A Study of Deep Learning for Legal Question Answering Systems

Tran Duc Vu

School of Information Science
Japan Advanced Institute of Science and Technology
September, 2016

Master's Thesis

A Study of Deep Learning for Legal Question Answering Systems

1410210 Tran Duc Vu

Supervisor	:	Associate Professor Minh Le Nguyen
Main Examiner	:	Associate Professor Minh Le Nguyen
Examiners	:	Professor Satoshi Tojo
	:	Associate Professor Kiyoaki Shirai

School of Information Science
Japan Advanced Institute of Science and Technology
August, 2016

Abstract

Question answering systems have been developed with the goal to produce high quality and human-like answers. Especially, question answering systems on legal domain are really practical. However, building legal question answering systems with good quality is quite a hard challenge. Not only reading and understanding legal terms requires expertise knowledge, but also legal language is at different level of complication.

We aim to build an appropriate deep learning model for legal question answering systems and particularly target the document re-ranking phase of the systems. With deep learning is emerging as a powerful machine learning approach, we evaluate several deep learning architectures for modeling text into computational vector space so as to compare them with similarity measures.

While there are numbers of deep learning architectures, we are interested in Convolutional Neural Networks for the task. The models have strong performance in Computer Vision, especially object recognition. Models based on Convolutional Neural Networks are also implemented for Natural Language Processing and have been gaining more improvement so as more attention, though, it's probably quite strange that the models treat a text as an image. Despite of the nature of deep learning models, which require no manual feature extraction, we investigate the capability of deep learning models with additional features for further improvement.

We evaluate the models on three question answering corpora with various characteristics. One of them is on open domain and the other two are on legal domain. Though they are all question answering related tasks, the actual requirements are quite distinguishable. Interestingly, one of our experimental corpora is limited in data size which is a theoretically and practically huge disadvantage for deep learning models which are data-oriented and hungry for observable instances. We then use a strong feature that can be extracted from the corpus to enrich the models. While combining additional features is not new, we experiment in a different way. We set the additional features as starting point of learning for the models. The experiment results show that deep learning models with appropriate configuration can perform well on legal domain data.

In the future, we'd like to further investigate the structural information of complicated texts like legal documents and building a deep architecture which is capable of encoding structural information into its parameters for more robust document modeling.

Keywords: Deep Learning, Question Answering, Legal Domain, Document Re-ranking

Acknowledgments

To the people I owe so much, I would like to express my sincerest gratefulness.

I would like to convey my faithful gratitude to my supervisor, Associate Professor Minh Le Nguyen, for his guidance and trust. I've been learning from him a lot, not only for my research methodology but also for developing creative ideas.

I would like to thank Professor Satoshi Tojo and Professor Kiyoaki Shirai for their constructive questions and suggestions for the improvement of my research.

I would like to thank Professor Mineo Kaneko, my minor research project advisor. The discussions with him widens my research point of views and inspire me with a handful of ideas.

I am grateful to Dr. Cuong Pham who showed me the science world and encourages me to become a scientist.

I am glad that I have been working with my excellent colleagues. Their innovative discussions and dedicated collaboration are for my inspiration.

I would like to thank LAC VIET Computing Corporation for their financial support and JAIST for providing us fantastic research condition and infrastructure during the time I'm doing this thesis.

Thank you, my friends who are always at my side and cheer me up.

I want to send my love to my father, my mother and my sister as their love for me.

Contents

Abstract i

Acknowledgments ii

1 Introduction 1
- 1.1 Research Aim . . . 1
- 1.2 Motivation . . . 1
- 1.3 Related Work . . . 2
- 1.4 Significance . . . 2
- 1.5 Contributions . . . 3
- 1.6 Thesis Outline . . . 3

2 Review of Literature 4
- 2.1 Legal Question Answering (QA) Systems . . . 4
- 2.2 Deep learning . . . 5
  - 2.2.1 Artificial Neural Networks . . . 5
  - 2.2.2 Convolutional Neural Networks (CNNs) . . . 12
  - 2.2.3 Pooling Layers . . . 13
- 2.3 Deep Learning for Modeling Texts . . . 14
  - 2.3.1 `word2vec`: Distributed Word Representations . . . 14
  - 2.3.2 Paragraph Vectors (PVs): Distributed Representations of Sentences and Documents . . . 17
  - 2.3.3 Deep Structured Semantic Models with Convolutional-Pooling Structure (CDSSMs) . . . 17
  - 2.3.4 Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks (L2RSTP-CNN) . . . 19

3 Method 21
- 3.1 Enriched Deep Learning Models with External Features . . . 21
- 3.2 Corpora . . . 26
  - 3.2.1 Corpora Overview . . . 26
  - 3.2.2 Corpora Analysis . . . 30
- 3.3 Experiments . . . 35
  - 3.3.1 Data Pre-processing . . . 35
  - 3.3.2 Experimental Models . . . 36
  - 3.3.3 Evaluation . . . 38
  - 3.3.4 Experimental Results . . . 39

4 Discussion **41**
4.1 Effectiveness 41
4.2 Limitations 45

5 Conclusions and Future Work **49**
5.1 Conclusions 49
5.2 Future Work 49

References **50**

Publications **54**

List of Figures

2.1 A Question Answering (QA) system. . . . 4
2.2 Document retrieval and re-ranking phases in a Question Answering (QA) system. . . . 5
2.3 An artificial neuron with n inputs. . . . 6
2.4 Sigmoid function. . . . 7
2.5 Hyperbolic tangent function. . . . 7
2.6 Rectifier function. . . . 8
2.7 Softplus function. . . . 8
2.8 Linear function. . . . 8
2.9 Softmax activation from left to right. . . . 9
2.10 An artificial neural layer with m artificial neurons. . . . 9
2.11 An artificial neural network with 2 layers. . . . 10
2.12 Gradient descent. . . . 12
2.13 Convolutional operations with filter size of 3×4, 2 feature maps, and stride of 1×0. If a region is highly similar to one feature map, its convoluted value (output) is high and vice versa. The operations result in two vectors with six convoluted values for each feature map. . . . 13
2.14 Max pooling operation. . . . 14
2.15 A `word2vec` model trained by skip-gram. . . . 15
2.16 Two-dimensional PCA projection of the 1000-dimensional Skip-gram vectors of countries and their capital cities. [1] . . . 16
2.17 Paragraph Vector (PV) model architecture. . . . 17
2.18 Deep Structured Semantic Model with Convolutional-Pooling Structure (CDSSM) architecture. . . . 18
2.19 Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks (L2RSTP-CNN) architecture. . . . 19

3.1 Enriched re-ranking model with external features. . . . 22
3.2 Composing word-level features from word embedding matrix $W_{embedding}$. 23
3.3 Convolutional-pooling operations with multiple-size feature maps. To capture context windows with the number of words less than the feature map sizes, we add dummy word represented by 0 vectors to the beginning and the end of the texts. . . . 24
3.4 Affine with non-linear transformation of convolutional-pooling features into latent features. . . . 25
3.5 Concatenation of latent features and external features. . . . 26
3.6 An example of a question and its relevant article in COLIEE 2015 dataset. 27
3.7 TREC 2011 Legal Track data examples. . . . 28

3.8 An example in WikiQA dataset. The last answer is the correct one out of all candidates. 29
3.9 The civil code's data structure. 30
3.10 Word count statistics for COLIEE 2015 corpus. 30
3.11 Term overlap rate of retrieved articles in top 1 on COLIEE 2015 dataset. . . 31
3.12 Term overlap rate of relevant articles not in retrieved top 10 on COLIEE 2015 dataset. 31
3.13 Recall rate of selecting top K documents from document retrieval K=[10,100] 32
3.14 Word count statistics of document candidates for TREC 2011 Legal Track corpus. 33
3.15 Word overlap distribution over document candidates for TREC 2011 Legal Track corpus. 33
3.16 Word count statistics for WikiQA corpus. 34
3.17 Word overlap distribution over answer candidates for WikiQA corpus. . 34
3.18 Data pre-processing flow. 35
3.19 Data pre-processing example. 36
3.20 Diagram of making training data from a corpus. 38

4.1 The article ranking of query H25-22-2 by our model and Term frequency-inverse document frequency (TF-IDF). Bold texts are regions of interest our model extracted with convolutional-pooling operations. 43
4.2 Example of a query and its relevant articles referring to other articles. . 46
4.3 Example of a query and its relevant article describing the general case with high level of abstraction. 47
4.4 Example of a query and its relevant article which composed in "requisite-effectuation" logical structures. 47
4.5 Deep learning architecture for structural input 48

List of Tables

2.1 Word clusters from training a `word2vec` model. 17

3.1 Article distribution: the number of articles in the Civil Code that occur in training set, test set and co-occur in both; and the number of relevant articles for each question. 32
3.2 Statistics of the WikiQA dataset [2]. 35
3.3 Performances are measured based on precision (Pre), recall (Rec), F-score (F_1), and mean average precision (MAP) calculated on positive instances (relevant or responsive documents) only. 40
3.4 Number of questions having relevant articles at top T=1 in COLIEE 2015 dataset. 40

4.1 Relevant article ranking of our approach compared to TF-IDF. Most of the cases, as the TF-IDF model gets the top 1 correct, our model gets the top 2. 42
4.2 Relevant article ranking of our approach compared to PV. 44
4.3 Relevant article ranking of our approach compared to L2RSTP-CNN . 44
4.4 Relevant article ranking of our approach compared to CDSSM. 45

Abbreviations

CDSSM Deep Structured Semantic Model with Convolutional-Pooling Structure. iii, v, vii, 17, 18, 37, 40, 41, 45, 49

CNN Convolutional Neural Network. iii, 2, 12, 14

L2RSTP-CNN Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. iii, v, vii, 19, 26, 37, 40, 41, 44

LSTM Long Short-Term Memory. 2

NLP Natural Language Processing. 1, 2, 4, 12

PV Paragraph Vector. iii, v, vii, 17, 37, 40, 44

QA Question Answering. iii, v, 1, 2, 4, 5

RNN Recurrent Neural Network. 2

TF-IDF Term frequency-inverse document frequency. vi, vii, 21, 25, 37, 38, 40, 42, 43, 49

Chapter 1

Introduction

1.1 Research Aim

This research aims to adapt deep learning method for QA systems especially on legal domain. For that, we set forth our study to find answers for the following questions. First, can we apply deep learning for legal question answering systems? We compare several developed deep learning models on designated legal question answering corpora. Second, can we improve legal question answering systems with deep learning? We investigate a way to enrich deep learning model with external features.

1.2 Motivation

QA is a research field of information retrieval and Natural Language Processing (NLP) and focuses on building systems that response user questions with answers instead of long list of relevant documents. In other words, a QA system stands between a normal search engine and users. The system analyzes result from the search engine, extract core information to response the users.

Legal is one of challenging domains on which QA systems can be developed. Legal documents are far more complicated than daily newspapers in the way that most of them require expert knowledge to understand correctly. Searching for an answer of a legal question, however, is very important yet not quite easy for common users. A well designed legal QA system not only saves time and effort but also supports dealing with technical difficulty of legal text.

Deep learning is a machine learning technique that aims at modeling high-level abstraction of data [3, 4] that higher level features are learned from lower level. For illustration, from pixels of an image as the first level, a deep learning model can learn vertical, horizontal, diagonal lines, or curves as the second level, then triangles, rectangles, circles as the third level, and so on. This phenomenon shows that deep learning is capable of generating very complex features from a simple input which is a major difference from costly hand crafted features required for machine learning techniques like support vector machine. Beside traditional machine learning techniques, deep learning is greatly concerned in this research because of its success in many research areas, for examples, signal processing, image processing, and also NLP.

Deep learning has been proved to be efficient in NLP. It has achieved state-of-the-art results in NLP sub-fields, such as constituency parsing [5], sentiment analysis [6], machine translation [7], and other NLP areas.

1.3 Related Work

Researches on QA tasks from the past to the present are usually linguistics analysis and deep learning. On one hand, linguistics methodology focuses on extracting linguistics features from corpus for learning a strong machine learning models such as support vector machine, adaboost. Wherein, tree-like structures, for example dependency trees, constituency trees with tree-based algorithms such as tree edit distance are very popular and efficient approaches [8, 9, 10, 11, 12, 13]. Additional approaches are lexical-based [14, 15], or word-alignment [16]. On the other hand, deep learning is an approach of automatic learning which reducing the heuristic processing of text. Despite of the deference, combinations of deep learning and linguistics are attractive.

Deep learning is recently gaining dramatically attraction in NLP, especially QA. For QA tasks, deep learning has been approached in several ways. Wang et al. use deep belief networks to model semantic relevance for community-based question-answer pairs [17]. The input of the model is a word-occurrence vector of a question or an answer. The approach focuses on its content and ignores any syntactic constraints which leads to loss of information. In the different way, Iyyer et al. [18] exploit the syntactic dependency tree of a question and apply a recursive neural network on this tree to classify a given set of answer candidates. The approach makes use of syntactic information but the architecture is very sensitive to word order and words modifying. One way that not requiring tree structures is using Recurrent Neural Network (RNN) or Long Short-Term Memory (LSTM) [19, 20]. This problem can be adapted by a Convolutional Neural Network (CNN) where related work are sentence modeling by Kalchbrenner et al. [21] for sentence classification tasks, answer selection by Yu et al. [22], vector representation similarity-based models [23, 24]. A CNN typically processes over windows of text which keeps local context and not too sensitive to word changing.

Legal, a very strict and challenging domain though, is practical for building QA systems. Quaresma and Rodrigues [25] build a QA system on legal domain by representing legal documents in logical forms. Their system exploit syntactic, semantic and legal ontology for the task to transform questions and legal clauses into discourse structures then applies logical inference to find the true answer. There are also work utilizing some characteristics of legal documents such as logical structure recognition, mining reference information, etc [26, 27, 28]. A different method is to use ranking models to assign score to each answer candidate [29], but may or may not consider logical relation between questions and answers.

1.4 Significance

Deep learning approach can be used to mitigate the lack of specialist knowledge to extract adequate features for machine learning methods such as support vector machine, adaboost. Deep learning models also have the benefits of their scalability, flexibility, adaptivity. We can change the size of the models or treat each models as specific modules to combine them into a more sophisticated architecture to use in specific problem without the hassle of feature engineering.

1.5 Contributions

We experiment variate deep learning models on designated legal question answering corpora. Then we evaluate the impact of additional features to the performance of deep learning models. We analyze the effectiveness and limitations of our method.

1.6 Thesis Outline

The remainder of this thesis is organized as follows:

- **Review of Literature** (Chapter 2): We give an overview of legal question answering systems, study-related deep learning techniques and models.
- **Method** (Chapter 3): We describe our approach to build a hybrid deep learning model and compare the model with other models described in Chapter 2. We also analyze some datasets for experiments of the models. We present our experiments. We analyze the results of the experiments.
- **Discussion** (Chapter 4): We discuss some findings from the experiments. We analyze the effectiveness and limitations of the models.
- **Conclusions** (Chapter 5): We conclude our study and state our future work direction.

Chapter 2

Review of Literature

2.1 Legal Question Answering (QA) Systems

Question Answering (QA) is a research field of information retrieval and Natural Language Processing (NLP) and focuses on building systems that response users questions with answers instead of long list of relevant documents. In other words, a QA system stands between a normal search engine and users. The system analyzes result from the search engine, extract core information to response the users. Thus, a Question Answering (QA) system often consists of several modules such as question analysis, indexed knowledge source, answer analysis, etc.

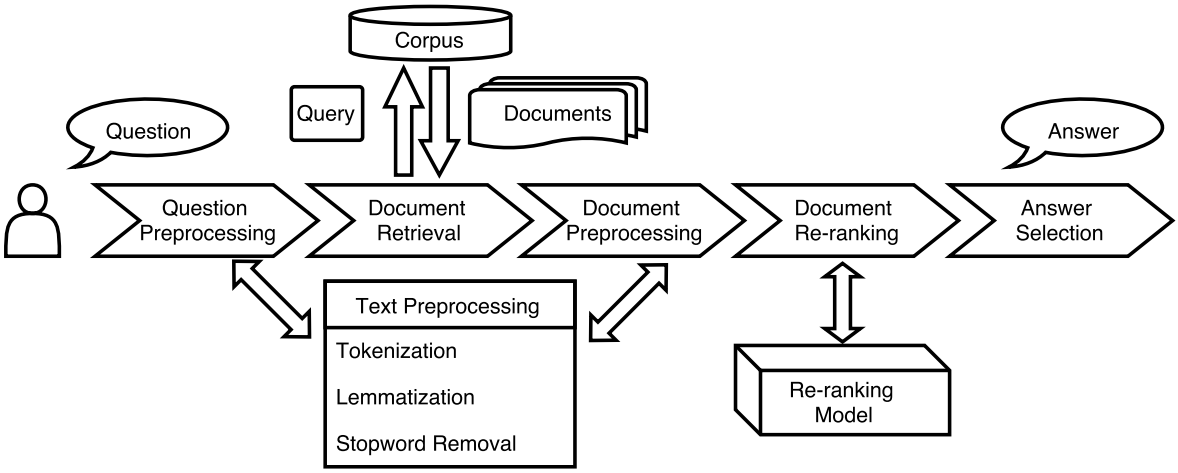

Figure 2.1: A Question Answering (QA) system.

