ENHANCING MODEL GENERALIZATION FOR RELATION EXTRACTION IN BIOMEDICAL DOMAIN

by

Peng Su

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Computer Science

Fall 2021

ENHANCING MODEL GENERALIZATION FOR RELATION EXTRACTION IN BIOMEDICAL DOMAIN

by

Peng Su

Approved: _____
Kathleen F. McCoy, Ph.D
Chair of the Department of Computer and Information Sciences

Approved: _____
Levi T. Thompson, Ph.D
Dean of the College of Engineering

Approved: _____
Louis F. Rossi, Ph.D.
Vice Provost for Graduate and Professional Education and
Dean of the Graduate College

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Vijay K. Shanker, Ph.D
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Li Liao, Ph.D
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Cathy H. Wu, Ph.D
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Zhiyong Lu, Ph.D
Member of dissertation committee

ACKNOWLEDGEMENTS

First of all, I would like to express my sincere gratitude to my advisor Prof. Vijay K. Shanker for the continuous support of my Ph.D research, also for his patience, enthusiasm, and immense knowledge. His guidance helped me constantly during the period of my Ph.D research and writing of this dissertation. I learned so many things for him and I could not image having better advisor and mentor for my Ph.D study.

Also, I would like to thank the rest of my committee members: Prof. Li Liao, Prof. Cathy H. Wu and Dr. Zhiyong Lu, for their insightful feedback and great suggestions.

Special thanks go to my labmates in the Biomedical Text Mining lab: Gang Li, Jia Ren, Samir Gupta, Ashique Mahmood, Debarati Roy Chowdhury, Mehmet Efruz Karabulut. I will miss the days I spent with you in the lab.

I would like to thank Yifan Peng for his help during our collaboration and thank Ziqing Luo for his suggestions during the preparation of my last journal paper.

Finally, I would like to thank my family for their unconditionally support, and a special thanks to my wife Yan Dong for her accompany during my Ph.D pursuit.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Significant progress has been made in applying deep learning on natural language processing tasks recently. However, deep learning models typically require a large amount of annotated training data and easily overfit when the training datasets are small, which usually leads to poor generalization. In this work, we consider four different methods to help deep learning models generalize better on relation extraction tasks in biomedical domain. Each of our methods improve on the state-of-the-art BERT-based model on benchmark sets in the biomedical literature.

First, we will investigate the method of enhancing the generalization of transformer-based BERT model on the relation extraction tasks. While there have been several adaptations of the BERT model to the biomedical domain, those models generalize differently on the downstream tasks. Motivated by this observation, we investigate the impact of additional domain adaptation by adding another level of adaptation on sub-domain data to bridge the gap between domain knowledge and task-specific knowledge. We show that further adaptation on task-specific sub-domains improves the results of leading system on different benchmark sets.

Second, we will refine the fine-tuning process of BERT-based models. We show that the traditional architecture used in relation extraction fails to utilize all the knowledge embedded in the BERT model. After using summarized information from all the outputs in the final layer, we can improve the performance of relation extraction models.

Third, we will try to improve model generalization by augmenting the manually annotated data using distant supervision. Distant supervision provides an inexpensive way to obtain annotated data. Using knowledge bases related to the relation extraction tasks, we can create large amounts of annotated data with no human effort. After

investigating multiple methods to reduce noise in the automatically created training sets, we find that simple combination of human-labeled and the automatically generated data does not necessarily result in improved performance. However, our experiments show that by applying transfer learning technique, we can obtain significant gains over models trained on just the human labeled sets.

Finally, we will investigate contrastive learning for improving the text representation from the BERT model for the relation extraction tasks. Contrastive learning can yield a better representation by comparing the similarity and dissimilarity of real data and augmented data. In our framework, we utilize a unique contrastive pre-training step tailored for the relation extraction tasks by seamlessly integrating linguistic knowledge into the data augmentation. Also, we investigate how large-scale data constructed from the external knowledge bases can enhance the generality of contrastive pre-training of BERT. The experiment results on three relation extraction benchmark datasets demonstrate that our method can improve the BERT model representation. In addition, we explore the interpretability of models by showing that BERT with contrastive pre-training relies more on rationales for prediction.

Chapter 1

INTRODUCTION

In the past few decades, biomedical literature has been growing explosively and biomedical text mining has become a unique part of modern biomedical research. Finding and extracting relevant information from the ever-increasing amounts of biomedical text is becoming a much-needed task. The rapid growth makes it difficult for researchers to catch up with the new findings in the biomedical literature. Also, biomedical literature contains essential information for clinicians to make important clinical decisions, for database curators to build up-to-date database, etc. In order to facilitate the biomedical research, many search tools and literature archives have been developed to make researchers easily access the biomedical text. Currently, this field has attracted considerable attentions due to the valuable knowledge hidden in the biomedical text.

Among all the biomedical text mining tasks, relation extraction (RE) plays a key role in information extraction and aids database curation for many disciplines. The task of relation extraction is to extract semantic relationships between two or more entities of a certain type from text. Relation extraction in biomedical domain is concerned about the relationships between biomedical entities such as gene, protein, and disease, etc. After the extraction of different relations, we can convert the unstructured text to structured information, which is a crucial step for natural language understanding applications like automated reasoning [77], machine translation [48], question answering [27], etc.

Developing high-performing systems to automatically extract relations from text is critical due to the fact that manual curation lags behind the growth in biomedical research literature. In recent years, applying deep learning models and natural language processing techniques has become a common way to solve problems in biomedical

domain. The deep learning models have better performance than most of the previous methods when we have enough training data. Especially, the recent trend of applying language models dominates the applications of relation extraction [15, 61, 17, 66].

In order to address relation extraction problem using deep learning, we usually see it as a classification problem, where an instance comprising of text (typically a sentence) and entities mentioned in the text is annotated as a certain relation type depending on whether the text expresses relations of interest among the marked entities.

However, each new relation extraction task requires its own annotated data for training the deep learning model. Currently, only small datasets are available for most tasks and this situation can hinder us from achieving good performance of deep learning models. Naturally, deep learning models need large dataset to train since the models usually have a large number of parameters to adjust during training. The language model method alleviates this problem to some extent with its representation learning on unlabeled data, however the applications on specific tasks still need large datasets to fine-tune the model parameters for good generalization. In the meantime, it is not practical to manually label a large dataset because the annotation process of data needs considerable human efforts to put a label on each data instance and often requires domain expertise, especially in specialized fields like Biomedicine. Thus, in order to achieve the full potential of deep learning models, developing automatic tools to label new training data or make the best use of available data is an important problem to solve.

In this dissertation, we will consider designing a few different methods to help the generalization of deep learning models using the available (sometimes limited) training data. First, we will propose methods to improve a language model (BERT [17]) during its pre-training and fine-tuning phase for our relation extraction tasks. Next, we will try to generate large amount of labeled training data automatically by utilizing distant supervision, which is a technique of labeling sentences under the guidance of an existing database that stores the related entity tuples. Then we utilize the DS-generated data (along with the original training data) to build better-generalized deep learning models.

Also, we will utilize contrastive learning to improve the representation from BERT model by involving the use of augmented data for our relation extraction tasks.

1.1 Improving Pre-training of BERT Model

Recently, the emergence of language models dominates the relation extraction field with their superior performance [15, 61, 17, 66]. These language model methods can make best use of a large amount of unlabeled data in an unsupervised pre-training phase to build a general language representation. This phase can be followed by supervised fine-tuning to learn the knowledge from task-specific datasets. Among all the language models, BERT [17]–a language representation model based on bidirectional transformer [78], attracted a lot of attention from researchers in different fields.

BERT is designed to learn a context-dependent and universal language representation using the transformer [78]. Originally, BERT was proposed for general domain, and in order to make the model generalize better in biomedical domain, several BERT models have been adapted for the biomedical domain by pre-training BERT using biomedical text.

Our goal to improve the pre-training of BERT model is motivated by the results of the experiments we conducted to investigate differing performances of several BERT-based models for the biomedical domain. Different models such as BioBERT [43], SciBERT [3] and BlueBERT [59] are developed to adapt to the biomedical domain. One of the primary differences between them is the corpora used in the pre-training for domain adaptation. Those pre-trained BERT models are suppose to have similar performance on similar applications since they were pre-trained on similar biomedical data. However, our experiments reveal that those models have significantly different performance on the same set of relation extraction tasks (Table 3.2 in Chapter 3). Therefore, we hypothesize that the text used in the pre-training for domain adaptation can have a significant impact on the downstream applications.

In order to leverage the pre-training for a specific task, we introduce another level of adaptation to adjust the domain adaptation to specific sub-domains in this work. We

call this part sub-domain adaptation. To fulfill this task, we add one more pre-training step on sub-domain data. For example, for a relation extraction task like the extraction of drug-drug interactions (DDI), we investigate whether adding more drug-specific text can help over general biomedical domain knowledge. After the sub-domain adaptation, we expect that the pre-trained BERT model will generalize better on the specific tasks.

1.2 Fine-tuning BERT Model with New Architecture

After the pre-training of BERT model, a general language representation is learned. In order to apply this task-agnostic representation on specific tasks, we need to fine-tune the BERT model via supervised learning on task-specific datasets.

Our investigation of improving fine-tuning mechanism is partially driven by the insights in [75], in which the authors find that the BERT model learns the representation similar to traditional natural language processing (NLP) pipelines. Based on the findings in [75], the basic syntactic aspects of the text appear to be learned in the lower layers, while high-level semantic information appears in higher layers of BERT model. Since relation extraction tasks are concerned with the semantic relations between entities, we wish to investigate whether the upper (including top) layers contain important information about RE tasks.

In addition, our motivation to improve the fine-tuning of BERT is also from the following observations: all previous works utilizing BERT-based models for classification tasks (including relation extraction) employ a standard way of fine-tuning using the classification token ([CLS]) alone among the last layer. Thereby all information contained in other final layer nodes is completely ignored during the fine-tuning process. However, the ignored information in the last layer of BERT model is utilized for other tasks like sequence tagging. From this point of view, the BERT model is fine-tuned with one less layer for classification tasks. This drives us to investigate the approach of utilizing all the information in the last layer for the classification tasks.

For this investigation, we first employ the edge probing technique [76] to measure how much relevant information (about relation extraction tasks) the last layer contains.

The results illustrate that the last layer contains useful information that is unused in the original fine-tuning method of only using classification token. To include the unused information in the last layer during fine-tuning, we explore two different methods: recurrent neural network (RNN) and attention mechanism [2]. The summarized knowledge will be concatenated with the classification token as the model output in a refined fine-tuning process. We call it fine-tuning with information summarization in the last layer (SLL fine-tuning).

1.3 Utilizing Weakly Labeled Data from Distant Supervision

Distant supervision (DS) is a technique to generate labeled data for supervised machine learning methods in the natural language processing (NLP) field. In the case of relation extraction, distant supervision uses an existing knowledge base containing instances in the learning task's relation of interest and then DS assumes that a piece of text (often a sentence) describes a relation between entities if these entities are related according to the database [53].

In this way, we could use public databases which store known related tuples to label the text in literature in biomedical field. Thus, we can automatically obtain large training datasets that could be used to train deep learning models. However, a well-known problem for distant supervision is that there might be noise in the DS-generated since the labeling process pays no attention to the content of the text. In order to reduce the noise in the DS data, we consider applying some heuristics to help clean the data in this work. With noise-reduced DS data used in the training of deep learning model, we can build better models. Normally, DS data has been used by itself and not in combination with manually annotated data. In order to leverage the knowledge in the DS data, we will design methods to train models that use both types of data: the DS-obtained data and manually annotated (MA) data. A simple and straightforward way is just to take the union of these two datasets and train the deep learning model on the united dataset. We also consider an alternate way – transfer learning, which is to pre-train the model on the DS-generated data and then further tune the model on

the human-labeled data. Both methods for combining DS-generated data and MA data will be evaluated on biomedical relation extraction tasks.

1.4 Improving Representation of RE with Contrastive Learning

Contrastive learning is a family of methods to learn a discriminative model by comparing input pairs [40]. The comparison is performed between positive pairs of "similar" inputs and negative pairs of "dissimilar" inputs. The positive pairs can be generated in an automatic way by transforming the original data to variants without changing the key information. Contrastive learning can encode general properties (e.g. invariance) in the learned representation while it is relatively hard for other representation learning methods to achieve [4, 40]. Therefore, contrastive learning provides a powerful approach to learn representations in a self-supervised manner and has shown great promise and achieved the state of the art results in recent years [25, 12].

Despite its advancement, contrastive learning has not been well studied in biomedical natural language processing (BioNLP), especially for relation extraction tasks. One obstacle lies in the discrete characteristics of text data. Compared to computer vision, it is more challenging to design a general and efficient data augmentation method to construct positive pairs. Instead, there have been significant advances in the development of pre-trained language models to facilitate downstream BioNLP tasks [17, 66, 59]. Therefore, leveraging contrastive learning in the large pre-trained language models to learn more general representation for relation extraction tasks remains unexplored.

To bridge this gap, we presents an innovative method of contrastive pre-training to improve the language model representation for biomedical relation extraction. As the main difference from the existing contrastive learning framework, we augment the datasets for relation extraction tasks by randomly changing the words that do not affect the relation expression. Here, we hypothesize that the shortest dependency path (SDP) between two entities [8] captures the required knowledge for the relation expression. We hence keep words on SDP fixed during the data augmentation. In addition, we utilize

external knowledge bases to construct more data to make the learned representation generalize better, which is a method that is frequently used in distant supervision [53].

1.5 Outline of This Dissertation

The rest of this dissertation is organized as follows. In Chapter 2, we will discuss the tasks, the evaluation sets and related work on models used for relation extraction. Then the methods of improving the pre-training and fine-tuning of BERT model are discussed in Chapter 3 and Chapter 4, respectively. Next, deep learning relation extraction method utilizing distantly supervised data will be explored in Chapter 5. In Chapter 6, we explore the technique of contrastive learning for improving the text representation of BERT model. We conclude in Chapter 7.

Chapter 2

PRELIMINARIES

In this chapter, we will provide the background information for the following chapters. Specifically, we first introduce the relation extraction tasks and the evaluation sets. Then, we will discuss the language model method for relation extraction, especially the BERT model, which will be presented in detail. Also, the architecture of convolutional neural network and recurrent neural network will be introduced in this chapter as we conduct some preliminary experiments using these two deep learning models.

2.1 Relation Extraction Tasks and Evaluation Datasets

There has been considerable efforts invested in the extraction of different relations in BioNLP. Typically, relation extraction in BioNLP can be seen as a classification problem: where an instance (a sentence and entities mentioned in the sentence) is annotated with a type of relation expression based on whether that sentence expresses a relation of interest among the labeled entities. Currently, language model methods dominate this field with their excellent performance. However, language model methods generalize differently on different tasks, which is (partially) caused by the situation of not having enough training data for the tasks. In this work, we will design approaches to make the model generalize better on a set of relation extraction tasks in BioNLP. Specifically, the relation extraction tasks that we will consider include: the protein-protein interaction (PPI) [37], the protein subcellular localization (PLOC) [35], the miRNA-gene regulation relation (MIRGENE) [45], the drug-drug interaction (DDI) [26], and the chemical-protein interaction (ChemProt) [38].

2.1.1 Human-annotated Sets

For PPI task, we will utilize the most well-known PPI relation corpus: AIMed [7]. AIMed is obtained by annotating 750 abstracts from Medline and it contains 1,000 positive (interacting protein pairs) and 4,834 negative (non-interacting protein pairs) instances. LocText corpus [11] will be our evaluation set for PLOC task, which is a well-known dataset annotated with tagtog tool [10]. As for miRNA-gene regulation task, we will use the dataset (call it MIRGENE) from [45]. Since there is no standard training and test set for these datasets, we will employ 10-fold cross-validation on them.

The corpus for ChemProt task contains a training set of 1,020 abstracts, a development set of 612 abstracts and a test set of 800 abstracts from PubMed [38]. The ChemProt corpus is labeled with six classes: CPR:3 (upregulator, activator, indirect_upregulator), CPR:4 (downregulator, inhibitor, indirect_downregulator), CPR:5 (agonist, agonist-activator, agonist-inhibitor), CPR:6 (antagonist), CPR:9 (substrate, product_of, substrate_product_of) and negative. For the DDI corpus, it has 792 documents from DrugBank database and 233 abstracts from Medline [26]. In DDI corpus, there are five labels: ADVICE, EFFECT, INT, MECHANISM and negative. For ChemProt and DDI task, we will use the the same spilt of training, development, test set with the PubMedBERT model [22] during the model evaluation. Please see Table 2.1 for the statistics of these corpora.

Corpus	Instance# (Total)	Train	Dev	Test
AIMed	5,834	-	-	-
DDI	33,508	22,233	5,559	5,716
ChemProt	45,048	18,035	11,268	15,745
MIRGENE	1,239	-	-	-
LocText	689	-	-	-

Table 2.1: Statistics of human-labeled datasets for our relation extraction tasks. For the corpora AIMed, LocText and MIRGENE, there is no standard split of training, development, and test, so we will utilize 10-fold cross-validation on them during evaluation.

2.2 Language Model Method for Relation Extraction

Recently, utilizing neural pre-trained language models has been shown to be an effective way to improve model performance of natural language processing (NLP) tasks including relation extraction [15, 61, 17, 66]. An advantage of language model methods is that they can leverage a large amount of unlabeled data to learn the context-dependent and universal language representation. The pre-trained language model can be applied on downstream applications through fine-tuning on the labeled data for specific tasks. Many language models have been proposed such as ELMo [61], GPT [65, 66, 6], BERT [17], Transformer-XL [16], XLNet [87], RoBERTa [47], etc. In this dissertation, we will employ the BERT model [17] to solve the relation extraction problem.

BERT [17] is a bidirectional Transformer model [78] for language representation. Using a "masked language model", BERT is able to learn bidirectional representation (both left and right context) of a word. Also, the sentence-level context can be learned through the "next sentence prediction". The application of BERT usually includes two steps: utilizing pre-training to gain general knowledge of a domain and employing fine-tuning to acquire task-specific knowledge of a task. These two steps make language model methods generalize well on various tasks by combining the learned universal representation from unlabeled data and task-specific knowledge from labeled data.

2.2.1 Pre-training of BERT

In the pre-training stage, BERT learns general language representation for both the words and the text sequence through two unsupervised tasks: masked language model (MLM) and next sentence prediction (NSP). The MLM technique randomly replaces a portion of tokens with a special token (e.g., [MASK]), and lets the language model predict them. In NSP, the model is trained to predict whether one sentence follows the other in the original text given a sentence pair. The pre-training procedure usually involves a large amount of unlabeled data. Originally, BERT is designed for general purpose and it is pretrained on English Wikipedia and BooksCorpus.

However, each domain has its specific knowledge and sometime it is impossible for the general BERT to gain such knowledge. For example, in biomedical domain, the biomedical entity names can not be represented well based on the general BERT model since its pre-training corpora are short of such names. Thus there have been efforts to adapt BERT model to biomedical domain before applying it to biomedical tasks (a.k.a., pre-training BERT model on biomedical data). Several BERT models have been adapted for the biomedical domain by pre-training the model on biomedical text such as BioBERT [43], SciBERT [3], BlueBERT [59], and PubMedBERT [22]. BioBERT [43] is pre-trained on PubMed abstracts and PMC full-length articles; BlueBERT [59] uses PubMed abstracts and MIMIC-III clinical notes [33] as the pre-training data; A sample of 1.14M papers from Semantic Scholar [1] are used in SciBERT model [3]; PubMedBERT [22] is pre-trained from scratch using PubMed abstracts.

There are two ways of adapting BERT model for biomedical domain: (1) additional pre-training using biomedical data from pre-trained BERT for general domain (e.g., BioBERT [43], SciBERT [3], BlueBERT [59]); (2) pre-training BERT on biomedical data from scratch (e.g., PubMedBERT [22]). Obviously, the difference between these two types of adapted models is whether to use the general domain data before the biomedical domain adaptation. Other than the initial pre-training selection, the first type of BERT model (BioBERT, SciBERT and BlueBERT) follows the convention of original BERT to select subwords to mask in the masked language model (MLM) task. The PubMedBERT (the second type of BERT model) utilizes whole-word masking (WWM) instead, which enforces the whole word must be masked in MLM if one or more of its subwords is chosen. In this dissertation, we will experiment with those two types of adapted BERT models for the biomedical domain.

