Doctoral Dissertation

# Toward Improving Attentive Neural Networks in Legal Text Processing

Nguyen Ha Thanh

Supervisor: Nguyen Le Minh

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology

March, 2022

# Acknowledgements

I hereby declare that this dissertation is written by myself, from the observations and experiments I made while studying at JAIST as a doctoral student. Even so, to obtain the results described in this dissertation, I am indebted to many people.

First and foremost, I would like to express my best gratitude to Professor Nguyen Le Minh, who has imparted to me many valuable research experiences. He made me broaden my mind and believe in the possibility of making the impossible possible. He is the inspiration for me to strive in my research career.

I would like to thank Professor Satoshi Tojo, Associate Professor Kiyoaki Shirai, Professor Ken Satoh, and Professor Shinobu Hasegawa for their guidance and constructive suggestions to improve my research.

I also want to express gratitude to Associate Professor Nguyen Viet Ha, Professor Nguyen Dinh Duc, Professor Nguyen Thanh Thuy, and other teachers at VNU-UET and VNU-LS, who have inspired and built me up a strong research background before studying at JAIST.

I would like to thank my co-authors, colleagues and fraternities for the opportunity to work with them. In addition, I also highly appreciate authors whose works are cited in this dissertation. This dissertation would never have been done without the foundations they created.

I am grateful to Ministry of Education, Culture, Sports, Science and Technology (MEXT) and other research funds for supporting the necessary expenses for my research activities in Japan. I would also like to thank JAIST for providing me with excellent living and research facilities.

Finally and most sincerely, I would like to thank my parents, family and friends, who have always supported and given me great motivation to overcome difficult challenges to obtain initial results in scientific research.

*Nguyen Ha Thanh*
*Japan, 2022*

## Abstract

In recent years, thanks to breakthroughs in neural network techniques especially attentive deep learning models, natural language processing has made many impressive achievements. However, automated legal word processing is still a difficult branch of natural language processing. Legal sentences are often long and contain complicated legal terminologies. Hence, models that work well on general documents still face challenges in dealing with legal documents. We have verified the existence of this problem with our experiments in this work. In this dissertation, we selectively present the main achievements in improving attentive neural networks in automatic legal document processing. Language models tend to grow larger and larger, though, without expert knowledge, these models can still fail in domain adaptation, especially for specialized fields like law.

This dissertation has three main tasks to achieve the goal of improving attentive models in legal document processing. First, we survey and verify the factors affecting the performance of the models when operating on a specific domain such as law. This investigation is to provide clearer insights to improve models in this domain. Second, as pretrained language models are recently the most well-known attentive approaches in natural language processing, we provide methods to create language models specific to the legal domain, producing state-of-the-art results on reliable datasets. These models are built on features from the data of legal documents, with the goal of overcoming the challenges found in our previous survey. Third, besides the approach to let the model learn completely from raw data, we propose and prove the effectiveness of using different knowledge sources to inject into the model in different ways to adjust their output. This approach not only increases explainability but also allows humans to control pretrained language models and take advantage of the knowledge resources available during the development of the field such as vocabulary, grammar, logic and law.

***Keywords:*** Legal Text Processing, Attentive Neural Network, Deep Legal, Pretrained Language Model, Knowledge Injection.

# Contents

# List of Figures

# List of Tables

# Glossary

**BOW** Bag-of-words. 23

**CBOW** Continous Bag-of-words. 23

**CNN** Convolutional Neural Network. 3

**COLIEE** Competition on Legal Information Extraction/Entailment. 8

**Deep legal** Deep learning in processing legal text. 2

**FC** Fully Connected. 17

**MLM** Masked Language Model. 41

**MLP** Multilayer Perceptrons. 12

**NFSP** Next Foreign Sentence Prediction. 45

**NLI** Natural Language Inference. 30

**NLP** Natural Language Processing. 16

**NMSP** Neighbor Multilingual Sentence Prediction. 46

**NSP** Next Sentence Prediction. 41

**OOV** Out of Vocabulary. 41

**TE** Textual Entailment. 30

**TF-IDF** Term Frequency — Inverse Document Frequency. 23

**TPU** Tensor Processing Unit. 42

**TRE** Transferred-Requisite-Effective. 64

# Chapter 1

# Introduction

## 1.1 Introduction

Automated processing of legal documents is an urgent need in today's information society. Besides the convenience of social media, our actions on these platforms may involve or result in many legal effects. Legal questions about freedom of speech raised around Twitter[1]'s banning of former American President Donald Trump on their platform [57] or Tesla[2] has to hire employees to control the legal risk of their chairman Elon Musk's statements are good examples attesting to this phenomenon. However, for social and technical reasons, the quality of automatic law processing systems has not yet met the needs of society.

In terms of social reasons, computer science has made significant results only in recent years, while the law is a field that has been attached to people for centuries since the formation of countries. The law exists parallel to the development of mankind and for a long time, without any requirement of technology. Besides, both law and computer science are specialized academic disciplines that do not have much in common. Therefore, it may take a long time to get breakthroughs in the application of computer science to law.

For technical reasons, the sentences are often long and have a complex semantic structure. It is even difficult for a human to understand the exact meaning of a legal sentence on the first reading. There must be an interpreting role of the Court in the common law system in countries such as the United Kingdom, the United States of America, Canada; and guiding documents in the civil law system like in Germany, Japan, Vietnam. Besides, legal documents are written in natural language, a means of communication that is not designed for correctness. The ambiguity in the natural language could be an obstacle for any intelligent system, even

[1]https://twitter.com
[2]https://www.tesla.com

for human beings. Especially in languages with multi-layered meanings (such as Chinese, Japanese, Vietnamese), understanding the exact meaning through sentences is a more difficult problem. In addition, the vocabulary used in the legal domain does not completely coincide with the words that people use to communicate every day. Therefore, it can be considered as a special sublanguage in our language.

Along with the growth of hardware computation capacity, deep learning and especially attentive models have proven their power in many different tasks in natural language processing. Delicate tasks such as speech recognition, question answering, and language generation are all well performed by systems using this approach. Given such achievements, we can expect the possibility of using deep learning models in dealing with more complex linguistic tasks in the legal domain. In this dissertation, we selectively report our research results in improving the performance and explainability of deep learning especially attentive models in processing legal text (we use the term Deep legal processing in short) . Since legal language is different from daily language, we need an appropriate approach for this kind of data. Besides the enhancement in performance, the dissertation also provides the informative characteristics of deep legal processing for the readers.

Transfer learning and pretrained attentive models are robust approaches in domain adaptation. However, in a specialized domain like law, without an understanding of the domain and the data, it is difficult for these models to yield good results. Therefore, a detailed investigation of the possibilities and methods of applying deep learning to legal text processing is useful information for the development of automation in this field. The three main research questions would be answered in this dissertation include:

1. What factors impact the performance of end-to-end deep learning models trained with mere provisioning data perform legal document processing tasks?

2. Pretrained language models have become one of the powerful approaches in deep learning. What characteristics in the legal text can be used to implement successful instances of these models?

3. How to make use of available knowledge sources to inject into the deep learning models to have a better performance? Which kinds of knowledge are available?

To answer these questions, we make hypotheses and test them in specific problems. For each problem, we propose methods, conduct experiments, observe, analyze experimental results and draw conclusions.

## 1.2 Motivations

### 1.2.1 Factor Analysis for Deep Legal Systems

The first motivation of this study is to understand the factors that influence a deep legal system and to propose appropriate improvements based on these understandings. Conducting the works introduced in this dissertation, we focus on improving both the performance and the apparentness of deep legal models. Deep learning models are often considered black boxes, as long as there is enough data, they will achieve the desired effect. Even so, the assumption of enough data is hard to be satisfied in all areas of daily life. Therefore, analyzing the characteristics of deep legal helps us to use data more effectively. This dissertation also conveys information about what tasks and under what conditions can deep learning models perform well in the legal domain. This work can also be seen as an effort to increase the explainability of deep legal models, which is crucial to bring these models to real-life applications.

Understanding the factors that can affect systems in a domain is an important requirement for good designs. Data characteristics in the legal domain are fragmented data, long legal sentences, and many specialized terms. Therefore, we choose to investigate in detail factors such as the amount of data, the way the data is represented, and the architecture of the model working on the data. For the data amount factor, we experiment on a problem with limited data, propose solutions to increase the data and compare the results in the new setting. To understand the impact of data representation, we propose a method to evaluate different embedding methods in both the general and legal domains. About the model architecture, we compare the performance of different architectures on the same problem. The experimental results show an interesting superiority of an attentive CNN network compared with a pre-trained cumbersome language model with the vanilla architecture.

### 1.2.2 Pretrained Language Models for Deep Legal Processing

Our second motivation is to verify the ability of the pretrained language model in the legal domain. In recent years, pretrained language models have gained popularity and made many breakthroughs in various problems in natural language processing. Following this trend, we design pretrained language models for deep legal tasks. Besides performance, an important factor to evaluate models, we focus on philosophy when designing them. The models introduced are the result of observations drawn from the factors affecting deep legal models obtained from our investigation. Pretrained language models often contain biases that exist in the training data, so often perform poorly on a very distinct domain. Fortunately, for the legal field, we can take advantage of the data properties of this domain to train or adjust the weights of these models.

From observing the importance of data representation in the legal field, we propose a pretrained language model named BERTLaw, which is trained from scratch using a large amount of legal data. Besides achieving good results in our experiment, this model also helps us confirm the importance of data representation. Having a good data representation is a prerequisite for a strong deep legal system. In addition to BERTLaw, we introduce Paralaw and Paraformer, models based on pretrained language models that overcome the issues of data amount and model architecture limitations.

### 1.2.3 Knowledge Injection for Deep Legal Models

Our third motivation is to perform and leverage legal and linguistic knowledge sources to improve the performance of deep legal models. Deep learning models can learn from data and demonstrate their effectiveness on a wide range of tasks. However, relying solely on data has three disadvantages. First, the quality of the model is based on the quality of the data, or in other words garbage in garbage out. This can be dangerous when lay users are too dependent on data. Second, humans will be less likely to participate in the decision-making process. This can lead to the abuse of power by intelligent systems. Third, these systems are considered black boxes and debugging them is extremely difficult. Therefore, we investigate and propose approaches to inject knowledge into deep learning models to guide the learning and generation processes of these models.

For linguistic knowledge, we introduce HYDRA, an architecture that allows to train the transformer model's attention heads separately and then graft them onto the original body. This approach leads to cost-effectiveness in training as well as storage. For legal knowledge, we experiment with knowledge of logical parts of legal sentences. We use a special mechanism to inject this knowledge into the different layers of the transformer model. Finally, with the model of language generation in the legal field, we propose a method that uses the knowledge of fairness to regulate the output of this system. These findings are the basis for using other types of knowledge resources to improve future deep legal models.

## 1.3 Contributions

The dissertation contributes three main values: performance improvement, methodology, philosophy. First, the systems proposed in this dissertation all have better performance than existing baselines. Some of them achieve state-of-the-art results on reliable datasets. Second, the performance improvements of the systems are all based on methods designed from observations of experimental results. We not only explain the proposed methods in each chapter but also outline the journey to build them. Third, the conclusions and discussions in the sections of this dissertation have philosophical value in the design of deep legal models.

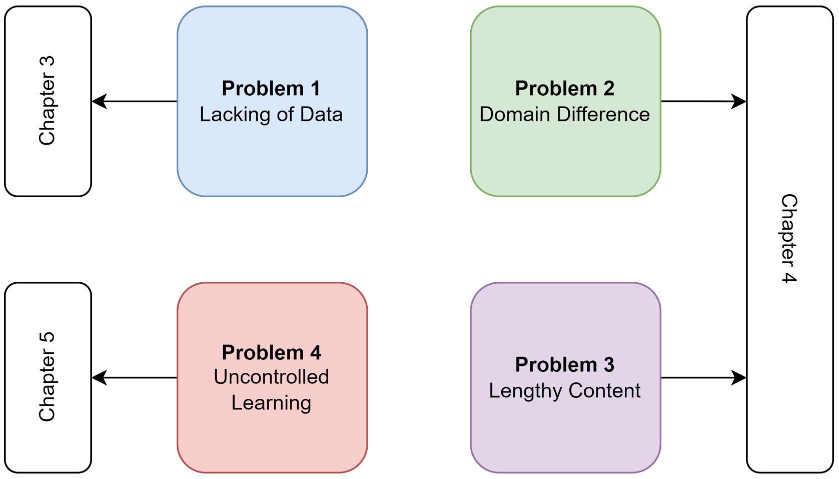


Figure 1.1: The main problems which are mentioned, analyzed and solved in the dissertation.

The main contribution of the dissertation includes discovery and settling 4 common problems for deep learning systems in the law domain, namely *lacking of data*, *domain difference*, *lengthy content*, and *uncontrolled learning* as demonstrated in Figure 1.1. Besides the non-architecture solutions, the models proposed in this dissertation all take advantage of the attention mechanism. The dissertation also shows that, without appropriate approaches, the power of attentive models can be wasted. This is especially demonstrated in the sections on Attentive CNN, pretrained language models, and Paraformer.

To this end, we provide qualitative and quantitative information about attentive neural networks in legal text processing. We propose different approaches to utilize the characteristics of legal text and supplementary knowledge to improve not only the performance of these models but also their explainability. Besides, we propose methods to customize the attentive architecture in neural networks for better designs. With the detailed explanation about different levels of intervention in the attentive network to inject expert knowledge, this dissertation could also be a good technical reference document for people who may concern.

As a collection of works toward improving attentive neural networks in legal text processing, the details about sub-contributions of this dissertation are listed in Table 1.1. The attentive models are investigated and proposed with our improvements from Section 3.4 to Section 5.4.

| Section | Contributions |
|---|---|
| 3.2 | *LVC*, *LECA* metrics, and a visualization method to understand about distribution of legal word vectors in the different legal embeddings. The measurement and visualization results suggest the problem of *domain difference*s in pretrained language models. |
| 3.3 | *Lawfulness classification*, a new way to consider the problem of textual-entailment-based question-answering for legal text, can boost the performance of the same model by increasing the data amount. The experimental results indicate the existence of the *lacking of data* problem in deep learning approaches for some legal tasks in this domain. |
| 3.4 | *Attentive CNN*, a simple convolutional neural network with attention mechanism. This model overperforms a robust pretrained language model (*XLM-RoBERTa*), which suggests the existence of lengthy input problem in the legal domain. |
| 4.2 | *BERTLaw*, a model with the vocabulary and the model's weights are constructed and trained from scratch using a large legal corpus. This model achieves the state-of-the-art result in COLIEE 2020's Task 4. |
| 4.3 | *NFSP* and *NMSP*, novel pretraining tasks based on the characteristic of translation alignment in legal data. Our model pretrained with *NFSP* task achieves state-of-the-art result in COLIEE 2021's Task 5. |
| 4.4 | *Paraformer*, a novel pretrained language model that could handle much longer input text than the base model. This characteristic is important to obtain good predictions for lengthy legal inputs. |
| 5.2 | *HYDRA*, an architecture-friendly and extensible method to improve the effectiveness of the transformer-based language models by pretraining and appending new knowledge-guided heads to their architecture. |
| 5.3 | *TRE* framework, a logic-structure knowledge injection approach for pretrained Transformer models. Our experiments indicate that a suitable strategy to inject legal logical knowledge can boost the performance of deep legal models. |
| 5.4 | *BART2S*, a regulated generator-discriminator generative framework for generating terms of services automatically using the generative models. We also propose a novel tuning process to adjust the fairness of the generated content based on the knowledge of fairness learned by the discriminator. |

Table 1.1: Sub-contributions of the dissertation.

The impact of this research may contribute to both scientific and practical meaning. The dissertation provides the whole picture of deep learning in legal text processing and related aspect in its content. In addition, embedding methods, training tasks, and architectural designs, which are the most important factors of every deep learning model, will be presented in this dissertation. From a practical viewpoint, the outcome of this research may contribute to bringing the most advanced techniques in deep learning to the legal domain. This document can be useful for researchers who seek for explainability of the deep learning model in the legal domain but not only using it as a black box. Explainability is a prerequisite for the deep legal system to be approved to operate in the real life.

## 1.4 Dissertation Outline

This dissertation is conducted with the purpose of analyzing and improving current state-of-the-art techniques in legal document processing using deep learning models. Firstly, we analyze different aspects of applying end-to-end deep learning models to a legal processing problem. By doing so, we obtain a clear insight to design effective models for each particular condition. Secondly, we propose novel ways to pretrain language models in the legal domain to improve their performance. Thirdly, we design approaches in using expert knowledge to support the models to have better learning and prediction in the legal domain.

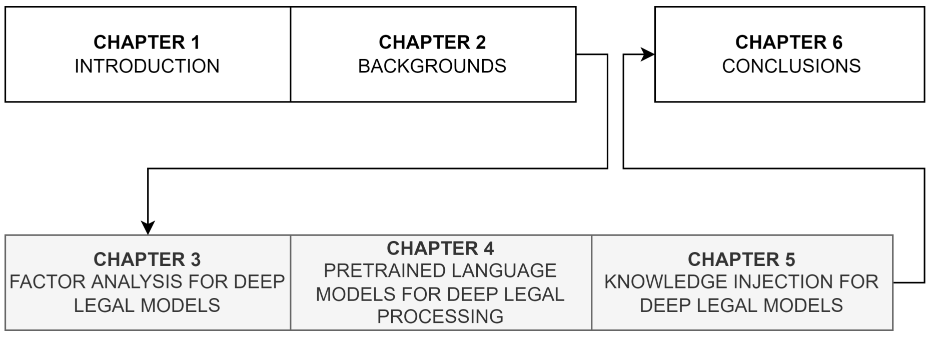

CHAPTER 1
INTRODUCTION
CHAPTER 2
BACKGROUNDS
CHAPTER 6
CONCLUSIONS
CHAPTER 3
FACTOR ANALYSIS FOR DEEP LEGAL MODELS
CHAPTER 4
PRETRAINED LANGUAGE MODELS FOR DEEP LEGAL PROCESSING
CHAPTER 5
KNOWLEDGE INJECTION FOR DEEP LEGAL MODELS

Figure 1.2: Outline of the dissertation.

The outline of the dissertation is presented in Figure 1.2. Firstly, we want to confirm the ability of deep learning models in performing a legal task, which often requires expert knowledge. We analyze the impact of data representation, data amount and the architecture of deep learning models. This confirmation is the first step in exploring the knowledge about deep legal processing. After that, we discover further about which characteristics of legal data can be utilized to pretrain strong legal language models, a family of multi-head attentive networks achieving many good results in natural language processing recently. The legal embedding, legal multilingual capacity and legal structure representation are addressed when answering this question. Lastly, we investigate the possibility of injecting knowledge into the neural networks to gain the performance and explainability of the models in this domain. Linguistic knowledge, legal knowledge and self-learned knowledge are investigated to answer this question.

Before answering the research questions, we dedicate Chapter 1 to introduce the research goals, the challenges as well as the motivations when we conduct this research and Chapter 2 to present the basic knowledge of deep learning, attention mechanism, and multi-head attentive models [62]. These technologies are highly influential when this dissertation was written. This knowledge not only provides the foundation for the reader to access the next chapters but also contributes to clarifying the context of the research. It is possible that these technologies will become obsolete and superseded in the future. However, the philosophy and the methodology of the dissertation still retain their reference values. Besides, we also present the characteristics of legal documents, the difference of legal documents from the daily text, challenges, and advantages when processing legal documents by deep learning.

The first research question is answered in Chapter 3. We examine in detail the factors that can affect deep learning models such as data representation, data amount, and model architecture. During our research on deep learning architectures, we found very simple architectures like SCNN [44], which have a small number of parameters that can still outperform other models. Interestingly, we also found the simple combination of CNN [35] architecture and attention mechanism [33] can give better results than bulky models in some specific cases. This chapter will answer the question of under what conditions can the end-to-end model perform well in a legal text processing task.

The next research question is answered in Chapter 4. In recent years, language models have been a powerful approach in deep learning. Pretrained with a huge amount of data, these models are capable of understanding the language and performing excellently on tasks in benchmark data. Models such as BERT [23], GPT-3 [13], and BART [37] make for a breakthrough in NLP compared to traditional NLP methods. These models take advantage of the idea of transfer learning, learning one task can improve the results of another. Many studies show that combining and interleaving tasks can improve the efficiency of the model. In our research, we present novel ways to pretrain language models. In the legal domain, our proposed model like BERTLaw [48], ParaLaw [46] proved their effectiveness in the COLIEE 2020 and COLIEE 2021 competitions with the standard data set provided by the organizers. With an end-to-end model, if garbage in, then garbage out, so having an appropriate training method is important to build a quality deep learning model.

The final research question is answered in Chapter 5. Besides the traditional training and pretrain-finetune paradigms, there exists the third approach, knowledge injection [47]. This approach is the use of expert knowledge to support learning models and decision-making. Instead of feeding the model with data so that it learns relationships on its own, we can directly feed expert knowledge into the model in the form of signals. This method helps to solve the problem of sparse, noisy data and leverages expert knowledge in training deep learning models. This expert knowledge can be in the form of linguistic features or semantic features. Through our experiments, we demonstrate that injecting this expert knowledge into the neural network will increase the performance of the model. Besides, this approach also helps to increase the accountability and debuggability in deep learning models.

The ultimate goal of the dissertation is to present our work on the road to improving attentive neural networks in legal text processing. The contents in Chapters 3 and 4 are our results and observations from our COLIEE participation. Chapter 5 presents preliminary investigations in an attempt to enhance the explainability of the attentive neural networks, which are considered black boxes. Although this work is done with meticulousness, there may be blind spots in the experiments and bias in the interpretation of the results. Therefore, in each work, we do not only report the performance as a single number but have a deeper analysis of the experimental results. At the end of each chapter, we summarize the main points of that chapter and the related discussions. Our final discussions and conclusions are presented in Chapter 6. This chapter enables the readers to understand our contributions as a coherent work towards improving attentive models in legal text processing. Last but not least, we outline future directions that can widen the scope and elevate this research to real-world applications.

