STUDIES ON DIVERSE INPUT REPRESENTATIONS AND CLASSIFIERS ON RELATION EXTRACTION DATASETS

Mingyou Sung

A thesis
in
The Department
of
Computer Science

Presented in Partial Fulfillment of the Requirements
For the Degree of Master of Computer Science
Concordia University
Montréal, Québec, Canada

April 2022

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Mingyou Sung**

Entitled: **Studies on diverse input representations and classifiers on relation extraction datasets**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

_____ Chair

Dr. René Witte

_____ Examiner

Dr. Yiming Xiao

_____ Supervisor

Dr. Sabine Bergler

Approved _____

Dr. Lata Narayanan
Chair of Department or Graduate Program Director

_____ _____

Date

Mourad Debbabi, Ph.D., Dean
Faculty of Engineering and Computer Science

Abstract

Studies on diverse input representations and classifiers on relation extraction datasets

Mingyou Sung

The relation extraction task which aims to identify the relationship between a specified pair of words is considered a significant task that can be expanded to be utilized in various ways. Therefore, the automatic relation extraction system is often considered a key to extracting systematic reusable information from sentences, paragraphs, or documents. Instead of achieving state-of-the-art performance, this thesis aims to explore the significance of various input & output representations, and the difference between a linear and a bilinear classifier on relation extraction tasks. For a thorough analysis, I experiment on a diverse group of relation extraction datasets and present a set of ablation studies. Moreover, experiments are compared not only based on their performance but also on the efficiency of resource usage. The analysis illustrates that the systems based on certain input & output representations yield the best performance in general even though introduced systems have less complexity compared to bilinear-classifier-based systems. Moreover, the straightforward systems studied in this thesis show results comparable to state-of-the-art systems (3% difference) in general.

Acknowledgments

This two-year journey was an opportunity to look back on my strengths and weaknesses and learn how to approach achievement step by step. Also, I could appreciate people who care for me. I would like to take this opportunity to thank everyone who supported me during this journey.

First and foremost, I sincerely appreciate Dr. Sabine Bergler who gave me this opportunity that I earnestly hoped for. It was a precious journey that I had never imagined before I came here. Every dialogue we had was the foundation of my growth. Again, I am truly grateful to Dr. Bergler for her endless advice, and humour that energizes me.

Life in the CLaC lab is one of the best luck of my life. I met so many amazing and warm-hearted individuals. Thank you to all my colleagues - Parsa, Nihatha, Narjes, Nadia S., Zhanfan, Harsh, Naida B., Sunanda, Benjamin, Claire. The discussion that we had in the lab broadened my perspective and taught me how to deliver the information and knowledge precisely to others. Parsa always gave me valuable suggestions when I missed something or struggled. Through the communication with Parsa, he navigated me how to surf this big wave in front of me. Also, the meme time after the weekly meeting was the vitamin of the day that make us laugh. Thank you again for everyone.

The word “Appreciation” is insufficient to describe how much support I received from my family. This journey was able to carry out with the sincere backing of my parents. My heartfelt gratitude to my parents for all the support, and for all the sacrifices they accepted for me. I want to give credit to my parents for this achievement.

I would like to thank all my best friends who share worries and pleasure. Looking back, it seems that all those worries of the day have been relieved by all my friends’ entertaining stories. Also, I would like to thank to whom for always supporting me, smiling at me, and worrying about me together. This thesis was able to say hello to the world with the steady support of my supervisor, family, colleagues, friends and significant other.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Natural language processing (NLP) is a field of study in computer science regarding interactions of human language and computers. The structure of the architectures applied in NLP tasks can be rule-based (symbolic), probabilistic, or connectionist (neural networks). The term *system* is defined in this thesis as a computing structure that learns from the textual input such as sentences, paragraphs or documents to perform various tasks such as information retrieval, text summarization, text classification, text generation, etc. Among these various tasks, the relation extraction task which aims to identify the relationship between a specified pair of words (usually nominals) is considered a significant task that can be expanded to be utilized for information extraction, recognizing textual entailment, document summarization, etc [Hendrickx et al., 2009]. In the relation extraction task, target words are words involved in the concerned relationships. Following the custom in the literature, I refer to the target word as an *entity* in this thesis. The output of the relation extraction task from the given data sample is a set of triples (*e1*,*e2*,r) where *e1* and *e2* are the two entities and r is the relation between them. The given data sample is a single sentence or multiple sentences in general. For instance, Example (1) from SemEval 2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals [Hendrickx et al., 2009] illustrates two entities *cup* and *water* that stand in the relation *Content-Container* and this is extracted as `Content-Container(water,cup)` triple.

(1) *The water was in a cup.*

`Content-Container(water,cup)`

This concise representation of the text can be applied to construct knowledge graphs that require entities and relations ([Ehrlinger and Wöß, 2016]; [Zhang et al., 2017]). Such knowledge graphs can enhance information retrieval by having access to structured information. For this reason, the automatic relation extraction system is considered key to extracting

systematic reusable information from sentences, paragraphs or documents.

The main objective of the relation extraction task is to identify the relationship between a specified pair of entities from the given textual input. In general, the pair of entities are provided from the given textual input ([Hendrickx et al., 2009]; [Zhang et al., 2017]; [Stoica et al., 2021]) as presented in Example (1). Therefore, the relationship is determined only by the given pair of entities in this case. On the other hand, some tasks only provide a set of entities and textual input without the information on which specific pair of entities are linked. Therefore, the goal of this type of task is not only to classify the relationship but also to detect a pair of entities that form a relation. Example (2) from Biocreative VII Track 1 - Text mining drug and chemical-protein interactions(DrugProt) [Miranda et al., 2021] presents a set of entities and textual input.

(2) *Hypoxemia associated with cimetidine therapy in a newborn infant. Cimetidine therapy used for the treatment of gastric bleeding due to tolazoline therapy in a newborn infant was temporally associated with episodes of severe hypoxemia. It appears likely that the histamine H2 receptor blocked by cimetidine obviated the pulmonary vasodilator effect of tolazoline therapy.*

`INHIBITOR(histamine H2 receptor, cimetidine)`

The highlighted and the underlined span from the above example represent entities. The linked entities are *histamine H2 receptor* and the fifth highlighted entity *cimetidine*. The textual input used in the experiments in this thesis come from diverse sources including online news-papers, discussion forums, blogs, and biomedical documents. In addition, the data sample of given textual input can be a single sentence ([Hendrickx et al., 2009]; [Zhang et al., 2017]; [Stoica et al., 2021]) or multiple sentences, paragraphs or documents ([Yao et al., 2019]; [Miranda et al., 2021]). In this thesis, SemEval 2010 Task 8: Multi-Way Classification of Semantic Relations between Pairs of Nominals (SemEval 2010 Task 8) [Hendrickx et al., 2009], TAC Relation Extraction Dataset (TACRED) [Zhang et al., 2017], Revised-TACRED (Re-TACRED) [Stoica et al., 2021] and Biocreative VII Track 1 - Text mining drug and chemical-protein interactions(DrugProt) [Miranda et al., 2021] are selected. SemEval 2010 Task 8 is to classify the relationship between two nominals from the given sentence. TACRED is to classify the relationship between people, organizations and locations. TACRED is bigger in dataset size compared to SemEval 2010 Task 8. Re-TACRED is also to classify the relationship between people, organizations, and locations like TACRED but it is the revised version of TACRED due to several issues such as the span of entities, ambiguous and inconsistent relation definitions, miscellaneous data samples which are not written in English, and partial span of entities [Stoica et al., 2021]. Therefore, Re-TACRED has a comparatively smaller dataset size than TACRED. Biocreative VII Track 1 is a distinctive

dataset compared to the datasets that are introduced above. This is because Biocreative VII Track 1 is extracted from the medical documents by experts, whereas SemEval 2010, TACRED, and Re-TACRED are based on general English. Therefore, Biocreative VII Track 1 aims to classify the relationship between chemical compounds/drugs and genes/proteins from the medical documents.

Prior to the emergence of deep-learning systems, conventional machine learning methods such as SVM or Bayes Net were extensively used in classifying relationships between entities ([Rink and Harabagiu, 2010]; [Chen et al., 2010]; [Tymoshenko and Giuliano, 2010]). These systems often applied various linguistic resources such as WordNet [Miller, 1995], POS tags and Dependencies to leverage information to the system. With the beginning of the flourishing era of deep-learning architectures, numerous deep neural network approaches have been used for relation extraction tasks such as Convolutional deep neural networks [Zeng et al., 2014], Bi-Long-Short-Term-Memory networks [Zhang et al., 2015], Graph convolution networks [Zhang et al., 2018] and Transformer-based networks [Soares et al., 2019].

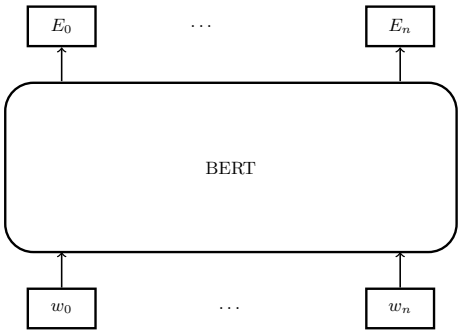


Figure 1: Simplified BERT architecture

A transformer based architecture called BERT is analyzed in this thesis because it has been the most extensively applied deep-learning architecture in NLP in the last two years that achieves state-of-the-art (SOTA) performance on various NLP tasks. BERT leverages tokens, which are small units of text (eg. word), and transforms them into real-valued vectors which are called word embeddings that correspond to each token. The simplified architecture of BERT is presented in Figure 1 where word embeddings $< w_0, \ldots, w_n >$ are the input of BERT and the real-valued vectors which are the output of BERT are $< E_0, \ldots, E_n >$. The most distinctive feature of BERT compared to previous deep-learning architectures is that the fundamental concept of BERT creates a representation of a token by thoroughly attending to all the words in the context. Therefore, it provides attention scores that decide which token needs to be concentrated on in the current context and which has to be less focused on.

[Soares et al., 2019] leveraged various input & output representations in relation extraction tasks. The definition of *input & output representation* described in this thesis is the manner of representing the textual input of BERT and the choice of the output vector from BERT. Even though various input & output representation methods were studied in ([Soares et al., 2019]); [Tao et al., 2019]; [Alt et al., 2019]; [Shi and Lin, 2019]), it was challenging to compare the effectiveness of input & output representation methods with one another. This is because most studies present their results by comparing SOTA, recent previous studies, or with a limited number of tasks. For this reason, a set of ablation studies is introduced in this thesis in order to achieve an extended comparison of different input & output representation methods on relation extraction tasks.

The basic manner of applying BERT to the NLP tasks is utilizing a vector of special token [CLS] which is placed prior to the tokenized textual input. This is because it is claimed that the result of CLS embedding embeds the entire input sequence by aggregating the textual input information ([Devlin et al., 2018]). However, [Wu and He, 2019] obtained improved performance on relation extraction tasks by leveraging not only CLS vector but also two averaged entity vectors to the linear classifier instead of adopting the basic manner of utilizing BERT. Moreover, the bilinear transformation is being extensively used as a classifier in document-level relation extraction tasks and obtained promising results ([Wang et al., 2019]; [Yao et al., 2019]; [Tang et al., 2020]). The major difference between a linear and a bilinear transformation is that a bilinear transformation can handle two input vectors simultaneously so two input vectors share the same learnable weight, whereas the two input vectors have to be combined in order to compute in a linear transformation. Figure 2 shows the simplified linear and bilinear classifier. Therefore, it seems the choice of

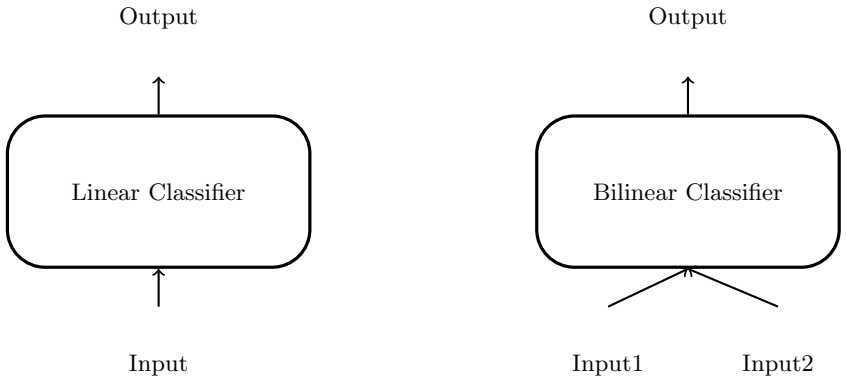


Figure 2: Simplified linear and bilinear classifier

a classifier potentially affect the performance on relation extraction tasks. For this reason, various experiments based on these approaches are introduced in this thesis.

After BERT achieved successful performance in NLP tasks, several BERT like language models such as RoBERTa [Liu et al., 2019], DistilBERT and DistilRoBERTa [Sanh et al., 2019] appeared. The term *language model* is defined in this thesis as a pre-trained word embedding system that is able to convert tokens into vectors.

Since BERT like language models were pre-trained on different architecture, different sizes of datasets, and methods (see Table 1), each language model yield different performance on General Language Understanding Evaluation benchmark (GLUE) as presented in Table 2. GLUE is a collection of datasets for training, evaluating, and analyzing NLP

Language Model	**Layers**	**Hidden Size**	**Attention Heads**	**Parameters (M)**
BERT$_{base}$	12	768	12	110
RoBERTa$_{base}$	12	768	12	125
DistilBERT$_{base}$	6	768	12	66
DistilRoBERTa$_{base}$	6	768	12	82

Table 1: Description of the architectures and number of parameters of four BERT like language models.

systems with one another [Wang et al., 2018]. In Table 2, it shows RoBERTa performs best on most of the tasks of GLUE. Moreover, DistilRoBERTa which is a smaller version of RoBERTa performs better than DistilBERT in most cases. Therefore, it is presumed that the performance of the system is highly related to the choice of the language model. For this reason, a set of ablation studies regarding different language models on four relation extraction tasks are introduced in this thesis in order to investigate the influence of the language model on its performance.

The goal of this thesis is not to achieve SOTA performance from the tasks but to explore the usefulness of BERT by applying various approaches to relation extraction tasks. Therefore, this thesis concentrates on the comparison of various input & output representations, two different classifiers, and language models on relation extraction tasks. The applications of input & output representations in BERT are 1) utilizing special tokens to specify the span of entities, 2) selecting different output vectors other than CLS vector from the last hidden state of BERT, and 3) attaching additional entity tokens next to the context at the input step. The application of utilizing special tokens to specify the span of entities illustrates the significance of straightforward and inspectable input encoding of BERT on the tasks that were not used for the pre-training of BERT. The selection of different output

System	GLUE Task							
	MNLI	RTE	QQP	QNLI	SST-2	CoLA	STS-B	MRPC
Pre-BERT SOTA	82.1	56.0	70.3	87.4	91.3	45.4	80.0	82.3
BERT_{base}	84.6	66.4	71.2	90.5	93.5	52.1	85.8	88.9
RoBERTa_{base}	**87.6**	**78.7**	**91.9**	**92.8**	**94.8**	**63.6**	**91.2**	**90.2**
DistilBERT_{base}	82.2	59.9	88.5	89.2	91.3	51.3	85.8	87.5
$\text{DistilRoBERTa}_{base}$	84.0	67.9	89.4	90.8	92.5	59.3	88.3	86.6

Table 2: Comparison of the performance of various BERT like language models on GLUE test dataset except WNLI task. Pre-BERT SOTA architecture is OpenAI GPT [Radford et al., 2018] which is based on Transformer that replaced previous SOTA architectures.

vectors other than CLS vector suggests the potential of extension of BERT output vector usage. Leveraging extra entity tokens that are attached next to the context clarifies the influence of the attached additional information related to the relation extraction task and the significance of the position of the attachment. This thesis presents the utility of a bilinear classifier on relation extraction tasks where two entities are key objective by comparing with a linear classifier. Last, this thesis applies four different BERT like language models to observe the significance of the language model on its performance.

This thesis presents the importance of straightforward input encoding by showing that systems that indicate the scope of entities by applying special tokens provide significantly improved performance compared to systems that are not applied. Moreover, the extended usage of the output vector of BERT rather than only the CLS vector yields improved performance. Furthermore, the application of attaching additional entity tokens next to the context generally shows improved performance regardless of its position of attachment. Regarding the choice of classifier, bilinear classifier systems achieved enhanced performance but the performance gap with the best systems of the linear classifier is trivial. As regards the usage of different language models, RoBERTa performs best in most cases compared to BERT, DistilBERT, and DistilRoBERTa. Interestingly, the performance of the best systems based on BERT in this thesis is close to SOTA performance despite SOTA systems applying multiple domain-specific BERTs or larger language models such as RoBERTa. Therefore, this thesis suggests that straightforward input encoding and the enriched choice of the output vectors enhance the usefulness of BERT.

This thesis is organized in the following order. Chapter 2 provides the background

knowledge which is the most important for understanding NLP notations and the neural networks applied. Moreover, Chapter 2 is organized chronologically, starting with the description of the conventional neural networks and ending with the description of Transformer based architectures that are frequently used in this thesis. Also, notions of activation functions, loss functions, classifiers that apply to this thesis are introduced, as well as the description of the relation extraction tasks. Chapter 3 contains the description of various input & output representations and architectures and their results. The in-depth analysis of various systems and the integrated results are presented with the SOTA system in Chapter 4. In Chapter 5, the conclusion of the study and the future direction is discussed.

Chapter 2

Background

2.1 NLP notations

Tokenization Tokenization is the first step in NLP where a piece of text such as a sentence or document is split into small pieces called tokens where the most widely used methods of tokenization are word-based, character-based, and subword-based tokenization. Word-based tokenization technique is most commonly used technique. The text is broken down into word level using a selected delimiter. The delimiter can be a blank space or punctuation. Different tokens are formed depending on the delimiter. Frequently used word level tokenization tools are ANNIE [Maynard et al., 2000], NLTK [Loper and Bird, 2002], Stanza [Qi et al., 2020], CoreNLP [Chen and Manning, 2014], SpaCy [1]. Character-level tokenization is a technique that separates the text based on character. A distinctive characteristic of character-level tokenization is that it significantly reduces the size of vocabulary. Also, this tokenization technique is able to handle Out Of Vocabulary (OOV) issue. OOV issue represents when the new words that appeared at the test step are not foreshadowed at the training step and do not exist in the vocabulary. This is a crucial problem to the systems that leverage word embeddings because OOV words cannot be converted to a real-value vector as a representation of a token in the vector space [Luong et al., 2014]. Subword tokenization technique is most widely adopted in Transformer-based architectures such as WordPiece [Wu et al., 2016] which is outlined in [Schuster and Nakajima, 2012] and Byte-Pair Encoding [Sennrich et al., 2016]. WordPiece algorithm is used for BERT [Devlin et al., 2018] and DistilBERT [Sanh et al., 2019] architecture. Byte-Pair Encoding is a data-driven technique used by RoBERTa [Liu et al., 2019] (also DistilRoBERTa) to approach Neural Machine Translation problems such as the OOV issue and word segmentation problem by limiting

[1] https://spacy.io/api/tokenizer

the size of vocabulary and split the word into subword level. Table 3 demonstrates the tokenization result of given text in word level, character-level and subword level.

Text	*Calluses are caused by improperly fitting shoes or by a skin abnormality.*
Word	calluses, are, caused, by, improperly, fitting, shoes, or, by, a, skin, abnormality, .
Character	c, a, l, l, u, s, e, s, a, r, e, c, a, u, s, e, d, b, y, i, m, p, r, o, p, e, r, l, y, f, i, t, t, i, n, g, s, h, o, e, s, o, r, b, y, a, s, k, i, n, a, b, n, o, r, m, a, l, i, t, y, .
Subword	call, ##uses, are, caused, by, improper, ##ly, fitting, shoes, or, by, a, skin, abnormal, ##ity, .

Table 3: Various tokenization results from the given text

Part-of-Speech (POS) POS tags are a set of linguistic categories for denoting grammatical role of words in a given context. Therefore, nonidentical words can be classified as the same POS and the same words can be classified as a nonidentical POS according to their syntactic role in the context. For instance, even though word *skin* and *abnormality* in Table 3 are different in form, both words are a noun. Moreover, the word *book* is tagged as a noun or verb depending on the usage and the context.

(3) *Writing the book took ten months of hard slog.*

(4) *Book her a hotel room in Salt Lake City.*

These two examples represent the same words are tagged as different POS tags where the word *book* in Example (3) is tagged as a noun and it refers to a set of printed sheets of paper that are held together inside a cover. However, the word *book* in Example (4) is tagged as a verb where it refers to register for some future activity or condition. A typical POS tag set for the English grammar includes noun, pronoun, adjective, determiner, verb, adverb, preposition, conjunction, and interjection. The most commonly used in NLP is the Penn Treebank tag set [Marcus et al., 1994] which has 36 POS tags. Among various POS tagging tools, Figure 3 is an example of a POS tagged result of the tokens in Table 3 using ANNIE POS tagger [Maynard et al., 2000].

Tokens:	Calluses	are	caused	by	improperly	fitting	shoes	or	by	a	skin	abnormality
POS tags:	NNP	VBP	VBN	IN	RB	JJ	NNS	CC	IN	DT	NN	NN

Figure 3: Example of POS tags

Dependency A dependency represents the grammatical relationship between a pair of words in a sentence [Mel'cuk et al., 1988]. [De Marneffe and Manning, 2008] introduced the typed dependency which is designed to provide the triple of dependency link between the

pair of words uniformly. Each pair of words is connected by one or zero links and this link represents the grammatical relation between words. In this typed dependency, the pair of words is divided into a governor (also called a head) and a dependent, and this pair is connected via dependency relation. Therefore, a dependent is related to only one governor while a governor may be related to multiple dependents. To generate a dependency parse for text, it must be tokenized into word level as needed in POS tagging because dependencies are the representation of a grammatical relationship between the lexical terms in the context. Figure 4 is an example of a typed dependency parse of a given text in Table 3 using Stanford parser [Chen and Manning, 2014].

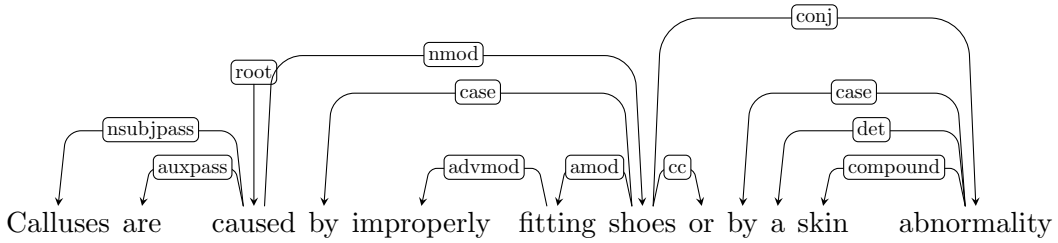


Figure 4: Example of typed dependency parsing

2.2 Neural networks for NLP

With the state-of-the-art performance of neural-network architectures, they are dominating NLP tasks these days. The basic concept of neural networks and word representation will be described in this section.

The most general usage of neural networks in NLP is to predict the label that satisfies the purpose of the task from word embeddings. Word embedding is a representation of a word or group of words that are converted into a group of numbers in form of vectors or matrices. To obtain certain predictions, the first step is to feed word embeddings to the neural networks. The output of the neural network system is represented as function f. To achieve the prediction, the Softmax function is often applied to the logits. Softmax normalizes the input value to a value of $\{x \in \mathbb{Q} \,|\, 0 \leq x \leq 1\}$, such that sum of all Softmax output values is always 1. Moreover, this normalized output is mapped to a probability distribution over the target of the classes. Therefore, Softmax is often applied as the last activation function of neural network architectures to normalize and map for the classification [Goodfellow et al., 2016].

The process of predicting task specific label is expressed as:

$$\begin{aligned} z &= f(X) \\ Softmax(z_i) &= \frac{exp(z_i)}{\sum_{j=1}^{n} exp(z_j)} \end{aligned} \tag{1}$$

where z_i is the logits for the class of index i and n is the size of class.

Word Embeddings To let the machine understand the human language, the tokenized text has to be represented in the numeric space and this can be done by transforming words into matrices which are called word embeddings. Word embedding is the form of real-valued vectors or matrices which indicate the word representation. Suppose a sentence consists of a sequence of words $< t_0, t_1, \ldots, t_T >$. The sequence of words can be represented by a matrix $X = [x_0 \oplus x_1 \oplus \cdots \oplus x_T]$ of word vector representations x_t, where $\oplus$ is the row-level concatenation operator.