2.2.2 Fine-tuning of BERT

The task-agnostic representation (from pre-training) can be fine-tuned for various downstream tasks via supervised training on labeled datasets. Given a task, we just need to append an additional output layer on top of the BERT output and then

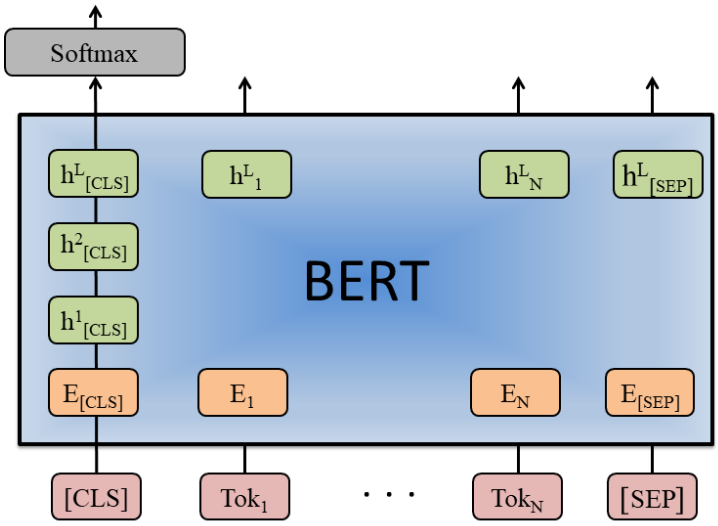


Figure 2.1: BERT model architecture on relation extraction.

feed the task-specific inputs and outputs into BERT model and fine-tune the BERT parameters like regular model training. For classification problems, a classifier can be built by putting a Softmax layer on the [CLS] token output of pre-trained BERT model. Typically, the [CLS] token is designed for classification purpose in BERT model and every input sequence will always start with this special token. The model architecture of fine-tuning based on [CLS] token is shown in Fig 2.1.

2.3 Deep Learning Methods for Relation Extraction

Before the emerging of language model methods, conventional deep learning methods (convolutional neural network and recurrent neural network) were the typical way to solve the relation extraction problem. Since we will demonstrate some preliminary experiment results from the these deep learning models, we introduce the basic knowledge of them here.

The applications of deep learning models were boosted by the availability of general and high-quality word representation (word embedding) [50]. Through word

embedding, each word could be represented by a low-dimensional and real-valued vector which contains the syntactic and semantic information of the word. Furthermore, a piece of text (or a sentence) could be represented by a sequence of vectors (or a matrix formed by putting vectors together in order). Convolutional neural network (CNN) and recurrent neural network (RNN) are the types of deep learning architecture that could deal with sequence (or matrix form) data.

CNN model ever achieved promising results on relation extraction tasks [31, 92] and it could be further improved by utilizing refined architecture to incorporate more lexical and syntactic information. [89] applied piece-wise max pooling process after convolutional layer to extract the structural features between the entities. The proposed method (piece-wise CNN) exhibited superior performance compared with pure CNN. Multi-channel CNN model in [58] added extra channel to capture the dependency information of the sentence syntactic structure, while the multi-channel CNN model in [64] integrated different versions of word embeddings to better represent the input words. Hua et al. [31] built a deep learning model based on shortest dependency path (SDP), which is considered to contain the most important information of the relation expression. A residual CNN model was proposed in [91] and achieved comparable performance with other deep learning models on protein protein interaction task.

Also, RNN model is a great solution for relation extraction problem because of its nature to model sequence data. The proposed method in [29] achieved great performance on PPI only using the word embedding as the input of LSTM model. In [90], the authors employed RNN-based model on two relation classification datasets, and illustrated its advancement of learning long-distance relation patterns. In [9], they combined convolutional neural network and recurrent neural network to make full use of the information in the shortest dependency path and the results showed that this model yields promising results on the SemEval-2010 Task 8 dataset.

In addition, attention mechanism is another way to boost the deep learning model performance by pay more attention on the important information of the input. In [80], the authors introduced multi-level attention on convolutional neural network

model to better detect patterns in heterogeneous contexts. In [32], the authors proposed a word level attention-based convolutional neural network to better determine which parts of the sentence are more influential and achieved impressive performance on SemEval-2010 relation extraction task 8. A attention-based LSTM model was proposed in [94] and it was shown that this model could better capture the most important semantic information in a sentence.

2.4 Deep Learning Model Architecture

In this section, we will introduce the architecture of deep learning models we will use in this dissertation – PCNN model and BiLSTM model.

2.4.1 PCNN Model

Usually, the CNN model for classification problem contains: 1). convolution layer(s) to detect the local features; 2). pooling layer(s) to summarize the local features; 3). fully connected layer(s) to classify each category; 4) a softmax layer to output a normalized probability of each category. Here we employ a refined version of CNN model — Piecewise CNN, which will be discussed in detail in the following subsections. Fig 2.2 shows the structure of piecewise CNN model.

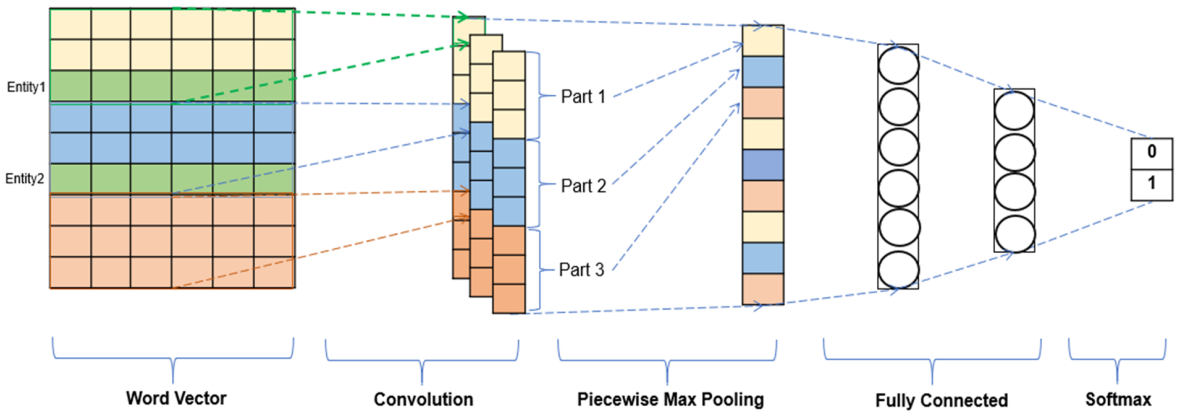


Figure 2.2: Structure of Piecewise CNN model.

2.4.1.1 Convolution Layer

Convolution is an operation between the input and a filter (a weight matrix), which could capture the local features in the sentence. After the convolution, we collect all the local features, so the output is sometimes referred to as the feature map.

Consider $x_i, i = 1, ..., n$ is the input (word vectors) and w is the filter, then the output of the convolution layer is:

$$C_i = wx_{i:i+k-1}$$

where index i ranges from 1 to n+k-1 and k is the window size of convolution.

In practice, we usually use many filters in the convolution layer to capture different local features. For example, if we use m filters, then we would obtain m convolution output C_i in this convolution window.

2.4.1.2 Pooling Layer

If convolution layer plays the role of finding the local features, then the pooling layer summarizes the nearby features and gives more high-level local features. In real-world application, convolutional networks may include local or global pooling layers to achieve this goal. Local pooling combines small clusters of nearby features, for instance 2 X 2. Global pooling operates on all the features of one feature map of the convolutional layer. In addition, pooling could be divided into several types based on the pooling operation, and max pooling and average pooling are two frequently-used ones. Max pooling takes the maximum value from each of the feature map, while average pooling uses the average value from each of the feature map. We will use max pooling in the model of this work.

Max pooling is used to find the most significant features in each feature map. The max pooling process can be expressed as:

$$P_j = max(C_{ij})$$

Regular max pooling does not give any structural information between the two entities, so we use a piecewise max pooling instead [89]. Specifically, we divide the sentence

into three parts based on the occurrences of entities of interest. Then max pooling operates piecewise on these three parts. For example, consider the sentence "We report an interaction between the human $\text{PS1}_{\text{PROTEIN}}$ or PS2 hydrophilic loop and $\text{Rab11}_{\text{PROTEIN}}$, a small GTPase belonging to the Ras-related superfamily." where the entities are $PS1$ and $Rab11$. The three parts now are: "We report an interaction between the human $\text{PS1}_{\text{PROTEIN}}$", "or PS2 hydrophilic loop and $\text{Rab11}_{\text{PROTEIN}}$", and "a small GTPase belonging to the Ras-related superfamily". The three outputs, obtained after convolution is applied on each part, are max-pooled independently. After that, these three outputs are concatenated together and form the final output of max pooling. See Fig 2.2, where the three parts of max pooling are shown in different colors.

Formally, the max pooling layer become:

$$P_{jl} = max(C_{ijl}), \quad l = 1, 2, 3$$

Then we concatenate these three outputs ($P_{j,1:3}$) and apply an activation function f to acquire the output of max pooling layer:

$$M = f(P_{j,1:3})$$

2.4.1.3 Fully-connected Layer

Fully connected layers in principle are the same as the traditional multi-layer perceptron neural network (MLP). It plays the role of classifying the features from the previous layer.

$$F = W_f M + b$$

where W_f is the weight matrix for the fully connected layer.

2.4.1.4 Softmax Layer

In order to obtain the probability of each category in classification problem, a softmax layer is an efficient way to achieve this goal.

$$p = softmax(F)$$

2.4.2 BiLSTM Model

In this section, we will introduce the architecture of BiLSTM model. The BiLSTM model contains: 1). an embedding layer to generate the input sequence; 2). two recurrent layers (forward and backward) to model the sequence data in bidirectional way; 3). one fully connected layer to classify each category; and 4). a softmax layer to output a normalized probability of each category. Please see Fig 2.3 for its architecture.

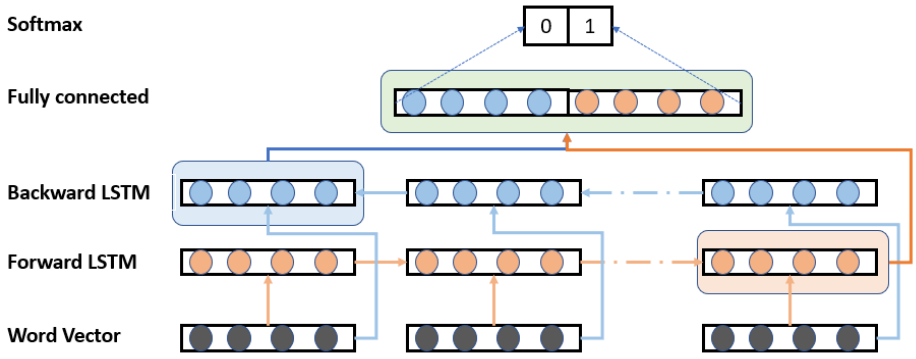


Figure 2.3: Structure of BiLSTM model.

2.4.2.1 Embedding Layer

The weights of embedding layer are the pre-trained word embedding vectors of vocabulary. It is to transform input index into word embedding, which is a low-dimensional (compare to vocabulary size) and real-valued vector.

2.4.2.2 Recurrent Layer

Recurrent layer is constructed by a sequence of LSTM cells. An LSTM cell contains a cell state to store information. The LSTM cell has the ability to remove or add information to the cell state by structures called gates. Each LSTM cell has three of these gates, to protect and control the cell state: input gate, forget gate and output gate. The input gate is to decide what new information to add in the cell state. The forget gate is to decide what information to throw away from the cell state. Finally,

the output gate regulates the output from the cell state. The structure of a LSTM cell is illustrated in Fig 2.4.

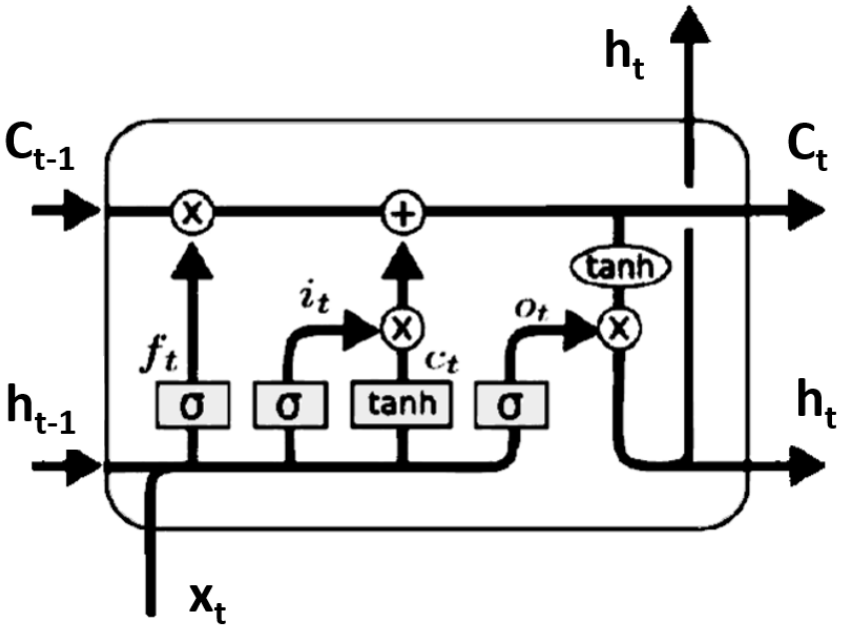


Figure 2.4: Structure of LSTM cell.

Formally, let x_t be the input, h_t be the cell output, c_t be the cell state at time t, and i_t, f_t, o_t be the input, forget, output gate respectively. Meanwhile, $W_{i,f,o,c}$ and $U_{i,f,o,c}$ denote the learnable weight matrix, and $b_{i,f,o,c}$ denotes the learnable bias in the cell. Then, the LSTM cell could be defined as:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i)$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o)$$

$$\tilde{c_t} = tanh(W_c x_t + U_c h_{t-1} + b_c)$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \tilde{c_t}$$

$$h_t = o_t \circ tanh(c_t)$$

where $\circ$ denotes the element-wise product of the vectors, σ and $tanh$ represent the sigmoid and hyperbolic tangent function respectively.

2.4.2.3 Fully-connected Layer and Softmax Layer

Similar to PCNN model, we use the fully-connected layer to classify the features from previous layer and utilize the Softmax layer to normalize the output to obtain probability of each category.

Chapter 3

IMPROVING BERT MODEL USING SUB-DOMAIN ADAPTATION FOR RELATION EXTRACTION

Recently, language model methods show great advancement in various NLP tasks and they have been applied to downstream tasks in various domains. The current state-of-art systems on biomedical relation extraction usually utilize adapted language model for biomedical domain. Among all the language models, we choose the BERT model in our investigation because it is well studied in many fields. As for biomedical domain, several versions of BERT models have been adapted: BioBERT [43], SciBERT [3], BlueBERT [59] and PubMedBERT [22]. These models are pre-trained using different collections of biomedical text and show different generalization behaviors on different biomedical tasks. Thus, we investigate how the pre-training text will impact upon the model generalization on downstream tasks.

We conduct a set of experiments to verify our hypothesis that the text used in pre-training for domain adaptation can have a significant impact. As can be seen from Table 3.2, the performance of BlueBERT model is significantly lower than the other two. Given the pre-training material used by BlueBERT and the other two models, we conjecture that this is due to the inclusion of clinical notes text that is not used by the other models. By removing the clinical notes text from its pre-training text, we observe a large improvement and comparable results for this third model as well (Table 3.3).

These experiments motivate us to further explore the impact of text used for domain adaptation in pre-training BERT models. Specifically, we will investigate whether more specific text about the entities involved in a specific biomedical relation might help over general biomedical domain knowledge. Therefore, we introduce another level of adaptation to adjust the domain adaptation to specific sub-domains. We call

it sub-domain adaptation. This sub-domain adaptation is fulfilled by pre-training the BERT model with sub-domain data, which is more related to the downstream tasks.

In this chapter, we will conduct experiments to investigate the effectiveness of sub-domain adaptation and also compare the generalization of different adapted BERT models. We evaluate our approach on three RE tasks: the protein-protein interaction (PPI) [37], the chemical-protein interaction (ChemProt) [38] and the drug-drug interaction (DDI) [26]. The experiment results show that another level of pre-training on task-related data boosts the model performance on our relation extraction tasks.

The rest of this chapter is divided into four sections. In Section 3.1, we will present the related work. The method of sub-domain adaptation for BERT model will be discussed in Section 3.2. In Section 3.3, we will present the experiment results of sub-domain adaptation using two BERT models. We will conclude in the last section.

3.1 Related Work

An advantage of language model methods is that it can leverage a large amount of unlabeled data to learn the context-dependent and universal language representations. In this chapter, we employ the BERT model [17] for our relation extraction tasks. In order to facilitate research in biomedical domain, several biomedical versions of BERT models (BioBERT [43], SciBERT [3], BlueBERT [59], and PubMedBERT [22]) have also been proposed and have achieved promising results on various biomedical tasks. While the first three biomedical BERT models are pre-trained based on the general-domain BERT, PubMedBERT [22] is pre-trained with whole-word masking from scratch using PubMed abstracts. As we discussed in Chapter 2, those models are pre-trained on different biomedical text. As a result, those models generalize different on specific tasks and here we propose an approach of sub-domain adaptation using task-related data to achieve better model generalization on specific tasks .

There are several studies utilize the labeled trained data in the pre-training of language model for better generalization [23, 62], however those methods are not

feasible for our tasks since we also need the context of the training sentences (for Next Sentence Prediction in the pre-training of BERT). Thus, we propose a new approach to acquire more general pre-training data for our tasks.

3.2 Methodology

In this section, we begin with a study of the impact of the corpora used to adapt BERT-based models to the biomedical domain. Since the evaluation results of these models in their original papers have been on different tasks and using different settings. There has not been any systematic comparison of the different BERT models adapted to the biomedical domain. Thus, we conduct a study to compare the three adapted BERT models, BioBERT [43], SciBERT [3] and BlueBERT [59] on the same set of tasks. Also, considering these three models have the same architecture and the primary difference between them is the pre-training data, this experiment essentially explores the effect of pre-training data on domain adaptation and downstream tasks they are applied to.

As it is shown in Table 3.2, the different pre-training data lead to significantly different performance on our focused tasks. Inspired by this observation, we will explore whether task-related pre-training data could improve BERT model. Specifically, we investigate adding an extra pre-training step using task-related data after the domain adaptation pre-training to help model generalization. We call it sub-domain adaptation as we are seeking to tailor the model further for specific tasks.

In sub-domain adaptation, we use the following technique to extract the sub-domain data that are related to a specific RE task. We assume that the text containing some type(s) of entity in a relation (e.g., protein for PPI relation) is relevant to the specific RE task. Since PubMed is our main source of pre-training data, we extract the PubMed abstracts containing specific types of entity as the sub-domain data. For example, for the PPI task, we extract abstracts from PubMed via the query "Protein OR Gene" and we obtain 7,729,611 abstracts for protein/gene domain using this query. Similarly, the DDI task involves the drug entities and so we use the PubMed query

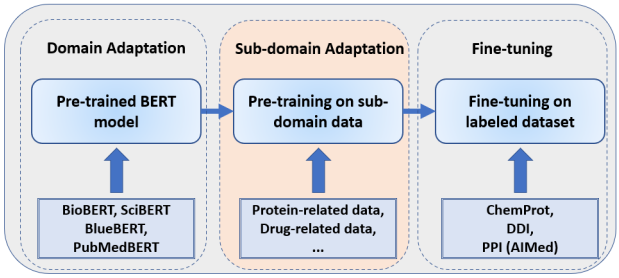


Figure 3.1: BERT model training process with sub-domain adaptation.

"Drug" and 5,714,799 abstracts are extracted as pre-training data for drug domain. For the ChemProt task, we utilize Pubtator [81] to extract the PubMed abstracts that contain protein/gene and chemical entities, and 3,375,380 abstracts are used as the sub-domain data.

The new training process of BERT model is illustrated in Fig 3.1. The first box represents the standard creation of BERT models for biomedical domain. The rightmost box also is the standard fine-tuning for the RE tasks. Our experiment here involves the inclusion of sub-domain adaptation in the middle box of Fig 3.1.

3.2.1 Experiment setup

For the sub-domain adaptation, we train the BERT with 100K steps on the sub-domain data using the learning rate of 2e-5, batch size of 192 and max sequence length of 128 in our experiments. In the sub-domain pre-training for BioBERT, SciBERT and BlueBERT, we follow their settings in masked language model task and only utilize subwords masking. In contrast, we employ whole word masking in the pre-training of PubMedBERT.

For the fine-tuning of the BioBERT models, we use the learning rate of 2e-5, batch size of 32, training epoch of 10 and max sequence length of 128. During the fine-tuning of PubMedBERT models, the learning rate of 2e-5, batch size of 16, training epoch of 10 and max sequence length of 256 are utilized.

Task	Label	Sentence examples
PPI	Positive	We have characterized the physical and functional interactions between @PROTEIN$ and @PROTEIN$.
	Negative	pRB, @PROTEIN$, @PROTEIN$(INK4d) and p27(KIP1) decrease in both cell types.
DDI	MECHANISM	@DRUG$ decreases the elimination of @DRUG$ causing an increase in overall exposure.
	EFFECT	Anticoagulants @DRUG$ may increase sensitivity to oral @DRUG$.
ChemProt	CPR:4	@CHEMICAL$ potently attenuated gene expressions involved in inflammation, such as iNOS, COX-2 and @GENE$.
	CPR:9	They suggest that TRPML1 works in concert with @CHEMICAL$ to regulate @GENE$ translocation between the cytoplasm and lysosomes.

Table 3.1: Examples after pre-processing from the three tasks.

3.2.2 Data pre-processing

As illustrated before, one instance of relation extraction task contains two parts: the text and the entity mentions in it. In order to make BERT model identify the position of the entities, we replace the entity names with predefined tags by following the standard pre-processing step for relation extraction. Specifically, all the protein names are replaced with @PROTEIN$, drug names with @DRUG$, and chemical names with @CHEMICAL$. In Table 3.1, we show some pre-processed examples from the three tasks.

