Minnesota, USA, June 2019. Association for Computational Linguistics.

[82] Alistair EW Johnson, Tom J Pollard, Lu Shen, H Lehman Li-wei, Mengling Feng, Mohammad Ghassemi, Benjamin Moody, Peter Szolovits, Leo Anthony Celi, and Roger G Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

[83] Yifan Peng, Shankai Yan, and Zhiyong Lu. Transfer learning in biomedical natural language processing: An evaluation of BERT and ELMo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, Florence, Italy, August 2019. Association for Computational Linguistics.

[84] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. Domain-specific language model pretraining for biomedical natural language processing. *arXiv preprint arXiv:2007.15779*, 2020.

[85] Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert: A pretrained language model for scientific text. *arXiv preprint arXiv:1903.10676*, 2019.

[86] Chanwoo Jeong, Sion Jang, Eunjeong Park, and Sungchul Choi. A context-aware citation recommendation model with bert and graph convolutional networks. *Scientometrics*, 124(3):1907–1922, 2020.

[87] Boran Hao, Henghui Zhu, and Ioannis Paschalidis. Enhancing clinical bert embedding using a biomedical knowledge base. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 657–661, 2020.

[88] Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. Knowledge enhanced contextual

word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China, November 2019. Association for Computational Linguistics.

[89] Maxwell A Weinzierl, Ramon Maldonado, and Sanda M Harabagiu. The impact of learning unified medical language system knowledge embeddings in relation extraction from biomedical texts. *Journal of the American Medical Informatics Association*, 27(10):1556–1567, 2020.

[90] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. K-bert: Enabling language representation with knowledge graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2901–2908, 2020.

[91] Arindam Mitra, Pratyay Banerjee, Kuntal Kumar Pal, Swaroop Mishra, and Chitta Baral. How additional knowledge can improve natural language commonsense question answering? *arXiv preprint arXiv:1909.08855*, 2019.

[92] Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1535–1545, 2011.

[93] Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 523–534, 2012.

[94] Luciano Del Corro and Rainer Gemulla. ClausIE: clause-based open information extraction. In *22nd International World Wide Web Conference (WWW)*, pages 355–366, 2013.

[95] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL)*, pages 344–354, 2015.

[96] Swarnadeep Saha et al. Open information extraction from conjunctive sentences. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2288–2299, 2018.

[97] Swarnadeep Saha, Harinder Pal, et al. Bootstrapping for numerical open ie. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 317–323, 2017.

[98] Harinder Pal et al. Demonyms and compound relational nouns in nominal open ie. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction*, pages 35–39, 2016.

[99] Janara Christensen, Stephen Soderland, Oren Etzioni, et al. An analysis of open information extraction based on semantic role labeling. In *Proceedings of the sixth international conference on Knowledge capture*, pages 113–120. ACM, 2011.

[100] Harinder Pal and Mausam. Demonyms and compound relational nouns in nominal open IE. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction, AKBC@NAACL-HLT 2016*, pages 35–39, 2016.

[101] Alan Akbik and Alexander Löser. Kraken: N-ary facts in open information extraction. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction, AKBC-WEKEX@NAACL-HLT 2012*, pages 52–56, 2012.

[102] Nikita Bhutani, H. V. Jagadish, and Dragomir R. Radev. Nested propositions in open information extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 55–64, 2016.

[103] Swarnadeep Saha, Harinder Pal, and Mausam. Bootstrapping for numerical open IE. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL), Short paper*, pages 317–323, 2017.

[104] Gabriel Stanovsky and Ido Dagan. Creating a large benchmark for open information extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2300–2305, 2016.

[105] Lei Cui, Furu Wei, and Ming Zhou. Neural open information extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 407–413, 2018.

[106] Mausam. Open information extraction systems and downstream applications. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4074–4077, 2016.

[107] Mingming Sun, Xu Li, Xin Wang, Miao Fan, Yue Feng, and Ping Li. Logician: A unified end-to-end neural approach for open-domain information extraction. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM)*, pages 556–564, 2018.

[108] Hao Peng, Chris Gates, Bhaskar Sarma, Ninghui Li, Yuan Qi, Rahul Potharaju, Cristina Nita-Rotaru, and Ian Molloy. Using probabilistic generative models for ranking risks of android apps. In *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, CCS '12, 2012.