The most basic manner of word representation is One-hot encoding where the word is often represented as $x_t \in \{0,1\}^{|V|}$, where only one entry is non-zero. $|V|$ refers to the cardinality of set of tokens. One-hot vector has a size of $|V|$ and each vector has only single 1 which represents the index of the vocabulary while other elements of the vector are filled with 0. Therefore, the word vectors can be stored in a matrix $E \in \mathbb{R}^{|V| \times |V|}$. E represents the matrices of the word embeddings. Even though the one-hot vector has advantages such as simplicity and it is easy to build, it has certain drawbacks such as it is difficult to measure the similarity between words because the one-hot vectors are orthogonal on one another.

The window-based co-occurrence matrix is frequency-based word representation which is focused on the idea that the words that have analogous usage tend to occur within a certain distance of similar words. The term *window* represents the span in which words are considered neighboring to each other. This co-occurrence matrix has a size of $|V| \times |V|$ if *window* is 1. Therefore, the trained word vectors can be stored in a matrix $E \in \mathbb{R}^{|V| \times |V|}$.

Moreover, other than word level representation, the document-level representation is vigorously applied in information retrieval. The basic document-level representation is TF–IDF (Term frequency-inverse document frequency) which is weighting the term based on word importance and rarity. TF–IDF is the multiplication of $tf(t,d)$ and $idf(d,t)$ where $tf(t,d)$ represents the number of occurrences of word t in document d and $idf(d,t)$ refers the inverse of the number of documents that word t appeared. Therefore, it indicates how important a word t is within a particular document from the group of documents. Even though the presented word and document representation methods above have several benefits such as simplicity and robustness, these manners have critical computation efficiency issues due to vector size and sparsity.

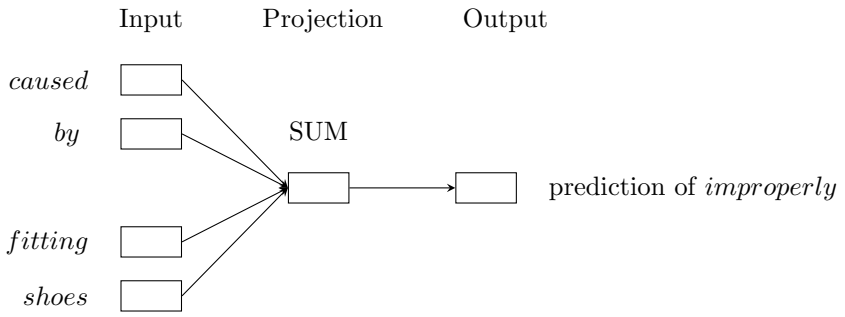


Figure 5: Example of CBoW of window size 4

To resolve this issue, Word2vec [Mikolov et al., 2013] and Glove [Pennington et al., 2014] which are a neural network architecture were introduced. Word2Vec is the fundamental method of the distributed representation that vectorizes the word representation in multi-dimensional space. Word2Vec proposed two different methods: Continuous Bag of Words (CBoW) and Skip-Gram. CBoW is a method of predicting the target word from words around the target word called surrounding words. In this case, the order of surrounding words does not influence prediction. Figure 5 represents the process of CBoW where predicts a target word (*improperly*) using surrounding words (*caused*, *by*, *fitting* and *shoes*) of Example (5). Since the window size is 4, the word embedding of preceding 2 words and subsequent 2 words of the position of the target words (one-hot encoding) are used for predicting the target word.

(5) *Calluses are caused by improperly fitting shoes or by a skin abnormality.*

CBoW consists of a single hidden layer and the input of this layer is the one-hot encoding of the token. Through the training, the hidden layer parameters are updating to correctly predict the token. Softmax function is applied at the classification step during the training. Once the training is done, the output of the hidden layer of the one-hot encoding of the token is used as word embedding. The size of the word embedding from CBoW is d which is the size of a hidden layer instead of $|V|$ which is the vocabulary size. The Skip-Gram method predicts surrounding words from the target word and it weighs closer words more than distant words. Figure 6 represents the process of Skip-Gram where it predicts surrounding words (*caused*, *by*, *fitting* and *shoes*) using a target word (*improperly*). Since the window size is 4, based on the word embedding of the target word (one-hot encoding), the word embedding of preceding 2 words and subsequent 2 words of the position of the target words are predicted. To obtain word embeddings, skip-gram uses the hidden layer output of the token once the system is trained. Therefore, the size of the word embedding from Skip-Gram is d which is the size of a hidden layer instead of $|V|$ which is the vocabulary size.

The trained word vectors can be stored in a matrix $E \in \mathbb{R}^{|d| \times |\mathbb{V}|}$ where d is the dimension of

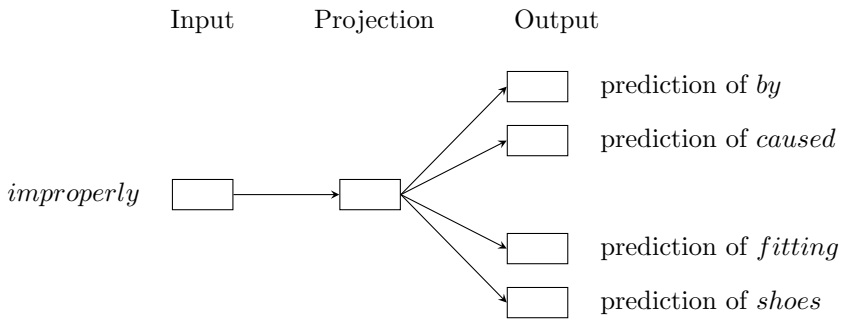


Figure 6: Example of Skip-gram of window size 4

the word representation vector (size of a hidden layer). Therefore, the word vectors are in a lower dimensional space (d = 50-300 in Glove) compared to the one-hot and co-occurrence based models.

ELMo (Embeddings from Language Models) which was introduced in [Peters et al., 2018] is a character-based word representation that contains bidirectionality. Unlike the traditional word embeddings like Word2Vec and Glove, ELMo solves the OOV issue by applying character-based technique. Moreover, ELMo achieves word representation in that each token is dependent on the context of the input sentence. The architecture of ELMo consists of two layers of Bidirectional LSTM which will be introduced in the next section. Due to the characteristics of the LSTM where it transfers the information that was previously processed to the later step, each word representation can traverse forward and backward depending on the directionality of LSTM. Therefore, it is available to distinguish the same words that exist in different texts and provide different word representations of the same word based on the context. For this reason, the word representation of ELMo is context-sensitive while Word2Vec and Glove provide fixed word representation.

2.2.1 Neural network architectures

Over the past few years, neural network architectures are extensively applied in NLP due to the outstanding performance of neural network structure.

Feed-forward neural network The feed-forward neural network is the basic concept of the neural network architectures which form is linear $f(X) = XW + b$. The X is input matrix, W is a weight matrix and b is a bias matrix. The W and b are learnable parameters that can be adjusted through the training while x is fixed. Figure 7 illustrates the basic architecture of feed-forward neural network.

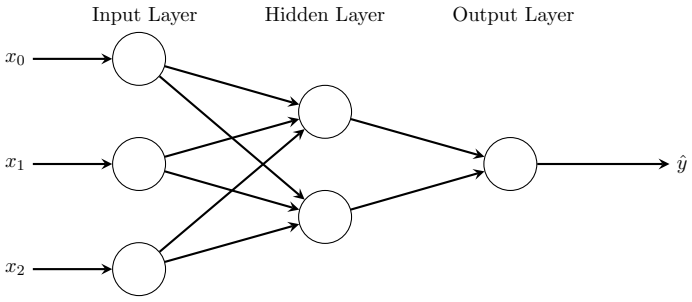


Figure 7: Feed-forward Network architecture

Feed-forward neural network is the simplest neural network architecture which is only able to move the information in only one direction (forward) and does not form a cycle [Zell, 1994]. The forward path in this case is from the input nodes to the next nodes which are the hidden nodes (if any) or the output nodes.

Recurrent Neural Networks Recurrent Neural Networks (RNNs) [Elman, 1990] which are the descendent of the feed-forward neural network forms a cycle that enables the process of sequential data such as text. Figure 8 illustrates the basic concept of RNNs. RNNs allow

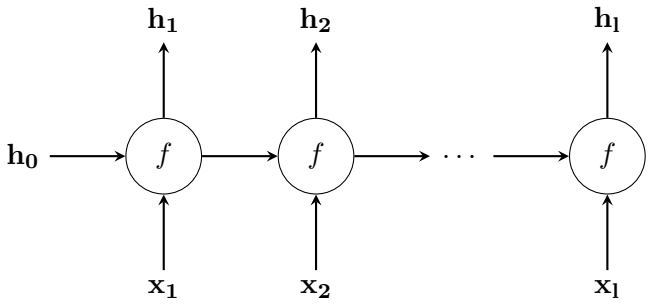


Figure 8: Unfolded Recurrent Neural Networks (RNN) architecture

the previous node's outputs to be used with the current node's inputs when computing the current node's output so the current node's output is enabled to be influenced by not only the current node's input but the previous node's output. Another distinct feature of RNNs are that each node in the same layer shares the weight and bias parameters. While the feed-forward network parameters that belong to each node in the same layer are different. However, it is difficult to handle information that has a long sequence due to vanishing gradient and exploding gradient issues. This problem can be solved by replacing activation functions or using Long-Short-Term-Memory networks (LSTM) [Hochreiter and Schmidhuber, 1997] or Gated Recurrent Units (GRU) [Cho et al., 2014]. LSTM has a special component called

forget gate which can decide whether to preserve all the information so far or forget everything. Figure 9 shows the basic architecture of LSTM. GRU is the simpler version of LSTM.

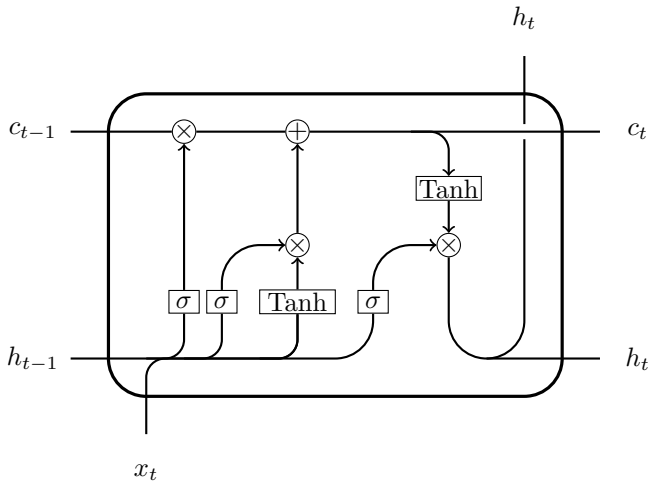


Figure 9: Long-Short Term Memory (LSTM) architecture

Transformer Transformer architecture was introduced from [Vaswani et al., 2017] which is the ancestor of BERT architecture [Devlin et al., 2018]. Transformer has stacks of encoder and decoder structure similar to the seq2seq architecture presented in Figure 10 but instead of using RNN layers, Attention layers are applied. In the existing seq2seq architecture,

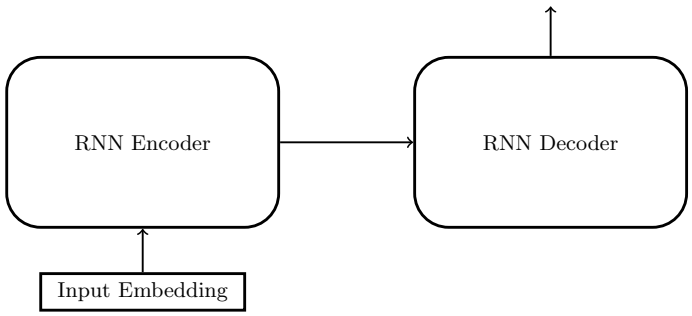


Figure 10: RNN-based seq2seq architecture

the encoder compresses the input sequence into one vector representation, and the decoder creates an output sequence by utilizing the last compressed vector of the encoder. However, there is a disadvantage in that some information of the input sequence is lost in the process of compressing the input sequence into one vector by the encoder if the input sequence is too long. Moreover, RNN based seq2seq architectures have problems of slow learning speed and

parallelization due to sequential computation. Therefore, to overcome this issue, RNN layers of both encoder and decoder are replaced with attention layers in [Vaswani et al., 2017]. Attention layers leverage attention scores to compute the importance of the tokens by comparing them with other tokens simultaneously. Therefore, attention layers are able to increase the learning speed through parallelization without the restrictions appearing in such sequential architectures. Transformer encoder architecture consists of Multi-Head Self-Attention layer, Add and Normalization layer and Feed-Forward layer. Figure 11 shows the architecture of the transformer encoder layer.

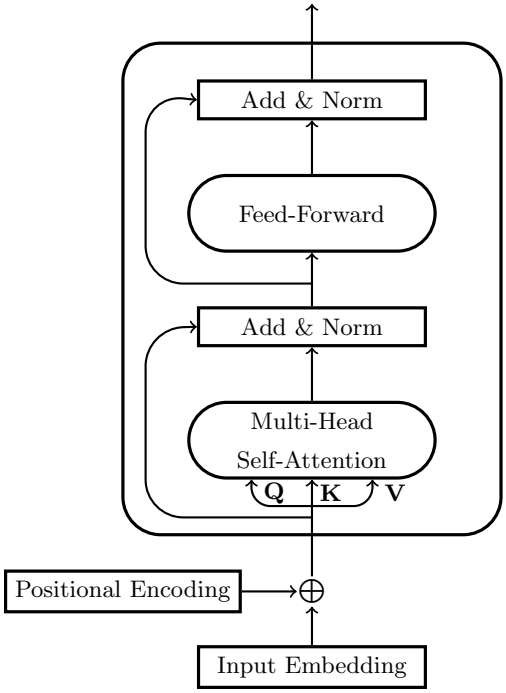


Figure 11: Transformer encoder architecture

RNN based architectures are able to have location information of each word due to the characteristics of RNNs that receive and process the input sequentially. However, the transformer encoder does not process the input sequentially, so there is the need to incorporate positional information. To resolve this problem, the positional encoding was introduced in [Vaswani et al., 2017] which is the vector representation of the positional information of the token by applying Sinusoidal functions which can be represented in continuous float from -1 to 1:

$$PE(t,i) = \begin{cases} sin(t/10000^{2k/d_{model}}), & \text{if i = 2k} \\ cos(t/10000^{2k/d_{model}}), & \text{if i = 2k+1} \end{cases} \tag{2}$$

where the positional encoding vector of position t and the index i of embedding vector is

$PE(t, i)$, d_{model} is the dimension of positional encoding which is the same to the embeddings so it can be summed. By summing the embedding vector and its positional encoding vector, so same words in different positions can be represented as dissimilar vectors.

The transformer encoder proposed in [Vaswani et al., 2017] has a stack of 6 encoding layers and each layer has two sub-layers. The first layer of the sub-layers is a Multi-Head-Self-Attention layer, and the second is a position-wise fully connected feed-forward networks layer. The residual connection is implemented twice in the encoder layer. The first residual connects the input of the encoding layer to the layer normalization process after the Multi-Head Attention layer and the second residual connects the output of normalized output of a Multi-Head attention layer to the layer normalization process after position-wise fully connected feed-forward layer. The input of the encoder layer is the sum of word embedding and positional encoding embedding and other encoders use the output of the previous encoder layer. All the encoders in the transformer are structurally the same while they do not share weights.

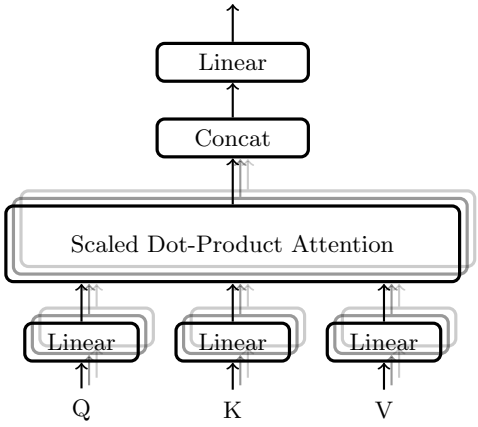


Figure 12: Multi-Head Attention architecture

As shown in Figure 12, the Multi-Head-Self-Attention consists of multiple Scaled Dot-Product Attention. Scaled dot-product attention is the result of computation results of query, key, and value. To understand query, key, and value, suppose the input representation is defined as matrices $X = [x_0, .., x_T]$ where T is the end of the sequence of the input. The query, key and value vectors are the projection of X on learnable linear transformation matrices W^Q, W^K, and W^V.

$$Q = XW_q, \quad K = XW_k, \quad V = XW_v \tag{3}$$

Therefore, the query, key and value vectors are defined as $Q = [q_0, .., q_T]$, $K = [k_0, .., k_T]$, $V = [v_0, .., v_T]$. To sum up, these Q, K, V vectors represent an abstract of the vectors

that are created by multiplying the input embedding by three different weights. The scaled dot-product attention is expressed as:

$$\textit{Scaled Dot-Product Attention}(Q, K, V) = Softmax\left(\frac{Q(K)^T}{\sqrt{d_k}}\right)V \tag{4}$$

where Q, K, V represent the result of dot product of query, key and value vectors and d_k is the size of vector K. The dot product of vector Q and K^T represent the score of how much the current scaled dot-product attention layer has to focus on other words when encoding a word at a certain position. For instance, if the score of $q_t \cdot k_t$ has a higher value than other index values, the scaled dot-product attention layer has to more focus on a word at position t. $\frac{1}{\sqrt{d_k}}$ is a scaling factor. Moreover, to make all scores positive and the sum of all to 1, the $Softmax$ activation function is applied to normalize the scores. The dot product of vector V plays a role in preserving the values of the token that has to be focused, and minimizing the values of the irrelevant tokens by multiplying them with tiny numbers. Therefore, the result of the scaled dot-product attention represents what to focus on by comparing each token to all other tokens. Instead of utilizing a single scaled dot-product attention layer, leveraging multiple scaled dot-product attention layers simultaneously so it computes multiple attention scores in a parallel manner is beneficial [Vaswani et al., 2017]. To perform this multiple self-attention manner, each W^Q, W^K, and W^V are divided by the number of heads and calculate the *Scaled Dot-Product Attention*(Q,K,V) in a parallel manner so individual set of query, key and value weights have different weights. Multi-head Attention is expressed as:

$$\begin{aligned} &\textit{Multi-Head Attention}(Q, K, V) = Concat(Attention_1, ..., Attention_n)W^O \\ &where \quad Attention_i = Attention(Q_i, K_i, V_i) = Attention(xW_i^Q, xW_i^K, xW_i^V) \end{aligned} \tag{5}$$

where n is the number of heads and W^O refers to the learnable weight of multi-head attention and each Attention represents Scaled Dot-Product Attention. The results of each self-attentions are linearly concatenated and so the merged attention score matrix is calculated by taking the dot product of the weight matrix W^O. Therefore, this multi-head attention enables to capture contextual information of similarity importance of tokens, and the self-attention process is not restrained to the direction information like RNN-based architectures so token similarity on both sides can be gathered.

2.2.2 Classifiers

The classifier is an indicator that represents the relationship between true label and input using weight and bias. The process of modifying this weight and bias to fit the classifier

architecture to the given data and true label is learning. Therefore, the classifier architecture is to find the weight and bias that best explain the relationship while learning. Since the task introduced in this thesis is mainly about classifying the relation between two entities, two different classification methods are applied.

Linear classifier The linear transformation is most extensively used classifier technique to solve not only binary classification but also multi-class classification problems. This is because a linear classifier works well in classification tasks and takes less time to train and use and has comparable performance compared to a non-linear classifier a linear classifier [Yuan et al., 2012]. For this reason, the basic classifier applied in this thesis is a linear classifier. A linear transformation is expressed as:

$$f(x_i) = x_i W^T + b \tag{6}$$

where x_i is a vector representations where $x_i \in \mathbb{R}^d$ and b is bias. The W is the weight of the linear classifier where $W \in \mathbb{R}^{l \times d}$ (l is number of labels, d is the size of input vector). As the formula above describes, a linear transformation leverages only one vector to update the weight and accomplish the purpose of the task. Therefore, if more than one vector is selected as an input of a linear classifier, these vectors have to be transformed into a single vector by linear concatenation.

Bilinear classifier The bilinear transformation takes two input vectors simultaneously without concatenation so two input vector shares the same learnable weight. Therefore, unlike a linear transformation, a bilinear transformation layer is able to leverage two vectors without a concatenation process. Due to the characteristic of bilinear transformation, it achieved the state-of-the-art fine-grained image representations by learning fine-grained detail features over a global image feature by computing the interactions between feature channels [Zheng et al., 2019]. Moreover, bilinear transformation as a classifier is widely used in document-level relation extraction tasks ([Wang et al., 2019]; [Yao et al., 2019]; [Tang et al., 2020]). As a consequence, a bilinear transformation is utilized to observe the different consequences of applying linear and bilinear classifiers. A bilinear transformation is expressed as:

$$f(x_i, x_j) = x_i^T W x_j + b \tag{7}$$

where x_i and x_j are two different vectors where $x_i \in \mathbb{R}^{d_1}$ and $x_j \in \mathbb{R}^{d_2}$ and b is bias. The W is a learnable weight of the bilnear transformation where $W \in \mathbb{R}^{l \times d_1 \times d_2}$ (l is number of labels, d_1 and d_2 are the size of input vectors). The major difference between a linear and bilinear transformation is that when leveraging two vectors, each vector interacts with

different elements of the weight vector of the linear transformation during the training while both vectors influence the same elements of the weight vector of bilinear transformation. Therefore, the dot product result of a bilinear transformation is the simultaneous interaction of two vectors to the same elements of the weight of a bilinear transformation.

2.2.3 Activation function

Activation functions are an essential element when using a neural network. The most extensively used activation functions in the neural networks are non-linear. These non-linear activation functions are leveraged to obtain a non-linear mapping of their input into the space of interest. The usage of the activation function in neural networks is to determine whether the output of the neural network system has to be activated or not based on the input and the goal of the system. The raw output of the neural network architecture is called logit in statistics. The range of logit is between negative infinite to positive infinite. Therefore, the activation function is mapping the logit into a limited range and supports the system to learn complex data.