In order to alleviate the out-of-vocabulary problem, BERT uses WordPiece embedding [86] and each input word will be split into subwords from WordPiece vocabulary. In addition, following the original BERT input representation, we add the segment and position information of the tokens in the input sentence through the segment embedding and position embedding.

3.3 Results and discussion

In this section, we provide the experiment results with two types of BERT model adapted for the biomedical domain: (1) BERT model adapted from general domain using biomedical text (e.g., BioBERT [43], BlueBERT [59], SciBERT [3]); (2) BERT model adapted from scratch using biomedical text (e.g., PubMedBERT [22]). Our study starts with the comparison of BERT models from the first type, and then we experiment with both types of BERT models on the proposed methods. We choose BioBERT as it generally achieves better performance on our tasks compared to SciBERT and BlueBERT (Table 3.2). For the second type of BERT model, we utilize PubMedBERT in our experiments.

To evaluate our approach, we use three well-known datasets of PPI, ChemProt, and DDI tasks. The statistics of these datasets can be found in Table 2.1 in Chapter 2. We use the the same spilt of training, development, test set with the PubMedBERT model [22] during the model evaluation on the DDI and ChemProt tasks. For the AIMed corpus, there is no standard split of training and test, we will employ 10-fold cross-validation on it. We use the standard precision (P), recall (R) and F1-score (F) to measure the model performance on PPI task since it is a binary classification task. However, the ChemProt and DDI tasks are multi-class classification problem, the models for these two tasks will be evaluated utilizing micro precision, recall and F1 score on the non-negative classes.

3.3.1 Impact of corpus on domain adaptation of different BERT models

The results of BERT models that are pre-trained on different biomedical text are summarized in Table 3.2. As it us shown, BlueBERT shows drop-off compared to the other two on our three datasets, especially on the PPI set. Thus, the text used in pre-training for domain adaptation appears to have a surprisingly significant influence on the performance of BERT models on downstream tasks. A noticeable difference with BlueBERT, when compared to the other two models, is the inclusion of clinical notes text in the domain adaptation process, which differs considerably from

Model	ChemProt	DDI	PPI
BioBERT	75.3	79.0	81.0
SciBERT	74.4	78.7	78.8
BlueBERT	71.2	76.8	71.9

Table 3.2: BERT model performance (F1 score) on ChemProt, DDI and PPI tasks.

Model	PPI			ChemProt			DDI		
	P	R	F	P	R	F	P	R	F
BlueBERT	69.3	75.0	71.9	70.9	71.5	71.2	76.2	77.4	76.8
BlueBERT (-M)	**76.6**	**83.1**	**79.6**	**74.7**	**75.8**	**75.2**	**80.0**	**78.5**	**79.2**

Table 3.3: BlueBERT model performance on PPI, ChemProt and DDI tasks before and after removing MIMIC-III from the domain adaptation data. -M: Subtract the MIMIC-III clinical notes.

the text used for pre-training other two models. Since all three evaluation sets are not related to the clinical domain, we conduct an experiment to see if the removal of this extraneous material from pre-training adaptation would impact the results. The results, shown in Table 3.3, suggest an improvement in BlueBERT's performance and it almost achieves the same results as the other two. This observation leads us to conjecture that task-related data in the pre-training might yield better generalization of BERT models on downstream tasks.

3.3.2 Sub-domain adaptation

While all BERT models try to use a large corpus of text from the biomedical domain to obtain the domain-adapted versions, our results indicate that the differences between the domain adaptation text might have a noticeable impact on individual tasks, as each task requires its own specific knowledge. Therefore, we wish to further consider the pre-training data for the pre-training phase.

Given our investigation involving several RE tasks, we consider the approach of utilizing task-related data to further adjust the adaptation for the domain underlying the tasks. Specifically, we extract data (unlabeled text) for three sub-domains from PubMed

for our tasks: protein/gene (P/G) domain, drug (D) domain and chemical+protein (CP) domain. These three sets of abstracts are used for sub-domain adaptation setup to produce three differently adapted BERT models: BERT(+P/G), BERT(+D) and BERT(+CP) respectively.

Model	PPI			DDI			ChemProt		
	P	R	F	P	R	F	P	R	F
BioBERT	79.0	83.3	81.0	79.9	78.1	79.0	74.3	**76.3**	75.3
BioBERT (+P/G)	**82.5**	**83.7**	**83.0**	76.1	77.6	76.9	76.5	74.2	75.3
BioBERT (+D)	81.5	80.9	81.2	**81.9**	78.4	**80.1**	**76.7**	74.4	75.6
BioBERT (+CP)	81.3	83.7	82.4	78.7	**79.0**	78.8	76.6	76.1	**76.4**
PubMedBERT	80.1	84.3	82.1	82.6	81.9	82.3	78.8	75.9	77.3
PubMedBERT (+P/G)	**81.2**	**85.5**	**83.3**	83.7	80.5	82.0	**80.5**	75.5	77.9
PubMedBERT (+D)	79.1	85.3	82.0	**84.1**	81.7	**82.9**	80.4	74.6	77.4
PubMedBERT (+CP)	79.6	84.7	82.0	81.1	**82.7**	81.9	79.4	**77.5**	**78.4**

Table 3.4: BERT performance after pre-training with sub-domain data. +P/G: add Protein/Gene-related PubMed abstracts as sub-domain data; +D: add Drug-related PubMed abstracts as sub-domain data; +CP: add protein-related and chemical-related PubMed abstracts as sub-domain data.

Table 3.4 presents the results of our experiments on sub-domain adaptation. The results are reported for both BioBERT and PubMedBERT, which have already been adapted to the biomedical domain. Their performances are shown in the first row and the fifth row, and serve as baselines to compare with those models that have additional pre-training for sub-domain adaptation. The other results in Table 3.4 shows the sub-domain adaptation can boost the model performance on related tasks. Note that not only are similar results obtained for BioBERT and PubMedBERT, but also for each task, the addition of sub-domain adaptation phase improves the performance. Furthermore, the maximum improvement is obtained when most relevant sub-domain text is used in this sub-domain adaptation phase. Specifically, the best results are obtained for the PPI task by using "protein/gene" related text. Also, the "drug" text leads to the highest performance for the DDI task. Both BioBERT and PubMedBERT obtain maximum benefits in the use of "chemical+protein" text for sub-domain adaptation on

the ChemProt task. On the other hand, adding drug related text does not help for the PPI task. Also, the addition of "protein/gene" text actually hurts the performance of both models when used for the DDI task.

All the experiment results indicate that the pre-trained BERT models generalize differently on different tasks, and it is beneficial to add another level of adaptation on task-specific data. In this chapter, we have demonstrated that using entity-related data in the sub-domain adaptation helps the model generalization on relation extraction tasks.

3.4 Summary

In this chapter, we first conduct an experiment to investigate the generalization of different adapted BERT models for biomedical domain on the same set of relation extraction tasks. The experiment results demonstrate that the corpora for domain adaptation affect the generalization of BERT models on different tasks. We also see that the use of task-irrelevant pre-training data (MINIC-III on BlueBERT) hurts the performance of BERT on the RE tasks. The results of these experiments motivate us to explore the approach of sub-domain adaptation on BERT models to help model generalization on specific tasks.

For the sub-domain adaptation, we add another level of pre-training utilizing the entity-related text for the relation extraction task as the pre-training data. We show that the knowledge gained from sub-domain adaptation boosts the BERT model performance on related relation extraction task, which indicates that the pre-training on task-related data help the model generalization on relevant tasks. Specifically, the "protein/gene" related text in sub-domain adaptation helps the BERT model generalize better on the PPI task. Similarly, the sub-domain adaptation on "drug" text and "chemical+protein" text yields better performance on DDI and ChemProt, respectively. In the future, we will explore on the impact of pre-training corpus for domain adaptation of BERT model on additional tasks other than relation extraction.

Chapter 4

IMPROVING BERT MODEL WITH REFINED FINE-TUNING FOR RELATION EXTRACTION

Since the development of BERT [17], previous works on relation extraction and other classification tasks have all utilized the suggested approach of using a special token called the classification token ([CLS]) in the topmost layer. Thus, during fine-tuning for classification tasks, all information is funneled through a part of the top layer during back-propagation. In this chapter, we will explore the potential of utilizing all the information in the last layer to improve model performance on relation extraction tasks.

Our work is partly driven by an observation in [75], where the nature of information represented in various layers of BERT model was studied. The authors found that the low-level morphological and syntactic aspects in language are represented in lower layers, and more semantic nature of text is learned in higher layers of BERT model. Since relation extraction task is clearly semantic in nature, important information for this task should reside in top layers of the BERT model. To test this hypothesis, we first employ the probing technique [76] to explore the information in the last layer of BERT model and find that the information in the last layer is useful to our relation extraction tasks. Therefor, we conjecture it will be beneficial to utilize all the knowledge in the last layer of BERT model during fine-tuning.

To incorporate the ignored information, we employ two different methods to summarize the information in the last layer: recurrent neural network (RNN) model [68] and attention mechanism [2], and then concatenate the summarized knowledge with the [CLS] output as the new output. We evaluate our methods on RE tasks in the biomedical domain: the protein-protein interaction (PPI) [37], the drug-drug interaction (DDI) [26], and the chemical-protein interaction (ChemProt) [38]. Our results show

that utilizing all the information from the last layer boosts the model performance on all three tasks. In addition, we achieve even better results on the three benchmark datasets after combining the refined fine-tuning with the sub-domain adaptation in Chapter 3.

We have organized the rest of this chapter in the following way. In Section 4.1, we will present the related work. In Section 4.2, we will discuss the details of methodology and experiment design. The experiment results and more analysis about our fine-tuning mechanism will be presented in Section 4.3. We will give a summary of this study in Section 4.4.

4.1 Related Work

BERT model [17] has been widely used on various natural language processing tasks in different domains. However, most of previous works only use the default way of fine-tuning for classification problem — utilizing the output of classification token ([CLS]) in the last layer of BERT model, which might discard the other useful information in the outputs of the BERT model. Few work have tried different fine-tuning for BERT model.

In the work [70], the authors explored the methods of extracting knowledge from the intermediate layers in BERT and demonstrated the advancement on aspect based sentiment analysis and natural language inference tasks. Since residual connections are employed in the Transformer, all important information in intermediate layers should be transmitted to the last layer of BERT model during training. We will compare this method with our approach of utilizing all the information from the last layer in BERT model.

In addition, the work [75] found that the BERT model represents the steps of the traditional NLP pipeline in a sequential way, which means the BERT model captures the linguistic information in the sequence of part-of-speech (POS) tagging, parsing, NER (Named Entity Recognition), semantic roles, then coreference. Their experiments also showed that the representation for relation classification continues to improve up to

the highest layer of the BERT model, indicating that the last layer contains important information for the relation extraction tasks.

4.2 Methodology

As we discussed before, the BERT model for relation extraction classification problem will only utilize the classification token when it is fine-tuned on the training dataset. If the last layer contains useful information about the task, we should include it to improve the model performance during fine-tuning. In this section, we will first introduce the edge probing technique and then utilize it to verify the usefulness of the information in the last layer of BERT. At last, we will propose two methods to involve the information from the last layer in a refined fine-tuning mechanism.

4.2.1 Introduction of edge probing

Edge probing was proposed in [76] to measure the quality of encoded information about linguistic structure in a pre-trained encoder (BERT in our case). Specifically, edge probing aims to evaluate how well the word is represented in each position, and what knowledge is learned about the structural information of the sentence. For instance, through edge probing, we can know if the constituent information (like noun phrase or verb phrase) of the words is encoded in the representation from the encoder. To achieve this goal, edge probing converts this problem into classification tasks and builds probing classifiers using the word representations and the expected labels. Based on the performance of the probing classifier, we can measure how well the structural information about that word in the sentence is encoded. Let us take the constituent type as an example, the phrase "is a global brand" in the sentence "The important thing about Disney is that it [is a global brand]." is a verb phrase (VP), so the probing classifier should predict the type "VP" when it is given the representation of "is a global brand". Usually, we use multi-layer perceptron (MLP) for probing classifier, and the parameters of the encoder are frozen during the training of probing classifier. For more

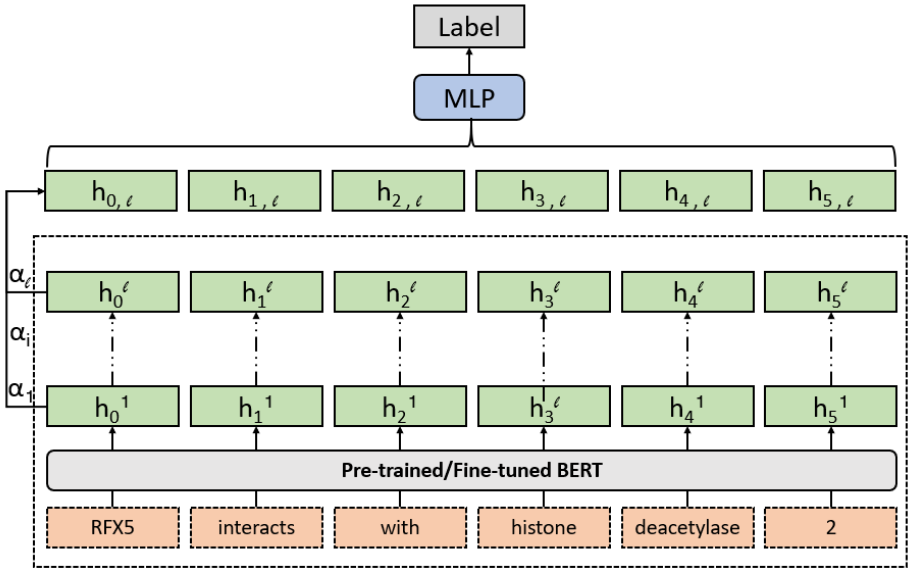


Figure 4.1: Probing classifier architecture. We freeze the parameters of BERT model during the training of probing classifier. Through the learned α, we can know the relevance between each layer and the task. Also, we can tell which layer learns the knowledge for a specific instance by building a series of probing classifier $\{P_\tau^l\}_{l=1}^L$. For the relation extraction instance "RFX5 interacts with histone deacetylase 2", if the probing classifier P_τ^l predicts the interacting relationship between proteins "RFX5" and "histone deacetylase 2" correctly using the information of the first l layers, but P_τ^{l-1} does not predict correctly using the information of the first (l-1) layers. We can say that the knowledge about this instance is learned in the l-th layer.

detailed description of probing classifier on different type of relations, we refer readers to the paper [76].

Originally, the edge probing was used to investigate the role of words in the sentence [76]. In [75], the scalar mixing technique [61] is combined with edge probing to explore the encoded information from different layers of the encoder. In particular, through this approach, we can tell which layer(s) are most relevant to a specific task. In formal, let $h^l = [h_0^l,\ h_1^l, ...,\ h_N^l]$ be the word representation after l-th layer of the encoder, then we can build a probing classifier (Fig 4.1) using the first l layers:

$$h_{i,l} = \gamma_l \sum_{k=0}^{l} \alpha_l^k h_i^k,\ \ i = 1, ..., N$$

$$P_\tau^l = MLP([h_{0,l}, h_{1,l}, ..., h_{N,l}])$$

where $\alpha_l = softmax([\alpha^{(0)}, \alpha^{(1)}, ..., \alpha^{(l)}])$ and N is the sequence length. During the training, the parameters γ_l and α_l will be jointly learned. After the training of P_τ^l, we can use the learned parameters α_l to estimate the contribution of each layer for our task. If the weight $\alpha^{(i)}$ for the i-th layer is high, we can say that the i-th layer contains more relevant information about our task. In this chapter, we will utilize the framework of edge probing to measure the learned knowledge in the layers of the BERT model.

4.2.2 Investigation of the knowledge in the last layer

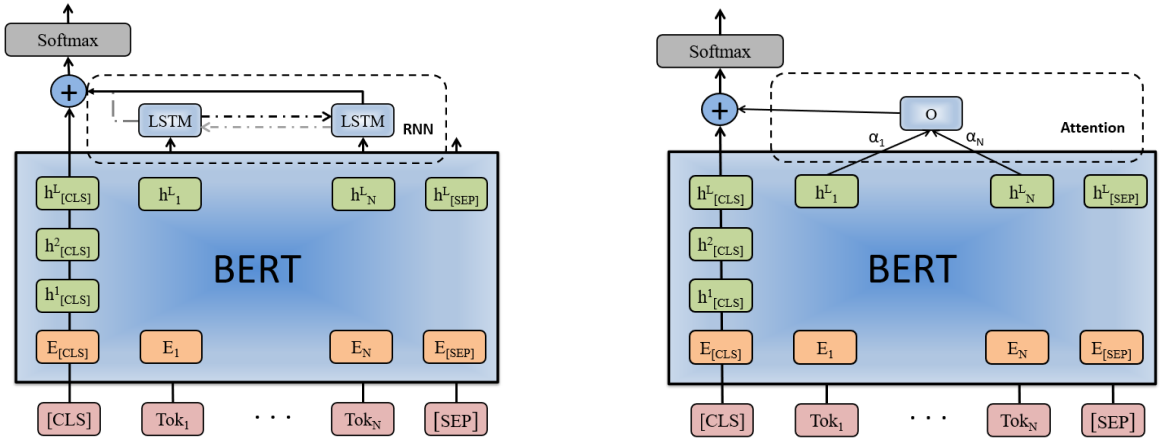
In the previous section, we demonstrate that the contribution of each layer in BERT can be learned through the weights for layers in a probing classifier. However, those weights are irrelevant to the training data distribution, which means we can not tell how many layers BERT needs to predict a specific instance correctly. This yields a problem: what knowledge about the training data is only learned in the l-th layer, but not learned in the first (l-1) layer(s)?

Similar to the exploration in [75], the above problem can be addressed by training a series of probing classifier $\{P_\tau^l\}_{l=1}^L$, where L is the total layers of the encoder. The classifier P_τ^l utilizes all the information from the first l layers, while the knowledge in the first l-1 layers is employed in the classifier P_τ^{l-1}. Therefore, the difference Δ_τ^l of performance score (F1 score) on test data between P_τ^l and P_τ^{l-1} stands for the learned knowledge only from the l-th layer:

$$\Delta_\tau^l = Score(P_\tau^l) - Score(P_\tau^{l-1})$$

After calculating $\{\Delta_\tau^l\}_{l=1}^L$, we can acquire the distribution of the learned knowledge from the training data in each layer.

Our purpose is to investigate whether the last layer contains useful information for our tasks, so here we build the probing classifiers using the fine-tuned BioBERT model [43]. During training, we freeze the weights of BioBERT and only adjust the parameters of the probing classifiers. We show the score Δ_τ^L (measurement of knowledge)



(a) RNN on the last layer

(b) Attention on the last layer

Figure 4.2: Model architectures of including all outputs in the last layer. In (a), we show both LSTM (only black in the RNN box) and biLSTM (both black and grey line in the RNN box).

of the last layer of BioBERT for the datasets of our three tasks in Fig 4.3. We can see that the last layer (12-th layer for $\text{BERT}_{\text{base}}$ model) contains useful information in all cases and it is necessary to include the discarded knowledge during fine-tuning.

4.2.3 Refined fine-tuning process of BERT model

We have shown that the fine-tuning method of only using classification token ([CLS]) discards some useful information from the last layer. Also, during the pre-training of BERT, the [CLS] token is only used in the next sentence prediction (NSP) task. This indicates that the [CLS] token might not encode the information about the interaction between entities because it is not designed to capture this kind of information. In this section, we will design a mechanism to include all the information in the last layer during the fine-tuning of BERT model. The proposed mechanism is called SLL fine-tuning: fine-tuning with information summarization in the last layer of BERT.

Our method includes two steps: 1). An new module to summarize the ignored outputs in the last layer; 2). Concatenating the summarized information with the [CLS] output as final output from BERT. In step 1, we explore two types of methods to

summarize the information in the last layer: recurrent neural network (LSTM [28] and biLSTM [21]) and attention mechanism. We show the model architectures of these two methods in Fig 4.2.

Formally, let H be the dimension of hidden states and L be the layer number of BERT model, then all the information in the last layer can be represented:

$$h^L_{all} = \{h^L_{CLS},\ h^L_1,\ h^L_2,\ ...,\ h^L_N,\ h^L_{SEP}\}$$

where h^L_{CLS} and h^L_{SEP} are the classification token output and separation token output respectively. Previously, only h^L_{CLS} is used for classification problem during fine-tuning. Here we first summarize the discarded information in the last layer, a.k.a., $h^L = \{h^L_1,\ h^L_2,\ ...,\ h^L_N\}$ (the sentences separation token h^L_{SEP} is ignored here) using the RNN sequence model and attention mechanism:

1. The first choice of model to summarize a sequence is the recurrent neural network. Among RNN models, we choose LSTM because it handles the "long-term dependencies" better. Also, we can utilize biLSTM to summarize the sequence in both forward and backward directions. We take the output of last LSTM unit as the representation of our sequence in LSTM. As for biLSTM, we just concatenate the outputs of first unit in backward direction and the last unit in forward direction as the final representation:

$$O = \begin{cases} LSTM(h^L_i) & (LSTM) \\ LSTM(\overrightarrow{h_i}^L) \oplus LSTM(\overleftarrow{h_i}^L) & (biLSTM) \end{cases}$$

2. Attention mechanism is another option of summarizing sequence by assigning a weight to each component. In particular, we will employ the additive attention to combine the sequence:

$$[\alpha_i] = softmax(h^L K)$$

$$O = \sum_{i=1}^{N} \alpha_i h^L_i$$

where $K^{H\times 1}$ are trainable parameters.