# Chapter 2

# Backgrounds

## 2.1 Premiliary about Deep learning

### 2.1.1 Linear Neural Networks

A few years ago, deep learning is a blue-sky discipline, not many researchers and engineers are interested in it because the hardware condition does not allow them to build anything meaningful with this approach. However, recently, deep learning has become the most powerful and well-known technology in many different fields of AI, computer vision and natural language processing. Deep learning is also the backbone technology in this dissertation containing multiple sub-studies in applying deep learning to legal text processing. For that reason, we dedicate this section to the preliminary of deep learning.

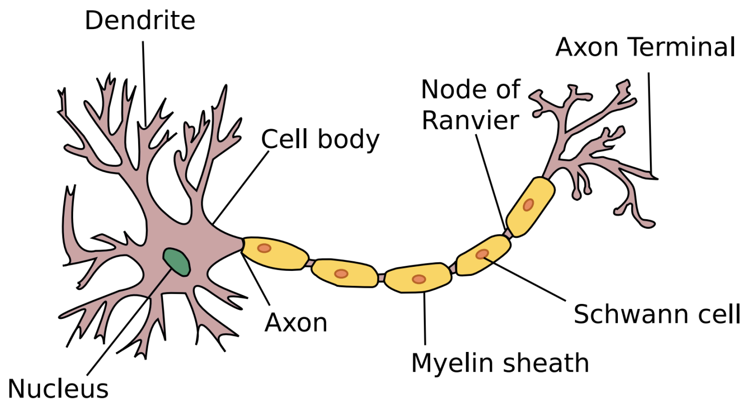


Figure 2.1: Structure of a real neuron.

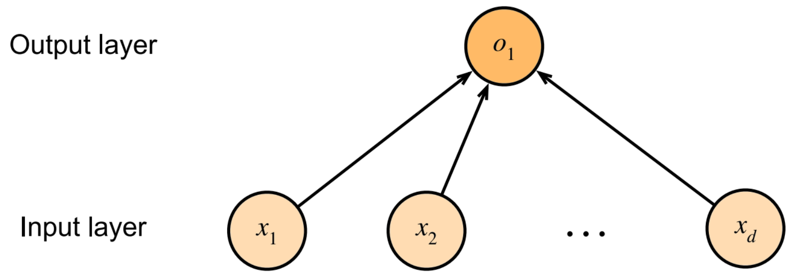


Figure 2.2: Demonstration of a linear regression model.

Deep learning is a technique that uses an artificial neural network, which is an architecture inspired by the real nervous system of animals. Figure 2.1 demonstrates the structure of a real neuron and Figure 2.2 shows a linear regression model, which is the simplest imitation of this computational unit. One single neuron receives signals from the others as inputs and computes its own signal. Satisfying certain conditions, this signal is passed to another neuron.

With $x_1$, $x_2$, ..., $x_d$ be the input signal, $w_1$, $w_2$, ..., $w_d$, $b$ be the parameters of this model, the prediction $\hat{y}$ value can be calculated as Equation 2.1.

$$\hat{y} = w_1x_1 + \ldots + w_dx_d + b \tag{2.1}$$

In another representation, let $\mathbf{x}$ be the input vector, $\mathbf{w}$ be the parameter vector, we have:

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b \tag{2.2}$$

Training the model in a dataset contain $n$ sample, we need to minimize the loss value given by the Equation 2.3 with the optimal parameters given by the Equation 2.4:

$$L(\mathbf{w}, b) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}\left(\mathbf{w}^\top \mathbf{x}^{(i)} + b - y^{(i)}\right)^2 \tag{2.3}$$

$$\mathbf{w}^*, b^* = \underset{\mathbf{w},b}{\operatorname{argmin}} L(\mathbf{w}, b) \tag{2.4}$$

As the simplest form of neuron imitation, linear regression can only predict points on a straight line. For nonlinear functions, this model cannot be fit, so its application is limited. To simulate the world, we need more complex functions than straight-line equations.

### 2.1.2 Multilayer Perceptrons

Apparently, the linear regression model can not perform well on classification problems. When we add more output nodes into the network, we obtain a softmax regression model. This model map a vector into a probability distribution, which can be used for a classification problem. The Figure 2.3 demonstrates a softmax regression network. The prediction value $\hat{\mathbf{y}}$ is calculated from the output $o_i$ as in the Equation 2.5

$$\hat{\mathbf{y}} = \mathrm{softmax}(\mathbf{o}) \text{ where } \quad \hat{y}_j = \frac{\exp(o_j)}{\sum_k \exp(o_k)} \tag{2.5}$$

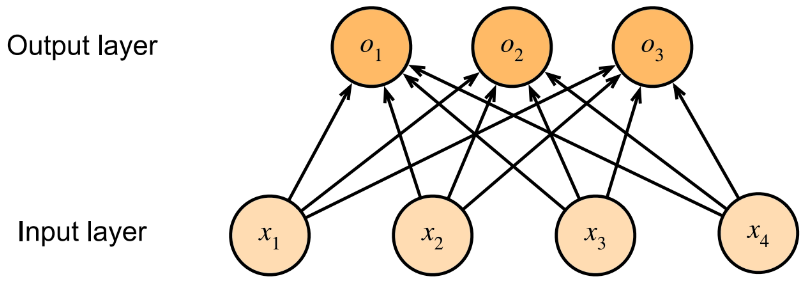


Figure 2.3: Demonstration of a softmax regression model.

The first architecture which is truly considered a deep learning model is multilayers perceptron (MLP). As demonstrated in Figure 2.4, this model contains at least one hidden layer between the input layer and the softmax layer. With this design, the model can learn both the representations in the hidden layers and the mapping function from the hidden signal to the output. It is proved that the multilayer perceptrons can approximate any function with arbitrary accuracy [29].

### 2.1.3 Convolutional Neural Networks

Although the multilayer perceptrons can approximate any function, this architecture requires many parameters when the size of input increases. Over more, the more parameters, the higher chance the model suffers overfitting issue. Convolutional neural network (CNN) is invented to constrain the network's parameters to reduce the number of them and avoid overfitting.

Figure 2.5 and 2.6 demonstrate the two main functions in a Convolutional Neural Network (*i.e.*, convolution function and pooling function). Applying these functions to the inputs, the model can produce cross-correlation features in the next layer. As a result, the number of parameters in the next layer is reduced and

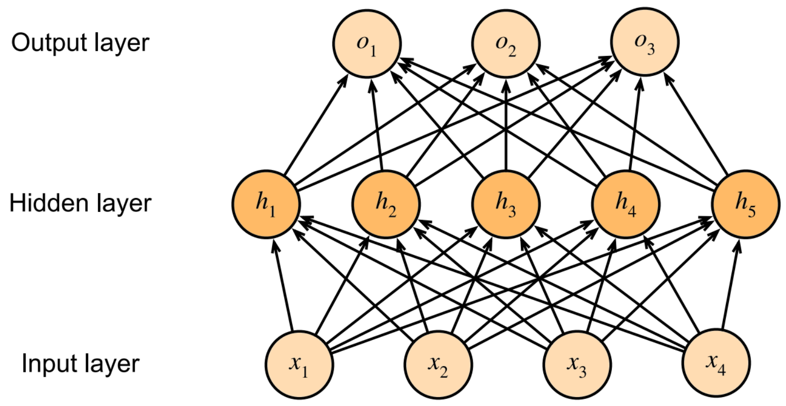


Figure 2.4: Demonstration of Multilayer Perceptrons.

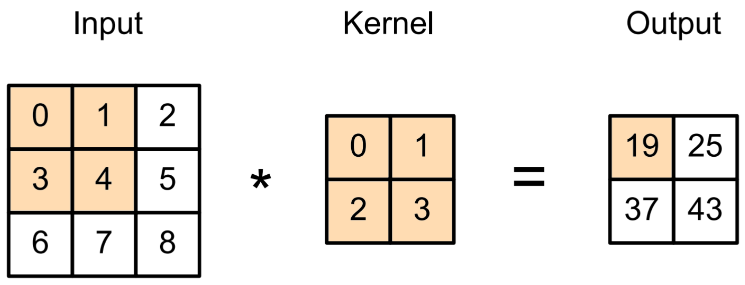


Figure 2.5: The convolution function in a CNN.

the model needs to extract the most important feature to optimize the loss value of the whole network.

### 2.1.4 Recurrent Neural Networks

The real-world data is often in the sequential format (*e.g.* documents, movies, stock market, events, conversations). As a result, it is required to have an appropriate architecture to model the sequence of data. Intuitively, the information from the previous signal in the sequence is necessary to predict the consequent signal.

Figure 2.7 shows a simple diagram of the recurrent neural network architecture. The general idea of this architecture is that a hidden state at time step $t$ is calculated from both the input and the hidden state at the previous timestep $t-1$ as in Equation 2.6. From the diagram, we can see that the computational route in a recurrent neural network is longer than in a feed-forward model. This characteristic may lead to a technical problem namely *vanishing gradient*. This problem is

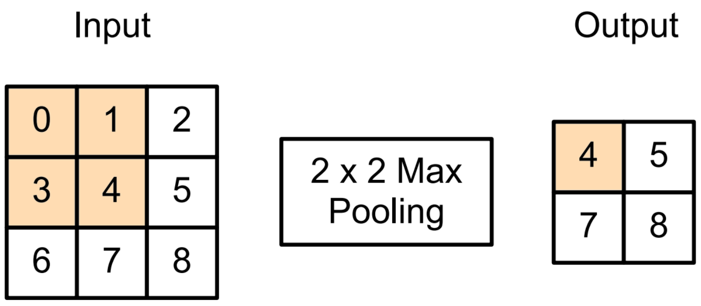


Figure 2.6: The pooling function in a CNN.

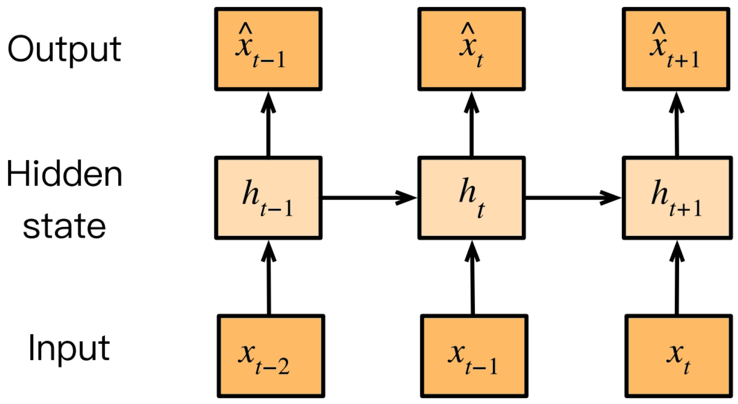


Figure 2.7: A simple diagram of the recurrent neural network.

solved by modern architectures with gating mechanisms.

$$h_t = f\left(x_t, h_{t-1}\right) \tag{2.6}$$

The recurrent neural network can also be designed in a bi-directional paradigm. As demonstrated in Figure 2.8, the input of a hidden node comes from both directions. This paradigm enables each node to access more information from the neighbor nodes in the sequence.

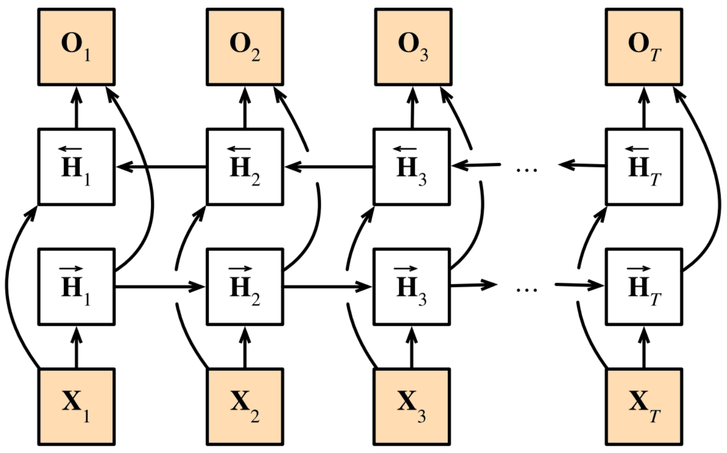


Figure 2.8: A recurrent neural network in a bi-directional paradigm.

## 2.2 Premiliary about Attentive Models

### 2.2.1 Attention Mechanism

In real life, information is abundant but meaningful information is not that abundant. In information processing, we do not create new information but extract meaningful information from the original one. Information often comes with noises, which is meaningless signals in the environment. The definition of noise depends on the specific task we conduct and the aspect we want to learn in the data. Attention is the mechanism that helps a model (*i.e.*, human model or machine model) to focus on the important parts of information.

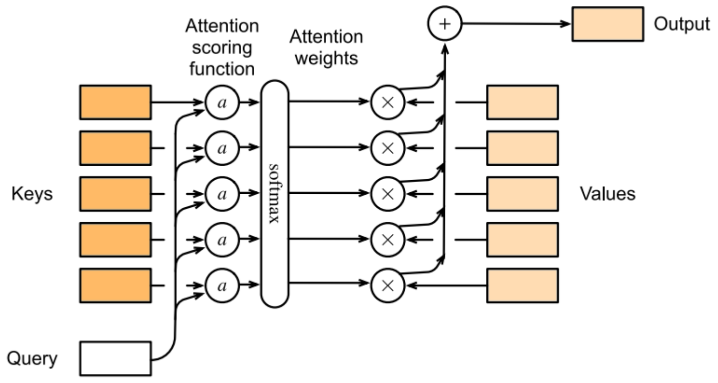


Figure 2.9: The attention mechanism is essentially a weighted sum of the values.

Figure 2.9 shows a diagram of a neural network using attention mechanism. There are three important components of this paradigm which are *query*, *keys* and *values*. *Query* is the signal for the model to know which option in the list of *(keys, values)* it should pay attention to. Let **q** be the query vector, $(\mathbf{k_i}, \mathbf{v_i})$ be the $i^{th}$ key, value pair in the candidate lists, function f to calculate the attentive output follows the Equation 2.7.

$$f\left(\mathbf{q},\left(\mathbf{k}_1,\mathbf{v}_1\right),\ldots,\left(\mathbf{k}_n,\mathbf{v}_n\right)\right)=\sum_{i=1}^{n}\alpha\left(\mathbf{q},\mathbf{k}_i\right)\mathbf{v}_i \tag{2.7}$$

In which $\alpha\left(\mathbf{q},\mathbf{k}_i\right)$ is calculated by applying a softmax function on the values returned by function $a$ calculating the alignment score between a query vector and a key vector as in Equation 2.8.

$$\alpha\left(\mathbf{q},\mathbf{k}_i\right)=\operatorname{softmax}\left(a\left(\mathbf{q},\mathbf{k}_i\right)\right)=\frac{\exp\left(a\left(\mathbf{q},\mathbf{k}_i\right)\right)}{\sum_{j=1}^{m}\exp\left(a\left(\mathbf{q},\mathbf{k}_j\right)\right)} \tag{2.8}$$

### 2.2.2 Self-attention

Self-attention is the featured mechanism of the Transformer-based architecture. This computation allows the model to integrate the signal from different positions in a sequence to obtain better representations without a recurrent design. This architecture is the base for a series of novel state-of-the-art pretrained models in many NLP tasks.

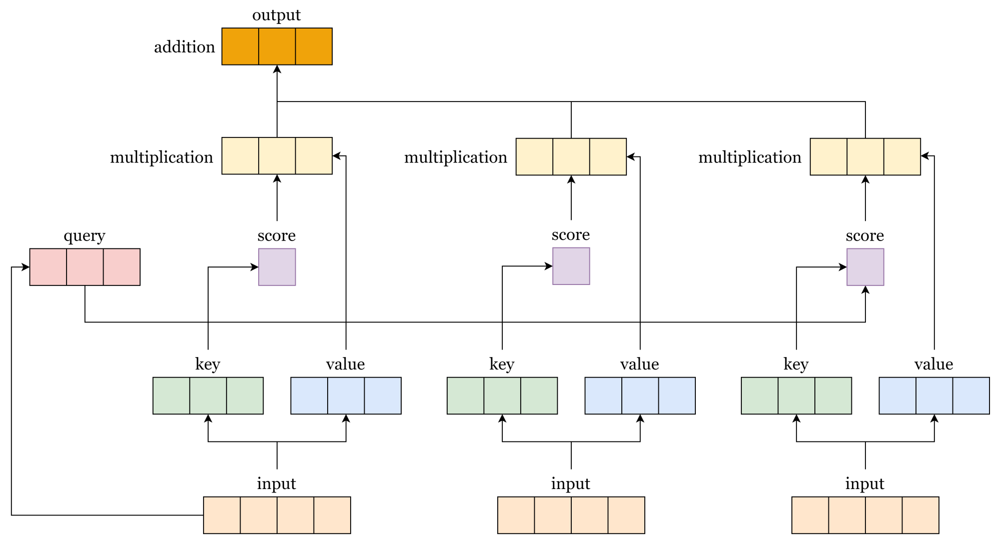


Figure 2.10: An example of self-attention computation.

Figure 2.10 demonstrates an example of self-attention computation. In this example, we are calculating the new representation for the first input vector. First, the query, key and value vectors are extracted from the inputs vector by corresponding weight matrices. After that, we apply the computation as described in Section 2.2.1 on the query vector of the first input and the (key, value) pairs of all the inputs to get the result. The process is repeated for the whole sequence and the new representations contain signals from all of the different positions in the original sequence with different weights.

### 2.2.3 Multi-Head Attention

Multi-head attention is another important idea in Transformer-based architectures. This paradigm allows the model to have multiple viewpoints in the way the alignments are constructed in an input sequence. As demonstrated in Figure 2.11, instead of using only one single attention module, this architecture calculates the attention representation in multiple subspaces (*i.e.*, multiple heads) then concatenate the signal from all heads to a vector. After that, this vector is transformed by a fully-connected (FC) layer to have the appropriate dimension.

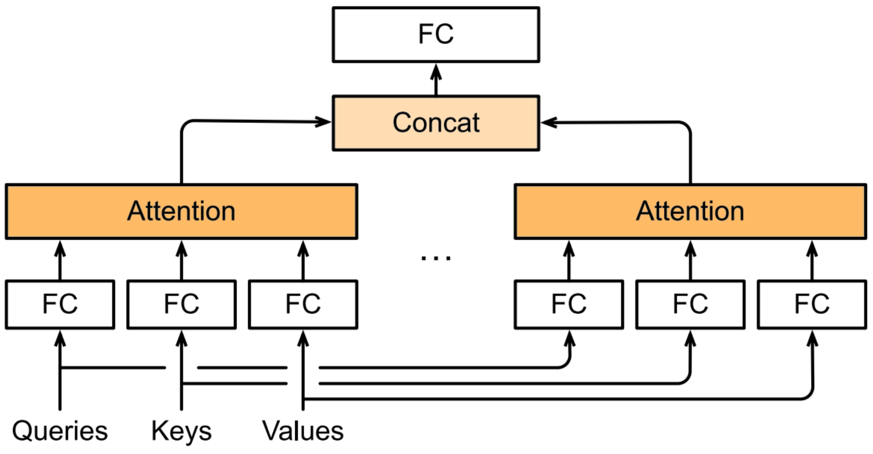


Figure 2.11: The representations of all heads are concatenated and transformed to get the final representation.

## 2.3 Characteristics of Legal Language

### 2.3.1 Legal Vocabulary

Language is the main communication method of human beings. For effective communication, the parties need to use the common language. However, in reality, every different group of people uses a different sublanguage. Young people may use language that older people do not understand and vice versa. To the extreme, everyone has unique thoughts so the vocabulary used by any two people can hardly be the same. Not being an exception, legal language is essentially a sublanguage dedicated to describing concepts in law with its own vocabulary and grammatical rules.

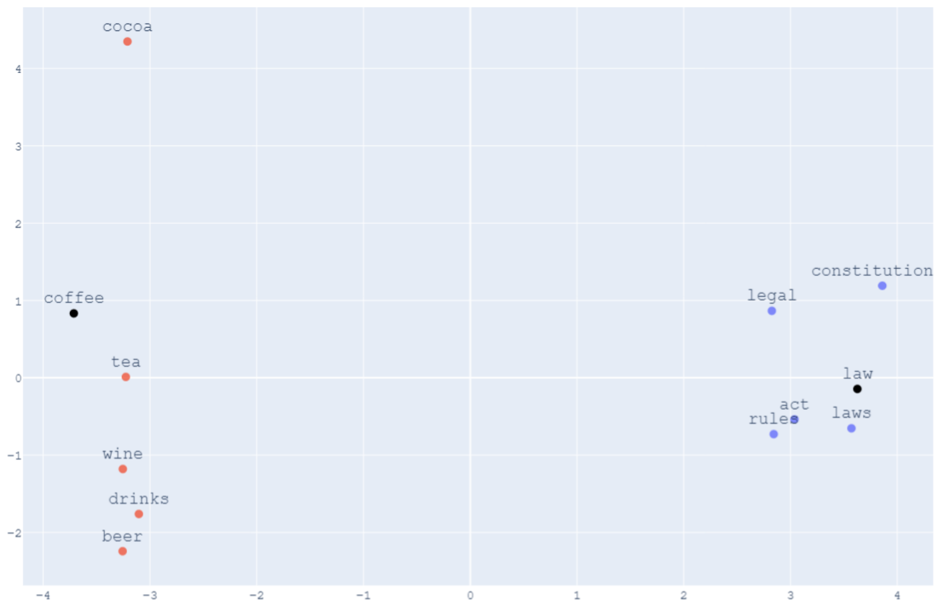


Figure 2.12: Distribution in the Word2Vec's vector space of the daily words and legal words.

Figure 2.12 visualizes the distribution in Word2Vec's vector space of some daily words and some legal words. The position distribution is very different between the two groups. This phenomenon may create a challenge for the state-of-the-art models in the general domain to perform well in the legal domain. Indeed, legal language is not easy for lay readers to use and understand. For example, not every English speaker understands the word *mutatis mutandis*, which is a common word in the legal domain. Hence, there need to be appropriate approaches to obtain good results in this domain.

### 2.3.2 Challenges In Legal Processing

Legal processing refers to automated extracting meaningful information from legal text on the aspect of interest. Because of its characteristics, there are challenges that need to be solved to obtain a good performance in this domain. These challenges come from the legal language, the complexity of the legal sentence and the scarcity of annotated data for some specific tasks in legal processing.