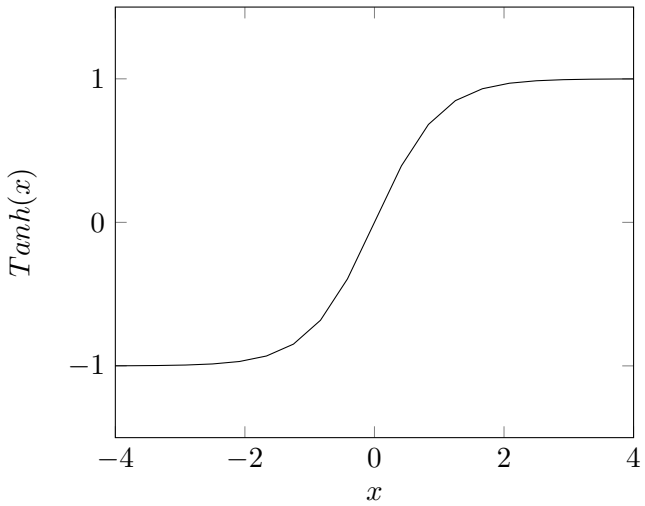


Figure 13: Hyperbolic Tangent

The Hyperbolic Tangent (Tanh) used to be the most extensively used activation function for multi-layer neural networks ([Neal, 1992]; [Karlik and Olgac, 2011]). Tanh transforms the input value to a value of $\{x \in \mathbb{Q} \mid -1 \leq x \leq 1\}$. Therefore, the output of Tanh of the larger input is close to 1.0, whereas the smaller input is close to -1.0. Figure 13 shows the Tanh function. Tanh is defined as:

$$Tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{8}$$

However, Tanh has the vanishing gradient problem which is a crucial problem in training neural networks during backpropagation. The gradient is a partial derivative when updating the weights in the neural network. The vanishing gradient is a problem that the gradient is getting smaller during the process and it approaches zero and vanishes. This vanishing gradient problem is tackled with the introduction of ReLU. ReLU is adopted in deep learning systems as an activation function for the first time in ([Nair and Hinton, 2010]). RELU transforms the input value to a value of $\{x \in \mathbb{Q} \,|\, 0 \leq x \leq 1\}$. Therefore, the output of ReLU of the larger input is the same value as the input of ReLU, whereas the smaller input is close to 0. Figure 13 illustrates ReLU function. ReLU is defined as:

$$ReLU(x) = \begin{cases} x, & \text{if x} > 0 \\ 0, & \text{if x} \leq 0 \end{cases} \tag{9}$$

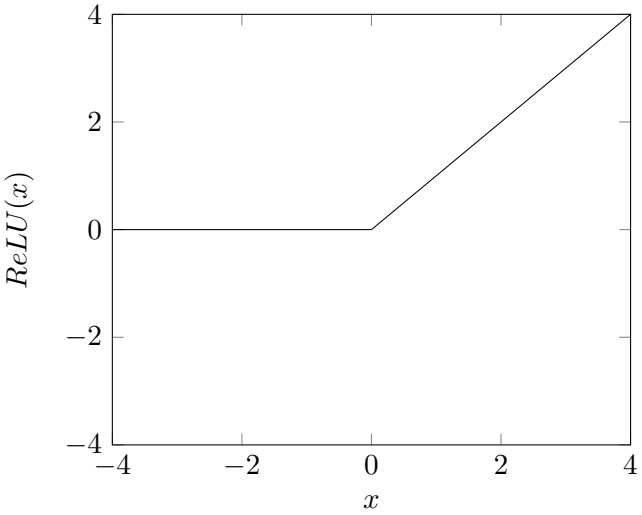


Figure 14: Rectified Linear Unit

2.2.4 Loss Function

The loss function is an indicator that outputs the difference between the measured value, which is the result of a system calculated using a given data set, and the true value. This is an optimization algorithm used in deep learning which is also referred to as cost function, loss function or objective function [Goodfellow et al., 2016]. So, depending on the result of the loss function, it measures how well the specific system fits the given dataset. Cross-Entropy Loss is the most widely used loss function when optimizing both binary and

multi-class classification architectures. To understand the Cross-Entropy Loss, the Cross-Entropy needs to be described prior. Cross-Entropy is calculating the difference between two probability distributions. Cross-Entropy is defined as:

$$CE(t,p) = -\sum_i t_i log(p_j) \tag{10}$$

where t is the distribution of true, p is the distribution of predicted by the system, t_i is the probability of true and p_j is the probability of prediction of index j. In the classification task, t_i is 1 only if the target is the true label so it can be rewritten as:

$$CE(t,p) = -log(p_j) \tag{11}$$

By combining the Softmax function and Cross-Entropy, Cross-Entropy Loss function is defined as:

$$CE_{loss}(x, class) = -log(\frac{exp(x[class])}{\sum_j exp(x[j])}) = -x[class] + log(\sum_j exp(x[j])) \tag{12}$$

where $x[class]$ is the probability of prediction of the true label and normalized by $\sum_j exp(x[j])$ which is the sum of the entire exponent outputs.

2.3 BERT

2.3.1 Various BERT like language models and architectures

The pre-trained language models such as BERT [Devlin et al., 2018] and its successors RoBERTa [Liu et al., 2019], DistilBERT and DistilRoBERTa [Sanh et al., 2019] uses a multi-layer bidirectional transformer encoder-based architecture where the representations are jointly conditioned on both the left-to-right and the right-to-left manner in all transformer encoder layers. The input representation of BERT, RoBERTa, DistilBERT and Distil-RoBERTa is constructed by summing the three embeddings: token embeddings, segment embeddings, and position embeddings. The term *language model* defined in this thesis describes various versions of BERT that have different structural characteristics, pre-training methods or datasets used in pre-training. BERT leverages absolute position embeddings which encode the absolute positions from 1 to maximum sequence length. Therefore, the absolute position embedding assists to differentiate the same token at the different positions by providing different position embedding based on the position of a token. The segment embeddings distinguish the textual input texts by assigning 0 and 1 when two sentences are the input of BERT. The segment embedding assigns 0 to all tokens that belong to the

first sentence, and 1 to all tokens that belong to the second sentence. This segment embedding provides the span information of different sentences, so different sentences are not considered a single sentence.

One of the attributes that set BERT language models apart from previous deep learning architectures such as RNN or LSTM is that BERT applied special tokens [CLS] and [SEP]. [CLS] token represents the classification token that aggregates full textual input information and it is attached prior to the tokenized textual input. [CLS] token is leveraged to the classification tasks during the pre-training and this is also generally used for the downstream tasks. SEP token is designed to separate the two sentences by locating in between for the next sentence prediction tasks.

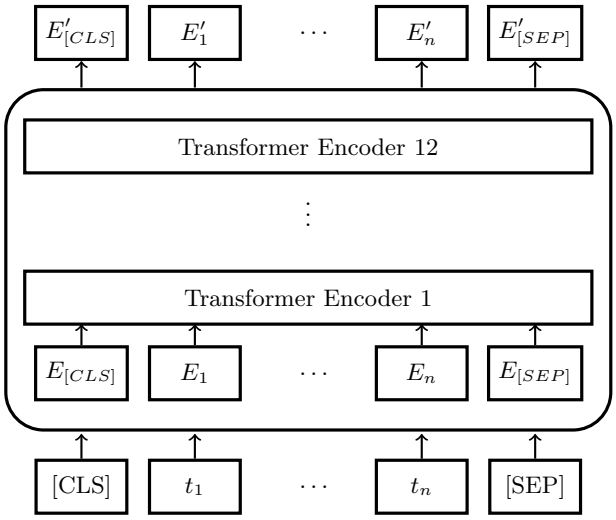


Figure 15: BERT & RoBERTa architecture. E_i is a real-value vector that represents the result of sum of token embedding, position embedding and segment embedding of token t_i. $\hat{E}_i$ refers to the output vector of the last hidden state from BERT of token t_i.

BERT and RoBERTa use the same architecture, shown in Figure 15. The base model of BERT and RoBERTa architecture contains 12 layers of transformer encoder and each of the transformer encoder layers consists of 12 attention heads (the larger model of BERT and RoBERTa contains 24 transformer encoder layers and 16 attention heads). The input of BERT and RoBERTa starts with [CLS] token and the tokenized full textual input follows. The form of input of BERT and RoBERTa starts with [CLS] token, the tokenized full textual input comes after [CLS] token, and SEP token placed after the end of the full textual input. Due to the fact that each embedding is not trained sequentially in a transformer encoder like RNN, positional information is added before the training. Moreover, since BERT also pre-trained the next sentence prediction task, segment information that separates the first

and the second sentence is added. Therefore, the input embedding of the first transformer encoder is the sum of token embedding, position embedding, and segment embedding as presented in Figure 15.

DistilBERT and DistilRoBERTa were proposed due to the challenge of operating large-scale pre-trained language models due to the constrained computational power and budget [Sanh et al., 2019]. To handle this problem, the knowledge distillation ([Bucilu et al., 2006]; [Hinton et al., 2015]) which is a architecture compression technique that trains the small architecture (the student architecture) such as DistilBERT or DistilRoBERTa to mimic the behaviour of the larger architecture (the teacher architecture) such as BERT or RoBERTa. The student architecture has a similar architecture to the teacher architecture in the sense that it also uses the transformer encoder layers. However, the number of layers of the student architecture is decreased to 6 while the teacher architecture uses 12 layers. Also, the segment embeddings and the pooler layer are removed. The architecture of DistilBERT and DistilRoBERTa adopts similar architecture shown in Figure 15 but distilled version contains 6 layers of transformer encoder instead of 12 layers and each of the transformer encoder layer consists of 12 attention heads.

2.3.2 Tokenization & Embedding technique

WordPiece algorithm that is adopted in BERT (also DistilBERT) for the subword segmentation algorithm is initially developed to resolve a Korean and Japanese segmentation problem that many Asian language words cannot be split by space due to its language characteristics for the Google speech recognition system [Schuster and Nakajima, 2012]. WordPiece model is similar to the Byte-Pair Encoding (also called BPE) [Sennrich et al., 2016] in the sense that both of them are data-driven models. However, WordPiece model collects the one that maximizes the likelihood of the training data, while BPE collects the most common symbol pairs when merging characters. WordPiece embedding is trained on the deep LSTM network with 8 encoder and 8 decoder layers using residual connections and the attention connections from the decoder layer to the encoder layer. BPE for subword tokenization algorithm consists of simple steps. The first step is to split the textual input based on the blank space and save these tokens with their frequency. Next, split these tokens into characters and save them with the frequency that is obtained from the first step. So, each character is stored with the word level token frequency and neighboring character within the word level. The next step is to iterate the steps that pairing most frequent tokens and merging them until the size of vocabulary reaches the limit. The iteration step is to select the most frequent character and merge it with the most frequent neighboring character

of the selected character. The neighboring character represents the character that occurs right next to the selected character. Therefore, only if the selected character has the most frequent neighboring character within the word level token, then merge it. The next step is to iterate the iteration step until the vocabulary size reaches the pre-determined size. The purpose of BPE is to ensure the most common words maintain as a single token in the vocabulary while the rare words are broken down into several subword tokens to reduce the size of the vocabulary. WordPiece algorithm is similar to BPE algorithm of building subword vocabulary. To begin with the process, both WordPiece algorithm and BPE split the text data based on the space, and save separated words with their occurrence. Once the space-based splitting process is done, split the saved words into character tokens and group each character that belongs to the original word together. Owing to the fact that both subword segmentation algorithms are designed to achieve an efficient size of vocabulary, these separated characters are needed to be merged. The difference between WordPiece algorithm and BPE occurred during the merging process where WordPiece merges the pair of characters that maximizes the likelihood of the training data while Byte-Pair focused on frequency. Therefore, both algorithms merge tokens repeatedly based on each method and stop once the vocabulary size reached the determined size. Furthermore, in order to avoid OOV problems, single letters are included to handle rare or unseen words.

2.3.3 Pre-training technique & Dataset

BERT is pre-trained for the masked language modeling (MLM) task and the next sentence prediction (NSP) task using BooksCorpus ([Zhu et al., 2015]) and English Wikipedia. The MLM is a pre-training method of BERT-like systems that randomly mask the token and predicts the tokens. The NSP is also a pre-training method of BERT-like systems that classify whether the second sentence is the subsequent sentence of the first sentence when the system receives pairs of sentences. RoBERTa was proposed to improve BERT and enhance its insufficient training by several simple modifications. To put it concretely, RoBERTa was trained longer with much larger batches compared to BERT. Also, RoBERTa used 16GB of dataset (BooksCorpus and English Wikipedia) that BERT originally utilized and additional 144GB dataset (CommonCrawl News dataset [Nagel, 2016], OpenWebText corpus [Gokaslan and Cohen, 2019] and Stories [Trinh and Le, 2018]) for pre-training. The difference between BERT and RoBERTa is with respect to not only the size of the dataset used during the pre-training but also the pre-training method. BERT was pre-trained for both MLM and NSP, RoBERTa was pre-trained for only MLM. RoBERTa was trained with full-length sequences while BERT used randomly inject short sequences and reduced sequence

length. Also, RoBERTa applied a different technique on masking by selecting the dynamic masking compared to BERT where the static masking is applied. Consequently, RoBERTa attained slight improvement compared to BERT on downstream task performance. DistilBERT was pre-trained on the same corpus as the original BERT used while DistilRoBERTa was pre-trained on a 38GB dataset (OpenWebText corpus) which is approximately 4 times less than its teacher architecture (RoBERTa). Table 4 illustrates the difference in architecture and number of parameters on BERT, RoBERTa, DistilBERT and DistilRoBERTa. BERT and RoBERTa utilize the same number of layers, hidden size and number of attention heads but the parameter size is different because the vocabulary size of RoBERTa is 50k while BERT is 30k.

Language Model	**Layers**	**Hidden Size**	**Attention Heads**	**Parameters**
BERT_{base}	12	768	12	110M
RoBERTa_{base}	12	768	12	125M
DistilBERT_{base}	6	768	12	66M
$\text{DistilRoBERTa}_{base}$	6	768	12	82M

Table 4: Description of the architectures and number of parameters of four BERT like language models.

2.4 Relation extraction task description

The relation classification task is to classify the relation from the given data sample which contains the corpus and the span of entities. This thesis uses the SemEval 2010 Task 8 [Hendrickx et al., 2009], TACRED [Zhang et al., 2017] datasets which are widely used, Re-TACRED [Stoica et al., 2021] a revised version of TACRED, and Biocreative VII Track 1 - Text mining drug and chemical-protein interactions(DrugProt) [Miranda et al., 2021] on biomedical documents. Each dataset contains a different number of relation labels and distribution of positive and negative samples. The data samples are divided into positive and negative samples according to the objective of the task. Therefore, if a given data sample satisfies the task, it is classified as a positive sample while it is classified as a negative sample if a given data sample is not suited to the task. As a consequence, the given data sample is classified as a positive sample if the annotated label belongs to the classifying relation labels in the relation extraction task otherwise, it is classified as a negative sample.

2.4.1 SemEval 2010 Task 8: Multi-Way Classification of Semantic Relations Between Pairs of Nominals

SemEval 2010 Task 8 task was designed to classify nine different semantic relations between two nominals and the direction of the relation from the given data sample. Therefore, the nominal is a word or a group of words that is indicated as target entity. SemEval 2010 Task 8 task is the direct successor of Semeval-2007 task 04: Classification of semantic relations between nominals [Girju et al., 2007] where it provides 7 positive relation labels and 1 negative label and each relation has 140 data samples. SemEval 2010 Task 8 task provides 9 positive relation labels and 1 negative label as shown in Table 5. The relation types and definitions of the relations are presented in Table 5.

Relation	Description
Cause-Effect	An event or object leads to an effect
Instrument-Agency	An agent uses an instrument
Product-Producer	A producer causes a product to exist
Content-Container	An object is physically stored in a delineated area of space
Entity-Origin	An entity is coming or is derived from an origin (e.g., position or material)
Entity-Destination	An entity is moving towards a destination
Component-Whole	An object is a component of a larger whole
Member-Collection	A member forms a nonfunctional part of a collection
Message-Topic	A message, written or spoken, is about a topic

Table 5: Definition of relation labels from SemEval 2010 Task 8 dataset.

The SemEval 2010 Task 8 dataset contains 10,717 annotated data samples including 8,000 data samples for training and 2,717 data samples for testing. Table 6 describes the overview of the distribution of relation labels, where 82.6% of the data samples belongs to the positive samples and the rest data samples are negative samples. The negative samples are classified as *Other* in this dataset. The data samples are evenly distributed in both the training and test datasets. Each data sample provides the full text with the span of two nominals *e1* and *e2*, and the corresponding relation label. *e1* and *e2* represent the first entity and the second entity respectively. The pair of nominals from the given data sample corresponds to only one relation where the sequence of two target entity *e1* and *e2* needs to be considered. For instance, if the relation label of the given data sample is *Cause-Effect(e1, e2)*, the *e1* corresponds to the *Cause* and *e2* corresponds to the *Effect* from the relation *Cause-Effect*. Therefore, if the system classified the given data sample relation label as *Cause-Effect(e2, e1)*, the system is indicated as misclassifying the relation label even though the predicted relation itself corresponds to the true label due to the

Relation	Train	Test	Total
Cause-Effect	1003 (12.5%)	328 (12.1%)	1331 (12.4%)
Component-Whole	941 (11.8%)	312 (11.5%)	1253 (11.7%)
Entity-Destination	845 (10.6%)	292 (10.7%)	1137 (10.6%)
Entity-Origin	716 (9.0%)	258 (9.5%)	974 (9.1%)
Product-Producer	717 (9.0%)	231 (8.5%)	948 (8.8%)
Member-Collection	690 (8.6%)	233 (8.6%)	923 (8.6%)
Message-Topic	634 (7.9%)	261 (9.6%)	895 (8.4%)
Content-Container	540 (6.8%)	192 (7.1%)	732 (6.8%)
Instrument-Agency	504 (6.3%)	156 (3.5%)	660 (6.8%)
Other	1410 (17.6%)	454 (16.7%)	1864 (17.4%)
Total	8000 (100%)	2717 (100%)	10717 (100%)

Table 6: Detailed SemEval 2010 Task 8 relations.

sequence of the two target entities. The organizers of the SemEval 2010 Task 8 describe the nominals are generally a single word but some lexicalized terms consist of multiple words such as *television network*, *credit card bill* or *arterial blood pressure*. The official evaluation scoring metric of SemEval 2010 Task 8 is a macro-averaged F1-score for 9 positive relation labels excluding *Other* label.

(6) *The **burst** has been caused by water hammer **pressure**.*
*Relation: **Cause-Effect(e2,e1)***
*e1: **burst***
*e2: **pressure***

In Example (6), *burst* is considered *e1* and *pressure* is considered *e2*. The relation *Cause-Effect(e2,e1)* describes *burst* and *pressure* are related through the *Cause-Effect* relation where *pressure* represents the cause the effect *burst* from the given data sample.

2.4.2 TACRED (Text Analysis Conference Relation Extraction Dataset)

TACRED is the most well-known and the largest relation classification dataset. This dataset is designed to classify 41 different semantic relations between two nominals from the given data sample. TACRED contains 41 positive relation labels and 1 negative relation label. The TACRED organizers described the SemEval 2010 Task 8 dataset as suboptimal because of two reasons: small in size and vagueness of relations. To resolve the shortcoming of SemEval 2010 Task 8 dataset, the TACRED organizers focused on large-scale and real-world corpus dataset. TACRED dataset was built over years of TAC KBP (Text Analysis

Relation	Train	Dev	Test	Total
org:alternate_names	808 (1.2%)	338 (1.5%)	213 (1.4%)	1359 (1.28%)
org:city_of_headquarters	382 (0.6%)	109 (0.5%)	82 (0.5%)	573 (0.54%)
org:country_of_headquarters	468 (0.7%)	177 (0.8%)	108 (0.7%)	753 (0.71%)
org:dissolved	23 (0.0%)	8 (0.0%)	2 (0.0%)	33 (0.03%)
org:founded	91 (0.1%)	38 (0.2%)	37 (0.2%)	166 (0.16%)
org:founded_by	124 (0.2%)	76 (0.3%)	68 (0.4%)	268 (0.25%)
org:member_of	122 (0.2%)	31 (0.1%)	18 (0.1%)	171 (0.16%)
org:members	170 (0.2%)	85 (0.4%)	31 (0.2%)	286 (0.27%)
org:number_of_employees/members	75 (0.1%)	27 (0.1%)	19 (0.1%)	121 (0.11%)
org:parents	286 (0.4%)	96 (0.4%)	62 (0.4%)	444 (0.42%)
org:political/religious_affiliation	105 (0.2%)	10 (0.0%)	10 (0.1%)	125 (0.12%)
org:shareholders	76 (0.1%)	55 (0.2%)	13 (0.1%)	144 (0.14%)
org:stateorprovince_of_headquarters	229 (0.3%)	70 (0.3%)	51 (0.3%)	350 (0.33%)
org:subsidiaries	296 (0.4%)	113 (0.5%)	44 (0.3%)	453 (0.43%)
org:top_members/employees	1890 (2.8%)	534 (2.4%)	346 (2.2%)	2770 (2.61%)
org:website	111 (0.2%)	86 (0.4%)	26 (0.2%)	223 (0.21%)
per:age	390 (0.6%)	243 (1.1%)	200 (1.3%)	833 (0.78%)
per:alternate_names	104 (0.2%)	38 (0.2%)	11 (0.1%)	153 (0.14%)
per:cause_of_death	117 (0.2%)	168 (0.7%)	52 (0.3%)	337 (0.32%)
per:charges	72 (0.1%)	105 (0.5%)	103 (0.7%)	280 (0.26%)
per:children	211 (0.3%)	99 (0.4%)	37 (0.2%)	347 (0.33%)
per:cities_of_residence	374 (0.5%)	179 (0.8%)	189 (1.2%)	742 (0.70%)
per:city_of_birth	65 (0.1%)	33 (0.1%)	5 (0.0%)	103 (0.10%)
per:city_of_death	81 (0.1%)	118 (0.5%)	28 (0.2%)	227 (0.21%)
per:countries_of_residence	445 (0.7%)	226 (1.0%)	148 (1.0%)	819 (0.77%)
per:country_of_birth	28 (0.0%)	20 (0.1%)	5 (0.0%)	53 (0.05%)
per:country_of_death	6 (0.0%)	46 (0.2%)	9 (0.1%)	61 (0.06%)
per:date_of_birth	63 (0.1%)	31 (0.1%)	9 (0.1%)	103 (0.10%)
per:date_of_death	134 (0.2%)	206 (0.9%)	54 (0.3%)	394 (0.37%)
per:employee_of	1524 (2.2%)	375 (1.7%)	264 (1.7%)	2163 (2.04%)
per:origin	325 (0.5%)	210 (0.9%)	132 (0.9%)	667 (0.63%)
per:other_family	179 (0.3%)	80 (0.4%)	60 (0.4%)	319 (0.30%)
per:parents	152 (0.2%)	56 (0.2%)	88 (0.6%)	296 (0.28%)
per:religion	53 (0.1%)	53 (0.2%)	47 (0.3%)	153 (0.14%)
per:schools_attended	149 (0.2%)	50 (0.2%)	30 (0.2%)	229 (0.22%)
per:siblings	165 (0.2%)	30 (0.1%)	55 (0.4%)	250 (0.24%)
per:spouse	258 (0.4%)	159 (0.7%)	66 (0.4%)	483 (0.45%)
per:stateorprovince_of_birth	38 (0.1%)	26 (0.1%)	8 (0.1%)	72 (0.07%)
per:stateorprovince_of_death	49 (0.1%)	41 (0.2%)	14 (0.1%)	104 (0.10%)
per:stateorprovinces_of_residence	331 (0.5%)	72 (0.3%)	81 (0.5%)	484 (0.46%)
per:title	2443 (3.6%)	919 (4.1%)	500 (3.2%)	3862 (3.63%)
no_relation	55112 (80.9%)	17195 (76.0%)	12184 (78.6%)	84491 (79.51%)
Total	68124 (100%)	22631 (100%)	15509 (100%)	106264 (100%)

Table 7: Detailed TACRED relations.

Conference Knowledge Base Population) challenges. TACRED contains 41 relation labels which occur between person, organization, date, city, country, and so on and *no_relation*

label. The training dataset of TACRED is based on TAC KBP 2009-2012, the validation dataset is based on TAC KBP 2013 and the test dataset is based on TAC KBP 2014. TACRED contains 106,264 annotated data samples including 68,124 data samples for training, 22,631 data samples for validation and 15,509 data samples for testing. Table 7 describes the overview of the distribution of relation labels, where 20.5% of the data samples belong to the positive samples and 79.5% of the data samples are negative samples. The negative samples are classified as *no_relation* label in this dataset. Unlike SemEval 2010 Task 8, TACRED indicates two target entities as a form of subject and object, not *e1* and *e2*. Therefore, for the consistent input representation, the subject and object are represented as *e1* and *e2*. The official evaluation scoring metric of TACRED is a micro-averaged F1-score for 41 positive relation labels excluding *no_relation* label.

(7) ***Carson**, 33, has been a member of the **Indianapolis City-County Council** since August.*
Relation: ***per:employee_of***
e1: ***Carson***
e2: ***Indianapolis City-County Council***

In Example (7), the relation *per:employee_of* describes *Carson* is the type of person and *Carson* is an employee of *Indianapolis City-County Council*.

2.4.3 Re-TACRED (Revised Text Analysis Conference Relation Extraction Dataset)

Re-TACRED is the re-annotated version of TACRED to reform shortcomings of existing TACRED such as incorrectly annotated subject and/or object, ambiguous and inconsistent relation definitions, miscellaneous data samples which are not written in English, and partial span of entities. Re-TACRED organizers refined ambiguous and inconsistent relation definitions, removed data samples that are not written in English, and partially annotated subject or object data samples. Therefore, Re-TACRED contains 39 different semantic relations to be classified between two nominals from the given data sample instead of 41 relations in TACRED. Re-TACRED contains 39 positive relation labels and 1 negative relation label. Re-TACRED consists of 91,467 annotated data samples including 58,465 data samples for training, 19,584 data samples for validation and 13,418 data samples for testing. Table 8 describes the overview of the distribution of relation labels, where 36.8% of the data belong to positive samples and the rest 63.2% of the data are negative samples (no_relation).