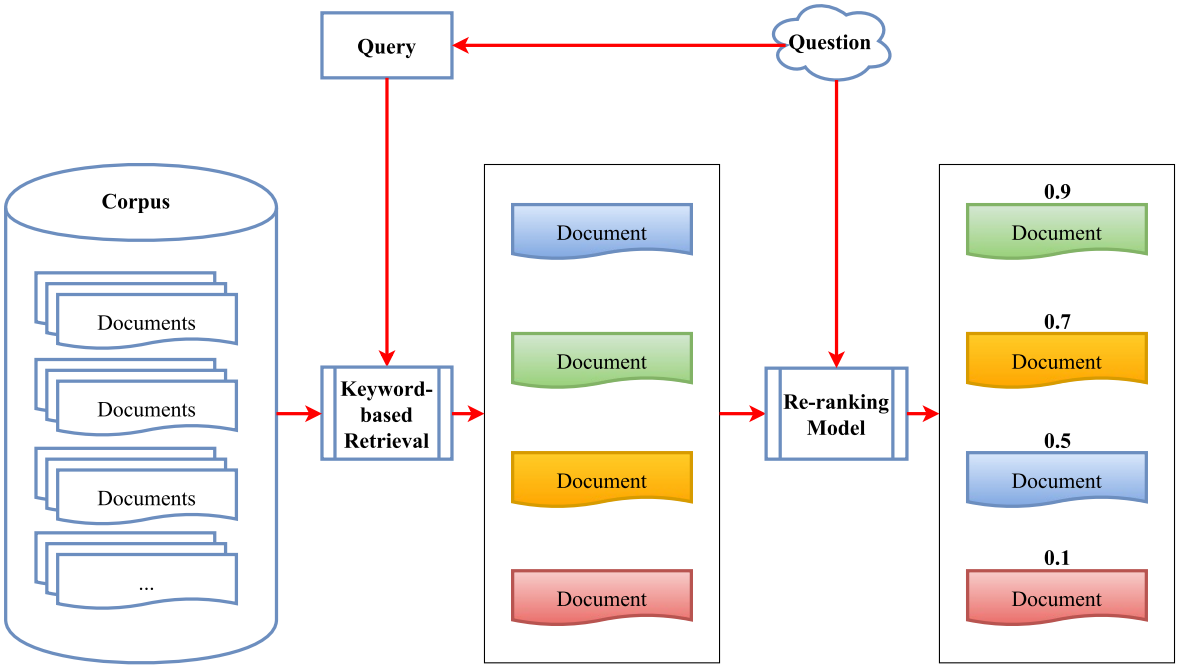

Figure 2.2: Document retrieval and re-ranking phases in a QA system.

Document re-ranking is a very important phase in a QA system. This phase usually involves building a re-ranking model. The re-ranking model receives retrieved documents from keyword-based retrieval module and the given question to compute the score of each document, then return the ranked list of documents ordered descending by the scores. The keyword-based retrieval module only applies keyword matching without the knowledge of question-document relationships. Therefore, the re-ranking model is trained to learn the relationships in order to re-ranking the documents more semantically.

Legal is one of closed domains on which QA systems can be developed. Legal documents are far more complicated than daily newspapers in the way that most of them require expert knowledge to understand correctly. Searching for an answer of a legal question, however, is very important yet not quite easy for common users. A well designed legal QA system not only saves time and effort but also supports technical difficulty of legal text.

2.2 Deep learning

Deep learning is a machine learning techniques that aims at modeling high-level abstraction of data in the form of artificial neural networks [3, 4, 30, 31, 32, 33, 34, 35].

2.2.1 Artificial Neural Networks

Artificial neural networks are composed of linear and non-linear transformations organized as multi-layer networks where each layer consists of artificial neurons.

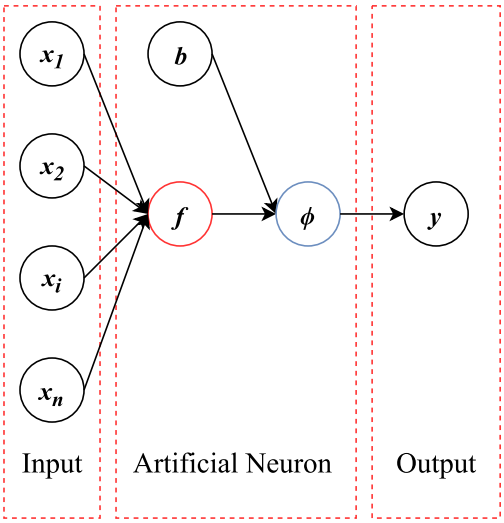

Figure 2.3: An artificial neuron with n inputs.

An artificial neuron (Figure 2.3) receives multiple inputs and emits one output signal. It comprises:

- A weighting ormapping function f with a set of weights θ
- A bias value b
- An activation or transformation function ϕ, which is often one of the following:
  - Sigmoid
  $$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{2.1}$$
  - Hyperbolic tangent
  $$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.2}$$
  - Rectifier
  $$rec(x) = \max(0, x) \tag{2.3}$$
  - Softplus
  $$softplus(x) = log(1 + e^{x}) \tag{2.4}$$
  - Linear
  $$linear(x) = x \tag{2.5}$$
  - Softmax
  $$softmax([x_1, ..., x_i, ..., x_K]) = \left[\frac{e^{x_1}}{Z}, ..., \frac{e^{x_i}}{Z}, ..., \frac{e^{x_K}}{Z}\right] \ , \ Z = \sum_{j=1}^{K} e^{x_j} \tag{2.6}$$

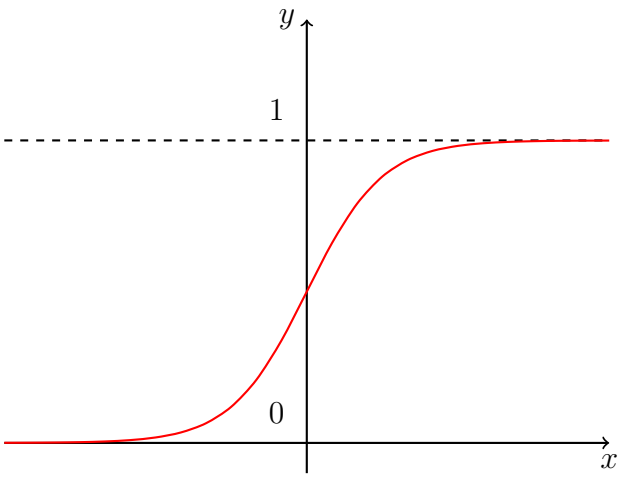

Figure 2.4: Sigmoid function.

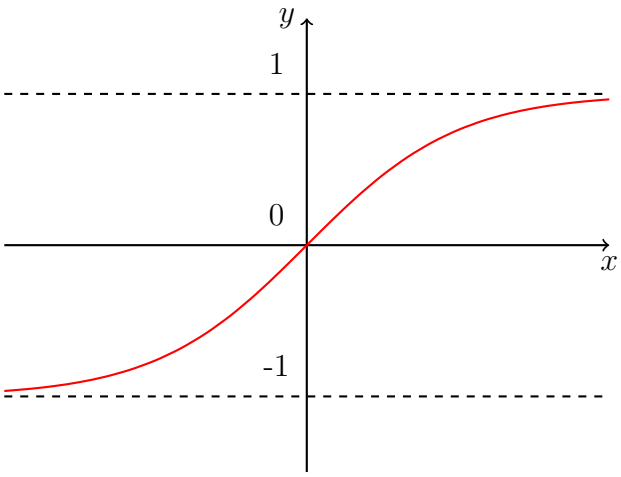

Figure 2.5: Hyperbolic tangent function.

Sigmoid and hyperbolic tangent functions are often used quite differently. On one hand, hyperbolic tangent function is usually used for non-linear transformation to output signal in positive (+) or negative (-), which represents two opposite states. On the other hand, sigmoid function is usually used for switches, gates in order to filter out some input values.

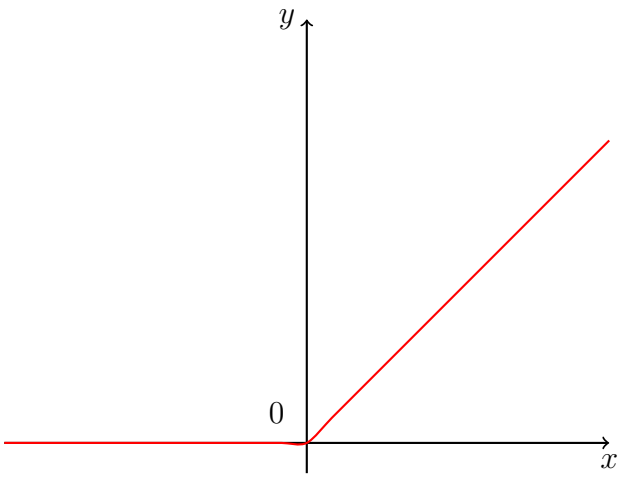

Figure 2.6: Rectifier function.

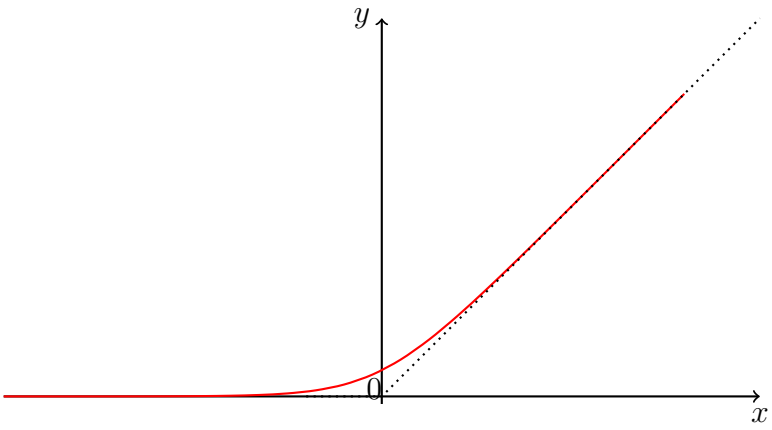

Figure 2.7: Softplus function.

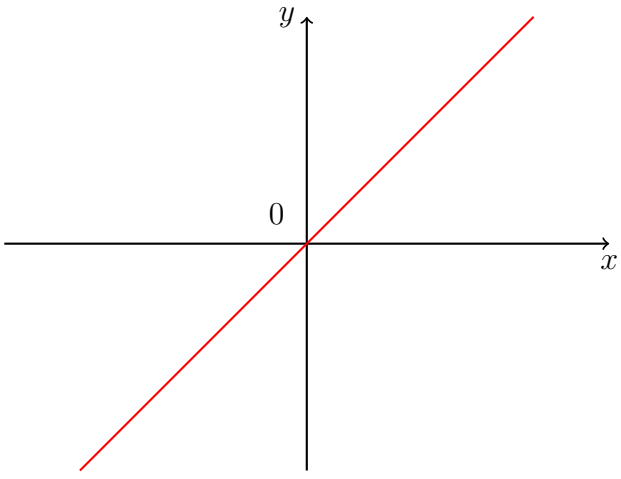

Figure 2.8: Linear function.

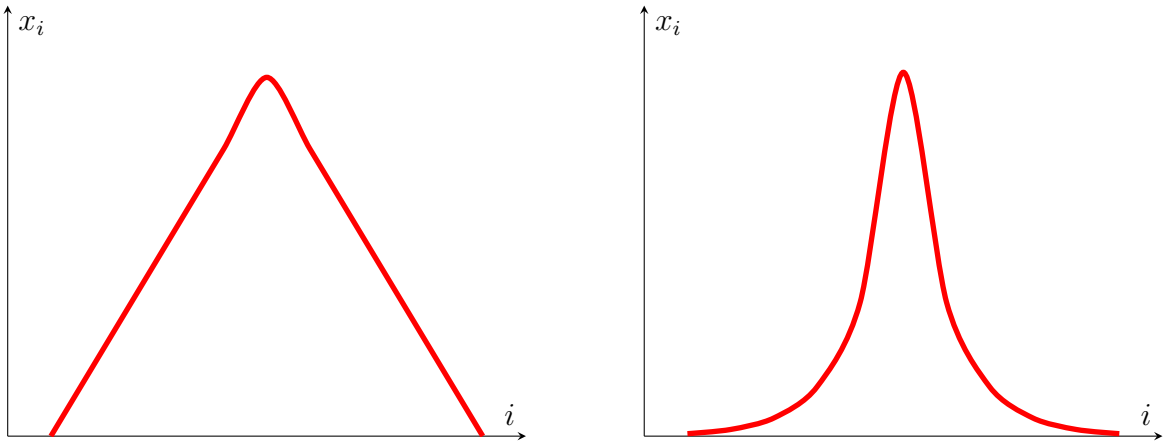

Figure 2.9: Softmax activation from left to right.

The output of the neuron is computed by:

$$y = \phi(f(\mathbf{x}) + b) \tag{2.7}$$

- $\mathbf{x} = [x_1, x_2, ..., x_n]$: input of the neuron.
- y : output of the neuron.
- f : weighting or mapping function.
- b : bias.
- ϕ : activation or transformation function.
- n : size of the input.

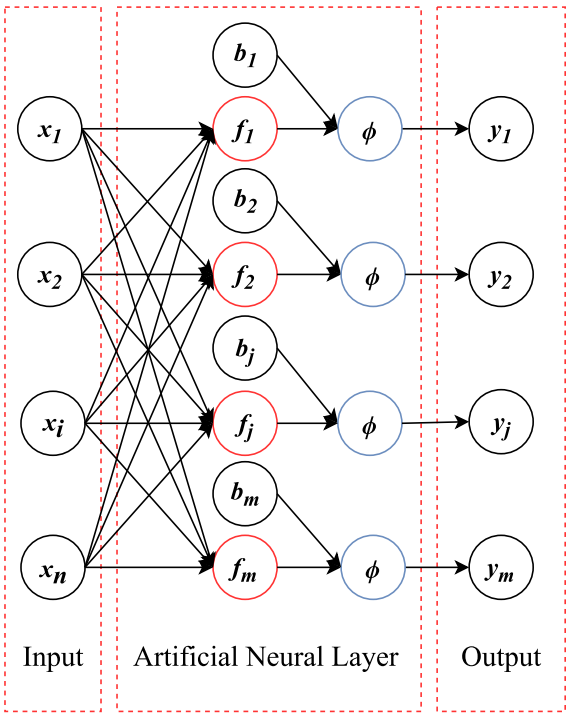

Figure 2.10: An artificial neural layer with m artificial neurons.

An artificial neural layer (Figure 2.10) consists of artificial neurons. The output of the layer is composed of the output of each neuron and computed by:

$$\mathbf{y} = \phi(\mathbf{f}(\mathbf{x}) + \mathbf{b}) \tag{2.8}$$

- $\mathbf{x} = [x_1, x_2, ..., x_n]$: input of the layer.
- $\mathbf{f} = [f_1, f_2, ..., f_m]$: weighting or mapping functions.
- $\mathbf{b} = [b_1, b_2, ..., b_m]$: biases.
- $\mathbf{y} = [y_1, y_2, ..., y_m]$: output of the layer.
- ϕ : activation or transformation function.
- n: size of input.
- m: number of neurons.

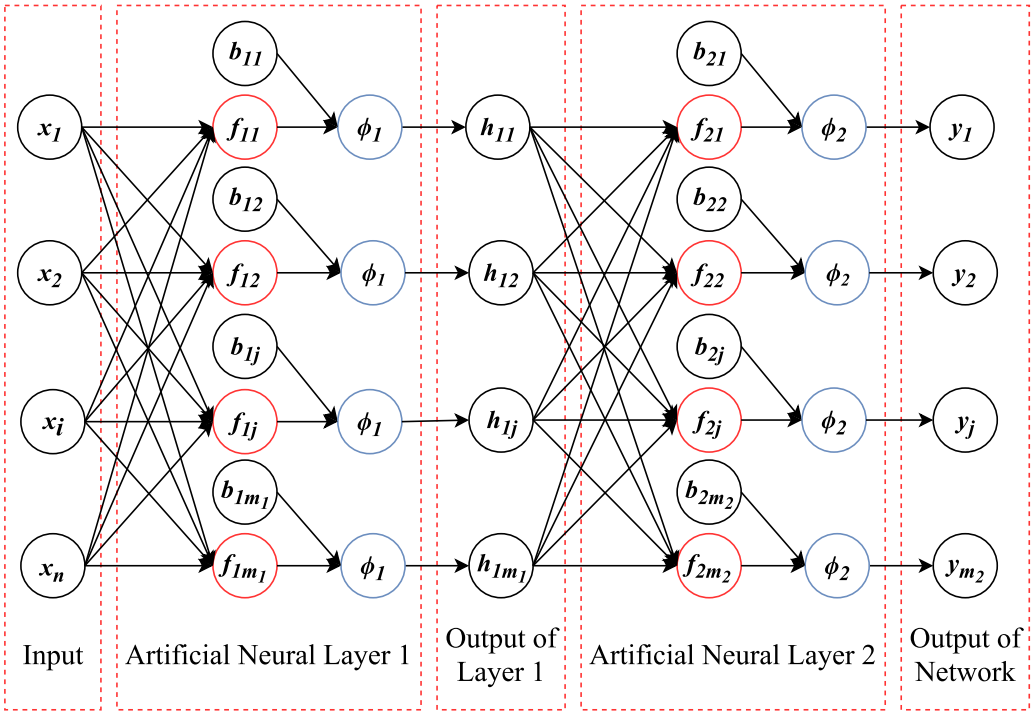

Figure 2.11: An artificial neural network with 2 layers.

