Label Semantics Embedding and Hierarchical Attentions for Text Representation Learning

Min Fuzhou

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Master of Engineering

2023

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

07-07-2022

.....

Date

.....

Min Fuzhou

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

07-07-2022

Date

Mao Kezhi

Prof. Kezhi Mao

Authorship Attribution Statement

This thesis does not contain any materials from papers published in peer-reviewed journals or from papers accepted at conferences in which I am listed as an author.

07-07-2022

.....

Date

.....

Min Fuzhou

Acknowledgements

I wish to express my sincere appreciation to my advisor Prof. Mao Kezhi for his guidance and continuous support throughout my master study for providing not only the lab resources but also mental care. Also I would like to express many thanks for my labmates for their meticulous guidance as well as friendly help to my research work and my daily life. I am grateful for NTU for providing special support during this COVID-19 period. Furthermore, I would like to thank my parents and my dear friends who teach me how to stay positive and peaceful to cope with ups and downs in lifetime.

Abstract

Text classification is one of the most widely-used and important NLP (Natural Language Processing) tasks that aim to deduce the most proper pre-defined label for a given document or sentence, such as spam detection, topic classification, sentiment analysis, and so forth. One of the key steps of text classification is text representation. With the rapid development of machine learning, neural network models such as Convolutional Neural Networks and Recurrent Neural Networks have been commonly employed for achieving text representation learning. Currently, in most existing text classification models, the labels of the classification task used by models are always represented as one-hot vectors, without the dependence on the semantics of text data itself. For example, in a sentiment analysis task, the labels "positive"and "negative "are encoded as [1,0] and [0,1], and the semantic information of the labels is not made full use of. However, the semantics of labels are highly related to the text classification task. Therefore, the information contained in labels can not be disregarded.

In this thesis work, we propose a Label Embedding-based Hierarchical Attention Model (LE-HAM) incorporating the semantic information of labels. We implement the semantic information of labels by jointly embedding the labels and words. Further, to solve the other problem, the structure of a single attention mechanism does not achieve satisfactory results for data with weak signals. We introduce a model that includes a two-level attention framework based on the label semantics embedding. This hierarchical attention structure aims at the text data with weak signals in the tasks, seeks to exploit the label information to choose the key sentences first, then use only these selected sentences combined with label information to build a text representation. Therefore, the majority of noises can be removed. The main novelty is that this method creatively uses the sentence-chosen mechanism. In this way, the model can find the key sentences when there are many noises in the text, then keywords can also be located more efficiently, and the accuracy of the text classification task on those "weak signal"datasets can be improved.

Key words: label embedding, attention, hierarchical, weak-signal

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

Text classification is one of the main tasks in Natural Language Processing (NLP) with wide applications such as intent recognition [9][10], sentiment analysis [11][12][13], spam detection [14][15], and news filtering [16]. Text representation is a decisive intermediate step [17]. Due to deep learning's unprecedented success, it is desirable to use deep learning instead of imprecise and time-consuming manual methods in the text representation procedure to produce more objective and reliable results. Therefore, research has boomed in this field in the past few decades. As revealed in recent surveys, there have been many advanced methods proposed.

Traditional methods employed for text representation aim at conveying pre-processed text data in a way that minimizes information loss and makes it easier for computers to "understand". Some examples are Bag-Of-Words (BOW) [18], N-gram [19][20], Term Frequency-Inverse Document Frequency (TF-IDF) [21][22].

Deep learning models proposed recent years have achieved outstanding performance compared with traditional models [1][23][24]. Based on deep neural networks which enable

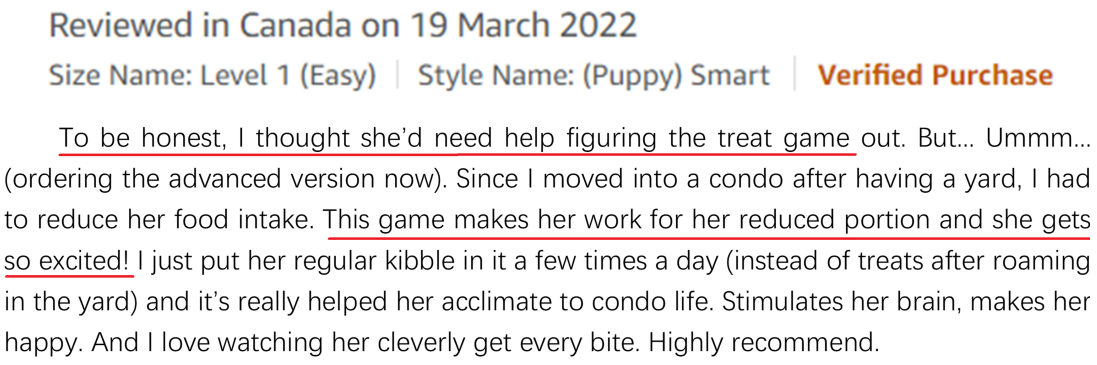
models to extract to learn higher-level features in the way of simulating the human brain. CNN (convolutional neural network) [25] and RNN (recurrent neural network) [26] are the two of the earliest deep learning models for text representation which are widely used and improve performance on text classification tasks. Then, to solve the gradient vanishing problem of the RNN method, LSTM (Long Short-Term Memory) [27][28] came out. As the improved variant of RNN, LSTM can alleviate the gradient vanishing problem effectively, deal with longer text data and capture contextual information better.

To improve the flexibility and interpretability of text representation, attention mechanisms [29][4] have been introduced and become an essential part of text classification models. Attention methods are employed for capturing the relevance which makes highly important contributions to the classification tasks. The training of the attention module is independent of the distance between the sequence's elements, therefore it can provide complementary information to the models based on RNN / CNN whose dependencies are related to distances.

More recent years have seen an epoch-making turning point in the research domain of text classification and other NLP domains because of the appearance of pre-trained language models [30] such as BERT (Bidirectional Encoder Representation from Transformers) [6]. BERT is well-known and widely employed for text classification tasks, and achieves more inspiring performance than the former models. Unsupervised methods are typically employed to achieve the goal that "letting machines understand semantics", pre-trained models are involved to mine semantic information and then construct pre-training targets automatically. Pre-trained language models can boost the performance of text classification tasks significantly for the reason that they can learn global semantic information effectively.

Word embedding [31][32][33] is a highly effective tool as well as a basic element widely used in NLP tasks including text classification, as an intermediate representation aims to capture semantic regularities between words. Syntactic and semantic meanings of

words are obtained from a large corpus and then both of them are embedded to produce a real-valued vector used for learning the text representation. Recently, several surveys [34][35][36][37] proved that the quality of the word embedding is crucial for the performance of text classification. In recent years the word embedding concept has been expanded to obtain embeddings that capture the semantic information and connection of word sequences (for example, phrases, sentences, and paragraphs) [38][39][40].


Reviewed in Canada on 19 March 2022

Size Name: Level 1 (Easy) | Style Name: (Puppy) Smart | Verified Purchase

To be honest, I thought she'd need help figuring the treat game out. But... Ummm... (ordering the advanced version now). Since I moved into a condo after having a yard, I had to reduce her food intake. This game makes her work for her reduced portion and she gets so excited! I just put her regular kibble in it a few times a day (instead of treats after roaming in the yard) and it's really helped her acclimate to condo life. Stimulates her brain, makes her happy. And I love watching her cleverly get every bite. Highly recommend.

FIGURE 1.1: An example of "noisy"text data

Various advanced models have been proposed and exploited by plenty of research works that achieved improvement in text representation. Even though they have obtained inspiring results on text classification tasks, there are still many challenges that we are facing. For example, "weak-signal"text is common in various NLP tasks. For instance, we can occasionally encounter user comments with this characteristic when we are browsing the user reviews of a certain product on the Amazon website. Those reviews may give a lengthy, trivial story about their daily life or mention other products and so on, before using only one or two sentences to evaluate this product. An example is shown in Figure 1.1: Only two short sentences with red lines allow us to make it clear that the user is talking about a pet toy product. The majority of this user's command makes it easier to regard it as a pet food review rather than a toy review. The aforementioned models and methods always work unsatisfactorily when applied to those text data that need to be categorized, such as determining which category of product the user is talking about in this review. Those models frequently take adequate large datasets for training and

learning to achieve a good performance on these "weak-signal"text data. Additionally, the methods based on the aforementioned models fall short when it comes to locating the key phrases and keywords in text data for later use (for instance, manufacturers would save a ton of time if they could identify key phrases in lengthy reviews that reflect user feedback on their products).

On the other hand, despite the fact that numerous deep learning models have rapidly, widely, and successfully demonstrated their effectiveness, it is not difficult to find that those deep learning models all ignore the important role that labels are playing in text classification [41]. In text classification tasks, labels are not independent of the text data. It can even be said that the connection between them is very close and crucial since the labels of text data are also words or phrases that have semantics. However, the labels are often transformed into one-hot vectors that are only used for the learning of the classifier, and the semantic information of the labels is not made full use of. To address the aforementioned problems, the label embedding-based methods have been proposed in recent year [42][43][44]. The effectiveness of label embedding has been demonstrated in various tasks of different domains [45][46][47]. For text classification, the semantic information of labels can be used to find those words in text data that are most relevant to the classification task. A method that constructs the text representation in the label-word joint embedding space where the word-label compatibility can be used directly was proposed and has shown the inspiring result [8]. Moreover, label embedding method has been employed in the multitask learning [48][49] and heterogeneous networks [50], etc.

1.2 Contributions

To address the problems of the classification tasks on the aforementioned "weak signal, strong noise"text data. Two novel methods are proposed in this thesis based on the label embedding method. They are built up for capturing the key sentences and keywords, retaining key information to the greatest extent possible while removing noise. Therefore,

the high-quality text representation for the text classification tasks can be constructed. The details of the Sentence Selection part will be explained in Chapter 3 and the adaptive method for Sentence Selection will be introduced in Chapter 4.

This thesis makes the following contributions and works:

- Implement a computation strategy for text data that has undergone sentence segmentation and other data preprocessing stages, based on the joint embedding of words and labels method. Hence, reliable weight values that can reflect the importance of the sentence can be obtained by computing the compatibility between words and labels and the relevance of sentences to the classification task. The more relevant this sentence is, the larger the weight value is.

- Design an adaptive selection mechanism to choose sentences from the text data with the most relevance to the classification according to the word-label compatibility and sentence weight values obtained. Only those selected sentences chosen from the text in the sentence-selection step are involved in the construction of the text representation. Therefore, noise in the text data can be reduced and valuable information can be captured from the noisy data. As a result, the next phase can result in a text representation of higher quality.

- Propose to learn the final text representation that is input into the classifier using both the contextual information and the compatibility between words and labels.

Combined with the methods summarized above, this model we propose can not only effectively classify noisy data, but also locate key sentences and keywords in text data, which has practical value in future applications.

1.3 Outlines

In this thesis, Chapter 1 introduces the background of my research work as well as the motivation of this thesis. It provides an overview of the development of deep learning models and methods employed on text classification tasks.

The literature and earlier studies on the understanding of some typical and significant deep learning networks and attention mechanisms are comprehensively covered in Chapter 2. In addition, a brief introduction is given to the label embedding-based approaches proposed recently.

In Chapter 3, a detailed explanation of the proposed label embedding-based attention model is given. The network is designed into a hierarchical structure where the attention mechanism is employed to both sentence level and word level. In Chapter 4, further in-depth ideas and attempts at adaptive sentence selection mechanisms are presented. The preliminary results in the field test are demonstrated in Chapter 5.

Chapter 6 concludes the current progress and briefly introduces ideas for future work.

Chapter 2

Literature Review

This chapter reviews relevant literature on several representative neural network-based NLP models. Attention mechanism and label embedding method are also detailedly introduced. They are of great significance to the development of text representation tasks. Section 2.1 introduces the existing typical deep learning-based models applied in the text representation. Section 2.2 describes the attention mechanism that is regarded as one of the most popular methods employed in text classification. Methods based on attention make a significant improvement to performance and also improve interpretability. The theoretical basis and implementation details of the label-word joint embedding method are explained in Section 2.3. This label embedding-based idea allows our proposed model to capture the relevance of words to the classification task. Some typical pre-trained language models are reviewed in Section 2.4.

2.1 Deep Learning Models

Natural Language Processing is known as a way to enable computers to analyze the human language and then understand and derive the meaning smartly and effectively. There are many challenges in the improvement of the NLP application, considering its target of

focusing on the interactions between computers and human language. The programming language used for communication between humans and computers is highly structured, accurate, and unambiguous. While the human language is always imprecise and ambiguous with the linguistic structures that can depend on dialects, cultural context, and many other variables[1]. Therefore, inspired biologically by the structure of the human brain, deep neural networks-based approaches enable computers to learn higher-level features from observed data automatically. No human intervention is needed for the text classification tool. Deep learning models consisting of neural networks proposed in the last decade have achieved inspiring results on NLP tasks far more than traditional models.

Text classification is one of the most crucial branches of Natural Language Processing, aiming at classifying a piece of text content into one or multiple categories. The vital intermediate step of text classification is text representation. CNN (convolutional networks), RNN (recurrent neural networks), LSTM (long short-term memory, which is based on RNN), and so forth, have been applied successfully in the text representation.

2.1.1 RNN

The most significant characteristic of the Recurrent Neural Network for NLP tasks is that it can capture the remote dependencies in sequence by recurrent calculation [51]. In earlier neural network models such as MLP-based models [52] (MultiLayer Perceptron), there is no connection between inputs and outputs. Since the words in the text data are highly relevant to each other and the contextual information is extremely important, the performance of those traditional neural models is unsatisfactory. In order to predict the next word in a sentence, it is necessary to know the words that came before it. Hence, RNN-based models are broadly employed in text classification tasks because they carry out the same task for each element of a sequence, with the output dependent on previous computations. It can be said that RNNs have a “memory”that stores information about previous calculations. In Figure 2.1, a typical RNN is illustrated.

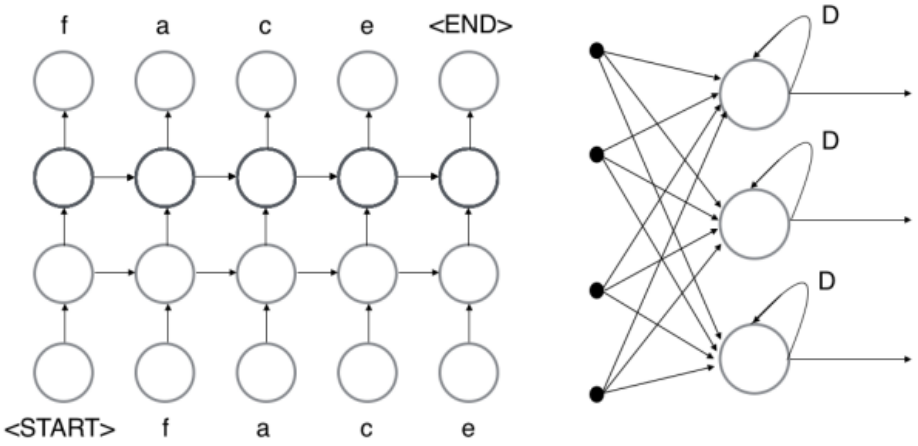


FIGURE 2.1: Recurrent network [1]

First, each input word is transformed as a unique vector via the word embedding method. The embedding vectors of words are then fed into the RNN unit individually, and the dimension of the output vector and the input vector are the same. The output of RNN units is then passed to the hidden layer. Different parts of RNN models share the same parameters and the weight of each input is also the same.

2.1.1.1 Bidirectional RNN

In the human linguistic system, predicting a missing word in a sentence requires not only the preceding information but also the following information, which is commonly referred to as contextual information. Inspired by this feature of human language, the idea on which Bidirectional RNN is based is that: the output at a certain time t depends on both previous elements and future elements in the sequence. Figure 2.2 shows the structure of a Bidirectional RNN composed of two RNNs stacked with each other on their top, the hidden states of the two RNNs are used to calculate the output.

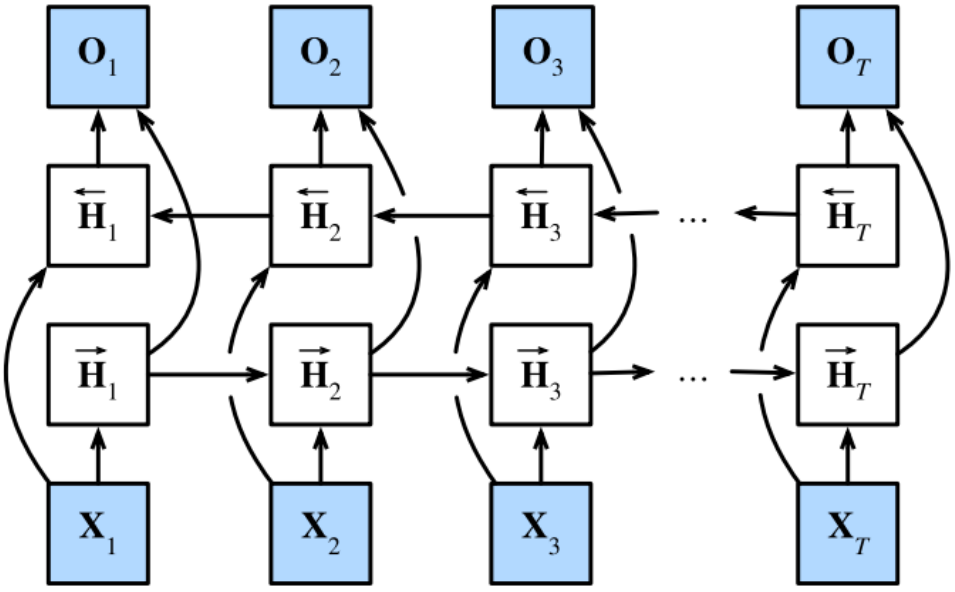


FIGURE 2.2: Bi-direction RNN [2]

2.1.1.2 Deep RNN

As shown in Figure 2.3, the only difference between Deep RNN and Bidirectional RNN is that there are multiple RNN layers in Deep RNN but Bidirectional RNN only has two. Using more RNN layers improves the learning capacity of models but the requirement of a large amount of train data is also necessary.

2.1.1.3 LSTM

In theory, RNNs can process and use the information contained in a sequence of arbitrary lengths. However, it is limited to only a few steps back practically. Due to the RNN's backpropagation where the weights are adjusted according to gradients as well as continuous derivative multiplications method to obtain the weights [53], it may cause what is called a gradient vanishing problem after the continuous multiplications of the derivatives when they are extremely small.

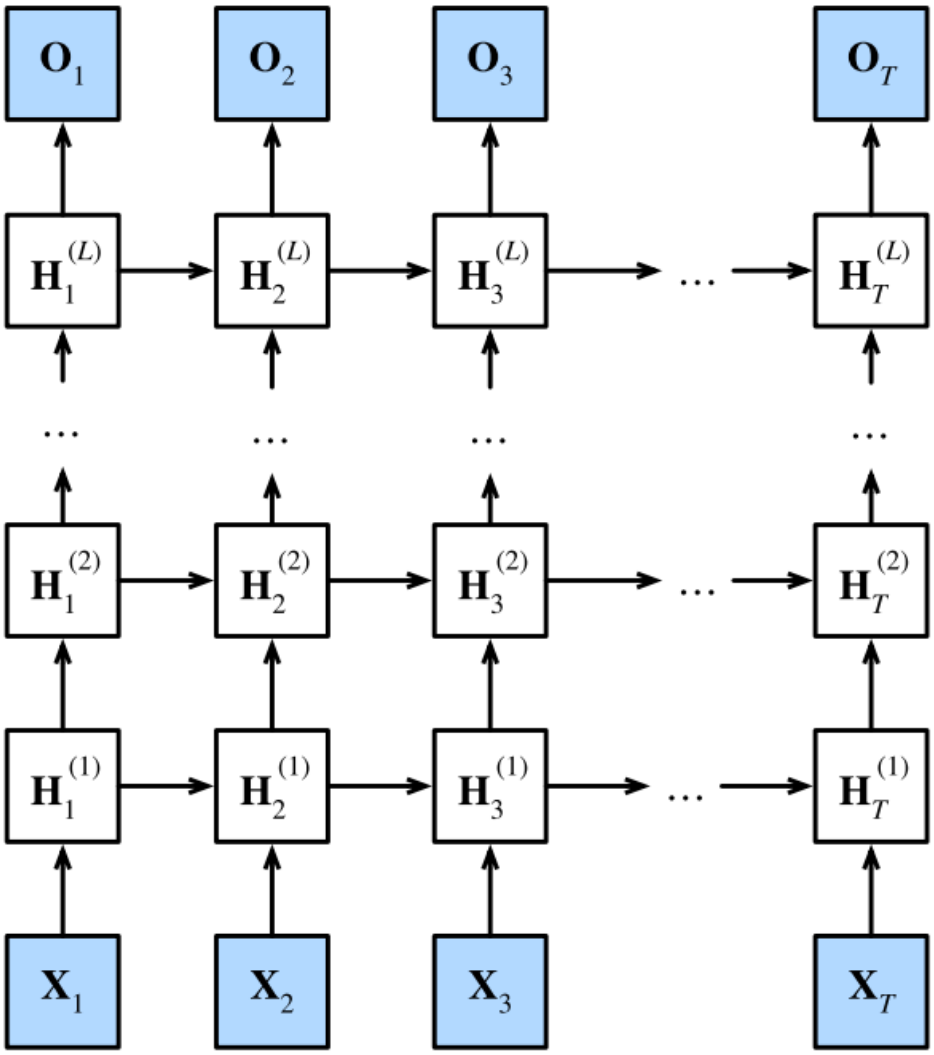


FIGURE 2.3: Deep RNN [2]

Based on RNN, LSTM (Long Short-Term Memory) is proposed as a significant enhancement to effectively alleviate the gradient vanishing problem. Except that the internal operation of the LSTM unit is more complicated, the architecture of LSTM doesn't have a fundamental difference from RNNs. Figure 2.4 illustrates the structure of a LSTM unit. Both the previous state and the current input are used as the input of an LSTM unit [54]. The input is denoted as x_t, h_{t-1} and h_t denotes respectively the hidden state of this LSTM unit when it is at time t and $t-1$. The cell state of this LSTM unit at time t and $t-1$ is denoted as c_t and c_{t-1}. c_t and h_t are both transferred as the inputs to the next time step. Three gates [55] are in an LSTM unit that are used as filters for the control of the information flow to adjust the cell state and the hidden state, including input gates i_t,

forget gates f_t and output gates o_t: The input gate controls the new information which is about to be stored in the cell state; The forget gate decides what is going to be removed from the cell state; The function of the output gate is specifying the information in the cell state that is used as the output. LSTM can capture and learn the dependency in both short-term and long-term series thanks to its complex structure with three gates.