Relation	Train	Dev	Test	Total
org:alternate_names	1319 (2.3%)	440 (2.2%)	337 (2.5%)	2096 (2.3%)
org:city_of_branch	622 (1.1%)	191 (1.0%)	129 (1.0%)	942 (1.0%)
org:country_of_branch	891 (1.5%)	338 (1.7%)	166 (1.2%)	1395 (1.5%)
org:dissolved	23 (0.0%)	7 (0.0%)	5 (0.0%)	35 (0.0%)
org:founded	80 (0.1%)	36 (0.2%)	34 (0.3%)	150 (0.2%)
org:founded_by	107 (0.2%)	76 (0.4%)	84 (0.6%)	267 (0.3%)
org:member_of	365 (0.6%)	88 (0.4%)	64 (0.5%)	517 (0.6%)
org:members	560 (1.0%)	194 (1.0%)	63 (0.5%)	817 (0.9%)
org:number_of_employees/members	54 (0.1%)	27 (0.1%)	13 (0.1%)	94 (0.1%)
org:political/religious_affiliation	190 (0.3%)	34 (0.2%)	29 (0.2%)	253 (0.3%)
org:shareholders	93 (0.2%)	80 (0.4%)	12 (0.1%)	185 (0.2%)
org:stateorprovince_of_branch	315 (0.5%)	98 (0.5%)	57 (0.4%)	470 (0.5%)
org:top_members/employees	1475 (2.5%)	462 (2.4%)	295 (2.2%)	2232 (2.4%)
org:website	119 (0.2%)	94 (0.5%)	30 (0.2%)	243 (0.3%)
per:age	421 (0.7%)	256 (1.3%)	208 (1.6%)	885 (1.0%)
per:cause_of_death	114 (0.2%)	193 (1.0%)	50 (0.4%)	357 (0.4%)
per:charges	87 (0.1%)	130 (0.7%)	126 (0.9%)	343 (0.4%)
per:children	275 (0.5%)	114 (0.6%)	55 (0.4%)	444 (0.5%)
per:cities_of_residence	188 (0.3%)	98 (0.5%)	125 (0.9%)	411 (0.4%)
per:city_of_birth	88 (0.2%)	43 (0.2%)	15 (0.1%)	146 (0.2%)
per:city_of_death	120 (0.2%)	148 (0.8%)	26 (0.2%)	294 (0.3%)
per:countries_of_residence	201 (0.3%)	192 (1.0%)	148 (1.1%)	541 (0.6%)
per:country_of_birth	25 (0.0%)	24 (0.1%)	0 (0.0%)	49 (0.1%)
per:country_of_death	6 (0.0%)	55 (0.3%)	14 (0.1%)	75 (0.1%)
per:date_of_birth	69 (0.1%)	31 (0.2%)	7 (0.1%)	107 (0.1%)
per:date_of_death	172 (0.3%)	234 (1.2%)	63 (0.5%)	469 (0.5%)
per:employee_of	2136 (3.7%)	576 (2.9%)	332 (2.5%)	3044 (3.3%)
per:identity	5320 (9.1%)	2293 (11.7%)	2036 (15.2%)	9649 (10.5%)
per:origin	295 (0.5%)	222 (1.1%)	115 (0.9%)	632 (0.7%)
per:other_family	105 (0.2%)	34 (0.2%)	52 (0.4%)	191 (0.2%)
per:parents	182 (0.3%)	69 (0.4%)	106 (0.8%)	357 (0.4%)
per:religion	74 (0.1%)	80 (0.4%)	59 (0.4%)	213 (0.2%)
per:schools_attended	142 (0.2%)	46 (0.2%)	33 (0.2%)	221 (0.2%)
per:siblings	211 (0.4%)	33 (0.2%)	66 (0.5%)	310 (0.3%)
per:spouse	271 (0.5%)	189 (1.0%)	73 (0.5%)	533 (0.6%)
per:stateorprovince_of_birth	44 (0.1%)	34 (0.2%)	9 (0.1%)	87 (0.1%)
per:stateorprovince_of_death	58 (0.1%)	44 (0.2%)	16 (0.1%)	118 (0.1%)
per:stateorprovinces_of_residence	261 (0.4%)	37 (0.2%)	73 (0.5%)	371 (0.4%)
per:title	2626 (4.5%)	998 (5.1%)	523 (3.9%)	4147 (4.5%)
no_relation	38761 (66.3%)	11246 (57.4%)	7770 (57.9%)	57777 (63.2%)
Total	58465 (100%)	19584 (100%)	13418 (100%)	91467 (100%)

Table 8: Detailed Re-TACRED relations.

2.4.4 Biocreative VII Track 1 - Text mining drug and chemical-protein interactions (DrugProt)

Biocreative VII Track 1 task is to aim automatically detect the relationship between chemical compounds/drugs and genes/proteins from PubMed titles and abstracts. The dataset

is hand-annotated by domain experts who have experience with biomedical documents.

Biocreative VII Track 1 provides three separate files regarding the task. The first file includes title and abstracts of biomedical document with document id. The second file contains mentioned type and span of chemical compounds/drugs and genes/proteins document id. The third file presents the relationship of chemical compounds/drugs and genes/proteins with document id. The original goal of the task is not only classifying the biologically relevant relation between chemical compounds/drugs and genes/proteins but also identifying the span of entities that form relationships from the title and abstract. The given data sample contains multiple sentences of the title and abstract, and the span of entities is provided in training and development dataset. An example from one of the Biocreative VII Track 1 dataset is given in Example (8).

(8) *Hypoxemia associated with cimetidine therapy in a newborn infant. Cimetidine therapy used for the treatment of gastric bleeding due to tolazoline therapy in a newborn infant was temporally associated with episodes of severe hypoxemia. It appears likely that the histamine H2 receptor blocked by cimetidine obviated the pulmonary vasodilator effect of tolazoline therapy.*

The highlighted spans from the above example represent chemical compounds or drugs and the underlined span represents genes or proteins. The gold label of Example (8) is *INHIBITOR* between *histamine H2 receptor* and *cimetidine* of the third sentence. Compared to SemEval 2010 Task 8, TACRED and Re-TACRED, Biocreative VII Track 1 dataset has distinctive characteristics that 1) each sentence of title and abstracts does not necessarily contains chemical compounds/drugs or genes/proteins, 2) the span of chemical compounds/drugs and genes/proteins may overlap, and 3) the chemical compounds/drugs and genes/proteins in a sentence may include many-to-many relations if there are more than one pair of entities in the same sentence. Therefore, Biocreative VII Track 1 dataset format is different from the SemEval 2010 Task 8, TACRED and Re-TACRED that each data sample includes only one pair of entities that may or may not form a relationship. For this reason, in order to fit the task to the purpose of this thesis, the task is simplified to detect the relationship of biomedical entities. Therefore, the processed dataset that is applied in this thesis only collects the sentence that contains biomedical relationships and two target entities are given in the sentence where the chemical compounds/drugs and genes/proteins are not overlapped. The first step of pre-processing is to collect the sentence if the sentence includes both chemical compounds/drugs and genes/proteins. For instance, the third sentence of Example (8) is collected. After this, forming data samples based on the pair of chemical compounds/drugs and genes/proteins that are not overlapping a word or a group of words. Therefore, the pre-processed data samples of Example (8) are presented

in Example (9).

(9)
- *It appears likely that the histamine H2 receptor blocked by cimetidine obviated the pulmonary vasodilator effect of tolazoline therapy.*
- *It appears likely that the histamine H2 receptor blocked by cimetidine obviated the pulmonary vasodilator effect of tolazoline therapy.*

The first data sample of Example (9) has a relation of *INHIBITOR* between *histamine H2 receptor* and *cimetidine* and the second data sample of Example (9) has *no_relation* relation between *histamine H2 receptor* and *tolazoline* according to the gold standard. As a result, the pre-processed Biocreative VII Track 1 dataset includes the same sentences with a different span of entities and relations and disregards the sentences if at least a pair of chemical compounds/drugs and genes/proteins that do not overlap one another does not exist in the sentence.

Relation	**Train**	**Dev**	**Total**
ACTIVATOR	1374 (2.10%)	243 (1.80%)	1617 (2.05%)
AGONIST	641 (0.98%)	131 (0.97%)	772 (0.98%)
AGONIST-ACTIVATOR	29 (0.04%)	10 (0.07%)	39 (0.05%)
AGONIST-INHIBITOR	12 (0.02%)	2 (0.01%)	14 (0.02%)
ANTAGONIST	949 (1.45%)	215 (1.60%)	1164 (1.48%)
DIRECT-REGULATOR	2193 (3.35%)	442 (3.28%)	2635 (3.34%)
INDIRECT-DOWNREGULATOR	1317 (2.01%)	308 (2.29%)	1625 (2.06%)
INDIRECT-UPREGULATOR	1371 (2.10%)	301 (2.23%)	1672 (2.12%)
INHIBITOR	5338 (8.16%)	1148 (8.52%)	6486 (8.22%)
PART-OF	882 (1.35%)	254 (1.88%)	1136 (1.44%)
PRODUCT-OF	915 (1.40%)	157 (1.16%)	1072 (1.36%)
SUBSTRATE	1993 (3.05%)	493 (3.66%)	2486 (3.15%)
SUBSTRATE_PRODUCT-OF	24 (0.04%)	3 (0.02%)	27 (0.03%)
no_relation	48347 (73.94%)	9771 (72.5%)	58118 (73.69%)
Total	65385 (100%)	13478 (100%)	78863 (100%)

Table 9: Detailed Biocreative VII Track 1 relations.

Biocreative VII Track 1 contains 13 different relationships with the pair of certain biomedical entities (chemical compounds/drugs and genes/proteins). The distribution of the collected data samples are provided in Table 9. The official evaluation metric of Biocreative VII Track 1 is micro-averaged scores of f-measure, precision and recall, so this evaluation metric is adopted in this thesis. Due to the timeline of Biocreative VII Track 1 that gold annotation of the test dataset is not revealed by the time this experiment was conducted, the development dataset was applied to measure the performance of the system.

For the consistency of defining entities, the chemical compounds and drugs are represented as $e1$, and the genes and proteins are represented as $e2$.

(10) *It appears likely that the* ***histamine H2 receptor*** *blocked by* ***cimetidine*** *obviated the pulmonary vasodilator effect of tolazoline therapy.*
Relation: ***INHIBITOR***
e1: ***histamine H2 receptor***
e2: ***cimetidine***

In Example (10), the relation *INHIBITOR* describes the chemical compounds *cimetidine* inhibits the gene *histamine H2 receptor*.

Chapter 3

Approaches and Experiment results

3.1 Input & Output Representation

This section describes various input & output representation methods and their brief experiment results. For instance, the application of entity markers that enclose entities (*Input type* module), the usage of various output vectors of BERT (*Output type* module) and the utilization of attached redundant entities next to the context (*Input format* module). In this thesis, the term *module* refers to various methods of representing textual input and leveraging output vectors. Since the integrated experiment results table that includes the entire experiment results of modules will be covered in Chapter 4, the experiment results on varied modules are briefly described by comparing the result of the baseline model in this section. The experimental results utilized $BERT_{base}$ language model.

3.1.1 Experiments with different input encodings

Relation Extraction tasks performed in this thesis provide the span of entities. A relation is assigned based on the indicated entities so entities are potentially one of the most decisive information when classifying the relation. To investigate the significance of the entity markers that enclose the entity, two input encoding methods `No Markers` and `Entity Markers` is introduced in this thesis. These two modules belong to *Input type* to differentiate with other input & output representation modules. *Input type* represents the form of tokenized textual input, especially whether entities are enclosed by the entity markers or not. [Soares et al., 2019] applied these input encoding methods on relation extraction tasks but the experiment results were based on the development dataset so it was challenging

to compare with other input & output representation modules. In order to investigate the importance of indication of the span of entities, the extended experiments focused on two different input type modules.

No Markers This is the baseline input type module. `No Markers` module does not introduce markers that indicate the span of entities. This is a baseline input type module because it provides the potential range of improvement. Therefore, BERT is required to identify where to concentrate on without the information of what entities are, and the classifier needs to classify the relation based on the result of BERT. Example (11) describes the input of `No Markers` module.

(11) *The water was in a cup.*

Entity Markers `Entity Markers` adds entity markers (<e1>, </e1> and <e2>, </e2>) in order to enclose entities as in Example (12). These entity markers are metalanguage tokens and will not be subtokenized by BERT. These entity markers possibly provide consistent information of where to focus to BERT. The example input of `Entity Markers` is described in Example (12).

(12) *The <e1> water </e1> was in a <e2> cup </e2> .*

Module	**SemEval 2010**			**TACRED**			**Re-TACRED**			**Biocreative VII**		
Input type	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
`NM`	78.0	78.1	78.0	28.0	13.1	17.8	39.1	37.8	38.4	53.3	28.2	36.9
`EM`	86.2	87.4	**86.6**	66.2	64.1	**65.1**	87.2	85.7	**86.4**	71.1	74.5	**72.7**

Table 10: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets. `NM` and `EM` represents `No Markers` and `Entity Markers` respectively.

Table 10 illustrates the experiment results of `NM` and `EM` modules on various relation extraction tasks. P, R and F represent Precision, Recall and F1-score of macro averaged results on SemEval 2010 Task 8 test sets and micro averaged results on TACRED and Re-TACRED test sets, and Biocreative VII Track 1 dev sets respectively. A linear classifier is selected as a classifier in this experiment and the selected output vector of the last hidden state of BERT is CLS. The experiment result of `Entity Markers` module shows dramatically increased performance on both precision and recall compared to `No Markers` on all tasks.

3.1.2 Experiments with different output encodings

The pre-trained [CLS] token is a special token that is placed before the tokenized textual input ([Devlin et al., 2018]). CLS vector is applied in MLM and NSP during the pre-training, where MLM is a token prediction task and NSP is a sentence pairing prediction task. [Devlin et al., 2018] suggested that CLS vector may represent the aggregation of the entire textual input so CLS vector of the last hidden state of BERT is extensively used as an input to the classifier in downstream classification tasks. However, this vector might not be a promising choice to classify the semantic relation between entities. This is because BERT is pre-trained on MLM and NSP but not on relation extraction tasks such that requires the simultaneous information of entities (token-level) and text that forms a relation of entities (sentence-level). For this reason, various output vectors from the last hidden layer of BERT are applied in this experiment in order to investigate the significance of the choice of output vectors on relation extraction tasks. The *Output type* represents the choice of output vectors from the last hidden state of BERT. The attempt of diversifying the choice of output vectors instead of selecting only CLS vector output improved performance on relation extraction tasks. [Soares et al., 2019] investigates the significance of output vectors by comparing the experimental results of the CLS and entity vectors on SemEval 2010 Task 8 and TACRED. [Wu and He, 2019] applied the concatenation of CLS and entity vectors to the input of a linear classifier and obtained improved performance on SemEval 2010 Task 8. [Tao et al., 2019] achieved a prominent performance by applying attached pre-processed tokens between entities after the full context and leveraged CLS vector, entity vectors and the attached vector to a linear classifier. However, most of these studies are highly focused on leveraging CLS and entity vectors. For this reason, this thesis presents a set of ablation studies and the comparison of each approach. In order to describe output type modules, suppose a given data sample consists of a sequence of words and the corresponding word embeddings are represented as $< w_1, \ldots, w_s >$ where s is the number of tokens to a given data sample. $< w_{[CLS]}, w_1, ..., w_n >$ represents input of BERT including special tokens ([CLS], [SEP], and entity markers) of a given data sample. $< h_{[CLS]}, h_1 ..., h_n >$ represents the last hidden state vectors of BERT of corresponding word embeddings.

[CLS] This is the baseline output type module that utilizes only CLS vector from the last hidden layer of BERT as input to a linear classifier. The example output representation of `[CLS]` module is depicted in Figure 16. Table 10 illustrates the baseline output type module result of both `No Markers` and `Entity Markers`.

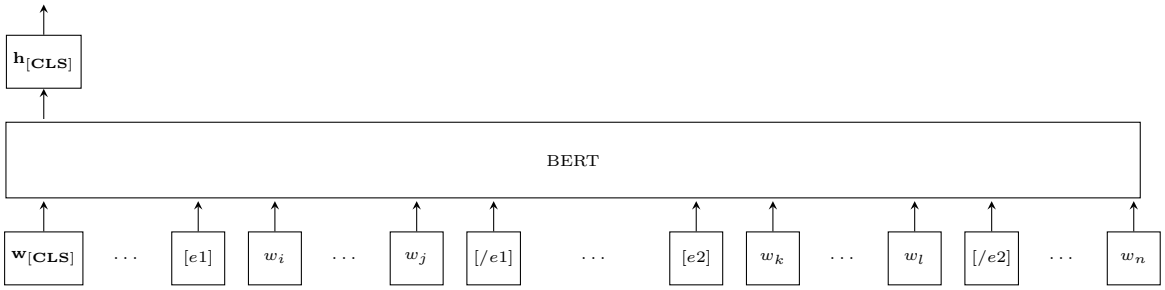


Figure 16: Output type module `[CLS]`

Es Instead of applying CLS vector in classification tasks, vectors of each entity `e1` and `e2` which are the object to identify the relation were applied in this module. This is because the objective of the tasks is to identify the relationship between entities rather than the task of requiring the information of the aggregated sentence level. The word embeddings of the span of the first entity is defined as $< w_i, ...w_j >$ and the span of the second entity is defined as $< w_k, ...w_l >$. Since entities are specified as a word or a group of words, and subword tokenization may sub-divide those entities into several tokens, the entity vector of each is applied as an input of a classifier by averaging each entity. Therefore, the last hidden state vector of the first entity is defined as $h_{e_1} = AVG([h_i, ..., h_j])$ and the last hidden state vector of the second entity is defined as $h_{e_2} = AVG([h_k, ..., h_l])$ where AVG represents average. The linearly concatenated h_{e_1} and h_{e_2} vector is the input of a linear classifier. The example output representation of `Es` module is represented in Figure 17. As

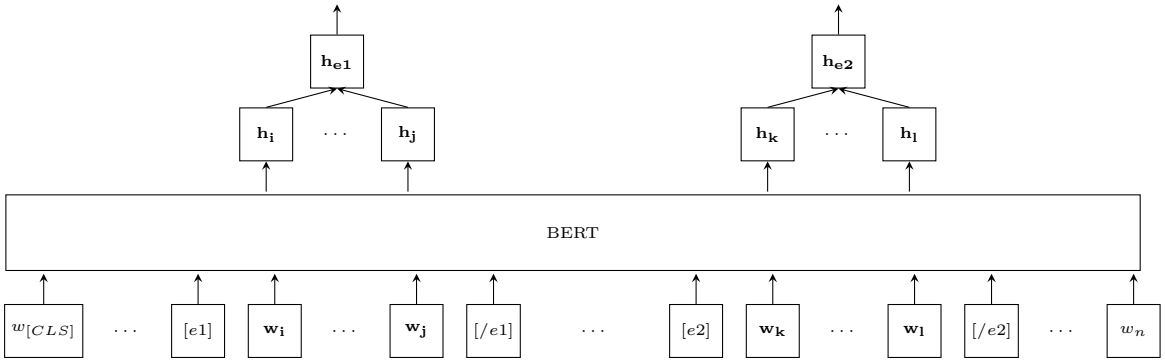


Figure 17: Output type module `Es`

depicted in Table 11, leveraging entity vectors instead of CLS vector when entity markers are not included dramatically improves their performance on every task. Moreover, leveraging entity vectors improves recall on `Entity Markers` in most cases.

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**OT**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
NM	[CLS]	78.0	78.1	78.0	28.0	13.1	17.8	39.1	37.8	38.4	53.3	28.2	36.9
NM	Es	84.9	84.7	84.7	64.9	45.8	53.7	81.5	74.5	77.8	72.5	66.6	69.5
EM	[CLS]	86.2	87.4	**86.6**	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
EM	Es	84.6	89.0	86.4	69.2	64.1	**66.6**	88.0	87.7	**87.8**	72.8	75.2	**74.0**

Table 11: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to `Input type` and `Output type`. Each `NM` and `EM` represents `No Markers` and `Entity Markers`.

[CLS]+Es The purpose of `[CLS]+Es` module is leveraging not only the sentence-level gathered information by adding CLS vector but also the token-level representation of entities by including entity vectors. Therefore, the classifier classifies the relationship based on two different levels of information. The linearly concatenated vector of [CLS] and entities is an input of a linear classifier. The example output representation of `[CLS]+Es` module is illustrated in Figure 18. The performance results of `[CLS]`, `Es` and `[CLS]+Es` module on `No`

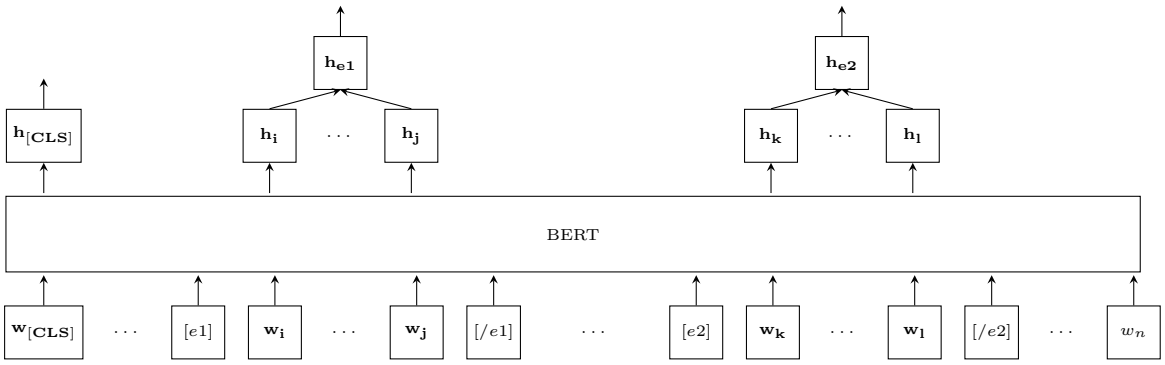


Figure 18: Output type module `[CLS]+Es`

`Markers` and `Entity Markers` input type modules are presented in Table 12. The experiment shows the `[CLS]+Es` module enhances both precision and recall on `No Markers` input type module compared to `[CLS]` module. Moreover, `[CLS]+Es` module yield improved precision compared to *Es* module in most cases. Overall, `[CLS]+Es` module acquires enhanced precision on `Entity Markers` than `No Markers`.

Es+mid `Es+mid` module is focused on the relationship between entities and related tokens which are closely connected between entities. The `mid` represents the vector that appeared between entities because of the varied number of the tokens that appear between entities.

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**OT**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
NM	[CLS]	78.0	78.1	78.0	28.0	13.1	17.8	39.1	37.8	38.4	53.3	28.2	36.9
NM	Es	84.9	84.7	84.7	64.9	45.8	53.7	81.5	74.5	77.8	72.5	66.6	69.5
NM	[CLS]+Es	86.2	83.8	84.9	67.3	50.7	57.8	82.1	71.4	76.4	72.1	67.5	69.7
EM	[CLS]	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
EM	Es	84.6	89.0	86.4	69.2	64.1	**66.6**	88.0	87.7	**87.8**	72.8	75.2	**74.0**
EM	[CLS]+Es	91.0	85.0	**87.8**	72.9	61.0	66.4	87.9	86.6	87.2	77.0	71.2	**74.0**

Table 12: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to `Input type` and `Output type`. Each `NM` and `EM` represents `No Markers` and `Entity Markers`.

The last hidden state representation h_{mid} is the aggregation of the last hidden state vectors placed between the end of the enclosed marker of the first entity and the start of the enclosed marker of the second entity. Therefore, the span of `mid` is defined as $< w_{j+2}, ..., w_{k-2} >$ and the last hidden state vector of `mid` is defined as $h_{mid} = AVG([h_{j+2}, ..., h_{k-2}])$. The input of the classifier of `Es+mid` module is linearly concatenated vector of entities and mid. The example output representation of output type **Es+mid** is depicted in Figure 19. In

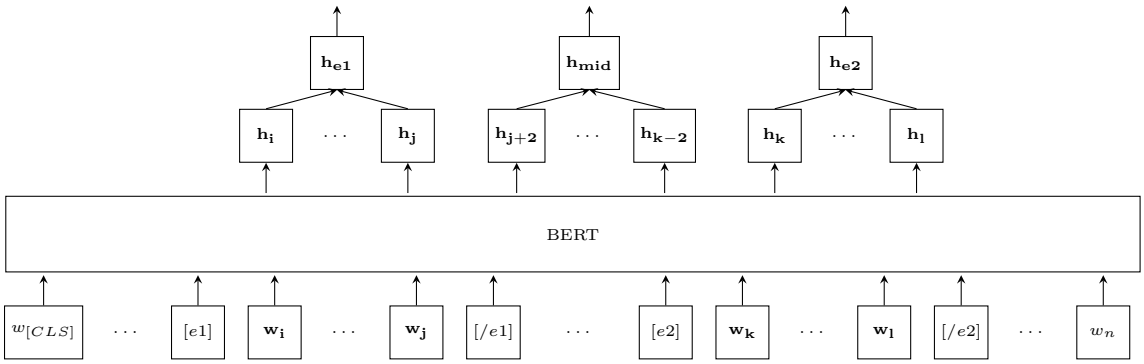


Figure 19: Output type module `Es+mid`

Table 13, the experiment result of `Es+mid` displays the improved precision compared to `[CLS]` and `Es` on SemEval 2010, TACRED and Biocreative VII. The precision of `Es+mid` on Re-TACRED shows somewhat declined performance compared to `Es` but it still yields better precision compared to `[CLS]`. While most of the experiment results show betterment on precision after adding `mid` at the classification step, most of the task results output dropped recall except on SemEval 2010 Task 8.