The following is an example of a legal article in the Japanese Civil Code:

> **Article 395**
>
> (1) A person that uses or profits from a building subject to a mortgage by virtue of a lease that cannot be duly asserted against the mortgagee, and that is set forth as follows (in the following paragraph referred to as "mortgaged building user") is not required to deliver that building to the purchaser thereof until six months have passed from the time when the purchaser purchased that building at auction:
>
> (i) a person that has been using or profiting from the building since prior to the commencement of auction procedures; or
>
> (ii) a person that is using or profiting from the building by virtue of a lease given after the commencement of auction procedures by the administrator of compulsory administration or execution against earnings from immovable collateral.
>
> (2) the provisions of the preceding paragraph do not apply if the purchaser, specifying a reasonable period of time, issues a notice to the mortgaged building user demanding payment of consideration for a period of one month or more with respect to the use of the building referred to in that paragraph that has been made after the time of purchase by the purchaser, and no payment is made within that reasonable period of time.

It can be seen that the above legal article contains long sentences, its format is very different from paragraphs in other types of documents like news, reports or novels. At this length, legal documents can make it difficult for robust models that have made many breakthroughs in the general domain.

Adding to the challenges of the inputs in legal processing, the outputs of legal automated systems need to meet higher quality conditions compared to ordinary systems. The accuracy of the system, as well as the quality of generated content, is crucial to bring these systems to real life.

## 2.4 Literature Review

Automated legal text processing is not a new research direction. This is a direction of research that dates back to the early years of computers. Approaches to various problems of automatic legal text processing evolve with the growth of the computing power of computers as well as the solidification of scientific and technological foundations. This is a difficult field, so each study can only solve part of the problems that exist in it. Even so, they are important foundations for driving the advancement of automated legal document processing.

Zhong *et al.* [71] demonstrate an overview of methods and applications in legal AI as in Figure 2.13. They divide the approaches of automated legal processing into two groups, symbol-based methods and embedding-based methods. The first group uses known knowledge to guide the system, the second group uses patterns that the model learns from the data to make decisions. The legal AI applications they list in their work are representative and do not cover the full range of real-life applications of the field.

Laying the first bricks in this field are rule-based systems [60, 14, 58, 56]. These systems take the form of expert systems or lexical matching systems. They make finding and retrieving information in the legal field easier. In addition, they can perform logical inferences in the law, as long as the data is described and represented appropriately for computers to understand. However, the disadvantage of these systems is the requirement of rules designed by humans. With simple sets of rules, these systems become rigid. More sophisticated rule sets require a large human effort.

With the development of the internet and the explosion of digital data, methods of using machine learning [6, 51], especially deep learning [20, 59, 33], are becoming more and more popular in NLP in general and automated legal processing in particular. Catering to this trend, a variety of datasets [68, 25, 19, 72, 36] and tasks [1, 15, 69, 9] are introduced. Besides, competitions [32, 52, 31, 43] are also organized to aggregate and search for optimal solutions to tasks within a given resource.

COLIEE [1] is an annual competition for automated legal document processing with high-quality data provided in both case law and statute law, which is the valuable resource for this dissertation. Case law and statute law are the two largest sources of law in the world, based on these two sources of law, social relations are adjusted in accordance with international practices and national laws. For case law, the cases that are heard first will be used as the basis for handling the following cases. In statute law, legal documents are the main basis for court decisions. The competition on building automated legal text processing systems

[1]https://sites.ualberta.ca/ rabelo/COLIEE2021/

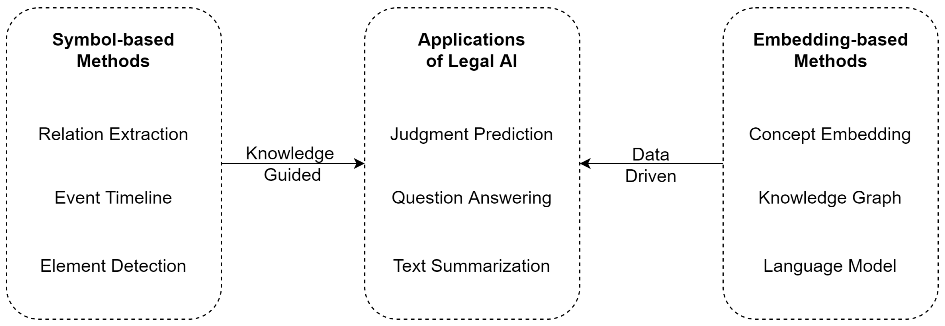


Figure 2.13: An overview of methods and applications in legal AI [71].

is a challenging and inspiring one. Until COLIEE 2020, there are in total of 4 tasks:

- Task 1 and Task 2 are case law retrieval and entailment problems.
- Task 3 and Task 4 are statute law retrieval and entailment problems.

COLIEE 2021 introduces one more task for legal question answering (Task 5).

# Chapter 3

# Factor Analysis for Deep Legal Models

## 3.1 Overview

This chapter introduces our three works in verifying what factors that affect the performance of deep learning models in general and deep legal models in particular. The finding in this chapter is the base for the problem formulation as well as the proposed approaches in the following chapters. The contribution of this chapter is as follows. First, we propose a novel way to assess different methods of data representation so that we can understand the correlation between data representation and performance in specific tasks. Second, we conduct experiments and study how data amount affects the learning ability of the models. Third, we verify the importance of understanding the data characteristic in designing appropriate architectures for a given task.

## 3.2 Impact of Data Representation

### 3.2.1 Introduction

Data representation is crucial to the construction of any effective system in computer science. Significantly contributing to the breakthroughs of the deep learning approach are the embedding techniques. Nowadays, there are two common families of embedding, which are word vectors (*e.g.* Word2Vec [39] and GloVe [50]) and contextual embeddings (*e.g.* BERT [23], GPT-2 [53], GPT-3 [13], BART [37]). We need the embeddings in deep learning models because the text needs to be converted into a numerical form to be processed by computers. Embedding emerges from the need to represent words in natural language in a numerical form that computers can process. These techniques evolved from the very simple approaches to the state-of-the-art methods we see today. In this section, we examine different embeddings and quantitatively compare their representational capacities for the legal domain.

Mentioning the classical methods, we have BOW, TF-IDF, which is merely based on the word occurrence to make the numeric representation. These approaches are good for lexical matching problems in the earlier days of NLP. However, in science and technology, we never stop expecting the systems to become more and more powerful and these methods become ineffective for problems requiring semantic or global context understanding. Bag-of-words vectors are obtained by spotting the word position in the dictionary while TF-IDF distributional vectors are based on the popularity of the words in the whole corpus.

The word embedding approaches are an improvement over counting-based data representation methods. Word2Vec is the algorithm based on prediction, the most well-known word embedding technique. The idea of this approach is that when trying to predict a word using its context, the model obtains the correlation information in the corpus. Continous Bag-of-words (CBOW) and Skip-gram are two methods in the Word2Vec approach. As a simple approach, Word2Vec's algorithms only consider the surrounding context and the target word, but not the full-text content. Dealing with that limit, GloVe uses neural embedding to parse co-occurrence matrices into more significant and weighted vectors. Trained on a large corpus, these embeddings provide interesting information from word vectors. For example, with well-trained vectors, we can find synonyms, antonyms, or even add and subtract meanings of words to get a new word in as the example in Figure 3.1.

Instead of using word vectors represented in the vector space, the Transformer [62] models use multi-head attention mechanism to identify the relationship between words in specific context usage, in different aspects. This method is therefore called contextual embedding. This method proves to be advantageous in cases where a word has multiple meanings. For example, the word *bank*, which could be a river's bank or financial bank. In word embedding, it can only be represented by a single vector, whereas, with contextual embedding, it can be interpreted in many aspects, depending on the surrounding words. In addition, some variants of this architecture that use subword tokenizer instead of word tokenizer have been shown to provide better performance in embedding representation. Figure 3.2 is the representation of word vectors of BERT in two-dimensional space for the two phrases *river bank* and *financial bank.* We can see that the word *bank* in two different phrases has different representation vectors. This allows the model to understand the semantics of words more flexibly, solving technical problems in the case of synonyms.

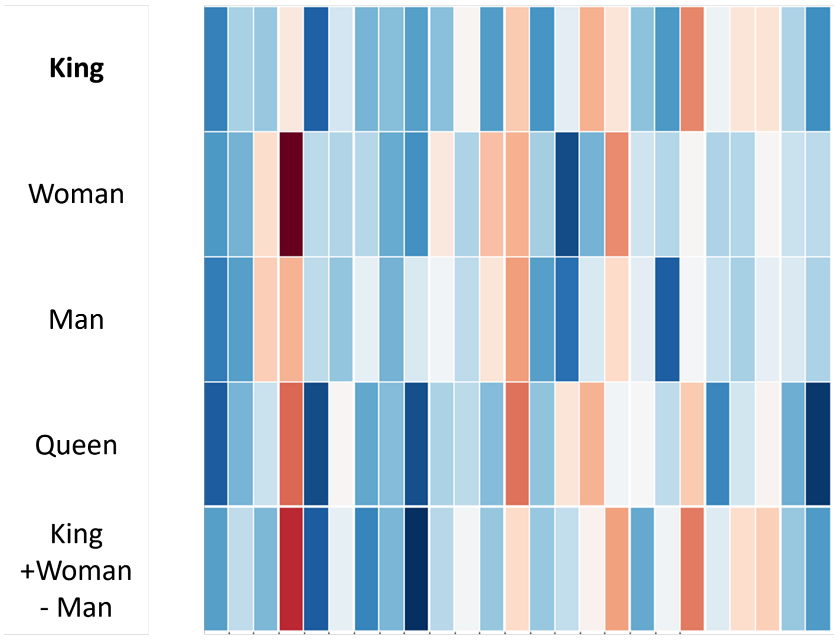


Figure 3.1: Visualizing in GloVe embedding space, we can create a vector that closely resembles the vector of *Queen* by adding and subtracting the vectors of *King*, *Woman* and *Man*.

The common mission of word embedding and contextual embedding is to convert words in the discrete representation into vectors in the continuous space. These representations are pretrained and can be used in downstream tasks. The information in these continuous representations directly affects the performance of the model. In this section, we study different pretrained approaches to represent the raw data into the embeddings' vector space.

### 3.2.2 Research Method

The quality of data representation has a direct influence on the performance of the models. If the input to the neural network contains all words that are outside the vocabulary, the model can not form a meaningful mapping from the input to the output in the training and prediction samples. In addition, the performance is also affected if the embedding layer is not well trained, or in other words, the words in the vocabulary are not represented by an exact relative position in the vector space. Hence, in this section, we propose quantitative metrics (Equations 3.2, 3.3) and introduce a visualization approach for embedding on the legal domain.

These metrics are based on an assumption that we have a set of standard legal terms $\mathscr{L}$ and a verification legal corpus $\mathscr{D}$ as a set of sentence $s_i$. The quality of

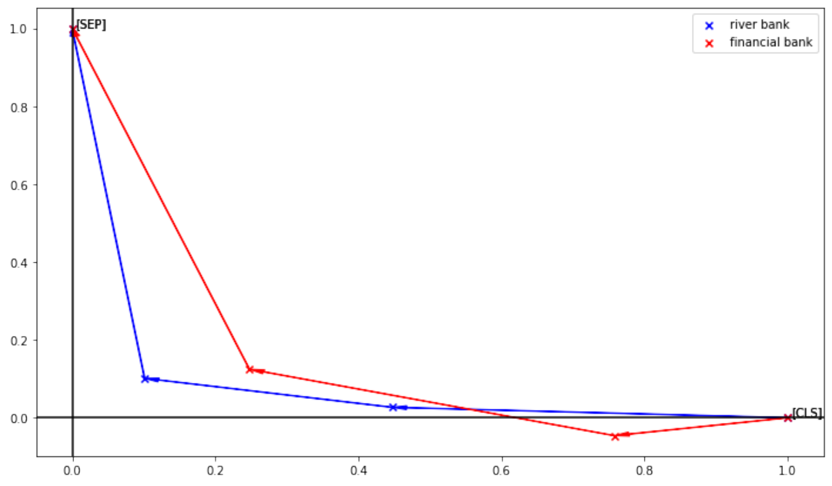


Figure 3.2: BERT represents the same word *bank* in two different phrases as different representation vectors.

embeddings is evaluated by the ability to map words in $\mathscr{D}$'s samples to the vector space and their relative positions.

Let $\mathscr{V}_E$ be the vocabulary of the embedding $E$, the first proposed metric $LVC_E$ (Legal Vocabulary Coverage) is calculated as follow:

$$\mathscr{C}_E = \mathscr{V}_E \cap \mathscr{L} \tag{3.1}$$

$$LVC_E = \frac{|\mathscr{C}_E|}{|\mathscr{L}|} \tag{3.2}$$

Although an important metric, $LVC$ is not good enough to evaluate $E$ because it does not take into account the position of the legal terms represented. Therefore, we propose $LECA_E$ (Legal Embedding Centroid-based Assessment) as a metric based on the position of the word vectors of the legal terms appearing in $\mathscr{V}$. Let $O_E$ be the centroid of points represented by $\mathscr{C}_E$'s vectors, $P_i^j$ be vector of the $j^{th}$ word in $s_i$, the $i^{th}$ sample in the corpus $\mathscr{D}$ and $d$ be a vectorial distance metric function, $LECA$ is calculated as follow:

$$LECA_E = \frac{1}{|\mathscr{D}|} \sum_{i=0}^{|\mathscr{D}|} \frac{1}{|s_i|} \sum_{j=0}^{|s_i|} d(P_i^j, O_E) \tag{3.3}$$

When applying $LVC$ and $LECA$ in practice, we need to use these two scales together to avoid misjudgments in extreme cases. In particular, with the $LVC$ being too small, the $LECA$ may not accurately reflect the effectiveness of the

Table 3.1: Experimental settings for word and contextual embeddings.

| **Systems** | **Vocabulary Size** | **Dimension** |
|---|---|---|
| FastText [40] | 30,000 | 300 |
| GloVe [50] | 30,000 | 300 |
| Law2Vec [20] | 30,000 | 200 |
| BERT [23] | 30,522 | 768 |
| LEGAL-BERT [17] | 30,522 | 768 |
| BERTLaw [48] | 32,000 | 768 |

embedding. In contrast, an embedding with large $LVC$ but its results on LECA are insignificant, the embedding fails to represent legal terms in its space.

In addition to quantitative measurement, visualization is an important aspect of the explanation. It helps us to focus on the important aspects of the phenomenon in order to understand it. Embedding visualization is essentially the representation of dimensional data that humans can perceive. For the problem posed in this section, the relative position of the legal term in the entire vocabulary is the most important information for understanding the nature of embedding in the legal domain. The core information that makes up the relative position of the vector representing terms in the corpus is the similarity between them. Hence, we propose to use the t-distributed stochastic neighbor embedding [61] technique to transform the vector space and visualize them in a smaller dimensional space.

### 3.2.3 Experiments

#### Experimental Settings

We conduct experiments to measure existing embedding systems on the metrics proposed in the previous section. The representatives of word embedding technique we choose are FastText [40], GloVe [50] and Law2Vec [20]. For contextual embedding, we use BERT [23], LEGAL-BERT [17] and BERTLaw [48]. Their configurations are all base, uncased. For the legal term set $\mathscr{L}$, we use 1,000 top terms provided by LexPredict[1]. For the legal corpus $\mathscr{D}$, we use 10,000 arbitrary legal sentences sampled from SigmaLaw dataset[2]. We use *cosine distance* as the distance metric function $d$.

As shown in Table 3.1 the vocabulary of contextual embeddings contains about 30K tokens. Towards a fair comparison, we also only consider the first 30K words in word embeddings. To extract word vectors from contextual embeddings, we

[1]https://www.lexpredict.com/

[2]https://osf.io/qvg8s/

Table 3.2: *LVC* and *LECA* score of word embeddings.

| **Systems** | **LVC** | **LECA** |
|---|---|---|
| GloVe [50] | 0.719 | 0.478 |
| FastText [40] | 0.725 | 0.467 |
| Law2Vec [20] | **0.770** | **0.434** |

Table 3.3: *LVC* and *LECA* score of contextual embeddings

| **Systems** | **LVC** | **LECA** |
|---|---|---|
| BERT [23] | 0.680 | 0.758 |
| LEGAL-BERT [17] | **0.737** | 0.618 |
| BERTLaw [48] | 0.689 | **0.612** |

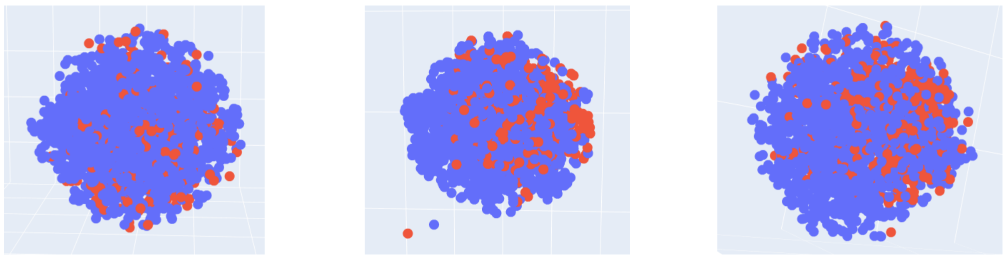
take the sum of the values of the last 4 hidden layers in the architecture. The dimensional size of contextual embeddings is the size of the hidden vector which is 768. In our experiment, we select the largest size of word embeddings provided by their authors (300D for GloVe, FastText and 200D for Law2Vec).

For word embedding, the vectors are taken directly from the pretrained data. For contextual embedding, vectors of legal terms in $\mathscr{L}$ are computed without context, whereas word vectors of samples in $\mathscr{D}$ are extracted from the computation on their contextual sentences. Since we could not find a reputable source for a set of legal subwords, we tokenize the input for contextual embedding on word level. This workaround may slightly degrade their actual performance. Therefore, in our experiment, we do not directly compare the results between word embedding and contextual embedding, but only compare embedding of the same type with each other.

### Experimental Result

Table 3.2 shows the results of word embeddings on the *LVC* and *LECA* metrics. Law2Vec achieves state-of-the-art results on both of these metrics. Comparatively, GloVe has a better *LVC* score and a worse *LECA* score than FastText. Models with lower *LVC* scores are more prone to out-of-vocab problems, and models with higher *LECA* scores are more likely to fall into local extremes or require more training iterations to converge. Therefore, for a problem in the legal domain, using Law2Vec with the word vector embedding method may lead to better results.

Table 3.3 shows the results of contextual embeddings. LEGAL-BERT scores state-of-the-art on *LVC* and BERTLaw scores state-of-the-art on *LECA*. From the experimental results, it can be seen that embeddings pretrained on the legal domain give better performance on the *LVC* and *LECA* metrics.



(a) GloVe (b) FastText (c) Law2Vec

Figure 3.3: Visualization of the word embeddings GloVe, FastText, and Law2Vec, respectively. The red points correspond to the legal terms in the set $\mathscr{L}$, the remaining points are blue.

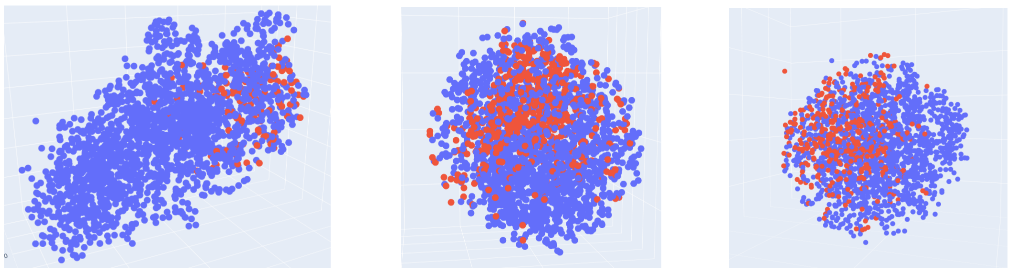
It can also be seen that contextual embeddings have lower *LVC* and higher *LECA* than word embeddings. This can be explained by the fact that these contextual embeddings are forced to calculate the vectors on the word level instead of the subword level as in their pretraining phase. Therefore, the cross-comparison between word embeddings and contextual embeddings may lead to a bias conclusion.

### Visualization

We visualize how the aforementioned embeddings depict legal and non-legal vectors in their space. We first filtered out the 2000 most common words in the vocabulary in each embedding for visualization purposes. We obtain the vector values for these terms and color the points respectively based on the *legal* and *nonlegal* labels. Then, we use the t-distributed stochastic neighbor embedding algorithm to find the appropriate 3-dimensional representation and display it.

Figure 3.3 shows the visualization of the word embedding space of GloVe, FastText, and Law2Vec. For all 3 embeddings, the positions of the points form a sphere in space. We can see that the legal points, although interspersed with other points, are concentrated in a particular region in space. Based on what is observed with the visualization, GloVe's embedding distinguishes legal points worse than FastText and Law2Vec.

Figure 3.4 demonstrates the visualization of the embedding space of BERT, LEGAL-BERT, and BERTLaw. These visualizations all show a pretty good distinction between legal points and the rest. The positions of the vectors in LEGAL-BERT and BERTLaw form a sphere in space, whereas the shape of the embedding distribution of BERT tends to be more distorted. This can be explained by the fact that this model is trained on a general domain with more diverse data domains than LEGAL-BERT or BERTLaw. Therefore, the survey in interdisciplinary data



(a) BERT (b) LEGAL-BERT (c) BERTLaw

Figure 3.4: Visualization of the contextual embeddings BERT, LEGAL-BERT, and BERTLaw, respectively. The red points correspond to the legal terms in the set $\mathscr{L}$, the remaining points are blue.

of legal and other domains is an interesting research direction in the future.

### 3.2.4 Discussions

This section analyzes the characteristics of different legal embedding techniques and proposes quantitative metrics and visualization for an explanation purpose. Through the experimental results and visualization, we have several conclusions. Firstly, the *LVC* and *LECA* metrics proposed in the article are suitable for the properties of the measured object and with the results reported in related articles. Secondly, embeddings that are pretrained with data in the legal domain tend to achieve higher results on these two metrics. Thirdly, a combination of quantitative and visualization scales can increase the explainability of embeddings in the legal domain.