After the summarization, we concatenate the output from RNN/attention mechanism with the [CLS] output to form the final output of BERT:

$$h = h^L_{CLS} \oplus O.$$

Then, we put a softmax layer on top of this representation to calculate the probability distribution on the labels:

$$p = softmax(W_f h + b_f)$$

where $W_f^{C \times H}$, $b_f^{C \times 1}$ are also trainable parameters and C is the number of classes (categories) of our classification task.

Meanwhile, the contribution of [CLS] token for the classification task needs to be explored. So we experiment with another fine-tuning process without [CLS] token. Formally, we only take $h = [O]$ as the final representation of the input and utilize the same softmax layer $p = softmax(W_f h + b_f)$ for prediction. The performance of these two proposed fine-tuning processes can shed light on the roles of the two parts (the [CLS] token and sentence outputs) of the last layer for classification problem.

4.2.4 Combining the techniques for pre-training and fine-tuning

In Chapter 3, we proposed sub-domain adaptation technique to improve BERT model in the pre-training stage. In this chapter, we design a refined mechanism for the fine-tuning stage, which is independent to the pre-training of BERT. So a natural idea is to combine these two techniques. For the new BERT model, we add an extra step for sub-domain adaptation in the pre-training and utilize the new mechanism in fine-tuning stage. In this way, we take the full advantage of the knowledge in the sub-domain data and the learned information in the last layer of BERT model.

4.2.5 Experiment setup

For the fine-tuning of the BioBERT models, we use the learning rate of 2e-5, batch size of 32, training epoch of 10 and max sequence length of 128. During the fine-tuning of PubMedBERT models, the learning rate of 2e-5, batch size of 16, training epoch of 10 and max sequence length of 256 are utilized. In the edge probing experiments, we use a two-layer fully-connected network as the probing classifier, and the hidden layer has 1024 units. During training of probing classifiers, we use learning rate of 2e-5 and epoch of 4.

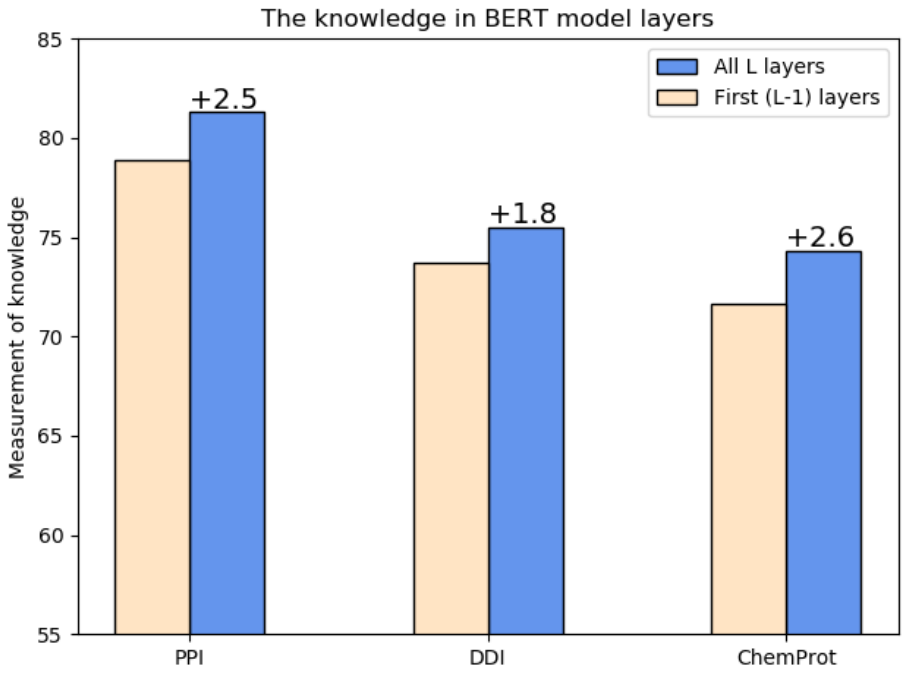


Figure 4.3: Learned knowledge of training data in the layers of BioBERT. L is the total layers of the BERT model. "Measurement of Knowledge" is the Δ_τ^L that is defined in Section 4.2.2

4.2.6 Data pre-processing

We utilize the same technique to pre-processing the input as it is illustrated in Chapter 3.

4.3 Results and Discussion

In this section, we first present the results of exploring the learned knowledge in the last layer and the SLL fine-tuning. Then, the model performance of combining the proposed approaches for the pre-training and fine-tuning is provided for the three tasks. Finally, we conduct an analysis on the learned attention weights from our SLL fine-tuning.

4.3.1 Learned knowledge in the last layer of BERT

Using the technique of probing classifier, we show that the last layer of BERT model contains useful information that can be exploited to improve performance on

Model	PPI			DDI			ChemProt		
	P	R	F	P	R	F	P	R	F
Bio	79.0	83.3	81.0	79.9	78.1	79.0	74.3	**76.3**	75.3
Bio_SLL_LSTM	80.2	84.0	82.0	80.5	78.5	79.5	77.6	74.4	76.0
Bio_SLL_biLSTM	80.2	82.7	81.4	80.8	78.5	79.6	**77.9**	73.9	75.9
Bio_SLL_Att	**80.7**	**84.4**	**82.5**	**81.6**	**79.4**	**80.5**	77.5	75.1	**76.3**
PubMed	80.1	84.3	82.1	82.6	81.9	82.3	78.8	75.9	77.3
PubMed_SLL_LSTM	79.8	**85.6**	82.6	82.6	**82.8**	82.7	**78.9**	77.0	**77.9**
PubMed_SLL_biLSTM	80.5	82.6	81.7	82.6	81.4	82.0	78.5	76.5	77.5
PubMed_SLL_Att	**81.3**	85.0	**83.1**	**84.3**	82.7	**83.5**	78.3	**77.6**	**77.9**

Table 4.1: BERT model performance on PPI, DDI and ChemProt tasks. Bio/PubMed: original BioBERT/PubMedBERT model; Bio/PubMed_SLL_LSTM: model of summarizing the outputs of the last layer using LSTM; Bio/PubMed_SLL_biLSTM: model of summarizing the outputs of the last layer using biLSTM; Bio/PubMed_SLL_Att: model of summarizing the outputs of the last layer using attention mechanism.

downstream tasks (Fig 4.3). Specifically, we compare the performance of probing classifiers using all L layers (L is the total layer number of the BERT model) and first (L-1)-th layers to measure the knowledge captured in the last layer (not present in the previous layers). We can see that the probing classifier using the information in all L layers performs better on all three tasks, which means many instances are predicted correctly by adding the knowledge in the last (L-th) layer, but not using the knowledge in the first (L-1)-th layers. Thus it is beneficial to utilize those information during fine-tuning of BERT. In addition, the information in the last layer is automatically generated during training, we can just utilize it without extra cost.

4.3.2 SLL fine-tuning: utilizing all the information from the last layer

Having shown that the last layer contains some useful information, which is not exploited by current transformer-based models for RE, we consider two different methods to utilize it and include the summarized information in a refined fine-tuning process. In Table 4.1, we provide the model performance using different methods: RNN models (both LSTM and biLSTM) and attention mechanism. Clearly, the method of utilizing

attention mechanism on the last layer (BERT_SLL_Att) achieves the best performance on all the tasks. Specifically, the attention-based method on BioBERT model achieves F1 score improvement of 1.5%, 1.5% and 1.0% on PPI, DDI and ChemProt, respectively. Similarly, we observe similar F1 improvements with the PubMedBERT model for the three tasks.

Even though the LSTM and biLSTM methods show less improvement compared to the use of attention mechanism, they can improve the model performance in most of the cases. We also observe that the LSTM and biLSTM have very similar performance, with LSTM being slightly better in general. Since the training of BERT consider the context of words from both directions, we hypothesize the position information is already encoded in the word representation and biLSTM will not provide extra information (which partially explains the experiment results). Therefore, we will only utilize LSTM when considering RNN model for the following experiments.

Model(BERT)	PPI			DDI			ChemProt		
	P	R	F	P	R	F	P	R	F
Bio	79.0	83.3	81.0	79.9	78.1	79.0	74.3	76.3	75.3
Bio_SLL_Att	80.7	84.4	82.5	81.3	80.1	80.7	76.5	**77.1**	76.8
Bio_SLL_Att (+P/G)	**83.1**	**84.7**	**83.8**	80.4	79.7	80.0	78.4	75.1	76.7
Bio_SLL_Att (+D)	81.5	84.5	82.9	**82.6**	**81.2**	**81.9**	76.8	74.7	75.7
Bio_SLL_Att (+CP)	82.5	84.2	83.3	81.7	77.0	79.3	**78.9**	75.2	**77.0**
PubMed	80.1	84.3	82.1	82.6	81.9	82.3	78.8	75.9	77.3
PubMed_SLL_Att	81.3	85.0	83.1	84.3	82.7	83.5	78.3	77.6	77.9
PubMed_SLL_Att (+P/G)	81.1	**87.1**	**84.0**	83.6	80.6	82.1	**79.8**	77.0	78.4
PubMed_SLL_Att (+D)	**81.4**	84.5	82.9	84.9	**83.2**	**84.0**	79.5	75.9	77.7
PubMed_SLL_Att (+CP)	**81.4**	85.7	83.4	**85.0**	81.4	83.2	79.7	**77.7**	**78.7**

Table 4.2: BERT performance after combining sub-domain adaptation and the refined fine-tuning mechanism. Bio/PubMed: original BioBERT/PubMedBERT model; Bio/PubMed_SLL_Att: model of summarizing the outputs of the last layer using attention mechanism. +P/G: add Protein/Gene-related PubMed abstracts as sub-domain data; +D: add Drug-related PubMed abstracts as sub-domain data; +CP: add protein-related and chemical-related PubMed abstracts as sub-domain data.

Model	PPI			DDI			ChemProt		
	P	R	F	P	R	F	P	R	F
BioBERT	79.0	83.3	81.0	79.9	78.1	79.0	74.3	76.3	75.3
BioBERT_SLL_Att	80.7	**84.4**	82.5	**81.3**	80.1	**80.7**	**76.5**	**77.1**	**76.8**
BioBERT_SLL_Att*	**82.3**	83.5	**82.8**	79.7	77.6	78.6	76.4	74.5	75.4
PubMedBERT	80.1	84.3	82.1	82.6	81.9	82.3	**78.8**	75.9	77.3
PubMedBERT_SLL_Att	**81.3**	85.0	**83.1**	**84.3**	**82.7**	**83.5**	78.3	77.6	**77.9**
PubMedBERT_SLL_Att*	80.0	**85.2**	82.4	82.5	80.9	81.7	75.7	**77.7**	76.7

Table 4.3: Model performance without using the [CLS] token in the last layer. BERT_SLL_Att*: models of fine-tuning without [CLS] token and only using the summarized information from attention mechanism.

4.3.3 Combining sub-domain adaptation and SLL fine-tuning mechanism

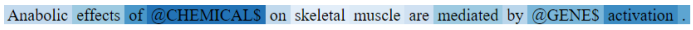
Since we propose techniques to improve two independent stages of BERT model, we can combine two techniques to explore the potential of their combination. As shown in Table 4.2, combining the sub-domain adaptation and the proposed fine-tuning mechanism can further boost the model performance on all the three tasks. The first two rows for BioBERT and PubMedBERT are just repetitions of the results from Table 4.1 and provide the baselines for the results using combined techniques. Compared with original BioBERT model, the model with combined techniques can achieve 2.8%, 2.9% and 1.7% F1 score improvement on PPI, DDI and ChemProt tasks respectively. Similarly, we can also boost the PubMedBERT model performance with 1.9%, 1.7% and 1.4% F1 score improvement on the three tasks, respectively.

4.3.4 More analysis: roles of classification ([CLS]) token in fine-tuning

We have shown that the proposed fine-tuning method outperforms original fine-tuning mechanism in Table 4.1. Both methods includes the use of [CLS] token of the last layer, so it will be helpful to understand the role of [CLS] token. Here we experiment with the methods of fine-tuning without [CLS] token to explore the contribution of [CLS] token. In particular, we just drop the [CLS] output and utilize the summarized information from attention mechanism as the output of BERT model.

(a) Example from PPI corpus (Label: Positive).

(b) Example from DDI corpus (Label: EFFECT).



(c) Example from ChemProt corpus (Label: CPR:3).

Figure 4.4: Attention weights visualization.

Here we experiment with both BioBERT and PubMedBERT models. In Table 4.3, the third and sixth row (BioBERT_SLL_Att* and PubMedBERT_SLL_Att*) show the model performance of utilizing the summarized outputs of last layer without [CLS] token. The rows before are only repetitions of the results from Table 4.1. We can see that the fine-tuning without [CLS] hurts performance on most cases and only BioBERT_SLL_Att* model performs better with small margin on PPI task. The results indicate that the [CLS] token contains key information for our classification tasks.

In addition, the authors in [70] explore the contribution of [CLS] tokens from the intermediate layers of BERT, we also experiment with this method and observe the drop in the performance after including the [CLS] outputs from the intermediate layers in all tasks. Thus, we will not present those results here, but the results imply that it is better to use information from the last layer since the useful information in the intermediate layers will be transmitted to the latter layers during training.

4.3.5 Analysis of attention weights in the SLL fine-tuning

In previous section, we illustrate that utilizing an additional attention mechanism yields better performance. We conduct some preliminary experiments to understand whether the additional mechanism focuses on specific parts of the input text for our tasks. We first visualize the attention weight distribution on the words of the sentences and examine the words the model is paying more attention to. In Fig 4.4, we have chosen one example from each of the three tasks that were misclassified by the original BioBERT model but are correctly predicted by the new method. In Fig 4.4, the darkness of the color represents the attention weight, with words in darker color having larger

weights.

We can see that the attention mechanism is assigning large weights on the words that indicate the relation between the entities, which are usually called "trigger words" for the relations in the NLP field. It appears that the attention mechanism is learning to focus on the "trigger words" in our tasks when making its predictions. For example, in the PPI sentence (Fig 4.4a), the words of entity token (@PROTEIN$) and the word "activates" (trigger word of PPI) have larger weights than other words. Similarly, the trigger words "inhibit" and "mediate" have larger attention weights in the DDI and ChemProt examples respectively. Recall these sentences were not correctly predicted when attention was not used to highlight the information from the "trigger word", which also demonstrates the importance of the information from the last layer.

In order to understand the attention weight especially on trigger words in a more general way, we choose to consider all the words and investigate the overall weight distribution on the words in the corpora. Given the fact that the trigger words usually appear near the entities of the relation, we collect all the weights of three words around the entities in the positive instances and take the average of the weights to acquire the global attention for each word. In Table 4.4, we show the top 10 words that have large learned weight from the attention mechanism for the dataset of each task. Among the top words, we can see that most of them can be considered as "trigger words" of the relations, which indicates that the attention mechanism is learning some key knowledge of relation expression.

4.3.6 Trigger word weight comparison between positive and negative instances

We have seen that the attention mechanism is giving relatively large weights on trigger words of relations on positive instances. For the negative instances, small weights should be learned on trigger words since there is no relationship between the entities. Therefore, we compare the trigger word weights between positive and negative instances and give six examples in Fig 4.5 from the three tasks. We can see that the learned

Task	Word Stem
PPI	activ(ate), complex, associ(ate), interact, human protein, bind, domain, specif(y), receptor
DDI	concomitantli, combin(e), concomit(ant), increas(e), use concurr(ent), decreas(e), inhibit, receiv(e), administ(er)
ChemProt	phosphoryl(ate), attenu(ate), stimul(ate), deriv(e), regul(ate) novel, metabol(ize), reduc(e), induc(e), inhibit

Table 4.4: Top words with large attention weight from corpora of the PPI, DDI and ChemProt tasks. Considering the different forms of the words, we utilize Porter's stemmer [63] to remove the morphological affixes from each word and only use the word stem for the global attention weight calculation. For instance, the stem for the word "activate" is "activ", and many other words like "activates" and "activation" have the same word stem.

trigger word weights in positive instances are much larger than the weights in the negative instances. This demonstrates that the attention mechanism can automatically learn the difference between positive and negative instances.

4.4 Summary

In this chapter, we propose a refined fine-tune process for BERT model to utilize all the knowledge from the last layer. Specifically, we explore recurrent neural network and attention mechanism to summarize the information in the last layer of BERT and then utilize the summarized information in the SLL fine-tuning process. We verify the effectiveness of the methods on three relation extraction tasks: PPI, DDI, and ChemProt. The experiment results show that the proposed fine-tuning with attention mechanism achieves better performance on all three tasks.

Also, we combine the proposed approach for improving pre-training of BERT model in the previous chapter with our refined fine-tuning , and we achieve even better performance on three benchmark datasets of the relation extraction tasks.

In addition, the role of classification token in the last layer of BERT model is further explored and the experiment results demonstrate that it contains important knowledge for our relation extraction tasks. Furthermore, the analysis on the attention

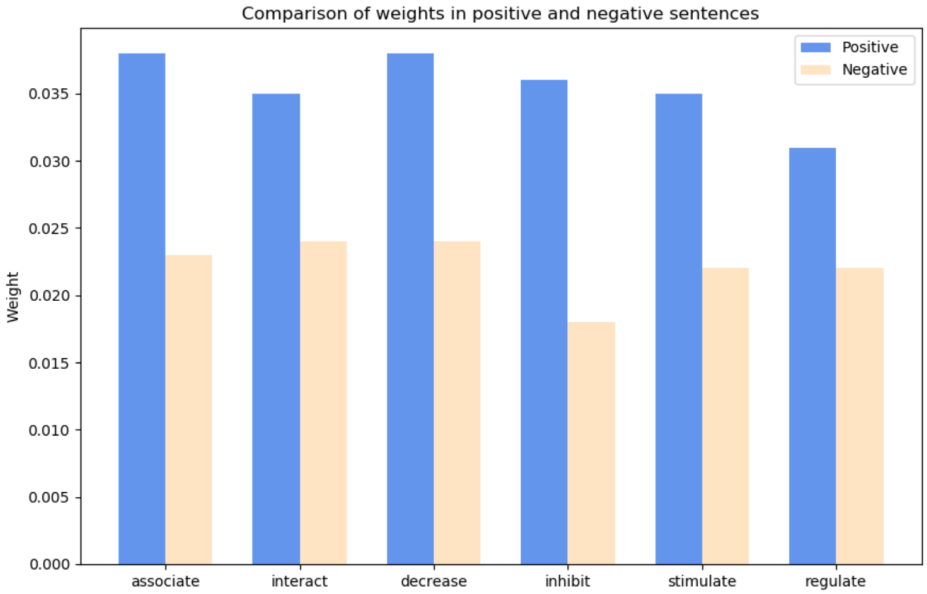


Figure 4.5: Comparison of trigger word weights in positive and negative instances. The first two words (associate and interact) are from the PPI task, the middle two words (decrease and inhibit) are from the DDI task and the last two words (stimulate and regulate) are from the ChemProt task.

weights in the refined fine-tuning process indicates that the attention mechanism is using the key information (trigger words of the relation expression) for classifying our relations between different biomedical entities.

Chapter 5

DISTANTLY SUPERVISED RELATION EXTRACTION

Recently, large deep learning models, especially language models, show great advancement in various natural language processing (NLP) tasks. However, each new task requires its own annotated data for training the deep learning model. Also, deep learning models usually require setting of a large number of parameters and hence typically require large datasets. Currently, only small datasets are available for a number of tasks and this situation can hinder us from achieving the full potential of deep learning models. Although language model methods alleviate this problem to some extent through the pre-training on unlabeled data, the generalization on specific tasks still relies on the task-specific (labeled) data. In order to address this problem, we consider the technique of distant supervision (DS) to help generate a large amount of labeled data and then use the DS-generated data to help the generalization of deep learning models. In addition, we will apply some heuristics to reduce the noise in the DS-generated data, and the results illustrate that the model built on noise-reduced data achieves better performance.

After we explored the behaviors of deep learning model on different DS-generated datasets with different noise reduction heuristics, it turns out that the deep learning models built on DS-acquired data do not perform as well as the ones based on much smaller sized human-labeled datasets. It might be the remaining noise in the DS data after noise reduction that affect the model performance. Thus, we have to design better ways to utilize the large amount of DS-generated data.

While DS data and manually labeled data have been used individually to train models, few works attempt to use them together for training model. In this chapter, we will design methods to train models that use both the DS-obtained data and manually

annotated (MA) data simultaneously. We hypothesize that combining the automatically generated data (DS data) and MA data could make the model learn the knowledge from the two types of data, and help the model generalize better. Specifically, we consider two methods in this work: simple union and transfer learning. Transfer learning is a technique where a model (often called the source model) developed for a task is reused as the starting point for training a second (target) model on another related task [41]. The hypothesis is that since the target model starts with learned knowledge from the source model, it will achieve better performance than the models trained from a random start. In addition, we also hypothesize that the learned knowledge from automatically generated data becomes more important when less manually labeled data are available.