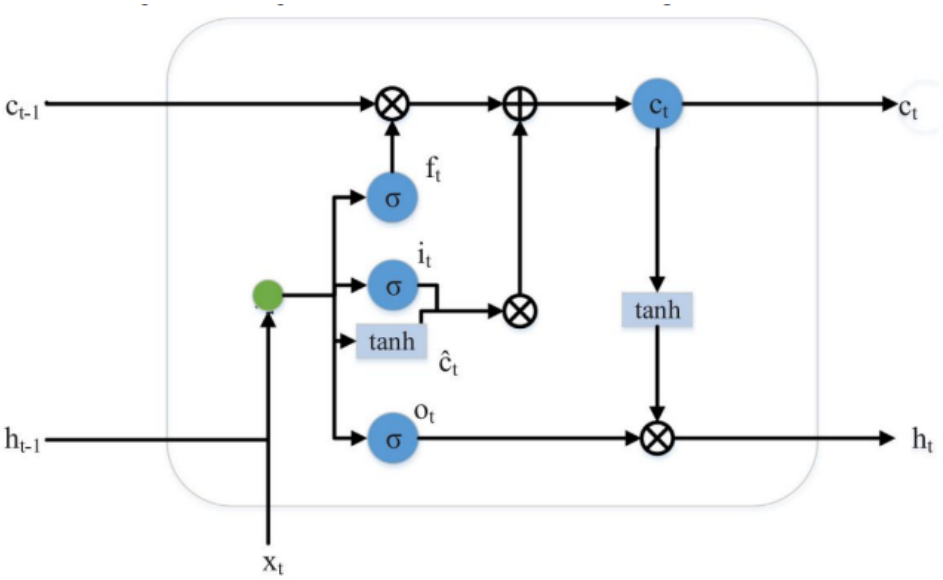


FIGURE 2.4: A LSTM cell [3]

2.1.1.4 GRU

GRU also contains a gated structure resembling the LSTM. The architecture of the GRU is simpler and the parameters are also fewer than LSTM whose structure is so sophisticate that causes a long-time training process [56]. The structure of a GRU unit is illustrated in Figure 2.5. There is a two-gate structure in a GRU unit: the update gate and the reset gate. The cell state c_t and the hidden state h_t merge into one and the update gate denoted as z_t. z_t controls the degree to which the information of c_{t-1} and h_{t-1} at the time $t-1$ transferred to the current time step t. The reset gate determines how much of the previous state's information is brought into the current state $\hat{c}_t$.

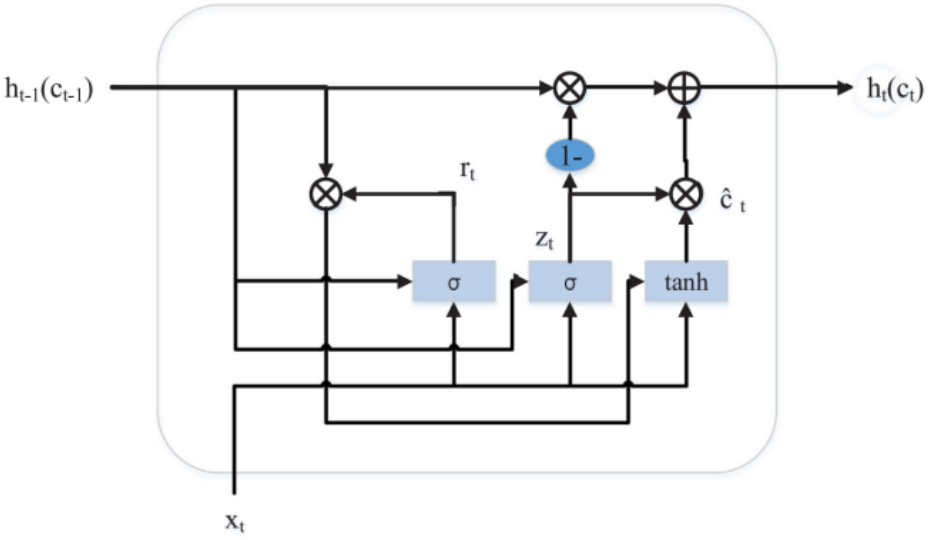


FIGURE 2.5: A GRU cell [3]

2.1.2 CNN

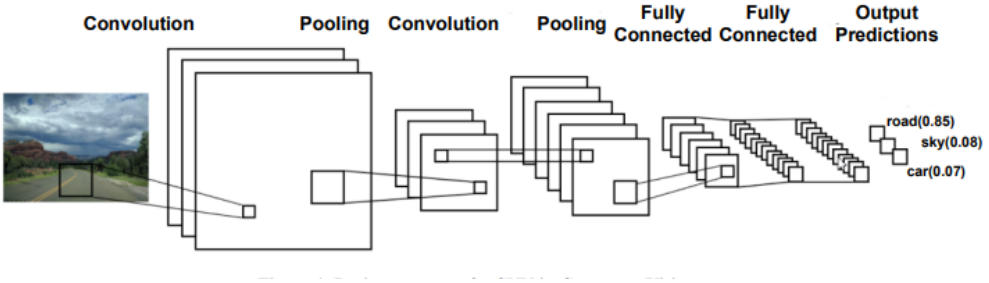


FIGURE 2.6: Basic structure of a CNN in Computer Vision [1]

CNN (Convolutional Neural Network) [57] is commonly related to Computer Vision. In particular, CNN has significant implications for a major breakthrough in image classification tasks, since its convolution filters enable it to extract features efficiently from images. CNN has become the heart of the majority of nowadays' computer vision systems. Figure 2.6 illustrates a basic structure of CNN when it is applied to computer vision tasks.

For the application of CNN in NLP tasks [58][59], it also has resulted in many encouraging results. Unlike the conventional neural network, which is referred to described as a fully-connected network since every input neuron is connected to every output neuron in the following layer. While the way of CNN obtains the output is by employing the

convolution operations on the input layer. In more detail, it is each region instead of each neuron on the input layer that is connected to each neuron in the output layer, which means that it can cause the local connection. The filters on every layer considered as extracting features are different. If the feature exists, the value of the output of this region where the corresponding filter is applied can be large, while the filter yields a small value in other regions. There are always hundreds even thousands of filters. The results of those filters are then combined to extract textual information from different perspectives.

The pooling layers play a crucial role in CNN, and the output of the convolution layers is the input of the pooling layers. The operation of pooling layers for subsampling their inputs can be max operation, median operation, and average operation [60], etc., and the max-pooling [61] is the most commonly used. In the application of CNN on the text classification tasks, the size of sentences is variable so is the number of filters. Therefore, pooling is significantly required since the output of the pooling operation is a fixed-size matrix, it allows us to obtain the output as the input of the classifier that has the same dimension. In addition, max-pooling can preserve the information about whether the feature appears in this region due to the detective characteristic of the filters. Although the global information in this region (about where it exactly appears) is lost, the filter can capture the local information. The values of CNN's filters are learned and adjusted during the training process based on the tasks.

When CNN is applied for image processing, as mentioned above, it automatically learns filter values depending on the task. For instance, in an image classification task, the first layer of CNN can be trained to use the raw pixels to detect the image edges; the edges information then is used in the second layer to obtain the simple shape; then the higher-level features such as the facial information are collected from the shape information; the last layer uses those higher-level features to perform classification which is also called a classifier.

As Figure 2.7 illustrates, the majority of NLP tasks, such as text classification, require sequences or documents as input, as opposed to computer vision tasks, which require

picture pixels. Each token—typically a word—is represented as an embedding vector in order to make the text representation resemble the image representation. As a result, the text data, such as sentences, are represented by a matrix where each row represents a word.

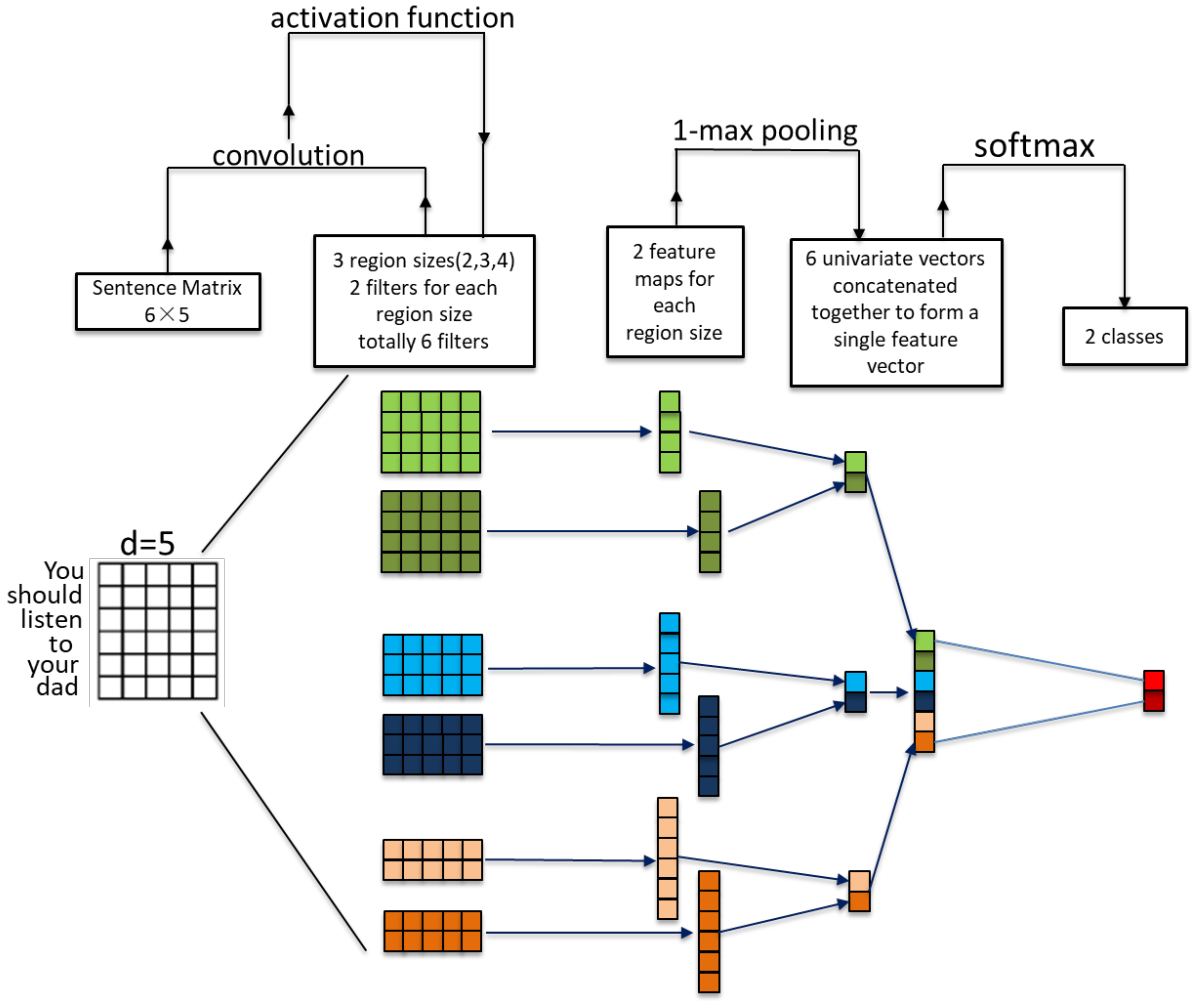


FIGURE 2.7: How the CNN works in NLP tasks [1]

2.2 Attention Mechanism

Although the traditional neural network models like CNN and RNN have achieved inspiring improvement on many tasks in various domains, these models all have the shortcoming

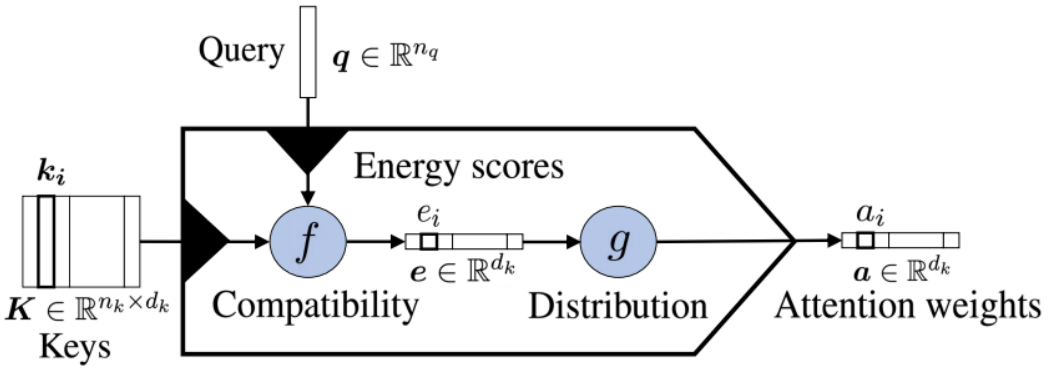


FIGURE 2.8: The structure of the "core"attention model [4]

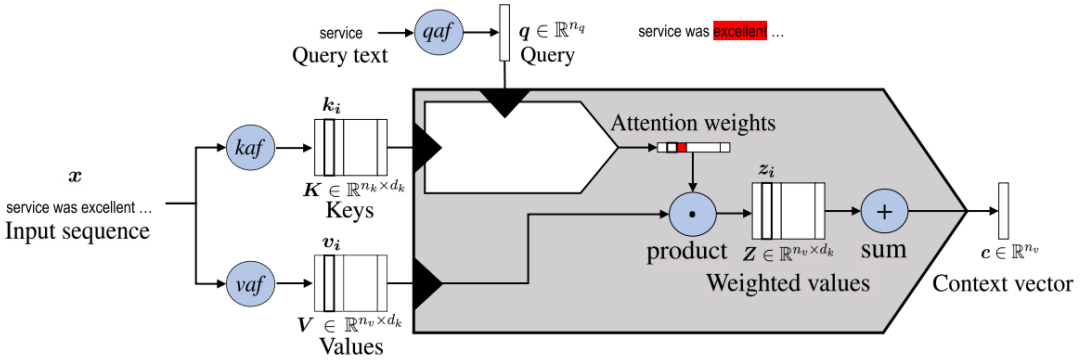


FIGURE 2.9: The structure of the general attention model [4]

that they are not intuitive and interpretable enough. For example, when the model classifies a document data wrongly in a text classification task, it cannot explain the reason for giving the wrong predicted label to the data. Inspired by human intuition, there have been various attention models proposed recently, and the basis of their approaches is the same. The basic idea of the attention mechanism is to focus more on target regions (for example, a certain part of an image [62] or a paragraph of an article) that contain important information while giving less attention to those less important parts (such as the surrounding images and some trivial sentences) [63]. Meanwhile, the focus, as well as the extent of attention, is being adjusted over the training process.

In short, "attention"in the attention mechanism can be generally regarded as a weight vector. Predicting an element such as a label or a sentence requires combining information from other elements in the data. Therefore, it is essential to estimate the degree to which these components are related to the element to be predicted. The attention

mechanism gives larger weights to those with higher relevance, and then the value of the weighted sum is used as an approximation to the target.

Attention-based neural network models have proven successful on many NLP tasks with satisfactory results. In a text classification task, the labels given to the text such as "entertainment"or "economy", "positive"or "negative"are always relevant to some elements (keywords or key sentences). These key parts in the text need to be taken into consideration by models. In the text classification domain, the input is typically a series of textual elements. On the other hand, the attention mechanism, as part of the model architecture, is used to dynamically highlight the information of the input data as the training process progresses. As mentioned above, the attention method can be regarded as a selection approach that generates a weight vector for the input data, more important elements correspond to larger weight values. Hence, the attention mechanism can be directly applied to the raw text in the input layer or the text representation on a higher level on a hidden layer. The attention mechanism adjusts its focus without dependency on the distance between the textual elements. Hence, it is of crucial benefit to the tasks, especially on long text data such as long sentences or documents [64]. Moreover, the neural networks with attention mechanisms not only improve the classification result but also give insight into those parts of the text data that play a role in classification decisions and which parts do not.

The basic attention mechanism introduced in this thesis is on the basis of the attention model proposed by Vaswani *et al.* [5] of which the core part has been shared by nearly all attention-based models in recent surveyed literature. Although some additional structures do not present universally, they still can be found in the majority of the existing attention models.

Figure 2.8 illustrates the core part of the attention-based model and a typical attention model is shown in Figure 2.9. The terms and symbols are listed in Table 2.1.

TABLE 2.1: Notation of the attention model

Symbol	***Name***	***Definition***
$\boldsymbol{x}$	Input sequence	The raw input consists of word sequence
$\boldsymbol{K}$	Keys	A matrix consists of d_k vectors whose length is n_k that contributes to the computation of attention weights, the i-th vectors of K is denoted as k_i, K $\in \mathbb{R}^{n_k \times d_k}$
$\boldsymbol{V}$	Values	A matrix consists of d_k vectors whose length is k_i that contributes to the computation of attention weights, the i-th vectors of K is denoted as v_i, corresponding k_i, $V \in \mathbb{R}^{n_k \times d_k}$
$\boldsymbol{q}$	Query	A n_q-length vector, one of the element in which attention weight computed, $\mathbf{q} \in \mathbb{R}^{n_q}$
kaf qaf vaf	Annotation functions	Encode the input $\boldsymbol{x}$ and query $\boldsymbol{q}$, the respective results of functions are $\boldsymbol{K}$, $\boldsymbol{q}$, $\boldsymbol{V}$
$\boldsymbol{e}$	Energy scores	A d_k-length vector of which element denoted as e_i(a scalar called "energy score"), computed by the compatibility function, reflects the relevance to the corresponding key k_i, $\boldsymbol{e} \in \mathbb{R}^{d_k}$
$\boldsymbol{a}$	Attention weights	A d_k-length vector of which element denoted as a_i(a scalar called "attention weight"), computed produced by the attention model, reflects the relevance to the corresponding key k_i, $\boldsymbol{a} \in \mathbb{R}^{d_k}$
$\boldsymbol{f}$	Compatibility function	Measures the relevance of $\boldsymbol{K}$ to $\boldsymbol{q}$ with the output vector $\boldsymbol{e}$, $\boldsymbol{e} = \boldsymbol{f}(\boldsymbol{q}, \boldsymbol{K})$
$\boldsymbol{g}$	Distribution function	$\boldsymbol{a} = \boldsymbol{g}(\boldsymbol{e})$, transfers the "energy score"into "attention score".
$\boldsymbol{Z}$	Weighted values	A matrix consists of d_k vectors whose length is n_v, denotes the representation from $\boldsymbol{V}$ weighted by $\boldsymbol{a}$, the i-th vectors of Z is denoted as z_i, $\boldsymbol{Z} \in \mathbb{R}^{n_k \times d_k}$
$\boldsymbol{C}$	Context vector	A n_v-length vector denotes the compact representation of $\boldsymbol{Z}$, $\boldsymbol{c} \in \mathbb{R}^{d_k}$

The core attention mechanism maps a sequence $\boldsymbol{K}$ called the ***keys*** which consists of d_k vectors k_i. a is the weights distribution whose element is a_i and the size is d_k. The features of input data are encoded as $\boldsymbol{K}$ on which the attention is calculated. $\boldsymbol{K}$ can be an embedding matrix of sentences or a document, the hidden states of a neural network, or multiple representations. For example, both the word embedding and position embedding, of the same object.

In most instances, $\boldsymbol{q}$ which is called ***query*** is the other input. If $\boldsymbol{q}$ is defined, it is regarded as the reference used to estimate how much the elements are related to the classification. Those considered more relevant will be given more "attention". Hence the parts of the input which contain important information will be emphasized. If $\boldsymbol{q}$ is not defined, it can be thought of as the features which are more relevant inherently that need to be emphasized. For example, the hidden states of an RNN can be used as ***query***.

As the *compatibility function* f (Eq 2.1), the keys $\boldsymbol{K}$ and the query $\boldsymbol{q}$ are used as two inputs to calculate a d_k-length vector $\boldsymbol{e}$. $\boldsymbol{e}$ is composed of e_i called energy score.

$$\mathbf{e} = \boldsymbol{f}(\boldsymbol{q}, \boldsymbol{K}) \tag{2.1}$$

Then the attention weights $\boldsymbol{a}$ can be obtain from $\boldsymbol{e}$ through g what called *distribution function* (Eq 2.2)

$$\mathbf{a} = \boldsymbol{g}(\boldsymbol{e}) \tag{2.2}$$

The SoftMax function is employed as the distribution function most commonly. Finally, $\boldsymbol{a}$ is the output of the attention mechanism that represents how strongly each element in the input data is related to the classification task.