In addition, we have some discussions about using the result in the section and the future directions. First, the values of the two proposed metrics would change for a different legal term set $\mathscr{L}$ and a legal corpus $\mathscr{D}$. Therefore, for a fair conclusion, they should be used for comparisons under the same conditions. Second, although limited by resources as described in Section 3.2.3, the formulas of these metrics are general. As a result, it is possible to compare word embeddings with contextual embeddings directly when the resource of the legal subword set is available. Third, visualization of the original BERT shows a difference in its word representation compared to other variants. Although legal variants are reported to be better on legal tasks, surveys of interdisciplinary data are also an interesting research direction.

## 3.3 Impact of Data Amount

### 3.3.1 Introduction

Data is the most important resource in machine learning, especially in deep learning. In this section, we qualitatively test this hypothesis by proposing a way to increase the number of training samples in a legal problem and compare the performance. The conclusions we draw from this section are important to prove the requirement of data amount to improve the performance in legal processing. Facing the sparse data problem, the same architecture may dramatically decrease the performance.

The idea of this research comes up when trying to solve the problem of legal question answering based on textual entailment in legal text in COLIEE 2019. Given a statement, we need to verify its lawfulness by finding its supportive article in the law as the following example:

- **Legal Statement:** An unborn child may not be given a gift on the donor's death.
- **Relevant Article:**
  Article 3 in Japanese Civil Code
  (1) The enjoyment of private rights shall commence at birth.
  (2) Unless otherwise provided by applicable laws, regulations or treaties, foreign nationals shall enjoy private rights.
- **Gold Answer:** True

Naturally, this problem is considered as a textual entailment problem with given related articles in the Japanese Civil Code. Textual entailment (TE) or Natural language inference (NLI) is a task in NLP. Given a termed text $t$ and hypothesis $h$, the models need to predict if $t$ entails $h$ ($t => h$) or not. The answers should be where *yes* or *no*.

Currently, there are several machine learning datasets for this problem. Samuel R. Bowman et al. [12] have introduced the Stanford Natural Language Inference (SNLI) Corpus that contains 570k pairs of written English sentences with three labels *entailment*, *contradiction*, and *neutral*. The Multi-Genre Natural Language Inference (MultiNLI) corpus [66] contains 433k written and spoken sentence pairs obtained by crowd-sourcing.

Compared to SNLI and MultiNLI, the total number of questions is extremely small. SNLI contains about 570,000 samples, MultiNLI contains 433,000 samples. This number is 716 in COLIEE 2019's dataset. In order to obtain good performance with this dataset, we propose a method to increase the number of training data for this task.

### 3.3.2 Research Method

As previously introduced, the goal of this task is to find the textual entailment between the statement and the articles to decide whether the statement is lawful or not. In this section, we propose three techniques to increase the number of training data which are *Problem Derivation*, *Sample Splitting* and *Data Augmentation*.

The given textual entailment problem can be stated as given a bar question $Q$ and all relevant articles $A$, the system needs to answer if $A$ entails $Q$ or not. Solving this problem, the model first needs to have an efficient way to encode each pair of the statement and relevant article, after that learn to abstract the patterns for entailed and non-entailed relationships between them. The biggest challenge in this problem is that the number of samples is limited in the samples given by the organizer.

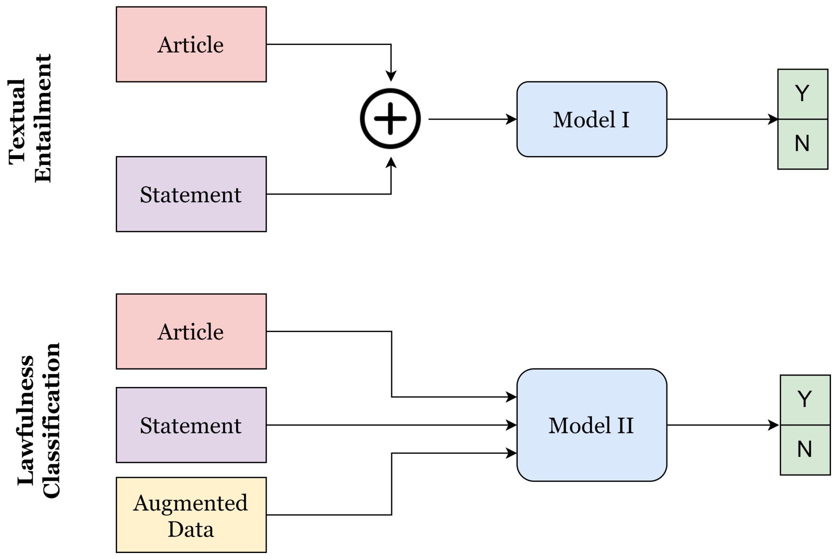


Figure 3.5: Problem derivation from textual entailment to lawfulness classification.

Figure 3.5 demonstate the original textual entailment problem and the new lawfulness classification problem. In the original problem, the model needs to process the pairs of articles and statements as given by the competition organizer. In the derived problem, the model treats each document as an independent input. This approach also allows us to create more augmented data for the learning process.

There are two parts in the data set, one contains articles in the Japanese Civil Code and the other contains statements in previous year competitions. For the lawfulness classification problem, the original data can be formatted as in the following rules:

- All articles in Japan Civil Code are lawful
- All statements in previous year datasets that are entailed by articles in Japan Civil Code are lawful
- All statements in previous year datasets that are not entailed by articles in Japan Civil Code are not lawful.

The average length of the articles is 43.72 words, with a standard deviation of 59.58 while these measurements for the questions are 19.61 and 40.71. The sentences in the Japanese Civil Code are averagely longer and more variant than the sentences in the questions in previous year datasets. From that observation, we split the articles to obtain more, shorter sentences with stabler length.

For each article, instead of adding the whole content into the data set, it is chunked into single statements. For example, article 329 (Order of Priority of General Statutory Liens) containing two statements could contribute 2 samples to the data set as *"(1) In cases where there is conflict among general statutory liens, [...] follow the order listed in each item of Article 306."* and *"(2) In cases where there is conflict between a general statutory lien [...] who received the benefit of the same."*. We assume that sub-articles in a lawful article are lawful. After applying the article chunking method, we have the average value and standard deviation in length of text in the civil code data set is 23.34 and 39.65.

Next, we use negation as the main data augmentation data method. Negation of examples in the data set is obtained by a set of rules listed in Table 3.4. After this phase, we obtained a data set, of which the total number of examples is 4,748. This data set is then fed into the deep neural network.

### 3.3.3 Experiments

To understand the impact of the amount of data on the model's performance, we use the same model (*i.e.*, Bi-LSTM) as proposed by Borges et al. [10] to run on the COLIEE dataset. In the textual entailment approach, the representation of the statement and the article are concatenated right before the final MLP layer. In the lawfulness classification approach, each input is fed individually to the network.

Table 3.5 shows the experimental results in accuracy. Bi-LSTM performs well on SNLI and MultiNLI with hundreds of thousand training samples. This model obtains 83.3% accuracy on SNLI and 67.5% on MultiNLI. In the same problem on COLIEE 2019's dataset, this model can only achieve 49.0%, which is even lower than a random guess (*i.e.*, 50%). With the problem of lawfulness classification, this model significantly improves its own performance by 8.1% (from 49.0% to 57.1%).

Table 3.4: Rules applied for negation statement generation.

| **Original Statement** | **Negation Statement Generation** |
|---|---|
| contains *not* | Remove *not* from original statement |
| contains *shall* | Replace *shall* with *shall not* |
| contains *should* | Replace *should* with *should not* |
| contains *may* | Replace *may* with *may not* |
| contains *must* | Replace *must* with *must not* |
| contains *is* | Replace *is* with *is not* |
| contains *are* | Replace *are* with *are not* |
| contains *will be* | Replace *will be* with *will not be* |
| contains *can* | Replace *can* with *cannot* |
| contains *cannot* | Replace *cannot* with *can* |
| contains *with* | Replace *with* with *without* |
| contains *without* | Replace *without* with *with* |
| contains *A* | Replace *A* with *No* |
| contains *An* | Replace *An* with *No* |

Table 3.5: Performance of BiLSTM in different datasets and approaches.

| **Dataset** | **Accuracy** |
|---|---|
| BiLSTM on SNLI [10] | 83.3% |
| BiLSTM on MultiNLI [10] | 67.5% |
| BiLSTM on COLIEE Textual Entailment | 49.0% |
| BiLSTM on COLIEE Lawfulness Classification | 57.1% |

From this result, we can see that the amount of data is an important feature to consider when applying deep learning techniques to a problem in general and a legal problem in particular. Data processing does not essentially produce more information. However, with the approach in this section, we can feed more information into the deep learning model, thereby making the technique applicable to the given problem. In addition, the trade-off of sentence number and sentence length in this case increases the advantage of deep learning models. This approach we proposed at COLIEE 2019, continues to be utilized and reaps good results at COLIEE 2020 and COLIEE 2021 with more robust deep learning models, which will be introduced in Chapter 4.

### 3.3.4 Discussions

Similar to mathematics, an automated legal processing problem can have many approaches. For each approach, we need to accept the trade-offs that come with it. For the deep learning approach, with the ability to self-synthesize patterns through examples, these models need a certain amount of data. Therefore, in this section, we propose a solution suitable for this characteristic. By deriving the problem from textual entailment to lawfulness classification, we achieve data superiority for deep learning models based on 3 aspects: data amount, sentence length and augmentation. Our hypothesis is that trading-off fewer long examples for more short examples might give the model advantage in this case.

From our experiments, we can see that with the same architecture, a lack of data may lead to a serious reduction of performance. Problem derivation, data reformation and data augmentation are good options in the domain with limited data. With the problem proposed in this section, we find that it is necessary to combine all three methods as a common approach to be effective. The reason could be that when we apply these methods to the law problem, we need to take into account the logical aspect of legal statements. For example, if chunking is not applied, we are negating only part of a multiple-item legal sentence, inverting the labels may produce incorrect results. This solution was proposed by us at COLIEE 2019, then applied and succeeded at COLIEE 2020 and COLIEE 2021. In addition, the observation in this section suggests we study the transfer learning techniques and pretrained models in legal text processing.

## 3.4 Impact of Model Architecture

### 3.4.1 Introduction

In addition to effective data representation and adequate data amount, model architecture is also a crucial factor that affects the performance in a deep legal problem. Although pretrained models have proven effective on a wide range of NLP tasks, with some specific data domains such as law, using the default settings of these models does not provide maximum performance. In this section, we find out that, in the specific domain of legal, a simple architecture designed with the understanding of the characteristic of data may surpass a bulky and powerful pretrained model.

This observation is obtained when we solve the problem of legal information retrieval (*i.e.*, given a query, the system needs to return the most appropriate legal articles related to the query). First, we construct a Vietnamese real-life dataset for legal question-answering. The dataset contains the real legal questions from legal consultant websites and a Vietnamese law corpus. Second, we experiment with different architectures and candidates on the dataset and observe the phenomenons. Finally, we propose a simple architecture solely empowered by a convolutional neural network and attention mechanism which obtain state-of-the-art performance.

The control model we use in this work is XLM-RoBERTa [21]. This model attests to the dominant of pretrained models with large scale in architecture as well as many pretraining tasks and datasets. Being trained with more than 2.5 terabytes of multilingual data, XLM-RoBERTa gains a significant performance in both high-resource and low-resource languages. However, in this work, the powerful model can not obtain a good result because its architecture is not designed to work with long text, an important characteristic of legal documents.

### 3.4.2 Reseach Method

The research in this section shows that, no matter how powerful, a model with incomplete information will make false predictions and is easily defeated by a simple model with enough information. The weakness of Transformer based models in this domain is that they treat inputs as continuos sequences and truncate samples which surpass theirs max length limitation. To design an effective architecture, we base on the key observation is that legal articles are mostly written in the form of a set of sentences. Hence, we design a simple attentive convolution neural network to capture the signal from every sentence and combine them together by a global attention mechanism.

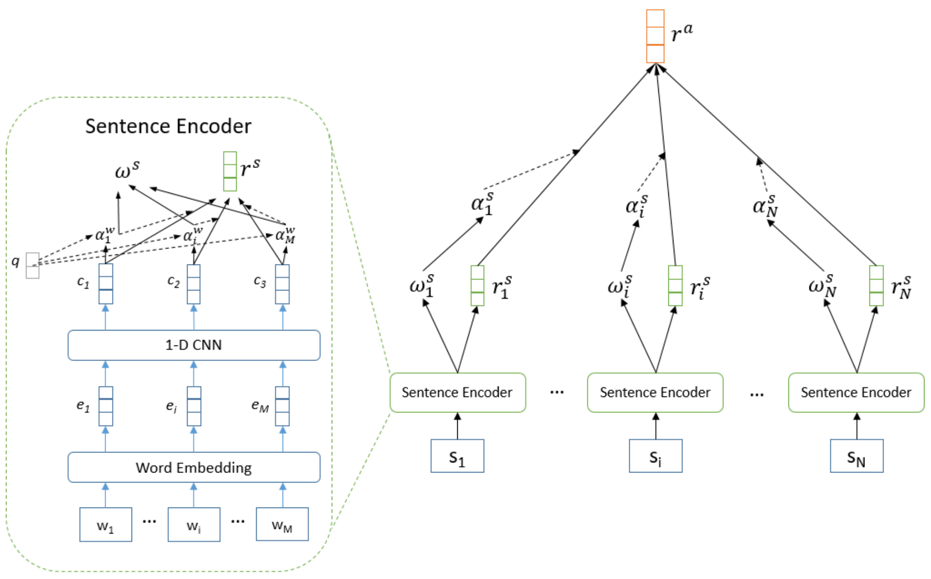


Figure 3.6: Attentive CNN architecture for long legal text processing.

Figure 3.6 demonstrates the architecture of the attentive convolution network. This architecture is constructed in a hierarchical paradigm. At the sentence level, each embedded word is processed by a convolutional layer. After that, a query-based attention mechanism is applied to the outputs to get the representation of the whole sentence. At the paragraph level, a sparse-max attention layer integrates the signal from the sentence representation to get the paragraph vector.

We train the proposed models using a negative sampling paradigm. First, the query is encoded by the sentence encoder and the article is encoded with the paragraph encoder. After that, we calculate the dot product between the two vectors as the similarity measurement. The dot product cares about both angle and magnitude, its value range is from negative to positive infinity. We normalize the probability article $i$ is related to a given query following Formula 3.4. We use the cross-entropy loss as the loss function for this approach.

$$p_i = \frac{\exp\left(\hat{y}_i^+\right)}{\exp\left(\hat{y}_i^+\right) + \sum_{j=1}^{K} \exp\left(\hat{y}_{i,j}^-\right)} \tag{3.4}$$

where $\hat{y}_i^+$ and $\hat{y}_{i,j}^-$ are the probabilities that article $i$ and article $j$, which belongs to the negative set of article $i$, is related to the query respectively.

Table 3.6: Value of parameters in Attentive CNN

| **Parameter** | **Value** |
|---|---|
| Size of Word Embedding layer | 512 |
| Number of CNN filter | 512 |
| Size of attention query vector | 200 |
| Dropout rate | 0.2 |

Table 3.7: Experimental Results on Vietnamese Legal Dataset

| **Systems** | **Precision** | **Recall** | **F2** |
|---|---|---|---|
| BM25 | 0.2395 | 0.1966 | 0.2006 |
| XLM-RoBERTa | 0.2395 | 0.1966 | 0.2006 |
| Attentive CNN | 0.5919 | 0.4660 | 0.4774 |

### 3.4.3 Experiments

The dataset contains two parts, which are a corpus of Vietnamese legal documents and a Vietnamese legal question-answering dataset containing queries coming along with their relevant articles. There are in total 8,586 documents with 117,545 articles in the legal document corpus, 5,922 queries coming along with their relevant articles in the question-answering dataset. We use 10% of the queries for testing and the remaining for the training and validation set. To rank the candidates, we combine the lexical score from BM25 and the deep learning model's score. Parameters of Attentive CNN are presented in Table 3.6.

Table 3.7 shows the experimental result on the Vietnamese legal dataset. XLM-RoBERTa contributes no improvement compared to the naive lexical approach using BM25. In contrast, Attentive CNN makes a big gap between the two control systems, overperforms them by a 0.2768 F2-score. This is a very interesting result because pretrained language models are often expected to yield better results than simple, unpretrained models. This result made us curious as to what makes the powerful model XLM-RoBERTa perform worse than Attentive CNN. Looking for the cause of this phenomenon, we discovered that the length of the article in our dataset is up to 253K characters, far beyond the encoding capacity of XLM-RoBERTa (514 tokens). This shows that lengthy content is an existential problem for legal documents and a suitable architecture for deep learning models is needed to solve this problem.

### 3.4.4 Discussions

In this section, by proposing and comparing a simple attentive convolutional neural network for a legal retrieval information problem, we obtain some important observations. First, besides data amount and data representation, model architecture is crucial to obtain a good performance in a deep legal problem. Second, using an attention mechanism to integrate the signals from elements of a long legal article can help the model to keep the full information without truncation. This observation is the basis for Paraformer, which is introduced in Chapter 4. Third, a simple architecture does not always perform worse than a complicated and bulky one especially when it can make better use of information from the data.

## 3.5 Summary of Chapter

In this chapter, we answer the question of what factors can impact an end-to-end model in deep legal processing. We examine in detail the factors that can affect deep learning models such as data representation, data amount, and model architecture. Better data in both quality and quantity is the requirement for a good performance in the legal domain. Besides, an appropriate architecture is also important for a model to access full information from the training data and perform better on a specific task. The observations from this chapter are the basis in exploring the knowledge about deep legal processing.

In the data representation aspect, we provide an insight into how embeddings perform across the legal domain. We propose two quantitative scales, LVC and LECA, for this purpose. In addition, our visualization method represents the position of word vectors in three-dimensional space. From there we can intuitively understand the existence of these vectors and their significance for the performance of the whole system. Not only based on speculation but also from our experimental results, we draw that pretrained embeddings in the law domain tend to represent better legal concepts than the embeddings pretrained with the general documents.

In terms of data amount, we experimentally prove that lack of data is a problem that directly affects the performance of deep learning models. With the same architecture, a deep learning model can give excellent or bad results depending on the amount of data it is trained on. Although this problem has been pointed out in many previous works, our conclusion once again confirms its existence in the legal domain. Within our scope, we offer specific solutions to increase data volume through problem derivation, data reformation and data augmentation.

In analyzing the architecture of the model, we prove that there is no free lunch for every problem. Through experimentation, we show that a simple architecture that properly models the nature of data can have better results than a bulky, pretrained architecture trained with a massive amount of data. This finding is the premise for us to design improvements to pretrained language models so that they can achieve the best performance on legal domains.

# Chapter 4

# Pretrained Language Models for Deep Legal Processing

## 4.1 Overview

This chapter dedicates to introducing our works related to pretrained language models in deep legal processing. There are three models mentioned in this chapter. The first model is BERTLaw, a language model trained on the legal corpus from scratch and finetuned for the question-answering task. This model achieves impressive state-of-the-art results in Task 4, COLIEE 2020. Second, we introduce Paralaw Nets, a family of language models pretrained by legal multilingual resources, in which the NMSP model becomes the best system in Task 5, COLIEE 2021. The third model is Paraformer, an attentive architecture leveraging the power of language models and succeeding in legal retrieval problems. This model is the extension of the model introduced in Section 3.4.

## 4.2 Legal Contextual Embedding

### 4.2.1 Introduction

As introduced in Chapter 2 law documents are written in a special sublanguage. This sublanguage is very different from the language we use in daily life. Legal English is used in drafting contracts, terms of service, regulations and other legal documents. As a result, a language model pretrained with data in the general domain may face trouble in tasks in the legal domain.

In this section, we introduce BERTLaw, a language model pretrained from scratch with a large legal corpus. Because of the superior legal vocabulary understanding, this model achieves impressive state-of-the-art results in Task 4, COL-

IEE 2020. This idea also attests to the observation in Chapter 3, with the same architecture, a better dataset could lead to better performance.

BERT [23] is one of the first pretrained language models based on Transformer achieving state-of-the-art results in a wide range of NLP tasks. The authors of this model propose two pretraining tasks which are masked language model (MLM) and next sentence prediction (NSP). In the MLM task, the model needs to make predictions to guess the masked words in a sentence and in the NSP task, the model needs to predict whether the two sentences are consecutive in a paragraph.

### 4.2.2 Research Method

We construct BERTLaw, a model pretrained with a large amount of legal text. A legal word stands alone can not convey any meaning. In a legal statement, there are both common words and legal words. As a result, we choose legal cases as our training data. In legal cases, everyday vocabulary and legal vocabulary appear together. This is important to get a good pretrained language model from scratch. We use 8.2 million sentences of American legal case data to pretrain our model.

Constructing the vocabulary of BERTLaw, we apply the method of SentencePieces [34]. A word can be split into multiple subwords for the most efficient representation. With subword representation, the problem of OOV could be mitigated. For example, suppose that *reconvention* does not appear in the vocabulary, it can be represented by 4 subwords as *rec*, *on*, *vent* and *ion*. Note that, in embedding, mapping a word into an index without appropriate weights, the mapping has little meaning. That's why we need to pretrain the model to learn the relationship between subwords in the vocabulary.

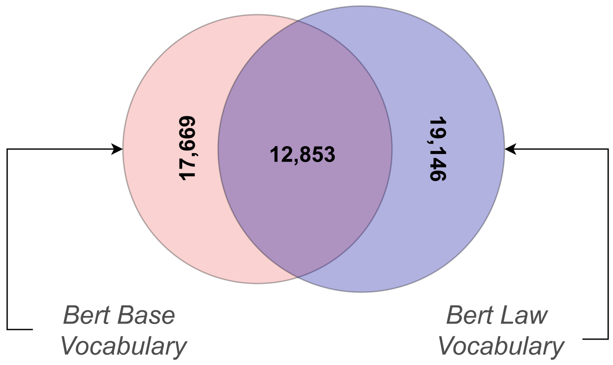


Figure 4.1: BERTLaw vocabulary compared to Google's BERT Base vocabulary.