An artificial neural network (Figure 2.11) is then built by stacking multiple artificial neural layers one by one. The output of one layer becomes the input for the next layer:

$$\mathbf{h}_l = \phi_l(\mathbf{f}_l(\mathbf{h}_{l-1}) + \mathbf{b}_l) \tag{2.9}$$

- l : layer l.
- $\mathbf{f_l}$: weighting or mapping functions of layer l.
- $\mathbf{b_l}$: biases of layer l.

- ϕ_l : activation or transformation function of layer l.
- $\mathbf{h}_l = [h_{l1}, h_{l2}, ..., h_{lm_l}]$: output of layer l.
- $\mathbf{h}_0 = \mathbf{x}$: input of the network.
- $\mathbf{h}_L = \mathbf{y}$: output of the network.
- m_l: number of neurons in layer l.
- L : total number of artificial neural layers.

Apparently, there's no definite solution for optimization of artificial neural networks. Gradient descent is a well-known method for estimating the weights $\boldsymbol{\theta} = [\vartheta_1 \ \vartheta_2 \ ... \ \vartheta_i \ ... \ \vartheta_T]$ resulting in approximately optimized artificial neural networks. Given an artificial neural network $G(\boldsymbol{\theta})$ which has an optimal weights $\boldsymbol{\theta}^{optimal}$, there's is a loss function:

$$L(\boldsymbol{\theta}) = Error(G(\boldsymbol{\theta}), G(\boldsymbol{\theta}^{optimal})) \tag{2.10}$$

Gradient of loss function is:

$$\nabla(L, \boldsymbol{\theta}) = \left[\begin{array}{ccccc} \frac{\delta L}{\delta \vartheta_1} & \frac{\delta L}{\delta \vartheta_2} & ... & \frac{\delta L}{\delta \vartheta_i} & ... \ \frac{\delta L}{\delta \vartheta_T} \end{array} \right] \tag{2.11}$$

While the gradient of a function refers to the incremental direction and velocity of the function, the direction of function variables to reduce the function is then inferred from the additive inverse of the gradient. The the loss function, thus, can be iteratively reduced by updating $\boldsymbol{\theta}$ in the gradient descent direction:

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \eta \nabla(L, \boldsymbol{\theta}) \tag{2.12}$$

The learning rate η defines how far the weights $\boldsymbol{\theta}$ change. If the learning rate is large, the update is fast but may jumps over the optimal point. Otherwise, if the learning rate is small, the update is slow but more probably converges to one optimal point. However, too small learning rate also may results in a local optimal point sincere most artificial neural networks are high dimensional.

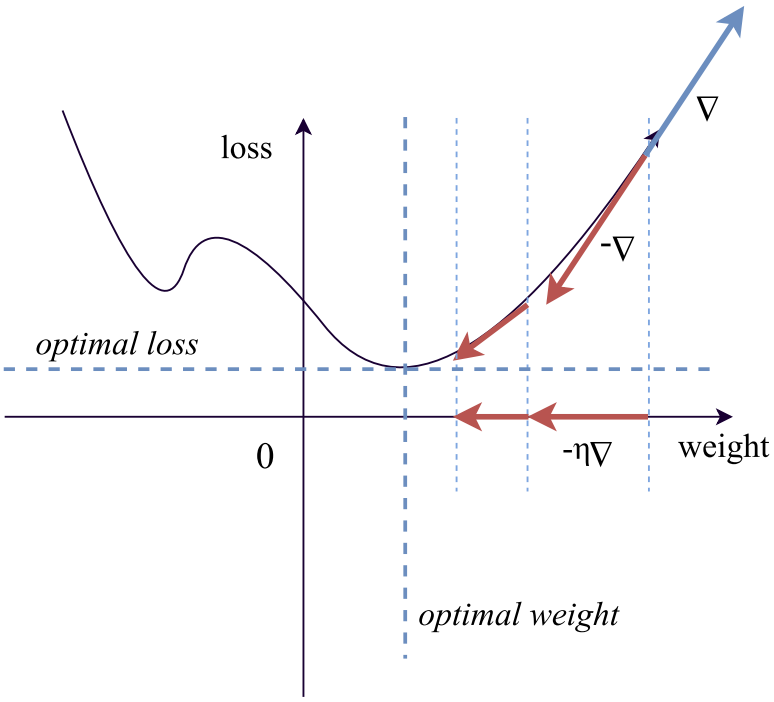

Figure 2.12: Gradient descent.

2.2.2 Convolutional Neural Networks (CNNs)

CNNs are deep neural networks that learn patterns from data with convolutions and stores them in their feature maps with the preferences of depth and breadth. CNNs have strong performances in computer vision, especially object recognition [36]. The way CNNs work is definitely the reason of that success. A CNN divides the input into sliced regions and scans through its input and applies its feature maps to find the regions of interest which are the portions of the input that most match with its defined or learned feature maps (Figure 2.13). Moreover, an object need to be recognized may locate in different regions for each image so as in NLP, important information may be some small piece of texts or contexts of interest in whole long documents.

Each convolutional operation, the main part of every CNN, is composed from the following parameters:

- Filter size: how large the feature maps of the network are. This defines the area that the needed information should fall in in usual input.
- Number of feature maps: how many possible patterns that the network should be able to capture.
- Stride: how the operations move to the next region. In Figure 2.13, stride is 1×0, which means that each time, the operations move one step vertically and no step horizontally.

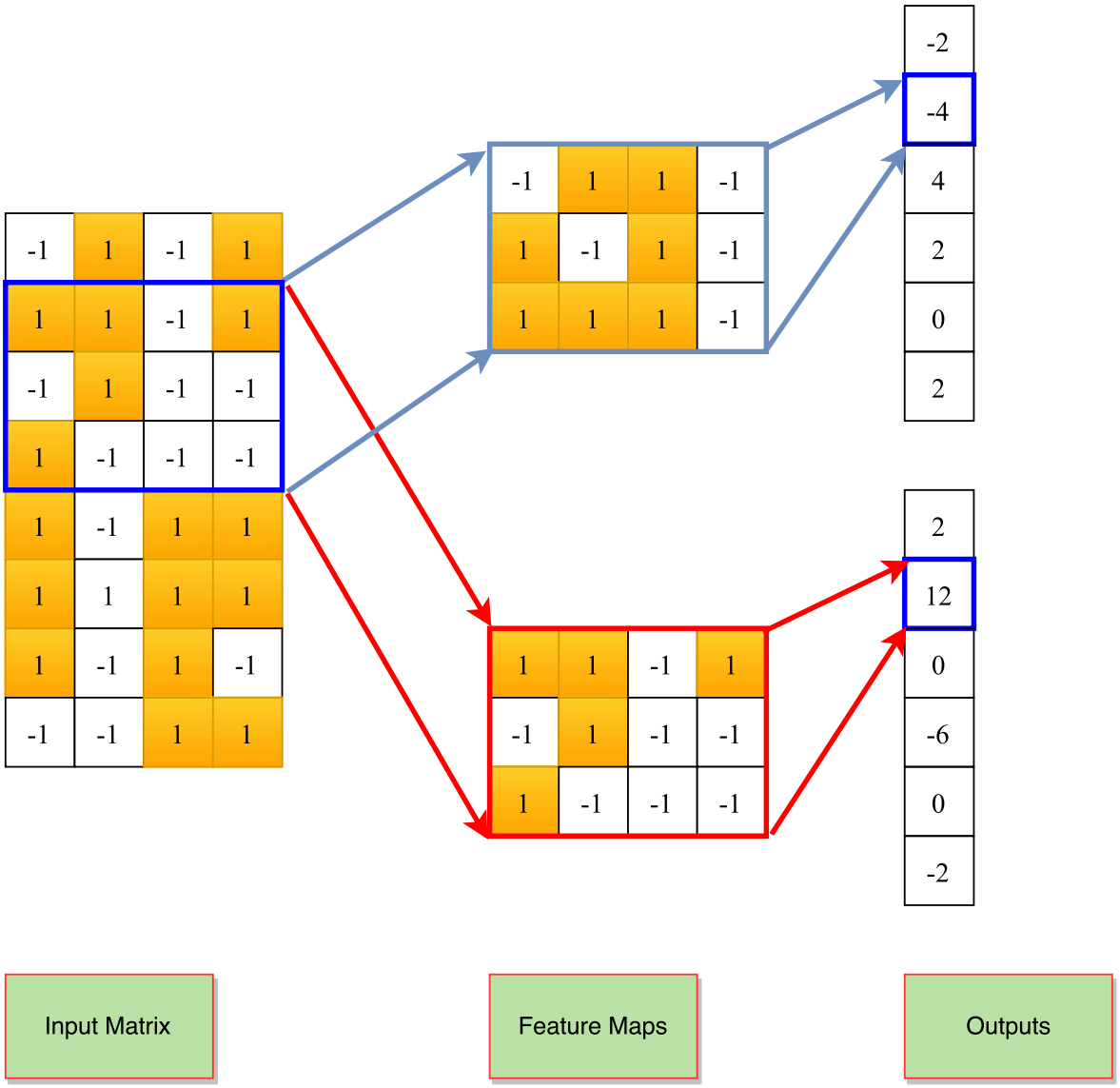

Figure 2.13: Convolutional operations with filter size of 3×4, 2 feature maps, and stride of 1×0. If a region is highly similar to one feature map, its convoluted value (output) is high and vice versa. The operations result in two vectors with six convoluted values for each feature map.

2.2.3 Pooling Layers

A pooling layer is usually an intermediate layer in a deep neural network. It often has linear activation function and bias value 0. It transforms its usually large and variant-size input into smaller and fixed-size output by applying its operation as follows.

$$h_i = \text{pooling}(h_{i-1}) \tag{2.13}$$

where the pooling operation is often one of the following.

- **max**: selecting the max value of the input.
- **average**: computing the average value of the input.

- **k-max pooling**: select the k max values of the input.
- **dynamic k-max pooling**: similar to k-max pooling with the value of k is determined by the input (for example, input size).

Pooling layers are often placed on top of convolutional layers in CNNs or the similar architectures to extract the regions of interest from input.

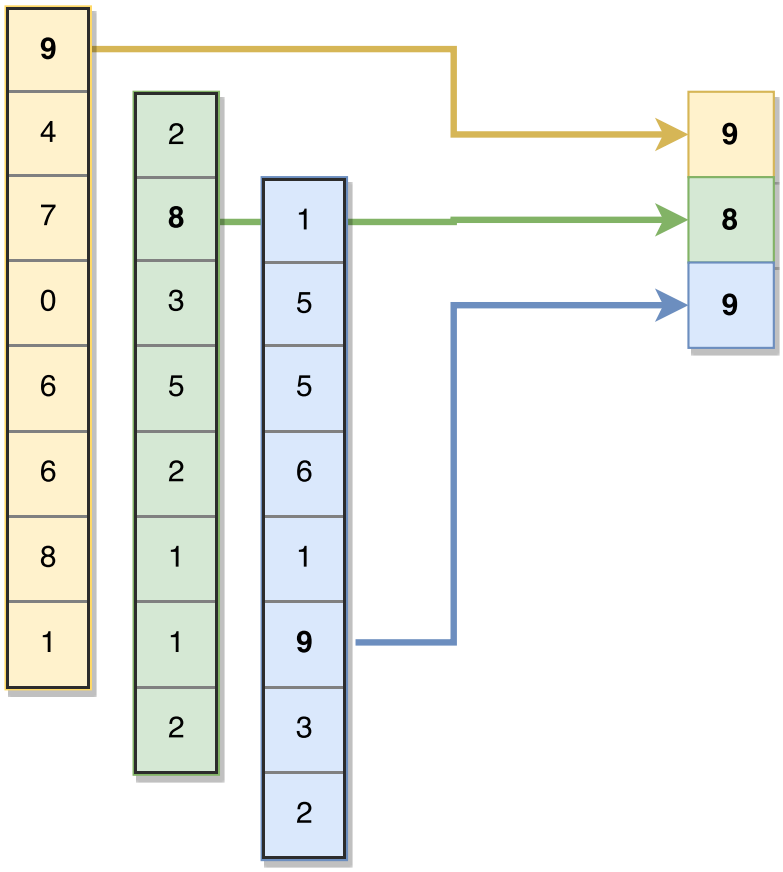

Figure 2.14: Max pooling operation.

2.3 Deep Learning for Modeling Texts

In this section, we describe several deep learning models that achieve high performances in tasks related to comparing texts and used in our experiments.

2.3.1 `word2vec`: Distributed Word Representations

A `word2vec` model encodes each words into a distributed representation, proposed by Mikolov et al. [1]. It can be trained in continuous-bag-of-word or skip-gram method. Both the methods have the same objective: a distributed representation of a word is estimated by possible surrounding context of this word. Thus, the representations hold the context distribution information of the word. In other words, if two representations are similar, their represented words have similar context usages or are used in similar situations.

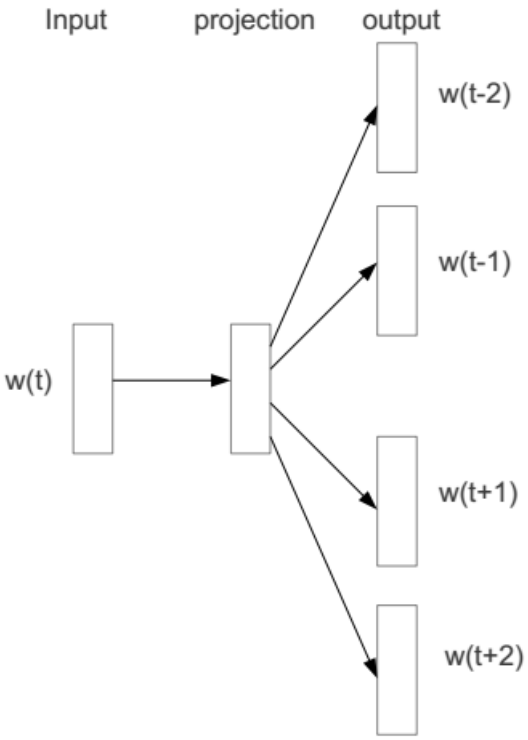

Figure 2.15: A `word2vec` model trained by skip-gram.

Skip-gram. From a corpus, a set of all text windows with size $2k + 1$ is extracted. For each text window corresponding to representation vectors w_{t-k}, ..., w_{t-1}, w_t, w_{t+1}, ..., w_{t+k}, the method estimates the representation w_t by predicting the surround representations w_{t-k}, ..., w_{t-1}, w_{t+1}, ..., w_{t+k}. This training method utilizes an artificial neural network with the following layers:

1. Input layer: receiving middle word's representation w_t
2. Projection layer: apply transformation from input vector w_t to output vector concatenated of w_{t-k}, ..., w_{t-1}, w_{t+1}, ..., w_{t+k}.

$$\text{projection}(w_t) = \begin{bmatrix} w_{t-k} \\ ... \\ w_{t-1} \\ w_{t+1} \\ ... \\ w_{t+k} \end{bmatrix} \tag{2.14}$$

3. Output layer: the predicted representations w_{t-k}, ..., w_{t-1}, w_{t+1}, ..., w_{t+k}.

As the name suggested, the model is a kind of distributional model. As the mechanism of skip-gram, by predicting the same word over different surrounding words or the variation of contexts, the model estimates the distribution of a word in all contexts and saves this into a distributed vector.