[109] Rahul Pandita, Xusheng Xiao, Wei Yang, William Enck, and Tao Xie. Whyper: Towards automating risk assessment of mobile applications. In *USENIX Security Symposium*, volume 2013, 2013.

[110] Zhengyang Qu, Vaibhav Rastogi, Xinyi Zhang, Yan Chen, Tiantian Zhu, and Zhong Chen. Autocog: Measuring the description-to-permission fidelity in android applications. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1354–1365. ACM, 2014.

[111] Rocky Slavin, Xiaoyin Wang, Mitra Bokaei Hosseini, James Hester, Ram Krishnan, Jaspreet Bhatia, Travis D Breaux, and Jianwei Niu. Toward a framework for detecting privacy policy violations in android application code. In *Proceedings of the 38th International Conference on Software Engineering*, pages 25–36. ACM, 2016.

[112] Ziyun Zhu and Tudor Dumitras. Featuresmith: Automatically engineering features for malware detection by mining the security literature. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 767–778. ACM, 2016.

[113] Corinne L Jones, Robert A Bridges, Kelly MT Huffer, and John R Goodall. Towards a relation extraction framework for cyber-security concepts. In *Proceedings of the 10th Annual Cyber and Information Security Research Conference*, page 11. ACM, 2015.

[114] Arnav Joshi, Ravendar Lal, Tim Finin, and Anupam Joshi. Extracting cybersecurity related linked data from text. In *2013 IEEE Seventh International Conference on Semantic Computing*, pages 252–259. IEEE, 2013.

[115] Robert A Bridges, Corinne L Jones, Michael D Iannacone, Kelly M Testa, and John R Goodall. Automatic labeling for entity extraction in cyber security. *arXiv preprint arXiv:1308.4941*, 2013.

[116] Nikki McNeil, Robert A Bridges, Michael D Iannacone, Bogdan Czejdo, Nicolas Perez, and John R Goodall. Pace: Pattern accurate computationally efficient bootstrapping for timely discovery of cyber-security concepts. In *2013 12th International Conference on Machine Learning and Applications*, volume 2, pages 60–65. IEEE, 2013.

[117] Houssem Gasmi, Jannik Laval, and Abdelaziz Bouras. Information extraction of cybersecurity concepts: An lstm approach. *Applied Sciences*, 9(19):3945, 2019.

[118] Swee Kiat Lim, Aldrian Obaja Muis, Wei Lu, and Chen Hui Ong. Malwaretextdb: A database for annotated malware articles. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1557–1567, 2017.

[119] Kiran Blanda and David Westcott. APTnotes, 2018.

[120] Özlem Uzuner, Jonathan Mailoa, Russell Ryan, and Tawanda Sibanda. Semantic relations for problem-oriented medical records. *Artificial intelligence in medicine*, 50(2):63–73, 2010.

[121] Kirk Roberts, Bryan Rink, and Sanda Harabagiu. Extraction of medical concepts, assertions, and relations from discharge summaries for the fourth i2b2/va shared task. In *Proceedings of the 2010 i2b2/VA Workshop on Challenges in Natural Language Processing for Clinical Data. Boston, MA, USA: i2b2*, 2010.

[122] G Divita, OZ Treitler, YJ Kim, et al. Salt lake city va's challenge submissions. In *proceedings of the 2010 i2b2/VA Workshop on Challenges in Natural Language Processing for Clinical Data*, 2010.

[123] Illés Solt, Ferenc P Szidarovszky, and Domonkos Tikk. Concept, assertion and relation extraction at the 2010 i2b2 relation extraction challenge using parsing information and dictionaries. *Proc. of i2b2/VA Shared-Task. Washington, DC*, 2010.

[124] JD Patrick, DHM Nguyen, Y Wang, and M Li. I2b2 challenges in clinical natural language processing 2010. In *Proceedings of the 2010 i2b2/VA Workshop on Challenges in Natural Language Processing for Clinical Data. Boston, MA, USA: i2b2*, 2010.

[125] Siddhartha Jonnalagadda and Graciela Gonzalez. Can distributional statistics aid clinical concept extraction. In *Proceedings of the 2010 i2b2/VA workshop on challenges in natural language processing for clinical data. Boston, MA, USA: i2b2*, 2010.

[126] Peter Anick, Pengyu Hong, N Xue, and David Anick. I2b2 2010 challenge: machine learning for information extraction from patient records. In *Proceedings of the 2010 i2b2/VA Workshop on Challenges in Natural Language Processing for Clinical Data. Boston, MA, USA: i2b2*, 2010.