Figure 4.1 demonstrates the comparison between the vocabulary of BERTLaw and Google's BERT Base vocabulary. The overlapped words are more than half of the vocabulary of each model. The vocabulary of BERTLaw is slightly bigger

Table 4.1: BERTLaw and BERT Base Performance.

| Model | Validation Accuracy | Test Accuracy |
|---|---|---|
| BERT Base | 0.7784 | 0.5625 |
| BERT Law | 0.8168 | 0.7232 |

than which of BERT Base. We pretrain our BERTLaw using Google's TPU on the corpus until the loss value on the MLM and NSP tasks stop reducing.

### 4.2.3 Experiments

We verify the effectiveness of our pretrained language model on the COLIEE 2020's task 4 dataset. This task is designed to verify the ability of the paralegals in answering the legal questions. Given a statement, the answer should be whether this statement is true or false. Our approach for this problem is the same as introduced in Section 3.3. The model is finetuned on the augmented data and needs to classify the lawfulness of an input.

After data augmentation, we have in total 5,000 samples, 90% for training and 10% for validation. The test set provided by the organizer contains 112 testing samples. Since we approach the task as a lawfulness classification problem, we do not need to deal with long input, the max length is set to 128.

We obtain the best configuration after 5 epochs of finetuning. Table 4.1 demonstrates the experimental results on the validation set and the official test set. Our BERTLaw outperforms Google's BERT Base by 4% on the validation set and 16% on the test set. This achievement brings us to the first position in the leaderboard of the competition in 2020[1].

Looking deeper into the vocabulary of the two models, we find interesting differences. Table 4.2 shows some examples that appear in BERT Law vocabulary, not in BERT Base vocabulary, which can impact the difference in performance in legal-related tasks. The subword *legal* does not appear in BERT Base vocabulary, which can make this model's vocabulary comprehension flawed in the legal domain. In addition, with the LVC and LECA measurements presented in Chapter 3 to evaluate embeddings in the legal domain, BERT Law is superior to BERT Base.

To better understand model behavior, we observe how they handle input queries. Here's how two models tokenize the same query "A contract of sales concluded by a minor may not be rescinded if it relates to daily life, even in cases the consent of the parental authority is not obtained":

[1]https://sites.ualberta.ca/ rabelo/COLIEE2020/task4_res.html

- **BERT Base:** a contract of sales concluded by a minor may not be **res ##cin ##ded** if it relates to daily life, even in cases the consent of the parental authority is not obtained
- **BERT Law:** a contract of sales **conclude ##d** by a minor may not be **rescind ##ed** if it **relate ##s** to daily life [UNK] even in cases the consent of the parental authority is not **obtain ##ed**

We bold the words that the models must use subwords to represent. In this example, BERT Base must use fewer subwords than BERT Law. Words like "concluded", "obtained", "related", BERT Law must separate the original word and the ending, BERT Base keeps the same words. However, with the word "rescinded", BERT Base must use 3 subwords to represent, while BERT Law separates it only into "rescind" and the "ed" suffix. "Rescind" is a legal term and the representation of BERT Law proves its legal embedding capacity.

### 4.2.4 Discussions

This section aims to verify that the issue of sublanguage in law happens with humans and also language models. We propose to construct the vocabulary and pretrain a language model from scratch using a large legal corpus *i.e.*, BERTLaw. Our contribution is not in suggesting a new method but in adapting an existing method in the general domain to the legal domain and analyzing the reasons for its success. The experimental result proves that the proposed approach is reasonable. Although BERTLaw is a single model pretrained in the legal domain, the sublanguage problem may occur in other fields. Therefore, building such pretrained language models is a suggestion to obtain the optimal performance for this kind of model.

Table 4.2: Examples that appear in BERT Law Vocabulary, not in BERT Base Vocabulary. Subwords start with ♯♯.

| **Token** | **Explanation** |
|---|---|
| ♯♯legal | Wordpiece in words containing "legal" (e.g. illegal, legally, legality, legalization) |
| contravention | An act which violates the law, a treaty or an agreement that the party has made. [11] |
| construe | To determine the meaning of the words of a written document, statute, or legal decision, based upon rules of legal interpretation as well as normal meanings. [11] |
| demurrer | An assertion by the defendant that although the facts alleged by the plaintiff in the complaint may be true, they do not entitle the plaintiff to prevail in the lawsuit. [11] |
| depose | To make a deposition; to give evidence in the shape of a deposition; to make statements that are written down and sworn to; to give testimony that is reduced to writing by a duly qualified officer and sworn to by the deponent. [11] |
| guardianship | The power or protective authority given by law, and imposed on an individual who is free and in the enjoyment of his rights, over one whose weakness on account of his age, renders him unable to protect himself. [11] |
| infringe | To transgress or exceed the limits of; violate: infringe a contract; infringe a patent. [41] |
| malfeasance | The commission of an act that is unequivocally illegal or completely wrongful. [11] |
| misdemeanor | Offenses lower than felonies and generally those punishable by fine, penalty, forfeiture, or imprisonment other than in a penitentiary. [41] |
| reimburse | To repay (money spent); refund. [41] |
| renounce | To give up a right; for example, an executor may renounce the right of administering the estate of the testator; a widow the right to administer to her intestate husband's estate. [41] |
| rescind | To declare a contract void—of no legal force or binding effect—from its inception and thereby restore the parties to the positions they would have occupied had no contract ever been made. [41] |
| rescission | The termination of a contract by mutual agreement or as a result of fraud or some legal defect. [41] |
| revoke | To invalidate or cause to no longer be in effect, as by voiding or canceling. [41] |
| tort | a civil wrong. Tortious liability arises from the breach of a duty fixed by law; this duty is towards persons generally and its breach is redressable by an action for unliquidated damages. [41] |
| tortious | Wrongful; conduct of such character as to subject the actor to civil liability under Tort Law. [41] |

## 4.3 Legal Multilingual Capacity

### 4.3.1 Introduction

In this section, we propose to use the multilingual information in the aligned translation pair as the pretraining resource for pretrained language models. The idea of this work starts from the observation that a translation of a sentence into another language could be used as information to reduce the ambiguity in that sentence. Over more, this information may help the language model to obtain better positions representing the words in the vector space.

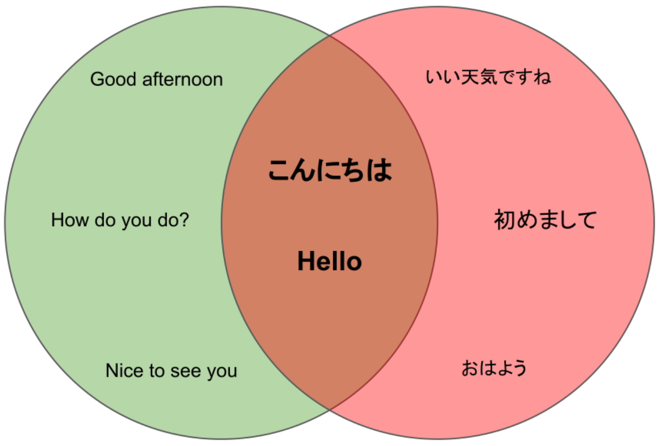


Figure 4.2: An example of a translation where a word can be translated by multiple candidates, depending on different contexts.

As can be seen Figure 4.2, there are multiple ways to translate こんにちは to English and multiple ways to translate "Hello" to Japanese. When these two words are aligned together, we know exactly that the speaker wants to use a word for the greeting purpose. This example presents to us that a good translation can be used to make the meaning clearer. Fortunately, there are good translations available in the legal domain.

### 4.3.2 Research Method

From the observation about the meaning of translation, we design two pretraining tasks to train multilingual pretrained language models. The first task is Next Foreign Sentence Prediction (NFSP), given a pair of sentences in different languages, the model needs to predict whether the two sentences are consecutive or not. As a simple example, we have a pair of sentences as "Hello! How are you?" and their

Table 4.3: Hyperparameters and performances in pretraining the models

| **Hyper Parameter / Performance** | **NFSP** | **NMSP** |
|---|---|---|
| Max Length | 512 | 512 |
| Batch Size | 16 | 16 |
| Number of Batches | 24,000 | 320,000 |
| Validation Accuracy | 94.4% | 88.0% |

translations “こんにちは。お元気ですか？”, examples can be constructed as follow:

- Hello. お元気ですか？
- こんにちは。How are you?

The second task is Neighbor Multilingual Sentence Prediction (NMSP). In this task, the model needs to predict not only whether the two multilingual sentences are consecutive but also which sentence is before, which sentence is after. The training data for this task accepts pairs of sentences in the same language. We apply a random negative sampling to construct the samples in which two sentences are not semantic consecutive.

To obtain these novel pretrained language models, we use the base architecture of BERT multilingual [23]. We reuse the vocabulary of the case-sensitive configurations. After that, we construct the dataset as described above with English-Japanese bilingual data crawled from the Japanese Law Translation website[2], we obtain 239,000 samples for NFSP and 718,000 samples for NMSP. We use 10% of the dataset as the validation set. We keep training the models until the performance stops to increase in the validation set. Table 4.3 shows the hyperparameters and the performances in pretraining the models.

### 4.3.3 Experiments

We finetune our multilingual language models for the question answering task in COLIEE 2021. The general approach is similar as introduced in Section 3.3. However, with the multilingual models, we can double our augmented data by using both English and Japanese datasets provided by the organizer. Besides the English negation rules in Table 3.4, we introduce the Japanese negation rules in Table 4.4. For contextual embeddings, the input text representation of the model depends on its entire weights and not just on the embedding layer as for word embeddings. For that reason, we can augment more training data in both languages even though the legal question answering task is not multilingual.

[2]https://www.japaneselawtranslation.go.jp

Table 4.4: Rules applied for negation statement generation in Japanese

| Original Statement | Negation Statement Generation |
|---|---|
| ません | ません → ます |
| できる | できる → できない |
| できない | できない → できる |
| した | した → しなかった |
| でない | でない → である |
| できた | できた → できなかった |
| させる | させる → させない |
| ている | ている → ていない |
| がない | がない → がある |
| ではない | ではない → ではある |
| ことがある | ことがある → ことがない |
| しなければならない | しなければならない → してはいけません |
| ならない | ならない → なる |

After augmenting the data, we obtain 7,000 samples. We use 700 samples for validation, the rest is for training. Because of the complexity in Japanese data, we first train 3 epochs with data augmented with the first three Japanese negation rules before training on the data augmented with the whole rules. On the validation test, NFSP achieves 71.0% accuracy while NMSP achieves 79.5%. We also train the vanilla BERT Multilingual in this task, the model only achieves 64.1% accuracy. This again confirms the usefulness of multilingual data for language models.

Table 4.5 shows the performance of different systems on the formal test set in COLIEE 2021. We are surprised that the NFSP overperforms NMSP on the test set and achieves state-of-the-art performance, nearly 4% higher than the second-best system. On the test set, BERT Multilingual performs not very well, its performance 0.4691 even lower than a random guess (*i.e.*, 0.5 accuracy).

### 4.3.4 Discussions

This section proposes and discusses using sentence-level cross-lingual information to pretrained language model. We introduce two pretraining tasks corresponding with two proposed models namely NFSP and NMSP. The models achieve impressive results in our validation set as well as the formal COLIEE 2021's test set. These results confirm the effectiveness of using translation as a training resource for language model pretraining. The finding in this section can also be applied to other domains in which good translations are available.

Table 4.5: Result of final runs on the test set

| Team | Run ID | Correct | Accuracy |
|---|---|---|---|
| | BaseLine | No 43/All 81 | 0.5309 |
| JNLP | NFSP | 49 | 0.6049 |
| UA | UA_parser | 46 | 0.5679 |
| JNLP | NMSP | 45 | 0.5556 |
| UA | UA_dl | 45 | 0.5556 |
| TR | TRDistillRoberta | 44 | 0.5432 |
| KIS | KIS_2 | 41 | 0.5062 |
| KIS | KIS_3 | 41 | 0.5062 |
| UA | UA_elmo | 40 | 0.4938 |
| JNLP | BERT Multilingual | 38 | 0.4691 |
| KIS | KIS_1 | 35 | 0.4321 |
| TR | TRGPT3Ada | 35 | 0.4321 |
| TR | TRGPT3Davinci | 35 | 0.4321 |

## 4.4 Legal Structural Representation

### 4.4.1 Introduction

In Section 3.4, we point out the limitation of the vanilla architecture of current Transformer based language models when working with legal data. Even so, language models have undeniable power when training data becomes scarce. In this section, we propose a novel architecture that combines the strength of both pretrained language models and the advantage of legal structural representation by using the global attention mechanism.

The length of legal documents is always a challenge for automated processing systems. A longer document means more signals need to be processed. More seriously, meaningful information may be in the later parts, which has been truncated due to exceeding the maximum length of the architecture. In a field that prioritizes correctness like law, solving this problem is a prerequisite for practical applications in the future.

The following example demonstrates how a model views an article exceeding its maximum length. The grey parts demonstrate the truncated content that the model does not process. The content could be trimmed at any position of the text depending on the maximum length configuration, which makes the information incomplete and incomprehensible.

**Article 330**

(1) In cases where there is conflict among special statutory liens with respect to the same movables, the order of priority shall follow the order listed below. In such cases, if there are two or more preservers with respect to the statutory liens for preservation of movables listed in item (ii), a new preserver shall prevail over previous preservers.

(i) Statutory liens for leases of immovable properties, lodging at hotels and transportation;

(ii) Statutory liens for the preservation of movables; and

(iii) Statutory liens for the sale of movables, the supply of seed or fertilizer, agricultural labor and industrial labor.

(2) In the cases provided for in the preceding paragraph, if a holder of a statutory lien ranked first knew at the time he/she acquired that claim of the existence of a holder of a statutory lien of the second or third rank, he/she cannot exercise his/her rights against those persons. The same shall likewise apply against persons who preserved Things on behalf of the holder of a statutory lien of the first rank.

(3) Regarding fruits, the first rank shall belong to persons who engage in agricultural labor, the second rand shall belong to persons who supply seed or fertilizer, and the third rank shall belong to lessors of land.

In a specific legal retrieval case, if the query is related to the order of priority of statutory liens over fruit, the model may not consider this article as a good candidate. The reason is simply that it does not have the information of fruit in the first part of the content.

### 4.4.2 Research Method

To overcome the problem of long legal documents and still take advantage of the pretrained language model, we propose an architecture namely Paraformer, which uses a transformer network to encode each sentence and integrate the signals to obtain the representation for the paragraph level. With this architecture, we can input the whole long article into the model as long as the physical memory condition is adequate.

Figure 4.3 demonstrates the architecture of the sentence encoder, which encodes an input sentence into a vector representation. Suppose that the language model contains $L$ layers, the input has $M$ tokens, after the layer $L-1^{th}$, we have the output $T = (t_1, t_2, ..., t_M)$. At the last layer, $T$ is multiplied by the attention matrices $Q$, $K$, $V$ to get the corresponding attention values. These attention values

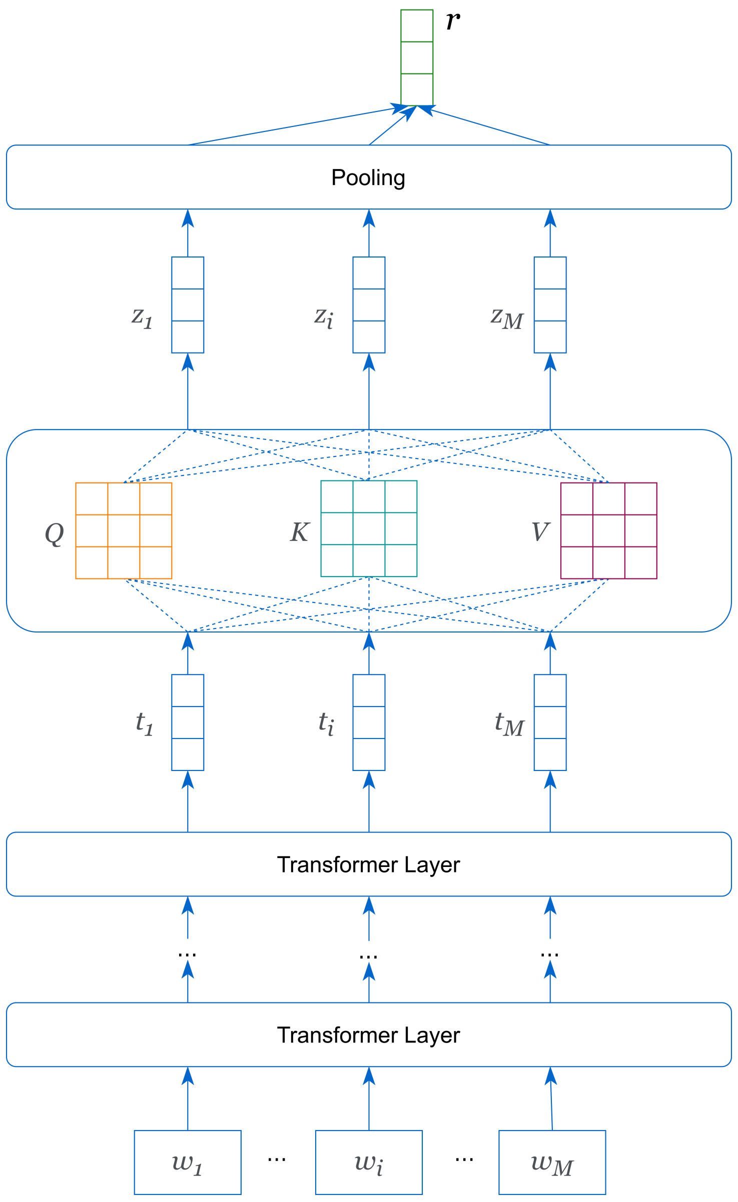


Figure 4.3: Paraformer's architecture of sentence encoder.

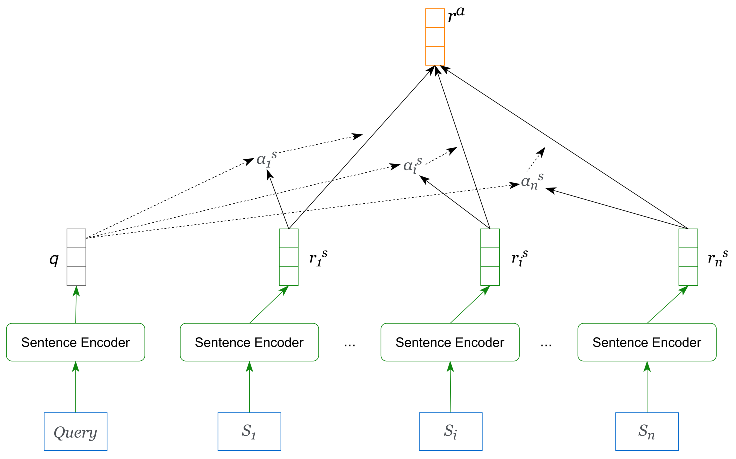


Figure 4.4: Paraformer's architecture of paragraph encoder.

are combined with the softmax function (Equation 4.1) to get the corresponding output at the last transformer layer $Z = (z_1, z_2, ..., z_M)$.

$$Z = softmax(\frac{Q \times K^\top}{\sqrt{d}} V) \tag{4.1}$$

An average pooling is applied on the $Z$ to get the final representation of the sentence:

$$r = \frac{1}{M} \sum_{i=1}^{M} z_i \tag{4.2}$$

The advantage of using the pretrained sentence encoder is that this module has been trained on a large corpus so it requires less data for the fine-tuning phase.

Figure 4.4 shows the architecture of the paragraph encoder, which combines the signal from the sentence encoders into the final representation. First, both the query and the article's sentences are encoded using the sentence encoder to get $q$ and $r_i^s$ vectors respectively. Then, with general attention, the representation of an article with a query is calculated by the Formula 4.3, 4.4, and 4.5.

$$a_i^s = q^T \tanh(A \times r_i^s + b) \tag{4.3}$$

$$\alpha_i^s = \text{sparsemax}(a_i^s)^* \tag{4.4}$$

$$r^a = \sum_{i=1}^{M} \alpha_i^s r_i^s \tag{4.5}$$

In which, $\alpha_i^s$ be the attention weights and $r^a$ be the final representation.

### 4.4.3 Experiments

The experimental data we use in this section is COLIEE 2021. This dataset is drawn from Japanese Legal Bar exams. The scope of the questions is limited to the Japanese Civil Code. Total samples in the training set and validation set are 806, the number of samples for testing is 81. The data is very valuable for us to experiment with our proposed approaches. Our validation set and test set are the formal evaluation data in COLIEE 2020 (65 queries) and COLIEE 2021 (81 queries), respectively, the rest is for training.

The retrieval process is done in two steps. The first step is choosing the top $N$ articles based on lexical matching to filter out clearly unrelated articles. This number can be different in the training phase and the prediction phase, we also have experiments to find the optimal value. The second step is reranking the article using the deep learning models.

We construct a relevance scoring function as in Equation 4.6.

$$S_f = \alpha \cdot S_l + (1-\alpha) \cdot S_s \tag{4.6}$$

where $S_f$ is the final score calculated from the lexical score $S_l$, and the semantic score $S_s$ is given by the deep learning model. $\alpha \in [0,1]$ is the hyperparameter determining the weights of the two scores. We apply grid-search to obtain the optimal value of $\alpha$.

We use XLM-RoBERTa as the backbone of our Paraformer. Table 4.6 is the performance comparison between Attentive CNN proposed in Section 3.4, vanilla XLM-RoBERTa and Paraformer without the ensembling with the lexical score. With Macro-F2@1 as the main evaluation metric, in general, Paraformer surpasses XLM-RoBERTa and Attentive CNN in both datasets. Its F2 scores in the English dataset and Japanese dataset are 0.3498, and 0.3182 respectively. Interestingly, from this result, pretrained models tend to achieve higher performance than the non-pretrained model (*i.e.*, Attentive CNN).

Table 4.7 presents the final performance on the test set after ensembling with the lexical score by the optimal value of $\alpha$. Paraformer outperforms other models and achieves state-of-the-art results in Precision (0.7901) and Macro-F2 (0.7407) and surpasses the current state-of-the-art system by Wehnert *et al.* [63]. The best recall belongs to the systems of Nguyen *et al.* [45] and Wehnert *et al.* [63].

To better explain the result, we visualize the attention weights of the model. Table 4.8 demonstrates the attention weights of Paraformer when answering queries related to Article 87 in Japanese Civil Code. As we can see in the table, the model focuses differently on the content of Article 87 depending on the given query. For the first two queries, the model pays attention to the first path of the article, for the last query, the weight of item (2) is superior.