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**OT**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
EM	[CLS]	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
EM	Es	84.6	89.0	86.4	69.2	64.1	**66.6**	88.0	87.7	**87.8**	72.8	75.2	**74.0**
EM	Es+mid	87.3	89.5	**88.3**	72.8	60.8	66.2	87.8	86.8	87.3	76.0	69.9	72.8

Table 13: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to `Input type` and `Output type`. `EM` refers to `Entity Markers`.

Es+post The purpose of the `Es+post` module is to observe the relationship between entities and tokens appeared after the second entity. The `post` represents the tokens appeared after the second entity. Since the tokens after the second entity is varied based on the given data sample, the last hidden state representation h_{post} is the aggregation of the last hidden state vectors placed after the second entity to the end of the sentence. Therefore, the span of `post` is defined as $< w_{l+2}, ...w_n >$ and the last hidden state vector of `post` is defined as $h_{post} = AVG([h_{l+2}, ..., h_n])$. The input of the classifier of `Es+post` module is linearly concatenated vector of entities and post. The example output representation of `Es+post` module is represented in Figure 20. Table 14 shows that including `post` enhances precision

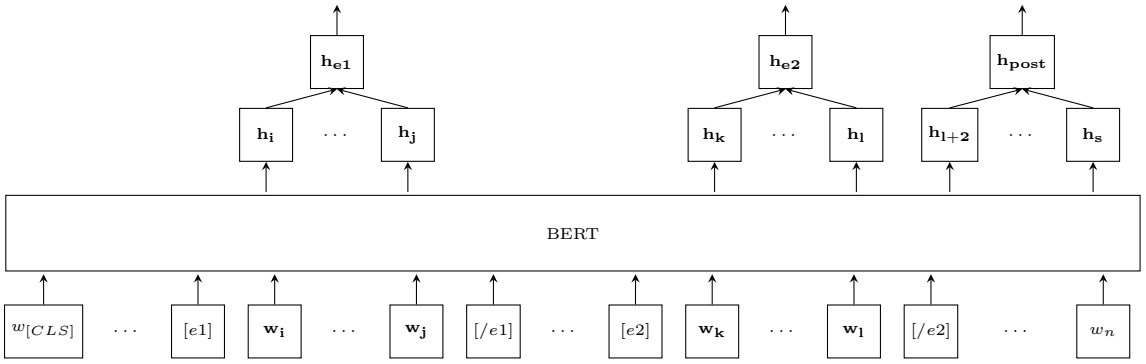


Figure 20: Output type module `Es+post`

on most of the task compared to `[CLS]` and `Es` module. However, similar to the result of adding `mid`, the lowered recall is detected on TACRED and Biocreative VII Track 1 when applying `Es+post`.

Es+mid+post Instead of utilizing one of the chunk of the information related to the entities (`mid` and `post`), this output type module leverages both `mid` and `post` with entities to investigate the synergy. The input of the classifier of `Es+post` module is linearly

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**OT**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
EM	[CLS]	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
EM	Es	84.6	89.0	86.4	69.2	64.1	**66.6**	88.0	87.7	**87.8**	72.8	75.2	**74.0**
EM	Es+post	88.4	88.7	**88.5**	69.8	60.3	64.7	87.9	87.3	87.6	77.0	69.5	73.1

Table 14: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to `Input type` and `Output type`. `EM` refers to `Entity Markers`.

concatenated vector of entities, mid and post. The example output representation of `Es+mid+post` module is illustrated in Figure 21. Table 15 shows similar patterns of the ex-

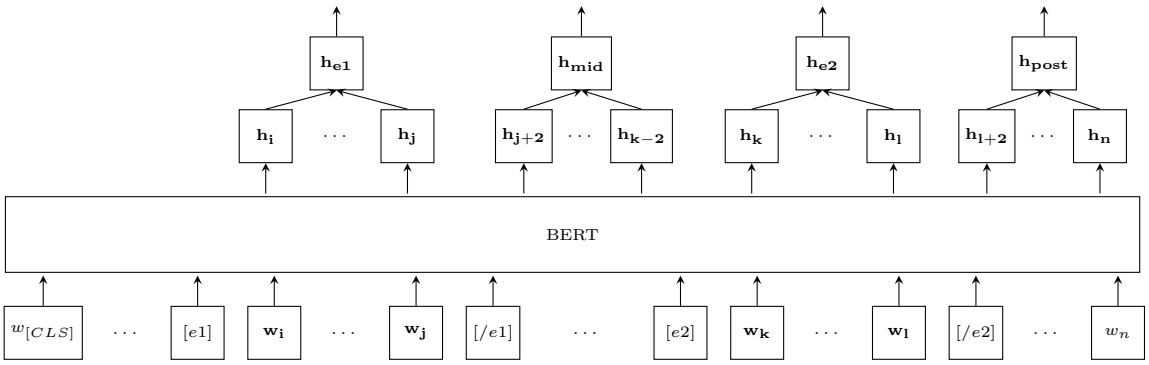


Figure 21: Output type module `Es+mid+post`

periment results of applying `Es+mid` and `Es+post` where both provide improved precision but lower recall on TACRED and Biocreative VII Track 1. However, the improvement of precision on `Es+mid+post` is lower than `Es+mid` and `Es+post` on SemEval 2010, whereas `Es+mid+post` provides dramatically enhanced precision on TACRED.

[CLS]+Es+mid & [CLS]+Es+post & [CLS]+Es+mid+post These output type modules are the form of adding CLS vector on top of the output type modules that leveraged the chunk. The input of the classifier of these output type modules is a linearly concatenated vector of entities and the selection of mid and/or post depends on the selected output type module. The example output representation of output type `[CLS]+Es+mid+post` is depicted in Figure 22. In Table 16, `[CLS]+Es+mid` yields better precision compared to `[CLS]` module in all tasks while recall is lower in most cases except on Re-TACRED task. The overall performance of precision, recall and f1-score of `[CLS]+Es` and `[CLS]+Es+mid` are similar in all tasks. Although `[CLS]+Es+post` provides the best recall compared to `[CLS]`, `[CLS]+Es`,

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**OT**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
EM	[CLS]	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
EM	Es	84.6	89.0	86.4	69.2	64.1	**66.6**	88.0	87.7	**87.8**	72.8	75.2	**74.0**
EM	Es+mid	87.3	89.5	88.3	72.8	60.8	66.2	87.8	86.8	87.3	76.0	69.9	72.8
EM	Es+post	88.4	88.7	**88.5**	69.8	60.3	64.7	87.9	87.3	87.6	77.0	69.5	73.1
EM	Es+mid+post	86.9	88.5	87.6	75.0	59.3	66.2	88.2	87.4	**87.8**	75.2	70.9	73.0

Table 15: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to `Input type` and `Output type`. `EM` refers to `Entity Markers`.

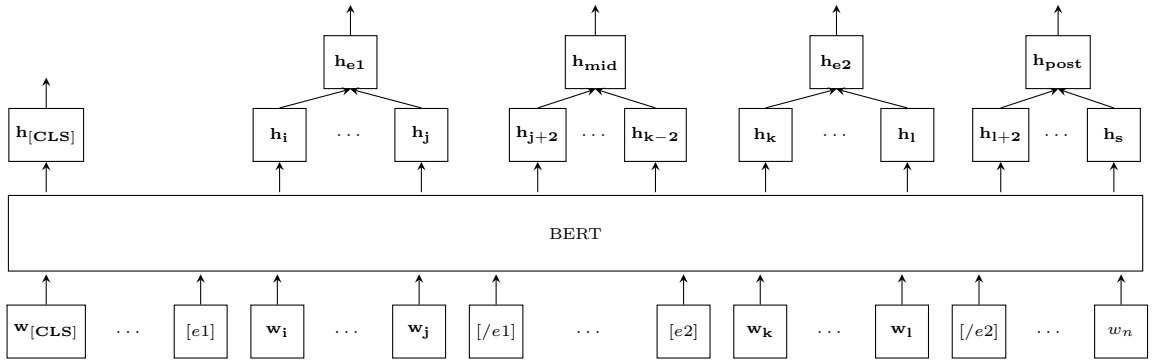


Figure 22: Output type module `[CLS]+Es+mid+post`

[CLS]+mid, and [CLS]+mid+post in SemEval 2010 Task 8, precision drops to the lowest performance of all [CLS] based modules. Moreover, this pattern was only observed in SemEval 2010 Task 8 while TACRED result shows the opposite where it achieves the best precision and the poor recall. In comparison with quite fluctuated result of [CLS]+Es+post, [CLS]+Es+mid+post achieved an improved f1-score in all cases except Biocreative VII Track 1.

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**OT**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
EM	[CLS]	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
EM	[CLS]+Es	91.0	85.0	87.8	72.9	61.0	66.4	87.9	86.6	87.2	77.0	71.2	**74.0**
EM	[CLS]+Es+mid	91.0	86.5	88.7	72.3	60.8	66.0	88.7	86.7	87.7	76.4	70.6	73.4
EM	[CLS]+Es+post	86.3	90.2	88.1	74.6	58.6	65.6	88.1	87.6	87.9	75.8	70.5	73.1
EM	[CLS]+Es+mid+post	90.2	87.9	**89.0**	72.0	62.7	**67.0**	88.4	87.6	**88.0**	76.6	69.3	72.8

Table 16: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to Input type and Output type. EM refers to Entity Markers.

3.1.3 Experiments with attached additional entity tokens

This section will describe the studies on additional input format modules. The *input format* represents the input to BERT, adding redundant entity tokens next to the tokenized textual input. The most extensively used input format is the form where [CLS] token is placed prior to the tokenized textual input and [SEP] is attached after the tokenized textual input. Instead of this standard input format, however, different input formats have the potential to improve its performance ([Alt et al., 2019]; [Shi and Lin, 2019]; [Tao et al., 2019]). [Alt et al., 2019] and [Shi and Lin, 2019] introduced that affixing entity tokens next to the full context can enhance its performance on relation extraction tasks. Also, [Tao et al., 2019] introduced that attaching surrounding tokens is the use of injecting task-related information to BERT. Therefore, in order to investigate how different input format modules and the position of attached tokens influences the system, a set of ablation studies were conducted. As depicted in Example (13), a given data sample represented as $T = [t_0, ..., t_s]$ where s is the number of tokens of the data sample. Entity Markers is the selected input type module and [CLS] is the selected output type module in this experiment.

(13) ***Form:*** *T*
$\Longrightarrow$ *The <e1> water </e1> was in a <e2> cup </e2>*

Standard This is the basic input format module of BERT where a special token [CLS] is prior to the input text and SEP token is placed at the end of the input text. Therefore, `Standard` module is selected as a baseline input format module for comparison. Example (14) represents the form and example of `Standard` module.

(14) ***Form:*** *[CLS]* $\oplus$ *T* $\oplus$ *[SEP]*
$\Longrightarrow$ *[CLS] the <e1> water </e1> was in a <e2> cup </e2> [SEP]*

Es+Standard This is the variant of `Standard` module where the module has additional representation of e_1 and e_2 are embedded right after the [CLS] token instead of given text. The representation of the attached entities and the representation of given text are separated by [SEP]. The purpose of the `Es+Standard` module is providing additional representation of entities before the full text so the module aims to give information to the system of where to focus. Example (15) represents the form of `Es+Standard` module and its example where T_{e_1} and T_{e_2} represent tokens of the first and second entity.

(15) ***Form:*** *[CLS]* $\oplus$ T_{e_1} $\oplus$ T_{e_2} $\oplus$ *[SEP]* $\oplus$ *T* $\oplus$ *[SEP]*
$\Longrightarrow$ *[CLS] water cup [SEP] the <e1> water </e1> was in a <e2> cup </e2> [SEP]*

Standard+Es This input format module is a flipped version of `Es+Standard` module where the given text is placed at the front right after the [CLS] token, and two additional entities are located after the SEP token which divides between the given text and entities. This input format module aims to discover the importance of the position of the given text when additional task-specific information is added by comparing with `Es+Standard` module. Example (16) represents the form of `Standard+Es` module and its example input format.

(16) ***Form:*** *[CLS]* $\oplus$ *T* $\oplus$ *[SEP]* $\oplus$ T_{e_1} $\oplus$ T_{e_2} $\oplus$ *[SEP]*
$\Longrightarrow$ *[CLS] the <e1> water </e1> was in a <e2> cup </e2> [SEP] water cup [SEP]*

As presented in Table 17, both `Es+Standard` and `Standard+Es` output improves precision compared to `Standard` module in general. In particular, precision climbed on both SemEval 2010 Task 8 and Biocreative VII Track 1. Both `Es+Standard` and `Standard+Es` shows improved f1-score on SemEval 2010 Task 8 while maintaining its recall. Due to the fact that recall dropped to a certain extent on Biocreative VII Track 1, however, the overall f1-score has no dramatic change.

Es(SEP)+Standard In general, attaching SEP token in order to separate the additional entity tokens that are attached next to the context is an extensively applied manner

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**IF**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
EM	Standard	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
EM	Es+Standard	89.3	87.3	**88.3**	65.4	67.2	**66.3**	88.8	83.6	86.1	75.7	69.8	72.7
EM	Standard+Es	89.1	86.1	87.6	66.6	65.1	65.8	87.2	86.1	**86.6**	74.8	71.3	**73.0**

Table 17: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets and Biocreative VII Track 1 on dev sets. **IT** and **IF** refers to `Input type` and `Input format`. `EM` refers to `Entity Markers`.

([Alt et al., 2019]; [Shi and Lin, 2019]) and it seems an intuitive manner due to the fact that entities have to be considered distinguished components. The role of SEP token in between entities that are affixed in front of the context is discarding potential ambiguity that may occur in `Es+Standard` due to no segregation between entities. Each entity token is separated by a SEP token and the full context is also divided by an additional SEP token placed prior to the full context. However, the difference between [Alt et al., 2019] and `Es(SEP)+Standard` is that [Alt et al., 2019] utilized additional delimiters such as [sep1], [sep2] and [start] instead of leveraging [SEP] and [CLS] tokens that included in BERT vocabulary. Example (17) represents the form of `Es(SEP)+Standard` module and its example input format.

(17) ***Form:*** *[CLS]* $\oplus$ T_{e_1} $\oplus$ *[SEP]* $\oplus$ T_{e_2} $\oplus$ *[SEP]* $\oplus$ T $\oplus$ *[SEP]*
$\Longrightarrow$ *[CLS] water [SEP] cup [SEP] the <e1> water </e1> was in a <e2> cup </e2> [SEP]*

Standard+Es(SEP) This input format is flipped version of `Es(SEP)+Standard` module where the given text is placed at the front right after the [CLS] token, and two separated entities are located after the SEP token which divides between the given text and two separated entities. [Shi and Lin, 2019] introduced attaching entity tokens after the full context and segregating not only the full context and the first entity but also the first entity and the second entity that are attached after the full context. However, [Shi and Lin, 2019] inserted Bidirectional-LSTM on top of BERT, and only utilized the context part without including the additional token part instead of applying CLS vector of the last hidden state of BERT. The purpose of applying `Es(SEP)+Standard` module is to investigate the significance of the SEP token that segregates each token attached after the context, and to investigate the importance of the position of attaching entities compared to `Es(SEP)+Standard` module. Example (18) represents the form of `Standard+Es(SEP)` module and its example input format.

(18) ***Form:*** *[CLS]* $\oplus$ *T* $\oplus$ *[SEP]* $\oplus$ T_{e_1} $\oplus$ *[SEP]* $\oplus$ T_{e_2} $\oplus$ *[SEP]*
$\Longrightarrow$ *[CLS] the <e1> water </e1> was in a <e2> cup </e2> [SEP] water [SEP] cup [SEP]*

The experiment results of `Es(SEP)+Standard` and `Standard+Es(SEP)` module are presented in Table 18. Both `Es(SEP)+Standard` and `Standard+Es(SEP)` module yield improved precision compared to `Standard` module. In the same manner that it is observed in the comparison of `Es+Standard` and `Standard+Es` module, the performance between `Es(SEP)+Standard` and `Standard+Es(SEP)` module is also similar to one another in most cases.

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**IF**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
`EM`	`Standard`	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
`EM`	`Es(SEP)+Standard`	89.4	86.5	87.9	68.5	64.3	66.3	87.6	85.7	86.6	74.9	73.2	**74.1**
`EM`	`Standard+Es(SEP)`	89.0	87.5	**88.2**	68.9	66.6	**67.7**	87.7	86.8	**87.2**	74.0	70.4	72.2

Table 18: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets and Biocreative VII Track 1 on dev sets. **IT** and **IF** refers to `Input type` and `Input format`. `EM` refers to `Entity Markers`.

3.1.4 Complexity of various input & output representations

The complexity of the system represents the size of the system. The complexity affects training time and the availability of training. Therefore, if the system is system is highly complex, it requires more time to be trained, and this system might not available to be trained due to the limitation of resource. Table 19 presents the complexity of each systems. The overall complexity difference is similar to one another.

3.2 Architectures based on a linear and a bilinear classifier

This section is focused on various architectures leveraging a linear and bilinear classifier. In order to concentrate on investigating the application of a linear and bilinear classifier, the same input type and input format are applied. `Entity Markers` is applied where the span of entities are explicitly identified in the textual input with entity markers. The input format is `Standard`. The selected output vectors in this experiment are CLS vector, entity vectors, or both. Based on the supposition of a sequence of tokens introduced before, $< w_{[CLS]}, w_1, ..., w_n >$ represents represents input of BERT including special tokens ([CLS],

Output Type	Input Format	Number of parameters (M)
`[CLS]`	`Standard`, `Es+Standard`, `Standard+Es`, `Es(SEP)+Standard`, `Standard+Es(SEP)`	109.5
`Es`	-	110.1
`[CLS]+Es`, `Es+mid`, `Es+post`	-	110.7
`Es+mid+post` `[CLS]+Es+mid` `[CLS]+Es+post`	-	111.3
`[CLS]+Es+mid+post`	-	111.9

Table 19: Complexity of various input & output representations.

[SEP], and entity markers) of a given data sample. $< h_{[CLS]}, h_1 ..., h_n >$ represents the last hidden state vectors of BERT.

3.2.1 M_{Base} architecture

This is the baseline architecture for comparing the performance of architectures that are based on a linear or a bilinear classifier. M_{Base} architecture selects CLS vector $h_{[CLS]}$ as an input of a linear classifier. Figure 23 shows the architecture of M_{Base}. Equation 13 presents the selection of output vector of BERT and a classifier where weight $W_{li} \in \mathbb{R}^{l \times d}$ (l is the number of relation labels), and bias $b_{li} \in \mathbb{R}^{l}$.

$$p = Softmax[W_{li} h_{[CLS]} + b_{li}] \tag{13}$$

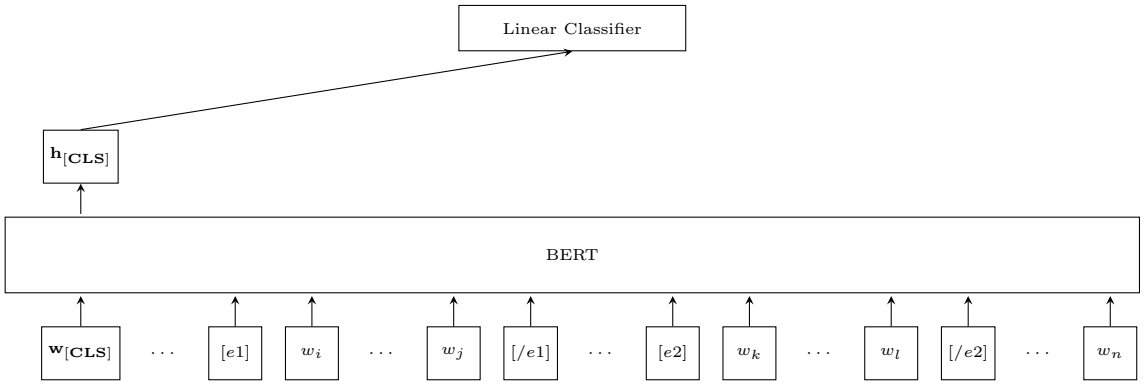


Figure 23: M_{Base} architecture

3.2.2 M_E architecture

M_E architecture utilizes the concatenation of entity vectors instead of CLS vector as an input of a classifier. This architecture aims to compare the effectiveness of a linear classifier with a bilinear classifier when entity vectors are the input of a classifier. Figure 24 shows the architecture of M_E. Equation 14 presents the selection of output vector of BERT and a classifier where CAT represents concatenation.

$$\begin{aligned} \hat{h}_{e1} &= W_1[ReLU(h_{e1})] + b_1 \\ \hat{h}_{e2} &= W_1[ReLU(h_{e2})] + b_1 \\ h_{CAT} &= CAT(\hat{h}_{e1}, \hat{h}_{e2}) \\ p &= Softmax[W_{li}(h_{CAT}) + b_{li}] \end{aligned} \tag{14}$$

W_1 is the weight of the fully connected layer where $W_1 \in \mathbb{R}^{d \times d}$ (d is the size of embedding dimension). $\hat{h}_1$ and $\hat{h}_2$ are the results of a fully connected layer and non-linear activation function of h_1 and h_2. W_{li} is a linear classifier weight where $W_{li} \in \mathbb{R}^{l \times 2d}$ (l is the number of relation labels).

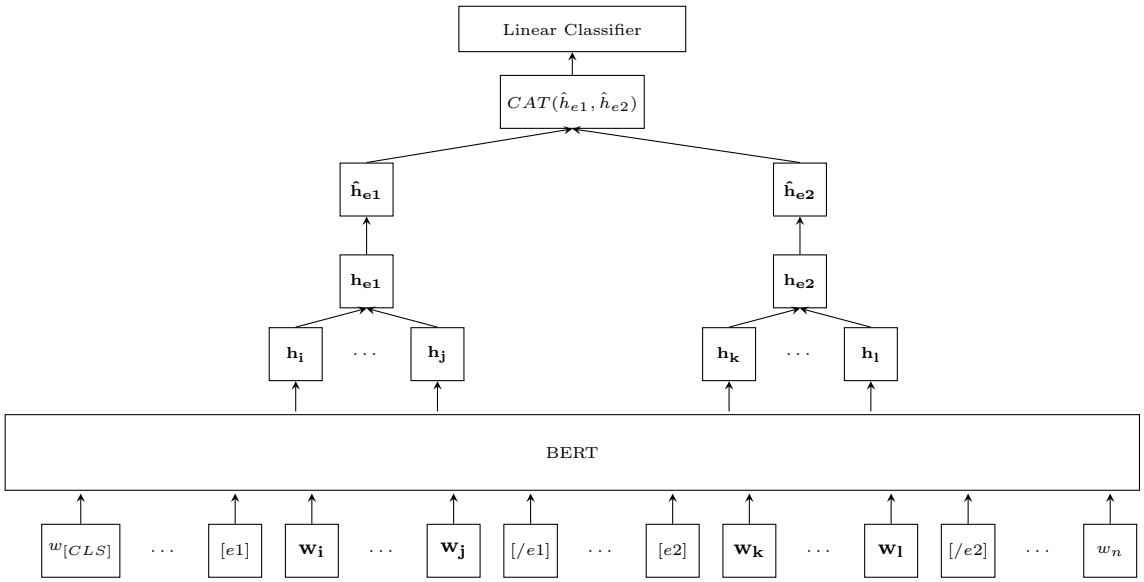


Figure 24: M_E architecture

In Table 20, M_E yields slightly increased precision on most of the tasks compared to M_{Base} except on SemEval 2010 Task 8. Moreover, recall is also increased in most cases. In most cases, M_E outperforms M_{Base} on f1-score but the difference is marginal. Therefore, it suggests a single CLS vector is able to be utilized to obtain a fair amount of performance but it is more reliable to utilize entity vectors to achieve slight improvement.

Architecture	SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
	P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
M_{Base}	86.2	87.4	**86.6**	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
M_E	84.6	89.0	86.4	69.2	64.1	**66.6**	88.0	87.7	**87.8**	72.8	75.2	**74.0**

Table 20: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets.