As is illustrated in Figure 2.9, the weighted values which are used as the text representation/context vector $\boldsymbol{C}$ in the text classification tasks are obtained by the sum of the values $\boldsymbol{V}$ (Eq 2.3):

$$\mathbf{C} = \sum a_i v_i \tag{2.3}$$

2.3 Word Embedding

It is well-known that recent advances in deep learning have resulted in significant progress on many NLP tasks with various advanced deep learning models proposed. Due to the fact that processing the semantic analysis of the text is the essence of the NLP task, the high-quality representation of text semantics is of great significance to the results. Hence, word embedding which is also called word representation or continuous / distributed representation is playing a vital role [65][66]. It has been shown in several surveys [33][67] recently that the effectiveness of word embedding is highly related to the performance of the deep learning method on text classification.

Word embedding is the dense representation of a word that is embedded with the meaning of the word in a vector space. The semantics between words are closer, and the distance between their corresponding embedding vectors will be closer. Therefore, word embedding methods can measure the similarity between words to capture the syntactic and semantic relationships in text data [37]. Word embedding methods are typically based on the *distributed hypothesis*: in a text such as a document, words that appear in similar contextual positions are more likely to have similar semantics [68]. The embedding vector of a word is built on the basis of its context on a large corpus. *Context-counting* approaches [69][70][71][72] generate word embedding vectors initially by collecting the word co-occurrence frequencies with some form of matrix factorization typically involved. Later *context-predicting* [34][65][73] approaches which are also called neural approaches were proposed and used on word embedding. By this neural approaches, the representation of words is also the component of the parameters trained by the model, targeting to predict some data distributional features.

Although there was a debate on which approach is more desirable [74], the two approaches [67] have been thought that they embody different ways to pursue the same objective [75]. These two methods inherently have different performances because of the difference in the designs and the settings of hyperparameters [76]. The main disadvantage

of context-counting approaches is that an entire co-occurrence matrix is needed that will become impractical on the large-amount dataset. Conversely, this problem does not exist in the neural approaches (context-predicting approaches). This is because the counts are incremental intrinsically by using stochastic optimization (which is adopted as a standard practice). Consequently, context-predicting methods are now applied in distributional semantics learning dominantly.

Word2vec [65] is a representative and popular neural approach for constructing word embedding vectors. In word2vec, a two-layer neural network is trained to make a prediction for words corresponding the center of a sliding context window (CBOW: *continuous bag-of-words* or the context of a central word [77], SG: *skip-gram* [77]). The first layer indexes the word embeddings that is acting as a look-up table, the input and the output are in the forms of one-hot vectors. Due to its hierarchical softmax and negative sampling [77] method, word2vec has achieved success in significantly increasing the speed of computation. Word2vec can therefore process large amounts of text data. GloVe [73] is also popular for obtaining word embeddings and performs better in reconstructing the co-occurrence probabilities between word pairs estimated by the dot product. The gloVe has been proven to outperform word2vec on many NLP tasks, both these two methods have been employed to produce mass word embeddings that have been publicly available and widely used.

2.4 Pre-trained Models

As previously stated in Section 2.1 and 2.2, thanks to the rapid development of deep learning, there have been various neural network-based methods applied in NLP (Natural Language Processing) tasks. Some examples are the well-known RNNs, CNNs, and attention mechanisms. A major advantage of methods based on neural network models is that these models can solve feature engineering problems to a large extent. Compared with the non-neural-based traditional models that rely heavily on hand-crafted features

[30] (for example, sparse lexical features like n-grams). The neural network models implicitly represent the semantic and syntactic features of text data by using the form of dense and low-dimensional vectors (also called distributed representation) [78]. These representations will be learned during model training according to specific tasks. Hence, neural network-based methods can make the processing of NLP tasks more intelligent and efficient. Although the application of neural-based models in the NLP field has achieved success, the improvement in the performance is still not very significant. This is mainly because neural networks contain a large number of parameters that need to be trained. However, the size of the current NLP datasets for supervised learning tasks is often insufficient for the training of these parameters, which may lead to overfitting [79]. Therefore, to avoid this overfitting problem, early neural models are usually designed to be shallow (no more than three-layer) [30]. Conclusively, these models can not generalize well in practice.

Substantial more recent literature has revealed that pre-trained models which learn general representations of language on a large size corpus. Pre-trained models contribute to the downstream work of NLP tasks because they avoid training a new model from scratch. The word embedding methods such as GloVe [73] that we briefly introduced in Section 2.3 are early pre-training models with the goal to learn high-quality word representations. These models themselves do not participate in the training of downstream tasks, so they contribute little to the effectiveness of the model. Although they can produce word embeddings that contain semantic features, they cannot capture higher-level contextual features (*e.g.*, anaphora, syntactic structures).

Recent years have witnessed the emergence of more advanced deep learning models represented by Transformers [5] and the continuous enhancement of training algorithms. Focusing on capturing contextual information to construct high-quality word embedding representations which contain high-level information of context, the architectures of pre-training models have been advanced from shallow to deep. More advanced and powerful pre-trained models such as ELMo [80], OpenAI GPT [81], BERT [6] have become available.

In this section, we will briefly review Transformer and BERT that have had a significant impact on the development of the NLP domain in recent years.

2.4.1 Transformer

Transformer has caused an uproar in the field of modern deep learning in recent years because of its effectiveness in various domains such as NLP (Natural Language Processing) and CV (Computer Vision) [82]. In the NLP domain, Transformer architecture has been playing an indispensable role. For Transformer-based models provide a substantive improvement in effectively capturing contextual connection and other aspects of text sequential manipulations. Since its inception in 2017, it has achieved the-state-of-the-art results in a variety of NLP tasks.

The outstanding performance of the Transformer is due to its constituents, which are the basis of many recent model variants. Stacked by many Transformer blocks, the models formed a multi-layer structure. Figure 2.10 illustrates a well-established model architecture of a Transformer block proposed by Vaswani *et al* in 2017 [5]. The basic block is composed of an encoder and a decoder, the output of both will be normalized by the SoftMax activation function. The main characteristics include position encoder, multi-head self-attention mechanism, layer normalization, and residual connectors. The input of the Transformer block is a word embedding vector combined with the position embedding vector from the position encoder. The position encoder produces a corresponding vector depending on the position of each word in the text sequence, hence the contextual information can be extracted. A sinusoidal/cosine input or an embedding that can be trained can be used for the position encoder.

The combined embeddings are then fed into the encoder composed of multi-head attention

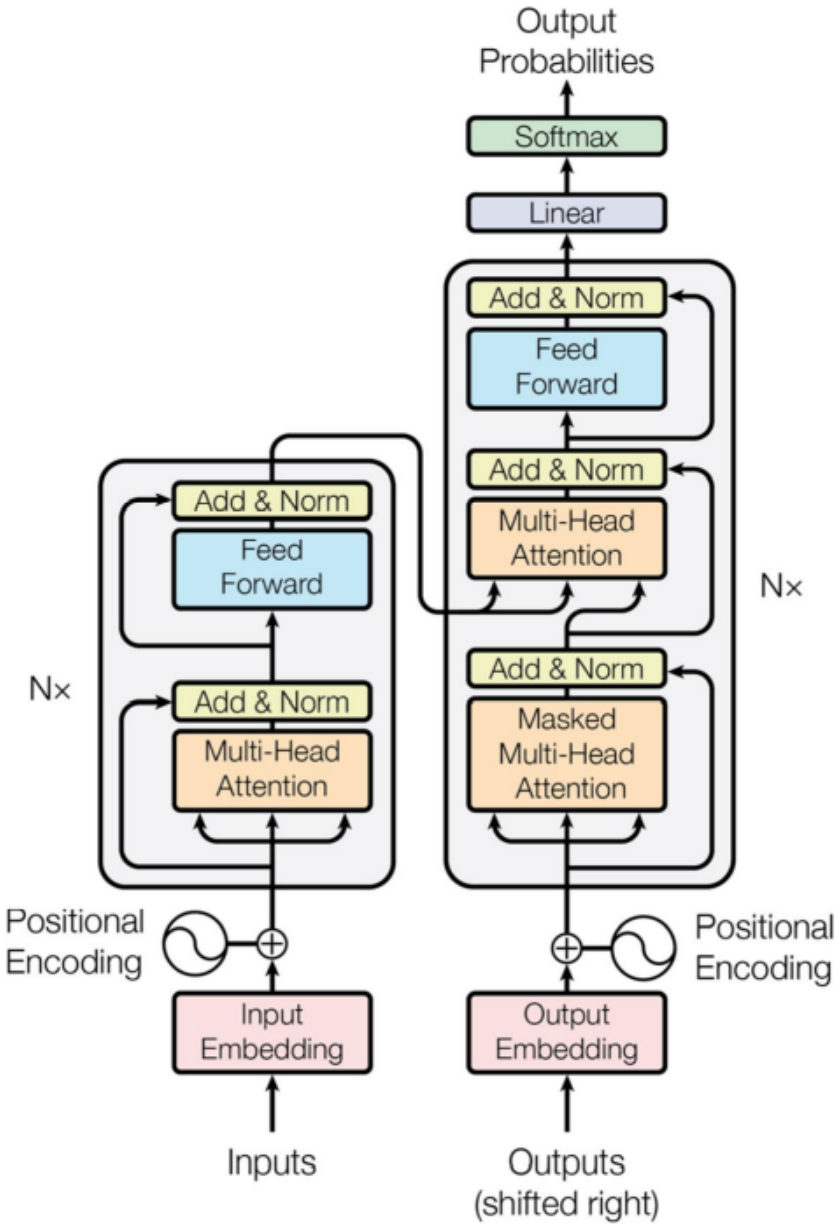


FIGURE 2.10: The architecture of a Transformer block [5]

and a feed-forward network. The multi-head attention mechanism produces the attention vector for every input to reflect how strong the relevance of each word is with other words in the same text sequence. Therefore, further contextual meaning can be captured. The output of the multi-head attention layer is then fed into a two-layer feed-forward network in the form of one vector at a time. Both the output and input of the multi-head attention layer and feed-forward network are connected by a similar structure consisting of residual connectors and a normalization layer. In this case, the output of the feed-forward network

will be the decoder's input. In particular, the attention is independent when dealing with each input, the multi-headed attention mechanism [7] also achieves parallelism.

The decoder has a similar position encoder and masked multi-head attention part as an encoder. The attention vectors from both the encoder and the masked multi-head attention are fed into another multi-head attention layer. Then the result passes through the feed-forward network, linear layer, and SoftMax activation function to obtain the output probability distribution.

The use of the Transformer architecture can vary according to the tasks to which it is applied: For machine translation, both encoder and decoder are employed; For text classification tasks, only the encoder part is used; For language modeling, only decoder is involved. Transformer-based models have been a formidable force in the development of various NLP tasks [82].

2.4.2 BERT

Bidirectional Encoder Representations from Transformers well known as BERT has been regarded as a groundbreaking new language model in the modern deep learning domain. Based on the architecture of the Transformer model, BERT can capture and learn the contextual information between words in the text sequence. The main strength of BERT is that it is essentially a pre-trained model which can achieve state-of-the-art performance on various NLP tasks. Moreover, it just needs to be fine-tuned on the specific datasets and downstream tasks. Thus, there is no need for developing and training a strongly-specific model started from scratch. Employing the encoder structure of the Transformer model as the sub-structure to perform the pre-trained tasks. As shown in Figure 2.11 there are two phases in BERT's work: pre-training and fine-tuning.

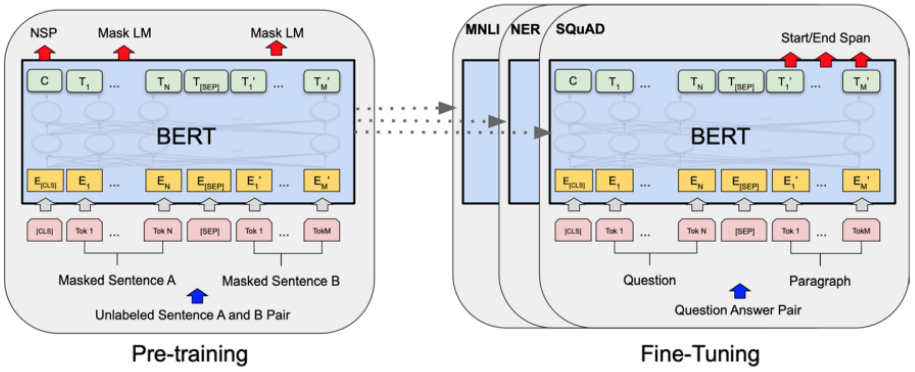


FIGURE 2.11: Two phases of BERT [6]

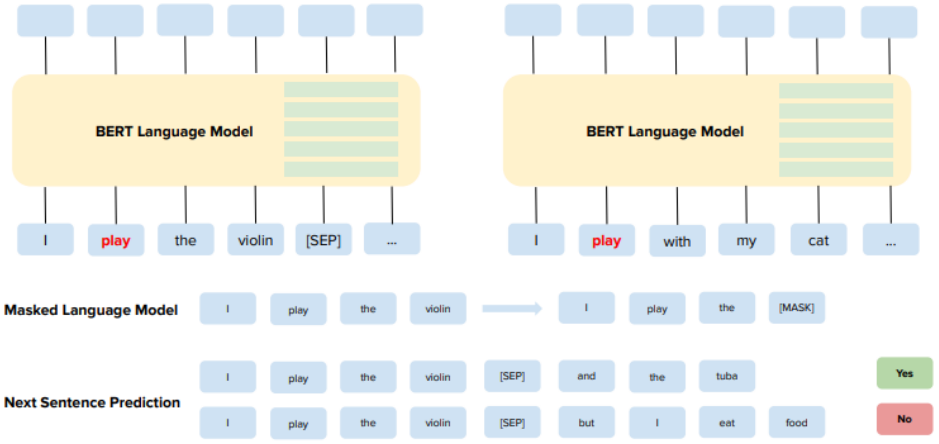


FIGURE 2.12: A schematic view of MLM & NSP [7]

(1) pre-training:

Due to the training on MLM (Masked Language Modeling) and the NSP mechanism (Next Sentence Prediction mechanism), BERT is enabled to learn the meaning of language. For some random sentences that are the inputs, MLM masks some words randomly, and then those masked words will be reconstructed from the contextual words. On the other hand, determining which one comes after the other one when there are two sentences, can be said that NSP is achieved. NSP mechanism enables the model to keep the spatial relationship between texts, especially if it is long-distance. A schematic representation of the work of MLM and NSP is introduced in Figure 2.12 [7]. BERT was pre-trained on the texts of 16 GB size from English Wikipedia and the BooksCorpus datasets.

(2) fine-tuning:

After the pre-training phase, the model is trained by supervised learning on the specific

NLP task and dataset. Then the fully-connected output of BERT is replaced by a set of new layers. Because except for the output's parameters that were trained from scratch that other parameters only need to be fine-tuned, the training speed of BERT is much faster than other models.

Based on the Transformer model architecture, the bi-directional characteristic of BERT enables the model to capture the contextual feature from the text. The essence that it is a pre-trained model makes the BERT-based model train fast and perform effectively. The emergence of BERT has great significance for the development of NLP applications for various tasks.

2.5 Label Embedding

As aforementioned, although there have been many advanced deep learning models that have been developed so rapidly and widely used for many NLP tasks and have proved their effectiveness. It is not hard to find that the learning paradigm of those models has the same components. These include a deep learning model for generating the text representation and a classifier targeting to compute the distribution of predicted labels. Classifier typically is learned from the cross-entropy loss (generally used) between the predicted distribution and the label vector represented by a one-hot vector.

But in text classification tasks, labels are not independent of the text data to be classified. One could even say that the connection between labels and text is very close and crucial. The labels of text data are also words or phrases with semantics, so it can be said that labels are related to the text classification tasks. However, the semantic information of the labels is not fully utilized as the labels are often transformed into one-hot vectors that only are used in the learning of the classifier.

To address the aforementioned problems, label embedding-based methods have been proposed in recent years. For text classification tasks, labels are of vital importance for the success of the final classification result. In recent years several researchers have proposed various methods combining the semantic information of labels to build the text representation. Wang *et al.* [8] proposed an attention model based on the word-label joint embedding method: The words of input text data and labels are embedded in the same latent space. In this label-embedding-based attention model, label information enables the model to locate and focus on the more relevant elements. Therefore, high-quality text representation can be constructed from the compatibility between words and labels. Conclusively, the performance of the classification can be effectively improved.

In our research work, we also utilize this method of directly calculating the compatibility between words and labels in their joint embedding space. Aim to obtain the degree of relevance of each word to the classification tasks, the key idea of the mechanism is to compute the matching/relevance score between words and labels. Thus, this mechanism can incorporate the text-label signal to construct the text representation for the classification task. By calculating text-label compatibility, a matrix will be obtained. Each entry of the matrix is a word-label matching score (cosine similarity calculation is typically used). This label embedding-based attention model proposed by Wang *et al.* [8] will be introduced in detail. Its essential differences from traditional models will also be presented:

(1) Traditional Learning Paradigm:

Given a training set $S = \{(X_n, y_n)\}_{n=1}^{N}$ which compose of pair-wise data. The text sequence is denoted as $X \in \mathcal{X}$ and $y \in \mathcal{Y}$ denotes the labels of the corresponding text data. Labels are always in the form of one-hot vectors when it is single-label (in the multi-label task, it is binary vectors).

The classification model is trained with the essential goal to learn the function $f : \mathcal{X} \mapsto \mathcal{Y}$

to minimize the empirical risk:

$$\min_{f \in \mathcal{F}} \frac{1}{N} \sum_{N}^{n=1} \delta(y_n, f(X_n)) \tag{2.4}$$

When the real label is y and the predicted one label computed is $f(x)$ (f belongs to the functional space $\mathcal{F}$). The loss incurred is measured by $\delta : \mathcal{Y} \times \mathcal{Y} \mapsto \mathbb{R}$. 0/1 loss is always employed in this evaluation step: $\delta(y, z) = 1$, if $y = z$. Cross Entropy Loss is used as the surrogate loss for the model's prediction for this single-label problem we discussed.

The process of a text classification method can be considered to be typically composed of three steps by a function decomposition, end-to-end. It can be written in the form of $f = f_0 \circ f_1 \circ f_2$ which is shown in Figure 2.13:

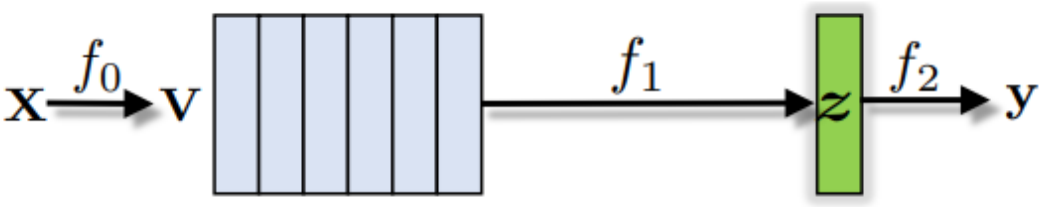


FIGURE 2.13: The pipeline of the traditional classification method [8]

a) $f_0 : \mathbf{X} \mapsto \mathbf{V}$

X is a L-length input sequence composed of word tokens: $X = \{x_1, \cdots, x_L\}$. The framework that is widely employed to map the words into a latent space to obtain the word representation relies on *word embedding* introduced before. V denotes the word embedding matrix of a text sequence whose size is $L \times P$, $V = \{v_1, \cdots, v_L\}$, $v_l \in \mathbb{R}^P$

b) $f_1 : \mathbf{V} \mapsto \mathbf{z}$

A compositional function is denoted by f_1, which has employed sophisticated neural network architectures such as RNN and CNN for state-of-the-art performance. The text representation z is constructed from the word embeddings by f_1, which is a fixed-length vector.

c) $f_2 : \mathbf{z} \mapsto \mathbf{y}$

f_2 is considered as a classifier whose input is test representation z and output is the predicted label.

(2) New Learning Paradigm based on Label Embedding:

Three steps of the training pipeline of the traditional text classification have been introduced above. It is not difficult to note that the first two steps only leverage the information from the input text sequence. The information on the label is only used in the last step f_2. The influence of label information on the word representation (f_0) and generating the text representation (f_1) is ignored or it can be said that it does not play a direct role. To solve this problem, a new method whose every step incorporates label information was proposed. The pipeline is illustrated in Figure 2.14.

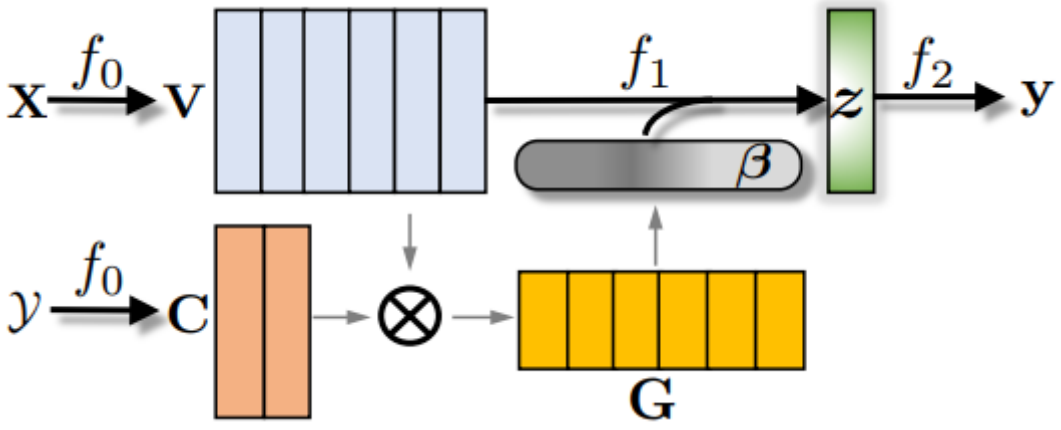


FIGURE 2.14: The pipeline of the labe embedding-based classification method [8]

Different from those aforementioned architectures based on neural networks which are always sophisticated and indirect (these model structures are devoted to constructing text representation z directly from the word embeddings V). The new method focuses on measuring the compatibility G between words and labels. Then β will be derived from G as the attention weights used to obtain z.