Table 4.6: Results on COLIEE Datasets without Lexical Score

| **Systems** | **Precision** | **Recall** | **F2** |
|---|---|---|---|
| English Dataset | | | |
| XLM-RoBERTa | 0.2099 | 0.1975 | 0.1989 |
| Attentive CNN | 0.0864 | 0.0864 | 0.0864 |
| Paraformer | 0.3827 | 0.3450 | **0.3498** |
| Japanese Dataset | | | |
| XLM-RoBERTa | 0.2940 | 0.3086 | 0.3086 |
| Attentive CNN | 0.2593 | 0.2222 | 0.2263 |
| Paraformer | 0.3457 | 0.3148 | **0.3182** |

Table 4.7: Performance comparison on the COLIEE 2021's formal test set

| **Run ID** | **Precision** | **Recall** | **F2** |
|---|---|---|---|
| **Paraformer*** | **0.7901** | 0.7346 | **0.7407** |
| OvGU_run1 | 0.6749 | 0.7778 | 0.7302 |
| JNLP.CrossLMultiLThreshold | 0.6000 | **0.8025** | 0.7227 |
| BM25.UA | 0.7531 | 0.7037 | 0.7092 |
| JNLP.CrossLBertJP | 0.6241 | 0.7716 | 0.7090 |
| R3.LLNTU | 0.6656 | 0.7438 | 0.7047 |
| R2.LLNTU | 0.6770 | 0.7315 | 0.7039 |
| R1.LLNTU | 0.6368 | 0.7315 | 0.6875 |
| JNLP.CrossLBertJPC15030C15050 | 0.5535 | 0.7778 | 0.6838 |
| OvGU_run2 | 0.4857 | **0.8025** | 0.6717 |
| TFIDF.UA | 0.6790 | 0.6543 | 0.6571 |
| LM.UA | 0.5679 | 0.5432 | 0.5460 |
| TR_HB | 0.3333 | 0.6173 | 0.5226 |
| HUKB-3 | 0.2901 | 0.6975 | 0.5224 |
| HUKB-1 | 0.2397 | 0.6543 | 0.4732 |
| TR_AV1 | 0.2622 | 0.5123 | 0.3599 |
| TR_AV2 | 0.1490 | 0.5556 | 0.3369 |
| HUKB-2 | 0.3272 | 0.3272 | 0.3258 |
| OvGU_run3 | 0.1570 | 0.7006 | 0.3016 |

| | **Article 87 (1):** If the owner of a first thing attaches a second thing that the owner owns to the first thing to serve the ordinary use of the first thing, the thing that the owner attaches is an appurtenance. | **Article 87 (2):** An appurtenance is disposed of together with the principal thing if the principal thing is disposed of. |
|---|---|---|
| **Query 1:** Extended parts of a house shall be regarded as appurtenance. | 🟪 | |
| **Query 2:** Extended parts of a house shall be disposed when the house is no longer used. | 🟪 | 🟪 |
| **Query 3:** When an appurtenance is disposed of together with the principal thing? | 🟪 | 🟪 |

Table 4.8: Weight visualization of Paraformer when answering querries related to Article 87 in Japanese Civil Code.

Through visualization, we see that, besides improving performance, this approach also allows us to debug the model more easily. Instead of accepting the model's prediction as an output of a black box, we can look at the attention weight to debug the model. This is important for improving models in the legal field. In addition, it also opens up potential on the application aspect as users can learn how to focus on important information from the data.

### 4.4.4 Discussions

For a better understanding of the model behavior on different content lengths, we divide the English dataset into five chunks. The first chunk contains articles that are fewer than 100 characters in length. The second, the third, and the fourth chunks contain articles between 100-200, 200-300, and 300-400 characters in length, respectively. The fifth chunk contains articles that are longer than 400 characters. We use the trained XLM-RoBERTa and Paraformer to predict and record the performance of each model in each chunk without the lexical score.

Figure 4.5 shows the performance of the two models and their trendlines. Although Paraformer overperforms XLM-RoBERTa in all chunks, the performance of both models reduces when the length of content increases. However, looking at the trendlines, we can see that XLM-RoBERTa's performance tends to reduce faster than Paraformer in long content. Note that in this chart, we only display the query length but in fact, these models need to handle both queries and articles as their inputs. From this observation, we can see that our proposed architec-

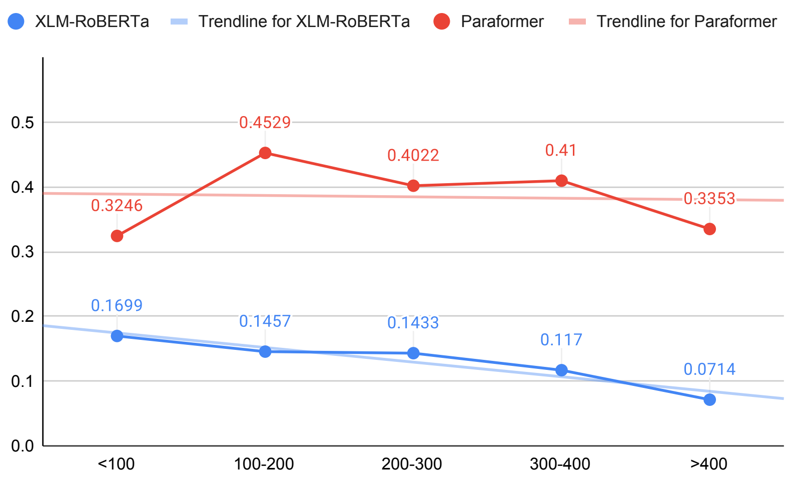


Figure 4.5: Performance of XLM-RoBERTa and Paraformer when working on different lengths of inputs.

ture is appropriate to solve the problem of content length and take advantage of pretrained language models.

## 4.5 Summary of Chapter

This chapter is about pretrained language models in deep legal processing. We propose BERTLaw, ParaLaw and Paraformer as three candidates to solve different limitations of current pretrained language models. From the results in Chapter 3, we see that the three factors that make a deep learning model work well in the legal domain are: good data representation, an adequate amount of data, and appropriate model architecture. The models proposed in this chapter are all based on those observations.

The proposal of BERTLaw proves that there is the problem of sublanguage in law and it can be solved by pretraining the model with the dataset in such sublanguage. The method of generating BERT Law is not a new method in deep learning. Even so, we apply this method to legal domains and prove it's effectiveness. BERT Law is created from scratch in both vocabulary and network weights. Note that BERT Law was introduced at the same time as LEGAL-BERT [18] and both confirm the superiority of this approach.

ParaLaw Nets show us the ability to use aligned translations available in legal documents to improve the performance of language models. With the need for legal internationalization of countries, high-quality translation resources of legal documents are available. This is a great strength that we can further pretrain multilingual embeddings to help them better model legal concepts. The two pretraining tasks we proposed in ParaLaw Nets force the models to predict the continuation of two multilingual legal sentences, thereby indirectly correcting the representation of concepts in vector space. In addition, this approach allows us to obtain more training data in the downstream tasks (English and Japanese). Since BERTLaw does not support Japanese, adding more Japanese data to train BERTLaw reduces the accuracy of this model, so it is difficult to have a fair configuration to compare the two winners of the two COLIEE seasons.

Paraformer contributes a novel architecture that leverages the power of pretrained language models with long content. A robust model can still fail without complete information. With technical limitations, pretrained language models are usually limited by maximum length setting. This is their weakness in legal domains with lengthy legal sentences. Paraformer's divide-and-conquer technique with the general attention mechanism helped us solve lengthy content problems while retaining the power of pretrained language models.

# Chapter 5

# Knowledge Injection for Deep Legal Models

## 5.1 Overview

In this chapter, we propose different techniques for injecting expert knowledge into the language model to improve the performance in deep legal processing. First, we investigate the linguistic knowledge and design a framework to inject this knowledge into the bulky language model without pretraining models from scratch. This is an efficient approach to improve the performance without increasing the computational resource. Second, we propose a novel technique to inject the knowledge about the structure of legal sentences into the Transformer models. We also conduct detailed experiments to find the best configuration for this approach. Third, we construct a novel conditional generative system that uses the knowledge of fairness to guide the generative models for high-quality terms of service content. Our recommendations in this chapter come from the insights gained from working on the previous chapters.

## 5.2 Linguistic Knowledge Injection

### 5.2.1 Introduction

In the digital age, the amount of data generated per second is enough to train a model capable of abstracting many different patterns. With a sufficiently large amount of data, these models are shown to be able to abstract the linguistic features on their own [38]. Even so, training the model unsupervised using data on the internet can generate noisy, biased models [7] and make deep learning black boxes to humans. This section introduces a novel method to pretrain a small part

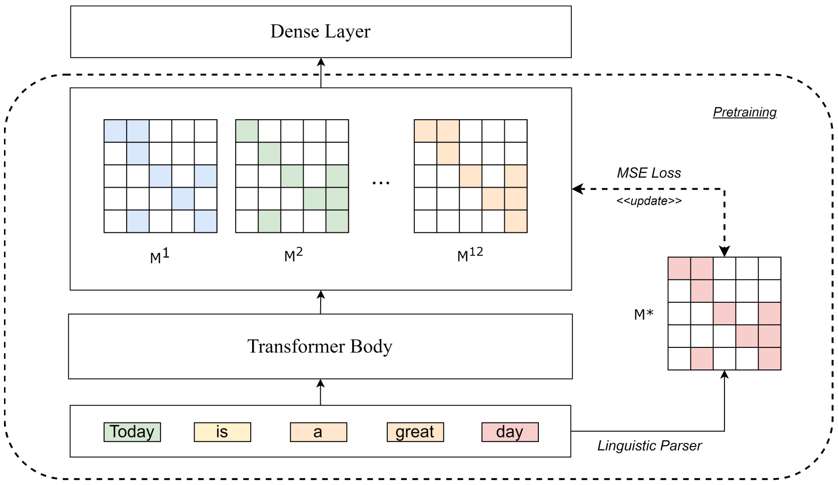


Figure 5.1: The general architecture of the framework. *Transformer Body* is the pre-trained model (e.g. BERT). SDOI matrix $(M^*)$ is the relation matrix parsed from an external Linguistic Parser tool (e.g. Stanford NLP tool or spaCy).

of the whole transformer architecture of the pretrained language model. Instead of pretrain a whole model, we append one more transformer layer, pretrain it to represent the linguistic knowledge. By doing so, the knowledge in this layer is utilized for the model to have better performance in the downstream tasks.

This approach has three advantages compared to other methods. First, this is a knowledge injection method that can take the edge of many different types of annotated resources in NLP. Linguistic is an important feature to understand language, especially in the legal domain. Second, the injected information is not rigid but only a reference source for the model to reach its final conclusion. Third, training and storing the appended component are extremely efficient, helping to solve environmental and financial problems.

### 5.2.2 Research Method

The overall architecture of the proposed framework is shown in Figure 5.1. Based on the transformer's architecture, we add one more transformer layer at the end of the architecture containing heads, which are pretrained with linguistic knowledge. The model can choose to use the information of these heads or ignore it with the residual connections. Therefore, we can use linguistic knowledge as a reference source for the model without forcing it to accept such knowledge rigidly. Since the idea is derived with a dependency structure, we give this architecture an inspiring name, *HYDRA*, which stands for *Hyper Dependency Representation Attentions*.

### Pretraining HYDRA Heads

Let the input be $X = \{x_1, x_2, ..., x_n\}$ with $n$ be the sequence length, we obtain an $n \times n$ matrix $M^*$ containing the linguistic relationship of the words as introduced by Zhang et al. [70]. After passing the input through the transformer layers, we get hidden state $H_l$ at the last layer with $l$ being the number of layers of the original transformer body. We initiate and append $l+1^{th}$ layer containing HYDRA heads and pass $H_l$ to this layer. With $W_q$, $W_k$ are learnable parameters in the layer $l+1^{th}$, we calculate query and key vectors as in Equations 5.1 and 5.2:

$$q^h_{l+1} = W_q \cdot H_l \tag{5.1}$$

$$k^h_{l+1} = W_k \cdot H_l \tag{5.2}$$

With $d_k$ be the dimension of the key matrix $k^h_{l+1}$, we calculate the attention matrix for each head $M^h$ following Equation 5.3:

$$M^h = \frac{q^h_{l+1} \cdot {k^h_{l+1}}^\intercal}{\sqrt{d_k}} \tag{5.3}$$

We then minimize the element-wise MSE loss ($\mathscr{L}$) between $M^*$ and $M^h$ for every HYDRA head:

$$\mathscr{L} = \frac{1}{n^2} \sum_{i=0}^{n^2} (M^*_i - M^h_i)^2 \tag{5.4}$$

where $i$ ($0 \le i < n^2$) is the index of element in flatten attention matrix SDOI ($M^*_i$). The weights of the transformer body are frozen and the weights in $Q^h_{l+1}$ and $K^h_{l+1}$ in every HYDRA heads are updated via the backpropagation process.

After pretraining, we get the HYDRA heads corresponding to the transformer body coupled to it. These HYDRA heads are saved in the storage and ready to be delivered as a form of deep learning resource. Compared to storing pretrained transformer models, storing HYDRA heads saves much more storage space. A transformer model checkpoint can be measured in gigabytes, while a HYDRA head checkpoint in our experiment is less than 10 megabytes.

### Fine-tuning with HYDRA Heads

Different from approaches that use linguistic knowledge that is rigidly injected into the model, we attach pretrained HYDRA heads to the transformer bodies. These models can refer to but are not limited by the linguistic knowledge learned in the HYDRA heads.

At this phase, the weights of the transformer body and HYDRA head are both updated to match the training data. The final model now contains $l+1$ layers, the first $l$ layers are from the original transformer body and the last layer contains HYDRA heads. We simply pass the $H_{l+1}$ to a fully connected layer to get the logits for each downstream task. The loss function of each downstream task is defined similarly to the work [23]. Intuition in this phase is that by acquiring linguistic knowledge, the system can properly model the problem rather than try to fit into the local minima.

### 5.2.3 Experiments

In the pretrain phase, we collect data from Wikimedia Downloads[1] using WikiExtractor package [3]. We use the variant of dependency parser described by Honnibal and Johnson [28], provided by spaCy[2]. After data processing, we obtain 330,000 samples for training and 50,000 samples for validation. We use BERT [23] as the base model to pretrain the HYDRA heads. We set the maximum sequence length to 512 and only use sentences whose length does not exceed this number to pretrain the HYDRA heads. The goal of this setting is to help the model learn the linguistic structure of complete sentences of the longest possible length. With 1.8 million parameters, in our experiment, it takes only one or two epochs for the HYDRA heads to reach optimal loss on both the training set and the validation set.

To understand the model's behavior and the method's effectiveness, we run experiments and compare the performance of the HYDRA variants with the corresponding non-HYDRA baselines. The experiments are run on datasets and standard settings of several famous benchmarks in natural language processing which include:

- QNLI [55], MNLI [65, 12], RTE [22, 5, 26, 8] : Natural Language Inference and Texture Entailment
  - Metric for QNLI and RTE: Accuracy.
  - Metric for MNLI: Accuracy for both matched (m) and mismatched (mm) versions.
- QQP [30] and STS-B [16]: Semantically Equivalence Judgment
  - Metric for QQP: Accuracy.
  - Metric for STS-B: Pearson Spearman Correlation.

[1]https://dumps.wikimedia.org/

[2]https://spacy.io/

Table 5.1: Experimental results on dev set of benchmark datasets

| **Benchmark** | **BERT** | **BERT HYDRA** |
|---|---|---|
| QNLI | 0.9065 | 0.9090 |
| MNLI_m | 0.8373 | 0.8401 |
| MNLI_mm | 0.8419 | 0.8458 |
| RTE | 0.6300 | 0.6336 |
| QQP | 0.9062 | 0.9067 |
| STS-B | 0.8800 | 0.8812 |
| SQuAD EM | 0.8105 | 0.8124 |
| SQuAD F1 | 0.8838 | 0.8851 |
| SQuAD 2.0 EM | 0.7137 | 0.7161 |
| SQuAD 2.0 F1 | 0.7458 | 0.7480 |
| COLIEE Task 5 | 0.5432 | 0.5679 |

- SQuAD [55], SQuAD 2.0 [54], COLIEE 2021 (Task 5) [31]: Question Answering
  - Metric for SQuAD and SQuAD 2.0: Exact Match and Macro-averaged F1 Score.
  - Metric for COLIEE 2021 (Task 5): Accuracy.

Table 5.1 shows the results of the models on the benchmarks and metrics described above. BERT is a very strong baseline, it achieved high performance on all benchmarks. Even so, our model still can slightly improve the results of BERT with the appended HYDRA heads. This result supports the hypothesis that linguistic knowledge pretrained in HYDRA heads can boost the performance of the vanilla model.

In addition to the performance improvement, this is a novel approach to inject linguistic knowledge into language models. One notable feature of our improvement is that this new component is lightweight, requires low pretraining computation cost and storage. With this paradigm, we can improve the bulky transformers models without pretraining the whole network again or forcing them to follow rigid linguistic rules.

### 5.2.4 Discussions

This section proposes an architecture-friendly and extensible method to improve the effectiveness of the transformer-based language models by pretraining and appending new knowledge-guided heads to their architecture. We conduct the experiment with BERT as the base model and the dependency information as the external knowledge. Our experiment shows that our lightweight component can help to boost the performance of transformer models and provide a flexible paradigm to partially inject the knowledge into bulky models without pretraining them again. Extending this work, we can analyze the possibility of pretraining this component with different knowledge forms for problems in narrower domains or explore the potential of this approach for other data such as photos or videos.

## 5.3 Legal Knowledge Injection

### 5.3.1 Introduction

In this section, we introduce TRE framework which is a knowledge injection framework for Transformer based models with requisite and effectuation data. Applying this framework, we investigate and pretrain variants of TREBERT, pretrained models on BERT. Logic structures are integral parts of legal sentences. Identifying criminals, breaches of contracts, and a host of other important legal decisions are all based on logic. A novel author can use better language than a lawyer but standing in court they cannot justify a person from the death penalty with their words without logic. Similarly, a language model trained on a giant corpus without knowledge of logic is intrinsically useless in law. This can make it difficult to answer the inference questions of the law, which are critical in being able to bring the results to reality.

Analyzing the previous pretraining methods in the legal domain, we find that these methods have the same thing in common, that they are pretrained unsupervised on a large corpus. By doing so, we can create language models that accurately describe the relationships of concepts, terms, and syntax used in legal documents. These models can also find latent rules expressed in words, use extrapolation to make decisions. However, it is impossible for the model to find all the latent rules just by identifying co-occurring terms. This is a process that requires much time, a lot of computational power, a huge amount of data. Similar to the issues in math problems, it is difficult for the model to find logical rules through unsupervised training. Our approach is a further pretraining method based on a supervised paradigm.

### 5.3.2 Research Method

With a different purpose than daily life sentences, legal sentences often require rigor and logic. As the product of thousands of years of human civilization, the logic of the existing laws reaches a very high level. From a syntactic point of view, two important components of a law sentence to form an equivalent logical proposition are requisite and effectuation. Requisite and effectuation can be formed from smaller logical parts such as antecedence, consequence, and topic.

With a classic example in the logic “If it rains, the road is wet.”, we can easily see that this sentence has a requisite segment and an effectuation segment. The requisite and effectuation segments are often complex in practice, they can be nested and even interleaved. In legal sentences, besides *requisite* and *effectuation*, another common logical structure is *unless*, which indicates exceptions where the main requisite and effectuation do not apply. Let’s consider the following exam-

Table 5.2: Multilayer annotation in the BIOE schema of *requisite*, *effectuation* and *unless* segments for the sample "Gifts not in writing may be revoked by either party; provided, however, that this shall not apply to any portion of the gift for which performance has been completed."

| **Token** | **L1** | **L2** | **L3** |
|---|---|---|---|
| Gifts | B-R | B-E | - |
| not | I-R | - | - |
| in | I-R | - | - |
| writing | E-R | - | - |
| may | - | I-E | - |
| be | - | I-E | - |
| revoked | - | I-E | - |
| by | - | I-E | - |
| either | - | I-E | - |
| party | - | E-E | - |
| ; | - | - | - |
| provided | - | - | B-U |
| , | - | - | I-U |
| however | - | - | I-U |
| , | - | - | I-U |
| that | - | - | I-U |
| this | - | B-E | I-U |
| shall | - | I-E | I-U |
| not | - | I-E | I-U |
| apply | - | I-E | I-U |
| to | - | I-E | I-U |
| any | - | I-E | I-U |
| portion | B-R | I-E | I-U |
| of | I-R | I-E | I-U |
| the | I-R | I-E | I-U |
| gift | I-R | E-E | I-U |
| for | I-R | - | I-U |
| which | I-R | - | I-U |
| performance | I-R | - | I-U |
| has | I-R | - | I-U |
| been | I-R | - | I-U |
| completed | E-R | - | E-U |
| . | - | - | - |

ple: "Gifts not in writing may be revoked by either party; provided, however, that this shall not apply to any portion of the gift for which performance has been completed." With such a complex sentence, it is easy to see that there is more than one requisite and effectuation pair in this sentence. Therefore, it is difficult for a language model with averaging and interpolation capabilities to infer logical structures on its own through unsupervised training. To correctly annotate law sentences with many interlocking logical structures, we need to use multilayer annotation [49]. Table 5.2 is the annotation of the above example.

With the goal of building a Transformer model that can learn to recognize the segments of logical structures, we propose the TRE (Transferred-Requisite-Effective) Framework. This framework makes it possible to inject the logical structure information into the self-attention layers of the Transformer so that the model can form the corresponding abstractions. Unlike conventional pretraining approaches, labels are usually provided at the last Transformer layer, within this framework, information about logical structures can be injected into the hidden layers (Figure 5.2).