[127] Aaron M Cohen, Kyle Ambert, Jianji Yang, Robert Felder, Richard Sproat, Brian Roark, Kristy Hollingshead, and Kari Baker. Ohsu/portland vamc team participation in the 2010 i2b2/va challenge tasks. In *Proceedings of the 2010 i2b2/VA Workshop on Challenges in Natural Language Processing for Clinical Data*. i2b2, Boston, MA, USA, 2010.

[128] Dina Demner-Fushman, Emilia Apostolova, R Islamaj Dogan, François-Michel Lang, JG Mork, A Neveol, SE Shooshan, M Simpson, and AR Aronson. Nlm's system description for the fourth i2b2/va challenge. In *Proceedings of the 2010 i2b2/VA Workshop on Challenges in Natural Language Processing for Clinical Data. Boston, MA, USA: i2b2*, 2010.

[129] Cyril Grouin, Asma Ben Abacha, Delphine Bernhard, Bruno Cartoni, Louise Deleger, Brigitte Grau, Anne-Laure Ligozat, Anne-Lyse Minard, Sophie Rosset, and Pierre Zweigenbaum. Caramba: concept, assertion, and relation annotation using machine-learning based approaches. In *i2b2 Medication Extraction Challenge Workshop*, 2010.

[130] Berry de Bruijn, Colin Cherry, Svetlana Kiritchenko, Joel Martin, and Xiaodan Zhu. Nrc at i2b2: one challenge, three practical tasks, nine statistical systems, hundreds of clinical records, millions of useful features. In *Proceedings of the 2010 i2b2/VA Workshop on Challenges in Natural Language Processing for Clinical Data. Boston, MA, USA: i2b2*, 2010.

[131] Russell J Ryan. *Groundtruth budgeting: a novel approach to semi-supervised relation extraction in medical language*. PhD thesis, Massachusetts Institute of Technology, 2011.

[132] Anne-Lyse Minard, Anne-Laure Ligozat, and Brigitte Grau. Multi-class SVM for relation extraction from clinical reports. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, Hissar, Bulgaria, September 2011. Association for Computational Linguistics.

[133] Sunil Sahu, Ashish Anand, Krishnadev Oruganty, and Mahanandeeshwar Gattu. Relation extraction from clinical texts using domain invariant convolutional neural network. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, Berlin, Germany, August 2016.

[134] Yuan Luo, Yu Cheng, Özlem Uzuner, Peter Szolovits, and Justin Starren. Segment convolutional neural networks (seg-cnns) for classifying relations in clinical notes. *JAMIA*, 25(1):93–98, 2018.

[135] Zhiheng Li, Zhihao Yang, Chen Shen, Jun Xu, Yaoyun Zhang, and Hua Xu. Integrating shortest dependency path and sentence sequence into a deep learning framework for relation extraction in clinical text. *BMC medical informatics and decision making*, 19(1):22, 2019.

[136] Dhanachandra Ningthoujam, Shweta Yadav, Pushpak Bhattacharyya, and Asif Ekbal. Relation extraction between the clinical entities based on the shortest dependency path based lstm. *arXiv preprint arXiv:1903.09941*, 2019.

[137] Yifan Peng, Shankai Yan, and Zhiyong Lu. Transfer learning in biomedical natural language processing: An evaluation of bert and elmo on ten benchmarking datasets. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 58–65, 2019.

[138] Qiang Wei, Zongcheng Ji, Yuqi Si, Jingcheng Du, Jingqi Wang, Firat Tiryaki, Stephen Wu, Cui Tao, Kirk Roberts, and Hua Xu. Relation extraction from clinical narratives using pre-trained language models. In *AMIA Annual Symposium Proceedings*, volume 2019, page 1236. American Medical Informatics Association, 2019.

[139] Zhanming Jie and Wei Lu. Dependency-guided lstm-crf for named entity recognition. *arXiv preprint arXiv:1909.10148*, 2019.

[140] Katrin Fundel, Robert Küffner, and Ralf Zimmer. Relex—relation extraction using dependency parse trees. *Bioinformatics*, 23(3):365–371, 2007.

[141] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[142] Dian Yu, Lifu Huang, and Heng Ji. Open relation extraction and grounding. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing, IJCNLP 2017, Taipei, Taiwan, November 27 - December 1, 2017 - Volume 1: Long Papers*, pages 854–864, 2017.