A new method based on label embedding proceeds is shown in Figure 2.14. In the function decomposition $f = f_0 \circ f_1 \circ f_2$, word-label joint embedding is the important part. $\bigotimes$ in Figure 2.14 denotes cosine similarity calculation method.

a) $f_0 : \mathbf{X} \mapsto \mathbf{V}, y \mapsto C$
In addition to the embedded words, all the labels are also embedded in the same embedding space, which is called the joint embedding space. The embedding space can be refined under the influence of label embeddings used as an 'anchor'.
b) $f_1 : \mathbf{V} \mapsto \mathbf{z}$
The compositional function is weighted by the attention score β which is derived from the compatibility between labels and words, aggregates word embeddings into z.
c) f_2 :
The learning of f_2 is the same that interacts with labels directly.

The label-word joint embedding-based attention model based on the proposed label embedding framework should be described specifically.

Joint Embeddings of Words and Labels

As introduced, the word embeddings and label embeddings are produced through the same latent space ($\triangle^D \mapsto \mathbb{R}^P$ and $\mathcal{Y} \mapsto \mathbb{R}^P$). $C = [c_1, \cdots, c_K]$ are label embeddings, where K if the number of the classes.

Calculating the cosine similarity can be a simple way to leverage the word-label pairs' compatibility

$$G = (C^T V) \oslash \widehat{G} \tag{2.5}$$

$\widehat{G}$ is the normalization matrix, of size $K \times L$. Each element of $\widehat{G}$ is calculated by the multiplication of l_2 norms of the $l-th$ word embedding and $c-th$ label embedding:

$$\widehat{g_{kl}} = \left\|c_k\right\| \left\|v_l\right\| \tag{2.6}$$

In the calculation of compatibility, to capture the further spatial information of those consecutive words such as phrases and simultaneously introduce non-linearity. A generalization form shown as Eq 2.7 is employed. For a phrase which centered at $l-th$ word and of $(2r+1)$ length, it corresponds a local matrix block of G denoted as $G_{l-r:l+r}$. $G_{l-r:l+r}$ measures the word-label compatibility to obtain the matching score of the phrase-label pairs. The higher-level compatibility of the labels with the $l-th$ phrase is :

$$u_l = ReLU(G_{l-r:l+r}W_1 + b_1) \tag{2.7}$$

$W_1 \in \mathbb{R}^{2r+1}$, $b_1 \in \mathbb{R}^K$ are parameters that needed to be trained. Then the largest value of the compatibility of the $l-th$ phrase will be collected:

$$m_l = maxpooling(u_l) \tag{2.8}$$

The final attention score/weight is always obtained via the SoftMax function:

$$\beta = SoftMax(m) \tag{2.9}$$

Finally, the text representation of the entire text sequence can be calculated as the weighted averaging sum of word embeddings:

$$z = \sum_l \beta_l v_l \tag{2.10}$$

2.6 Conclusion

The substantive literature review shows that there is plenty of research done on text representation, especially various deep learning models such as RNNs, and CNNs. Furthermore, the attention mechanism introduced more recently has become popular, because of its ability to make those aforementioned deep learning models more flexible for text representation. There also has been abundant research literature on the development of word embedding methods for text representation. Most methods ignore the particularity of text data (these methods often require only a little modification to be applied to the field of computer vision or others). Moreover, a large amount of annotated data is crucial to the satisfactory performance of deep learning models. On the other hand, if there is strong noise in the text data that needs to be classified, the classification result will be severely diminished.

Therefore, in this thesis, we will try to exploit the semantic information of the labels:

(1) The key idea is constructing the text representation by directly using the compatibility between text and label. We build up a framework without multiple complicate steps which are needed in traditional models to solve intermediate problems.
(2) Based on the idea of computing word-label compatibility in the word-label joint embedding space, the attention mechanism in our model only involves a few basic algebraic operations and few parameters.
(3) Differ from the traditional methods that only intercept a fixed-length text sequence from a document after tokenization, as the input into the word embedding space to get the mapped vector. We have changed the way the document data is preprocessed which remain the structure of each sentence.
(4) Thus, the processing in (3) makes it possible that the input to our model will be in the form of sentences rather than sequences of text. We built a two-level attention network that selects the sentences that are most likely to contain important information. We then use only the representations of those selected sentences as input to the classifier.

Therefore, the noise in the text data can be largely reduced, and the accuracy of the classification can be improved. Furthermore, we can locate the location of those "key sentences"in each document. Moreover, the learning of the model can achieve an ideal performance even without relying on large datasets.

Chapter 3

Label Embedding-based Hierarchical Attention Model

3.1 Introduction

Previous chapters cover deep learning research that has already been carried out in the area of text representation. Meanwhile, the existing problems and challenges are also introduced. For example, the challenge of performing classification tasks on noisy datasets. In this chapter, the main part of the research and engineering work so far are described, including some modifications in the label embedding-based method and work details of the whole attention framework. Section 3.2 introduces some special work on data processing because we need to retain the sentence structure of data, for this reason, our model processes the data from the sentence level to the word level. However, most existing classification models focus only on the information contained in the words themselves and the relationship between word tokens in a text sequence.

As the most important part of this thesis, all the implementation details, and novel applicable methods as well as their theoretical basis are shown in Section 3.3. The core part is a Label Embedding-based Hierarchical Attention model we proposed.

3.2 Data Preprocessing

3.2.1 Traditional Preprocessing

Generally, there are four steps in the pre-processing of input text data: Tokenization; Stopwords Removing (redundant words); Stemming; Word Embedding:

(1) ***Tokenization***: Removes special characters, and white spaces to only remain word tokens.
(2) ***Stopwords Removing***: "Stopwords"refers to those words that have only a functional role in the text document. They contain very little semantic information and serve only grammatical meaning normally used to construct sentence structure. On the other hand, these words are common and they occupy a non-negligible proportion of the text. Removing these stop words, therefore, helps to reduce computational and time costs while retaining the important semantic feature of the text data. A set of widely used stop words summarised by experts includes "the", "that", "of", "on"etc. Thereafter, the word tokens set from the Tokenization phase is checked and each word occurring in the stopword list will be removed.
(3) ***Stemming***: It is well known that each word exists in various forms of variants in different locales, such as past tense (mostly with -ed), continuous tense (mostly with -ing), etc. However, the semantic meaning of the word itself remains the same. Hence, Stemming aims to remove those prefixes/suffixes of those word variants to maintain the term stems. Then the number of keywords in the dictionary can be decreased by Stemming. Therefore, different variants of a word can share the same word representation that is derived from a single keyword, providing the enhancement of the performance of classification tasks.
(4) ***Word Embedding***: As introduced in Section 2.3, word tokens are passed through an embedding vector space and then transferred into word embedding vectors. Word embeddings represent the semantic and syntactic features. The success of a text classification task is strongly related to the quality of word embeddings.

The pipeline of a traditional text pre-processing is shown in Figure 3.1.

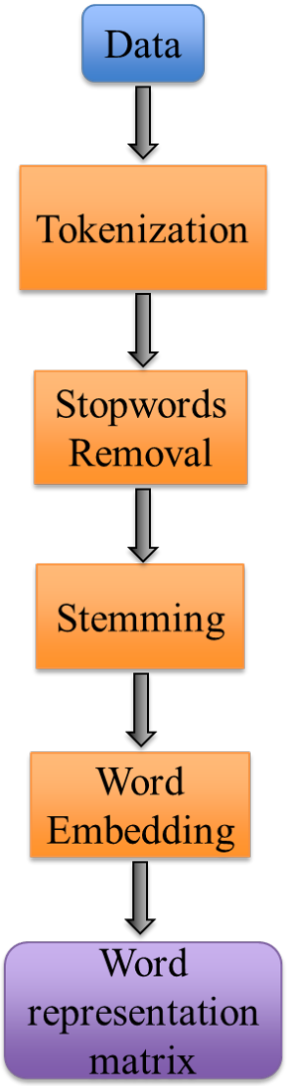


FIGURE 3.1: Traditional text pre-processing

3.2.2 Text Preprocessing: Remain sentence structure of raw data

As Chapter 1 especially Section 1.2 introduced in the previous part in this thesis, the object of the sentence-level attention framework is to choose the sentences which are

most likely to contain keywords, to contribute to the final text representation for the classification. The selection of sentences is based on their "attention score", which are also calculated in the sentence-level attention layer. Moreover, the semantic meaning of words and sentences is highly dependent. Therefore, on the basis of the label-embedding mechanism, the "score"of a sentence is derived from the compatibility between the labels and each word in this sentence.

According to the introduction in Section 3.2.1. In the traditional NLP tasks, the method of text preprocessing to obtain an input sequence to word embedding is always: Remove all the punctuation of the text document first so that only words remained, and each word is denoted as a token; Then each token sequence will be truncated to the same fixed length as the input of the deep learning NLP model. This is because the input should be a tensor with the same length in the same dimension.

As shown in Figure 3.2, in our framework, the first step of the process is separating each text document into several sentences before *Tokenization* and *Stopwords removal*. As explained before, the sentence structure of each document needs to be remained. Then, considering how many sentences are in each text and the number of words in each sentence are all different, we use the method of padding at both the sentence level and word level. As a result, each text data has N sentences and each sentence consists of L word tokens, to ensure the preprocessed text data can be transformed into tensor as the input of the word embedding space before being fed into the attention model.

In practice, data is fed into the model in form of batches which we called tensors, and elements of tensors have the same size in each dimension. The number of sentences in each text and the number of words in each sentence are not always the same. Hence, the padding method is employed to transfer the input data into a tensor. The other important part we should consider is that only the original parts of the text should influence the compatibility between sentences/words and labels. In the padding step after the sentence separation, the padding sentences and padding words should have no interference. Therefore, the compatibility between the padding parts and the labels will be 0. Thus, they will

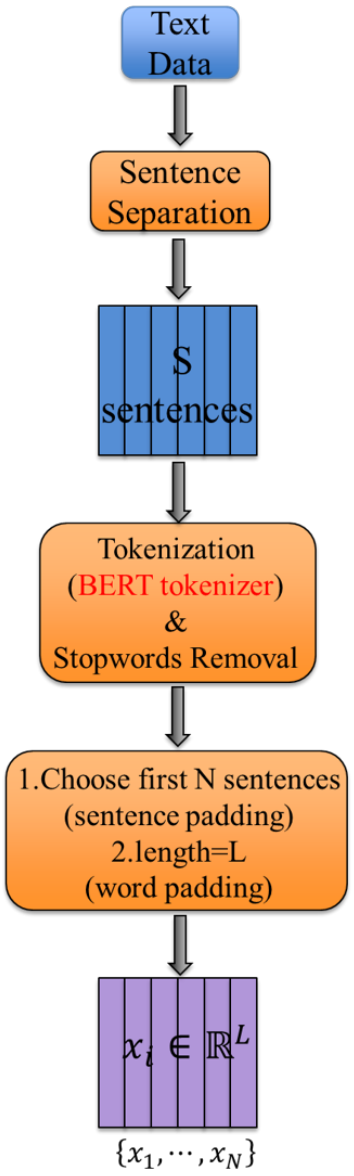


FIGURE 3.2: A new text preprocessing procedure

have no influence on sentence selection and the learning of weights in the attention model.

Moreover, the stopwords which are always not relevant to the classification results are removed in the text preprocessing. Therefore the noise and the cost of computation can be reduced. The index of the document is used to find out the key sentences and keywords that the attention model selects after learning. We can evaluate the performance of our label-embedding hierarchical attention model more comprehensively by analyzing whether these selected sentences do contain keywords.

3.2.3 Extra Information

Regarding the method of combining label information, we creatively introduce additional information derived from label derivatives. Derivative words together with the labels themselves used as label information on the basis of the idea of label embedding mentioned in Section 2.5. Richer label information helps to more effectively assess the relevance of the words in the text document to the classification task.

To obtain a label list consisting of labels and label-derived words, we pre-collect the derived words associated with the labels of the corresponding dataset through methods such as Google searches and references to Wikipedia.

3.3 Hierarchical Attention Framework

In this section, the details of the proposed method will be comprehensively explained. The whole architecture is shown in Fig 3.3.

3.3.1 Introduction

Chapter 1 has briefly introduced the datasets for which our proposed model is targeted and the main working mechanism of the model. Based on the fact that many text data are so noisy that means that there are too many redundant parts. Sentences and words containing important information are in a small proportion of the entire text document. Most NLP models directly deal with the whole text sequence to locate and capture semantic and syntactic features relevant to classification tasks. Therefore, it is a challenge for them to perform satisfactorily on those text data with "strong noise, weak signal"characteristics.

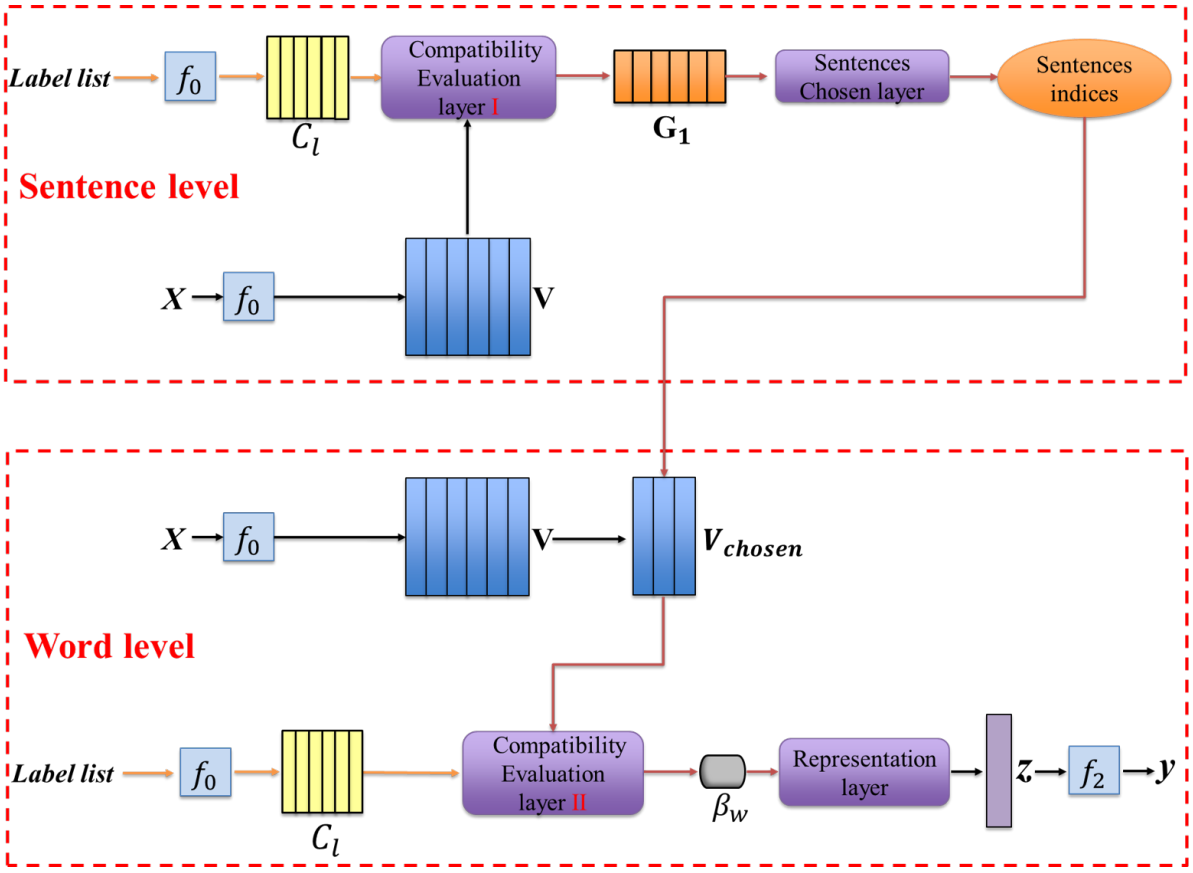


FIGURE 3.3: The whole architecture of Label Embedding-based Hierarchical Attention Networks

To address the above problems, we provide a new idea of text classification inspired by human intuitive thinking: When encountering those texts with strong noise, how do we identify the subject of this document? The method adopted by humans is to browse the whole text and first find sentences that are relevant to the topic of the document. Then simply by reading these sentences in detail, we can understand the subject matter. On the other hand, there is no need to read the rest of the text word by word. Therefore, a hierarchical attention model from sentence level to word level is designed, aiming to perform as the way of human thinking. The attention layer acts on sentence structure to select the relevant sentences. Then only these chosen sentences containing important information for the classification tasks, are fed into the next attention layer on the word level. In this word level of model, the final text representation is constructed, which is then used as input to the classifier.

Figure 3.3 illustrates the overall architecture of the proposed framework. We combine

the label information to calculate the degree of relevance of sentences/words to the classification task. Then we use the calculation result for sentence selection at the sentence structure level, as well as the acquisition of text representation at the word structure level. As mentioned in Section 2.5, we make some modifications based on the jointly word-label embedding method proposed by Wang *et al.* [8]. The same word can be of different importance in different sentences. Therefore the joint embedding space is different in these two label embedding-based attention levels. At the same level, word embedding and label embedding are in the same latent space.

The two-level framework will be introduced separately next.

3.3.2 Sentence-level attention: achieve sentence selection

The overall attention framework at the sentence level is shown in Figure 3.4. Two main parts that deserve our attention are the calculation of the sentence compatibility score G_1, and how sentences are selected from the entire text document according to the calculated score. We will explain them in detail respectively in this section. (which sentences to choose and how many sentences to choose, reflected by the obtained sentence indices list).

The input text data X is a matrix of size $N \times L$ since a text consists of N L-length sentences. V is the output matrix of the word embedding space of which the size is $N \times L \times P$. P is the dimension of the embedding space. Label list includes the labels of this text classification task and the derivatives of these labels considered as extra information explained in Section 3.2.3. The number of words in label list is l, then l pass through the same embedded space as text data X. The label embedding matrix obtained C_l is of size $l \times P$.

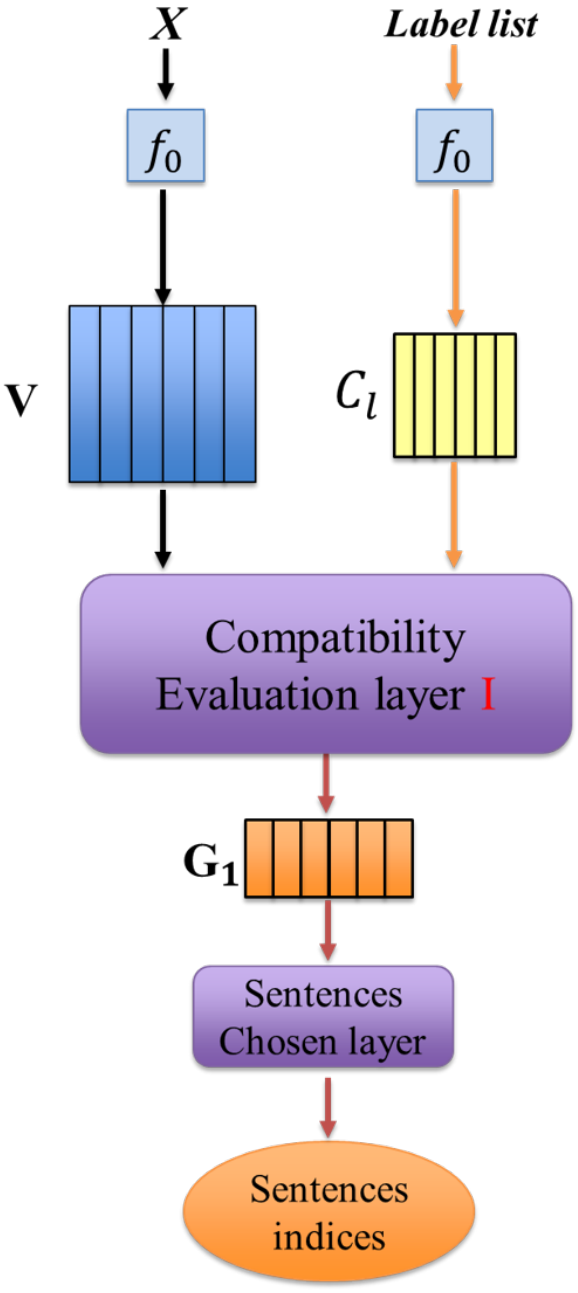


FIGURE 3.4: The architecture of sentence-level attention network

3.3.2.1 Compatibility Evaluation in Sentence-level Attention

This subsection will explain the detailed calculation process about how to combine the label information with extra information to compute the reliable and interpretable scores. The scores should be a reflection the relevance of sentences to the classification.

The information of the words and sentences is highly dependent since sentences consist of words. Therefore, to obtain the "relevant score" we depend on deciding which sentences to choose. The compatibility of each word needs to be calculated first. Then the word compatibility values are aggregated into a "score" by some methods, which can represent the relevance and importance of this sentence reliably.

According to the data preprocessing represented in Section 3.2.2, each input matrix consists of N token sequences denoted as N sentences of length L in a text document. The compatibility of label-token pairs in each sentence will be measured respectively. Figure 3.5 describes the detailed process of how the compatibility/attention score from the word embedding of one sentence is calculated. The way we apply via the cosine similarity is inspired by the method introduced in Section 2.5. The major difference between the method we proposed is that we make some modifications to the calculation of the compatibility of each word in the sentence. The reason is that the LEAM model designed by Wang *et al.*[8] to construct a text representation only needs to calculate the relevance score of each word in the text document. And then calculate the weighted sum of the word embeddings of text with this score. Therefore, as Eq 3.9, the computed cosine similarity by Eq 3.7 and Eq 3.8 is further operated for the consideration of combining word contextual information and the existence of phrases. Meanwhile, nonlinearity is also introduced. LEAM model obtains the final value of the attention weight of the word after max-pooling operation (Eq 3.4) and SoftMax activation function (Eq 3.5).