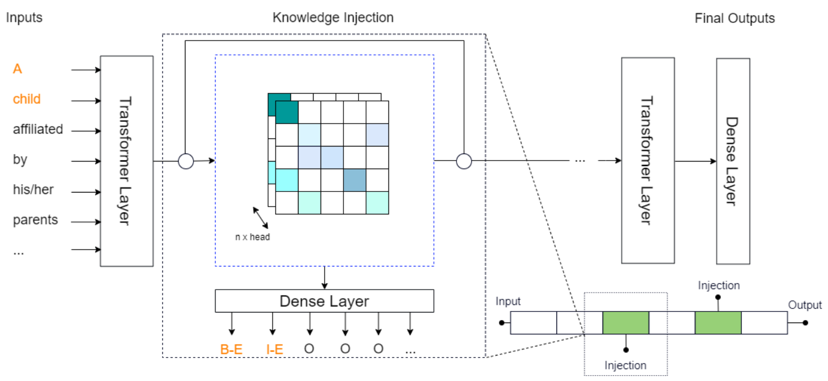


Figure 5.2: General flow of TRE Framework for Transformer models.

With this framework, knowledge injection is done through a gradient descent process. Instead of rigidly specifying information about logical structures through constants, the model's parameters need to be updated so that corresponding abstractions can be formed. The novelty of this framework lies in the Transferred-Requisite-Effective self-attention layers (TRE layers). These layers stores the knowledge about recognizing logical structures in the legal sentence, which is learned from the provided labels in pretraining data.

Suppose after the layer $i-1^{th}$, we have a signal sequence of length $M$ which can be presented as $E^{i-1} = (e_1^{i-1}, e_2^{i-1}, ..., e_M^{i-1})$. The $i^{th}$ layer is a TRE layer, basically, the information flow is the same as in a regular Transformer layer. At the $i^{th}$ layer, vector $E^{i-1}$ is multiplied by the attention matrices $Q^i$, $K^i$, $V^i$ to get the corresponding attention vectors. These attention vectors are combined according to Equation 5.5 to get the corresponding output at the $i^{th}$ transfomer layer $Z^i = (z_1^i, z_2^i, ..., z_M^i)$.

$$Z^i = softmax(\frac{Q^i \times K^{i\top}}{\sqrt{d}} V^i) \tag{5.5}$$

In the forward direction of Transformer architecture, we normalize $Z^i$ with a layer normalization [4] and a dense layer, the signal can also be transmitted directly through the residual connections. To pass the labels of the logical structures to the network, at the branching direction as an injection needle, $Z^i$ after going through a dense layer and the softmax function as in Equation 5.6, predicted labels and gold labels are compared and the loss is backpropagated for updating parameters.

$$L = softmax(Z^i \times W + b) \tag{5.6}$$

Table 5.3: Tag distribution following BIOE schema of the pretraining data.

| Tag | Meaning | Occurence |
|---|---|---:|
| B-E | Begin Effectuation Part | 1,982 |
| B-R | Begin Requisite Part | 2,408 |
| B-U | Begin Unless Part | 260 |
| E-E | End Effectuation Part | 2,088 |
| E-R | End Requisite Part | 2,395 |
| E-U | End Unless Part | 259 |
| I-E | Inside Effectuation Part | 26,762 |
| I-R | Inside Requisite Part | 31,474 |
| I-U | Inside Unless Part | 6,270 |
| O | Others | 148,540 |

### 5.3.3 Experiments

We pretrain TREBERT as described in Section 5.3.2. This pretraining process has the following characteristics. Firstly, it is further pretraining on a Transformer model, *i.e.*, the original model needs to be pretrained on basic linguistic tasks to form contexture embedding on a specific vocabulary before being pretrained with logical structure data. Secondly, it is implemented as supervised learning, *i.e.*, trained with labeled data. Thirdly, the TRE layers where the label is injected will be trained in parallel.

From the characteristics of this process, we see that, instead of assigning just one configuration, we can experiment with different configurations of the framework. This experiment not only helps to find the best configuration but also helps us better understand the behavior of the model during this phase. Since pretraining is done in the form of supervised training, we can track the performance of the model on a validation set. Our assumption is that a good configuration can help the model to make a good abstraction, thereby making accurate predictions on the validation set.

With statistics from Table 5.3, we can see that this is a classification problem with unbalanced labels. Therefore, we use the precision, recall, and F1 metrics on logical structure parts to evaluate and compare

We pretrain the variants of TREBERT considering two aspects: the position of the TRE Layers and the loss portion between them. Just considering the position of the TRE layer on the 12 Transformer layers of BERT, we have ${}^{12}P_3 = 1320$ cases. Testing all configurations is resource-intensive, so we use the boundary value analysis technique to generate representative configurations for the positions and random search for the loss portion. Table 5.4 represents positional configurations and their performances on pretraining data. We also included in the table

Table 5.4: Representative positional configuration's performances on validation set. TREBERT_X_Y_Z stands for the best configuration that the knowledge is injected in Transformer layers $X^{th}$, $Y^{th}$ and $Z^{th}$.

| **Configuration** | **Precision** | **Recall** | **F1 Score** |
|---|---|---|---|
| *Uniform loss portion* | | | |
| TREBERT_8_10_12 | 0.5890 | 0.7000 | 0.6373 |
| TREBERT_7_8_9 | 0.5441 | 0.7102 | 0.6151 |
| TREBERT_7_9_11 | 0.5420 | 0.7305 | 0.6140 |
| TREBERT_10_11_12 | 0.5544 | 0.6661 | 0.6064 |
| TREBERT_6_9_12 | 0.5262 | 0.7424 | 0.5959 |
| TREBERT_4_8_12 | 0.4500 | 0.6966 | 0.5102 |
| TREBERT_2_7_12 | 0.3994 | 0.7136 | 0.4549 |
| TREBERT_4_5_6 | 0.3550 | 0.5814 | 0.4235 |
| TREBERT_1_6_11 | 0.3490 | 0.7085 | 0.4027 |
| TREBERT_2_4_6 | 0.3266 | 0.6678 | 0.3894 |
| TREBERT_1_5_9 | 0.3260 | 0.7254 | 0.3704 |
| TREBERT_1_4_7 | 0.2745 | 0.6695 | 0.3254 |
| TREBERT_1_3_5 | 0.2199 | 0.5695 | 0.2726 |
| TREBERT_1_2_3 | 0.1655 | 0.4932 | 0.2124 |
| Avg Random (30 runs) | 0.3654 | 0.5826 | 0.4088 |
| *Optimized loss portion* | | | |
| TREBERT_8_10_12 | 0.5807 | 0.7373 | 0.6555 |
| TREBERT_7_8_9 | 0.5725 | 0.6814 | 0.6313 |
| **TREBERT_7_9_11** | **0.6300** | **0.7723** | **0.6939** |

30 configurations where the TRE layers are randomly decided. When conducting experiments with TRE layer position, we fixed the portion loss of all three logical structure layers equally.

From the experiment, it can be seen that the good positions to inject the knowledge of the logical structure are in the deeper layers. This can be explained that the knowledge injection needs to match the level of abstraction of the neural network. For deep learning models, in the early layers, the abstractions are low-level. At the deeper layers, the level of abstraction increases. The knowledge injected into the early layers will deviate from the abstraction level compared to the unsupervised features formed during the previous pretraining. The table also shows that the distance between the injection needle positions should be 1 to 2 layers. We select the 3 best configurations to continue conducting random searches to find the best portion of the loss functions for each. Different positional configurations have different optimal loss portions.

Table 5.5: Performance on the test set of COLIEE 2021.

| **Model / Configuration** | **Correct** | **Accuracy** |
|---|---|---|
| TREBERT_8_10_12 | 52 | 0.6420 |
| TREBERT_7_9_11 | 52 | 0.6420 |
| TREBERT_7_8_9 | 49 | 0.6049 |
| LEGAL_BERT_SC | 47 | 0.5802 |
| LEGAL_BERT_FC | 46 | 0.5679 |
| Original BERT | 44 | 0.5432 |
| TREBERT_1_2_3 | 44 | 0.5432 |

We use question answering data of COLIEE 2021 competition to verify the effectiveness of TREBERT. For each statement, the model needs to predict whether the statement is lawful or not. This is data that can evaluate the strength of pretrained models because of the small number of samples. The systems need to perceive the semantics in the sentence to give the correct answer.

In the data provided by COLIEE, the official test set includes 81 yes/no questions. To generate the training data, we use the method proposed by [42], data from previous years and from the Japanese civil code are augmented with simple negation rules. After the augmentation process, we have 4,000 samples. We spend 10% for the development set and the rest is for finetuning TREBERT and baseline models.

Our experimental baselines include original BERT, LegalBERT_SC (Legal BERT from scratch), and LegalBERT_FP (Legal BERT further pretrained). These baselines have the common feature of being pretrained unsupervised on lexical tasks and not pretrained with logical structures. The measurement used in this task is accuracy. TREBERT_8_10_12, TREBERT_7_8_9 and TREBERT_7_9_11, the variants of TREBERT that performed best on the pretraining task, are included in the experiment. In addition, we also used a configuration with poor pretraining results, TREBERT_1_2_3, to further understand the behavior of this model family.

To ensure fairness in the number of parameters, the baselines and variants of TREBERT all use the architecture and configuration of BERT Base Uncased. Note that TREBERT's injection needles are removed after pretraining so they don't affect the parameter count. The experiments are conducted with GPU Tesla P100-PCIE-16GB.

Experimental results in Table 5.5 show that TREBERT_8_10_12 and TREBERT_7_9_11 lead the rankings with 52 correct answers out of a total of 81 questions in the test set, followed by TREBERT_7_8_9, LEGAL_BERT_SC and LEGAL_BERT_FC with 49, 47 and 46 correct answers. Original BERT and TREBERT_1_2_3 answered 44 questions correctly. This result shows us that the

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Gift | not | in | writing | may | be | revoked | by | either |
| party | ; | provided | , | however | , | that | this | shall |
| not | apply | to | any | portion | of | the | gift | for |
| which | performance | has | been | completed | . | | | |
| Gift | not | in | writing | may | be | revoked | by | either |
| party | ; | provided | , | however | , | that | this | shall |
| not | apply | to | any | portion | of | the | gift | for |
| which | performance | has | been | completed | . | | | |

Table 5.6: Self-attention visualization of the 11th layer in TREBERT_7_9_11.

hypothesis made at the beginning of the section is reasonable. Pretraining the models using a logical structure helps them to make better predictions in tasks that require understanding in the legal domain. Compared with the results announced by COLIEE-2021 [3], top TREBERT variants (TREBERT_8_10_12, TREBERT_7_9_11) achieved state-of-the-art performance.

To better understand the behavior of the model, we visualize the self-attention weight of the last layer where the injection needle is attached. If the model generates the correct abstraction, the attention matrix must reflect that. Table 5.6 is about the self-attention visualization of the 11th layer in TREBERT_7_9_11 with the input as "Gifts not in writing may be revoked by either party; provided, however, that this shall not apply to any portion of the gift for which performance has been completed.".

Looking at Table 5.6, it can be seen that TREBERT_7_9_11 has adjusted its attention to the logical structures. With the token "gift", from Table 5.2, we can see that it belongs to both logical parts requisite and effectuation. TREBERT_7_9_11's attention weights correctly tie "gift" to the tokens to the requisite (gift not in writing) and to the effectuation (gift may be revoked by either party) parts. This could explain the outperforming on the test set of the model.

TREBERT_1_2_3 has bad performance on the pretraining task, resulting in no performance improvement compared to the original BERT. As analyzed above, knowledge injection fails due to the incompatibility of abstraction between data and model architecture. This result is consistent with the assumption that a model capable of analyzing the logical structures can answer the legal questions better. Working with different configurations in the pretraining phase helps us find the right place to inject each type of knowledge.

[3]https://sites.ualberta.ca/ rabelo/COLIEE2021/results
/task5_res.html

### 5.3.4 Discussions

In this study, we propose and investigate TRE framework, a logic-structure knowledge injection approach for pretrained Transformer models. We then apply the TRE framework to pretrain the variants of TREBERT from the original BERT model. Our detailed experiments and surveys show the effectiveness and explainability of the method. A model having good skill in recognizing logical structure performs better on legal question answering.

Although in this section, the TRE framework is proposed with logical structures, the idea is general and extensible. Variations of knowledge can be discovered and utilized in different tasks. In future studies, we want to inject NLP annotated resources into Transformer models with TRE framework to get better and more explanatory pretrained models. In addition, the limitation of this approach is that finding good configurations consumes a lot of computing power for hyperparameter optimization, overcoming this limitation is also an interesting research direction.

## 5.4 Pretrained Self-regulated Generation

### 5.4.1 Introduction

In this section, we introduce a novel pretrained self-regulated generative framework with the aim of introducing the idea that knowledge learned from one model can be used to improve the results of another generative model in the framework. Generative models have long been introduced as a way of demonstrating machine intelligence to language. They developed from rule-based models [64] to statistical-based models [2, 27, 67] and the latest to attentive deep learning-based models like GPT-2 [53], GPT-3 [13] and BART [37], which recently received great attention the society.

Not only that, these models also raise public concerns with destructive applications such as fake news. This shows that it is almost impossible to distinguish human-written text from the language-generated text. The question is, is it possible to use a form of human-accepted knowledge to guide these models to produce high-quality texts. Looking for an answer to this question, we experiment with the terms-of-service generation problem. This problem contains two main difficulties. First, it requires a balance between automation and the will of the editor. Second, the meaning of the content that the system generates should be of high quality and fairness.

In this section, we introduce BART2S, a framework that contains two components, generator and discriminator. The generator is responsible for generating terms of use, taking into account the balance between editor will and automation. To this end, we design this component as a sequence-to-sequence model on the title-based generation problem. The discriminator is trained with the knowledge of fairness, then plays the role of a regulation component to ensure the output quality of the generative framework with few-shot tuning. With this process, our framework introduces a novel way to use encoded knowledge for the high-quality content generation problem.

### 5.4.2 Research Method

The challenge of the title-based generation problem is signal recovery. We need to recover the information of an entire clause based on their brief title. Therefore, we propose 3 training tasks for generators: next sentence generation, title-based generation and paraphrasing. Each task contributes a skill to the model that can generate content from the title.

For the discriminator, we let this component learn the knowledge of fairness through the fairness classification task. In essence, this is a binary learning task. For each text input, the model needs to learn to evaluate whether the text is a legally fair text. This is the component that sets our framework apart from other systems and helps us to create high-quality content from the generative model.

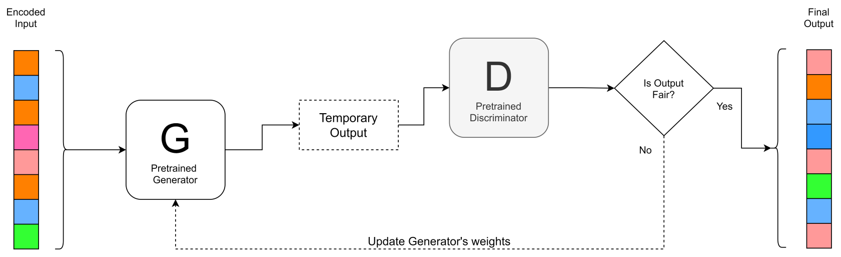


Figure 5.3: Fewshot tuning for generation in BART2S framework.

The few-shot tuning process can be described as in Figure 5.3. We consider the two models as differentiable functions $G(x,\theta_g)$ and $D(x,\theta_d)$ with $x$, $\theta_g$, and $\theta_d$ are the input, generator's parameters and discriminator's parameters, respectively. To generate high quality content, we minimize $log(1-D\circ G(x,\theta_g))$ using gradient descent process. An important modification in our framework is the *SoftArgMax* function at the last layer of the generator. To pass the loss across the two models, we need to replace the standard *ArgMax* with the function as follows:

$$\text{SoftArgMax}(x) = \sum_i \frac{e^{\beta x_i}}{\sum_j e^{\beta x_j}} i \tag{5.7}$$

where $x = [x_1, x_2, x_3, ..., x_n]$ and $\beta \geq 1$.

In the few-shot tuning process, we do not update the weights of the discriminator by freezing their gradient. This is important so that this component becomes an independent observer. The only standard it bases on is the knowledge of fairness it learns at the pretraining phase. As a result, the total loss reduction of the framework comes from the better content generated by the generator. This design makes a bold difference in our system compared to naive copy/paste systems and other non-regulated systems.

### 5.4.3 Experiments

Table 5.7 contains the information about the data to train the generator and discriminator. Pretraining the generator, each input is transformed using Token Masking, Deletion and Infilling [37]. For the discriminator, we use the knowledge of fairness from the ToS;DR project, which provides the rating of legal experts over different Terms of Service from many different companies.

To evaluate the system, we use both automatic and human-based metrics. For the generator, we use BLEU scores on the 1000 most popular contract terms according to Law Insider statistics. For the discriminator, we use accuracy on the

| **Component** | **Task** | **Data source** | **Samples** |
|---|---|---|---|
| Generator | Task 1: Next Sentence Generation | Crawled Terms of Service | 5,323 |
| | Task 2: Title-based Generation | LawInsider Contracts[a] | 901 |
| | Task 3: Paraphrasing | MSRP dataset [24] | 3,728 |
| Discriminator | Fairness Classification | ToS;DR project[b] | 4,152 |

[a]https://lawinsider.com
[b]https://tosdr.org/

Table 5.7: Data for training generator and discriminator.

| **Model / Approach** | **Performance** |
|---|---|
| Generator - All Tasks | 60.1 |
| Generator - w/o Task 1 | 59.9 |
| Generator - w/o Task 2 | 57.3 |
| Generator - w/o Task 3 | 56.3 |
| Discriminator | 66.0 |

Table 5.8: Performance of components with automatic metrics. Generator is evaluated in BLEU, Discriminator is evaluated in accuracy.

10% of data as the validation set. For the entire framework, we manually evaluate 4 criteria *Grammar*, *Readability*, *Relevance*, and *Fairness*. We select control systems from different robust configurations of BART Large [37]. Each model is passed in 30 short titles with an average length of 23 characters. Their output is evaluated by 10 independent evaluators according to the formula:

$$score_a(\mathrm{M}) = \frac{1}{n}\sum_{i=1}^{n}\frac{p_a^i}{s} \tag{5.8}$$

In which, $score_a(\mathrm{M})$ is the evaluation score of model $M$ in aspect $a$, $s$ is the total of sentences, $n$ is the number of evaluators, $p_a^i$ is the number of sentences evaluated as possitive by $i^{th}$ evaluator in the aspect $a$.

Table 5.8 shows the performance of components with automatic metrics, in which the generator is evaluated in BLEU and the discriminator is evaluated in accuracy. We can see that the generator trained with all proposed tasks achieve the best performance with a 60.1 BLEU score. Eliminating any task causes results to drop. The discriminator is trained with early stopping. This model achieves 66% accuracy on the validation set, which indicates that the problem of fairness classification is not straightforward.

Table 5.9 shows the performance of the proposed framework and control systems in the human evaluation as described. Because of the technical limitation, the maximum length of the generated content of all systems is 512 subwords.

| System | Grammar | Readability | Relevance | Fairness |
|---|---|---|---|---|
| BART Large w/o Ft | 0.34 | 0.31 | 0.41 | 0.43 |
| BART Large MNLI | 0.32 | 0.33 | 0.37 | 0.37 |
| BART Large CNN | 0.69 | 0.73 | 0.72 | 0.86 |
| BART2S | **0.80** | **0.82** | **0.87** | **0.94** |

Table 5.9: Evaluation results on grammar, readability, relevance, and fairness of each system. The underlined line indicates our proposed system.

With the multi-task learning and novel few-shot tunning procedure, our BART2S Framework achieves state-of-the-art results in all metrics. Although the results yield positive possibilities, note that all scores are measured in the local scope. For example, we can see that BART2S achieve a very high result in terms of fairness but this number only reflects that the evaluators find few fairness concerns in the generated text, which does not guarantee that the combination of them at the document level would remain the same fairness.

### 5.4.4 Discussions

The BART2S framework proposed in this section illustrates the possibilities in using knowledge of fairness to improve the output quality of generative models. The framework is designed for terms-of-service generation problem with two components, generator and discriminator. The generator is trained on multi-task to generate content from a short title. The discriminator learns about knowledge of fairness, then plays the role of constraining the output quality. The few-shot tunning framework proposed in this section can be widely applied in problems requiring output quality according to a standard of available knowledge.

## 5.5 Summary of Chapter

The content of this chapter is about using different sources of knowledge to enhance the robustness and explainability of pretrained language models through a technique we call knowledge injection. Instead of rigidly imposing human knowledge sources or letting deep learning models self-synthesize completely from data, this technique allows the models to update the weights based on the injected knowledge. To this end, we introduce HYDRA, TRE and BART2S frameworks, which implement different ways to inject the knowledge into the models.

HYDRA is our novel framework using the knowledge of dependency relationships to guide the prediction of the model. A new transformer layer is added to the existing transformer body and pretrained to imitate the syntactic dependency of interest matrix generated from an existing linguistic parser. This architecture is interesting in that the pretraining process does not update the weights of the entire model but only the added lightweight layer. HYDRA is named after a mythical creature with many heads and it is also our goal in the future to add more attention heads with different types of knowledge to enhance the power of this framework.

For TREBERT, knowledge of the legal structure of the legal sentence is injected into the model through injection needles into different layers. TRE in Vietnamese means bamboo, a tree with many sections, similarly, this framework allows knowledge to be injected into different layers of the transformer model. Through experimentation, we found that injecting knowledge into the layers at the end of the model brings better results. With legal knowledge injected, this model outperforms the control models in our experiment.

BART2S is designed as a GAN-like architecture containing a generator and a discriminator. The generator's outputs are regulated by the discriminator, which is pretrained on the knowledge of fairness. Our novel few-shot tuning framework is proved to enhance the quality of the generated terms-of-service content

With the proposed idea of knowledge injection, instead of letting them deal with raw data on their own, humans can engage these systems more efficiently and responsibly. Despite the preliminary positive results, there is still work that needs to be done to fully prove the effectiveness of these approaches in particular and knowledge injection for legal attentive models in general. In future work, we plan to conduct more experiments, ablation studies, and explore more knowledge source and their characteristic to produce more reliable and robust models in legal document processing.

# Chapter 6

# Conclusions

## 6.1 Conclusions

Using individual studies as basic material, this dissertation proposes an overall solution toward improving attentive neural networks in legal text processing. Legal text processing is a special subfield of natural language processing. The complexity of legal sentences and the high quality requirements of legal AI systems lead to a series of problems for models with attentive neural networks, which have been very successful and have brought many surprises to everyone when applying to problems in the general domain. Within the scope of a doctoral dissertation, we raise and address four main problems of these models working with legal text. To achieve this goal, the dissertation is presented in 6 chapters with the first two chapters providing introductory information, the main problems being handled in the next three chapters, and the last chapter is used to discuss, conclude and propose future directions from the initial findings of this dissertation.