Since the distant supervision technique can only label the data with positive or negative by checking whether this relation appear in the used database or not. In this chapter, we only consider binary relation extraction tasks in BioNLP for our method evaluation. Specifically, we will use the following tasks to verify the effectiveness of our method: the protein-protein interaction (PPI) [37], the miRNA-gene regulation (MIRGENE) [45], and the protein subcellular localization (PLOC) [35].

The remainder of this chapter is organized in the following way. We will discuss the related work in Section 5.1. In Section 5.2, we will describe the methods of utilizing data from distant supervision and experiment settings, followed by the evaluation results and discussion in Section 5.3. We will conclude in Section 5.4.

5.1 Related Work

Even though great advantage has been gained by deep learning, the limited human-labeled data restrict its use because each new task requires its own annotated data for training the deep learning model. To train deep learning models effectively, it usually needs large datasets, which requires significant human efforts. In order to alleviate the data limitation problem, distant supervision has been proposed as a technique of labeling data for relation extraction automatically. Mintz et al. [53] first introduced the term "distant supervision" and applied this technique to generate a large

dataset for Freebase relation extraction. Before that, Craven et al. [14] had already used the relation instances (tuples) gathered from some databases to label abstracts gathered from Medline, which pioneered the distant supervision method. Since then, distant supervision has been applied on many NLP tasks. Go et al. [20] applied distant supervision to automatically classify the sentiment of Twitter messages, and Surdanu et al. [72] used distant supervision approach for the TAC-KBP slot filling task. In the biomedical field, distant supervision has also been proven to be effective on extracting protein subcellular localization [93] and microRNA-gene relations [39]. In the case of relation extraction, distant supervision can also be used to automatically obtain large training datasets using a knowledge base and large amounts of literature.

Noise in the labeling from distant supervision is a well-known problem and can adversely affect the performance of deep learning models [67]. To reduce the noise, many techniques have been proposed and the results show their effectiveness in the improving performance of DS-based models. One solution is to relax the original strong assumption of DS, which assumes that all text mentioning a entity pair from the knowledge base express that relation. At the same time, many other methods have also shown their advantages of reducing noise in DS data. In [93], the authors introduced a threshold for the frequency of dependency paths among positive examples to filter out noisy examples. A novel generative model that directly models the heuristic labeling process of distant supervision was presented in [74]. Min et al. [52] proposed algorithm that learns from only positive and unlabeled data to alleviate the incomplete knowledge base problem. Li et al. [46] applied three heuristics (closest pairs, top trigger words, high-confidence patterns) to reduce the noise in the generated data, and demonstrated the improvement on performance.

Transfer learning is a technique where what has been learned in one model for a task is exploited to improve generalization in another model for a related task [41]. Transfer learning is assumed to be able to improve the performance in a new task through the transfer of knowledge from a related task that has already been learned. It has been applied on many tasks in natural language processing with good effect. In

[42], they demonstrated that neural network model achieved state-of-the-art results on a dataset of Named-Entity Recognition task after utilizing the knowledge that transferred from the model trained on a large labeled dataset. Yang et al. [88] utilized a hierarchical recurrent neural network model trained with plentiful annotations to improve performance on a related sequence tagging task with fewer available annotations. More applications of transfer learning can be found in two survey papers [56] and [84]. In this work, we will transfer the knowledge gained on dataset generated from distant supervision to the human-labeled dataset.

5.2 Methodology

In this chapter, we aim to build better deep learning models with the help of automatically generated data by distant supervision. After acquiring the DS data besides the manually annotated data, we design two methods to combine DS and MA (manually annotated) data: simple union and transfer learning method. In addition, the behavior of transfer learning model with less MA data is a key part of the work, which will provide us insights to build a better model when the MA dataset is small. Thus, the research conducted here can be divided into four parts.

- Part 1: Evaluate the performance of deep learning models on DS-generated data (with and without noise reduction) and manually annotated data separately. The performance of those deep learning models will serve as the baselines of the following experiments.
- Part 2: Consider the methods of combining two types of data (DS and MA data) – simple union and transfer learning, then evaluate deep learning models performance after using combining methods.
- Part 3: This part investigates how the performance of the transfer learning process changes as the amount of MA data is changed. Specifically, we will explore how transfer learning help performance gain of deep learning models with less training MA data.
- Part 4: Verify the effectiveness of transfer learning technique on language model method.

5.2.1 Neural Network Model

In this chapter, we first use two traditional deep learning models to evaluate our method: a CNN-based model called PCNN [89] as well as a RNN-based model called BiLSTM [29] in our experiments. We then verify the effectiveness of our method on the BERT model. The architectures of these three models have been introduced in Chapter 2.

5.2.1.1 Model Input Representation

As introduced in Chapter 3, BERT model uses WordPiece embedding [86] to convert the text into embedding. So, we only discuss the techniques that PCNN and BiLSTM models used for the input representation.

We first represent each word by a word vector and then put all the word vectors in the same order with the sentence as our model input. Also, we find that the model performs better if we include more information about the input than what was included in the original papers [89]. Specifically, we include in addition to the word embedding, POS tag, entity type, relative distance to two entities (see below), and incoming dependency relation in the word representation. The original PCNN model [89] only uses word embedding and positional embedding (relative distance to two entities) to represent each word.

We use the word embedding pre-trained on the PubMed using skip-gram model[13]. The dimension of each word embedding vector is 200. For POS tag and incoming dependency, we extract this information from the parse results of Bllip parser [49] and convert them to unique 10-dimension vectors. The relative distance to entities (to entity 1 (d_1) and to entity 2 (d_2)) is calculated by counting the words between the target word and the entities and the distance will be marked as negative if a word appears at the left side of the entity. After acquiring the distance numbers, we will map each number to unique 5-dimension vector. From the perspective of entity type, all the words in a sentence could be divided into four types: ENTITY1, ENTITY2, ENTITY, O. ENTITY1 and ENTITY2 are the two interacting entities, ENTITY is used for the

other entities in the sentence, and O stands for other words. We use one-hot vector to represent this feature.

5.2.2 Generation of DS data

In this section, we first introduce the technique of distant supervision and then discuss the noise reduction methods we will adopt in our experiments. At last, the statistics of the DS-generated datasets will be given.

5.2.2.1 Distant Supervision

In the natural language processing field, distant supervision has been introduced as a method to automatically annotate datasets. DS assumes that the text (often a sentence) expresses a relation between entities if these entities are found in a database that captures that relationship [53]. Formally, suppose an entity pair $(e1, e2)$ are known to be related by a relation R according to a database and suppose there is a sentence that mention those two entities $e1$ and $e2$ in them. Distant supervision will label such sentences with these two entities as positive instances for relation R (Note that an instance is a sentence with two entities). On the other hand, if two entities are not in the database and there is a sentence containing the mentions of these two entities, the corresponding instance is taken as a negative instance for relation R.

Utilizing this technique, we can automatically obtain large training datasets using the corresponding databases for relations to label the text in literature. Let us take PPI relation as an example, assuming a database includes *RFX5* and *histone deacetylase 2* as an interacting protein pair, then the following sentence with these two entities will be seen as a positive instance for PPI relation: "**RFX5** specifically interacts with **histone deacetylase 2** (HDAC2) and the mammalian transcriptional repressor (mSin3B)". Furthermore, if *RFX5* and *mSin3B* are not included in this database as an interacting pair, then the above sentence with these two entities will be annotated as a negative instance for the PPI relation by the DS method.

We now discuss the knowledge databases used in the data labeling process of distant supervision. For PPI task, we use IntAct database as the interacting protein pairs database, which is a freely available, open source database system for molecular interaction data [55]. As for MIRGENE task, we utilize the TarBase [79] and miRTarBase [30] database to match miRNA-gene regulation relations. We choose UniProt database [5] as our distantly supervised database for protein subcellular localization relation, which is a freely accessible resource of protein sequence and functional information.

We also need biomedical text to generate the distantly labeled data. Medline contains abstracts for biomedical literature from around the world and it is our first choice of text source, we use it for protein subcellular localization and miRNA-gene regulation by randomly sampling 30,000 abstracts that contains at least one pair of protein and subcellular location within one sentence. As it is shown in [46], it gives us a skewed dataset for the PPI task– positive/negative ratio is 1: 7.4. In order to acquire more balanced positive and negative instances for PPI, we just use the literature found in the IntAct database as our text source (Positive:Negative=1:1.5).

Given the required knowledge bases and text source for distant supervision, the last thing we have to consider is the detection of all the entity names (protein, subcellular location and miRNA names) in the biomedical literature. For protein names, we utilize the output of GNormPlus [82], which is an end-to-end system that detect gene/protein names. For the miRNA entities, we use the TarBase [79] and miRTarBase [30] database to recognize miRNA mentions by regular expression in text. For the subcellular location names, we use location names from UniProt as a dictionary to match the mentions in the Medline text.

We then can consider each pair of detected entities in the biomedical text, and label it as a positive instance if this pair appears in the database for a specific relation. Otherwise, a negative instance will be generated by distant supervision.

5.2.2.2 Noise Reduction Heuristics

Distant supervision can give us a large amount of labeled data to train deep learning model, however its labeling process is noisy. It can generate false positive instances since it will always label all the sentences mentioning two entities stored in the known relation database as positive, regardless of the semantic meaning of the sentence. For example, the instance "These data suggest that the functions of the Cul2-**Rbx1** and **Cul5**-Rbx2 modules are distinct." will be wrongly labeled as positive by DS even though there is no relation between the two highlighted entities. Distant supervision might also incorrectly assign instances as negative due to the incompleteness of the relation database. For instance, DS will label the sentence "RFX5 specifically interacts with histone deacetylase 2 (HDAC2) and the mammalian transcriptional repressor (mSin3B), whereas **RFX1** preferably interacts with HDAC1 and **mSin3A**." as negative since the knowledge base does not include the interacting pair of RFX1 and mSin3A, even though the sentence is expressing the PPI relation.

A number of heuristics have been proposed for DS noise reduction. We eventually decided to use the ones chosen in the work of [46] and also the shortest dependency path (SDP) length will be considered as a heuristic rule (similar to the SDP frequency heuristic in [93]). Among these heuristics, Closest Pairs (CP), Trigger Words (TW) and Shortest Dependency Path of length n (SDP_n : n is a threshold parameter) heuristics will be applied on the positive instances and High-confidence Patterns (HP) heuristic will be applied on negative instances.

Closest Pairs: there are many sentences in which one pair of proteins appear more than once, and only one of mentioned relations is true in most cases. This heuristics assumes the pair with the closest distance on dependency path is the interacting one with higher probability. So, only the closest pair is assumed to be positive and the others will be removed from the positive instances. For example, let us consider this sentence "The interaction between **bICP0** and **IRF7** correlates with reduced trans-activation of the IFN-beta promoter by IRF7.", in which protein "IRF7" appears two times. The proteins "bICP0" and "IRF7" are interacting to each other according to the IntAct

database. So two positive instances will be generated, but only the highlighted pair is true positive, and the other pair is wrongly labeled as positive. In this case, after applying the closest pair heuristic, the wrongly-labeled instance will be removed because its entity distance is not the closest on the dependency path.

Trigger Words: typically, a "relation" trigger word is used to express a relation in a sentence. These kind of trigger words are usually verbs, but can sometimes appear in their nominal or adjectival form. For example, "affect" is the trigger word in the sentence "Acanthamoeba **profilin** affects the mechanical properties of **nonlamentousactin**". Using a heuristic to automatically guess the trigger words from text, we can remove all the DS-generated positive instances which have no trigger words in them.

Shortest Dependency Path of length n: shortest dependency path is considered to contain the key information between entities for relation expression [31]. Also, it is found that the SDP lengths between two related entities are relatively short (as it is shown in Figure 5.1). Thus, we could set a threshold for the length of SDP to filter the noise in the DS data: the instances that have the SDP length is greater than threshold n will be filtered out as noise.

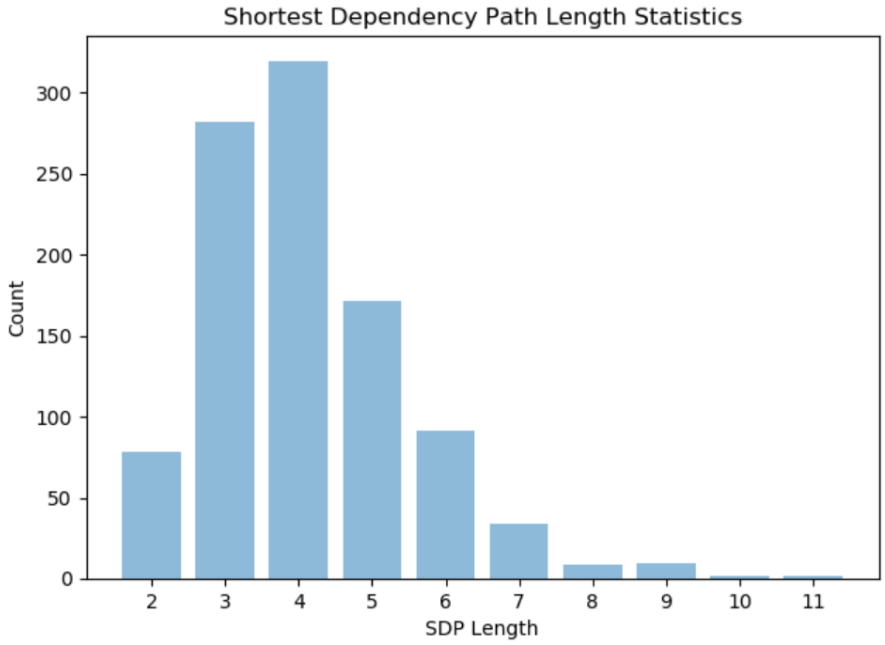


Figure 5.1: Statistics of Shortest Dependency Path Length for AIMed.

High-confidence Pattern: note that the knowledge database does not record all the interacting entity pairs in it, so those interacting entity pairs that are not in the knowledge base will be labeled as negative instances. In order to remove this kind of noise from the negative instances, the high-confidence pattern heuristic is introduced. If some pattern appears repeatedly in many positive instances, we could consider them as high-confidence patterns of expressing a relationship. Applying those high-confidence patterns on the negative instances, we can remove the wrongly labeled negative instances. For example, the sentence "RFX5 specifically interacts with histone deacetylase 2 (HDAC2) and the mammalian transcriptional repressor (mSin3B), whereas **RFX1** preferably interacts with HDAC1 and **mSin3A**." is labeled as negative by DS, since the pair of RFX1 and mSin3A does not exist in the knowledge base. After applying the high-confidence pattern, it will be removed from the negative instances because it follows the pattern "PROTEIN 1 interacts with PROTEIN 2", which is frequently found in the positive instances.

The definition of trigger word is the verb (or its nominal/adjectival form) that expresses the relation between two entities, but the related two entities do not have verbal trigger word in many cases in PLOC task. So we will not apply heuristic TW on positive instances for the PLOC task. Since we have one less heuristic on PLOC DS dataset, we further filter out the noise by choosing the top 20 location names based on their frequency.

Another thing we have to consider is the threshold parameter n in the Shortest Dependency Path of length n heuristic. As we discussed before, the SDP length for related entities is relatively short. As it can be seen in Figure 5.1 that most of the SDP length of positive instances are in range of 2 and 9. Thus, the most reasonable choice for this parameter n is 9 since it keep most of the correctly labeled data. However, the experiment results show that when we try to keep more correctly labeled data (choose a big n), we include more noise at the same time. In order to reduce more noise, we choose the parameter of n from the range between 4 and 6.

5.2.3 DS Corpora Statistics

In this section, the statistics of different DS datasets for each task will be provided. The first row in Table 5.1 shows the original number of positive and negative instances of DS-generated data for our relation extraction tasks. The next few rows show the dataset size after applying different heuristics and their combinations. For the PLOC task, we apply one less heuristics on its DS data (without TW rule).

Dataset	PPI		MIRGENE		PLOC	
	P#	N#	P#	N#	P#	N#
RAW	54,170	82,517	75,632	97,118	19,654	32,254
CP	38,644	82,517	58,159	97,118	15,519	32,254
SDP_4	33,220	82,517	31,814	97,118	14,684	32,254
SDP_5	45,147	82,517	50,371	97,118	19,756	32,254
SDP_6	55,125	82,517	63,365	97,118	23,723	32,254
HP	54,170	77,559	75,632	74,852	19,654	30,206
CP+TW	25,294	82,517	48,370	97,118	-	-
CP+HP	38,644	77,559	58,159	74,852	15,519	30,206
SDP_4+HP	33,220	77,559	31,814	74,852	14,684	30,206
SDP_5+HP	45,147	77,559	50,371	74,852	19,756	30,206
SDP_6+HP	55,125	77,559	63,365	74,852	23,723	30,206
CP+TW+HP	25,294	77,559	48,370	74,852	-	-

Table 5.1: DS data statistics for PPI, MIRGENE and PLOC task. P#: number of positive instances; N#: number of negative instances; RAW: original DS-labeled data without any heuristic; CP: Apply closest pair heuristic on DS data; SDP_n: Apply SDP length of n heuristic on DS data; HP: Apply high-confidence pattern heuristic on DS data; CP+TW: Apply closest pair and trigger word heuristics on DS data; CP+HP: Apply closest pair and high-confidence pattern heuristics on DS data; SDP_n+HP: Apply SDP length of n and HP heuristic on DS data; CP+TW+HP: Apply closest pair, trigger word and high-confidence pattern heuristics on DS data.

5.2.4 Transfer Learning

Recall that we will employ transfer learning to combine the DS-labeled data and manually annotated data. In this section, we will explain why transfer learning is a feasible approach to learn knowledge from two different datasets for same task.

Transfer learning has become a popular approach in deep learning where pre-trained models are reused on related computer vision and natural language processing tasks [69] [73] [50] [60]. In this chapter, we will try to combine two different types of data for the same task: DS data and manually annotated data. Thus, we consider the model trained on DS data as a source model and transfer the gained knowledge to the task which is defined by the MA data. Thus, we consider the underlying tasks defined by DS data and MA data are two different but related tasks. The DS data for the relation extraction tasks are generated without paying any attention to the semantic meaning of text. On the other hand, the MA data are labeled by human based on their understanding of the relation in the text. So we can see that the models trained on these two datasets are gaining (perhaps) slightly different knowledge. Thus, we can take the two models built on DS and MA data as two different models for closely related tasks.

5.2.5 Experiments Conducted

In this section, we will conduct experiments to address the problems regarding the use of DS-generated data to "augment" manually annotated data. In other words, we will focus on the problem of designing methods to combine DS and MA data.

5.2.5.1 Experiment 1: Developing Models Based on DS Data

After acquiring a large amount of DS data, we first investigate the performance of models trained on the automatically derived DS data. Then we also explore the effectiveness of the noise reduction heuristics. In [46], the logistic regression model performs better on noise-reduced DS data (DSNR). We conduct a similar exercise using the noise-reduction heuristics in conjunction with our deep learning models. Specifically, we train the deep learning model on DS-obtained dataset, test the model on manually annotated data and then apply the same process on datasets obtained after applying different heuristics.

5.2.5.2 Experiment 2: Data Combining Methods

As we have large DS-labeled datasets available, an obvious question is to consider how to combine it with the manually annotated dataset. The most straightforward way to combine these two datasets is to simply take the union of DS data and MA data, and we will use it as a baseline here. We will also employ transfer learning to combine DS-obtained and MA data by pre-training the model on (noise-reduced) DS datasets, then fine tuning the model on MA training set to further adjust the parameters of the model.

In the pre-trained model, the learned knowledge from DS data stores in the hidden layers' weights. These weights mean convolution filter (feature map) weights and the fully connected layer weights for CNN model, meanwhile mean recurrent cell weights and the fully connected layer weights in RNN model. Since the fully connected layer weights play the role of classifying the label of instance based on acquired features in theory, convolution filter weights and recurrent cell weights contain the most important information learned from the pre-training data. We do not eliminate fully connected layer weights directly, since their functionality is not well studied. Instead, we design two options for transfer learning: 1). only transfer the convolution filter weights/recurrent cell weights; 2). transfer both convolution filter weights/recurrent cell weights and fully connected layer weights.

Figure 5.2 shows the pipeline of transfer learning model. When we use manually annotated data in both training and test process, we will perform cross validation to obtain the final results.

5.2.5.3 Experiment 3: Impact of Size of MA Training Data

The motivation for distant supervision is to obtain improved performance when there is only a limited amount of human-labeled data. Thus, it is worth examining the impact of the size of manually annotated data on the model performance. For this set of experiments, we obtain transfer learning models pretrained on DS data and then use different sizes of manually annotated data to evaluate the dataset size effect.