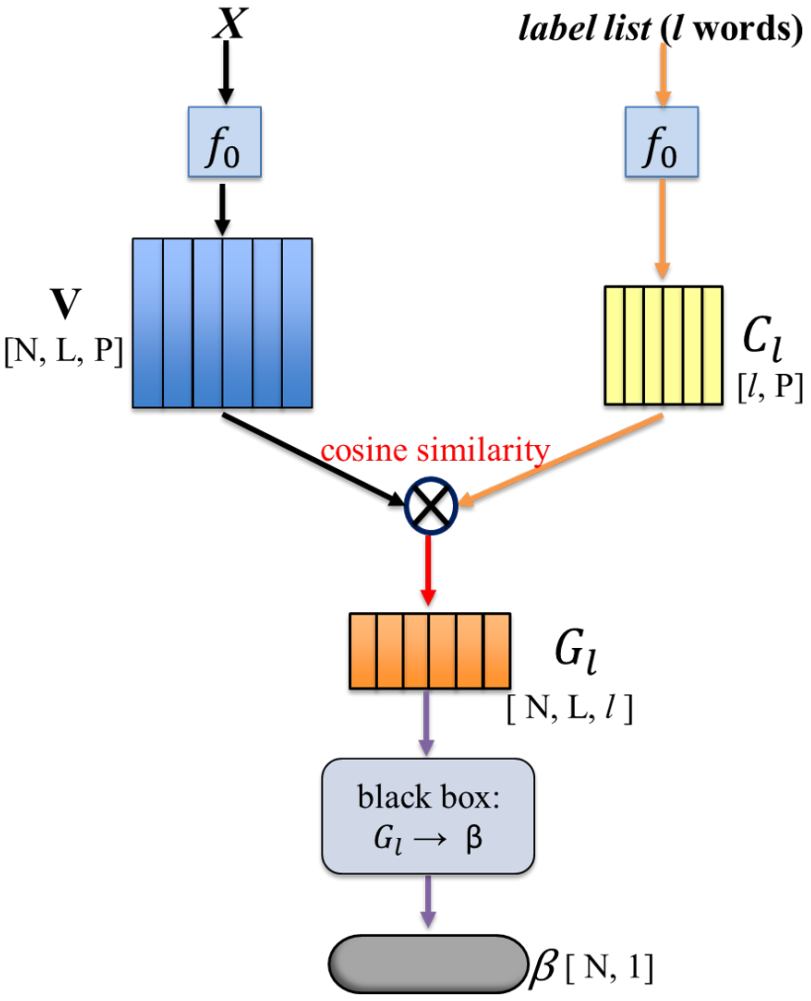


FIGURE 3.5: Compatibility computation in Sentence-level

$$G = (C^T V) \oslash \widehat{G} \tag{3.1}$$

$$\widehat{g_{kl}} = \left\|c_k\right\| \left\|v_l\right\| \tag{3.2}$$

$$u_l = ReLU(G_{l-r:l+r} W_1 + b_1) \tag{3.3}$$

$$m_l = maxpooling(u_l) \tag{3.4}$$

$$\beta = SoftMax(m) \tag{3.5}$$

However, in our model, the label embedding-based attention score for each word in this sentence is leveraged to obtain a score for this entire sentence, rather than obtaining the

final text representation of the sentence. By imitating human thinking, we can know that when a word related to a text topic appears in a sentence, no matter where it appears, this sentence is most likely to be relevant to text classification. Hence, in this compatibility layer, the cosine similarity G_1 is simply calculated by Eq 3.7 and Eq 3.8 without considering of context. It is worth emphasizing that, unlike the LEAM model that only combines label information, we also introduce the extra information from label derivatives. So C represents a list with l words (the number of labels is K) the Eq 3.7 is rewritten as:

$$G_l = (C_l{}^T V) \oslash \widehat{G_l} \tag{3.6}$$

The size of G_l is $N \times L \times l$ with the meaning that there are N sentences in a text document. Each sentence consists of N word tokens and the cosine similarity between each token. Every word in the l-length label list will be computed.

It can be seen from Section 3.2.3 that all the collected words are included in the label list C_l, which is used for the calculation of the similarity value of each word in the text. However, it is clear that not every word in the label list C_l is related to a certain text. If this text belongs to a category, only the cosine matching scores, calculated from word tokens in text sequence with those label derivatives belonging to this category, is considered to be used to build the relevance value of the sentence that reflects its importance reliably. On the contrary, if all the words in the C_l are used, the redundant extra information will interfere with the calculation of sentence relevance weights. As a result, the subsequent sentence selection mechanism that relies on the weight value may make wrong decisions due to redundancy. Therefore, for each word token of the sequence, we will sort the obtained vector composed of cosine similarity values between the tokens and all the words in the label list. Then only the top m_1 values are selected and summed to a single value, which is considered as the relevance of the words in this sentence for the classification task. Figure 3.6 shows the detailed process about how to aggregate the cosine similarity Value $\boldsymbol{G}$ to the weight value β in the "black box"in the Figure 3.5.
Following a similar idea, not every word in a sentence contribute to the relevance of the sentence. Especially the datasets targeted by our proposed model have the characteristic

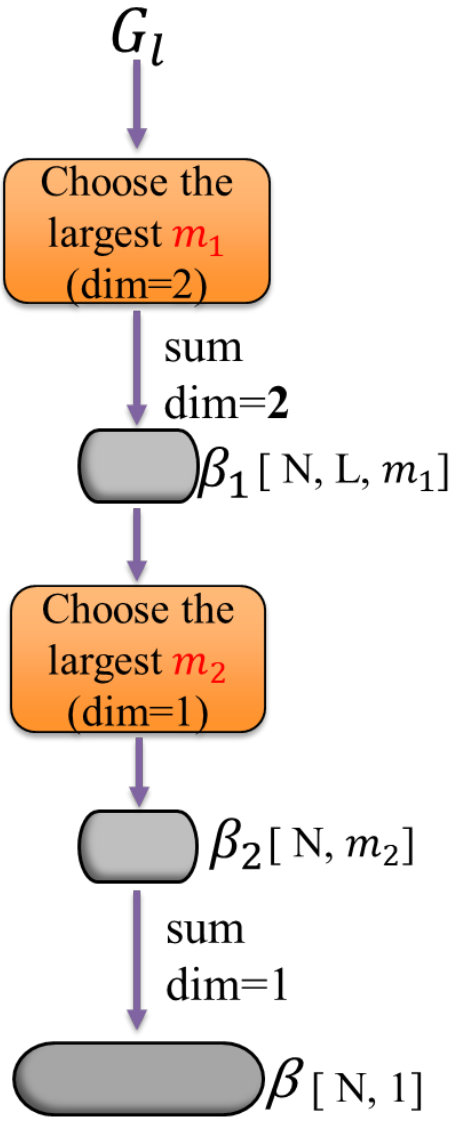


FIGURE 3.6: The detailed computational work execution in Sentence-level

of large noise and few keywords. Thus, the matching score values corresponding to all words in a sentence are also sorted first. The top m_2 largest values are then selected (which means the top m_2 most relevant words are selected). The sum of these values represents the compatibility of the sentence to the class label.

Following the explained calculation process as shown in Figure 3.5, the relevance score of each sentence in the text document is collected, as the basis of the next step sentence selection.

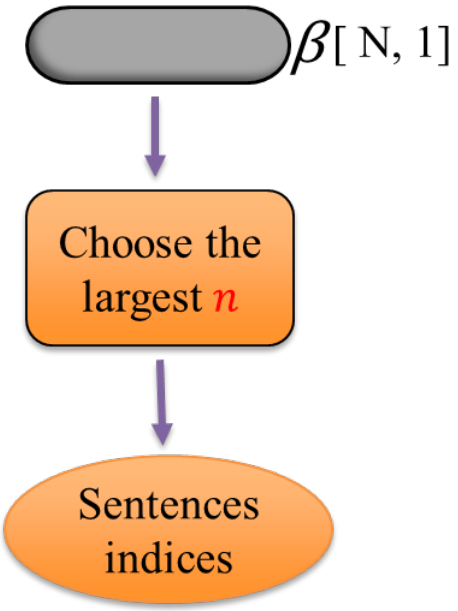


FIGURE 3.7: Select a fixed number of sentences

3.3.2.2 Sentence Selection in Sentence-level Attention

Section 3.3.2.1 has exhaustively introduced the label embedding-based approach, combined with the extra information from the label derivatives words. The sentence compatibility scores should reliably represent the degree of relevance to the classification task. Therefore, those sentences with higher relevance scores need to be chosen for the subsequent process.

In machine learning, data is passed in batches for the model training and validation. Moreover, it should be pointed out that, in practice, data will be represented as tensors. This means the elements in the same batch are required to have the same size in each dimension. Therefore, as Figure 3.7 shows, after obtaining the corresponding compatibility scores β of sentences in the text document from the compatibility computing layer, there is a simple way to directly select the n sentences corresponding to the top n largest compatibility scores. The corresponding indices of the chosen sentences will be obtained and those sentences selected used for the work at the word level.

Since the compatibility score of a sentence is calculated from the compatibility of the

words that make it up, it reliably reflects the relevance of the sentence to the classification task. The higher the score, the stronger the relevance. Therefore, it is reasonable to select the n sentences with the highest n corresponding compatibility score values from each text document.

3.3.3 Word-level attention: obtain text representation

The attention network on the word level is shown in Figure 3.8. The text representation is constructed in this level **from only the sentences selected by the sentence selection mechanism of the sentence-level attention**. The compatibility scores of the words to the classification will be calculated in the compatibility layer and the text representation is the output of the representation layer.

3.3.3.1 Compatibility Evaluation in Word-level Attention

As in Section 3.3.2.1, the calculation of the word matching score still needs to combine the label information provided by the labels and their derivative words. C_l denotes the label list, and the computation of the cosine similarity is the same as Equation 3.6 and Equation 3.8. Furthermore, the importance of the contextual information and the existence of the phrase *et al.* can not be ignored at the word level where the text representation is produced. Therefore, we add context information in addition to the cosine similarity calculation between the word embeddings and the embeddings of the label list, following the calculation method introduced in Section 2.5. Figure 3.9 explains the details of the calculation of the compatibility scores of words in the j-th selected sentence of a text document.

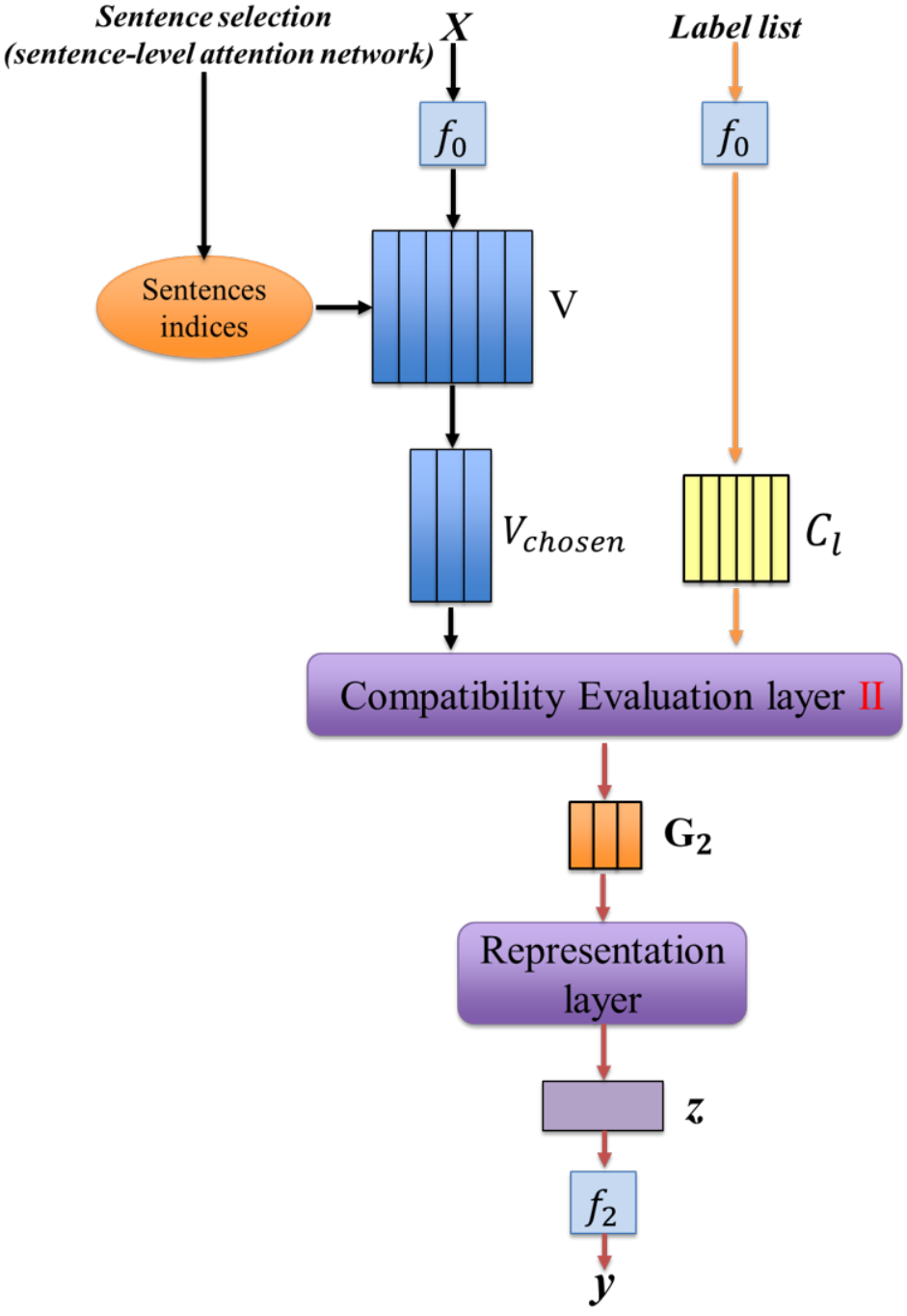


FIGURE 3.8: The architecture of word-level attention network

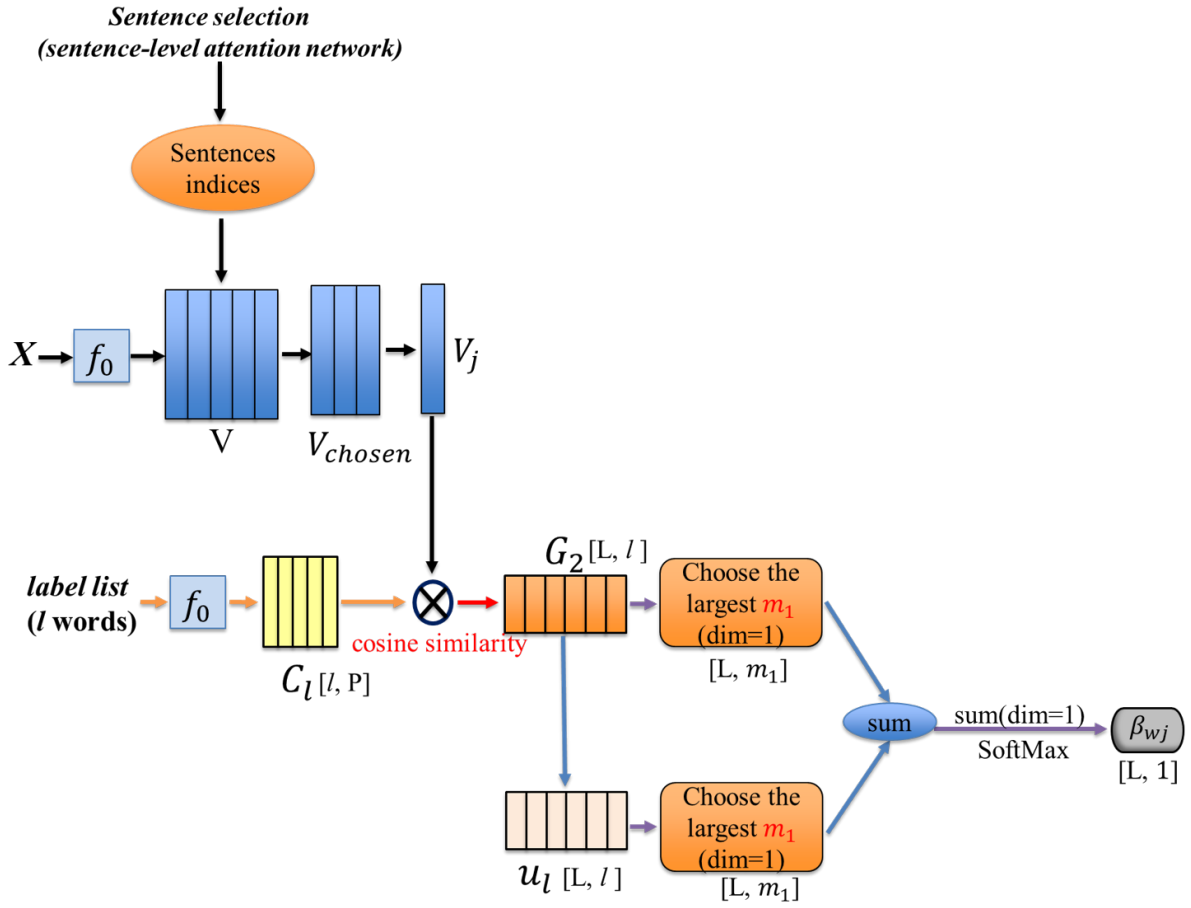


FIGURE 3.9: Compatibility computation in word-level

The computation of the cosine similarity is the same as what we introduced in Section 2.5 :

$$G_l = (C_l^T V) \oslash \widehat{G_l} \tag{3.7}$$

$$\widehat{g_{kl}} = \left\| c_k \right\| \left\| v_l \right\| \tag{3.8}$$

Considering the distance reliability between words (including context and phase, etc.), the method is the same as introduced in Section 2.5 (the non-linearity is also introduced):

$$u_l = ReLU(G_{l-r:l+r} W_1 + b_1) \tag{3.9}$$

Similar to what is explained in Section 3.3.2.1, not all words in the label list C_l are related to the label of a certain text. Therefore, the redundant interference needs to be reduced to obtain a higher-quality text representation. For each word, the computed cosine similarity values corresponding to the label list G_2 as well as the matching scores

containing contextual information u_l will be sorted first. Only the top m_2 largest values of sorted G_2 and sorted u_l will be remained, then they are added into one the followed by SoftMax activation function. β_W denotes the relevance score of the word for the classification task.

3.3.3.2 Representation result of Word-level Attention

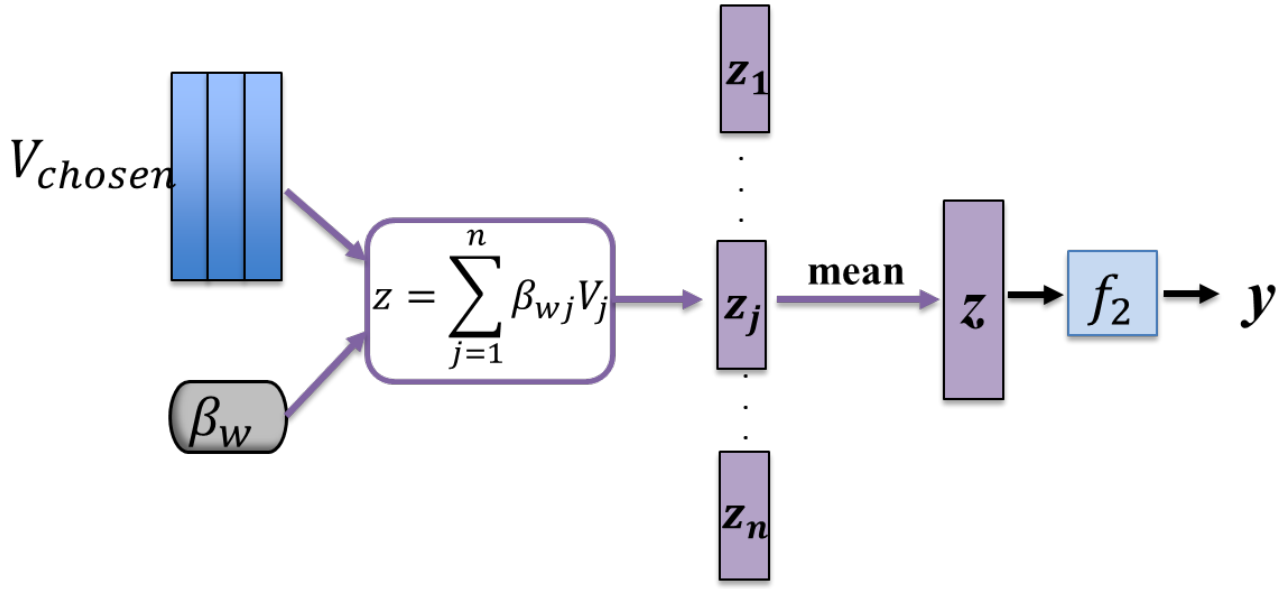


FIGURE 3.10: Representation computation in word-level

It needs to be emphasized again that only sentences selected from the sentence-level attention network can be used for the calculation of the text representation. How to obtain the text representation z is shown in Figure 3.10.

The number of the selected sentences is n. The compatibility/relevance scores of word tokens in each selected sentence are collected from the compatibility calculation layer as Section 3.3.3.1. The weighted sum of the word embeddings of each sentence V_j with the corresponding computed attention weights β_{wj} is obtained as the representation of the j-th selected sentence. Then the mean vectors of the representation vectors of all the selected sentences are considered as the text representation.