The first problem is lacking of data for deep learning models. Deep learning models in general and attentive models, in particular, are data-hungry computational models. Unlike classical approaches based on rules, frames or heuristic algorithms, these models need a certain amount of data to achieve good results. This is also the reason why neural networks were created in the early years of computing, but until recently, with the huge amount of data generated by the Internet, these models have received the attention they deserve. The problem needs to be solved in order to be able to apply deep learning to legal text processing. We examine the problem and introduce our solution in Chapter 3 of the dissertation. This approach has proven effective in our COLIEE entries in 2020 and 2021. Besides, in Chapter 3, we also introduce the features of embeddings, model architecture in deep legal processing. This is an important premise for us to propose solutions to the problems in the following chapters.

The second problem is domain difference for pretrained language models. Pretrained language models are models that are pretrained on large amounts of text by unsupervised learning (*i.e.*, no labeled data is provided). Tasks designed to pretrain these systems have the purpose of supporting them to model concepts in the language. Law is a sublanguage with vocabulary and concepts different from everyday language. Therefore, we base on this feature and the unique resources available in the legal field to obtain better-pretrained language models for problems in this domain. Specifically, we introduce BERT Law and ParaLaw Nets, pretrained language models in the legal domain, with positive results. Our contribution for BERT Law is to confirm the effectiveness of a simple solution in domain adaptation (*i.e.*, change the data) for the legal domain. For ParaLaw Nets, we propose novel pretraining tasks to help these models enhance language comprehension through cross-lingual understanding tasks.

The third problem is lengthy content, a constant feature of legal text. This is a serious problem for pretrained language models. These models are limited to maximum length during pretraining. This technical limitation means that these models can only obtain information at the beginning of a paragraph for a long text. No matter how powerful a model is, it can't make accurate predictions if it doesn't have enough information. This problem was discovered by us in Chapter 3 and solved in Chapter 4 with a novel architecture named Paraformer. With the modeling of a paragraph as a set of sentences, we encode individual sentences and then synthesize the signal through the general attention mechanism. This approach allows us to encode longer legal sentences compared to the standard approach, which brings better performance.

The last problem introduced in this dissertation is uncontrolled learning. Although proven effective, leaving self-learning models entirely based on data precludes the possibility of human involvement in the process. As a result, models may interpret data differently than humans do, leading to reduced explainability of the model. Our proposed solution is to use knowledge sources to guide the learning of these models through different strategies introduced in Chapter 5. HYDRA is a novel framework using a dependency structure as a linguistic knowledge injected into the attention matrix of the newly appended layer. TRE is a newly proposed framework with logical structures of law sentences trained on different layers of a Transformer-based model. BART2S is a tunable framework that uses knowledge of fairness to constrain the generator to produce high-quality output.

## 6.2 Discussions and Future Works

The studies in this dissertation can contribute to a revolution in a broader scope. The findings in Chapter 3 are useful references for scientists and industry when approaching legal text processing problems with the use of deep learning models in general and attentive models in particular. Issues such as data amount, data representation, and model architecture occur not only for legal domain adaptation but for all narrow domains. Our approach in this dissertation can help them get ideas to develop suitable solutions to their own problems. The solutions mentioned in Chapter 4 are topical solutions. Pretrained language models are getting the most attention lately, and using them with possible enhanced solutions can help increase efficiency and save time. The approaches proposed in Chapter 5 are innovatory ideas. Using knowledge as a resource to increase model strength and explainability is a compromise between rigid rules and uncontrolled self-learning. This is an approach that requires the participation of experts in various fields, and much work remains to be done to achieve the goals set out in this chapter in a wider scope.

With the initial positive results in this dissertation, we plan to pursue more ambitious future works. First and most straightforward, we intend to examine factors other than those introduced in this dissertation, namely possible issues of deep legal processing, enhancement techniques, and possible knowledge source for guiding deep legal systems. Second, we expand our exploration of unresolved issues with deep legal systems. These problems are numerous and often very challenging. For example, they could be the ability to recognize and understand relationships between entities that appear in a sentence, the ability of reasoning inside a single document and inter-document, or the ability to detect users' latent intent from a query. To solve these problems, we need to make more use of data representations (*e.g.* logical, semantic, AMR, etc.) and corresponding knowledge sources. Last but not least, all scientific works are for only reference until it is actually applied to real life. We plan to pursue this line of research for a long time, perfecting it, commercializing it, and improving people's quality of life through smart and reliable legal services.

# Bibliography

[1] N. Aletras, D. Tsarapatsanis, D. Preoţiuc-Pietro, and V. Lampos. Predicting judicial decisions of the european court of human rights: A natural language processing perspective. *PeerJ Computer Science*, 2:e93, 2016.

[2] G. Angeli, P. Liang, and D. Klein. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512, 2010.

[3] G. Attardi. Wikiextractor, 2015.

[4] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[5] R. Bar-Haim, I. Dagan, B. Dolan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor. The second pascal recognising textual entailment challenge. In *Proceedings of the second PASCAL challenges workshop on recognising textual entailment*, volume 6, pages 6–4. Venice, 2006.

[6] M. P. Basgalupp, R. C. Barros, A. C. de Carvalho, A. A. Freitas, and D. D. Ruiz. Legal-tree: a lexicographic multi-objective genetic algorithm for decision tree induction. In *Proceedings of the 2009 ACM symposium on Applied Computing*, pages 1085–1090, 2009.

[7] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623, 2021.

[8] L. Bentivogli, P. Clark, I. Dagan, and D. Giampiccolo. The fifth pascal recognizing textual entailment challenge. In *TAC*, 2009.

[9] P. Bhattacharya, K. Hiware, S. Rajgaria, N. Pochhi, K. Ghosh, and S. Ghosh. A comparative study of summarization algorithms applied to legal case judgments. In *European Conference on Information Retrieval*, pages 413–428. Springer, 2019.

[10] L. Borges, B. Martins, and P. Calado. Combining similarity features and deep representation learning for stance detection in the context of checking fake news. *Journal of Data and Information Quality (JDIQ)*, 11(3):1–26, 2019.

[11] J. Bouvier. *A Law Dictionary, Adapted to the Constitution and Laws of the United States of America, and of the Several States of the American Union: with References to the Civil and other Systems of Foreign Law*, volume 2. GW Childs, 1870.

[12] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*, 2015.

[13] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[14] B. G. Buchanan and T. E. Headrick. Some speculation about artificial intelligence and legal reasoning. *Stan. L. Rev.*, 23:40, 1970.

[15] C. Cardellino, M. Teruel, L. Alemany, and S. Villata. Legal nerc with ontologies, wikipedia and curriculum learning. In *15th European Chapter of the Association for Computational Linguistics (EACL 2017)*, pages 254–259, 2017.

[16] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

[17] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos. Legal-bert: The muppets straight out of law school. *arXiv preprint arXiv:2010.02559*, 2020.

[18] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos. LEGAL-BERT: The muppets straight out of law school. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 2898–2904, Online, Nov. 2020. Association for Computational Linguistics.

[19] I. Chalkidis, M. Fergadiotis, P. Malakasiotis, and I. Androutsopoulos. Large-scale multi-label text classification on eu legislation. *arXiv preprint arXiv:1906.02192*, 2019.

[20] I. Chalkidis and D. Kampas. Deep learning in law: early adaptation and legal word embeddings trained on large corpora. *Artificial Intelligence and Law*, 27(2):171–198, 2019.

[21] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

[22] I. Dagan, O. Glickman, and B. Magnini. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer, 2005.

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

[24] W. B. Dolan, C. Quirk, and C. Brockett. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, pages 350–356, 2004.

[25] X. Duan, B. Wang, Z. Wang, W. Ma, Y. Cui, D. Wu, S. Wang, T. Liu, T. Huo, Z. Hu, et al. Cjrc: A reliable human-annotated benchmark dataset for chinese judicial reading comprehension. In *China National Conference on Chinese Computational Linguistics*, pages 439–451. Springer, 2019.

[26] D. Giampiccolo, B. Magnini, I. Dagan, and B. Dolan. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9. Association for Computational Linguistics, 2007.

[27] A. Holtzman, J. Buys, M. Forbes, A. Bosselut, D. Golub, and Y. Choi. Learning to write with cooperative discriminators. *arXiv preprint arXiv:1805.06087*, 2018.

[28] M. Honnibal and M. Johnson. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1373–1378, 2015.

[29] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[30] S. Iyer, N. Dandekar, and K. Csernai. First quora dataset release: Question pairs, 2017.

[31] R. Juliano, G. Randy, K. Yoshinobu, K. Mi-Young, Y. Masaharu, and S. Ken. Summary of the competition on legal information extraction/entailment (coliee) 2021. *Proceedings of the COLIEE Workshop in ICAIL*, 2021.

[32] Y. Kano, M.-Y. Kim, M. Yoshioka, Y. Lu, J. Rabelo, N. Kiyota, R. Goebel, and K. Satoh. Coliee-2018: Evaluation of the competition on legal information extraction and entailment. In *JSAI International Symposium on Artificial Intelligence*, pages 177–192. Springer, 2018.

[33] P. M. Kien, H.-T. Nguyen, N. X. Bach, V. Tran, M. Le Nguyen, and T. M. Phuong. Answering legal questions by learning neural attentive text representation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 988–998, 2020.

[34] T. Kudo and J. Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*, 2018.

[35] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*, pages 319–345. Springer, 1999.

[36] E. Leitner, G. Rehm, and J. Moreno-Schneider. A dataset of german legal documents for named entity recognition. *arXiv preprint arXiv:2003.13016*, 2020.

[37] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

[38] N. F. Liu, M. Gardner, Y. Belinkov, M. E. Peters, and N. A. Smith. Linguistic knowledge and transferability of contextual representations. *arXiv preprint arXiv:1903.08855*, 2019.

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

[40] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[41] W. Morris. *American heritage dictionary of the English language*. American Heritage Pub. Co., 1969.

[42] H. Nguyen, V. Tran, and L. Nguyen. A deep learning approach for statute law entailment task in coliee-2019. *Proceedings of the 6th Competition on Legal Information Extraction/Entailment. COLIEE*, 2019.

[43] H. T. Nguyen, M. Q. Bui, M. C. Nguyen, T. T. Le, M. P. Nguyen, T. B. Dang, T. H. Y. Vuong, R. Teeradaj, L. M. Nguyen, D. V. Tran, V. A. Phan, T. S. Nguyen, T. H. Nguyen, B.-i. Bhumindr, V. Peerapon, and B. Prachya. A summary of the alqac 2021 competition. *In Proceedings of the 13th International Conference on Knowledge and Systems Engineering (KSE)*, 2021.

[44] H.-T. Nguyen and L.-M. Nguyen. Swarm filter-a simple deep learning component inspired by swarm concept. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 1541–1545. IEEE, 2019.

[45] H.-T. Nguyen, P. M. Nguyen, T.-H.-Y. Vuong, Q. M. Bui, C. M. Nguyen, B. T. Dang, V. Tran, M. L. Nguyen, and K. Satoh. Jnlp team: Deep learning approaches for legal processing tasks in coliee 2021. *arXiv preprint arXiv:2106.13405*, 2021.

[46] H.-T. Nguyen, V. Tran, P. M. Nguyen, T.-H.-Y. Vuong, Q. M. Bui, C. M. Nguyen, B. T. Dang, M. L. Nguyen, and K. Satoh. Paralaw nets–cross-lingual sentence-level pretraining for legal text processing. *arXiv preprint arXiv:2106.13403*, 2021.

[47] H. T. Nguyen, T. K. Vu, T. Racharak, L. M. Nguyen, and S. Tojo. Knowledge injection to neural networks with progressive learning strategy. In *Agents and Artificial Intelligence: 12th International Conference, ICAART 2020, Valletta, Malta, February 22–24, 2020, Revised Selected Papers 12*, pages 280–290. Springer International Publishing, 2021.

[48] H.-T. Nguyen, H.-Y. T. Vuong, P. M. Nguyen, B. T. Dang, Q. M. Bui, S. T. Vu, C. M. Nguyen, V. Tran, K. Satoh, and M. L. Nguyen. Jnlp team: Deep learning for legal processing in coliee 2020. *arXiv preprint arXiv:2011.08071*, 2020.

[49] T.-S. Nguyen, L.-M. Nguyen, S. Tojo, K. Satoh, and A. Shimazu. Recurrent neural network-based models for recognizing requisite and effectuation parts in legal texts. *Artificial Intelligence and Law*, 26(2):169–199, 2018.

[50] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, Oct. 2014. Association for Computational Linguistics.

[51] M. O. Pflueger, I. Franke, M. Graf, and H. Hachtel. Predicting general criminal recidivism in mentally disordered offenders using a random forest approach. *BMC psychiatry*, 15(1):1–10, 2015.

[52] J. Rabelo, M.-Y. Kim, R. Goebel, M. Yoshioka, Y. Kano, and K. Satoh. A summary of the coliee 2019 competition. In *JSAI International Symposium on Artificial Intelligence*, pages 34–49. Springer, 2019.

[53] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners, 2019.

[54] P. Rajpurkar, R. Jia, and P. Liang. Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.

[55] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[56] E. Rissland. Artificial intelligence and legal reasoning: A discussion of the field and gardner's book. *AI Magazine*, 9(3):45–45, 1988.

[57] J. Roberts. Trump, twitter, and the first amendment. *Alternative Law Journal*, 44(3):207–213, 2019.

[58] R. E. Susskind. Expert systems in law: A jurisprudential approach to artificial intelligence and legal reasoning. *The modern law review*, 49(2):168–194, 1986.

[59] V. Tran, M. Le Nguyen, S. Tojo, and K. Satoh. Encoded summarization: summarizing documents into continuous vector space for legal case retrieval. *Artificial Intelligence and Law*, 28(4):441–467, 2020.

[60] S. S. Ulmer. Quantitative analysis of judicial processes: Some practical and theoretical applications. *Law and Contemporary Problems*, 28(1):164–184, 1963.

[61] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

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

[63] S. Wehnert, V. Sudhi, S. Dureja, L. Kutty, S. Shahania, and E. W. De Luca. Legal norm retrieval with variations of the bert model combined with tf-idf vectorization. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law*, pages 285–294, 2021.

[64] J. Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.

[65] A. Williams, N. Nangia, and S. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics, 2018.

[66] A. Williams, N. Nangia, and S. R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

[67] T. Wolf, V. Sanh, J. Chaumond, and C. Delangue. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*, 2019.

[68] C. Xiao, H. Zhong, Z. Guo, C. Tu, Z. Liu, M. Sun, Y. Feng, X. Han, Z. Hu, H. Wang, et al. Cail2018: A large-scale legal dataset for judgment prediction. *arXiv preprint arXiv:1807.02478*, 2018.

[69] H. Ye, X. Jiang, Z. Luo, and W. Chao. Interpretable charge predictions for criminal cases: Learning to generate court views from fact descriptions. *arXiv preprint arXiv:1802.08504*, 2018.

[70] Z. Zhang, Y. Wu, J. Zhou, S. Duan, H. Zhao, and R. Wang. SG-Net: Syntax guided transformer for language representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[71] H. Zhong, C. Xiao, C. Tu, T. Zhang, Z. Liu, and M. Sun. How does nlp benefit legal system: A summary of legal artificial intelligence. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5218–5230, 2020.

[72] H. Zhong, C. Xiao, C. Tu, T. Zhang, Z. Liu, and M. Sun. Jec-qa: A legal-domain question answering dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9701–9708, 2020.

# Related Publications

[1] Nguyen Ha Thanh, Nguyen Minh Phuong, Vuong Thi Hai Yen, Bui Minh Quan, Nguyen Minh Chau, Dang Tran Binh, Vu Tran, Nguyen Le Minh, Ken Satoh (2022). *Transformer-based Approaches for Legal Text Processing* The Review of Socionetwork Strategies. `https://doi.org/10.1007/s12626-022-00102-2`.

[2] Nguyen Ha Thanh, Phi Manh Kien, Ngo Xuan Bach, Vu Tran, Nguyen Le Minh, Tu Minh Phuong (2021). *Attentive Deep Neural Networks for Legal Document Retrieval* Artificial Intelligence and Law. Springer. *(Submitted).*

[3] Nguyen Ha Thanh, Vu Trung Kien, Teeradaj Racharak, Nguyen Le Minh, Satoshi Tojo (2020). *Knowledge Injection to Neural Networks with Progressive Learning Strategy* In Agents and Artificial Intelligence: 12th International Conference, ICAART 2020, Valletta, Malta, February 22-24, 2020, Revised Selected Papers (Vol. 12613, p. 280). Springer Nature.

[4] Nguyen Ha Thanh, Nguyen Le Minh, Shirai Kiyoaki (2021). *Few-shot Tuning Framework for Automated Terms of Service Generation* In Proceedings of the 34th International Conference on Legal Knowledge and Information Systems.

[5] Nguyen Ha Thanh, Dang Tran Binh, Bui Minh Quan, Nguyen Le Minh (2021). *Evaluate and Visualize Legal Embeddings for Explanation Purpose* In Proceedings of the 13th International Conference on Knowledge and Systems Engineering (KSE).

[6] Phi Manh Kien, Nguyen Ha Thanh, Ngo Xuan Bach, Vu Tran, Nguyen Le Minh, Tu Minh Phuong (2020, December). *Answering Legal Questions by Learning Neural Attentive Text Representation.* In Proceedings of the 28th International Conference on Computational Linguistics (pp. 988-998). *The first two authors have equal contribution.*

[7] Nguyen Ha Thanh, Tran Duc Vu, Dang Tran Binh, Bui Minh Quan, Nguyen Minh Phuong, Nguyen Le Minh (2021). *HYDRA - Hyper Dependency Representation Attentions* In Proceedings of the 5th International Workshop on SCIentific DOCument Analysis (SCIDOCA2021). *The first two authors have equal contribution.*

[8] Nguyen Ha Thanh, Nguyen Minh Phuong, Vuong Thi Hai Yen, Bui Minh Quan, Nguyen Minh Chau, Dang Tran Binh, Tran Duc Vu, Nguyen Le Minh, Ken Satoh (2021). *JNLP Team: Deep Learning for Legal Processing in COLIEE 2021.* In Proceedings of the 8th Competition on Legal Information Extraction/Entailment 2021.

[9] Nguyen Ha Thanh, Vu Tran, Nguyen Minh Phuong, Vuong Thi Hai Yen, Bui Minh Quan, Nguyen Minh Chau, Dang Tran Binh, Nguyen Le Minh, Ken Satoh (2021). *ParaLaw Nets - Cross-lingual Sentence-level Pretraining for Legal Text Processing* In Proceedings of the 8th Competition on Legal Information Extraction/Entailment 2021. (Best system Task 5).

[10] Nguyen Ha Thanh, Vuong Thi Hai Yen, Nguyen Minh Phuong, Dang Tran Binh, Bui Minh Quan, Vu Trong Sinh, Nguyen Minh Chau, Vu Tran, Ken Satoh, Nguyen Le Minh (2020). *JNLP Team: Deep Learning for Legal Processing in COLIEE 2020.* In Proceedings of the International Workshop on Juris-Informatics 2020 (JURISIN 2020). (Best system Task 4).

[11] Nguyen Ha Thanh, Vu Tran, Nguyen Le Minh (2019). *A deep learning approach for statute law entailment task in COLIEE-2019.* Proceedings of the 6th Competition on Legal Information Extraction/Entailment. COLIEE.

# Other Publications

[1] Nguyen Ha Thanh, Bui Minh Quan, Nguyen Minh Chau, Le Thanh Tung, Nguyen Minh Phuong, Dang Tran Binh, Vuong Thi Hai Yen, Teeradaj Racharak, Nguyen Le Minh, Tran Duc Vu, Phan Viet Anh, Nguyen Truong Son, Nguyen Tien Huy, Bhumindr Butr-indr, Peerapon Vateekul, Prachya Boonkwan (2021). *A Summary of the ALQAC 2021 Competition* In Proceedings of the 13th International Conference on Knowledge and Systems Engineering (KSE).

[2] Bui Minh Quan, Tran Duc Vu, Nguyen Ha Thanh, Dang Tran Binh, Nguyen Le Minh (2021). *How Curriculum Learning Performs on AMR Parsing* In Proceedings of the 13th International Conference on Knowledge and Systems Engineering (KSE). *All authors contributed equally.*

[3] Bui Minh Quan, Tran Duc Vu, Nguyen Ha Thanh, Nguyen Le Minh (2020). *How State-Of-The-Art Models Can Deal With Long-Form Question Answering.* In Proceedings of the 34th Pacific Asia Conference on Language, Information and Computation (pp. 375-382).

[4] Dang Tran Binh, Nguyen Ha Thanh, Nguyen Le Minh (2020). *Latent Topic Refinement based on Distance Metric Learning and Semantics-assisted Nonnegative Matrix Factorization.* In Proceedings of the 34th Pacific Asia Conference on Language, Information and Computation (pp. 70-75).

[5] Nguyen Ha-Thanh & Nguyen, Le-Minh. (2019, November). *Swarm Filter-A Simple Deep Learning Component Inspired by Swarm Concept.* In 2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI) (pp. 1541-1545). IEEE.

[6] Nguyen Ha-Thanh, Dang Tran Binh, Nguyen Le Minh, (2019, October). *Deep Learning Approach for Vietnamese Consonant Misspell Correction.* In International Conference of the Pacific Association for Computational Linguistics (pp. 497-504). Springer, Singapore.

# Awards

- Research Grants for JAIST Students, 2019.
- COLIEE 2019 Winning Group: the best performance on the Legal Case Retrieval Task of the Competition on Legal Information Extraction/Entailment.
- COLIEE 2020 Winning Group: the best performance on the Legal Case Entailment Task and Statute Law Textual Entailment Task of the Competition on Legal Information Extraction/Entailment.
- COLIEE 2021 Winning Group: the best performance on the Statute Law Question Answering Task of the Competition on Legal Information Extraction/Entailment.
- KSE 2021: Runner-up student paper award.