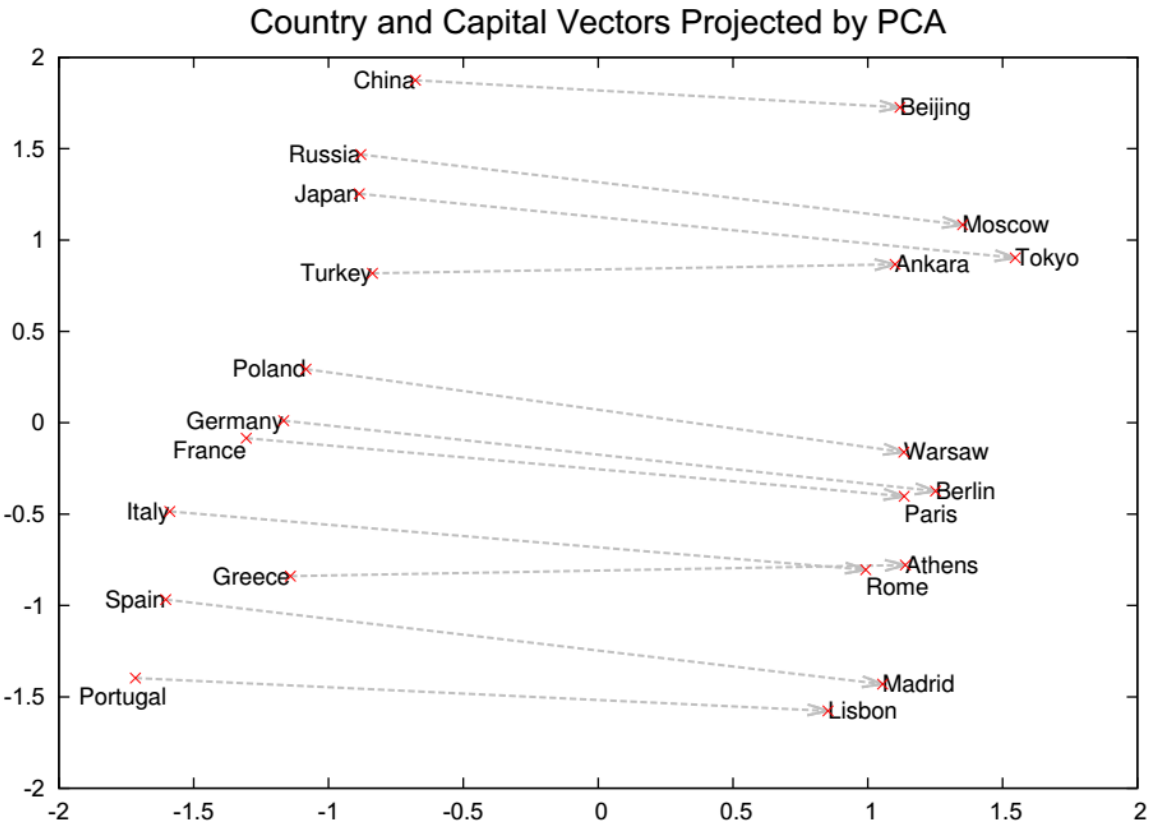

Figure 2.16: Two-dimensional PCA projection of the 1000-dimensional Skip-gram vectors of countries and their capital cities. [1]

Table 2.1: Word clusters from training a `word2vec` model.

proviso	**provision**	**obligee**	**consent**	**seller**	**buyer**
wherein	oversight	obligor	authorization	retailer	bidder
stipulation	provisions	he/she	stipulated	buyer	seller
therefor	requirement	mandatory	injunction	collector	broker
disclaiming	limits	pledgees	obligated	vendor	lender
offeror	reduction	mortgagee	stipulate	trader	purchaser
accordance	guidelines	mortgagor	stipulating	dealer	investor
offeree	protections	sub-pledge	obligation	merchant	shareholder

2.3.2 Paragraph Vectors (PVs): Distributed Representations of Sentences and Documents

This method is proposed by Le and Mikolov [37] to model variant-length text chunks into fixed-size vectors.

The training mechanism is similar to `word2vec` model's with one additional representation of the text chunk itself (Figure 2.17). For each text chunk, a set of text windows is also extracted. Each context window is now represented by these vectors D, w_{t-k}, ..., w_{t-1}, w_t, w_{t+1}, ..., w_{t+k}, where D is the additional representation of the text chunk participating in the process of predicting surrounding context as seen in `word2vec` training process.

The training has two steps. First, the distributed representations of all words in a given corpus is estimated and fixed. Second, the distributed representation D of each text chunk in the corpus is estimated.

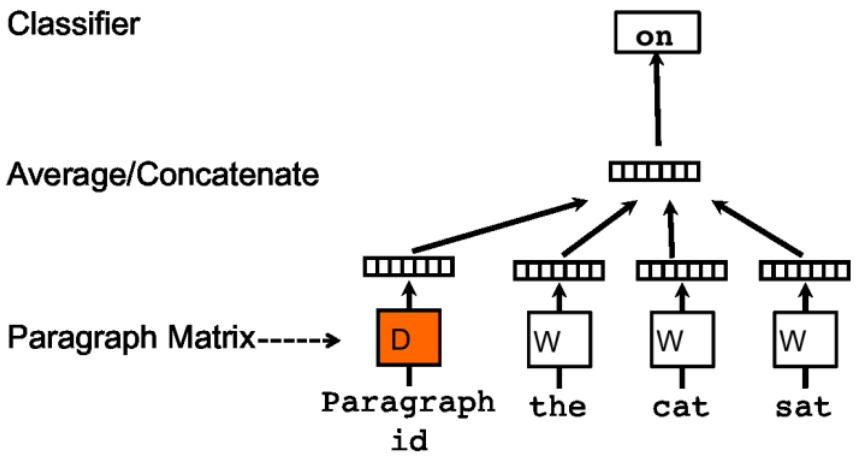

Figure 2.17: PV model architecture.

2.3.3 Deep Structured Semantic Models with Convolutional-Pooling Structure (CDSSMs)

The architecture is proposed by Shen et al. [38] which uses the similar idea of modeling the semantic relationship of a given document pair by using convolution layers instead

of simple non-linear dense layers by Huang et al. [24]. The model is evaluated on dataset from Web documents.

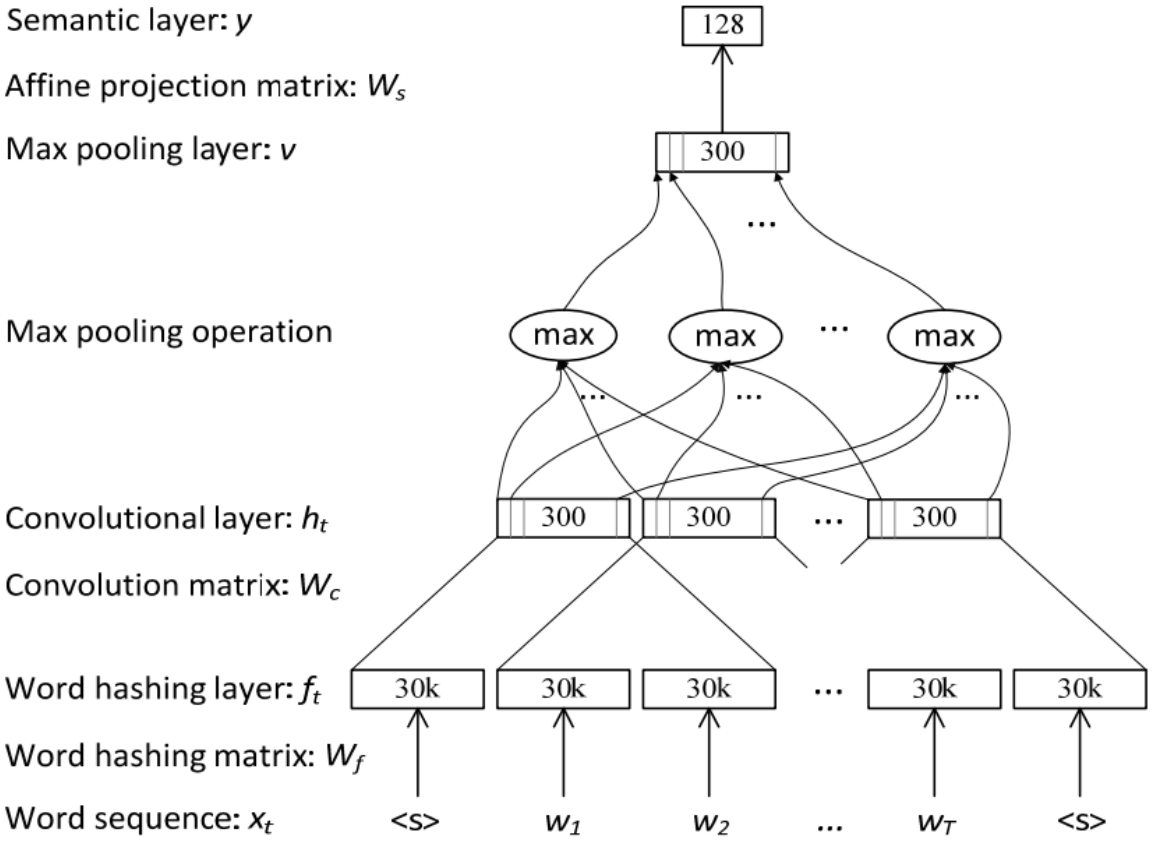

Figure 2.18: CDSSM architecture.

The architecture (Figure 2.18) has the following layers:

1. Word hashing layer: tri-letter hashing. For each word of a given text, it is split into a set of tri-letter grams. For example, the word 'buyer' is split into {'#bu', 'buy', 'uye', 'yer', 'er#' }. A tri-letter vocabulary is built from all sets of tri-letter grams. The vocabulary is then used to map each set into a one-hot vector representing the occurrence of each tri-letter gram by value 1, otherwise 0. One-hot vectors of all words are stacked to form the input of the next layer, convolutional layer.

2. Convolutional layer. Usual convolutional operation on the input from tri-letter hashing.

3. Max pooling layer. Select the max values as the most salient features out of the convolutional layer output.

4. Semantic layer. The layer applies affine projection matrix to generate more complex features from pooled features.

The output of semantic layer is the final representation of each input text. Similarity of a text-pair is calculated by cosine of the two corresponding final representations.

Given a set of similar text-pairs, the training algorithm of the model is first generate a set of dissimilar text-pairs from randomly sampling each new negative text-pair by replacing its second text with the second text of the other pairs. Then the training algorithm estimates the model parameters by maximizing the similarity of similar pairs and dissimilarity of dissimilar pairs.

2.3.4 Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks (L2RSTP-CNN)

This architecture is proposed by Severyn and Moschitti [23] which learns semantic relationship from input text pairs with a convolutional-pooling structure combined with additional features by softmax layer. The model is evaluated in reranking short text pairs where questions and documents are limited to a single sentence.

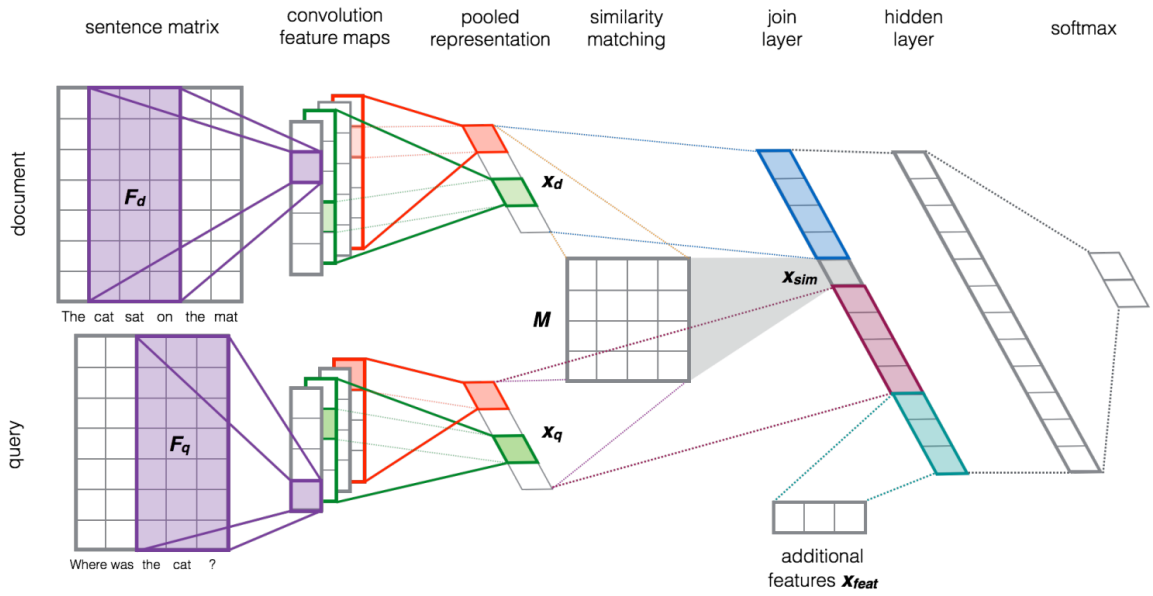

Figure 2.19: L2RSTP-CNN architecture.

The architecture (Figure 2.19) contains the following layers:

1. Word embedding layer. This layer maps each word in each of a given text pair into a vector representation by using a pre-trained `word2vec` model. The word vectors are stacked into a sentence matrix.

2. Convolution layer. Convolutional operations slice over the sentence matrices to capture the similarity of context patterns in the input with the own feature maps.

3. Max pooling layer. This layer selects the maximum similarity score for each feature map as the most salient score and then outputs pooled representations.

4. Join layer. This layer concatenates pooled representations of the given text pair, the similarity matching of the pooled representations, and additional features (word overlapping features).

5. Hidden layer with softmax. This layer applies affine function to map the input into a vector with output size of 2. Then the softmax operation selects the greater of the two elements of the output vector as the predicted label (relevant or irrelevant).

The model is trained as a binary classification problem. In the training process of the model, two groups of text-pairs are generated. One group is similar or relevant text-pairs which is assigned with label 1. The other group is dissimilar or irrelevant text-pairs which is assigned with label 0. Then a categorical loss function is used to estimate the parameters of the model. After training, the model outputs the label distribution for each text-pair which is then treated as ranking score.

Chapter 3

Method

3.1 Enriched Deep Learning Models with External Features

In this section, we describe how we enrich the selected deep learning model with external features. Our approach to build the re-ranking model, shown in figure 3.1, is based on the architectures from Kim [39] and Huang et al. [24] that use convolution layers for auto feature extraction, and additionally combining with external document features. The limitation of the two models is that they only extract features from document words, but lack some features from the corpus and domain of the documents. Therefore, together with convolutional features extracted from convolution layers, we combine them with additional external features (e.g. TF-IDF [40]).

The model consists of two sub-models with the same architecture but different trained weights. One is for modeling questions and the other is for modeling documents. The sub-model receives an input text, scans through each windows of words, matches them with the convolutional feature maps, then pools the most matched one to collect convolutional features. The features, then, are combined with external features with a weighting scheme, to output the final representation. These outputs are used to compute score of a document d given a question q: $Score(d|q)$. Finally, we train the model to maintain the below constraint:

$$Score(d_{positive}|q) - Score(d_{negative}|q) > 0 \tag{3.1}$$

With two separated sub-models, the model is expected to represent questions and documents in their own space, then learns the relationship mapping from question space to document space and vice versa.

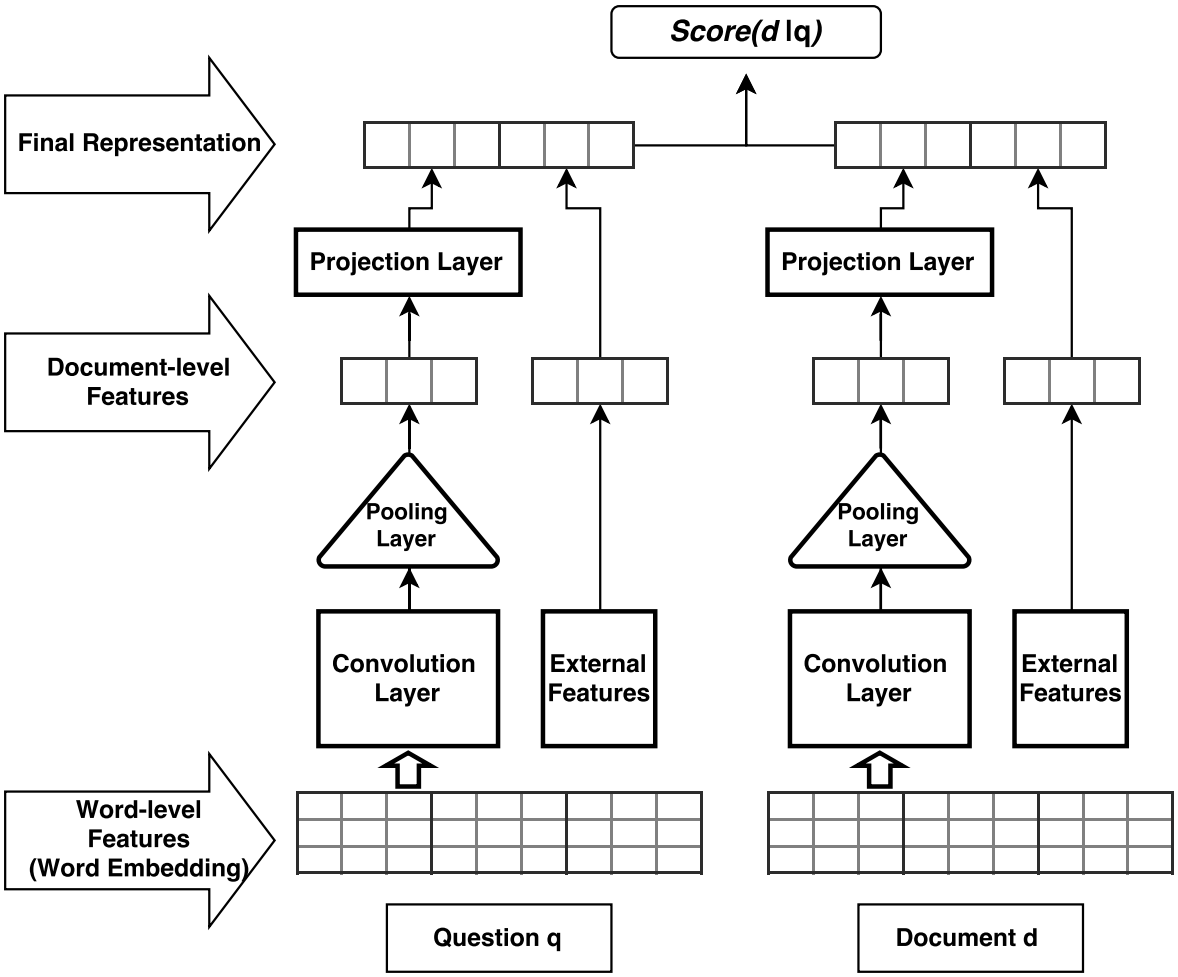

Figure 3.1: Enriched re-ranking model with external features.