3.2.3 M_R architecture

M_R architecture is introduced in [Wu and He, 2019] as R-BERT that aims to achieve enriching the entity information by leveraging CLS and entity vectors. Since the classifier adopted in this architecture is a linear classifier, a linearly concatenated vector of CLS and entity vectors is leveraged. Figure 25 shows the architecture of M_R. Equation 15 mathematically presents the usage of [CLS] and entity vectors in this architecture.

$$\begin{aligned} \hat{h}_{[CLS]} &= W_0[Tanh(h_{[CLS]})] + b_0 \\ \hat{h_{e1}} &= W_1[ReLU(h_{e1})] + b_1 \\ \hat{h_{e2}} &= W_1[ReLU(h_{e2})] + b_1 \\ h_{CAT} &= CAT(\hat{h}_0, \hat{h}_{e1}, \hat{h}_{e2}) \\ p &= Softmax[W_{li} h_{CAT} + b_{li}] \end{aligned} \tag{15}$$

W_0 and W_1 are the weight of fully connected layer where $W_0 \in \mathbb{R}^{d \times d}$ and $W_1 \in \mathbb{R}^{d \times d}$ (d is the size of embedding dimension). $\hat{h}_{e1}$ and $\hat{h}_{e2}$ are the forwarded vector of h_{e1} and h_{e2}. W_{li} is a linear classifier weight where $W_{li} \in \mathbb{R}^{l \times 3d}$ (l is the number of relation labels).

The experiment result of M_R is presented with M_{Base} and M_E in Table 21. M_R shows enhanced precision compared to both M_{Base} and M_E in most cases. Especially, precision is considerably increased when CLS and entity vectors are applied at the classification step on Biocreative VII Track 1. In contrast to the tendency that most of the task results yield

Architecture	SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
	P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
M_{Base}	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
M_E	84.6	89.0	86.4	69.2	64.1	**66.6**	88.0	87.7	**87.8**	72.8	75.2	**74.0**
M_R	91.0	85.0	**87.8**	72.9	61.0	66.4	87.9	86.6	87.2	77.0	71.2	**74.0**

Table 21: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets.

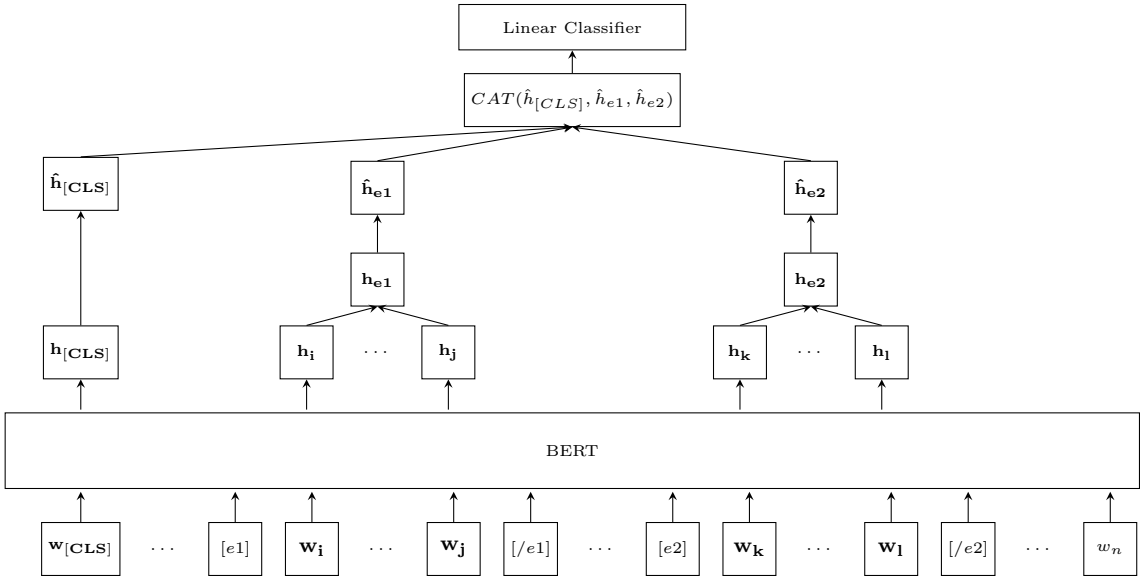


Figure 25: M_R architecture

improved precision while recall is dropped, both precision and recall are rose in SemEval 2010 Task 8. The results illustrate that utilizing the CLS and entity vectors that form a relation has a strong potential of improving precision compared to adopting one of each.

3.2.4 M_{Bie} architecture

This architecture adopted a bilinear classifier that transforms two vectors into one vector. Since a bilinear classifier requires two input vectors unlike a linear classifier, entity vectors are selected as inputs. Figure 26 shows the architecture of M_{Bie}. M_{Bie} architecture is presented mathematically in Equation 16.

$$\begin{aligned} \hat{h}_{e1} &= W_1[ReLU(h_{e1})] + b_1 \\ \hat{h}_{e2} &= W_1[ReLU(h_{e2})] + b_1 \\ p &= Softmax[\hat{h}_{e1} W_{bi} \hat{h}_{e2} + b_{bi}] \end{aligned} \tag{16}$$

W_1 is the weight of fully connected layer where $W_1 \in \mathbb{R}^{d \times d}$ (d is the size of embedding dimension). $\hat{h_{e1}}$ and $\hat{h_{e2}}$ are the results of a fully connected layer and non-linear activation function of h_1 and h_2. W_{bi} is the bilinear classifier weight where $W_{bi} \in \mathbb{R}^{l \times d \times d}$ (l is the number of relation labels).

The experiment result of M_{Bie} presented in Table 22 with the result of M_{Base} and M_E for comparison. M_{Bie} outputs dropped recall compared to M_{Base} and M_E but precision is increased in most cases. Interestingly, even though both M_{Bie} and M_E utilized the same input vectors as an input of a classifier, M_{Bie} presents superior precision compared to M_E.

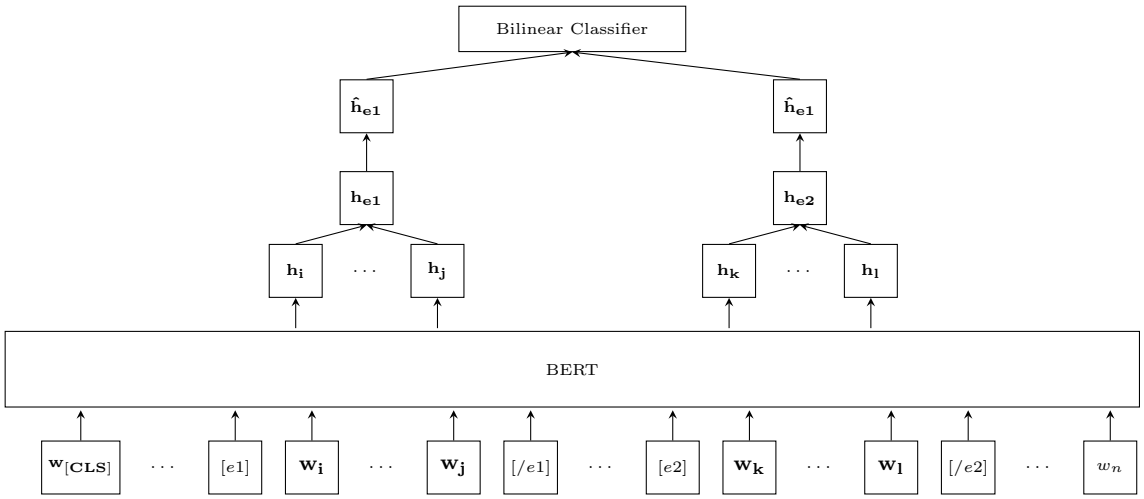


Figure 26: M_{Bie} architecture

Architecture	**SemEval 2010**			**TACRED**			**Re-TACRED**			**Biocreative VII**		
	P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
M_{Base}	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
M_E	84.6	89.0	86.4	69.2	64.1	**66.6**	88.0	87.7	**87.8**	72.8	75.2	**74.0**
M_{Bie}	90.4	87.2	**88.7**	72.9	60.5	66.1	88.3	86.6	87.4	74.3	73.1	73.7

Table 22: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets.

3.2.5 M_{Bicls} architecture

M_{Bicls} architecture applies CLS and entity vectors similar to M_R architecture. The first input vector of M_{Bicls} is a linearly concatenated of first entity and CLS vector, and the second input vector is a linearly concatenated second entity and CLS vector. Figure 26 shows the architecture of M_{Bicls}. M_{Bicls} architecture is presented mathematically in Equation 17.

$$\begin{aligned} \hat{h}_{[CLS]} &= W_0[Tanh(h_{[CLS]})] + b_0 \\ \hat{h}_{e1} &= W_1[ReLU(h_{e1})] + b_1 \\ \hat{h}_{e2} &= W_1[ReLU(h_{e2})] + b_1 \\ h_{cat1} &= CAT(\hat{h}_{[CLS]}, \hat{h}_{e1}) \\ h_{cat2} &= CAT(\hat{h}_{[CLS]}, \hat{h}_{e2}) \\ p &= Softmax[h_{cat1} W_{bi} h_{cat2} + b_{bi}] \end{aligned} \tag{17}$$

W_0 and W_1 are the weight of two different fully connected layers respectively where $W_0 \in \mathbb{R}^{d \times d}$ and $W_1 \in \mathbb{R}^{d \times d}$ (d is the size of embedding dimension). $\hat{h_{e1}}$ and $\hat{h_{e2}}$ are forwarded vector of h_{e1} and h_{e2}. The W_{bi} is the bilinear classifier weight where $W_{bi} \in \mathbb{R}^{l \times d \times d}$ (l is the number of relation labels).

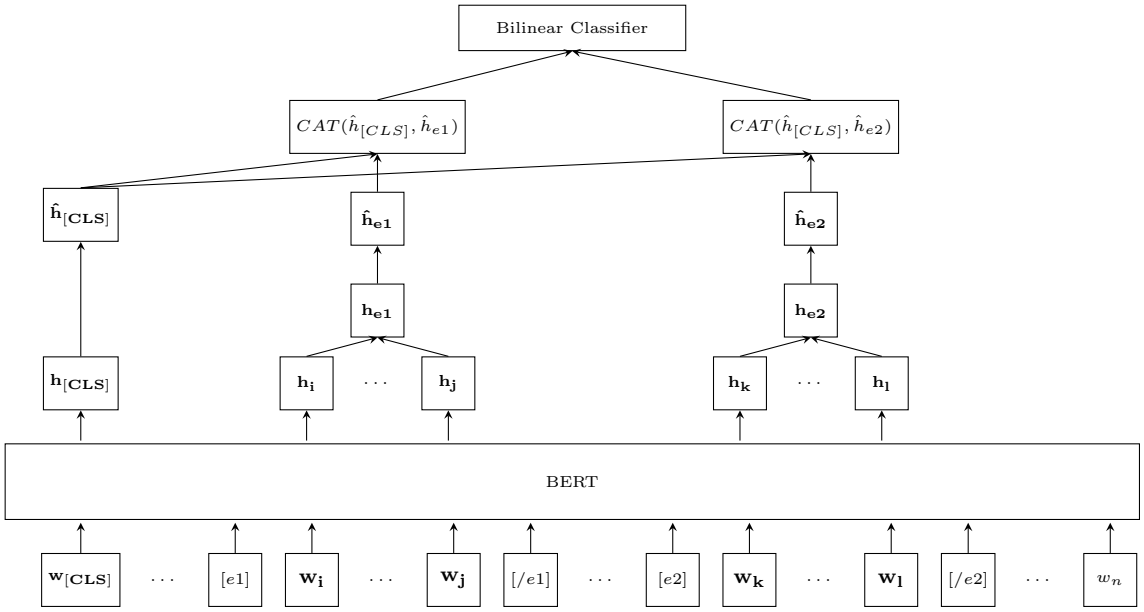


Figure 27: M_{Bicls} architecture

The result of M_{Bicls} architecture presented in Table 23. M_{Bicls} outperforms M_{Base} on precision in all tasks. Even though recall has somewhat fluctuated, the improvement of precision surpasses the decrement of recall so it leads to the increased f1-score. In terms of

Architecture	SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
	P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
M_{Base}	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
M_R	87.0	89.8	88.3	72.9	61.0	66.4	87.9	86.6	87.2	77.0	71.2	74.0
M_{Bie}	90.4	87.2	**88.7**	72.9	60.5	66.1	88.3	86.6	**87.4**	74.3	73.1	73.7
M_{Bicls}	90.4	87.0	88.6	70.5	65.2	**67.8**	88.3	85.5	86.9	75.8	72.6	**74.2**

Table 23: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets.

the selection of vectors to a classifier, both M_{Bicls} and M_R leveraged CLS and entity vectors as input. However, the method of supplying vectors to the classifier and the selection of the classifier affect the experimental results. For instance, M_R shows somewhat improved performance on both precision and recall on SemEval 2010 Task 8 and Re-TACRED compared to M_{Base}. However, M_{Bicls} precision outperforms both M_{Base} and M_R while slightly degraded recall. With regards to the choice of classifier, both M_{Bicls} and M_{Bie} adopted a bilinear classifier. Even though the selection of input vectors is different in the manner of adding CLS vector or not, M_{Bicls} and M_{Bie} show similar performance output in most tasks. In particular, both architectures gained improvement on precision and recall on SemEval 2010 and Re-TACRED and the actual performance score is similar.

3.2.6 Complexity of various architectures

Table 24 presents the complexity of various architecture introduced in this thesis. M_{Base}, M_E, and M_R show somewhat similar complexity to one another, whereas M_{Bie}, M_{Bicls} show big difference. This is because of a choice of a classifier. The complexity of a linear

Architecture	Number of parameters (M)
M_{Base}	109.5
M_E	110.1
M_R	110.7
M_{Bie}	121.9
M_{Bicls}	155.5

Table 24: Complexity of various architectures.

classifier is linearly increased based on the size of input vector. The number parameters of a linear classifier is $d_{input} * d_{output} + d_{output}$ where d_{input} and d_{output} is the size of input and output vector of a classifier respectively. Since d_{output} is the size of labels that can

be ignored, the complexity of a linear classifier is determined by d_{input}. Unlike a linear classifier, complexity of a bilinear classifier is $d_{input_i} * d_{input_j} * d_{output} + d_{output}$ where d_{input_i} and d_{input_j} is the input size of two input vectors and d_{output} is the size of output vector. Since d_{output} is the size of labels that can be ignored, complexity of a bilinear classifier is determined by the square of input vector size, whereas a linear classifier is determined by the input vector size.

Chapter 4

Discussion

4.1 Study on Input type modules

This section analyzes two input type modules by providing experimental results on relation extraction tasks and compares them on three different output type modules. To investigate input type modules in detail, a thorough analysis based on the predicted data samples is presented.

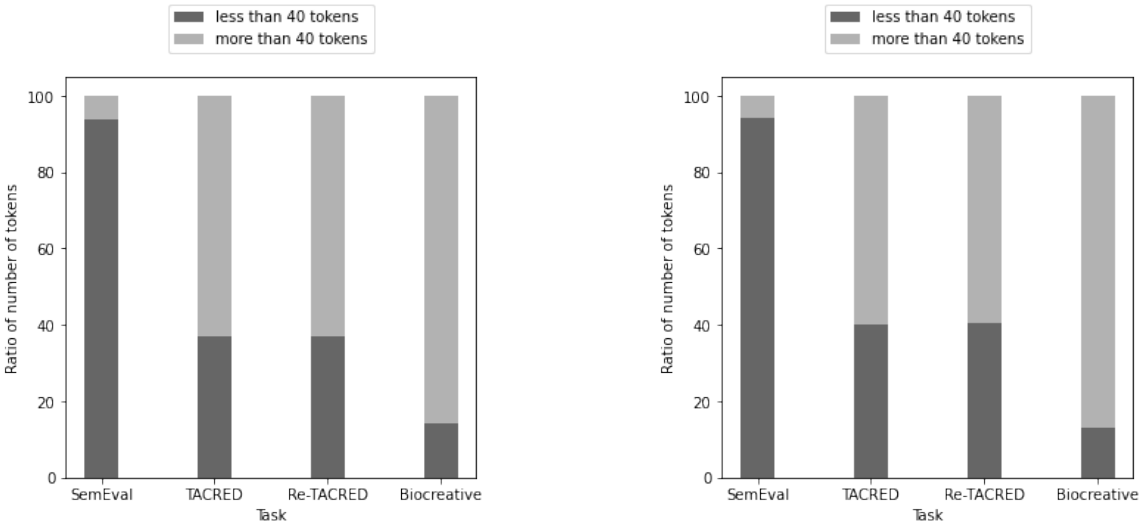
4.1.1 Comparison on the baseline system

The experiment results of the baseline system on two input type modules are presented in Table 25. A baseline system achieved greater improvement on `Entity Markers` module

Module		**SemEval 2010**			**TACRED**			**Re-TACRED**			**Biocreative VII**		
IT	**OT**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
`NM`	`[CLS]`	78.0	78.1	78.0	28.0	13.1	17.8	39.1	37.8	38.4	53.3	28.2	36.9
`EM`	`[CLS]`	86.2	87.4	**86.6**	66.2	64.1	**65.1**	87.2	85.7	**86.4**	71.1	74.5	**72.7**

Table 25: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to `Input type` and `Output type`. `NM` and `EM` represents `No Markers` and `Entity Markers` respectively.

compared to `No Markers` module in all tasks. While `No Markers` module yields poor performance on most tasks, this system output a decent performance on SemEval 2010 Task 8. To investigate this particular result, I present detailed descriptions of this dataset by comparing each other. The major difference between SemEval 2010 Task 8 and other tasks is that most of the data samples in SemEval 2010 Task 8 are comparatively shorter than other tasks as presented in Figure 28. Also, SemEval 2010 Task 8 has a higher ratio of



(a) Train dataset distribution

(b) Test & Dev dataset distribution

Figure 28: Distribution of number of tokens in SemEval 2010 Task 8, TACRED, Re-TACRED and Biocreative VII Track 1 dataset

positive samples than negative samples compared to other tasks as presented in Table 26. In addition, this dataset has only nine relation labels to be classified which is less than other tasks. For this reason, SemEval 2010 Task 8 is a relatively less challenging task compared to other tasks and this allows even `No Markers` module can output a decent performance.

Data sample	**SemEval 2010**	**TACRED**	**Re-TACRED**	**Biocreative VII**
Positive case	2263 (83.3%)	3325 (21.4%)	5648 (42.1%)	3707 (27.5%)
Negative case	454 (16.7%)	12184 (78.6%)	7770 (57.9%)	9771 (72.5%)
Total	2717 (100%)	15509 (100%)	13418 (100%)	13478 (100%)

Table 26: Distributions of positive and negative data samples of SemEval 2010 Task 8, TACRED and Re-TACRED test dataset and Biocreative VII Track 1 dev dataset.

The presented data samples in Table 27 are misclassified examples on `No Markers` module while these are correctly predicted on `Entity Markers` module. The most frequent false-negative cases of `No Markers` module on SemEval 2010 Task 8 are *Component-Whole* and *Instrument-Agency* relations. The data samples of these two relations tend to be comparatively more dependent on entities than on surrounding words. Thus, certain surrounding words that are fairly associated with relation less tend to be observed in *Component-Whole* and *Instrument-Agency* relations. For instance, the word *cause* is the most frequent surrounding word that can be easily detected in *Cause-Effect* relation. Also, *into* is the most

common surrounding word in *Entity-Destination* relation.

Dataset	Gold	Data sample
SemEval 2010	Component -Whole	*The disgusting scene was retaliation against her brother Philip who rents the room inside this apartment house on Lombard street.*
		But for an old AT keyboard the keypad keys produced digits when Numlock was on or Shift.
	Instrument -Agency	*Experienced bakers hold the door open a crack with a wooden spoon handle.*
		For straight cuts, a tile cutter is the best tool for a do-it-yourselfer.
TACRED	per:title	*'Big Bang' physicist Andrew Lange dead at 52.*
		ALERT US missionary Laura Silsby freed in Haiti : lawyer
	org:country _of_headquarters	*Founded in 1992 in Schaumburg, Illinois, the ACSE is one of the largest Chinese-American associations of. professionals in the United States*
		AIG said it had transferred ownership to the Federal Reserve Bank of parts of two subsidiaries, ALICO which is active in life assurance in the United States and AIA which provides life assurance abroad.

Table 27: False-negative examples of test dataset from SemEval 2010 Task 8 and TACRED. The underlined words represent the span of entities. *Gold* represents the gold standard.

With respect to TACRED, most of the false-negative samples of `No Markers` module can be observed from *per:title*, *org:alternate_names* and *org:country_of_headquarters* relations. These relations have a larger number of data samples than other relations, and most of these data samples are difficult to classify without the span of entities. As presented in Table 27, these examples can be classified into different relations when the span of entities is not indicated. For instance, *'Big Bang' physicist Andrew Lange dead at 52.*, which is the first example of *per:title* in Table 27, can be classified as *per:date_of_death* relation if *Andrew Lange* and *52* are considered entities. While `No Markers` module struggles to classify the relation on those relations, it well predicts data samples in some relations where certain words are the strong evidence of the relation of entities. These certain words can be observed in *org:top_members/employees* relation and *per:age* relation. In *org:top_members/employees*

relation, *President, CEO, deputy, director* are the most frequent surrounding words. Moreover, two-digit number and *year-old* is repeatedly detected on *per:age* relation.

The false-positive cases of `No Markers` module is presented in Table 28 while these examples are correctly predicted by `Entity Markers` module. In SemEval 2010 Task 8,

Dataset	Prediction(Gold)	Data sample
SemEval 2010	Cause-Effect (Entity-Origin)	*Most oil polluting the oceans comes from runoff, rivers, small boats, not tanker spills.*
		The sound came from a wooden structure two hundred yards away.
		The breaking news from the article was that the first recorded use of the word gongoozler took place in the late 1800s in the book Narrow Boat by T.C. Rolt.
TACRED	per:employee_of (per:title)	*In 2006, he became a senior research scientist at NASA's Jet Propulsion Laboratory and was appointed chairman of Caltech's physics, mathematics and astronomy division in 2008.*
		Ivory Coast's new UN ambassador, Youssoufou Bamba, said he is worried about his country's future and is consulting with members of the Security Council ahead of a meeting next week on ways to help Ouattara assume power.
	per:employee_of (org:top_members /employees)	*DENVER "She was a loving mentor to me," said Vernon Jordan, civil rights leader and former adviser to President Bill Clinton, who succeeded Whitney Young as head of the National Urban League.*
		– Rev. Gary Simons , minister at High Point Church in Arlington, justifying the church's decision.

Table 28: False-positive examples of test dataset from SemEval 2010 Task 8 and TACRED. The underlined words represent the span of entities. *Gold* represents the gold standard.

most of false-positive cases of `No Markers` are negative labels. The most often false-positive cases from positive labels are *Cause-Effect*, and `No Markers` system often predicts *Entity-Origin* relation as *Cause-Effect*. Since the span of entities is not provided, the system performance highly depends on the given data samples. As presented in Table 28, the majority of false-positive examples in *Cause-Effect* relation contain the word *from* in their textual input. The word *from* is also one of the most frequent words in both *Cause-Effect* and *Entity-Origin* relations, whereas it is a relatively infrequent word in other relations.

In TACRED, most false-positive cases of `No Markers` system are also negative labels. The most frequent false-positive relation from positive labels is *per:employee_of* relation. `No Markers` system often classifies *per:employee_of* relation to *org:top_members/employees* or *per:title* relation. The commonly observed characteristics of *per:employee_of*, *per:title* and

org:top_members/employees relation is that these relations have similar information such as the name of the organization, name of the person, and title as presented in Table 28. For instance, the first example of *per:title* can be classified as *per:employee_of* if `No Markers` system focuses on *he* and *NASA* not *senior research scientist*. Moreover, *per:employee_of*, *per:title* and *org:top_members/employees* relations not only include similar information but also contain the same words as presented in Table 29. Table 29 presents examples that are correctly predicted in `Entity Markers` module but not in `No Markers` module. These examples are often detected in both TACRED and Re-TACRED.

Gold	Data sample
per:employee_of	*Against this background, Ouattara's new United Nations ambassador Youssoufou Bamba meanwhile gave a stark warning as he received his credentials from UN Secretary-General Ban Ki-moon.*
per:title	*Ouattara's new United Nations ambassador Youssoufou Bamba meanwhile gave a stark warning as he received his credentials from UN Secretary-General Ban Ki-moon.*

Table 29: Examples that are similar in the usage of words in the context from TACRED. The underlined words represent the span of entities. *Gold* represents the gold standard.