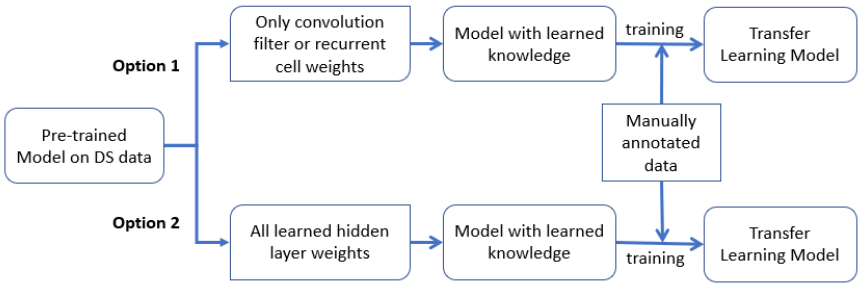


Figure 5.2: Pipeline of transfer learning model on both DS data and human-labeled data.

Specially, we will utilize 25%, 50% and 75% of the manually annotated data in the transfer learning training process to evaluate the performance of models.

5.2.5.4 Experiment 4: Applying Transfer Learning on Language Model Method

We have shown that the task-related text helps the language model generalization on specific tasks in Chapter 3. The labeled data generated by distant supervision should also be highly related with the relation extraction tasks. The entity-related text used in Chapter 3 is utilized in the pre-training stage of BERT model. Differently, we can utilized the DS-generated data in the fine-tuning stage since we have labels for them. In this experiment, we will first fine-tune the BERT model using DS data and then further fine-tune it with the human-labeled data to verify the effectiveness of transfer learning on the BERT model. Considering the size of dataset for PLOC might be too small for the fine-tuning of BERT model, we will only apply transfer learning (using BERT model) on the PPI and MIRGENE tasks.

5.2.6 Evaluation Sets

AIMed [7] will be our evaluation set for PPI. We will use the MIRGENE set for miRNA-gene regulation task [45]. For PLOC task, we will use LocText corpus [11] as our evaluation set. Please see Table 2.1 in Chapter 2 for the statistics of these three corpora.

5.2.7 Parameter Choice

For the PCNN and BiLSTM model, the maximum length of sentence is also set to 100, which mean the longer sentences are pruned and the shorter sentences are padded with zeros. The batch size is 128 during model training. The convolution filter number is 400 and the learning rate is 0.001 for PCNN model. Also, we apply decayed learning rate on PCNN with 0.95 decay rate and 1000 decay steps. For BiLSTM, we set the hidden state dimension to be 400 and utilize constant learning rate of 0.001. We also apply dropout in these two models with drop rate of 0.5 on convolution/recurrent layer(s), and drop rate of 0.2 on dense layers. The training epoch of transfer learning on MA data is 200 for PCNN and 100 for BiLSTM (BiLSTM is trained with less epochs since it needs more time to train). Plus, the window size for PCNN is 3.

For the fine-tuning of the BioBERT models, we use the learning rate of 2e-5, batch size of 32, and max sequence length of 128. During the fine-tuning of PubMedBERT models, the learning rate of 2e-5, batch size of 16, and max sequence length of 256 are utilized. For the fine-tuning of BERT models, the training epoch is 2 on DS-generated data and it is 10 on human-labeled data.

5.3 Results and Discussion

Throughout this section, we use precision, recall, F1 score as measurement to evaluate the performance of deep learning models.

5.3.1 Model Built on DS Data

In the previous section, we first generate a large amount of labeled data utilizing the technique of distant supervision and then reduce the noise in the DS data using some heuristics. Then we build deep learning models based on both the raw DS data and noise-reduced DS data. The model built on noise-reduced DS data should achieve better performance as the heuristics on positive and negative instances will improve the precision and recall respectively. The noise in positive instances will make the model predict the negative instances as positive, so removing noise in positive instances should

bring false positive rate down, i.e., precision should improve. The noise in negative instances will lead the model to predict the positive instances as negative, so reducing noise in negative instances should make false negative rate decrease, i.e., recall should increase.

The experiment results using PCNN model show the effectiveness of the heuristics in improving the performance. Table 5.2 shows the results of model performance on the three tasks. As is to be expected, precision improves with the use of heuristic on positive instances and the heuristic on negative instances makes the recall better. For example, the application of CP and TW causes the expected increase in precision in the PPI task. As expected, the addition of HP boosts the recall in all cases.

Dataset	PPI			MIRGENE			PLOC		
	P	R	F	P	R	F	P	R	F
RAW	34.0	65.2	44.7	55.1	60.3	57.6	57.4	79.8	66.7
CP	56.1	48.6	52.1	67.5	39.4	49.8	66.6	67.5	67.0
SDP_4	52.6	31.8	39.6	68.5	24.8	36.4	80.2	43.9	56.7
SDP_5	52.0	50.7	51.3	64.2	43.8	52.1	67.8	70.9	69.4
SDP_6	43.8	47.4	45.5	62.5	64.2	63.3	57.4	76.9	65.8
CP+TW	57.5	52.9	55.1	72.5	41.4	52.7	-	-	-
CP+HP	50.2	63.7	56.2	65.3	73.9	69.4	68.6	75.2	71.7
SDP_4+HP	47.0	48.0	47.5	66.9	45.3	54.0	77.8	45.9	57.7
SDP_5+HP	41.5	66.4	51.1	65.1	69.2	67.1	69.2	69.2	69.2
SDP_6+HP	39.1	71.5	50.6	61.4	77.6	68.6	60.7	84.6	70.7
CP+TW+HP	56.2	62.3	59.1	67.3	73.1	70.0	-	-	-

Table 5.2: PCNN model performance on DS data for PPI, MIGENE and PLOC task.

In addition, we see that the addition of these noise-reduction heuristics helps boost the performance in most cases, which indicates that different heuristics help reduce different types of noise to make the DS data cleaner.

Even though all heuristics show their effectiveness of reducing the noise in the DS data, we find that the heuristic CP + TW perform better than the heuristic SDP_n on positive instance noise reduction. Also, the combination CP + TW + HP (CP + HP for PLOC task) is proven to be the best way to reduce the noise in DS data for all

the tasks. Thus, in the following experiments of this chapter, we will use CP, TW and HP as our default way for noise reduction.

5.3.2 Combining DS and MA Data

This subsection is concerned with the core question of this work: how much improvement can we obtain by augmenting manually annotated data with (noise reduced) DS data. Table 5.3 and 5.4 show the results of various models. The first row in both tables is not based on the use of the combined data but instead repeat the results from previous experiment and provide the context for the new results using combined training data. The first row corresponds to the model $Model_{DSNR}$ obtained by training on noise-reduced DS data, whereas the second row corresponds to the model $Model_{MA}$ obtained by training on purely manually annotated data. As mentioned earlier, the pure data combination is the simplest way to utilize these two datasets, and hence we first consider the union of human-labeled data and (noise reduced) DS data. The third row of Table 5.3 and 5.4 shows the performance of the resulting trained models, designated $Model_{Mix}$. The drop in the performance observed by comparing the second row suggests that simply taking the union of the instances on the two data sets may not be an appropriate way of augmenting the manually annotated data. Both precision and recall drop in all four cases. We hypothesize that the drop in performance might be due to some remaining noise in the DS data and/or that there might be some additional constraints in the manual annotation guidelines that might not be captured in the DS data.

Another way to combine the DS and human-labeled data is to use those pre-trained models as initial points, then further train the neural network model on manually annotated dataset, i.e. transfer learning. We have explored two options for transfer learning: 1). $TL_{CON/REC}$: transfer learning with only convolution filter weights or recurrent cell weights; 2). TL_{ALL}: transfer learning with all the weights of pre-trained model. The performance of these models trained in these manners is also shown in Table 5.3 and Table 5.4.

Model	PCNN			BiLSTM		
	Precision	Recall	F score	Precision	Recall	F score
$Model_{DSNR}$	49.7	66.3	56.8	50.7	50.1	50.4
$Model_{MA}$	75.6	76.1	75.8	78.1	69.7	73.7
$Model_{Mix}$	65.0	68.2	66.5	64.2	54.9	59.2
$Model_{TL\_CON}$	76.7	79.3	78.0	78.3	74.5	76.4
$Model_{TL\_ALL}$	77.1	81.2	**79.1**	78.9	74.7	**76.8**

Table 5.3: Results of deep learning models on PPI. $Model_{DSNR}$: model built on noise-reduced DS data; $Model_{MA}$: model built on manually annotated data (AIMed for PPI); $Model_{TL\_CON}$: transfer learning using only the convolutional features; $Model_{TL\_REC}$: transfer learning using only the recurrent cell features; $Model_{TL\_ALL}$: transfer learning using all the pretrained parameters. 10-fold cross validation is performed in these experiments.

Model	PCNN			BiLSTM		
	Precision	Recall	F score	Precision	Recall	F score
$Model_{DSNR}$	65.9	46.5	54.5	63.5	57.6	60.4
$Model_{MA}$	74.1	83.8	78.6	76.0	70.5	73.2
$Model_{Mix}$	71.4	69.9	70.6	72.7	63.7	67.9
$Model_{TL\_CON}$	75.3	82.1	78.6	79.2	78.3	78.8
$Model_{TL\_ALL}$	76.1	84.9	**80.3**	80.3	78.4	**79.4**

Table 5.4: Results of deep learning models on PLOC.

These two tables shows that both transfer learning models perform better than the models built on DS-labeled data as well as human-labeled dataset (AIMed and LocText). In fact, as hoped, the performance exceeds that of all other models, and obtains the best results ever. This implies that the deep learning models learn the knowledge in both DS and human-labeled data, and even though there may still be noise in DS data, the transfer learning process utilizes the human-labeled data to remedy the mistakes before and lead the learning in right direction in the second phase of model training. Thus, transfer learning is an effective way to make the best of DS labeled data and limited human-annotated data.

For the two options of transfer learning, we notice that the way of transferring all weights of pre-trained model obtains slightly better results overall. Thus, transferring

all weights is our default way of transfer learning in our following experiments.

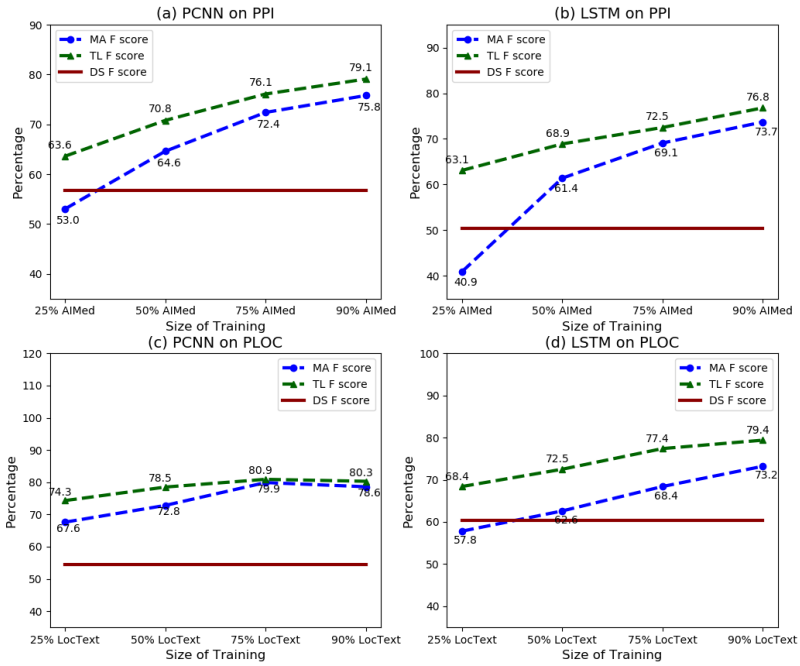


Figure 5.3: Trend of F score with different size MA data in transfer learning. MA F score means the F score acquired from models built on MA data only; TL F score means the F score acquired from models built on transfer learning; DS F score means the F score acquired from models built on DS data.

5.3.3 Effect of Human-labeled Dataset Size

Any potential gains of the data augmentation method are more meaningful when the amount of available human-labeled dataset is not large. However, this is also a situation where any noise in DS derived data discrepancy between it and human-labeled data might hamper the effectiveness of data augmentation with DS data. This motivated the third set of experiments where we use noise-reduced DS data (and transfer learning) in conjunction with 25%, 50%, 75% of the human-labeled data in the model training process.

Figure 5.3 shows the F1 score corresponding to different sizes of human-annotated data, where the 90% case corresponds to the results from previous subsection. The performance of the model obtained using transfer learning is shown and compared

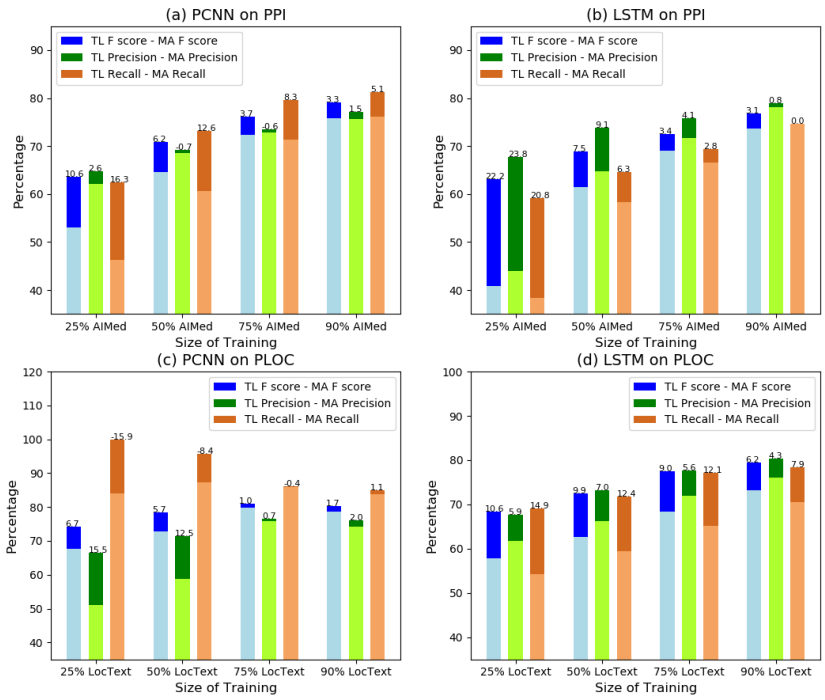


Figure 5.4: Size effect of human-labeled dataset. The number on each bar stands for the difference between None Transfer Learning and Transfer Learning model. Positive number means Transfer Learning improves the metric, while negative number means Transfer Learning deteriorates the metric.

with those obtained with just the human-annotated (of the same size) data and with DS data. For example, training on 25% of AIMed data on the PPI task, the transfer learning method enables us to improve the performance by 10.6% and 22.2% using PCNN and BiLSTM respectively over the models of training on corresponding size of human-labeled data alone. We believe this shows reasonably good performance can be achieved with just 25% of manually labeled data using transfer learning, especially compared to using manually labeled data alone. Notice that with 25% of the data, the performance of the model trained on manually labeled data is worse than the model trained using DS data alone in three cases out of four. The improvement using transfer learning narrows as the size of the human-labeled data increases. Improvement is also seen on the PLOC data, although the improvement is less than what was obtained for the PPI task. These results show that transfer learning and data augmentation

Model	AIMed			MIRGENE		
	P	R	F	P	R	F
BioBERT	79.0	83.3	81.0	89.7	91.3	90.4
BioBERT_TL	83.1	85.1	84.1	92.0	95.5	93.7
PubMedBERT	80.1	84.3	82.1	90.8	94.0	92.2
PubMedBERT_TL	81.7	86.7	84.1	931.	95.3	94.1

Table 5.5: BERT model performance using transfer learning technique on DS data. BioBERT/PubMedBERT_TL: the BERT model that is first fine-tuned on DS-generated data and then further fine-tuned on the human-labeled data.

approach always improves over the training on manual data alone, with the larger improvement shown when the size of human-labeled data is smaller, i.e., when there are limited human-labeled data, a situation which motivates this work.

Figure 5.4 additionally presents the precision and recall numbers for more detailed analysis. For the PPI task with smaller amount of human-labeled data, most of the gains of transfer learning over just human-labeled data training are due to improvement in recall, although for BiLSTM-based model, the gains in precision are also substantially resulting in higher F1 score gain. With PLOC case, the gains in precision and recall are noticed.

5.3.4 BERT Model Performance with Transfer Learning

Recall that in Section 5.2, we discussed that we only evaluate the BERT model on the PPI and MIRGENE tasks. In Table 5.5, we show the BERT model performance before and after utilizing transfer learning technique to first learn the knowledge in the DS-generated data. The first and third rows in Table 5.5 show the results of original BioBERT and PubMedBERT model, which serve as the baselines to compare with the new results. Compared to the baselines, we observe the improvements of 3.2% and 2% on the BioBERT and PubMedBERT models, respectively. The performance improvements indicate that the BERT models can also benefit from transfer learning even though they already have domain knowledge gained in the pre-training step.

Model	AIMed		
	P	R	F
BioBERT_TL	83.1	85.1	84.1
BioBERT_Sub_SLL_Att_TL	83.5	85.5	84.4
PubMedBERT_TL	81.7	86.7	84.1
PubMedBERT_Sub_SLL_Att_TL	83.6	87.9	85.6

Table 5.6: BERT model performance after combining transfer learning technique with previous methods. BioBERT/PubMedBERT_TL: the BERT model with transfer learning. BioBERT/PubMedBERT_Sub_SLL_Att_TL: the BERT model with sub-domain adaptation, SLL fine-tuning and transfer learning.

In addition, the performance of language model with transfer learning is still the best on the PPI task if we put it in a broader context (compared with the results in Chapter 3 and 4). From this perspective, we can say that the generated data from distant supervision can provide more useful information than the data for sub-domain adaptation on the related task.

5.3.5 Combining transfer learning with previous methods

In Chapter 3 and 4, we proposed two independent methods to improve the pre-training and fine-tuning of BERT model. Also, the experiment results demonstrated that the combination of these two methods can further improve the model performance in Table 4.2. In this section, we consider combining the previously proposed methods with the transfer learning method. Specifically, we utilize the sub-domain adaptation (proposed in Chapter 3) in the pre-training of BERT model. Next we employ the SLL fine-tuning architecture proposed in Chapter 4 to fine-tune the BERT model on DS-generated data first and then further fine-tune the BERT model on the human-labeled data.

The experiment results on the AIMed corpus can be found in Table 5.6. The first row for BioBERT and PubMedBERT is just the repetition of results from Table 5.5 and provides the baselines for the results using combined techniques. As shown in Table 5.6,

combining the those three methods can further boost the model performance. Especially, the combined method obtains a F1 score improvement of 1.5% on PubMedBERT model.

5.4 Summary

In order to improve the performance of deep learning models on small datasets, we have considered augmenting them with automatically obtained datasets using distant supervision. We show that some heuristics can be used to alleviate the well-known noisy annotation issue with distant supervision. Improvement of performance of PCNN model on three tasks is obtained. We also illustrate that the PCNN model built on only DS data does not show improvement over the model using human-labeled data alone.

Also, two methods of utilizing both DS data and manual data are discussed. Mixing DS data and human-labeled data to obtain the training data for deep learning model is the simplest way to combine data, but the performance does not show improvement over using human-labeled data alone. We show that the mechanism of transfer learning provides much better results than either of these two types of data individually.

In addition, we explore the feasibility of reducing the size of manual data with the availability of large DS dataset. It can be seen that impact of transfer learning is much more beneficial when the manual data size is small (F score increased 10.6% when using 25% of AIMed). So when developing large human-labeled dataset is not feasible, applying transfer learning on DS data becomes more important.

Furthermore, better results are also obtained after we apply the transfer learning technique on the BERT model. In comparison with the models in previous chapters, we observe the best performance on the AIMed dataset after applying transfer learning on the BERT model. In the future, we will continue to pursue other heuristics to further reduce the noise in the automatic corpus creation with DS. Given the imbalance in the distribution of positive/negative instances in these datasets, we plan to conduct additional research to address this issue.

Chapter 6

IMPROVING BERT MODEL USING CONTRASTIVE LEARNING FOR BIOMEDICAL RELATION EXTRACTION

Contrastive learning has shown great promise in computer vision, achieving state-of-the-art results in recent years [25, 12]. Technically, contrastive learning is to learn general representation from the augmented data of a dataset, where the augmented data are generated by transforming the original data to variants without changing the key information in the data. During the training of contrastive learning, it learns the representation by comparing among the pairs of augmented data and encoding the similarity and dissimilarity of them [34]. As a result, the similar examples will have similar representations while the dissimilar examples will have dissimilar ones.

Because of the discrete characteristics of the text data, contrastive learning is not well studied in natural language processing (NLP) field. It is much more difficult to design data augmentation method to construct similar text data. In this chapter, we propose a novel technique for constructing similar text. Specifically, we use the shortest dependency path (SDP) [8] between two entities when we try to augment the dataset of relation extraction tasks for the contrastive learning. We assume that the words on SDP capture all the knowledge for the relation expression, so we keep it unperturbed during the data augmentation. Then we can randomly change the words outside the SDP to generate similar data for contrastive learning. After acquiring similar data for relation extraction task, we need large datasets for contrastive learning to achieve better generalization. In this chapter, we employ external knowledge bases to construct more data during contrastive training.

To verify the effectiveness of the proposed method, we use the transformer-based BERT model as a backbone [17] and evaluate our method on three RE tasks: the

chemical-protein interactions (ChemProt) [38], the drug-drug interactions (DDI) [26], and the protein-protein interactions (PPI) [37]. The experiment results show that our method boosts the BERT model performance on all three tasks.