Word-level features: transforming input text to matrix representation by word embedding layer.

For each word in the input text, the model looks up the corresponding embedding vector w in the word embedding matrix $W_{embedding} \in \mathbb{R}^{|V| \times embsize}$ (V is vocabulary and $embsize$ is the size of each word embedding vector) and concatenates all vector into a matrix representing the given input text (Equation 3.2).

$$[\ w_1;\ w_2;\ w_3;\ ...;\ w_s\ ] = \text{embedding-layer}(W_{embedding}, \text{input text}) \tag{3.2}$$

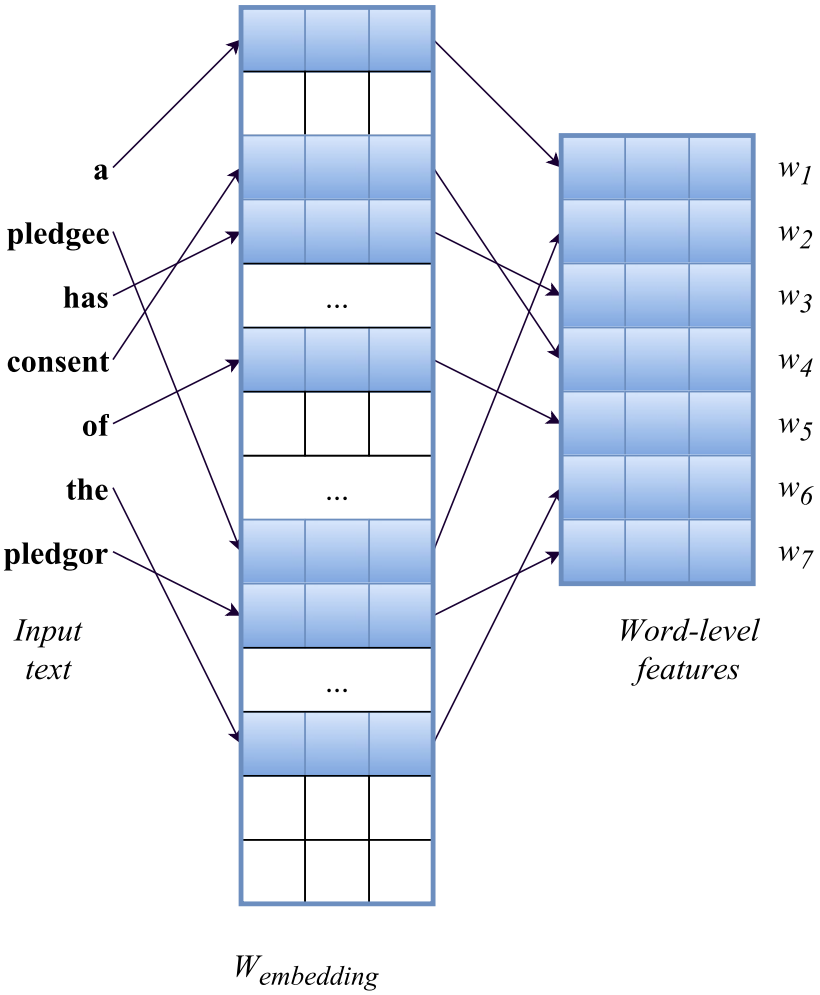

Figure 3.2: Composing word-level features from word embedding matrix $W_{embedding}$.

Convolutional-pooling operations.

We then apply convolutional operations on the embedding matrix. To capture local context information with various granularity, we adopt multiple-size feature mapping from Kim [39]. From experiments, we select 3, 4 and 5 as feature map sizes for every 150 feature maps. For each feature map, the model slices it through the input matrix to calculate the similarity between the feature map and each window in the input matrix. Then, max pooling is applied to select the most salient feature. Totally, the operation will produce an output vector of size 450 referred as convolutional-pooling features.

$$v_{convoluted} = sum \left(\begin{bmatrix} w_t \\ w_{t+1} \\ \dots \\ w_{t+k-1} \end{bmatrix} \circ W_{feature-map} \right) \tag{3.3}$$

- $W_{feature-map}$: A feature map with size $k \times embsize$.

- $\begin{bmatrix} w_t \\ w_{t+1} \\ ... \\ w_{t+k-1} \end{bmatrix}$: a sliding window with size of k from word t^{th} to word $(t+k-1)^{th}$.
- $\circ$: element-wise multiplication (Hadamard product).
- sum: summation of all elements.

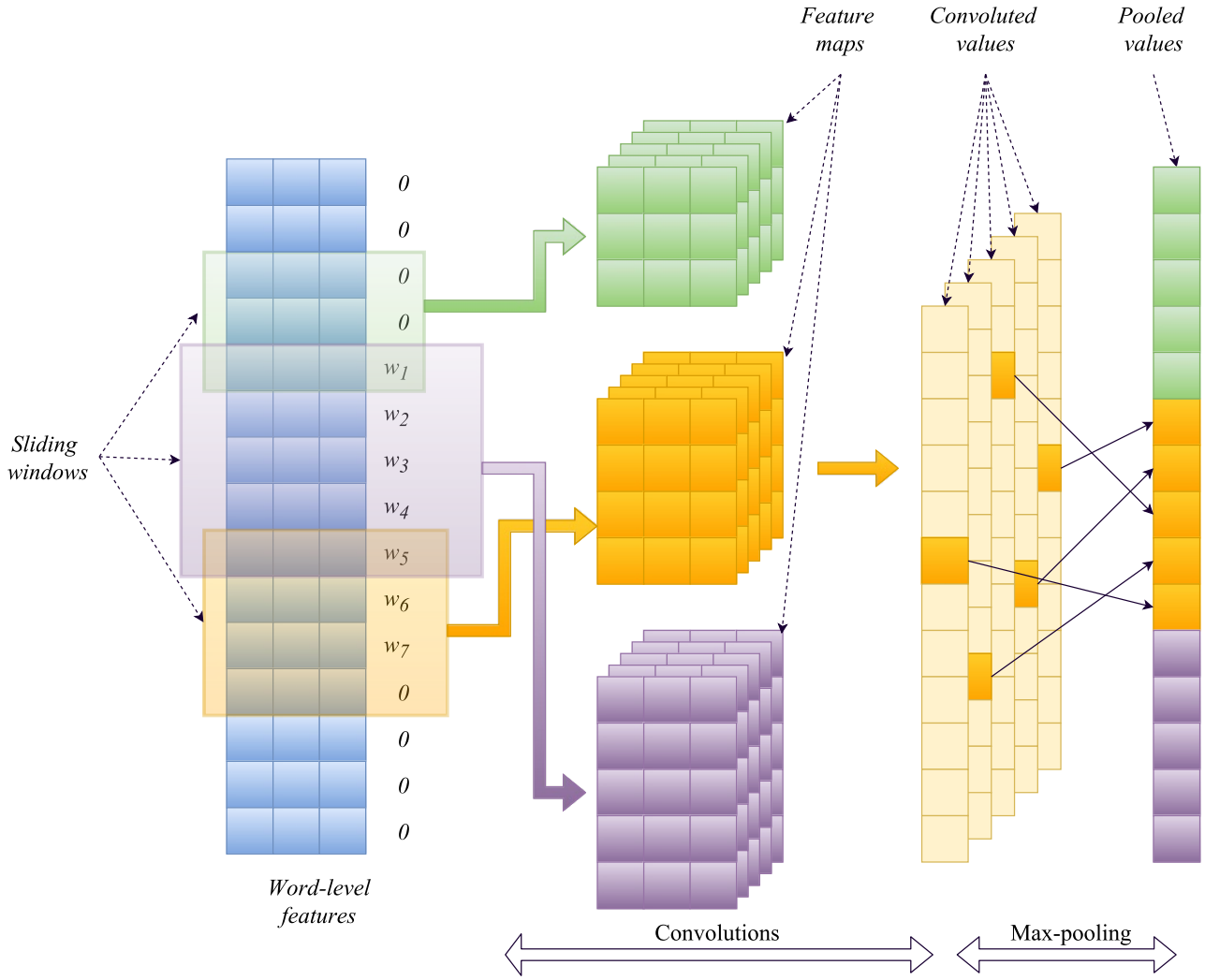

Figure 3.3: Convolutional-pooling operations with multiple-size feature maps. To capture context windows with the number of words less than the feature map sizes, we add dummy word represented by 0 vectors to the beginning and the end of the texts.

Document-level features.

We extract two sets of features from each document. In addition to the convolutional-pooling features from convolutional-pooling operations, we collect external features $v_{external}$ which characterize the document in its corpus.

Projection layer: multiple linear combinations with a non-linear activation.

From the basic convolutional features $v_{convolutional-pooling}$, a set of linear combinations W_{proj} is carried out to compute more complex derived features v_{latent}. A non-linear transformation is applied to encourage the non-linearity of the features. The non-linear transformation involves applying linear weighting matrix W_{proj} and then non-linear activation *tanh* function.

$$v_{latent} = tanh(W_{proj} \cdot v_{convolutional-pooling}) \tag{3.4}$$

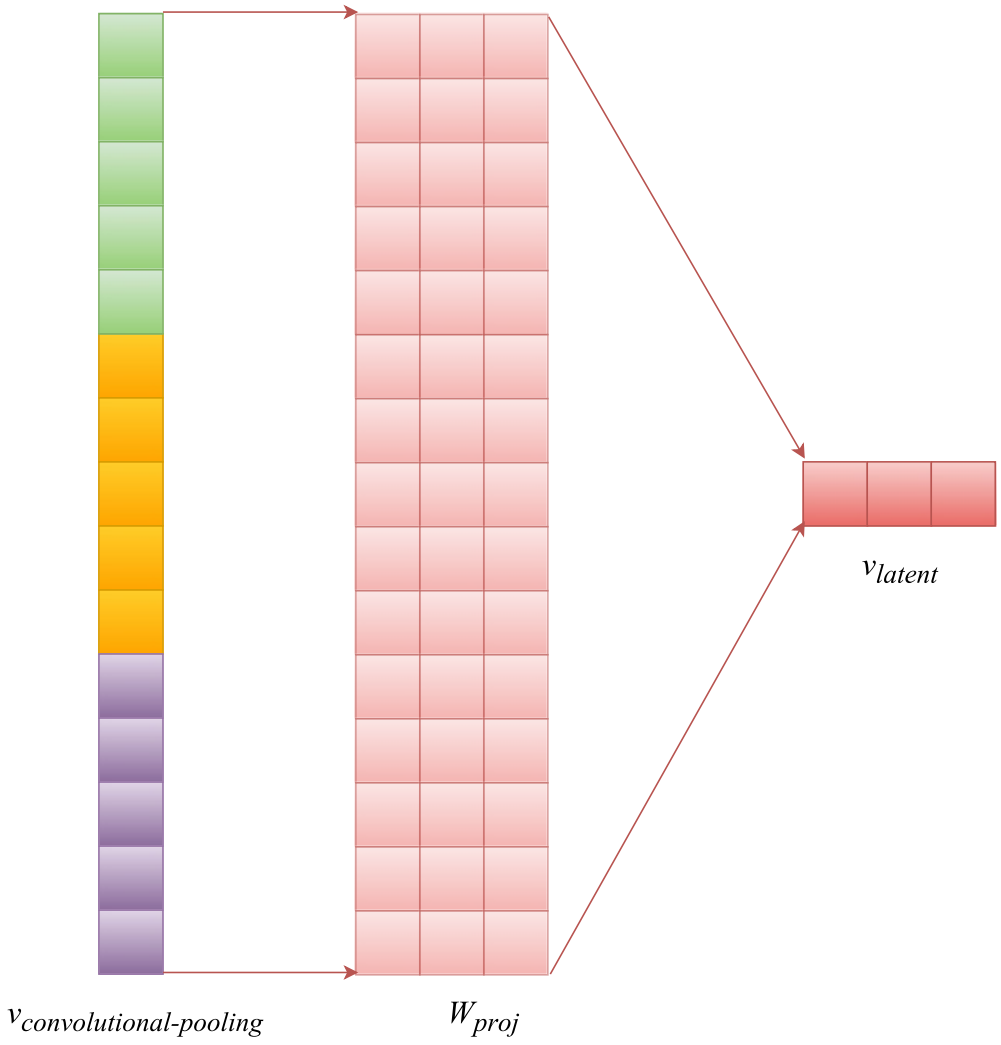

Figure 3.4: Affine with non-linear transformation of convolutional-pooling features into latent features.

Final representation: combination of latent features and external features.

Along with latent features extracted from neural modules, we extracted some hand-crafted features $v_{external}$, for example TF-IDF. Then we do a simple concatenation of the external features $v_{external}$ and the latent features v_{latent} to form the final representation v_{final} of the input text.

$$v_{final} = \begin{bmatrix} v_{latent} \\ v_{external} \end{bmatrix} \tag{3.5}$$

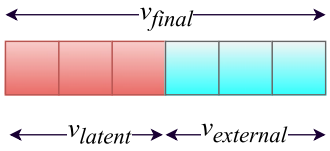

Figure 3.5: Concatenation of latent features and external features.

Relevant Score.

The relevant score $Score(d|q)$ of a question q and document d is calculated by *cosine* of their two final representations $v_{final}(q)$ and $v_{final}(d)$ respectively.

$$Score(d|q) = cosine(v_{final}(d), v_{final}(q)) \tag{3.6}$$

Our model encourage the impact of external features to the relevant score computation. By not putting v_{final} through another layer, which is different from L2RSTP-CNN, we reduce the effect that external features are out-weighted by latent features from neural layers. With the random initialization of artificial neurons in the model, the external features which are fixed can act as the starting point of searching space. This also means the external features have strong impact to the performance of the model, which will be discussed in Chapter 4.

3.2 Corpora

This section, we describe the data we used to evaluate the method. The corpora used in the experiments are named WikiQA, COLIEE 2015, and TREC 2011 Legal Track. Though the corpora are common in that annotated data is in sentence-sentence pairs with positive (relevant/responsive) or negative (irrelevant/non-responsive) labels, each corpus has it own typical characteristic which affects the performance correlation of experimented models.

3.2.1 Corpora Overview

COLIEE 2015

This is a dataset from Competition on Legal Information Extraction/Entailment (COLIEE) 2015[1] for a legal bar examination. Each question is paired with a couple of civil articles. While three subtasks involving this dataset are extraction, entailment and both, only extraction subtask is dealt in the experiments. The task is that given a question or a statement, retrieve its relevant civil articles from 755 articles in Japanese Civil Code. The dataset contains 267 training questions in group H[18-23] with and 66 testing questions are in group H25.

[1] `http://webdocs.cs.ualberta.ca/~miyoung2/COLIEE2015/`

(Keeping the Thing Retained by Holders of Rights of Retention)
Article 298
(1) A holder of a right of retention must possess the Thing retained with the care of a good manager.
(2) A holder of rights of retention may not use, lease or give as a security the Thing retained unless he/she obtains the consent of the obligor; provided, however, that this shall not apply to uses necessary for the preservation of that Thing.
(3) If the holder of a right of retention violates the provisions of the preceding two paragraphs, the obligor may demand that the right of retention be extinguished.

H20-12-5
If a holder of a right of retention has consent of the obligor, or if a pledgee has consent of the pledgor, then they each may lease any collateral.

Figure 3.6: An example of a question and its relevant article in COLIEE 2015 dataset.