[143] Amina Kadry and Laura Dietz. Open relation extraction for support passage retrieval: Merit and open issues. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, August 7-11, 2017*, pages 1149–1152, 2017.

[144] Benjamin Roth, Costanza Conforti, Nina Pörner, Sanjeev Karn, and Hinrich Schütze. Neural architectures for open-type relation argument extraction. *CoRR*, abs/1803.01707, 2018.

[145] Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing*, 45(11):2673–2681, 1997.

[146] Filipe de Sá Mesquita, Jordan Schmidek, and Denilson Barbosa. Effectiveness and efficiency of open relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 447–457, 2013.

[147] Ying Xu, Mi-Young Kim, Kevin Quinn, Randy Goebel, and Denilson Barbosa. Open information extraction with tree kernels. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 868–877, 2013.

[148] Luheng He, Mike Lewis, and Luke Zettlemoyer. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 643–653, 2015.

[149] Julian Michael, Gabriel Stanovsky, Luheng He, Ido Dagan, and Luke Zettlemoyer. Crowdsourcing question-answer meaning representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Short Paper*, pages 560–568, 2018.

[150] François Chollet et al. Keras. https://keras.io, 2015.

[151] Rudolf Schneider, Tom Oberhauser, Tobias Klatt, Felix A. Gers, and Alexander Löser. Analysing errors of open information extraction systems. *CoRR*, 2017.

[152] Mausam Mausam. Open information extraction systems and downstream applications. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 4074–4077. AAAI Press, 2016.

[153] Ivan Kirillov, Desiree Beck, Penny Chase, and Robert Martin. Malware attribute enumeration and characterization, 2011.

[154] Arpita Roy, Youngja Park, Taesung Lee, and Shimei Pan. Supervising unsupervised open information extraction models. In *Proceedings of the 2019 Conference on EMNLP-IJCNLP*, Hong Kong, China, November 2019.

[155] Yatian Shen and Xuanjing Huang. Attention-based convolutional neural network for semantic relation extraction. In *Proceedings of COLING 2016*, Osaka, Japan, December 2016.

[156] Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. *JAMIA*, pages 552–556, 2011.

[157] Emily Alsentzer, John Murphy, William Boag, Wei-Hung Weng, Di Jindi, Tristan Naumann, and Matthew McDermott. Publicly available clinical bert embeddings. *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78, 2019.

[158] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196, 2014.

[159] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the ACL: system demonstrations*, pages 55–60, 2014.

[160] Yuhao Zhang, Peng Qi, and Christopher D Manning. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on EMNLP*, pages 2205–2215, 2018.

[161] Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[162] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

[164] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv preprint arXiv:1906.08237*, 2019.

[165] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[166] Peng Shi and Jimmy Lin. Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*, 2019.

[167] Chen Jia, Yuefeng Shi, Qinrong Yang, and Yue Zhang. Entity enhanced bert pre-training for chinese ner. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6384–6396, 2020.

[168] Zhengjie Gao, Ao Feng, Xinyu Song, and Xi Wu. Target-dependent sentiment classification with bert. *IEEE Access*, 7:154290–154299, 2019.

[169] Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, USA*, pages 8449–8456. AAAI Press, 2020.

[170] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194, 2021.

[171] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy, July 2019. Association for Computational Linguistics.

[172] Shanchan Wu and Yifan He. Enriching pre-trained language model with entity information for relation classification. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 2361–2364, 2019.

[173] Guergana K Savova, James J Masanz, Philip V Ogren, Jiaping Zheng, Sunghwan Sohn, Karin C Kipper-Schuler, and Christopher G Chute. Mayo clinical text analysis and knowledge extraction system (ctakes): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 17(5):507–513, 2010.

[174] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems (NIPS)*, pages 1–9, 2013.

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

[176] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*, pages 2071–2080. PMLR, 2016.

[177] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[178] Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. A novel embedding model for knowledge base completion based on convolutional neural network. *arXiv preprint arXiv:1712.02121*, 2017.

[179] Bin He, Di Zhou, Jinghui Xiao, Xin Jiang, Qun Liu, Nicholas Jing Yuan, and Tong Xu. BERT-MK: Integrating graph contextualized knowledge into pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2281–2290, Online, November 2020. Association for Computational Linguistics.