Interest has also grown in designing interpretable BioNLP models that are both plausible (accurate) and rely on a specific part of the input (faithful rationales) [18, 44]. Here rationale is defined as the supporting evidence in the inputs for the model to make correct predictions. In this direction, we propose a new metric, "prediction shift", to measure the sensitivity degree to which the small changes (out of the SDP) of the inputs will make model change its predictions. We show that the contrastively pre-trained model is more robust than the original model, suggesting that our model is more likely to make predictions based on the rationales of the inputs.

The rest of this chapter is organized as follows. In Section 6.1, we will discuss the related work of contrastive learning. The framework of contrastive learning will be presented in Section 6.2, which is followed by the experiment design in Section 6.3. In Section 6.4, we will discuss the results and discussion. We will summarize the results of the research in the last section.

6.1 Related Work

The history of contrastive representation learning can be traced back to [24], in which the authors explore the method of representation learning that similar inputs are mapped to nearby points in the representation space. Recently, with the development of data augmentation techniques, deep neural network architectures, contrastive learning regains attention and achieves superior performance on visual representation learning [25, 12]. In [25], the Momentum Contrast (MoCo) framework was designed to learn representation using the mechanism of dictionary look-up: an encoded example (the query) should be similar to its matching key (augmented sample from the same data example) and dissimilar to others. In [12], the authors proposed the SimCLR frame to learn the representations by maximizing the agreement between augmented views of the same data point.

The contrastive representation has all the properties that a good representation should have: 1) Distributed property; 2) Abstraction and invariant property; 3) Disentangled representation [4, 40]. The distributed property emphasizes the expressive aspect of the representation (different data points should have distinguishable representations). The capture of abstract concepts and the invariance to small and local changes are concerned in the abstraction and invariant property. From the disentangled representation's perspective, it should encode as much information as possible. In this chapter, we will show contrastive learning can improve the invariant aspect of the representation.

In the natural language processing (NLP) field, several works have utilized the contrastive learning technique. In [19], the authors proposed a pre-trained language representation model (CERT) using contrastive learning at the sentence level to benefit the language understanding tasks. The proposed method in [36] employed contrastive self-supervised learning to solve the commonsense reasoning problem. In [57], the authors proposed a self-supervised pre-training framework for relation extraction to explore the encoded information for the textual context and entity type. Compared with the previous works, we employ different data augmentation techniques and utilize data from external knowledge bases in contrastive learning to improve the model for relation extraction tasks.

In recent years, there is increasing interest in designing more interpretable NLP models that reveal the logic behind model predictions. In [18], multiple datasets of rationales (from human experts) were collected to facilitate the research on interpretable models in NLP. In [44], the authors proposed an encoder-generator framework to automatically generate candidate rationales to justify the predictions of neural network models.

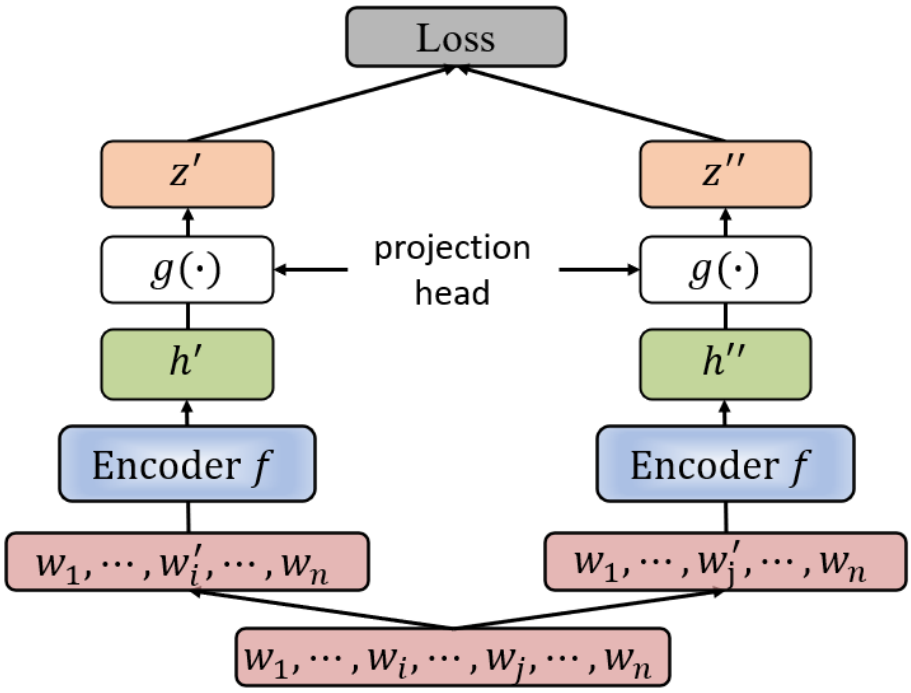


Figure 6.1: The framework of contrastive learning. For the data augmentation of relation extraction, we randomly replace some words that are not affecting the relation expression ($w_i \rightarrow w_i^{'}$ in the left sample, $w_j \rightarrow w_j^{'}$ in the right sample).

6.2 Methodology

6.2.1 The framework of contrastive learning

As we mentioned previously, the goal of contrastive learning is to learn a text representation by making the similar inputs have similar representations and the dissimilar inputs have dissimilar representations. Specifically, contrastive learning aims to maximize the agreement between the representations of similar inputs (we call them positive pairs) via a contrastive loss in the latent space. The positive pairs are generated by utilizing the data augmentation technique to acquire slightly different examples with the original data. After the training of contrastive learning, the learned representation can then be used for our relation extraction tasks.

Figure 6.1 shows our framework of contrastive learning. Given a sentence $s = w_1, ...w_n$, we first produce two augmented views (a positive pair) $v' = w_1, ..., w_i', ...w_n$ and $v'' = w_1..., w_j', ...w_n$ $(i \neq j)$ from s by applying text augmentation technique

Original	We further show that @PROTEIN$ directly interacts with @PROTEIN$ and Rpn4.
After SR	We further show that @PROTEIN$ **straight** interacts with @PROTEIN$ and Rpn4.
After RS	**Further we** show that @PROTEIN$ directly interacts with @PROTEIN$ and Rpn4.
After RD	We further show that @PROTEIN$ interacts with @PROTEIN$ and Rpn4.

Table 6.1: Examples after the three operations for data augmentation. The shortest dependency path between two proteins is "@PROTEIN$ interacts @PROTEIN$", which is marked with underline in the examples. The changed words are also marked with bold font.

(Section 6.2.1.1). Our framework then uses one neural network to encode the two inputs, which consists of a neural network encoder f (Section 6.2.1.2) and a projection head g (Section 6.2.1.3). From the first augmented view v', we output a *representation* $h' \triangleq f(v')$ and a projection $z' \triangleq g(h')$. From the second augmented view v'', we output $h'' \triangleq f(v'')$ and another projection $z'' \triangleq g(h'')$.

The contrastive loss will be calculated based on the projections z' and z'' (Section 6.2.1.4). If z' and z'' are from two similar examples, the loss will be small, otherwise the loss will be large. During the training, the representations (h' and h'') are learned by leading the positive pairs (similar inputs) to have similar representations and making negative pairs (dissimilar inputs) have dissimilar representations. In the applications of contrastive learning, the positive pairs are usually from the augmented data of the same sample, and the negative pairs are generated by selecting augmented data from different samples.

At the end of training, we only keep the encoder f as in [12]. For any text input x, $h = f(x)$ will be the representation of x from contrastive learning.

6.2.1.1 Data augmentation for relation extraction

The data augmentation module is a key component of contrastive learning, which needs to randomly generate two correlated views for the original data point. At

the same time, the generated data should be different from each other to make them distinguishable (from the model's perspective), but should not be significantly different to change the structure and semantics of the original data. It is especially difficult to augment the text data of relation extraction. In this chapter, we only focus on binary relations. Given $< s, e_1, e_2, r >$, where e_1 and e_2 are two entity mentions in the sentence s with the relation type r, we keep e_1 and e_2 in the sentence and retain the relation expression between e_1 and e_2 in the augmented views.

Specifically, we propose a data augmentation method utilizing the shortest dependency path (SDP) between the two entities in the text. We hypothesize that the shortest dependency path captures the required information to assert the relationship of the two entities [8]. Therefore we fix the shortest dependency path, and randomly change the other tokens in the text to generate the augmented data. This idea is inspired by [83], which employed easy data augmentation techniques to improve model performance on text classification tasks.

As the preliminary study, we experiment with three techniques to randomly replace the tokens to generate the augmented data and choose the best one for our contrastive learning method: 1) Synonym replacement (SR), 2) Random swap (RS), and 3) Random deletion (RD).

Table 6.1 gives some samples after applying the three operations on a sentence from the PPI task. For the synonym replacement, we randomly replace n words with their synonyms. To acquire the synonym of a word, we utilize the WordNet database [51] to extract a list of synonyms and randomly choose one from the list. For the random swap, we swap the positions of two words and repeat this operation n times. For the random deletion, we delete some words with the probability p. The probability p is set to 0.1 in our experiments and the parameter n for SR and RS is calculated by $p \times l$, where l is the length of the sentence.

To examine which operation performs better for relation extraction tasks, we train three BERT models using the three types of augmented data (combined with the original training data). Table 6.4 shows that the synonym replacement (SR) operation

achieves the best performance on all three tasks and we will employ this operation in our data augmentation module in our contrastive learning experiments (We will further discuss it in Section 6.4.2).

6.2.1.2 The neural network encoder

In this chapter, we employ the BERT model [17] as our encoder for the text data and the classification token ([CLS]) output in the last layer will be the representation of the input.

6.2.1.3 Projection head

As demonstrated in [12], adding a nonlinear projection head on the model output will improve the representation quality during training. Following the same idea, a multi-layer perceptron (MLP) will be applied to the model output h. Formally,

$$z = g(h) = W^2\phi(W^1 h)$$

and ϕ is the ReLU activation function, W^1 and W^2 are the weights of the perceptron in the hidden layers.

6.2.1.4 Contrastive loss

Contrastive learning is designed to make similar representations be learned for similar inputs (a.k.a., the augmented samples from the same data point). During contrastive learning, the contrastive loss is calculated based on the augmented batch derived from the original batch. Given N sentences in a batch, we first employ the data augmentation technique to generate two views for each sentence in the batch. Therefore, we have $2N$ views in the augmented batch. If two views are generated from the same sentence, we treat this pair as a positive pair, otherwise it will be a negative pair. We follow the work of [12] to design the loss function for a positive pair:

$$l(z', z'') = -log\frac{exp(sim(z', z'')/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[z_k \neq z']} exp(sim(z', z_k)/\tau)}$$

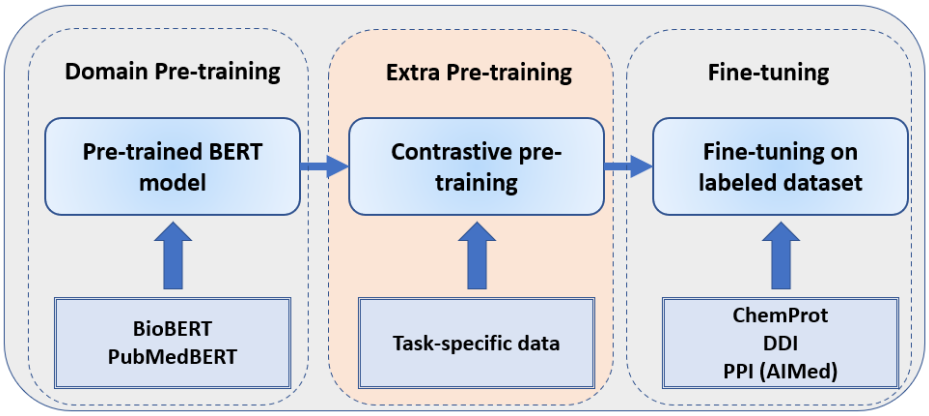


Figure 6.2: The pipeline of BERT model training with contrastive pre-training.

where $sim(\cdot, \cdot)$ is the cosine similarity function, $\mathbb{1}_{[z_k \neq z']}$ is the indicator function and τ is the temperature parameter. The final loss L is computed across all positive pairs, both (z', z'') and (z'', z'), in a batch.

For computation convenience, we arrange the $(2k-1)$-th example and the $2k$-th example in the augmented batch are generated from the same sentence, a.k.a., $(2k-1, 2k)$ is a positive pair. Please see Algorithm 1 for the calculation of the contrastive loss in one batch. Then we can update the parameters of the BERT model and projection head g to minimize the loss L.

Algorithm 1: Contrastive loss in a batch

Input: encoder f (BERT), project head g, data augmentation module, data batch $\{s_k\}_{k=1}^{N}$;

for *k=1,...,N* **do**

$v', v'' = data\_augment(s_k)$;

$z_{2k-1} = g(f(v'))$;

$z_{2k} = g(f(v''))$;

end

$L = \frac{1}{2N} \sum_{k=1}^{N} [l(z_{2k-1}, z_{2k}) + l(z_{2k}, z_{2k-1})]$

6.2.2 Training procedure of contrastive learning

Theoretically, contrastive learning can learn the representation of the input from scratch with the use of the augmented data. However, we already have good

representation available from language model (e.g., BERT) for our biomedical NLP tasks. Thus, instead of learning representation from scratch, we try to improve the representation from BERT model here. Figure 6.2 shows the training procedure of our framework. It consists of three stages. First, we load the pre-trained BERT model for a specific domain(e.g., biomedical domain). In our experiments, we use two versions of pre-trained BERT model for the biomedical domain: BioBERT [43] and PubMedBERT [22]. Second, we conduct contrastive pre-training on task-specific data as a continual pre-training step after the domain pre-training of BERT model. In this way, we retain the learned knowledge from general pre-training, and add the new features from contrastive learning. Finally, we fine-tune the model on the RE tasks to further gain task-specific knowledge through supervised training on the labeled datasets.

6.2.3 A knowledge-based method to enrich training dataset for contrastive learning

To learn a well-generalized representation, we usually need large datasets for the training. Currently, we only have the human-labeled datasets available for our relation extraction tasks. In order to acquire large-scale datasets for our contrastive learning, we utilize external databases for the relations to acquire extra training instances. Furthermore, the databases can provide the entity information for relation since we need it to construct the augmented instances using the SDP between two entity mentions in the text. This method of generating data under the guidance of a database is similar to distant supervision [53], but the labels are not used in our contrastive learning framework.

Formally, assuming a curated database for relation r contains all the relevant entities and text, we consider every combination of the entity pairs in one sentence and use them as examples for this relation. For instance, there are three proteins in the sentence s: "Thus NIPP1 works as a molecular sensor for PP1 to recognize phosphorylated Sap155." We will generate three examples for PPI task from this sentence: $<$ s,NIPP1,PP1,PPI$>$, $<$ s,NIPP1,Sap155,PPI$>$ and $<$ s,PP1,Sap155,PPI$>$.

Task	ChemProt	DDI	PPI
EK	35,500	67,959	97,853

Table 6.2: Statistics of datasets generated by external knowledge bases for contrastive pre-training.

Model	ChemProt			DDI			PPI		
	P	R	F	P	R	F	P	R	F
BioBERT	74.3	**76.3**	75.3	79.9	78.1	79.0	79.0	**83.3**	81.0
BioBERT+CL	**77.0**	74.7	75.8	82.6	77.4	79.9	79.8	83.1	81.3
BioBERT+CLEK	76.6	76.0	**76.3**	**82.9**	**78.4**	**80.6**	**81.1**	83.2	**82.1**
PubMedBERT	78.8	75.9	77.3	82.6	81.9	82.3	**80.1**	84.3	82.1
PubMedBERT+CL	79.6	76.2	77.8	**83.3**	81.5	82.4	79.4	85.6	82.4
PubMedBERT+CLEK	**80.6**	**76.9**	**78.7**	**83.3**	**82.4**	**82.9**	79.9	**85.7**	**82.7**

Table 6.3: BERT model performance on ChemProt, DDI and PPI tasks. BioBERT/PubMedBERT: original BERT model; BioBERT/PubMedBERT+CL: BioBERT/PubMedBERT with contrastive pre-training on the training set of human-labeled dataset; BioBERT/PubMedBERT+CLEK: BioBERT/PubMedBERT with contrastive pre-training on the data from the external knowledge base.

In our experiments, we use the IntAct database [54] as the external database for the PPI task. Similarly, DrugBank [85] and BioGRID [71] are utilized for DDI and ChemProt, respectively. In Table 6.2, we show the statistics of datasets for each task generated by external knowledge bases. We can see that the datasets from the external database are much larger than that of the human-labeled datasets in the Table 2.1 of Chapter 2.

6.3 Experiments

As discussed before, we will utilize the BERT model as the encoder for the inputs. In particular, we will employ two BERT models pre-trained for the biomedical domain in our experiments: BioBERT [43] and PubMedBERT [22].

6.3.1 Datasets and evaluation metrics

We will evaluate our method on three benchmark datasets: ChemProt, DDI, and PPI (AIMed). The statistics of these datasets is shown in Table 2.1. PPI is a binary classification problem, and we will use the standard precision (P), recall (R) and F1-score (F) to measure the model performance. However, the ChemProt and DDI tasks are multi-class classification problems. The models for ChemProt and DDI will be evaluated utilizing micro precision, recall and F1 score on the non-negative classes.

6.3.2 Data pre-processing

Following the same convention with previous chapters, we replace the relevant entity names with predefined tags: protein names are replaced with @PROTEIN$, drug names with @DRUG$, and chemical names with @CHEMICAL$. In Table 6.1, we show a pre-processed example of the PPI task.

6.3.3 Training setup

For the fine-tuning of the BioBERT models, we use the learning rate of 2e-5, batch size of 16, training epoch of 10, and max sequence length of 128. During the fine-tuning of PubMedBERT models, the learning rate of 2e-5, batch size of 8, training epoch of 10 and max sequence length of 256 are utilized.

In the contrastive pre-training step of the BERT models, we use the same learning rate with the fine-tuning, and the training epoch is selected from [2, 4, 6, 8, 10] based on the performance on the development set. If there is no development set (e.g., PPI task), we will use 6 as the default training epoch. Since contrastive learning benefits more from larger batch [12], we utilize the batch size of 256 and 128 for BioBERT and PubMedBERT respectively. In addition, the temperature parameter τ is set to 0.1 during the training.

6.4 Results and discussion

6.4.1 BERT model performance with contrastive pre-training

Table 6.3 demonstrates the experiment results using the BERT models with contrastive pre-training and external datasets. The first row is the BioBERT model performance without applying contrastive learning. The following two rows demonstrate the results after adding the contrastive pre-training step in BioBERT. The "BioBERT+CL" stands for the BioBERT model with contrastive pre-training on the training set of the human-labeled dataset, while "BioBERT+CLEK" is for the BioBERT model with contrastive pre-training on the data from the external knowledge base. Similarly, we give the PubMedBERT model performance of our method in the last three rows of Table 6.3.

We can see that the contrastive per-training improves the model performance in both cases. However, contrastive pre-training on human-labeled dataset only improves the model with a small margin. We hypothesize that the limited improvement might be due to the poor generalization on small training set. Therefore, we include more data (EK data) in contrastive learning to enhance the model generalizability. The data generated from the external knowledge base are much more than the training data of the human-labeled dataset (column "EK" and "train" in Table 6.2). As shown in the third and sixth row in Table 6.3, contrastive learning with more external data can further boost the model performance. Compared with the BERT models without contrastive pre-training, we observe an averaged F1 score improvement (on the two BERT models) of 1.2%, 1.2%, and 0.85% on ChemProt, DDI, and PPI datasets, respectively.

6.4.2 Comparison of data augmentation techniques

Table 6.4 shows the BERT model performance after including three types of augmented data. We can see that the synonym replacement (SR) operation yields the best results on all three tasks. Therefore we use it as our default operation to generate augmented data in all our contrastive learning experiments. We also notice that the augmented data from the random swap (RS) operation hurt the model performance

Training data	ChemProt	DDI	PPI
Original	75.3	79.0	81.0
+RS	75.6	78.4	75.4
+RD	75.4	79.8	81.2
+SR	**76.0**	**80.1**	**81.9**

Table 6.4: BioBERT model performance (F1 score) using different types of augmented data. RS: random swap; RD: random deletion; SR: synonym replacement.

	Input sentence	Prediction
(1)	Instead, radiolabeled @CHEMICAL$ resulting from @PROTEIN$ hydrolysis were observed.	CPR:9
(2)	**Or else**, radiolabeled @CHEMICAL$ resulting from @PROTEIN$ hydrolysis were observed.	False
(1)	These results indicate that membrane @PROTEIN$ levels in N-38 neurons are dynamically autoregulated by @CHEMICAL$.	CPR:3
(2)	These results indicate that membrane @PROTEIN$ levels in N-38 **nerve cell** are dynamically autoregulated by @CHEMICAL$.	False

Table 6.5: Examples of prediction shift. (1): Original sentence; (2): Augmented sentence.

on the DDI and PPI tasks, which indicates that this operation might change the relation expression in the sentence. Thus it is necessary to verify the effectiveness of the operations before applying them on contrastive learning.