TREC 2011 Legal Track

TREC 2011 Legal Track[2] has a single task: identifying documents responsive to requests for production that are typical in civil litigation. The dataset contains 3 requests (topics) and about 670,000 documents. Each topic is a long and complicated sentence with ≈70 words. Besides, the documents are in different formats where they can be email messages, tables, or even unreadable formats, whose content lengths varies in a large scale. In the experiment, "rel.401", "rel.402", and "rel.403" files consisting of 4,625 documents are used, and long documents are truncated to maximum length of 100 words.

[2] http://plg.uwaterloo.ca/~gvcormac/legal11/treclegal11.html

Topic 401. All documents or communications that describe, discuss, refer to, report on, or relate to the design, development, operation, or marketing of enrononline, or any other online service offered, provided, or used by the Company (or any of its subsidiaries, predecessors, or successors-in-interest), for the purchase, sale, trading, or exchange of financial or other instruments or products, including but not limited to, derivative instruments, commodities, futures, and swaps.

[✓]**Responsive Message.**

Enron
P.O.Box 1188
Houston, TX 77251-1188
Mark Palmer
(713) 853-4738
ENRON RESUMES MARKET-MAKING ACTIVITY
IN NORTH AMERICAN NATURAL GAS AND POWER
FOR IMMEDIATE RELEASE: Wednesday, Sept. 12, 2001
HOUSTON Enron announced today it will resume its market-making activity in North American natural gas and power. Enron will buy and sell natural gas and power by phone and over its online transaction platform EnronOnline until noon CDT. We see no reason for North American natural gas and power markets to become unstable in the aftermath of yesterdays tragedies, said Greg Whalley, Enron president and chief operating officer. These are domestic commodities, and the physical infrastructure is secure and operating. Enrons markets outside of North America will operate according to their normal schedule. Enron is one of the worlds leading energy, commodities and services companies. The company markets electricity and natural gas, delivers energy and other physical commodities, and provides financial and risk management services to customers around the world. Enrons Internet address is www.enron.com. The stock is traded under the ticker symbol ENE.

[✗]**Non-responsive Message**

Rick,
Please see attached a summary on consultancy and Audit/legal spend for May
Ytd 2001.
Govt Affairs Consultancy FINAL.xls
thanks
Greg

Figure 3.7: TREC 2011 Legal Track data examples.

WikiQA

WikiQA[3] is a dataset for open-domain question answering presented by Microsoft Research. Each question is paired with a paragraph or a list of sentences. A subset of the sentences is marked as they can answer the question. Thus, the task is to identify the sentences answering the question given a question and a paragraph. The dataset contains 2,118 training questions and 633 testing questions.

Question: HOW AFRICAN AMERICANS WERE IMMIGRATED TO THE US ?
Answer Candidates:

[✗] African immigration to the United States refers to immigrants to the United States who are or were nationals of Africa .

[✗] The term African in the scope of this article refers to geographical or national origins rather than racial affiliation .

[✗] From the Immigration and Nationality Act of 1965 to 2007 , an estimated total of 0.8 to 0.9 million Africans immigrated to the United States , accounting for roughly 3.3 % of total immigration to the United States during this period .

[✗] African immigrants in the United States come from almost all regions in Africa and do not constitute a homogeneous group .

[✗] They include people from different national , linguistic , ethnic , racial , cultural and social backgrounds .

[✓] **As such , African immigrants are to be distinguished from African American people , the latter of whom are descendants of mostly West and Central Africans who were involuntarily brought to the United States by means of the historic Atlantic slave trade .**

Figure 3.8: An example in WikiQA dataset. The last answer is the correct one out of all candidates.

[3] http://aka.ms/WikiQA

3.2.2 Corpora Analysis

COLIEE 2015

The civil code is organized in the following structure: Part → Chapter → Section → Subsection → Article → Paragraph (Figure 3.9). While we see that this is valuable information, we've not yet made use of it.

```
- Part I
    - Chapter 1
        - Section I
            - Subsection 1
                - Article 1
                    - Paragraph (1)
                    - Paragraph (2)
                - Article 2
            - Subsection 2
        - Section II
    - Chapter 2
- Part II
- Part III
```

Figure 3.9: The civil code's data structure.

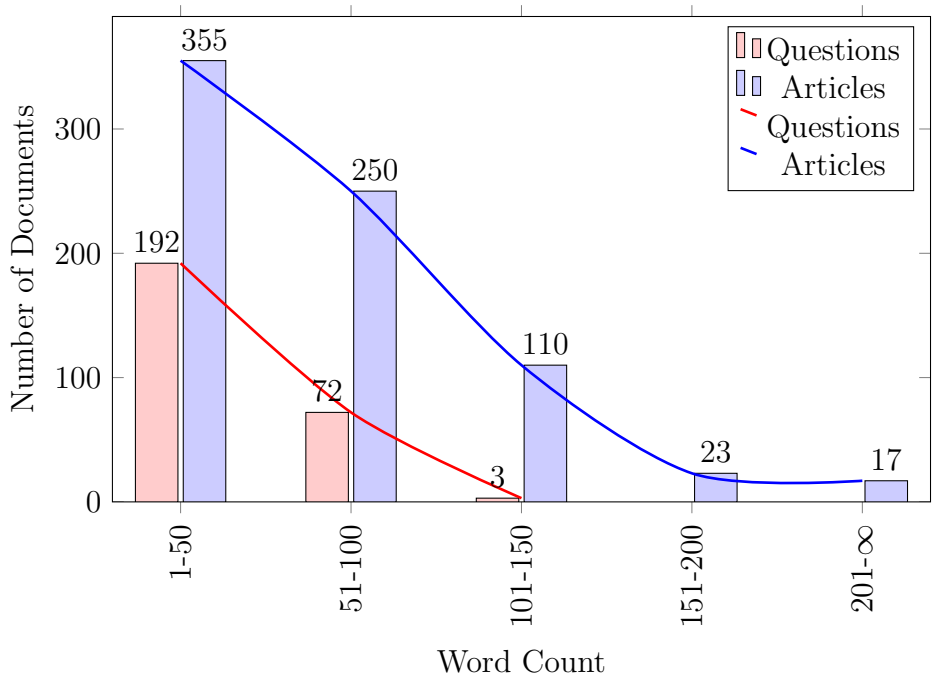

Figure 3.10: Word count statistics for COLIEE 2015 corpus.

We calculate the word overlapping ratios for each question-article pair after retrieving top N=10 articles from the civil code using keyword-based retrieval method. The statistical information in Figure 3.11 and 3.12 shows that there's a portion of questions having word overlapping ratios of relevant articles higher than irrelevant articles.

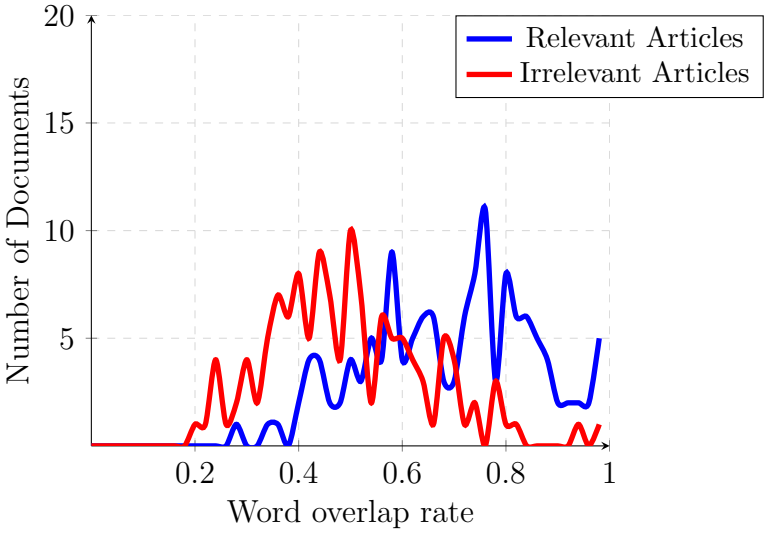

Figure 3.11: Term overlap rate of retrieved articles in top 1 on COLIEE 2015 dataset.

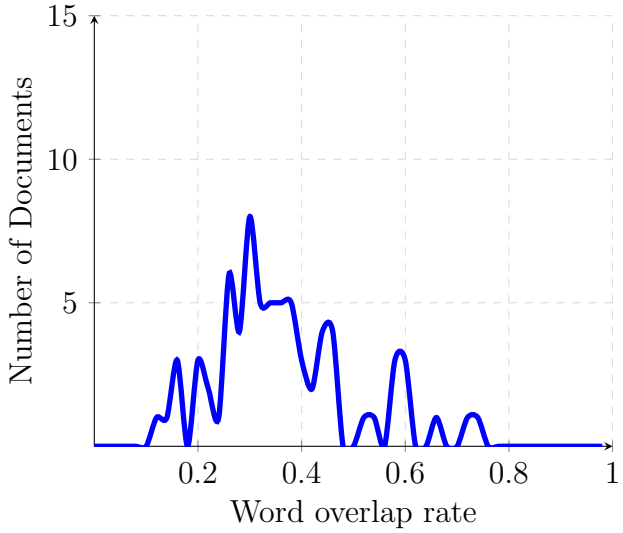

Figure 3.12: Term overlap rate of relevant articles not in retrieved top 10 on COLIEE 2015 dataset.

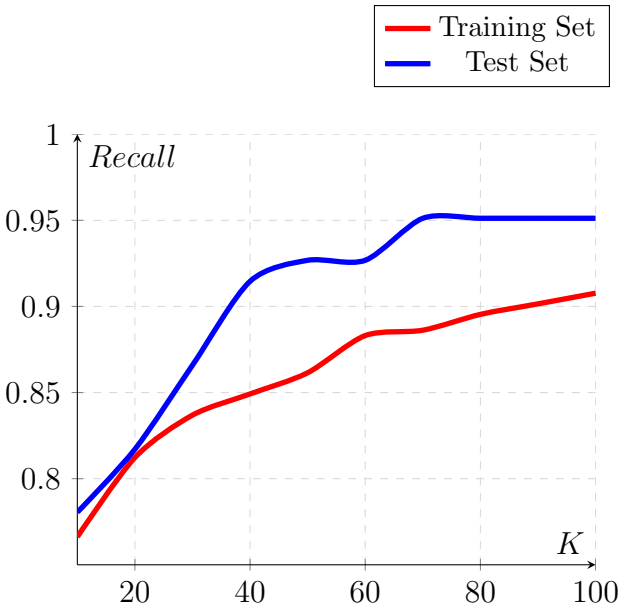

Figure 3.13: Recall rate of selecting top K documents from document retrieval K=[10,100]

Table 3.1: Article distribution: the number of articles in the Civil Code that occur in training set, test set and co-occur in both; and the number of relevant articles for each question.

	Training Set	Test Set	Both	Civil Code	Per Question
No. Articles	220	72	41	755	1.2 ± 0.5

TREC 2011 Legal Track

The corpus contains files in the following formats:

- email messages
- attached documents with text paragraphs
- attached statistical tables
- non-text (binary)

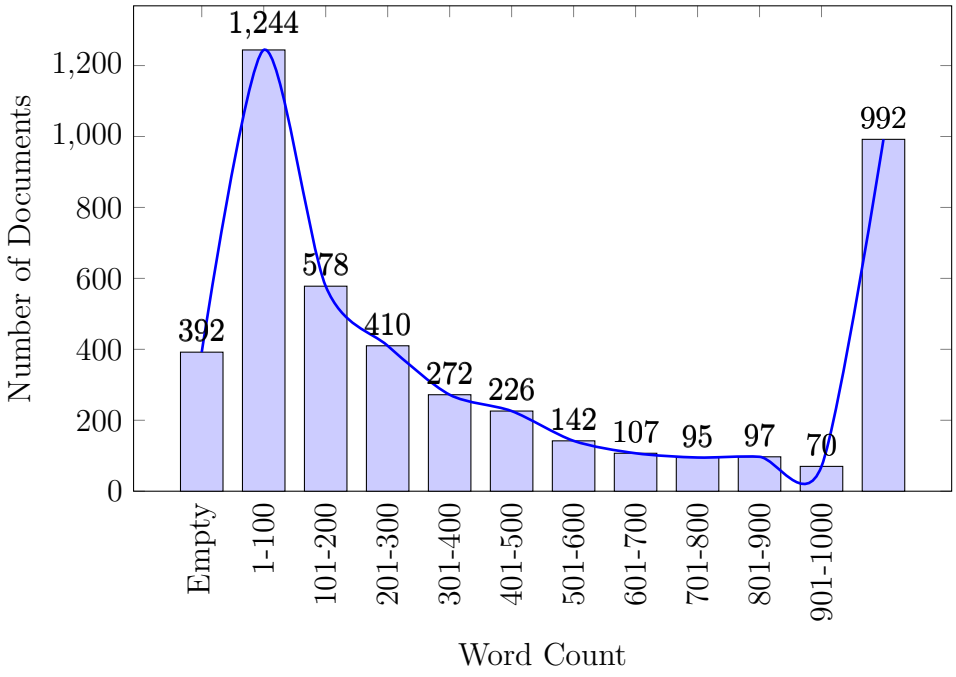

Figure 3.14: Word count statistics of document candidates for TREC 2011 Legal Track corpus.

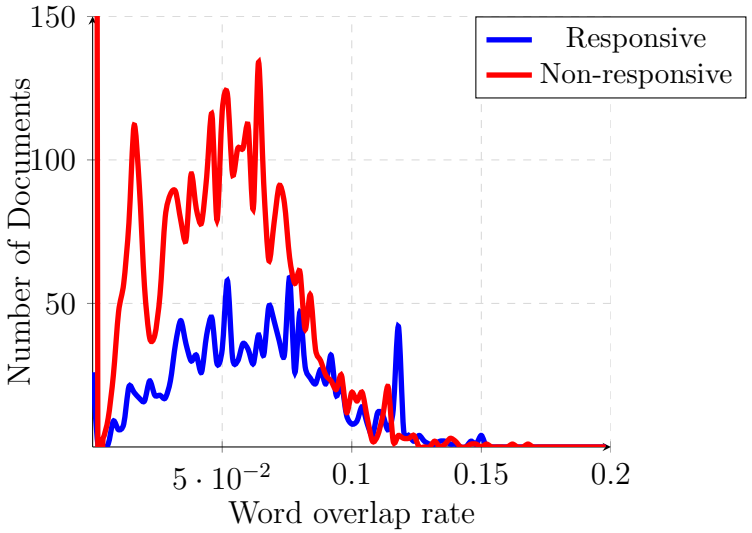

Figure 3.15: Word overlap distribution over document candidates for TREC 2011 Legal Track corpus.

The huge variation in document format and length (Figure 3.14), and the low word overlap rate, mostly below 10% as shown in Figure 3.15, suggests that this is a challenging corpus.

WikiQA

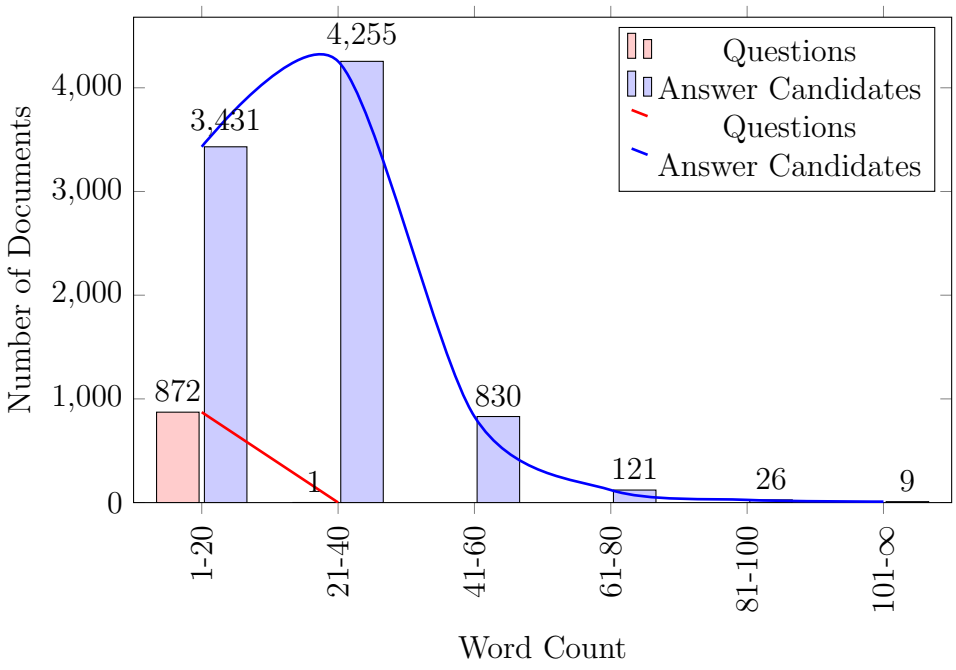

Figure 3.16: Word count statistics for WikiQA corpus.