Finally, the text representation is fed into the classifier to produce the prediction of the label.

3.4 Conclusion

In this chapter, the main framework of our proposed Label Embedding-based Hierarchical Attention model as well as the improved data preprocessing method is introduced exhaustively. The relevance between text and label is calculated and used in the sentence-level attention mechanism and word-level attention mechanism. Through a few basic algebraic operations, the score which reflects the relevance of each sentence and each word is obtained, selecting only those high-score sentences and words to construct the text representation can largely reduce noise. The key sentences and keywords also can be located according to the relevance scores.

As introduced in Section 3.3.2.2 and shown in Figure 3.7, the number of sentences selected from each text is fixed because the data need to exist in tensor form. Apparently, selecting fixed-number key sentences isn't consistent with the fact that how many key sentences are in each text is uncertain. Hence, we need to make some improvements to let the sentence selection be adaptive.

Chapter 4

Further Optimisation: Adaptive Selection Mechanisms at the Sentence Level

4.1 Introduction

A novel idea of a hierarchical attention framework based on label embedding is described in detail in Chapter 3. The text input remains the sentence structure consisting of several word sequences rather than being represented as a whole word token sequence. There are two attention layers at the sentence level and word level respectively: At the sentence level, the relevance of sentences is evaluated and then those sentences of highest importance are selected from each text as the input to the word level network; At the word level, the compatibility scores of words to the classification task are calculated and the final text representation is constructed.

In Chapter 3, the number of sentences selected from each text is fixed due to the case where the data exists in tensor form. However, this does not correspond to a variable number of key sentences in each text. This method of selecting a fixed number of sentences from each text as key sentences is not in line with reality as well as human thinking.

Therefore it will have an impact on the quality of the information captured from the data. The accuracy of text classification will be reduced.

4.2 Limitations of fixed number of sentence selection mechanisms

As briefly explained in the previous section, the aim of the attention-based model at the sentence level in our hierarchical model in Section 3.3.2 was to extract key sentences. These key sentences are used in subsequent operations, which ultimately result in a textual representation. The number of key sentences varies from text to text so a fixed-number sentence selection would reduce the quality of the text representation.

To be more specific, if a fixed number of key sentences are selected from each document data, the value of n that should be set to cannot be determined in any case. It will be inevitable that some key sentences are not selected in the texts with a larger number of key sentences than n. Or in the text with a smaller number of key sentences than n, Sentences that are irrelevant to the classification will also be selected. The former case may result in insufficient information for the text representation, while the latter case introduces noise that may cause interference.

For our proposed hierarchical attention model with a fixed number of sentences selected, there are two examples here. Figures 4.1 and 4.2 show two examples of the model making incorrect predictions about the data labels when performing a classification task on the Amazon product dataset.

As shown in Figure 4.1, the number of key sentences selected by the model is set to 1, i.e. n=1. It is a user comment about a kind of protein bar. The correct category

true label	predicted label	selected sentences	text
health personal care	baby product	oh, what's that you say, your kids think they're candy bars.;	they taste great.i'll give them that.oh, what's that you say, your kids think they're candy bars.well, that's because they are, look at the ingredients.the first ingredient is high fructose corn syrup.as far as taste is concerned, these bars are tops.but the protein count is low compared to other supplement on the market and the empty calories are through the roof. let's face it, it's a candy bar

FIGURE 4.1: An example of fixed sentence selection: n=1

label would be "health personal care", with the key sentences underlined in red and the incorrectly selected sentences underlined in yellow. However, on the one hand, the sentence "*oh, what's that you say, your kids think they're candy bars.*"contains the keyword "kids"which is closely related to baby product. On the other hand, the model is set to select only one sentence from the document as the key sentence. As a result, the sentence "*oh, what's that you say, your kids think they're candy bars.*"was selected incorrectly and the real key sentence "*but the protein count is low compared to other supplement on the market and the empty calories are through the roof.*"was not selected. As can be seen from Section 3.3.3, only selected sentences involved in the construction of the text representation. Conclusively, this text was given the wrong label "baby product"rather than "health personal care".

As shown in Figure 4.2, the number of key sentences selected by the model is set to 2, i.e. n=2. It is a user comment about a kind of face cream. The correct category label would be "beauty", with the key sentences underlined in red and the incorrectly

true label	predicted label	selected sentences	text
beauty	health personal care	i purchased this item based on seeing a magazine i read at a doctor's office.; i feel this facial cream has a clean and fresh smell.;	i purchased this item based on seeing a magazine i read at a doctor's office.this purchase was made without trying a sample at a retailer.i feel the product becomes subtle after a while and stays at that degree of smell for 3-4 hours. i feel this facial cream is a clean and fresh smell.hope you enjoy it, if you decide to make purchase.the merchant from whom i ordered from was right on the money.

FIGURE 4.2: An example of fixed sentence selection: n=2

selected sentences underlined in yellow. "*I purchased this item based on seeing a magazine I read at a doctor's office.*"contains the keyword "doctor", which is a health care keyword. Therefore, based on the calculation method we detailed in the previous section, this sentence will also receive a high relevance score. It will therefore be selected together with"*I feel this facial cream has a clean and fresh smell.*", as the model is set to select exactly two sentences as key sentences in each document. However, the "*I purchased this item based on seeing a magazine I read at a doctor's office.*"will not help to get that text correctly classified. Instead, it will interfere with the classification results of the model. Conclusively, this text was given the wrong label "health personal care"rather than "beauty".

Try to imitate the human way of thinking: if we read a user review like the one in Figure 4.1. When looking only at the phrase "*oh, what's that you say, your kids think they're candy bars.*", humans also tend to categorize it as user feedback on a baby product. But when combined with the "*but the protein count is low compared to another supplement on the market and the empty calories are through the roof*", we know that

it is a product review about healthy personal care. This way of thinking is also known as contextualizing information. So, in order to give our model the ability to make such human-like judgments, we need to have the model that can select both sentences. Then, in the next operation, we will combine the contextual information to make a deeper judgment. Eventually, the correct label prediction is obtained.

Therefore, we tentatively propose an adaptive sentence selection mechanism. We aim to enable our proposed model to perform the sentence selection following how many key sentences each text document contains.

4.3 Preliminaries: The Insight Of Compatibility

As mentioned above, the number of key sentences in each text is variant. Therefore, the sentence selection mechanism needs to try to achieve such a goal: How many key sentences are related to classification in a text document, how many sentences are selected from this text, which is called adaptive selection.

Try to follow the human way of thinking: If a certain word that is truly strongly relevant to one of the labels to the text classification appears in a sentence. For instance, if a sentence contains some words belonging to the label list we collected (Section 3.2.3), then this sentence is highly likely to contain information related to classification. We need to pay more attention to those sentences.

As Section 2.3 has introduced briefly, there have been various advanced word embedding methods employed in NLP models. Especially pre-trained initialization methods for embeddings such as word2vec and glove which are dominated widely applied. But in the thesis, we just use them randomly initialized word embedding method for our model. One of the reasons is: We focus on highly noisy datasets where the features of the connection

TABLE 4.1: An example of the computed cosine similarity values. (the example word from label list is **'awesome'**)

word	Riba	foods	based	in	Houston
g	0.0977	-0.0383	0.02096	0.03952	0.01044
word	,	Texas	Makes	this	wesome
g		0.01603	0.44097	-0.01365	1.0000
word	black	bean	dip	,	it
g	0.002013	0.02559	0.06755		-0.019937
word	is	pretty	natural	.	And
g	-0.023173	0.090085	0.013334		-0.011257
word	without	the	lard	that	regular
g	-0.01857	0.01895	0.03002	-0.03502	0.072532
word	refried	bean	dip	has	in
g	-0.3888	0.02559	0.06755	-0.06596	0.03952
word	it	I	am	not	sure
g	-0.019937	0.07200	-0.02041	-0.08192	0.062106
word	why	in	TX	it	can
g	0.042870	0.03952	-0.10777	-0.019937	0.03382
word	not	be	found	.	I
g	-0.08192	0.07248	-0.2289		0.07200
word	make	the	traditional	dip	with
g	-0.01531	0.01895	0.08623	0.06755	0.03669
word	this				
g	-0.01365				

between words are hard to extract. In the case where the dataset is not large, hence, the pre-trained word embedding models are not applicable.

On the other hand, there is the other vital and also novel reason: As shown in Table 4.1, when we look at the cosine similarity values computed between words in a text sequence (here is a text from the Amazon sentiment analysis dataset) and the word in label list (here is "awesome"). We notice that there are and only those words that appear both in the sentence and the label list will produce a large value (close to 1), for the reason that it is the same word and the word embedding vector is the same word. Conversely, different words will result in a very small cosine similarity value. Hence, following this characteristic brought by random-initialized embedding, we can locate the keywords and key sentences where the cosine similarity value is large.

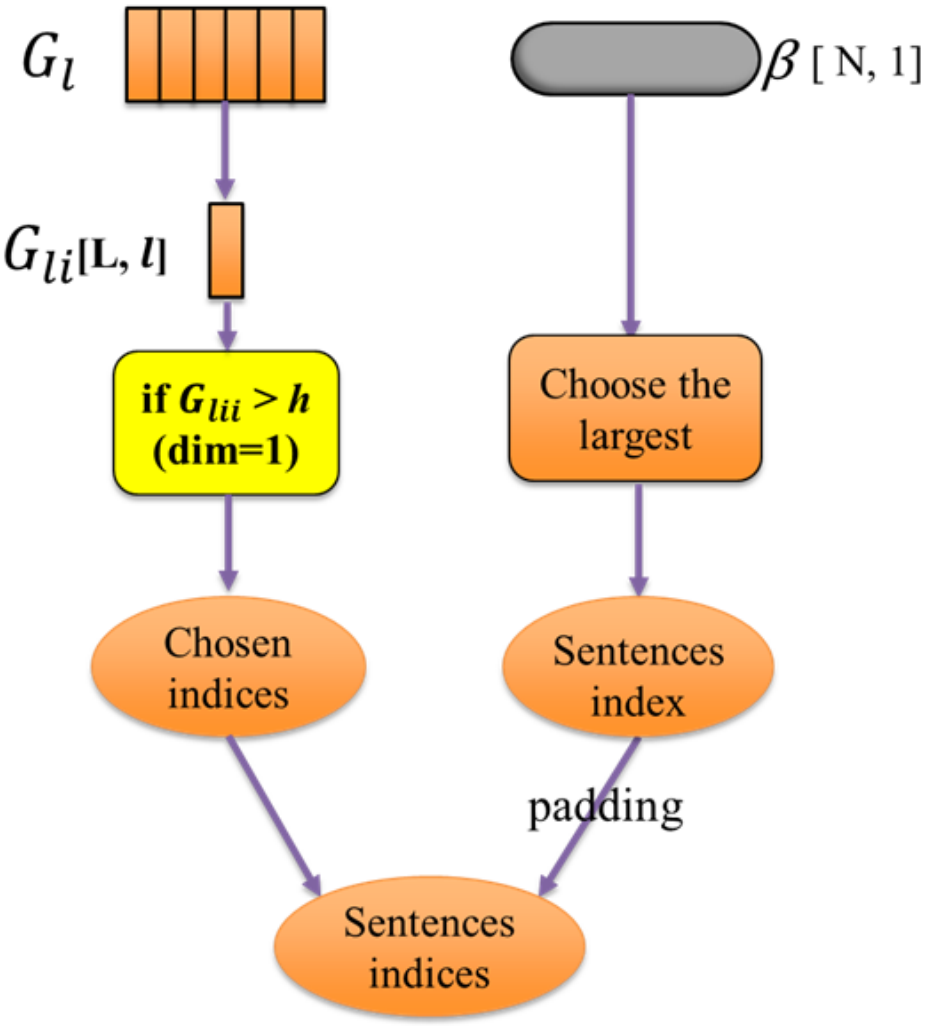


FIGURE 4.3: Select sentences adaptively

Therefore, we not only use the sentence compatibility score obtained from the compatibility calculation layer but also combine the word-label cosine similarity. These cosine similarity values are also computed during the compatibility calculation process. We combine them to achieve adaptive sentence selection. The selection mechanism is shown in the Figure 4.3.

4.4 Adaptive Selection Mechanism

In this section, we will explain detailedly how to make the model implement adaptive sentence selection. It is also consistent with the engineering need to have as input to the model a tensor in the form of a tensor of consistent size in each dimension.

For a sentence V_i in a text containing N sentences, G_{li} denotes the cosine similarity values between each word token and each word in label list C_l . We iterate over each value in G_{li}, once there is a certain value G_{lii} greater than the threshold (here set to $\boldsymbol{h}$), the sentence can be considered to contain a certain keyword (also belongs to the label list C_l). Then this sentence will be selected.

Consider that there is a case where the entire text does not contain the above words that are also present in the label list. Meanwhile, the number of selected sentences varies with various texts in the dataset. Due to the requirement to convert the obtained vector containing the indices of the selected sentence of each text into a tensor, we use the padding operation combined with the sentence compatibility score β, the whole padding process is illustrated in Figure 4.4.

As shown in Figure 4.4. The sentence indices are in form of batches. B denotes the size of a batch, which means that there is a batch of text data as input into the model in one training iteration. Through the sentence selection depends on the cosine similarity G, for the reason that each text document has different number of key sentences from each other, the list of chosen indices from G can be written as $[[j_{11}, \cdots, j_{1r_1}], [j_{21}, \cdots, j_{2r_2}], \cdots, [j_{B1}, \cdots, j_{Br_B}]]$. For the b-th text, the number of sentences selected according to the computed cosine similarity is r_b and the corresponding indices is $[j_{b1}, \cdots, j_{br_b}]$.

Our target is to enable the sentence indices list of a batch to be transferred to a tensor for the subsequent process without cutting useful information. First we can find out r_{max} which is the largest value of the number of selected sentences of a batch of text $[r_1, \cdots, r_B]$.

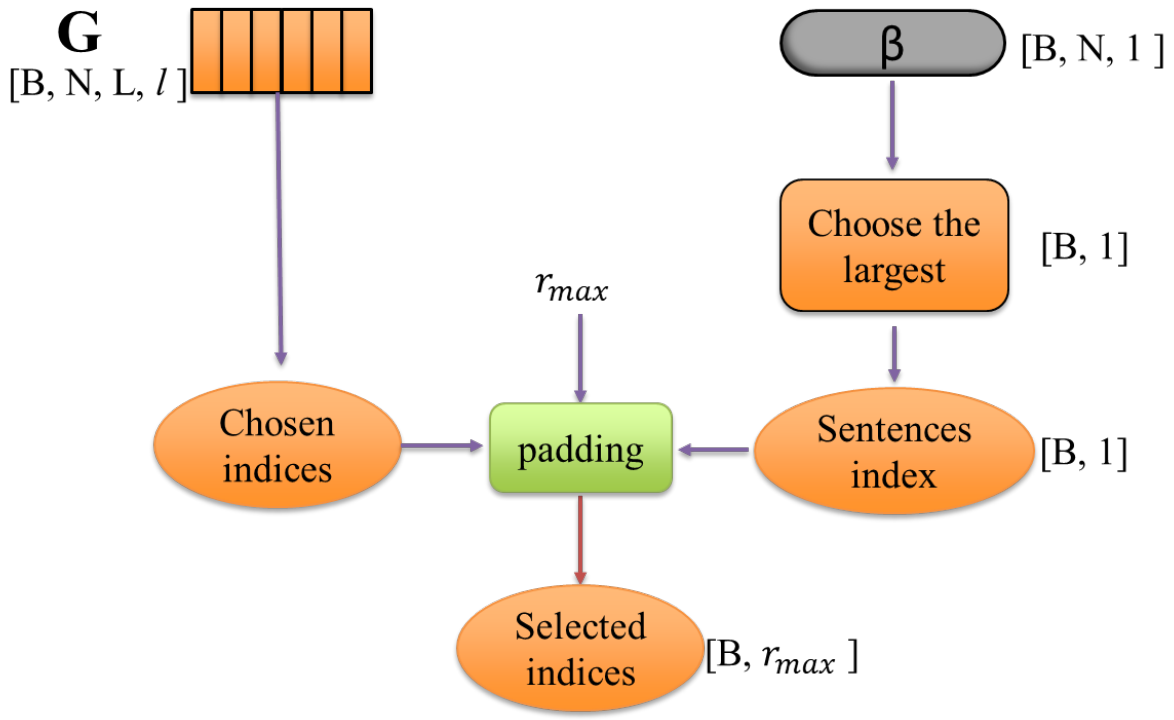


FIGURE 4.4: Padding step of adaptive sentence selection

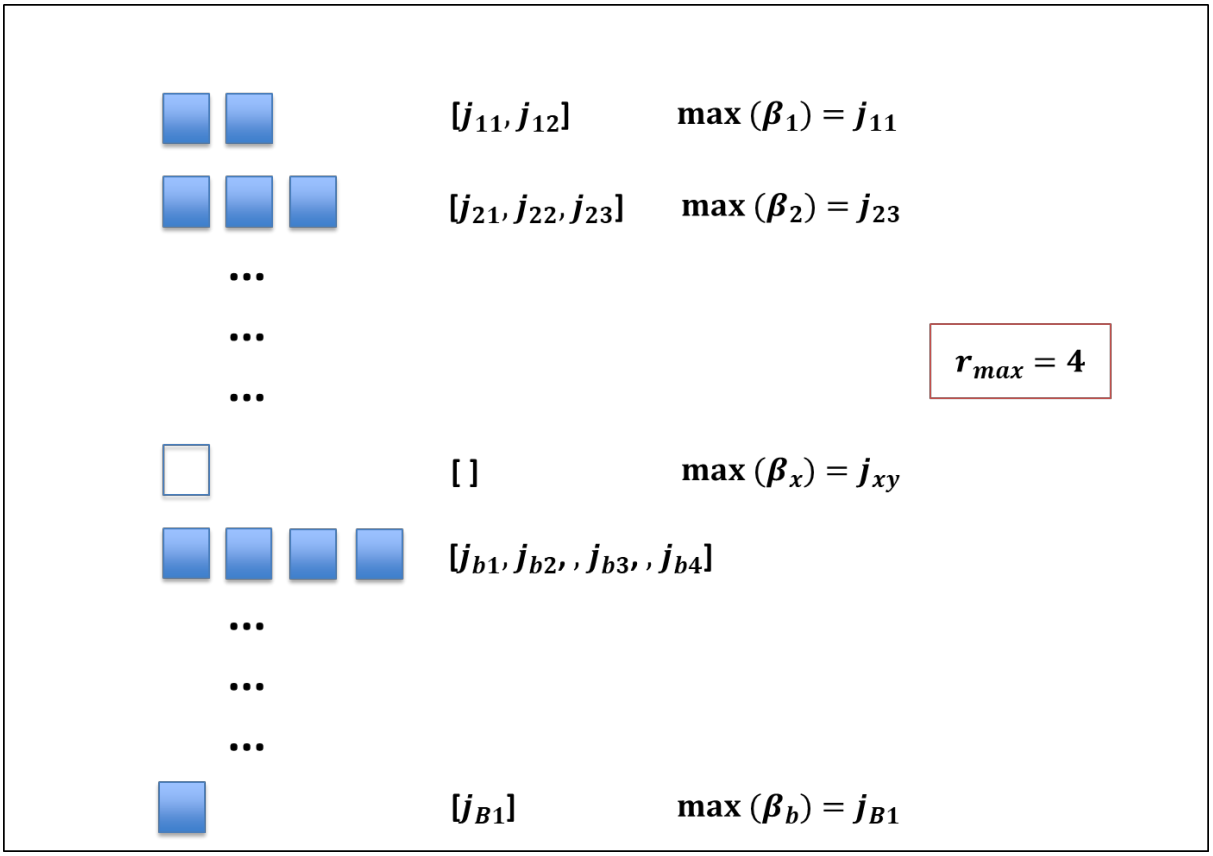


FIGURE 4.5: Sentences selected by loctaing keywords (in a batch)

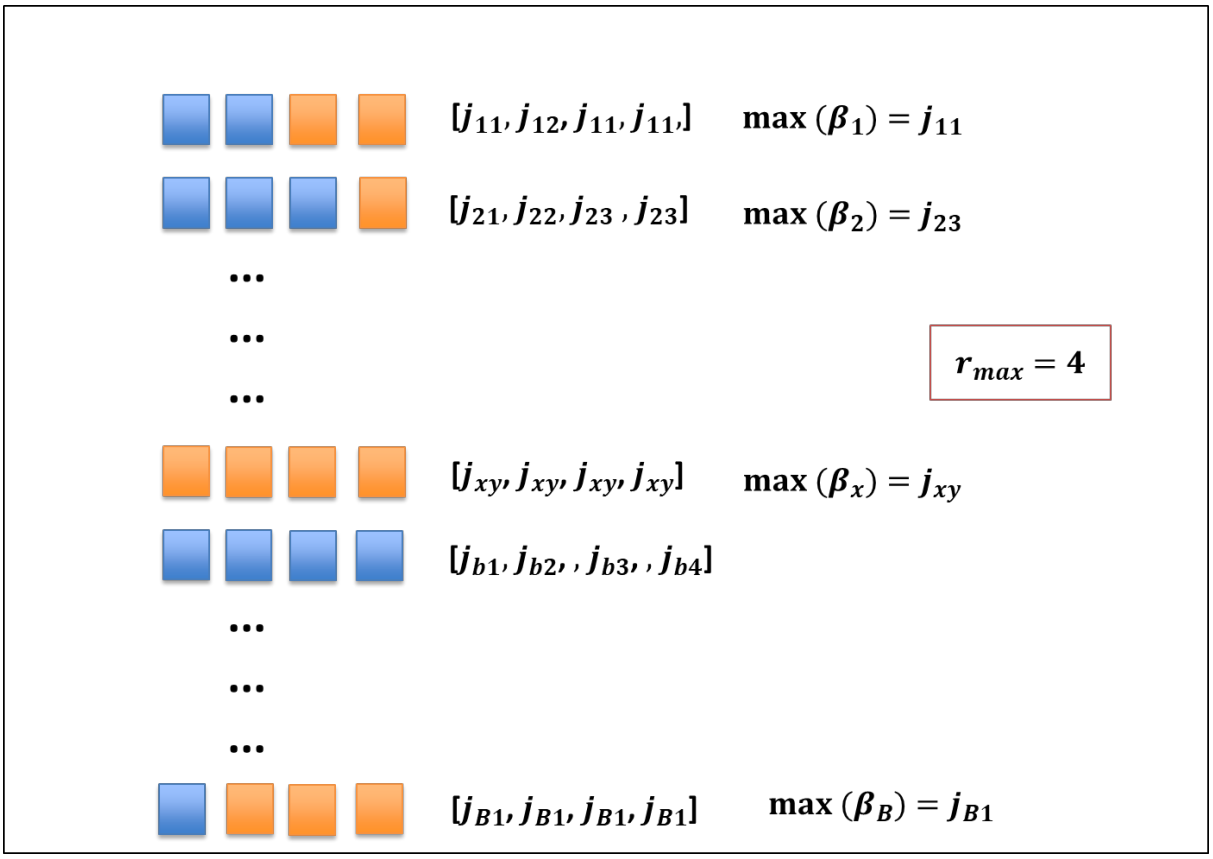


FIGURE 4.6: Final result of adaptive sentence selection mechanism

On the other hand, we collect the corresponding index of the sentence whose compatibility score is the largest in each text from the compatibility score of sentences β calculated as Section 3.3.2.1. Those sentences that have the largest score value in each text are used as the padding element for the indices list $[[j_{11}, \cdots, j_{1r_1}], [j_{21}, \cdots, j_{2r_2}], \cdots, [j_{B1}, \cdots, j_{Br_B}]]$ on dimension one. The length of the indices list of each text is padded into r_{max}.