The analysis of a baseline system on two input type modules illustrates `No Markers` module performance is highly affected by certain trigger words. Furthermore, if the dataset has similar data samples like TACRED and Re-TACRED, `No Markers` module tend to output poor performance.

4.1.2 Integrated comparison of Input type modules

To investigate whether a poor performance of a baseline system on `No Markers` is because of the choice of output vector or presence of entity markers, a set of ablation studies on different output vectors is studied. The experiments based on the additional output type module `Es` and `[CLS]+Es` are performed in order to observe `No Markers` module from different cases. `No Markers` module shows improved performance on `Es` and `[CLS]+Es` output type module as presented in Table 30. However, a baseline system on `Entity Markers` module still outperforms `No Markers` module systems regardless of the choice of output vector. Most of the loss of `No Markers` module is because of the incorrect prediction of positive samples to negative labels. However, these cases are dramatically reduced on `Entity Markers` module so a system outputs enhanced recall on TACRED, Re-TACRED, and Biocreative VII Track 1 where negative samples outnumber positive samples.

In conclusion, `Entity Markers` outperforms `No Markers` module regardless of the choice

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**OT**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
NM	[CLS]	78.0	78.1	78.0	28.0	13.1	17.8	39.1	37.8	38.4	53.3	28.2	36.9
NM	Es	84.9	84.7	84.7	64.9	45.8	53.7	81.5	74.5	77.8	72.5	66.6	69.5
NM	[CLS]+Es	86.2	83.8	84.9	67.3	50.7	57.8	82.1	71.4	76.4	72.1	67.5	69.7
EM	[CLS]	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
EM	Es	84.6	89.0	86.4	69.2	64.1	**66.6**	88.0	87.7	**87.8**	72.8	75.2	**74.0**
EM	[CLS]+Es	91.0	85.0	**87.8**	72.9	61.0	66.4	87.9	86.6	87.2	77.0	71.2	**74.0**

Table 30: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to `Input type` and `Output type`. `NM` and `EM` represents `No Markers` and `Entity Markers` respectively.

of output vector as depicted in Table 30. The importance of indicating the span of entities can be well observed when CLS vector is the only selected output vector. Also, the performance gap between `No Markers` and `Entity Markers` increases depending on the characteristics of the dataset. Therefore, entity markers that indicate the span of entities influences systems in general.

4.2 Study on Output type modules

This section investigates a variety of output type modules on two input type modules by comparing experiment results and examining the predicted data samples.

4.2.1 Comparison on `No Markers` input type module

The experiment results on three different output type modules on `No Markers` module are presented in Table 31. Since the span of entities are not specified, `Es` and `[CLS]+Es` module that use entity vectors outperforms `[CLS]` module in all cases as expected. The improvement of `Es` and `[CLS]+Es` compared to `[CLS]` is significant on TACRED, Re-TACRED and Biocreative VII Track 1. This can be explained by the characteristics of the dataset. These datasets include data samples that have the same text input but differ in relation to the span of entities. Table 32 presents examples that are well classified on `Es` and `[CLS]+Es` module but misclassified on `[CLS]` module. As presented in Table 32, the relation of this textual input is indistinguishable without the information of the span of entities. These examples are not appeared in SemEval 2010 Task 8, whereas these can be observed in other tasks. The experimental results indicate if the span of entities is not explicitly specified,

entity vectors are the most decisive information. Moreover, the significance of using entity vectors is greater when the dataset includes data samples that are certainly not able to be classified without the span of entities.

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**OT**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
`NM`	`[CLS]`	78.0	78.1	78.0	28.0	13.1	17.8	39.1	37.8	38.4	53.3	28.2	36.9
`NM`	`Es`	84.9	84.7	84.7	64.9	45.8	53.7	81.5	74.5	**77.8**	72.5	66.6	**69.5**
`NM`	`[CLS]+Es`	86.2	83.8	**84.9**	67.3	50.7	**57.8**	82.1	71.4	76.4	72.1	67.5	69.7

Table 31: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to `Input type` and `Output type`. `NM` represents `No Markers`.

Dataset	Gold	Data sample
TACRED	per:employee_of	*But Professor Chen Tao, of with Beijing-based China Youth University for Political Sciences, said the public should focus on problems behind the case rather than the reputations of celebrities.*
	per:title	*But Professor Chen Tao, of with Beijing-based China Youth University for Political Sciences, said the public should focus on problems behind the case rather than the reputations of celebrities.*
	no_relation	*But Professor Chen Tao, of with Beijing-based China Youth University for Political Sciences, said the public should focus on problems behind the case rather than the reputations of celebrities.*
Biocreative	INHIBITOR	*Nicotinic-receptor potentiator drugs, huprine X and galantamine, increase ACh release by blocking AChE activity but not acting on nicotinic receptors.*
	UNK	*Nicotinic-receptor potentiator drugs, huprine X and galantamine, increase ACh release by blocking AChE activity but not acting on nicotinic receptors.*
		Nicotinic-receptor potentiator drugs, huprine X and galantamine, increase ACh release by blocking AChE activity but not acting on nicotinic receptors.

Table 32: Examples of same textual input with the different span of entities and relation labels from TACRED and Biocreative VII Track 1. *Gold* represents the gold standard.

4.2.2 Comparison on `Entity Markers` input type module

This section analyzes various output type modules on `Entity Markers` input type module. To explore the usefulness of enriching the usage of output vectors, the analysis is presented of `Es` module, and `[CLS]+Es` module with the variation of them using surrounding vectors.

As it is presented in Table 33, `Es` module outputs slightly improved performance compared to `[CLS]` module in most cases. `Es` module can be enhanced by enriching the

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**OT**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
EM	[CLS]	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
EM	Es	84.6	89.0	86.4	69.2	64.1	66.6	88.0	87.7	87.8	72.8	75.2	**74.0**
EM	Es+mid	87.3	89.5	88.3	72.8	60.8	66.2	87.8	86.8	87.3	76.0	69.9	72.8
EM	Es+post	88.4	88.7	88.5	69.8	60.3	64.7	87.9	87.3	87.6	77.0	69.5	73.1
EM	Es+mid+post	86.9	88.5	87.6	75.0	59.3	66.2	88.2	87.4	87.8	75.2	70.9	73.0
EM	[CLS]+Es	91.0	85.0	87.8	72.9	61.0	66.4	87.9	86.6	87.2	77.0	71.2	**74.0**
EM	[CLS]+Es+mid	91.0	86.5	88.7	72.3	60.8	66.0	88.7	86.7	87.7	76.4	70.6	73.4
EM	[CLS]+Es+post	86.3	90.2	88.1	74.6	58.6	65.6	88.1	87.6	87.9	75.8	70.5	73.1
EM	[CLS]+Es+mid+post	90.2	87.9	**89.0**	72.0	62.7	**67.0**	88.4	87.6	**88.0**	76.6	69.3	72.8

Table 33: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to `Input type` and `Output type`. `EM` represents `Entity Markers`.

choice of vectors such as `Es+mid`, `Es+post`, and `Es+mid+post`. These systems output improved f1-score by increased precision and recall on SemEval 2010 Task 8 compared to `Es`. However, `Es+mid`, `Es+post`, and `Es+mid+post` present different performance on TACRED and Biocreative VII Track 1. For instance, these systems show dropped recall, whereas precision is increased. `[CLS]+Es` module outputs enhanced precision in most tasks and this pattern also can be observed on `[CLS]+Es+mid` module. However, `[CLS]+Es+post` shows somewhat different results compared to `[CLS]+Es` and `[CLS]+Es+mid`. For instance, `[CLS]+Es+post` shows decreased precision, whereas it achieved the best recall on SemEval 2010 Task 8. Also, the improvement of precision is slightly weakened on Biocreative VII Track 1. `[CLS]+Es+mid+post` outputs the best f1-score in most tasks by providing improved precision and a decent recall.

The experimental results show `Es` and `[CLS]+Es` achieved the best performance on Biocreative VII Track 1, whereas `[CLS]+Es+mid+post` module outputs the best f1-score in

the other tasks. The experiment results on Biocreative VII Track 1 are somewhat contrary to the assumption that the system can benefit from ample information by extending the usage of output vectors. Table 34 presents examples that are well classified on `Es` and `[CLS]+Es`, but not on `[CLS]+Es+mid+post` through thorough analysis on Biocreative VII Track 1. The most frequently misclassified data samples of `[CLS]+Es+mid+post` module in Biocreative VII Track 1 belong to *DIRECT-REGULATOR* relation. The highlighted word *antagonist* from Table 34 can be regularly detected in *ANTAGONIST* relation rather than *DIRECT-REGULATOR* relation. Moreover, the highlighted words on *INHIBITOR* prediction examples that represent inhibiting or decreasing the activity are commonly observed in *INHIBITOR* relation. Since *INHIBITOR* relation data samples outnumber *DIRECT-REGULATOR* relation data samples, `[CLS]+Es+mid+post` module tends to predict *DIRECT-REGULATOR* relation to *INHIBITOR* relation more often when these highlight words are appeared. Therefore, `Es` and `[CLS]+Es` module outperforms other output type module systems that utilize `mid` or `post` on Biocreative VII Track 1.

Gold	Prediction	Data sample
DIRECT-REGULATOR	ANTAGONIST	*In conclusion, these findings indicate that [3H]SR 142948A is a new potent antagonist radioligand which recognizes with high affinity both neurotensin NT1 and NT2 receptors and represents thus an excellent tool to study neurotensin receptors in the rat brain.*
	INHIBITOR	*Citalopram protected against the RTI-76-induced inhibition of SERT binding.*
		The structurally diverse opioids codeine and eseroline, like galantamine, are also nAChR-APL that have greatly diminished affinity for AChE, representing potential lead compounds for selective nAChR-APL development.
		Salicylates inhibit (125)I-ET-1 binding to recombinant rat ETA receptors.
		Losartan (parent compound), has moderate affinity for the AT(1) receptor (competitive inhibition).

Table 34: Examples that are miclassified in `[CLS]+Es+mid+post` from Biocreative VII Track 1. The underlined words represent the span of entities. *Gold* represents the gold standard.

4.2.3 Integrated comparison of Output type modules

Table 35 presents various output type modules on both `No Markers` and `Entity Markers`. The comparison of the experimental results from run 1, 2, 3 and 4 suggests `Entity Markers`

outperform `No Markers`. The experiment on run 4, 5 and 9 indicates `Es` and `[CLS]+Es` output better performance than `[CLS]`. The experiment on run 12 suggests `[CLS]+Es+mid+post` module performs well in most tasks but this approach yields poor performance on Biocreative VII Track 1.

To conclude, leveraging entity vectors shows more promising performance compared to the system that only uses CLS vector regardless of adding entity markers.

Idx	Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
	IT	OT	P	R	F	P	R	F	P	R	F	P	R	F
1	NM	[CLS]	78.0	78.1	78.0	28.0	13.1	17.8	39.1	37.8	38.4	53.3	28.2	36.9
2	NM	Es	84.9	84.7	84.7	64.9	45.8	53.7	81.5	74.5	77.8	72.5	66.6	69.5
3	NM	[CLS]+Es	86.2	83.8	84.9	67.3	50.7	57.8	82.1	71.4	76.4	72.1	67.5	69.7
4	EM	[CLS]	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
5	EM	Es	84.6	89.0	86.4	69.2	64.1	66.6	88.0	87.7	87.8	72.8	75.2	**74.0**
6	EM	Es+mid	87.3	89.5	88.3	72.8	60.8	66.2	87.8	86.8	87.3	76.0	69.9	72.8
7	EM	Es+post	88.4	88.7	88.5	69.8	60.3	64.7	87.9	87.3	87.6	77.0	69.5	73.1
8	EM	Es+mid+post	86.9	88.5	87.6	75.0	59.3	66.2	88.2	87.4	87.8	75.2	70.9	73.0
9	EM	[CLS]+Es	91.0	85.0	87.8	72.9	61.0	66.4	87.9	86.6	87.2	77.0	71.2	**74.0**
10	EM	[CLS]+Es+mid	91.0	86.5	88.7	72.3	60.8	66.0	88.7	86.7	87.7	76.4	70.6	73.4
11	EM	[CLS]+Es+post	86.3	90.2	88.1	74.6	58.6	65.6	88.1	87.6	87.9	75.8	70.5	73.1
12	EM	[CLS]+Es+mid+post	90.2	87.9	**89.0**	72.0	62.7	**67.0**	88.4	87.6	**88.0**	76.6	69.3	72.8

Table 35: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets. **IT** and **OT** refers to `Input type` and `Output type`. `NM` and `EM` represents `No Markers` and `Entity Markers` respectively.

4.3 Study on Input format modules

This section analyzes various input format modules based on the experiment results. The performance results of various input format module are presented in Table 36. The analysis of input format modules is based on two major perspectives: the influence of the position of attachment, and the significance of the presence of [SEP] token between attached entities.

In terms of the influence of the position of attachment, `Es+Standard` and `Standard+Es` module output improved performance compared to `Standard` module in general as presented in Table 36. This improvement can be observed on precision rather than recall. Through the in-depth analysis of `Es+Standard` and `Standard+Es` module, these modules enhance precision in different manners. `Es+Standard` module gains its improvement by increasing true-positive cases while `Standard+Es` module enhances precision by decreasing

Module		SemEval 2010			TACRED			Re-TACRED			Biocreative VII		
IT	**IF**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
EM	Standard	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
EM	Es+Standard	89.3	87.3	**88.3**	65.4	67.2	66.3	88.8	83.6	86.1	75.7	69.8	72.7
EM	Standard+Es	89.1	86.1	87.6	66.6	65.1	65.8	87.2	86.1	86.6	74.8	71.3	73.0
EM	Es(SEP)+Standard	89.4	86.5	87.9	68.5	64.3	66.3	87.6	85.7	86.6	74.9	73.2	**74.1**
EM	Standard+Es(SEP)	89.0	87.5	88.2	68.9	66.6	**67.7**	87.7	86.8	**87.2**	74.0	70.4	72.2

Table 36: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets and Biocreative VII Track 1 on dev sets. **IT** and **IF** refers to `Input type` and `Input format`. `EM` refers to `Entity Markers`.

false-positive cases. The relation extraction tasks studied in this thesis are multi-class classification tasks that the results of the negative label are not considered. For this reason, the performance only relies on the classification results of positive labels. The method of increasing true-positive cases in these classification tasks is the same as in binary classification tasks. The increase in true-positive cases can be obtained by predicting the positive label well. In binary classification tasks, the decrease in false-positive cases represent the increase of true-positive cases. However, the decrease in false-positive cases in multi-class classification tasks can be obtained either by increasing true-positive cases of positive labels or decreasing false-negative cases of negative label. `Es+Standard` module improves precision by increasing true-positive cases, whereas `Standard+Es` module enhances precision by lowering false-negative cases of negative label. Therefore, the improvement of `Standard+Es` module on precision indicates that the misclassified positive labels are classified as negative label rather than other positive labels. Moreover, both `Standard+Es` and `Standard+Es(SEP)` modules show similar pattern of `Es+Standard` and `Standard+Es` modules.

The comparison of input format modules based on the significance of the presence of [SEP] token between attached entities is presented in Table 36. `Es+Standard` and `Es(SEP)+Standard` module shows a somewhat different performance of precision and recall. For instance, `Es+Standard` module yields declined performance of recall on Re-TACRED and Biocreative VII Track 1, whereas `Es(SEP)+Standard` module achieved improved performance of recall on Re-TACRED. However, `Standard+Es` and `Standard+Es(SEP)` module outputs similar precision performance in all tasks.

In conclusion, the variation of input format modules improves precision in general. Moreover, this improvement can be observed regardless of the position of the attachment and the presence of [SEP] token between attached entities. However, precision may be improved differently depending on the position of attachment.

4.4 Architectures based on a linear and a bilinear classifier

This section investigates different architectures that are based on two different classifiers by providing the experimental results. The experiment result of different architectures based on a linear and a bilinear classifier are presented in Table 37. M_{Base}, M_E and M_R architectures applied a linear classifier and M_{Bie} and M_{Bicls} used a bilinear classifier. To evaluate the significance of the choice of a classifier, the analysis is focused on the performance comparison between M_E and M_{Bie}, and M_R and M_{Bicls}. The input type applied in this experiment is `EM` and the output type is `[CLS]`.

Architecture	**SemEval 2010**			**TACRED**			**Re-TACRED**			**Biocreative VII**		
	P	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
M_{Base}	86.2	87.4	86.6	66.2	64.1	65.1	87.2	85.7	86.4	71.1	74.5	72.7
M_E	84.6	89.0	86.4	69.2	64.1	66.6	88.0	87.7	**87.8**	72.8	75.2	74.0
M_R	91.0	85.0	87.8	72.9	61.0	66.4	87.9	86.6	87.2	77.0	71.2	74.0
M_{Bie}	90.4	87.2	**88.7**	72.9	60.5	66.1	88.3	86.6	87.4	74.3	73.1	73.7
M_{Bicls}	90.4	87.0	88.6	70.5	65.2	**67.8**	88.3	85.5	86.9	75.8	72.6	**74.2**

Table 37: Experiment results of SemEval 2010 Task 8, TACRED and Re-TACRED on test sets, and Biocreative VII Track 1 on dev sets.

To investigate the importance of a classifier when entities are the input of a classifier, the comparison of M_E and M_{Bie} are focused. M_E and M_{Bie} architecture are the same in terms of leveraging vectors, but they are different in terms of a choice of a classifier. The experiment on SemEval 2010 Task 8 shows that M_E outputs improved recall but dropped precision compared to M_{Base}. However, M_{Bie} yields somewhat opposite results where precision is enhanced while maintaining recall. M_E shows improved precision and recall on other tasks. M_{Bie} consistently shows improved precision but it outputs dropped recall on TACRED and Biocreative VII Track 1. The experiment results on M_E and M_{Bie} show that even though both of them used the same vectors, these architectures represent improvement in a different manner. For instance, M_E tends to output improved recall while M_{Bie} tends to yield enhanced precision.

To study the influence of the choice of a classifier, the comparison of M_R and M_{Bicls} where both use CLS and entity vectors are focused. M_R shows improved precision on all tasks compared to M_{Base}, especially the improvement is impressive except on Re-TACRED. M_{Bicls} also outputs improved precision in most tasks. The difference between M_R and M_{Bicls} is that M_R yields poor recall in most tasks, whereas M_{Bicls} shows slightly improved recall compared to M_R.

In general, applying a bilinear classifier tends to achieve improved performance compared to a linear classifier as presented in Table 37. However, the performance gap between classifiers is approximately 1% on SemEval 2010 Task 8, TACRED and Biocreative VII Track 1. Moreover, a linear classifier outperforms a bilinear classifier on Re-TACRED.

Overall, the experimental results demonstrate that M_R, M_{Bie}, and M_{Bicls} consistently output improved precision compared to M_{Base}. M_{Bicls} also shows a decent performance on recall so it achieved a prominent f1-score in most cases. However, the achievement is not greater compared to the best linear classifier systems.

4.5 Integrated analysis

Table 38 presents SOTA systems on each task and the integrated experiment results of all approaches introduced in this thesis. The boldface and the underlined performance represent the best and the second-best respectively. As presented in Table 38, `Entity Markers` module outperforms `No Markers` module in all tasks. `[CLS]+Es+mid+post` module achieved the best performance on SemEval 2010 Task 8 and Re-TACRED. M_{Bicls} system obtained the best performance on TACRED and Biocreative VII Track 1 but `Es(SEP)+Standard` and `Standard+Es(SEP)` show similar performance.

To investigate the level of achievement of the systems that are introduced in this thesis, SOTA systems on each task are described by comparing it with the best of introduced systems in this thesis. [Cohen et al., 2020] (Question Answering) earned SOTA performance on SemEval 2010 Task 8 by treating relation extraction task as a span-prediction problem, similar to question answering. In order to train the relation extraction tasks on question answering format, the hand-annotated question templates are required based on the relation and entity types. This SOTA system selected $BERT_{Large}$ language model. `[CLS]+Es+mid+post` module achieved near SOTA performance (2.9% difference). [Lyu and Chen, 2021] (Entity Type Restriction) obtained SOTA performance on TACRED by restricting candidate relations based on entity types. GCN [Zhang et al., 2018] and SpanBERT [Joshi et al., 2020] were used for the word representation. M_{Bicls} and `Standard+Es(SEP)` module improved performance compared to a baseline system, the difference between SOTA and these systems is greater than other tasks (7.5% difference). [Park and Kim, 2021] (Curriculum Learning) showed SOTA performance on Re-TACRED applying curriculum learning with $RoBERTa_{Large}$ language model. In Curriculum learning, the system begins training from easy data samples and gradually trains difficult data samples. [Park and Kim, 2021] used the cross review method [Xu et al., 2020] based on the prediction of the system in order to measure the difficulty of data samples. `[CLS]+Es+mid+post` module achieved near SOTA

System				Task			
State-of-the-art				**Sem**	**TAC**	**Re-TAC**	**Bio**
Question Answering [Cohen et al., 2020]				**91.9**	74.8	-	-
Entity Type Restriction [Lyu and Chen, 2021]				-	**75.2**	-	-
Curriculum Learning [Park and Kim, 2021]				-	75.0	**91.4**	-
Five BERT like systems+Majority Voting [Karabulut et al., 2021]				-	-	-	**77.7**
IT	**OT**	**IF**	**Arch**	**Sem**	**TAC**	**Re-TAC**	**Bio**
NM	[CLS]	-	-	78.0	17.8	38.4	36.9
NM	Es	-	-	84.7	53.7	77.8	69.5
NM	[CLS]+Es	-	-	84.9	57.8	76.4	69.7
EM	[CLS]	Standard	M_{Base}	86.6	65.1	86.4	72.7
EM	Es	-	M_E	86.4	66.6	87.8	74.0
EM	Es+mid	-	-	88.3	66.2	87.3	72.8
EM	Es+post	-	-	88.5	64.7	87.6	73.1
EM	Es+mid+post	-	-	87.6	66.2	87.8	73.0
EM	[CLS]+Es	-	M_R	87.8	66.4	87.2	74.0
EM	[CLS]+Es+mid	-	-	88.7	66.0	87.7	73.4
EM	[CLS]+Es+post	-	-	88.1	65.6	87.9	73.1
EM	[CLS]+Es+mid+post	-	-	**89.0**	67.0	**88.0**	72.8
EM	-	Es+Standard	-	88.3	66.3	86.1	72.7
EM	-	Standard+Es	-	87.6	65.8	86.6	73.0
EM	-	Es(SEP)+Standard	-	87.9	66.3	86.6	74.1
EM	-	Standard+Es(SEP)	-	88.2	67.7	87.2	72.2
EM	-	-	M_{Bie}	88.7	66.1	87.4	73.7
EM	-	-	M_{Bicls}	88.6	**67.8**	86.9	**74.2**

Table 38: F1-score of macro averaged results on SemEval 2010 Task 8 (Sem) test sets and micro averaged results on TACRED (TAC) and Re-TACRED (Re-TAC) test sets, and Biocreative VII Track 1 (Bio) dev sets. `IT`, `OT`, `IF` and `Arch` represent `Input type module`, `Output type module`, `Input format module` and `Architecture` respectively.

performance (2.4% difference). [Karabulut et al., 2021] (Five BERT like systems+Majority Voting) achieved SOTA performance on Biocreative VII Track 1 leveraging a group of BERT and majority voting methods. The group of BERT includes PubMedBERT [Gu et al., 2021], fine-tuned and class-weighted loss function applied PubMedBERT, PubMedBERT that utilizes LSTM at the last layer, BioMELECTRAL ([Alrowili and Vijay-Shanker, 2021]; [Clark et al., 2020]), and BioBERT [Lee et al., 2020] where the input uses entity markers. M_{Bicls} and `Es(SEP)+Standard` module obtained near SOTA performance (3.6% difference).

In conclusion, `Entity Markers` module outperforms `No Markers` module regardless of the choice of output vector. Moreover, leveraging CLS, entity, and surrounding vectors perform well on most tasks. The system of selecting a bilinear classifier shows improved results compared to a baseline system but the performance is not greater when it comes to comparing it with the best system of input & output representation system. SOTA systems used a larger language model or group of language models, and leveraged task-specific methods rather than generalizable methods. However, this thesis shows that input & output representation methods can achieve performance close to SOTA in most tasks with $BERT_{Base}$ language model without changing the method based on the task. Also, the complexity of various input & output representation methods are similar to a baseline system as presented in Table 39.