6.4.3 Measurement of rationale faithfulness

As discussed previously, we hypothesize the words on the shortest dependency path (SDP) as the rationales in the input. Therefore, the model should make its predictions based on them. If the model predictions are all made based on a specific part of the input, we can define this specific part of the input to be the completely faithful rationales. In practice, the rationales are more faithful means they are more influential on the model predictions.

In this chapter, we define a new metric to measure the faithfulness of the rationales: "prediction shift". If the model predicts one test example (non-negative)

Task	Model	Prediction Shift	
ChemProt	BioBERT	246	
	BioBERT+CLEK	191	(22% ↓)
	PubMedBERT	248	
	PubMedBERT+CLEK	189	(24% ↓)
DDI	BioBERT	111	
	BioBERT+CLEK	89	(20% ↓)
	PubMedBERT	90	
	PubMedBERT+CLEK	75	(17% ↓)
PPI*	BioBERT	51	
	BioBERT+CLEK	33	(35%↓)
	PubMedBERT	49	
	PubMedBERT+CLEK	34	(31%↓)

Table 6.6: Count of prediction shift on the "augmented" test set. *: The sum of counts on the 10 folds.

with label L_t, but changes its prediction on its neighbor (the augmented data point) with another label $L_t^{'}$, we will say a "prediction shift" happens (In Table 6.5, we give two examples of prediction shift on PubMedBERT model). Fewer "prediction shift" indicates the information outside of SDP influences the prediction less, which means the rationales are more faithful.

To generate a similar set (with test set) for the measurement of "prediction shift", we apply the same synonym replacement (SR) technique on the original test data. Since we retain the words that are on the shortest dependency path between the two entities, the generated data should express the same relation with the original ones. The trained model should predict them with the same labels if the rationales of input are utilized during inference, and in that case, we say the rationales are faithful.

We compare the number of "prediction shift" on two types of BERT model: the original BERT and the BERT model with contrastive pre-training. Table 6.6 illustrates that the BERT models with contrastive pre-training dramatically reduce the number of "prediction shift". Those results indicate that the BERT models with contrastive pre-training rely more on the information of shortest dependency path for prediction,

a.k.a., the rationales are more faithful. From another perspective, the results in Table 6.6 also demonstrate that the BERT models with contrastive pre-training are resilient to small changes of the inputs, which means the models are more robust.

6.5 Summary

In this chapter, we propose a contrastive pre-training method to improve the text representation of the BERT model. Our approach differs from previous studies in the choice of text data augmentation with linguistic knowledge and the use of the external knowledge bases to construct large-scale data to facilitate contrastive learning. The experiment results demonstrate that our method outperforms the original BERT model on three relation extraction benchmarks. Additionally, our method shows robustness to slightly changed inputs over the BERT models. In the future, we will investigate different settings of data augmentation and contrastive pre-training to exploit their capability on language models. We also hope that our work can inspire researchers to design better metrics and create high-quality datasets for the exploration of model interpretability.

Chapter 7

CONCLUSION

In recent years, applying deep learning and natural language processing techniques on the tasks in biomedical domain has become a common way to facilitate the biomedical research. In this dissertation, the focus has been considering methods to help the deep learning model generalization, especially when training dataset size is small.

First, we proposed methods to improve the BERT model in its pre-training and fine-tuning stages. In Chapter 3, we added another level of adaptation in the pre-training phase of BERT model using sub-domain data to help the BERT model generalization on the downstream tasks. The further pre-trained BERT models achieved better performance than the BERT models that are pre-trained with only general domain data (e.g. biomedical domain). Furthermore, we proposed a method to incorporate the information in the last layer of BERT that is not fully utilized during the original fine-tuning process in Chapter 4. The experiment results demonstrated the last layer of BERT model contains useful information for the relation extraction tasks and utilizing this information could boost the BERT model performance. In addition, after combining our new methods for the pre-training and fine-tuning of BERT model, better performance was obtained on our relation extraction tasks. Specifically, the model with combined techniques can achieve 2.8%, 2.9% and 1.7% F1 score improvement (compared with original BioBERT model) on the PPI, DDI and ChemProt tasks, respectively. Similarly, we can also improve the PubMedBERT model performance with 1.9%, 1.7% and 1.4% F1 score refinement on the three tasks, respectively.

Also, we considered using the annotated datasets from distant supervision (DS) to help the deep learning model generalization in Chapter 5. We first explored various

methods to reduce noise in the automatically generated data, and then investigated methods to combine the human-labeled data and DS-generated data. The experiment results showed that the mechanism of transfer learning provides much better results than either of these two types of data individually. We also applied the transfer learning technique on the recently-proposed BERT model. It was shown that the transfer learning can improve the BERT model performance on our relation extraction tasks. In particular, the use of transfer learning improved the BioBERT model with 3.1% and 3.3% F1 score refinement for PPI and MIRGENE tasks, respectively. For PubMedBERT, we observed 2.0% and 1.9% F1 score improvement on PPI and MIRGENE tasks, respectively.

In Chapter 6, we proposed a contrastive pre-training method to improve the input representation of the BERT model. In our contrastive learning framework, we designed a novel data augmentation method for relation extraction task with the use of linguistic knowledge. Also, we were able to construct large-scale data to help the generalization of contrastive learning with the help of external knowledge bases. The results of our experiments demonstrated that the proposed method outperforms the original BERT model on three relation extraction benchmarks. Specifically, we observed an averaged F1 score improvement (on the BioBERT and PubMedBERT models) of 1.2%, 1.2%, and 0.85% on the ChemProt, DDI, and PPI datasets, respectively. Additionally, we showed that the BERT models with contrastive pre-training are more resilient to small changes of the inputs, which indicates the robustness of the models.

At last, we will provide some directions for the future work. In Chapter 3, we employed a simple way (using PubMed query or Pubtator entity detection) to generate the training data in the sub-domain adaptation process. The PubMed query might be too broad and may yield abstracts that are not relevant for the tasks. Besides, some of the abstracts might not even contain any specific named entities. With more advanced queries, we maybe be able to only extract the related abstracts about our relations. For the next step, we can investigate the impact of queries (ways of extracting training data) on the sub-domain adaptation. For the proposed SLL fine-tuning mechanism in Chapter

4, we conducted a preliminary experiment to explore the where the attention mechanism is paying more attention to in the text sequence. In the future, we can explore how the attention mechanism can be applied to help detect the trigger information of relation expression. In Chapter 5, we experimented with transfer learning on binary classification problems, which leaves its applications on multi-class relation extraction tasks unexplored. Even though distant supervision can only give us binary labeled data, the generated dataset contains important knowledge about the relation expression for both binary and fine-grained (multi-class) relation extraction. Following this direction, we can continue our investigation of applying transfer learning to multi-class relation extraction problems. In Chapter 6, we have demonstrated that synonym replacement is the best technique to generate similar pairs in the procedure of contrastive learning. Considering the synonym database we are using is for general English, it might not have appropriate synonyms for the biomedical entities. Thus, we can explore some synonym databases in biomedical domain and see if they can help us on the synonyms of biomedical entities in the next step.

BIBLIOGRAPHY

[1] Waleed Ammar, Dirk Groeneveld, Chandra Bhagavatula, Iz Beltagy, Miles Crawford, Doug Downey, Jason Dunkelberger, Ahmed Elgohary, Sergey Feldman, Vu Ha, et al. Construction of the literature graph in semantic scholar. *arXiv preprint arXiv:1805.02262*, 2018.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Iz Beltagy, Kyle Lo, and Arman Cohan. SciBERT: a pretrained language model for scientific text. *arXiv:1903.10676 [cs]*, September 2019.

[4] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: a review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1798–1828, June 2013.

[5] Emmanuel Boutet, Damien Lieberherr, Michael Tognolli, Michel Schneider, and Amos Bairoch. Uniprotkb/swiss-prot. In *Plant bioinformatics*, pages 89–112. Springer, 2007.

[6] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[7] Razvan Bunescu, Ruifang Ge, Rohit J Kate, Edward M Marcotte, Raymond J Mooney, Arun K Ramani, and Yuk Wah Wong. Comparative experiments on

learning information extractors for proteins and their interactions. *Artificial intelligence in medicine*, 33(2):139–155, 2005.

[8] Razvan C Bunescu and Raymond J Mooney. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, pages 724–731. Association for Computational Linguistics, 2005.

[9] Rui Cai, Xiaodong Zhang, and Houfeng Wang. Bidirectional recurrent convolutional neural network for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 756–765, 2016.

[10] Juan Miguel Cejuela, Peter McQuilton, Laura Ponting, Steven J Marygold, Raymund Stefancsik, Gillian H Millburn, and Burkhard Rost. tagtog: interactive and text-mining-assisted annotation of gene mentions in plos full-text articles. *Database*, 2014, 2014.

[11] Juan Miguel Cejuela, Shrikant Vinchurkar, Tatyana Goldberg, Madhukar Sollepura Prabhu Shankar, Ashish Baghudana, Aleksandar Bojchevski, Carsten Uhlig, André Ofner, Pandu Raharja-Liu, Lars Juhl Jensen, et al. Loctext: relation extraction of protein localizations to assist database curation. *BMC bioinformatics*, 19(1):15, 2018.

[12] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607, 2020.

[13] Billy Chiu, Gamal Crichton, Anna Korhonen, and Sampo Pyysalo. How to train good word embeddings for biomedical nlp. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, pages 166–174, 2016.

[14] M. Craven and J. Kumlien. Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 77–86, Heidelberg, Germany, 1999. AAAI Press.

[15] Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. In *NIPS*, pages 3079–3087, 2015.

[16] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *ACL*, pages 4171–4186, June 2019.

[18] Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. ERASER: a benchmark to evaluate rationalized NLP models. In *ACL*, pages 4443–4458, July 2020.

[19] Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. CERT: Contrastive Self-supervised Learning for Language Understanding. *arXiv:2005.12766 [cs, stat]*, June 2020.

[20] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12), 2009.

[21] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. Bidirectional lstm networks for improved phoneme classification and recognition. In *International conference on artificial neural networks*, pages 799–804. Springer, 2005.

[22] 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:2007.15779*, February 2021.

[23] Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A Smith. Don't stop pretraining: Adapt language models to domains and tasks. *arXiv preprint arXiv:2004.10964*, 2020.

[24] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, volume 2, pages 1735–1742, June 2006.

[25] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9726–9735, June 2020.

[26] María Herrero-Zazo, Isabel Segura-Bedmar, Paloma Martínez, and Thierry Declerck. The DDI corpus: an annotated corpus with pharmacological substances and drug-drug interactions. *Journal of Biomedical Informatics*, 46(5):914–920, October 2013.

[27] Lynette Hirschman and Robert Gaizauskas. Natural language question answering: the view from here. *natural language engineering*, 7(4):275, 2001.

[28] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[29] Yu-Lun Hsieh, Yung-Chun Chang, Nai-Wen Chang, and Wen-Lian Hsu. Identifying protein-protein interactions in biomedical literature using recurrent neural networks with long short-term memory. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 240–245, 2017.

[30] Sheng-Da Hsu, Yu-Ting Tseng, Sirjana Shrestha, Yu-Ling Lin, Anas Khaleel, Chih-Hung Chou, Chao-Fang Chu, Hsi-Yuan Huang, Ching-Min Lin, Shu-Yi Ho, et al. mirtarbase update 2014: an information resource for experimentally validated mirna-target interactions. *Nucleic acids research*, 42(D1):D78–D85, 2014.

[31] Lei Hua and Chanqin Quan. A shortest dependency path based convolutional neural network for protein-protein relation extraction. *BioMed research international*, 2016, 2016.

[32] Xuanjing Huang et al. Attention-based convolutional neural network for semantic relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2526–2536, 2016.

[33] 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.

[34] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

[35] Jin-Dong Kim, Yue Wang, Toshihisa Takagi, and Akinori Yonezawa. Overview of genia event task in bionlp shared task 2011. In *Proceedings of the BioNLP Shared Task 2011 Workshop*, pages 7–15. Association for Computational Linguistics, 2011.

[36] Tassilo Klein and Moin Nabi. Contrastive self-supervised learning for commonsense reasoning. In *ACL*, pages 7517–7523, 2020.

[37] Martin Krallinger, Florian Leitner, Carlos Rodriguez-Penagos, and Alfonso Valencia. Overview of the protein-protein interaction annotation extraction task of biocreative ii. *Genome biology*, 9(2):S4, 2008.

[38] Martin Krallinger, Obdulia Rabal, Saber A. Akhondi, Martín Pérez Pérez, Jesús Santamaría, Gael Pérez Rodríguez, Georgios Tsatsaronis, Ander Intxaurrondo, José Antonio López1 Umesh Nandal, Erin Van Buel, Akileshwari Chandrasekhar, Marleen Rodenburg, Astrid Laegreid, Marius Doornenbal, Julen Oyarzabal, Analia

Lourenço, and Alfonso Valencia. Overview of the BioCreative VI chemical-protein interaction track. In *Proceedings of the BioCreative workshop*, pages 141–146, 2017.

[39] Andre Lamurias, Luka A Clarke, and Francisco M Couto. Extracting microrna-gene relations from biomedical literature using distant supervision. *PloS one*, 12(3):e0171929, 2017.

[40] Phuc H. Le-Khac, Graham Healy, and Alan F. Smeaton. Contrastive representation learning: a framework and review. *IEEE Access*, 8:193907–193934, 2020.

[41] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[42] Ji Young Lee, Franck Dernoncourt, and Peter Szolovits. Transfer learning for named-entity recognition with neural networks. *arXiv preprint arXiv:1705.06273*, 2017.

[43] 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 (Oxford, England)*, 36(4):1234–1240, February 2020.

[44] Tao Lei, Regina Barzilay, and Tommi Jaakkola. Rationalizing neural predictions. In *EMNLP*, pages 107–117, 2016.

[45] Gang Li, Karen E Ross, Cecilia N Arighi, Yifan Peng, Cathy H Wu, and K Vijay-Shanker. mirtex: a text mining system for mirna-gene relation extraction. *PLoS computational biology*, 11(9):e1004391, 2015.

[46] Gang Li, Cathy Wu, and K Vijay-Shanker. Noise reduction methods for distantly supervised biomedical relation extraction. *BioNLP 2017*, pages 184–193, 2017.

[47] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A

robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[48] Klaus Macherey, Franz Josef Och, and Hermann Ney. Natural language understanding using statistical machine translation. In *Seventh European Conference on Speech Communication and Technology*, 2001.

[49] David McClosky. Any domain parsing: automatic domain adaptation for natural language parsing. 2010.

[50] 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.

[51] George A. Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, November 1995.

[52] Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. Distant supervision for relation extraction with an incomplete knowledge base. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 777–782, 2013.

[53] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics, 2009.

[54] Sandra Orchard, Mais Ammari, Bruno Aranda, Lionel Breuza, Leonardo Briganti, Fiona Broackes-Carter, Nancy H. Campbell, Gayatri Chavali, Carol Chen, Noemi del-Toro, Margaret Duesbury, Marine Dumousseau, Eugenia Galeota, Ursula Hinz, Marta Iannuccelli, Sruthi Jagannathan, Rafael Jimenez, Jyoti Khadake, Astrid Lagreid, Luana Licata, Ruth C. Lovering, Birgit Meldal, Anna N. Melidoni, Mila

Milagros, Daniele Peluso, Livia Perfetto, Pablo Porras, Arathi Raghunath, Sylvie Ricard-Blum, Bernd Roechert, Andre Stutz, Michael Tognolli, Kim van Roey, Gianni Cesareni, and Henning Hermjakob. The MIntAct project–IntAct as a common curation platform for 11 molecular interaction databases. *Nucleic Acids Research*, 42(Database issue):D358–363, January 2014.

[55] Sandra Orchard, Mais Ammari, Bruno Aranda, Lionel Breuza, Leonardo Briganti, Fiona Broackes-Carter, Nancy H Campbell, Gayatri Chavali, Carol Chen, Noemi Del-Toro, et al. The mintact project—intact as a common curation platform for 11 molecular interaction databases. *Nucleic acids research*, 42(D1):D358–D363, 2013.

[56] Sinno Jialin Pan, Qiang Yang, et al. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[57] Hao Peng, Tianyu Gao, Xu Han, Yankai Lin, Peng Li, Zhiyuan Liu, Maosong Sun, and Jie Zhou. Learning from context or names? An empirical study on neural relation extraction. In *EMNLP*, pages 3661–3672, 2020.

[58] Yifan Peng and Zhiyong Lu. Deep learning for extracting protein-protein interactions from biomedical literature. *arXiv preprint arXiv:1706.01556*, 2017.

[59] 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 Workshop on Biomedical Natural Language Processing (BioNLP)*, pages 58–65, June 2019.

[60] 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.

[61] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *NAACL*, pages 2227–2237, 2018.

[62] Jason Phang, Thibault Févry, and Samuel R Bowman. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*, 2018.

[63] Martin F Porter et al. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[64] Chanqin Quan, Lei Hua, Xiao Sun, and Wenjun Bai. Multichannel convolutional neural network for biological relation extraction. *BioMed research international*, 2016, 2016.

[65] Alec Radford, Karthik Narasimhan, Time Salimans, and Ilya Sutskever. Improving language understanding with unsupervised learning. *Technical report, OpenAI*, 2018.

[66] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

[67] Benjamin Roth, Tassilo Barth, Michael Wiegand, and Dietrich Klakow. A survey of noise reduction methods for distant supervision. In *Proceedings of the 2013 workshop on Automated knowledge base construction*, pages 73–78. ACM, 2013.

[68] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[69] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[70] Youwei Song, Jiahai Wang, Zhiwei Liang, Zhiyue Liu, and Tao Jiang. Utilizing bert intermediate layers for aspect based sentiment analysis and natural language inference. *arXiv preprint arXiv:2002.04815*, 2020.

[71] Chris Stark, Bobby-Joe Breitkreutz, Teresa Reguly, Lorrie Boucher, Ashton Breitkreutz, and Mike Tyers. BioGRID: a general repository for interaction datasets. *Nucleic Acids Research*, 34(Database issue):D535–539, January 2006.

[72] Mihai Surdeanu, David McClosky, Julie Tibshirani, John Bauer, Angel X Chang, Valentin I Spitkovsky, and Christopher D Manning. A simple distant supervision approach for the tac-kbp slot filling task. 2010.

[73] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[74] Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 721–729. Association for Computational Linguistics, 2012.

[75] Ian Tenney, Dipanjan Das, and Ellie Pavlick. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*, 2019.

[76] Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. What do you learn from context? probing for sentence structure in contextualized word representations. *arXiv preprint arXiv:1905.06316*, 2019.

[77] Frank Van Harmelen, Vladimir Lifschitz, and Bruce Porter. *Handbook of knowledge representation*. Elsevier, 2008.

[78] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

[79] Ioannis S Vlachos, Maria D Paraskevopoulou, Dimitra Karagkouni, Georgios Georgakilas, Thanasis Vergoulis, Ilias Kanellos, Ioannis-Laertis Anastasopoulos, Sofia Maniou, Konstantina Karathanou, Despina Kalfakakou, et al. Diana-tarbase v7. 0: indexing more than half a million experimentally supported mirna: mrna interactions. *Nucleic acids research*, 43(D1):D153–D159, 2014.

[80] Linlin Wang, Zhu Cao, Gerard De Melo, and Zhiyuan Liu. Relation classification via multi-level attention cnns. 2016.

[81] Chih-Hsuan Wei, Hung-Yu Kao, and Zhiyong Lu. Pubtator: a web-based text mining tool for assisting biocuration. *Nucleic acids research*, 41(W1):W518–W522, 2013.

[82] Chih-Hsuan Wei, Hung-Yu Kao, and Zhiyong Lu. Gnormplus: an integrative approach for tagging genes, gene families, and protein domains. *BioMed research international*, 2015, 2015.

[83] Jason Wei and Kai Zou. EDA: easy data augmentation techniques for boosting performance on text classification tasks. In *EMNLP-IJCNLP*, pages 6381–6387, 2019.

[84] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big Data*, 3(1):9, 2016.

[85] David S. Wishart, Craig Knox, An Chi Guo, Dean Cheng, Savita Shrivastava, Dan Tzur, Bijaya Gautam, and Murtaza Hassanali. DrugBank: a knowledgebase for drugs, drug actions and drug targets. *Nucleic Acids Research*, 36(Database issue):D901–906, January 2008.

[86] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[87] 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.

[88] Zhilin Yang, Ruslan Salakhutdinov, and William W Cohen. Transfer learning for sequence tagging with hierarchical recurrent networks. *arXiv preprint arXiv:1703.06345*, 2017.

[89] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762, 2015.

[90] Dongxu Zhang and Dong Wang. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*, 2015.

[91] Yijia Zhang and Zhiyong Lu. Exploring semi-supervised variational autoencoders for biomedical relation extraction. *arXiv preprint arXiv:1901.06103*, 2019.

[92] Zhehuan Zhao, Zhihao Yang, Hongfei Lin, Jian Wang, and Song Gao. A protein-protein interaction extraction approach based on deep neural network. *International Journal of Data Mining and Bioinformatics*, 15(2):145–164, 2016.

[93] Wu Zheng and Catherine Blake. Using distant supervised learning to identify protein subcellular localizations from full-text scientific articles. *Journal of biomedical informatics*, 57:134–144, 2015.

[94] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212, 2016.