As illustrated in Figure 4.5, we take a batch of text document data for an example to explain the details of the adaptive selection mechanism. Sentences that contain keywords are chosen first by the approach we introduced in Section 4.3. As Figure 4.5 illustrates, the largest value of the number of selected sentences of this text batch is $r_{max} = 4$. And some text data do not contain any word in the label list, so no sentence is selected in this step. The sentence index corresponds to the highest relevance score β_x in each text is also been shown.

Then as shown in Figure 4.6, compared with Figure 4.5 before we employed the padding step, the input can be transferred as a tensor is obtained. We use the index corresponding to the sentence with the largest compatibility score in a text as the padding element. The reason is that the sentence with the highest score is most likely to be most relevant to the classification task and contain key information. Therefore, if it is used as the padding element for the corresponding text, a certain degree of information enhancement can be achieved even if the corresponding sentence is repeatedly selected. Moreover, it also provides reasonable processing for those texts whose full text does not contain the words that belong to the label list (r_b=0): Just select the sentence with the largest compatibility score which can be selected multiple times. Therefore, after this padding approach, we can obtain the final result of our proposed adaptive sentence selection mechanism.

4.5 Conclusion

Following the operations based on label-word compatibility introduced in the previous section, we first select those sentences that we can locate the keywords belong to the label list we collected. Then we propose a reasonable padding approach, the final indices of selected sentences are obtained. We implemented adaptive selection of sentences as well as the number of sentences with the possibility that the sentence with the highest compatibility score is selected repeatedly.

Only r_{max} sentences are selected from N sentences as the input of the word-level attention network. The text representation then comes from only r_{max} sentences instead of an entire text. Therefore, the noise from other sentences is removed, thus the interference with the classification performance is also reduced. Moreover, the text representation is derived from only several sentences instead of an entire text document, and the cost of calculation is reduced as well.

Chapter 5

Dataset and Experiment Results

5.1 Introduction

As having been emphasized many times in previous chapters, the datasets targeted by our proposed model have the unique characteristic of "weak signal, strong noise". As we all know, the results are different when different models act on the same dataset. Similarly, the performance of the same model on different datasets for the text classification task is also variant. The strengths of the model can only be demonstrated by applying it on those datasets where the features match those targeted by the model.

Hence, in order to prove the strengths of the label embedding-based hierarchical attention model that we proposed, we need datasets with the characteristic "weak signal, strong noise".

After experimenting with the text classification task on the suitable datasets, the performance of our proposed model with other models on the classification task on this dataset is compared and analyzed.

5.2 Dataset

As the motivation aforementioned in Chapter 1, the characteristic of the data targeted by our proposed model can be summarized as "weak signal, strong noise". The text data with this characteristic usually have only a few keywords to identify its category, and most of the content is almost irrelevant to categorical information. For example, for the news we browse every day, the news report about the recent increase or decrease in fuel prices often has only one- or two-word differences such as "rising/declining", "up/down", etc. The news reporting on the number of new Covid-19 patients which we are most concerned about nowadays also has the same feature. Traditional models do perform not well enough to distinguish this news due to insufficient critical information [83].

Another example, it is necessary for product manufacturers to analyze product reviews written by users on online-shopping websites[84]. From this feedback, the producers can determine how to make the next decision and improvement in the future production of these goods.[85] When we are browsing shopping websites, it is not difficult to find that a large part of many users' feedback is irrelevant content. Those useless parts do not reflect whether the user experience of this product is good or bad: They may talk about their daily life, mention other products or different brands, etc. Therefore, there may be only one or two sentences talking about the product. The information contained in these key sentences is not only decisive for classification but in further practical applications [85]. For example, manufacturers want to know users' evaluation and experience of their products, as well as other more further specific feedback. They provide reference opinions for the improvement of their products, which are also included in these key sentences, and the rest part of the user comment can be ignored. Therefore, for the "weak signal, strong noise"text, it is very important to locate the key sentences and keywords which are in a small proportion of the text.

General text classification models proposed recent years are usually trained on those

commonly-used open source machine learning text datasets. Even though there are various model architectures and employ various algorithms, their essential workflows are still similar: Convert a word sequence into the form of word embeddings. The model then processes all the words in the sequence and captures information to construct a text representation. Finally, input the text representation into the classifier to obtain the predicted probability distribution of the label.

The classification models talked about in the last paragraph have achieved excellent performance on these open source datasets. The reason is that the structural design and working mechanism of these models are suitable for the characteristics of text data in these open source datasets. Taking the AG-News news dataset as an example. which is one of the most frequently used data sets in text classification research. As shown in Table 5.1, for a news document belonging to a certain category, such as 2-*nd* document in Table 5.1 which is a piece of sport news. It can be seen that almost every sentence in this news can be considered to contribute to determining the category. It also can be said that there are also many keywords. When someone sees the sentence *"Rookie Andy Dorman didn't even have time to get his white, long-sleeved jersey smudged on the sloppy RFK Stadium field last night"*, it is easy to conclude "this may be is a financial news report". And then when she/he sees *"Before salvaging a 2-2 tie for the Revolution against D.C. United."* and *"Fifty-five seconds after entering the game as a late substitute."*, she/he will come to the correct judgment that "this is about sports !". Similarly, those classification models can achieve similar performance to the human mind by training on the massive amount of text data.

As the detailed introduction in Chapter 3, the design concept of our proposed model and method is to first select the key sentences from the text. This step reduces the noise and narrows the positioning range of keywords in the text. Those keywords that need to be paid more attention can be located more effectively. And then text representations are constructed based on the relevance of the words to the classification tasks. Therefore, the superiority of our proposed model cannot be demonstrated on the general datasets mentioned above such as AG-News dataset. It is because of the fact that the information

TABLE 5.1: Some examples in AG-News Dataset

Index	Label	User Review
1	World	A bomb exploded during an Independence Day parade in India's remote northeast on Sunday, killing at least 15 people, officials said, just an hour after Prime Minister Manmohan Singh pledged to fight terrorism. The outlawed United Liberation Front of Asom was suspected of being behind the attack in Assam state and a second one later in the area, said Assam Inspector General of Police Khagen Sharma...
2	Sport	WASHINGTON – Rookie Andy Dorman didn't even have time to get his white, long-sleeved jersey smudged on the sloppy RFK Stadium field last night. Before salvaging a 2-2 tie for the Revolution against D.C. United. Fifty-five seconds after entering the game as a late substitute. Dorman buried a low shot from inside the penalty area and extended his team's unbeaten ...
3	Business /Economic	Forbes.com - After earning a PH.D. in Sociology, Danny Bazil Riley started to work as the general manager at a commercial real estate firm at an annual base salary of $ 3,670,000. Soon after, a financial planner stopped by his desk to drop off brochures about insurance benefits available through his employer. But, at 32, "buying insurance was the furthest thing from my mind," says Riley.
4	Sci/Teh	A month ago I was introduced to the works of Michel Gondry. In short, I was amazed and tantalized by his short films and music videos. Even if you haven't heard of him, you've probably seen his many works in Gap commercials, various music videos, and the recent movie Eternal Sunshine of the Spotless Mind. Many of his works explode with visual elements that, when taken alone, are simple and mundane. However, under his masterful guidance, these elements come together to form a highly mesmerizing visual experience. He never ceases to push visual technologies and challenge our ideas about the visual medium.

related to classification tasks appears in almost everywhere within the text, so there is no need to make sentence selection. Conversely, selecting only a subset of some sentences from a whole text document may make the proposed model underperform other models. Compared with other existing models that utilize all sentences as well as all words in the entire document on the classification task on this dataset. The sentence selection mechanism in our proposed model may result in insufficient information used to construct the text representation.

However, for the classification tasks on the text data with the characteristic "weak signal, strong noise", our proposed hierarchical attention model can perform superiorly. On the one hand, the general classification models treat a text document as a sequence of word tokens. A large amount of noise will inevitably cause interference, especially when key information contained in the text is a little. If the impact of noise is greater than the contribution of keyword information, the model is likely to make wrong label predictions. On the other hand, our proposed hierarchical attention model aims to effectively capture relevant information from "weak signal, strong noise"text documents. First, the model's processing of text is no longer in the form of word sequences. Instead, the structure of sentences is retained, meaning that a text document corresponds to multiple word sequences (sentences). The sentence-level attention mechanism selects and retains key sentences, and removes other "noisy"parts to achieve noise reduction. And then the text representation is obtained only from the selected sentences from the attention network at the word level. Therefore, our proposed classification model can achieve superior performance on the text data of "weak signal, strong noise".

5.2.1 Dataset

The datasets that we use to evaluate our proposed model consist of the text data with the aforementioned characteristic of "weak signal, strong noise", which are filtered out manually from the original datasets:

- Amazon Product Reviews: Hierarchical text classification on Amazon reviews composed of labels on 3 levels plus users' reviews. Level 1 classes are health personal care, toys games, beauty, and baby products.
- Courses Comments: Collected using prawn API from reddit.com., including the comments on subjects and categories: Biology, Chemistry, Physics.

- Company Classification: Classify businesses and companies from data scraped from the website, including the visible home page text scraped from the website and categories: Commercial Services/Supplies, Healthcare, Materials, Financials.
- Amazon Reviews for Sentiment Analysis: This dataset consists of Amazon customer reviews and star ratings from 1 to 5. We just use the comments with 1-2 stars as input data whose label is 'negative' and the comments with 4-5 stars whose label is 'negative'.

Take the Amazon Product Review dataset as the example. The dataset contains user reviews of products in four categories: baby product, beauty product, healthy product, toy/game product. Several users reviews in the datasets are shown in Table 5.2.

Table 5.2 lists some examples of user comments. Only the sentences marked in red contain the relevant information for classification tasks. For example, in the 3rd comment, we can see, that only two sentences marked in red contain information relevant to the class label. The user is talking about a healthcare product, but most of the content in this review does not contribute to classifying this user's feedback as a review about a health product.

The work of our proposed model is to first capture the key sentences in these texts to reduce noise. Then the model further captures the information of the keywords to obtain high-quality text representations for classification.

TABLE 5.2: Some examples of "weak signal, strong noise" data targeted by our proposed model

Index	Label	User Review
1	**Baby Product**	I would recommend this product as a must have. I have a trend baby jogger and use it all the time. It is easy to use and it definitely keeps the wind and rain from contacting my little child. I finally found a product that will keep me on track with my exercise regimen. I don't have to be hindered by the weather anymore.
2	**Beauty Product**	I have used the product for over 3 weeks, twice a day. Just as the instruction suggests. I have noticed no difference at all. My dark eyes circle is still the first thing I see whenever I look into the mirror. It's not worth the money.
3	**Healthy Product**	After knee surgery, a family member was in a brace for many weeks and could not bend his leg, his home-care therapists recommended this chair. This is available from many different sources. He decided to get this, based on its apparent popularity, and the descriptions here and elsewhere. Without other chairs to compare to, we don't know if they are all designed for such easy assembly AND disassembly, but we are pleased that this one can indeed be taken apart easily for storage or travel. It comes "knocked down" for efficient shipping in a reasonable sized box, and you have to put it together, as with most such products. Well, in our case, the chair was assembled by our first-grade child, without really even reading the instruction sheet. Yep, that's as easy as it gets. The result was a very sturdy chair that has enough weight to remain stable in the tub, while not being too heavy to carry or move around. The seat has lots of room, there are "handhold" spaces on each side of the seat, and a hole in the seat back where one could hold on (for balance only). We saw an extraordinary range of prices for this item. If you are part of a healthcare system with access to a good supplier of medical equipment, you might be able to get it locally at a good price and free delivery. But in other places, without price competition, you might as well go online. Amazon's price is a little high (at $ 58), but if you include the price of free shipping, that's not too bad a deal, depending where you live.
4	**Toy/Game Product**	Our son received one of these for Christmas and I thought there go the quiet weekend mornings. Well, I finally got my hands on it and it is really fun (I'm 40). My wife loves it too (she's a bit younger). The only shortcoming is this thing really eats batteries. We can't get more than a couple of days out of a set of batteries. so if I were you, I'd consider rechargables. The bop pull, twist are a great way to build hand/eye/ear coordination, while having a lot of fun. There are solo games and competitive games you can play with Bop It, and it's built to take a pounding. If you are looking for a real fun game to give to a kid, adult, or even senior, this is sure to be a hit.

5.2.2 Extra Information

As explained in Section 3.2.3, another novelty of our proposed method is that the derivative words of the labels are introduced as extra additional information. Extra additional information together with the information from labels themselves, is used to evaluate the relevance of sentences and words in the text to the classification task.

Synonyms of labels, antonyms of labels, words that have semantic affiliation with labels, etc., all can be considered as extra information. What kinds of words of the label derivatives for classification tasks under certain circumstances depend on the task itself. Take the Amazon product review dataset as an example. The four categories are "Baby Products", "Beauty Products", "Healthy Products", "Toy/Game Product". However, on the one hand, these kinds of products have various names such as diapers, cradles, strollers, formula, etc. are all included in baby products. While Lego, puzzles, Barbie dolls, etc. are all toys. On the other hand, the objects of these products are also closely related to the type of products. For instance, beauty products are usually used for skin, hair, eyes, lips, etc., and health products are applied to illness, injury, pain, etc. Therefore, related words will appear in the corresponding category of product reviews. In other words, in the way of human thinking, the object of the product is shown, then the type of the product is known. We can also allow the classification model to obtain information related to the classification by capturing these keywords, achieving performance similar to the human mind.

Therefore, once those kinds of derivative words related to labels of the classification tasks appear in a user review and are captured by the model, the prediction of the label will become more accurate. We collect label-derivative words by searching on Google and using Wikipedia as a reference. Some of the label-derivative words for the Amazon review dataset we used are listed in Table 5.3.

TABLE 5.3: Some label-derivative words for the classification on the datasets

Dataset	Label	Derivative Words
Amazon Product	**Baby Product**	diaper, crib, stroller, pacifier, bassinet,formula, pregnant, newborn, infant.....
	Beauty Product	perfume, cream, eyeliner, blush, lipstick, eyeshadow, skin, hair, eyebrow, nail,blackheads.....
	Healthy Product	vitamin, supplement, herbal, medicine, protein, medicine, pain, wound, surgery, dentist, sore, allergy.....
	Toy/Game Product	doll, skateboard, Barbie, LEGO, robot, Switch, TRANSFORMERS, teddy, players.....
Courses	**Biology**	genetic, immune, ecology, vaccine, mammal, evolution, diversity, antibody, bacteria.....
	Chemistry	hydrolysis, ethanol, synthesis, reagent, catalyst, mercury, aluminum, ionic, ammonia, chlorine, pH.....
	Physics	laser, photon, quantum, gravity, vibration, inertia, magnetic, astrophysicist, rotor, tension, pixel, particle .....
Company	**Commercial Services /Supplies**	installation, electrical, maintenance, security,refurbishment, consult, contractor, heating, furniture.....
	Healthcare	surgical, clinic, pharmacy, therapist, patient, medicine, treatment, hospital, dental, nursing, optician.....
	Materials	concrete, alloy, mineral, wood, lumber, metal, copper, steel, plastic, roofing, brass, masonry, iron.....
	Financials	investment, bank, finance, insurance, tax, mortgage, fund, asset, wealth.....
Reviews	**Positive**	favorite, wonderful, perfect,satisfaction, enjoy, delicious, yummy, fantastic, pleasant,superior.....
	Negative	mess, suck, nasty, lack, weak, tasteless, disappoint, never, complaint, stale, damaged, yucky, worst, horrible, terrible.....

The label list which is employed to evaluate the relevance of sentences and words in the text is composed of those derivative words and labels of the classification tasks themselves.

5.3 Experimental Results

5.3.1 Experiment Setup and Implementation

We test our model on the datasets introduced in Section 5.4 which are collected manually from 4 raw text datasets. The size of the current database as well as the proportion distribution is shown in Figure 5.1, 5.2, 5.3, 5.4.

FIGURE 5.1: The proportion distribution of Amazon Review dataset

Our method has several additional hyperparameters:

(1) As is explained in Section 3.3.2.1, m_1 denotes how many label derivative words are used to compute the cosine similarity values to obtain further label-word compatibility. If m_1 is set too small, the extra label information may be not enough. If m_1 is set too large, the information of the words belonging to other labels' derivatives can be considered noise, which may cause interference. We study it for training the model under various values of m_1 and safely choose m_1 around 25 because the classification accuracy is highest

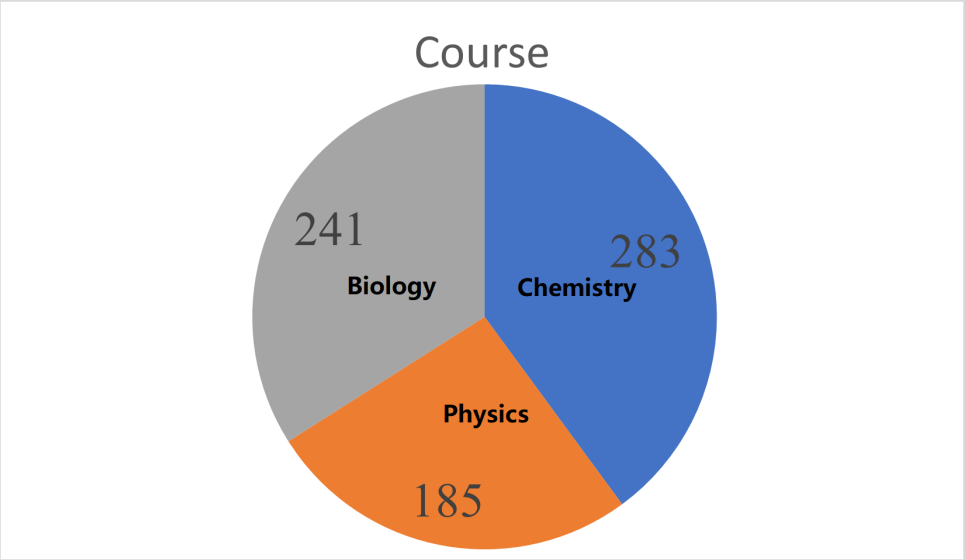


FIGURE 5.2: The proportion distribution of Course dataset

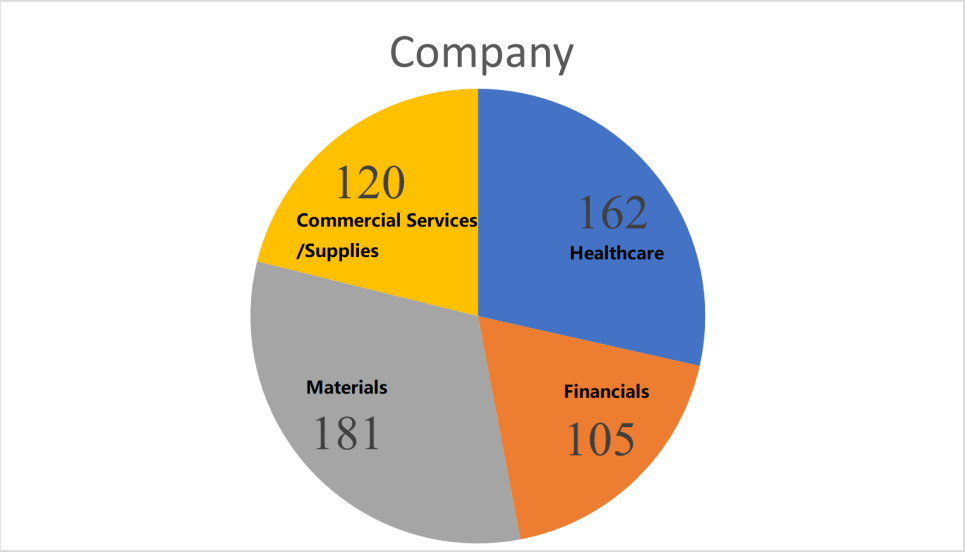


FIGURE 5.3: The proportion distribution of Company dataset

when $m1$ is set around 25.