Output Type	**Input Format**	**Arch**	**Number of parameters (M)**

`[CLS]`	`Standard`
	`Es+Standard, Standard+Es,`
	`Es(SEP)+Standard, Standard+Es(SEP)`
		M_{Base}	109.5

`Es`	-	M_E	110.1
`[CLS]+Es, Es+mid, Es+post`	-	M_R	110.7

`Es+mid+post`
`[CLS]+Es+mid`
`[CLS]+Es+post`
	-	-	111.3

`[CLS]+Es+mid+post`	-	-	111.9
-	-	M_{Bie}	121.9
-	-	M_{Bicls}	155.5

Table 39: Complexity of various input & output representation modules and architectures.

Chapter 5

Conclusion and Future Directions

In this thesis, a set of ablation studies of various input & output representations, and architectural approaches based on a linear and a bilinear classifier were studied to a variety of relation extraction tasks. The main contribution of this thesis is to investigate the significance of leveraging various representations and the influence of a classifier by comparing them based on the performance and the complexity as well. To analyze this thoroughly, SOTA performance was also presented in order to evaluate the achievement of applied methods.

As a result of the analysis of experiments, there are several observations to achieve improved performance to the basic usage of BERT. First of all, the use of straightforward and inspectable input encoding has a great influence on the performance of relation extraction tasks. This suggests the performance can be improved by explicitly indicating the span of entities regardless of the choice of vector to a classifier. In addition, this input encoding provides the same complexity as a system that does not use this method. Second, enriching the use of output vectors has a strong potential of enhancing its performance. The experiment results indicate leveraging entity vectors improves the performance compared to applying CLS vector. Moreover, the performance of the system containing surrounding vectors is close to SOTA performance in most tasks. This approach slightly increases the complexity of the system, but the difference is marginal. Next, attaching additional entity tokens to the context improves precision in general. This improvement can be confirmed regardless of the attachment location or the separation of attached entities. The complexity of this approach is the same as the basic usage of BERT. Last, a bilinear classifier tends to yield improved performance compared to a linear classifier. However, in contrast to the experiment results from the literature review, the performance of a bilinear classifier was not always superior to a linear classifier, even though the complexity of a bilinear classifier

is larger than a linear classifier.

Considering the analysis of the integrated experiment results, the best performance introduced in this thesis is close to SOTA performance (3% difference) in general. Since $BERT_{Base}$ language model is applied in this thesis, this result suggests input & output representations approaches can achieve a promising performance with the smaller size of the language model and the less complexity.

As for future work, the observation from this study suggests the extension of input encoding. This study showed the significance of explicitly indicating the span of entities by comparing `No Markers` and `Entity Markers`. This suggests that this straightforward and inspectable input encoding improves the usefulness of BERT on relation extraction tasks regardless of its dataset characteristics. Considering this observation, potential improvement can be obtained by extending the use of markers that indicate the span of specific tokens. For instance, Example (19) illustrates the relationship between entities *Calluses* and *skin abnormality*.

(19) *Calluses are caused by improperly fitting shoes or by a skin abnormality.*

`Cause-Effect(skin abnormality, calluses)`

As regards the typed dependency perspective, Example (19) is presented as Figure 29 where entities are linked through typed dependency relations in a sentence. Since entities are

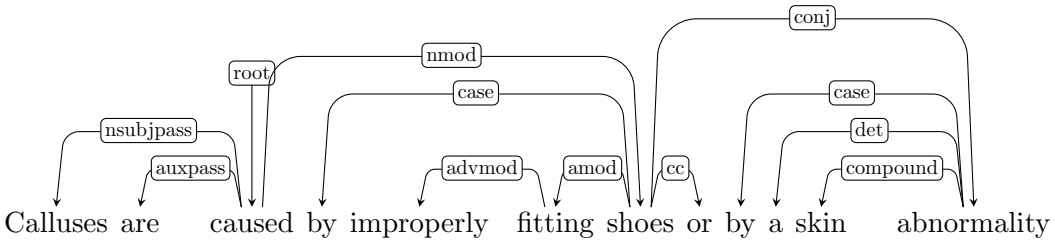


Figure 29: Example of typed dependency parsing.

grammatically connected through word *caused* and *shoes* based on the typed dependency result, these two words are the candidates to be enclosed by entity markers. Moreover, entities have a relation of `Cause-Effect` as described in Example (19) and the relation can be derived from the candidate words that grammatically connects entities. This represents indicating the span of all candidate words or some of the selected candidates can provide BERT how entities are connected in the context. For this reason, it is worth further study on the extension of the straightforward and inspectable input encoding.

Appendix A

Studies on various BERT-based language models

In this section, the experiment on several BERT like language models such as RoBERTa [Liu et al., 2019], DistilBERT and DistilRoBERTa [Sanh et al., 2019] is introduced. Since this experiment is comparatively less relevant to approaches introduced in this thesis, the experimental results are presented in this section. Table A.1 is the official result of BERT,

System	**GLUE Task**							
	MNLI	RTE	QQP	QNLI	SST-2	CoLA	STS-B	MRPC
Pre-BERT SOTA	82.1	56.0	70.3	87.4	91.3	45.4	80.0	82.3
BERT_{base}	84.6	66.4	71.2	90.5	93.5	52.1	85.8	88.9
RoBERTa_{base}	**87.6**	**78.7**	**91.9**	**92.8**	**94.8**	**63.6**	**91.2**	**90.2**
DistilBERT_{base}	82.2	59.9	88.5	89.2	91.3	51.3	85.8	87.5
$\text{DistilRoBERTa}_{base}$	84.0	67.9	89.4	90.8	92.5	59.3	88.3	86.6

Table A.1: Performance of various BERT-based language models on GLUE test dataset except for WNLI task. Pre-BERT SOTA system is OpenAI GPT [Radford et al., 2018] which is based on transformer that replaced previous SOTA systems.

RoBERTa, DistilBERT and DistilRoBERTa on GLUE test dataset. RoBERTa, which has the largest number of parameters and has been pre-trained on relatively large data compared to other language models, yields the best performance among different BERT-based language models. Moreover, DistilRoBERTa which is the lighter version of RoBERTa outputs improved performance compared to DistilBERT.

The objective of the GLUE benchmark is to observe the same level of success on different NLP tasks. This is because before the advent of this benchmark NLP systems were usually

Dataset	Task	Task description
MNLI	Natural Language Inference	Given a premise sentence and a hypothesis sentence, the task is to predict whether the premise entails, contradicts, or neither the hypothesis.
RTE	Natural Language Inference	Given sentence A and sentence B, the task is to predict whether sentence A entails sentence B.
QQP	Paraphrase	Given two questions, the task is to predict whether two questions are semantically equivalent.
QNLI	Question Answering	Given a question sentence and context sentence, the task is to predict whether the context sentence is the answer to the question sentence.
SST-2	Sentiment	Given a movie review, the task is to predict whether the review is positive, neutral, or negative.
CoLA	Acceptability	Given a sentence, the task is to predict whether the sentence is either grammatical or ungrammatical.
STS-B	Sentence Similarity	Given two sentences, the task is to predict the score of similarity between two sentences.
MRPC	Paraphrase	Given two sentences, the task is to predict whether the sentences are semantically equivalent.
WNLI	Coreference	Given sentence A, and sentence B which replaces sentence A's pronoun with one of the nouns, the task is to predict whether the replaced noun in sentence B is the correct referent.

Table A.2: Brief description of GLUE task

built based on a certain task so it was difficult to expand its usage. Therefore, each NLP system was difficult to generalize and to compare performance with each other. GLUE consists of nine task datasets designed to test NLP systems as presented in Table A.2. GLUE benchmark datasets are diverse in dataset size, difficulty, context and objective. In recent days, announcing the experiment results of GLUE datasets is considered one of the essential indicators in order to investigate the generalization and flexibility of the NLP system.

	Number of parameters (M)			
Architecture	**BERT**	**RoBERTa**	**DistilBERT**	**DistilRoBERTa**
M_{Base}	109.5	124.7	66.4	82.1
M_{E}	110.1	125.3	67.0	82.7
M_{R}	110.7	125.9	67.6	83.3
M_{Bie}	121.9	137.0	78.8	94.5
M_{Bicls}	155.5	170.7	112.4	128.1

Table A.3: Number of parameters on various architectures of varied BERT language models. Note that, all language models in this table are base models.

In order to investigate the relationship of performance, the size of the model, and the size of the pre-trained dataset a set of ablation studies on various BERT-based language models were conducted. As depicted in Figure A.3, the size of architecture is affected by not only the choice of classifier and its input size but also the selection of language model as well.

Table A.4 presents the experiment results of various BERT-based model on SemEval 2010 Task 8. Even though M_{Bicls} is more complex system than M_{Bie}, M_{Bie} yields enhanced performance. While M_{Bie} of RoBERTa yields the best f1-score, both BERT and DistilBERT yield promising performance close to the performance of RoBERTa. Therefore, M_{Bie} of RoBERTa is not a promising choice based on the complexity perspective.

The experiment results on TACRED is illustrated in Table A.5. Interestingly, M_{Bicls} of BERT outputs the best f1-score in spite of the fact that RoBERTa is pre-trained on a bigger dataset and has a bigger vocabulary size than BERT. This represents quite opposite pattern to the results of the GLUE task where RoBERTa outperforms BERT. In TACRED, DistilRoBERTa outperforms DistilBERT, unlike SemEval 2010 Task 8 result. However, M_{Bicls} of DistilBERT yields the best precision among all results even though M_{Bicls} of DistilBERT performance provides less promising f1-score. Unlike the pattern that is observed in Table A.4 where the performance of DistilBERT and DistilRoBERTa systems surpass some

SemEval 2010 Task 8												
	BERT			**RoBERTa**			**DistilBERT**			**DistilRoBERTa**		
Architecture	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
M_{Base}	86.2	87.4	86.6	88.5	85.9	87.1	84.8	86.8	85.7	85.0	84.2	84.5
M_E	84.6	89.0	86.4	87.6	88.4	87.9	88.7	85.1	86.7	84.2	85.8	84.9
M_R	91.0	85.0	87.8	87.6	90.5	89.0	88.9	86.3	87.6	84.6	87.8	86.1
M_{Bie}	90.4	87.2	**88.7**	87.8	90.4	**89.1**	89.7	87.2	**88.4**	85.7	88.0	**86.8**
M_{Bicls}	90.4	87.0	88.6	88.8	85.2	86.8	85.2	87.6	86.2	85.7	84.1	84.8

Table A.4: Macro averaged Precision (P), Recall (R) and F1-score (F) results on SemEval 2010 Task 8 test sets on various architectures with various language models.

TACRED												
	BERT			**RoBERTa**			**DistilBERT**			**DistilRoBERTa**		
Architecture	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
M_{Base}	66.2	64.1	65.1	71.0	64.7	**67.7**	66.3	62.0	64.1	66.4	62.4	64.3
M_E	69.2	64.1	66.6	67.2	65.8	66.5	71.1	59.1	**64.6**	69.2	59.9	64.2
M_R	72.9	61.0	66.4	68.2	66.6	67.4	69.8	59.3	64.1	68.2	64.1	**66.1**
M_{Bie}	72.9	60.5	66.1	65.7	67.2	66.5	66.0	62.9	64.4	67.8	62.8	65.2
M_{Bicls}	70.5	65.2	**67.8**	72.9	62.2	67.1	76.2	50.3	60.6	69.1	60.1	64.3

Table A.5: Micro averaged Precision (P), Recall (R) and F1-score (F) results on TACRED test sets on various architectures with various language models.

RoBERTa-based performance, both DistilBERT and DistilRoBERTa performance were not able to achieve a comparable performance of RoBERTa systems in TACRED.

As presented in Table A.6, systems that adopted a bilinear classifier tend to show better performance in Re-TACRED. M_{Bie} of RoBERTa provides the best performance. Unlike SemEval 2010 Task 8 experiment results, DistilRoBERTa consistently yields better performance compared to DistilBERT in Re-TACRED. Even though a bilinear classifier applied systems output improved performance compared to a linear classifier, the performance gap is trivial.

Re-TACRED												
	BERT			**RoBERTa**			**DistilBERT**			**DistilRoBERTa**		
Architecture	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
M_{Base}	87.2	85.7	86.4	88.1	88.9	88.5	84.6	82.4	83.5	83.9	85.5	84.7
M_E	88.0	87.7	**87.8**	89.1	87.2	88.1	87.1	84.2	85.6	88.0	83.8	85.8
M_R	87.9	86.6	87.2	86.8	88.4	87.6	83.4	84.6	84.0	83.9	88.5	86.1
M_{Bie}	88.3	86.6	87.4	88.6	88.6	**88.6**	83.9	85.2	84.6	84.4	86.5	85.4
M_{Bicls}	88.3	85.5	86.9	88.1	88.9	88.5	86.9	84.5	**85.7**	88.4	84.5	**86.4**

Table A.6: Micro averaged Precision (P), Recall (R) and F1-score (F) results on Re-TACRED test sets on various architectures with various language models.

As it is illustrated in Table A.7, the performance gap between four different language models of M_{Base} is trivial. However, BERT and RoBERTa show a more clear achievement compared to the distilled version of them when two entity vectors are applied at the classification step. In most cases, BERT obtained better recall, but RoBERTa outperformed BERT in f1-score because of its improvement in precision.

Biocreative VII Track 1												
	BERT			**RoBERTa**			**DistilBERT**			**DistilRoBERTa**		
Architecture	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
M_{Base}	71.1	74.5	72.7	71.4	71.1	71.2	75.4	69.1	72.1	71.0	68.3	69.6
M_E	72.8	75.2	74.0	73.2	74.7	74.0	76.0	67.2	71.3	72.7	69.1	70.1
M_R	77.0	71.2	74.0	78.0	72.6	**75.2**	72.1	72.1	72.1	74.2	69.2	71.6
M_{Bie}	74.3	73.1	73.7	78.2	71.2	74.5	75.8	69.4	**72.4**	73.4	70.8	72.1
M_{Bicls}	75.8	72.6	**74.2**	76.0	71.9	73.9	76.8	67.2	71.7	75.1	70.5	**72.7**

Table A.7: Micro averaged Precision (P), Recall (R) and F1-score (F) results on Biocreative VII Track 1 dev sets on various architectures with various language models.

In conclusion, RoBERTa outperforms BERT, DistilBERT, and DistilRoBERTa on relation extraction tasks conducted in this thesis in most cases similar to as it is shown on official performance on GLUE task. However, the performance gap is trivial, and BERT outperforms RoBERTa in some cases. In general, both distilled versions of BERT and RoBERTa output a decent performance, even though both are relatively less complex system than the original version. Therefore, if the complexity is a critical issue when performing the task, DistilBERT and DistilRoBERTa are considered appropriate choices. As regards with the application of a classifier, both a linear and a bilinear classifier yield the best performance depending on its task and language models and the performance gap is trivial. Overall, the choice of the language model is more important than the choice of a classifier in order to obtain improved performance.

Bibliography

[Alrowili and Vijay-Shanker, 2021] Alrowili, S. and Vijay-Shanker, K. (2021). Biomtransformers: building large biomedical language models with bert, albert and electra. In *Proceedings of the 20th Workshop on Biomedical Language Processing*, pages 221–227.

[Alt et al., 2019] Alt, C., Hübner, M., and Hennig, L. (2019). Improving relation extraction by pre-trained language representations. *arXiv preprint arXiv:1906.03088*.

[Bucilu et al., 2006] Bucilu, C., Caruana, R., and Niculescu-Mizil, A. (2006). Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541.

[Chen and Manning, 2014] Chen, D. and Manning, C. D. (2014). A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 740–750.

[Chen et al., 2010] Chen, Y., Lan, M., Su, J., Zhou, Z. M., and Xu, Y. (2010). Ecnu: effective semantic relations classification without complicated features or multiple external corpora. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 226–229.

[Cho et al., 2014] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

[Clark et al., 2020] Clark, K., Luong, M.-T., Le, Q. V., and Manning, C. D. (2020). Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

[Cohen et al., 2020] Cohen, A. D., Rosenman, S., and Goldberg, Y. (2020). Relation classification as two-way span-prediction. *arXiv preprint arXiv:2010.04829*.

[De Marneffe and Manning, 2008] De Marneffe, M.-C. and Manning, C. D. (2008). The stanford typed dependencies representation. In *Coling 2008: proceedings of the workshop on cross-framework and cross-domain parser evaluation*, pages 1–8.

[Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

[Ehrlinger and Wöß, 2016] Ehrlinger, L. and Wöß, W. (2016). Towards a definition of knowledge graphs. *SEMANTiCS (Posters, Demos, SuCCESS)*, 48(1-4):2.

[Elman, 1990] Elman, J. L. (1990). Finding structure in time. *Cognitive science*, 14(2):179–211.

[Girju et al., 2007] Girju, R., Nakov, P., Nastase, V., Szpakowicz, S., Turney, P., and Yuret, D. (2007). Semeval-2007 task 04: Classification of semantic relations between nominals. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 13–18.

[Gokaslan and Cohen, 2019] Gokaslan, A. and Cohen, V. (2019). Openwebtext corpus.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

[Gu et al., 2021] Gu, Y., Tinn, R., Cheng, H., Lucas, M., Usuyama, N., Liu, X., Naumann, T., Gao, J., and Poon, H. (2021). Domain-specific language model pretraining for biomedical natural language processing. *ACM Transactions on Computing for Healthcare (HEALTH)*, 3(1):1–23.

[Hendrickx et al., 2009] Hendrickx, I., Kim, S. N., Kozareva, Z., Nakov, P., Séaghdha, D. Ó., Padó, S., Pennacchiotti, M., Romano, L., and Szpakowicz, S. (2009). Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *SEW@ NAACL-HLT*.

[Hinton et al., 2015] Hinton, G., Vinyals, O., and Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

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

[Joshi et al., 2020] Joshi, M., Chen, D., Liu, Y., Weld, D. S., Zettlemoyer, L., and Levy, O. (2020). Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

[Karabulut et al., 2021] Karabulut, M. E., Vijay-Shanker, K., and Peng, Y. (2021). Cu-ud: text-mining drug and chemical-protein interactions with ensembles of bert-based models. *arXiv preprint arXiv:2112.03004*.

[Karlik and Olgac, 2011] Karlik, B. and Olgac, A. V. (2011). Performance analysis of various activation functions in generalized mlp architectures of neural networks. *International Journal of Artificial Intelligence and Expert Systems*, 1(4):111–122.

[Lee et al., 2020] Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., and Kang, J. (2020). Biobert: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 36(4):1234–1240.

[Liu et al., 2019] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

[Loper and Bird, 2002] Loper, E. and Bird, S. (2002). Nltk: The natural language toolkit. *arXiv preprint cs/0205028*.

[Luong et al., 2014] Luong, M.-T., Sutskever, I., Le, Q. V., Vinyals, O., and Zaremba, W. (2014). Addressing the rare word problem in neural machine translation. *arXiv preprint arXiv:1410.8206*.

[Lyu and Chen, 2021] Lyu, S. and Chen, H. (2021). Relation classification with entity type restriction. *arXiv preprint arXiv:2105.08393*.

[Marcus et al., 1994] Marcus, M., Kim, G., Marcinkiewicz, M. A., MacIntyre, R., Bies, A., Ferguson, M., Katz, K., and Schasberger, B. (1994). The penn treebank: Annotating predicate argument structure. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

[Maynard et al., 2000] Maynard, D., Cunningham, H., Bontcheva, K., Catizone, R., Demetriou, G., Gaizauskas, R., Hamza, O., Hepple, M., Herring, P., Mitchell, B., et al. (2000). A survey of uses of gate. Technical report, Citeseer.

[Mel'cuk et al., 1988] Mel'cuk, I. A. et al. (1988). *Dependency syntax: theory and practice*. SUNY press.

[Mikolov et al., 2013] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[Miller, 1995] Miller, G. A. (1995). Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

[Miranda et al., 2021] Miranda, A., Mehryary, F., Luoma, J., Pyysalo, S., Valencia, A., and Krallinger, M. (2021). Overview of drugprot biocreative vii track: quality evaluation and large scale text mining of drug-gene/protein relations. In *Proceedings of the seventh BioCreative challenge evaluation workshop*.

[Nagel, 2016] Nagel, S. (2016). Cc-news.

[Nair and Hinton, 2010] Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Icml*.

[Neal, 1992] Neal, R. M. (1992). Connectionist learning of belief networks. *Artificial intelligence*, 56(1):71–113.

[Park and Kim, 2021] Park, S. and Kim, H. (2021). Improving sentence-level relation extraction through curriculum learning. *arXiv preprint arXiv:2107.09332*.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

[Peters et al., 2018] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

[Qi et al., 2020] Qi, P., Zhang, Y., Zhang, Y., Bolton, J., and Manning, C. D. (2020). Stanza: A python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 101–108.

[Radford et al., 2018] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training.

[Rink and Harabagiu, 2010] Rink, B. and Harabagiu, S. (2010). Utd: Classifying semantic relations by combining lexical and semantic resources. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 256–259.

[Sanh et al., 2019] Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

[Schuster and Nakajima, 2012] Schuster, M. and Nakajima, K. (2012). Japanese and korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5149–5152. IEEE.

[Sennrich et al., 2016] Sennrich, R., Haddow, B., and Birch, A. (2016). Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

[Shi and Lin, 2019] Shi, P. and Lin, J. (2019). Simple bert models for relation extraction and semantic role labeling. *arXiv preprint arXiv:1904.05255*.

[Soares et al., 2019] Soares, L. B., Fitzgerald, N., Ling, J., and Kwiatkowski, T. (2019). Matching the blanks: Distributional similarity for relation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905.

[Stoica et al., 2021] Stoica, G., Platanios, E. A., and Póczos, B. (2021). Re-tacred: Addressing shortcomings of the tacred dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13843–13850.

[Tang et al., 2020] Tang, H., Cao, Y., Zhang, Z., Cao, J., Fang, F., Wang, S., and Yin, P. (2020). Hin: Hierarchical inference network for document-level relation extraction. *Advances in Knowledge Discovery and Data Mining*, 12084:197.

[Tao et al., 2019] Tao, Q., Luo, X., Wang, H., and Xu, R. (2019). Enhancing relation extraction using syntactic indicators and sentential contexts. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1574–1580. IEEE.

[Trinh and Le, 2018] Trinh, T. H. and Le, Q. V. (2018). A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847*.

[Tymoshenko and Giuliano, 2010] Tymoshenko, K. and Giuliano, C. (2010). Fbk-irst: Semantic relation extraction using cyc. In *Proceedings of the 5th international workshop on semantic evaluation*, pages 214–217.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

[Wang et al., 2018] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. (2018). Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

[Wang et al., 2019] Wang, H., Focke, C., Sylvester, R., Mishra, N., and Wang, W. (2019). Fine-tune bert for docred with two-step process. *arXiv preprint arXiv:1909.11898*.

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

[Wu et al., 2016] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.

[Xu et al., 2020] Xu, B., Zhang, L., Mao, Z., Wang, Q., Xie, H., and Zhang, Y. (2020). Curriculum learning for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104.

[Yao et al., 2019] Yao, Y., Ye, D., Li, P., Han, X., Lin, Y., Liu, Z., Liu, Z., Huang, L., Zhou, J., and Sun, M. (2019). DocRED: A large-scale document-level relation extraction dataset. In *Proceedings of ACL 2019*.

[Yuan et al., 2012] Yuan, G.-X., Ho, C.-H., and Lin, C.-J. (2012). Recent advances of large-scale linear classification. *Proceedings of the IEEE*, 100(9):2584–2603.

[Zell, 1994] Zell, A. (1994). *Simulation neuronaler netze*, volume 1. Addison-Wesley Bonn.

[Zeng et al., 2014] Zeng, D., Liu, K., Lai, S., Zhou, G., and Zhao, J. (2014). Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344.

[Zhang et al., 2015] Zhang, S., Zheng, D., Hu, X., and Yang, M. (2015). Bidirectional long short-term memory networks for relation classification. In *Proceedings of the 29th Pacific Asia conference on language, information and computation*, pages 73–78.

[Zhang et al., 2018] Zhang, Y., Qi, P., and Manning, C. D. (2018). Graph convolution over pruned dependency trees improves relation extraction. *arXiv preprint arXiv:1809.10185*.

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

[Zheng et al., 2019] Zheng, H., Fu, J., Zha, Z.-J., and Luo, J. (2019). Learning deep bilinear transformation for fine-grained image representation. *arXiv preprint arXiv:1911.03621*.

[Zhu et al., 2015] Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. (2015). Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27.