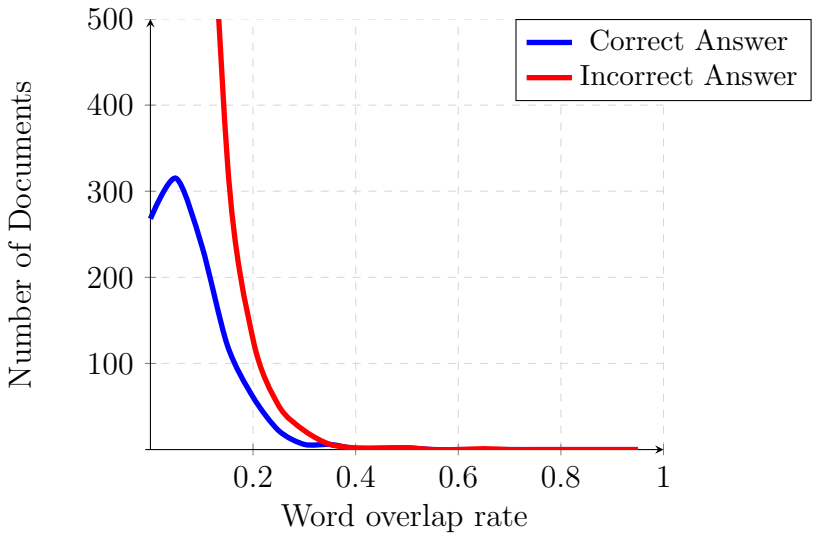

Figure 3.17: Word overlap distribution over answer candidates for WikiQA corpus.

Table 3.2: Statistics of the WikiQA dataset [2].

	Train	Dev	Test	Total
# of questions	2,118	296	633	3,047
# of sentences	20,360	2,733	6,165	29,258
# of answers	1,040	140	293	1,473
Average length of questions	7.16	7.23	7.26	7.18
Average length of sentences	25.29	24.59	24.95	25.15
# of questions without answers	1,245	170	390	1,805

3.3 Experiments

The goal of our experiments is to evaluate the performances of different models with various model configurations on the three datasets described in Section 3.2.

3.3.1 Data Pre-processing

Stanford Core NLP toolkit is a popular tool for common NLP tasks developed by Stanford NLP group. It provides the state of the art performances for preprocessing texts and helps analyze the corpus.

We preprocess all the three corpora in the following steps:

1. Tokenization: split each input document into the sequence of words.
2. Lowercase: all words are convert to lowercase.
3. Lemmatization: convert each word back to its lemma form. For example, the words "buy", "buys", "bought" are converted to the base form "buy".

While tokenization is a must to provide the input for the models, the preprocessing steps 2 and 3 are optional which reduce the variety of word forms to increase matching effect.

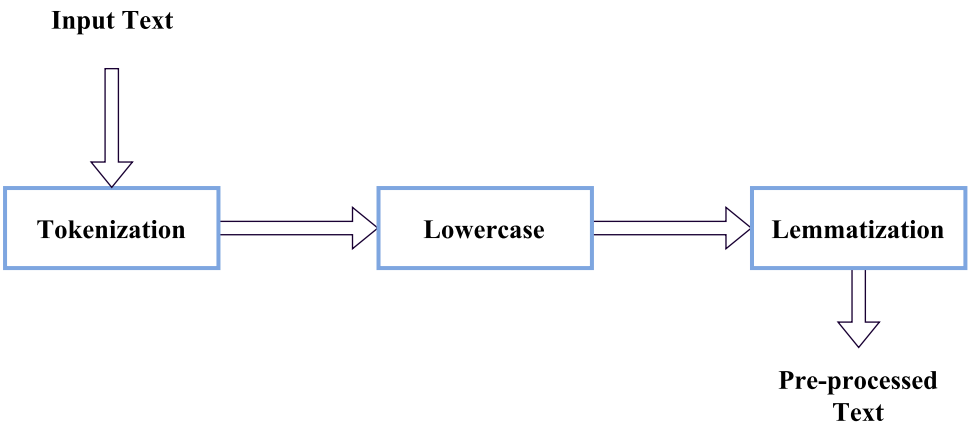

Figure 3.18: Data pre-processing flow.

(Urgent Management of Business)
Article 698
If a Manager engages in the Management of Business in order to allow a principal to escape imminent danger to the principal's person, reputation or property, the Manager shall not be liable to compensate for damages resulting from the same unless he/she has acted in bad faith or with gross negligence.

(Urgent Management of Business) Article 698 If a Manager engages in the Management of Business in order to allow a principal to escape imminent danger to the principal 's person , reputation or property , the Manager shall not be liable to compensate for damages resulting from the same unless he/she has acted in bad faith or with gross negligence .

(urgent management of business) article 698 if a manager **engages** in the management of business in order to allow a principal to escape imminent danger to the principal 's person , reputation or property , the manager shall not be liable to compensate for damages **resulting** from the same unless he/she **has acted** in bad faith or with gross negligence .

(urgent management of business) article 698 if a manager **engage** in the management of business in order to allow a principal to escape imminent danger to the principal 's person , reputation or property , the manager shall not be liable to compensate for damages **result** from the same unless he/she **have act** in bad faith or with gross negligence .

Figure 3.19: Data pre-processing example.

3.3.2 Experimental Models

We conduct experiments on the above corpora to compare our approach with the following models (some models are described in section 2.4):

- TF-IDF: a common method for retrieving documents based on term occurrence and term weighting [40]. The term occurrence of term t in document d is computed by $tf(t, d)$ as follows:

$$tf(t, d) = \text{the number of times term } t \text{ appearing in document } d \tag{3.7}$$

The term weighting of term t in set of documents D is computed by $idf(t, D)$ as follows:

$$idf(t, D) = log\frac{|D|}{|\{d \in D : t \in d\}|} \tag{3.8}$$

Then, given a set of documents D, the TF-IDF score of term t in document $d \in D$ is computed by $\textit{tf-idf}(t, d \in D)$ as follows:

$$\textit{tf-idf}(t, d \in D) = tf(t, d) \cdot idf(t, D) \tag{3.9}$$

Finally, the relevance score of two documents d_1 and d_2 is as follows:

$$relevance(d_1, d_2) = \frac{\sum_{t \in d_1, t \in d_2} \textit{tf-idf}(t, d_1) \cdot \textit{tf-idf}(t, d_2)}{\sqrt{\sum_{t \in d_1} \textit{tf-idf}(t, d_1)^2}\sqrt{\sum_{t \in d_2} \textit{tf-idf}(t, d_2)^2}} \tag{3.10}$$

- Paragraph Vector (PV) (Chapter 2 Section 2.3.2). The relevance score of two documents is the cosine similarity of their corresponding paragraph vectors.

- Deep Structured Semantic Model with Convolutional-Pooling Structure (CDSSM) (Chapter 2 Section 2.3.3). The relevance score of two documents is the cosine similarity of their corresponding semantic vectors.

- Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks (L2RSTP-CNN) (Chapter 2 Section 2.3.4). The relevance score of two documents is given by the softmax operation of the model.

- Our Approach (Chapter 3 Section 3.1) We experiment our approach with the following configurations:

  - Word embedding: `word2vec` model with vector size of 64 by skip-gram.
  - Convolutional filter sizes: a sub set of {3, 4, 5, 6}.
  - Number of feature maps: from 100 to 200 feature maps for each convolutional filter size.
  - External Features: we use TF-IDF as external feature because it is simple yet strong feature in COLIEE 2015 (the main corpus) (Table 3.3).
  - Objective: maximizing the score difference between positive document $d_{positive}$ and negative article $d_{negative}$ of a question q for all questions:

$$maximize(Score(d_{positive}|q) - Score(d_{negative}|q))\forall q \tag{3.11}$$

  - Training strategy: we apply pair-wise ranking optimization. For each question q, together with its positive documents $D_+(q) = \{d_+\}$, we sample its negative documents $D_-(q) = \{d_-\}$ from the source dataset. For example, with COLIEE 2015 dataset, given a question and its relevant article, we

select 50 irrelevant articles. Then we minimize the following loss function L to obtain the objective in Equation 3.11. We use exponential function to magnify the loss if the score of negative documents is much higher than positive documents.

$$L = \frac{1}{|Q|} \sum_{q \in Q} \frac{1}{|D_+(q)|} \sum_{d_+ \in D_+(q)} \frac{1}{|D_-(q)|} \sum_{d_- \in D_-(q)} e^{Score(d_-|q) - Score(d_+|q)} \quad (3.12)$$

* Q: the set of all questions.
* $D_+(q)$: the set of positive (relevant/correct/responsive) documents respecting to question $q \in Q$.
* $D_-(q)$: a sampled set of negative (irrelevant/incorrect/non-responsive) documents respecting to question q.

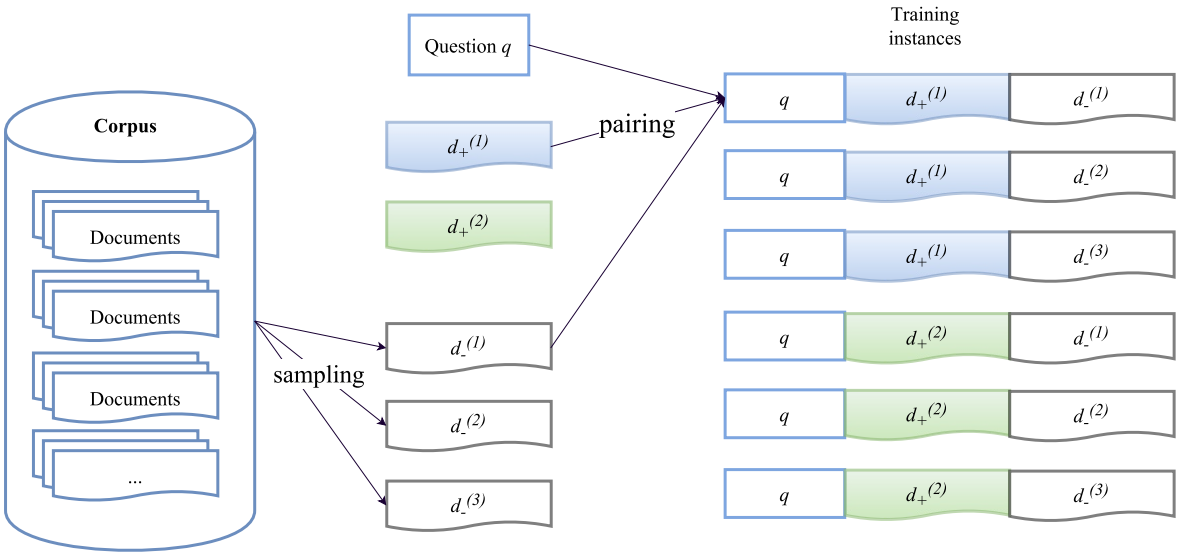

Figure 3.20: Diagram of making training data from a corpus.

Training Configurations.

With the dataset without given validation data (TREC 2011 Legal Track and COLIEE 2015), we select 90% as training data and 10% as validation data. We train with maximum 100 epochs and select the trained parameters producing best performance on validation data. With models that use `word2vec` embedding vectors as input, we train one `word2vec` model for each corpus by running skip-gram (Chapter 2 Section 2.3.1) algorithm on the joint texts of the corpus and a general large text corpus.

3.3.3 Evaluation

We evaluate each corpus as follows:

- COLIEE 2015. For each test question, top K (K=50) documents are retrieved by TF-IDF model. After re-ranking the retrieved documents, top T documents are assigned as relevance. As shown in table 3.1, the average number of articles per question is 1.2, so we select T=1.

- WikiQA. We select the first answer candidate as correct answer.
- TREC 2011 Legal Track. We use validation set to calculate score threshold for selecting responsive messages.

The evaluation metrics including mean average precision, precision, recall, and f-score are used to evaluate the performances of the models. These metrics are described as follows.

Precision

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cup \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|} \tag{3.13}$$

Recall

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cup \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|} \tag{3.14}$$

F-Score

$$\text{F-score} = \frac{2 * \text{precision} * \text{recall}}{\text{precision} + \text{recall}} \tag{3.15}$$

Mean Average Precision (MAP)

$$\text{MAP} = \frac{\sum_{q \in Q} \text{AveP}(q)}{|Q|} \tag{3.16}$$

$\text{AveP}(q)$ is average precision given question q computed by the following equation.

$$\text{AveP} = \frac{\sum_{k=1}^{n} P(k) * \text{rel}(k)}{|\{\text{relevant documents}\}|} \tag{3.17}$$

$P(k)$ is the precision of selecting top k documents as the retrieved documents. $\text{rel}(k)$ is function to check if the document k^{th} of the retrieved documents is relevant or not, and returns 1 if the document is relevant, otherwise 0.

MAP is one of the effective metric to evaluate the performance of a ranking algorithm in order to rank relevant instances higher than irrelevant ones.

3.3.4 Experimental Results

The results show a large variation of performances for each corpus of the experimented models.

Table 3.3: Performances are measured based on precision (Pre), recall (Rec), F-score (F_1), and mean average precision (MAP) calculated on positive instances (relevant or responsive documents) only.

Model	WikiQA				COLIEE 2015				TREC 2011 Legal Track			
	Pre@1	Rec@1	F_1@1	MAP	Pre@1	Rec@1	F_1@1	MAP	Pre	Rec	F_1	MAP
TF-IDF	0.4280	0.3549	0.3881	0.6021	0.6667	0.5366	0.5946	0.7034	0.1855	0.3312	0.2378	0.2348
PV	0.3333	0.2765	0.3022	0.5136	0.5758	0.4634	0.5135	0.5610	0.4135	0.2792	0.3333	0.2967
CDSSM	0.3457	0.2867	0.3134	0.5177	0.2121	0.1707	0.1892	0.2913	0.3132	0.9610	0.4725	0.5581
L2RSTP-CNN	0.5309	0.4403	0.4813	0.6663	0.6061	0.4878	0.5405	0.6902	0.6724	0.7040	0.6878	0.6530
Our Approach	0.4774	0.3959	0.4328	0.6397	0.7121	0.6732	0.6351	0.7561	0.3846	0.4545	0.4167	0.4138

Table 3.4: Number of questions having relevant articles at top T=1 in COLIEE 2015 dataset.

	TF-IDF	PV	CDSSM	L2RSTP-CNN	Our approach
No. of questions	43	38	14	34	47

Chapter 4

Discussion

4.1 Effectiveness

Without the effort of extracting specialized features, with appropriate configuration, deep learning models are able to perform well on legal data.

Deep learning models are data-oriented. Deep learning models with pure artificial neurons show the underperformances on COLIEE 2015 dataset With limited size, but high performances on TREC 2011 Legal Track dataset with big size. This means without enough of training examples, deep learning models are susceptible to under-train. One reason is that deep learning models usually have multiple layers with remarkable number of artificial neurons, or are highly dimensional functions. With the nature of deep learning models is to find the regularity of data from its observable instances automatically, the size of training data is vital to the model performances.

In our approach of enriching deep learning models with external features. By no additional transformation on external features, We limit the search space when training our model by enforce a starting point from the external features. The approach works as expected as it improves performance from the starting point. With a good starting point on COLIEE 2015 dataset, it is able to perform better than L2RSTP-CNN which using the similar strategy of using external features in different way. With a bad starting point on TREC 2011 Legal Track, though, it improves from the starting point but underperforms the pure deep learning models (CDSSM).

Performances on COLIEE 2015 dataset

As our approach performs best on COLIEE 2015, we look at the ranked documents in the test data to review the actual outputs.

Table 4.1: Relevant article ranking of our approach compared to TF-IDF. Most of the cases, as the TF-IDF model gets the top 1 correct, our model gets the top 2.

Compared to TF-IDF

Rank Up			Rank Down		
Query	From	To	Query	From	To
H25-8-1	2	1	H25-3-4	1	2
H25-16-3	3	1	H25-11-E	1	2
H25-19-A	2	1	H25-13-3	1	2
H25-22-2	13	1 (+12)	H25-14-E	1	2
H25-23-I	2	1	H25-24-A	1	2
H25-30-3	2	1	H25-29-O	7	22 (-15)
H25-11-U	9	7			
H25-17-I	27	11 (+16)			
H25-17-E	21	8 (+13)			
H25-17-O	42	33			
H25-21-U	26	5 (+21)			
H25-21-E	34	6 (+28)			

The improvement in ranking despite of not in top 1 suggests that our model is actually capable of learning features other than just word overlapping.

H25-22-2
The obligor **shall take on** responsibilities which arise from default **of monetary** debt unless **he/she may raise** the defense of force majeure.

[✓] **Article 419** (relevant, selected by our model, rank 13 by TF-IDF)
(1) The amount of the damages for failure to perform **any obligation for** the delivery **of any money** shall **be determined with** reference to the statutory **interest** rate; **provided**, however, that, **in cases** the agreed interest rate exceeds **the statutory interest** rate, the agreed interest rate **shall** prevail.
(2) The obligee **shall not be** required to prove his/her **damages with respect** to the damages set forth in the preceding paragraph.
(3) The obligor **may not raise** the defense of force majeure **with respect to** the damages referred to in paragraph 1.