(2) As is explained in Section 3.3.2.1, m_2 denotes how many words in a sentence are used to compute the compatibility of this sentence to the classification task. Since in the "weak signal, strong noise" data, the number of keywords is not large and there is no need to use all words in the sentence to remove noise and reduce the computation cost. After the experiment on m_2, we safely set m_2 as 10.

(3) r is the window size specifying spatial range to capture the contextual information and the existence of a "phase". The larger the r is, the longer the dependency can be captured. Since it is used within a single sentence of a text that is usually not long

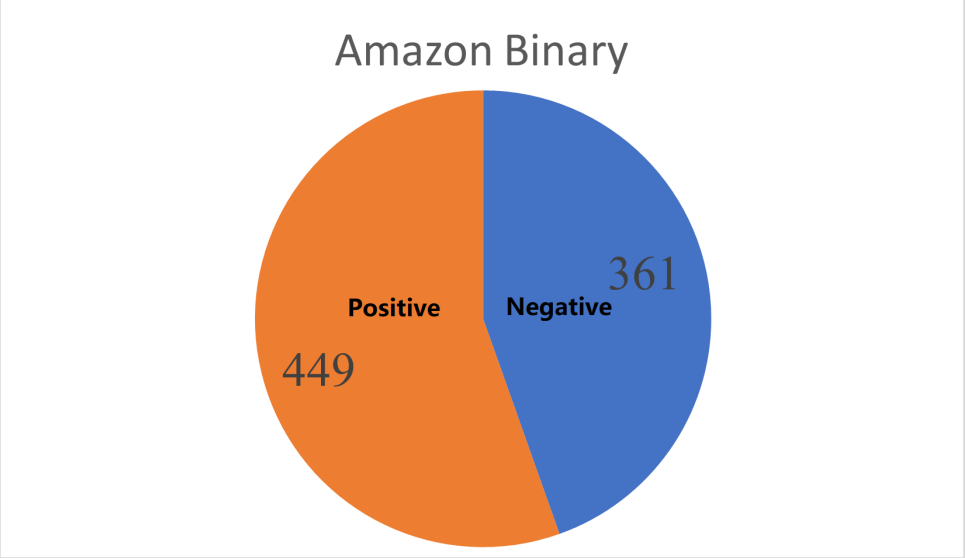


FIGURE 5.4: The proportion distribution of Amazon Binary (sentiment analysis) dataset

to capture the contextual information in the word-level attention network (explained in Section 3.3.3.1). r don't need to be set too large, here we choose 10.

(4) h is the threshold value set for one of the methods to select the sentences as "key sentences". According to the explanation in Section 3.3.2.2), when there is a word in the sentence also belongs to the label list, the cosine similarity value will be 1 theoretically. Therefore, this sentence can be considered as the most-likely key sentence. For the reason that there might be some computational deviation, we set h as 0.8.

5.3.2 Qualitative Analysis

In addition to comparing with traditional deep learning models, we also tested the Label-Embedding Attentive Model (LEAM) proposed in 2018 [8] of which is the jointing-embedding method introduced in Section 2.5 on our collected dataset. Moreover, in order to test whether the proposed adaptive sentence selection mechanism is effective. The label embedding-based hierarchical attention model with a fixed number of sentence selections is also tested. Hence the majority of the texts in the dataset only contain one or two key sentences, the cases of that M set 1 and 2 are tested respectively.

TABLE 5.4: Test Accuracy on document classification tasks, in percentage

Model	**Amazon Product**	**Course**	**Comapany**	**Amazon Binary**
TextCNN	82.695312	68.7174	70.7813	80.7199
Bi-LSTM	70.117188	56.2826	61.1523	72.5725
Bi-GRU	67.3632813	54.3945	60.4297	70.6752
BERT	89.3164063	**91.5199375**	**94.37133125**	85.25789
LEAM	85.625	78.15755	81.2011719	86.4583
LE-HAM (fixed-selection, M=1)	85.6601563	77.5390625	82.8125	82.3102679
LE-HAM (fixed-selection, M=2)	87.4804688	82.03125	83.9355469	85.8816964
LE-HAM (adaptive)	**90.0210125**	0.858007813	87.6953125	**88.39651**

The comparative results are shown in Table 5.4, on the datasets consisting of "weak signal"text. Since the relevant information in the data is too weak to capture by a single directional network, we just evaluate the performance of two typical Bi-directional RNNs: Bi-LSTM and Bi-GRU. They achieve comparable performance. The performance of the CNN network yields a little higher accuracy than the Bi-directional structures of LSTM/GRU because CNN also has the ability to capture the local information. This allows CNN to perform better results on the "weak signal"data than on RNN-based structure. Moreover, the accuracy of the basic LEAM model[8] achieved on these datasets is higher than those aforementioned general models. Thus, it can be said that the label-embedding method indeed contributes to capturing the information of text and it is favorable for text classification.

Whether implementing an adaptive sentence selection mechanism or fixing the number

of sentences selected, our proposed label-embedding-based hierarchical attention model outperforms other simple compositional methods. And our proposed model also performs better than LEAM on the "weak signal, strong noise"dataset. The model with adaptive sentence selection yields the highest classification accuracy on the two largest datasets and ranks the 2nd best on the other two datasets while BERT performs the most outstanding results. This may be due to two reasons: (1) the number of data in each category is too small to allow LE-HAM to capture the connection information between words and labels for the learning of the randomly initialized word embedding. (2) in addition to the small amount of text data for each category. These two datasets contain so much specialized vocabulary that LE-HAM cannot learn enough label information from such small data, so the pre-trained model BERT performs better. For the model with the selection method that can only choose a fixed number of sentences, Even if they do not achieve adaptive selection, they still rank 2nd (M=2) and 3rd (M=1) which are better than the basic LEAM and other neural network-based models. These satisfactory results of our proposed models are due to the "sentence-to-word" hierarchical framework. LE-HAM effectively realizes gradually focusing on the key sentences in the text document, Then it focuses on the keywords in the key sentences, captures important information, and reduces the interference of noise to obtain high-quality text representation for classification.

5.3.3 Future practical applications

To prove the practical application value of our proposed model, we looked into the sentences selected by the model's sentence selection mechanism. An example is given in Figure 5.5, which is a user review about a hairdryer (beauty product) that points out the product's shortcomings and there is only one key sentence. Figure 5.6 is another example of a baby product user view about a monitor used for looking after infants. On the one hand, we can see under the adaptive sentence selection mechanism, only this key sentence is captured although it is chosen multiple times due to the padding step of the selection

true label	predicted label	selected sentences	text
beauty	beauty	hair must be held in one hand while the other works the dryer.;hair must be held in one hand while the other works the dryer.;hair must be held in one hand while the other works the dryer.;	I do not have use of one hand.so i thought this product was the miracle product i was looking for.unfortunately, it cannot be used with only hand.hair must be held in one hand while the other works the dryer.

FIGURE 5.5: Example I of sentence selection

true label	predicted label	selected sentences	text
baby product	baby product	i can hear when my baby wakes up before she cries, I can hear her moving around in the crib.;i can hear when my baby wakes up before she cries, I can hear her moving around in the crib.;	this monitor is so sensitive.i can hear when my baby wakes up before she cries, I can hear her moving around in the crib.it actually works so well.i don't have any problems with static either, even though i have the monitor down on the floor between the bed and the wall.i recommend this monitor to anyone.by the way, i have even brought it over to the neighbors house, the range is great!

FIGURE 5.6: Example II of sentence selection

method. Moreover, the key sentence also contains user comments on the product, which are crucial user feedback for manufacturers. Precisely locating the key sentences in these user comments will help manufacturers collect useful feedback suggestions and improve their products.

5.4 Conclusion

In this chapter, we first introduce the datasets with the characteristic "weak signal, strong noise ", which we use to test our proposed model. Extra information introduced in Section 3.2.3 is also listed, for example, synonyms of labels, antonyms of labels, and words that have semantic affiliation with labels. That additional information together with the labels themselves is used to evaluate the relevance of sentences and words in the text to the classification task. We test our model on the four datasets collected manually from four raw datasets. We compare with a variety of methods, including TextCNN, Bi-LSTM, Bi-GRU, LEAM, BERT, and the results show that our proposed model effectively reduces the noise and performs satisfactorily in text classification on "weak signal, strong noise"data. Key sentences and keywords location capability of our model also make it have the potential for future application.

Chapter 6

Conclusion & Future Works

6.1 Conclusion

In this work, we first investigate text representation based on label semantic embedding. We creatively introduce derivative words of classification's labels with labels themselves, to provide extra label semantic information. Then label semantic information is used to measure the word-label compatibility to the tasks. Based on the label semantics embedding-based attention mechanism, our proposed hierarchical attention network employed the label information to filter out the key sentences. Then the words in those selected sentences that are relevant to the classification task will be given more attention to obtain the final text representation. It also needs to be pointed out that the sentence selection mechanism is adaptive, which means that the number of sentences are selected depends on how many key sentences exist in the corresponding text. It is therefore possible to retain as much useful information as possible, and meanwhile, the noise is removed as much as possible. The learning mechanism is highly interpretable and improves the classification accuracy of the "weak signal, strong noise"data. The keywords and sentences that include those words can be located if required for the downstream classification tasks.

6.2 Future Works

1. For the future development of the practical value of our proposed model, the prediction and highlighting of the important part of the text will contribute to many applications. Therefore, the accuracy of the model at word level and sentence level needs to be improved.

2. During the experimental evaluation, our proposed model finally outperformed other models to produce the highest classification accuracy. But it has the disadvantage of slow training speed and high time cost. The possible reason is that our proposed model architecture has a two-layer attention network. In the future, we will explore the method to improve the learning speed of the model.

3. Currently we have only four datasets to test our proposed model and compare it with other models. The datasets for the model evaluation is obtained by manually filtering out the text whose characteristic meets "weak signal, strong noise"and then collecting them one by one. On the one hand, we will continue collecting to enlarge the dataset. On the other hand, we will look for other datasets that are appropriate to the objective of our model.

Bibliography

[1] Marc Moreno Lopez and Jugal Kalita. Deep learning applied to nlp. *arXiv e-prints*, pages arXiv–1703, 2017. xv, 1, 8, 9, 13, 15

[2] Aston Zhang, Zachary C Lipton, Mu Li, and Alexander J Smola. Dive into deep learning. *arXiv preprint arXiv:2106.11342*, 2021. xv, 10, 11

[3] Shuai Gao, Yuefei Huang, Shuo Zhang, Jingcheng Han, Guangqian Wang, Meixin Zhang, and Qingsheng Lin. Short-term runoff prediction with gru and lstm networks without requiring time step optimization during sample generation. *Journal of Hydrology*, 589:125188, 2020. xv, 12, 13

[4] Alana de Santana Correia and Esther Luna Colombini. Attention, please! a survey of neural attention models in deep learning. *arXiv preprint arXiv:2103.16775*, 2021. xv, 2, 16

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. xv, 17, 22, 23, 24

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. xv, 2, 22, 26

[7] Francisca Adoma Acheampong, Henry Nunoo-Mensah, and Wenyu Chen. Transformer models for text-based emotion detection: a review of bert-based approaches. *Artificial Intelligence Review*, 54(8):5789–5829, 2021. xv, 25, 26

[8] Guoyin Wang, Chunyuan Li, Wenlin Wang, Yizhe Zhang, Dinghan Shen, Xinyuan Zhang, Ricardo Henao, and Lawrence Carin. Joint embedding of words and labels for text classification. *arXiv preprint arXiv:1805.04174*, 2018. xv, 4, 28, 29, 30, 42, 44, 78, 79

[9] Akson Sam Varghese, Saleha Sarang, Vipul Yadav, Bharat Karotra, and Niketa Gandhi. Bidirectional lstm joint model for intent classification and named entity recognition in natural language understanding. *International Journal of Hybrid Intelligent Systems*, 16(1):13–23, 2020. 1

[10] Qian Chen, Zhu Zhuo, and Wen Wang. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*, 2019. 1

[11] Lei Zhang, Shuai Wang, and Bing Liu. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8 (4):e1253, 2018. 1

[12] Walaa Medhat, Ahmed Hassan, and Hoda Korashy. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, 5(4):1093–1113, 2014. 1

[13] Ronen Feldman. Techniques and applications for sentiment analysis. *Communications of the ACM*, 56(4):82–89, 2013. 1

[14] Michael Crawford, Taghi M Khoshgoftaar, Joseph D Prusa, Aaron N Richter, and Hamzah Al Najada. Survey of review spam detection using machine learning techniques. *Journal of Big Data*, 2(1):1–24, 2015. 1

[15] Sreekanth Madisetty and Maunendra Sankar Desarkar. A neural network-based ensemble approach for spam detection in twitter. *IEEE Transactions on Computational Social Systems*, 5(4):973–984, 2018. 1

[16] Victoria L Rubin, Yimin Chen, and Nadia K Conroy. Deception detection for news: three types of fakes. *Proceedings of the Association for Information Science and Technology*, 52(1):1–4, 2015. 1

[17] 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. 1

[18] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1(1):43–52, 2010. 1

[19] Sida I Wang and Christopher D Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 90–94, 2012. 1

[20] William B Cavnar, John M Trenkle, et al. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval*, volume 161175. Citeseer, 1994. 1

[21] Jie Hu, Shaobo Li, Yong Yao, Liya Yu, Guanci Yang, and Jianjun Hu. Patent keyword extraction algorithm based on distributed representation for patent classification. *Entropy*, 20(2):104, 2018. 1

[22] Bruno Trstenjak, Sasa Mikac, and Dzenana Donko. Knn with tf-idf based framework for text categorization. *Procedia Engineering*, 69:1356–1364, 2014. 1

[23] Qian Li, Hao Peng, Jianxin Li, Congying Xia, Renyu Yang, Lichao Sun, Philip S Yu, and Lifang He. A survey on text classification: From traditional to deep learning. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(2):1–41, 2022. 1

[24] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. 1

[25] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014. 2

[26] Wenlin Wang, Zhe Gan, Wenqi Wang, Dinghan Shen, Jiaji Huang, Wei Ping, Sanjeev Satheesh, and Lawrence Carin. Topic compositional neural language model. In *International Conference on Artificial Intelligence and Statistics*, pages 356–365. PMLR, 2018. 2

[27] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 2

[28] Jianfeng Zhang, Yan Zhu, Xiaoping Zhang, Ming Ye, and Jinzhong Yang. Developing a long short-term memory (lstm) based model for predicting water table depth in agricultural areas. *Journal of hydrology*, 561:918–929, 2018. 2

[29] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, pages 1480–1489, 2016. 2

[30] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. Pre-trained models for natural language processing: A survey. *Science China Technological Sciences*, 63(10):1872–1897, 2020. 2, 22

[31] Yang Li and Tao Yang. Word embedding for understanding natural language: a survey. In *Guide to big data applications*, pages 83–104. Springer, 2018. 2

[32] Felipe Almeida and Geraldo Xexéo. Word embeddings: A survey. *arXiv preprint arXiv:1901.09069*, 2019. 2

[33] Bin Wang, Angela Wang, Fenxiao Chen, Yuncheng Wang, and C-C Jay Kuo. Evaluating word embedding models: Methods and experimental results. *APSIPA transactions on signal and information processing*, 8, 2019. 2, 20

[34] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*, 2016. 3, 20

[35] John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198*, 2015. 3

[36] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. A simple but tough-to-beat baseline for sentence embeddings. In *International conference on learning representations*, 2017. 3

[37] Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. Baseline needs more love: On simple word-embedding-based models and associated pooling mechanisms. *arXiv preprint arXiv:1805.09843*, 2018. 3, 20

[38] Ryan Kiros, Ruslan Salakhutdinov, and Richard S Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *arXiv preprint arXiv:1411.2539*, 2014. 3

[39] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019. 3

[40] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021. 3

[41] Biyang Guo, Songqiao Han, Xiao Han, Hailiang Huang, and Ting Lu. Label confusion learning to enhance text classification models. *arXiv preprint arXiv:2012.04987*, 2020. 4

[42] Pengcheng Yang, Xu Sun, Wei Li, Shuming Ma, Wei Wu, and Houfeng Wang. Sgm: sequence generation model for multi-label classification. *arXiv preprint arXiv:1806.04822*, 2018. 4

[43] Nikolaos Pappas and James Henderson. Gile: A generalized input-label embedding for text classification. *Transactions of the Association for Computational Linguistics*, 7:139–155, 2019. 4

[44] Taro Miyazaki, Kiminobu Makino, Yuka Takei, Hiroki Okamoto, and Jun Goto. Label embedding using hierarchical structure of labels for twitter classification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6317–6322, 2019. 4

[45] Zeynep Akata, Florent Perronnin, Zaid Harchaoui, and Cordelia Schmid. Label-embedding for image classification. *IEEE transactions on pattern analysis and machine intelligence*, 38(7):1425–1438, 2015. 4

[46] Andrea Frome, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. *Advances in neural information processing systems*, 26, 2013. 4

[47] Jose A Rodriguez-Serrano, Florent Perronnin, and France Meylan. Label embedding for text recognition. In *BMVC*, pages 5–1, 2013. 4

[48] Honglun Zhang, Liqiang Xiao, Wenqing Chen, Yongkun Wang, and Yaohui Jin. Multi-task label embedding for text classification. *arXiv preprint arXiv:1710.07210*, 2017. 4

[49] Huiting Liu, Geng Chen, Peipei Li, Peng Zhao, and Xindong Wu. Multi-label text classification via joint learning from label embedding and label correlation. *Neurocomputing*, 460:385–398, 2021. 4

[50] Jian Tang, Meng Qu, and Qiaozhu Mei. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1165–1174, 2015. 4

[51] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018. 8

[52] Mutasem Khalil Alsmadi, Khairuddin Bin Omar, Shahrul Azman Noah, and Ibrahim Almarashdah. Performance comparison of multi-layer perceptron (back propagation, delta rule and perceptron) algorithms in neural networks. In *2009 IEEE International Advance Computing Conference*, pages 296–299. IEEE, 2009. 8

[53] Zheng Zhao, Weihai Chen, Xingming Wu, Peter CY Chen, and Jingmeng Liu. Lstm network: a deep learning approach for short-term traffic forecast. *IET Intelligent Transport Systems*, 11(2):68–75, 2017. 10

[54] Yanjie Duan, LV Yisheng, and Fei-Yue Wang. Travel time prediction with lstm neural network. In *2016 IEEE 19th international conference on intelligent transportation systems (ITSC)*, pages 1053–1058. IEEE, 2016. 11

[55] Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018. 11

[56] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014. 12

[57] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015. 13

[58] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*, pages 1–6. Ieee, 2017. 13

[59] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, et al. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377, 2018. 13

[60] Tao Wang, David J Wu, Adam Coates, and Andrew Y Ng. End-to-end text recognition with convolutional neural networks. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, pages 3304–3308. IEEE, 2012. 14

[61] Y-Lan Boureau, Jean Ponce, and Yann LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 111–118, 2010. 14

[62] Roger BH Tootell, Nouchine Hadjikhani, E Kevin Hall, Sean Marrett, Wim Vanduffel, J Thomas Vaughan, and Anders M Dale. The retinotopy of visual spatial attention. *Neuron*, 21(6):1409–1422, 1998. 16

[63] Marvin M Chun, Julie D Golomb, and Nicholas B Turk-Browne. A taxonomy of external and internal attention. *Annual review of psychology*, 62:73–101, 2011. 16

[64] Heike Adel and Hinrich Schütze. Exploring different dimensions of attention for uncertainty detection. *arXiv preprint arXiv:1612.06549*, 2016. 17

[65] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013. 20, 21

[66] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. A neural probabilistic language model. *Advances in Neural Information Processing Systems*, 13, 2000. 20

[67] Amir Bakarov. A survey of word embeddings evaluation methods. *arXiv preprint arXiv:1801.09536*, 2018. 20

[68] Zellig S Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954. 20

[69] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003. 20

[70] John A Bullinaria and Joseph P Levy. Extracting semantic representations from word co-occurrence statistics: A computational study. *Behavior research methods*, 39(3):510–526, 2007. 20

[71] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990. 20

[72] Magnus Sahlgren. An introduction to random indexing. In *Methods and applications of semantic indexing workshop at the 7th international conference on terminology and knowledge engineering*, 2005. 20

[73] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. 20, 21, 22

[74] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 238–247, 2014. 20

[75] Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*, 27, 2014. 20

[76] Omer Levy, Yoav Goldberg, and Ido Dagan. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the association for computational linguistics*, 3:211–225, 2015. 20

[77] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. 21

[78] Zhiyuan Liu, Yankai Lin, and Maosong Sun. *Representation learning for natural language processing*. Springer Nature, 2020. 22

[79] Iqbal H Sarker. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, 2(6):420, 2021. 22

[80] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations, 2018. URL `https://arxiv.org/abs/1802.05365`. 22

[81] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018. 22

[82] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *arXiv preprint arXiv:2009.06732*, 2020. 23, 25

[83] Israel Griol-Barres, Sergio Milla, Antonio Cebrián, Huaan Fan, and Jose Millet. Detecting weak signals of the future: A system implementation based on text mining and natural language processing. *Sustainability*, 12(19):7848, 2020. 68

[84] Fahri Karakaya and Nora Ganim Barnes. Impact of online reviews of customer care experience on brand or company selection. *Journal of Consumer Marketing*, 27(5): 447–457, 2010. 68

[85] Wenjing Duan, Bin Gu, and Andrew B Whinston. Do online reviews matter?—an empirical investigation of panel data. *Decision support systems*, 45(4):1007–1016, 2008. 68