[✗] **Article 348** (irrelevant, selected by TF-IDF, rank 2 by our model)
Pledgees **may sub-pledge the** Thing pledged within the duration **of their rights**, **upon their** own responsibility. In such cases, the pledgees shall **be responsible for** any loss arising from the -pledge **even if the** same is caused by force majeure.

Figure 4.1: The article ranking of query H25-22-2 by our model and TF-IDF. Bold texts are regions of interest our model extracted with convolutional-pooling operations.

For query H25-22-2 (Figure 4.1), while TF-IDF model select Article 348 as predicted relevant article which is incorrece, our model retrieved the correct relevant Article 419. On one hand, while both articles share key words "force majeure", the shorter length of Article 348 gives it the advantage over lengthy Article 419 in term of TF-IDF calculation. On the other hand, our model is able to capture the word "monetary" in the question and the word "money" in the first paragraph of Article 419 which are strongly related to each other. Additionally, with the presence of the key word "force majeure", our model then selects Article 419 as correct relevant article. This suggests that deep learning models have potential to learn the question-article relationship and extract the key points of the relationship.

Table 4.2: Relevant article ranking of our approach compared to PV.

Compared to PV								
Rank up			Rank up			Rank down		
Query	From	To	Query	From	To	Query	From	To
H25-2-U	27	2	H25-17-O	>50	33	H25-13-3	1	2
H25-2-E	9	1	H25-19-A	>50	1	H25-14-E	1	2
H25-2-O	7	5	H25-19-I	>50	1	H25-15-4	19	>50
H25-8-1	26	1	H25-21-U	49	5	H25-24-A	1	2
H25-11-A	37	1	H25-21-E	36	6	H25-29-E	1	2
H25-11-U	>50	7	H25-22-2	>50	1			
H25-11-E	>50	2	H25-23-U	20	1			
H25-14-A	8	1	H25-23-O	3	1			
H25-16-4	10	1	H25-24-O	2	1			
H25-17-I	>50	11	H25-26-1	48	9			
H25-17-U	>50	13	H25-29-O	>50	22			
H25-17-E	>50	8	H25-30-3	>50	1			

Table 4.3: Relevant article ranking of our approach compared to L2RSTP-CNN

Compared to L2RSTP-CNN					
Rank up			Rank down		
Query	From	To	Query	From	To
H25-3-1	2	1	H25-2-U	1	2
H25-3-5	2	1	H25-2-O	4	5
H25-6-E	2	1	H25-11-U	2	7 (-5)
H25-14-A	6	1 (+5)	H25-11-E	1	2
H25-14-I	2	1	H25-17-I	4	11 (-7)
H25-16-3	2	1	H25-17-E	2	8 (-6)
H25-16-4	3	1	H25-17-O	11	33 (-22)
H25-17-U	20	13	H25-26-1	2	9 (-7)
H25-21-U	38	5 (+33)	H25-29-E	1	2
H25-21-E	21	6 (+15)			
H25-22-1	2	1			
H25-22-3	2	1			
H25-22-4	2	1			
H25-23-O	4	1			
H25-24-A	3	2			
H25-26-3	3	1			
H25-29-O	29	22			

Table 4.4: Relevant article ranking of our approach compared to CDSSM.

Compared to CDSSM								
Rank up			Rank up			Rank down		
Query	From	To	Query	From	To	Query	From	To
H25-2-I	2	1	H25-19-I	6	1	H25-29-E	1	2
H25-2-U	10	2	H25-21-U	>50	5	H25-29-I	3	>50
H25-2-E	4	1	H25-21-E	>50	6	H25-15-4	49	>50
H25-2-O	17	5	H25-22-1	>50	1			
H25-3-4	12	2	H25-22-2	12	1			
H25-3-5	8	1	H25-22-3	>50	1			
H25-4-U	2	1	H25-22-4	>50	1			
H25-6-E	19	1	H25-23-I	2	1			
H25-8-3	>50	1	H25-23-O	38	1			
H25-8-4	8	1	H25-24-A	6	1			
H25-11-A	>50	1	H25-24-I	4	1			
H25-11-U	>50	7	H25-24-O	2	1			
H25-11-E	3	2	H25-26-1	27	9			
H25-11-O	>50	1	H25-26-2	5	1			
H25-13-2	29	1	H25-26-3	4	1			
H25-13-3	3	2	H25-26-4	2	1			
H25-13-4	10	1	H25-27-1	>50	1			
H25-13-5	21	1	H25-27-2	25	1			
H25-14-I	2	1	H25-27-4	8	1			
H25-14-E	41	2	H25-29-A	6	1			
H25-16-4	23	1	H25-29-U	6	1			
H25-17-I	>50	11	H25-29-O	>50	22			
H25-17-U	>50	13	H25-30-3	10	1			
H25-17-E	>50	8	H25-30-4	>50	1			
H25-17-O	>50	33						

4.2 Limitations

In current study, we limit ourselves to not yet handle some characteristics of legal domain which are references, logical structures and language abstraction.

Content/context references

For the sake of consistency and organisability of legal documents, a part of legal text can refer to other part. For example, the first paragraph Article 568 refers to the provisions from Article 561 through to the preceding Article (Figure 4.2).

References can be resolved with the following information.

- Reference expression: the phrase that contains one or more references.
- Referred source: the name or position where a reference points to, which can be one or several sentences, paragraphs or articles.
- Reference type: the exact content should be referred, usually the cases, the applicable objects or the provisions of the referred source.

(Warranty in cases of Compulsory Auctions)
Article 568
(1) The successful bidder at compulsory auction may cancel the contract or demand a reduction from the purchase money against the obligor in accordance with **the provisions from Article 561 through to the preceding Article.**
(2) In the cases set forth in the preceding paragraph, if the obligor is insolvent, the successful bidder may demand total or partial reimbursement of the proceeds against the obligees who received the distribution of the proceeds.
(3) In the cases set forth in the preceding two paragraphs, if obligors knew of the absence of the object or right and did not disclose the same, or if obligors knew of the absence but demanded an auction, the successful bidder may demand compensation for damages against those persons.

(Seller's Warranty against Defects)
Article 570
If there is any latent defect in the subject matter of a sale, **the provisions of Article 566** shall apply mutatis mutandis;provided, however, that this shall not apply in cases of compulsory auction.

H18-1-2
A compulsory auction is also a sale, so warranty is imposed the same as for an ordinary sale.

Figure 4.2: Example of a query and its relevant articles referring to other articles.

Language abstraction

In the example shown in Figure 4.3, while the article needs abstraction to describe all acts applied the same provision, the query, which describe an act happening in practice, is a specialization under the general case in the article ("a Manager" refers to "an individual" or "the rescuer", "principal" refers to "another person", "imminent danger" refers to "getting hit by a car" and "damages" refers to "luxury kimono to get dirty").

While the abstraction is a part of natural language, the level of abstraction on legal domain is quite challenging.

(Urgent Management of Business)
Article 698
If a Manager engages in the Management of Business in order to allow a principal to escape imminent danger to the principal's person, reputation or property, the Manager shall not be liable to compensate for damages resulting from the same unless he/she has acted in bad faith or with gross negligence.

H18-2-2
In cases where an individual rescues another person from getting hit by a car by pushing that person out of the way, causing the person's luxury kimono to get dirty, the rescuer does not have to compensate damages for the kimono.

Figure 4.3: Example of a query and its relevant article describing the general case with high level of abstraction.

Logical structures

Question H20-12-5 showed in Figure 4.4 is one example of logic matching. The answer to the correctness of the statement in the question can be drawn just from the paragraph 2 of Article 298. This also means if we can locate exact predicate from the article to match with the corresponding predicate in the question, unnecessary and noisy text matching can be efficiently mitigated.

(Keeping the Thing Retained by Holders of Rights of Retention)
Article 298
(1) A holder of a right of retention must possess the Thing retained with the care of a good manager.
(2) A holder of rights of retention may not use, lease or give as a security the Thing retained unless he/she obtains the consent of the obligor; provided, however, that this shall not apply to uses necessary for the preservation of that Thing.
(3) If the holder of a right of retention violates the provisions of the preceding two paragraphs, the obligor may demand that the right of retention be extinguished.

H20-12-5
If a holder of a right of retention has consent of the obligor, or if a pledgee has consent of the pledgor, then they each may lease any collateral.

Figure 4.4: Example of a query and its relevant article which composed in "requisite-effectuation" logical structures.

One solution for this can be to build a module just to decompose the structure of

the article into designated parts and then only match the associated parts. This can be modeled as in Figure 4.5.

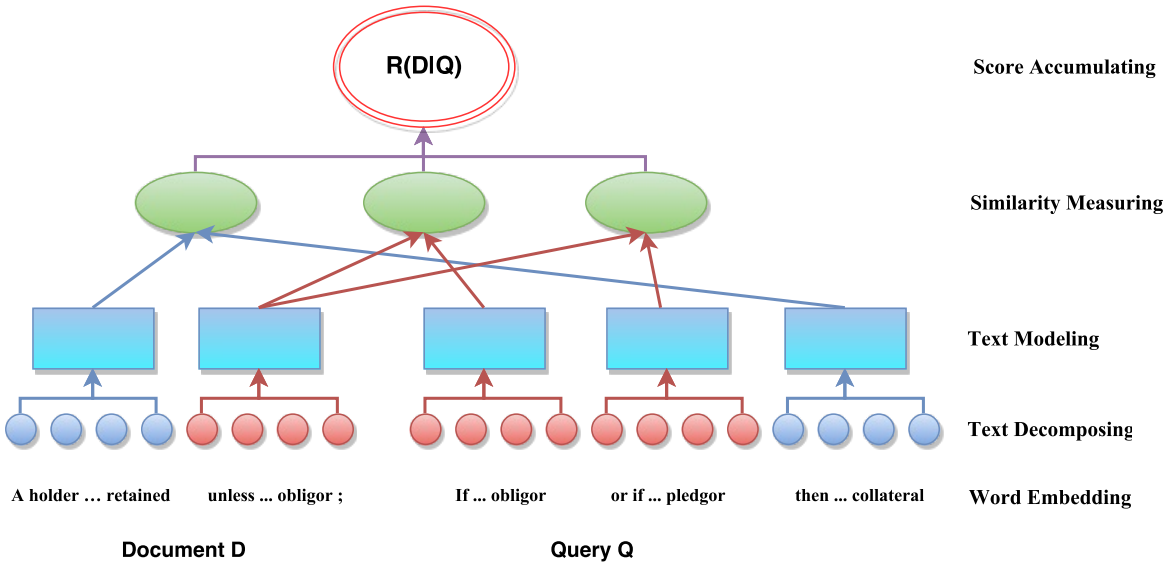

Figure 4.5: Deep learning architecture for structural input

Chapter 5

Conclusions and Future Work

5.1 Conclusions

We have presented our study of deep learning for legal question answering systems. First, we describe our approach to enrich deep learning models with additional features from the training corpus. Second, we evaluate our model and other deep learning models on both open-domain and legal-domain question answering datasets. The results show the huge impact of data characteristics over the performances of the models. Our model produces expected performances on COLIEE 2015 and WikiQA datasets with improvement over TF-IDF and CDSSM, a pure neural-based model. The impact of the additional features shows clearly by the underperformance of our model on TREC 2011 Legal Track where the additional feature is TF-IDF, a very weak feature. Finally, we state the limitations that our model has not yet handled, which also means the room for further improvement.

5.2 Future Work

In our future work, we'd like to resolve the mentioned limitations. First, we're developing frameworks for analyzing legal references, abstraction and structural decomposition. Second, a sophisticated architecture that connects these as features comprehensively will be built. Besides, deep learning with its scalability, flexibility and adaptivity is promisingly appropriate for the model construction.

Bibliography

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

[2] Yi Yang, Wen-tau Yih, and Christopher Meek. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of EMNLP*, pages 2013–2018. Citeseer, 2015.

[3] Ian Goodfellow Yoshua Bengio and Aaron Courville. Deep learning. Book in preparation for MIT Press, 2016. URL `http://www.deeplearningbook.org`.

[4] Li Deng and Dong Yu. Deep learning. *Signal Processing*, 7:3–4, 2014.

[5] Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. Parsing with compositional vector grammars. In *ACL (1)*, pages 455–465, 2013.

[6] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642. Citeseer, 2013.

[7] Jianfeng Gao, Xiaodong He, Wen-tau Yih, and Li Deng. Learning continuous phrase representations for translation modeling. In *In ACL*. Citeseer, 2014.

[8] Vasin Punyakanok, Dan Roth, and Wen-tau Yih. Mapping dependencies trees: An application to question answering. In *Proceedings of AI&Math 2004*, pages 1–10, 2004.

[9] Hang Cui, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. Question answering passage retrieval using dependency relations. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 400–407. ACM, 2005.

[10] Michael Heilman and Noah A Smith. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019. Association for Computational Linguistics, 2010.

[11] Mengqiu Wang and Christopher D Manning. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1164–1172. Association for Computational Linguistics, 2010.

[12] Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. Answer extraction as sequence tagging with tree edit distance. In *HLT-NAACL*, pages 858–867. Citeseer, 2013.

[13] Aliaksei Severyn and Alessandro Moschitti. Automatic feature engineering for answer selection and extraction. In *EMNLP*, pages 458–467, 2013.

[14] Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. Sentence similarity learning by lexical decomposition and composition. *arXiv preprint arXiv:1602.07019*, 2016.

[15] Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. Question answering using enhanced lexical semantic models. 2013.

[16] Zhiguo Wang and Abraham Ittycheriah. Faq-based question answering via word alignment. *arXiv preprint arXiv:1507.02628*, 2015.

[17] Baoxun Wang, Bingquan Liu, Xiaolong Wang, Chengjie Sun, and Deyuan Zhang. Deep learning approaches to semantic relevance modeling for chinese question-answer pairs. *ACM Transactions on Asian Language Information Processing (TALIP)*, 10(4):21, 2011.

[18] Mohit Iyyer, Jordan L Boyd-Graber, Leonardo Max Batista Claudino, Richard Socher, and Hal Daumé III. A neural network for factoid question answering over paragraphs. In *EMNLP*, pages 633–644, 2014.

[19] Di Wang and Eric Nyberg. A long short-term memory model for answer sentence selection in question answering. *ACL, July*, 2015.

[20] Ming Tan, Bing Xiang, and Bowen Zhou. Lstm-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*, 2015.

[21] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[22] Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. Deep learning for answer sentence selection. *arXiv preprint arXiv:1412.1632*, 2014.

[23] Aliaksei Severyn and Alessandro Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 373–382. ACM, 2015.

[24] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2333–2338. ACM, 2013.

[25] Paulo Quaresma and Irene Pimenta Rodrigues. A question answer system for legal information retrieval. In *JURIX*, pages 91–100, 2005.

[26] Oanh Thi Tran, Bach Xuan Ngo, Minh Le Nguyen, and Akira Shimazu. Answering legal questions by mining reference information. In *JSAI International Symposium on Artificial Intelligence*, pages 214–229. Springer, 2013.

[27] Ngo Xuan Bach, Nguyen Le Minh, and Akira Shimazu. Recognition of requisite part and effectuation part in law sentences. In *Proceedings of (ICCPOL)*, pages 29–34, 2010.

[28] Ngo Xuan Bach, Nguyen Le Minh, Tran Thi Oanh, and Akira Shimazu. A two-phase framework for learning logical structures of paragraphs in legal articles. *ACM Transactions on Asian Language Information Processing (TALIP)*, 12(1):3, 2013.

[29] Huu-Thanh Duong and Bao-Quoc Ho. A vietnamese question answering system in vietnams legal documents. In *IFIP International Conference on Computer Information Systems and Industrial Management*, pages 186–197. Springer, 2014.

[30] David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[31] Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[32] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[33] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[34] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521 (7553):436–444, 2015.

[35] Itamar Arel, Derek C Rose, and Thomas P Karnowski. Deep machine learning-a new frontier in artificial intelligence research [research frontier]. *IEEE Computational Intelligence Magazine*, 5(4):13–18, 2010.

[36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[37] Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*, 2014.

[38] Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 373–374. ACM, 2014.

[39] Yoon Kim. Convolutional neural networks for sentence classification. *EMNLP 2014*, 2014.

[40] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. In *Document retrieval systems*, pages 132–142. Taylor Graham Publishing, 1988.

Publications

[1] Tran, Duc Vu, Phan, V. A., Trieu, H. L., and Nguyen, L. M. (2015). An approach for retrieving legal texts. In *Ninth International Workshop on Juris-informatics, COLIEE*, pages 231–244.

[2] Tran, Q. H., Tran, Vu, Vu, T., Le Nguyen, M., and Pham, S. B. (2015). Jaist: Combining multiple features for answer selection in community question answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval*, volume 15, pages 215–219.