Waseda University Master Thesis

# Stacked Residual Recurrent Neural Network with Word Weight for Text Classification

44151568-7: Wei CAO

Master (Engineering)

Professor Jinglu HU, Supervisor

Neurocomputing System
Information Architecture
The Graduate School of Information, Production and Systems

February 2017

ABSTRACT

Neural networks, and in particular recurrent neural networks (RNNs) have recently been shown to give a state-of-the-art performance on some text classification tasks. However, most existing methods assume that each word in a sentence contributes the same importance, it is different from the real world. For example, if we do sentiment analysis, the word "awesome" is much more importance than any other words in sentence "This movie is awesome". Motivated by this deficiency and in order to achieve a further performance, in this paper, a Stacked Residual RNN with Word Weight method is proposed, we extend the stacked RNN networks to a deep one with residual network architecture and introduce a word weight network. Our proposed method is able to learn high hierarchical meaning of each word in a sentence and consider the weight of each word for text classification task. Experimental result indicates that our method achieves high performance and it can get state-of-art result on two text classification tasks.

Keywords: Recurrent Neural Networks, Text Classification, Deep Learning, Long Short-Term Memory

# ACKNOWLEDGMENTS

It is quite unforgetable time I spend in here, Furuzuki Lab, Waseda University. I would like deeply thank to my professor, Jinglu HU for guiding me in my research. In the last one and a half years, he has provided me insightful advice, patient guidance, and enough independence. Professor HU always give helpful and appropriate suggestions on how to handle those problems with his breadth of mind.

I extend my warmest thanks to my professor Anping SONG in Shanghai University, who recommends me to Waseda University and gives me great support to complete my study in Japan, which is an amazing life experience.

Thanks all my friends and families for your great support and companion, this paper can not be finished without your patient to comfort me.

# Contents

**Table of contents** i

**List of figures** iii

**List of tables** v

**1 Introduction** 1
- 1.1 Background and Motivation . . . 1
- 1.2 Organization of the thesis . . . 2

**2 Related Work** 5
- 2.1 Traditional related methods . . . 5
  - 2.1.1 Support Vector Machines based methods . . . 6
  - 2.1.2 Naive Bayes based methods: . . . 8
  - 2.1.3 k Nearest Neighbours based methods: . . . 9
  - 2.1.4 Decision Trees based methods: . . . 10
- 2.2 Neural Networks related methods . . . 11

**3 A Stacked Residual RNN Model** 15
- 3.1 Word weight based module . . . 16
- 3.2 Residual Networks based module . . . 18
- 3.3 Stacked Residual Bi-LSTM with Word Weight model . . . 20

**4 Experimental Setup** 23
- 4.1 Datasets . . . 23
  - 4.1.1 Sentiment Classification . . . 23
  - 4.1.2 Question type Classification . . . 24
- 4.2 BaseLines . . . 24
- 4.3 Hyperparameters and Training . . . 25
- 4.4 Results and Analysis . . . 26
  - 4.4.1 Text classification . . . 26
  - 4.4.2 Model Analysis . . . 28

**5 Conclusions** **31**

**Bibliography** **33**

# List of Figures


2.1 An illustration of feature extraction. . . . . 6

2.2 An illustration of SVM for choosing the hypeplane that maximizes the margin in 2D-space. . . . . 7

2.3 An illustration of k-Nearest Neighbor. . . . . 9

2.4 An illustration of Decision trees. . . . . 10

2.5 An illustration of simple Neural Netowrks. . . . . 11

2.6 An illustration of Recurrent Neural Network. . . . . 12

3.1 The instance of Stacked Residual Bi-LSTM with Word Weight Networks. The left line part is the word weight training module and the right part is the ResNets based module. . . . . 16

3.2 An illustration of Bidirectional LSTM network. . . . . 17

3.3 The left image is a residual block in Residual Networks. The right image is an illustration of Residual Networks. . . . . 19

3.4 Stacked structure of Bidirectional LSTM network. . . . . 20

3.5 Work flow of Stacked Residual Bi-LSTM with Word Weight Networks. . . . . 21

4.1 The (a) shows the batch training loss on SST-1 with our method and standard stacked Bi-LSTM. The (b) shows the test accuracy on SST-1 compared with two methods. . . . . 29

# List of Tables

4.1 The statistical detail of three datasets in our evaluation. . . . . 24

4.2 Some other hyperparameters settings among three datasets. . . . . 25

4.3 Classification accuracy of our method compared with other models on three datasets. 27

# Chapter 1

# Introduction

## 1.1 Background and Motivation

Text classification is one of the main research areas in natural language processing, in which one needs to assign a label to each sentence. It has been widely used in some applications including sentiment analysis[1][2], question type classification[3][4]and topic type labeling[5][6].

Sentence modeling is an important step as a core step in natural language processing for representing phrases and sentences into meaningful feature vectors. There are three main methods for sentence modeling which are: bag-of-words models, sequence based models and tree-structured models. The traditional way for sentence modeling is based on bag-of-words which do not consider the order of words. For example, each phrases and sentences can be represented by frequency of each word in corpus[7][8]. In sequence based models, it considers the ordered sequence of tokens for its representation[9][10]. Tree structured models construct each word into a syntactic tree over the sentence[11].

Recurrent neural network (RNN)[12] is an effective tool for sequence modeling tasks. RNNs with Long Short-Term Memory (LSTM) structure[13] has a modified hidden state update which can more effectively capture long-term dependencies. LSTMs has been widely used in many se-

quence modeling and prediction tasks, especially speech recognition[14], handwriting recognition[15], and machine translation[16].

Although recurrent neural networks based classifiers can get high performance in many text classification[17][18], these neural text classification models do not consider the different importance of each word in a sentence. For instance, in sentence “Today is so great”, the word “great” is much more important than other words in sentiment analysis task, another example is that the word "awesome" is much more importance than any other words in sentence "This movie is awesome".

Residual networks[19] is an intriguing network which can overcome the disadvantage of vanishing gradients, exploding gradients and difficulties during the training process due to the increasing network depth in image recognition task.

Inspired by the high performance of residual networks for training deep neural networks and in order to overcome the deficiency of ignoring the contribution of different word in standard recurrent neural networks, in this paper, we propose a stacked RNNs model which is based on word weight and residual networks. In this model, “word weight” part of network can consider the different contribution of words in a sentence, “residual learning” part of network can improve the performance of stacked recurrent neural networks. We evaluate our model on three experiment of two datasets, our proposed method is able to learn high hierarchical meaning of each word in a sentence and consider the weight of each word for text classification task, the experiment results show that our model outperforms the existing systems.

## 1.2 Organization of the thesis

The remainder of this thesis is structured as follows: Chapter II introduces some related works in the field of text classification and describes some commonly model in this area. In Chapter III, we describe the proposed method, which describe the structure and mechanism of Stacked Residual Bi-LSTM with Word Weight model. To evaluate the proposed method and testify its effectiveness,

Chapter IV evaluates our model on three text classification tasks. The thesis ends in Chapter V with conclusions on contributions of this work.

# Chapter 2

# Related Work

With the growth of Web information, textual data gains increasingly high importances. People largely access information from these online textual data sources rather than being limited to archaic paper sources like books, magazines etc. But the the biggest problem is that this information lacks organization which makes it difficult to search and manage. Text classification is one of the efficient techniques used for organizing such kind of digital data.

The aim of text classification task is to build systems which are able to automatically classify documents into labels based on their content. A supervised classification algorithm allows us to access the data labels during the training and testing steps. In this section, we provides a review of some related works of generic text classification methods which use machine learning algorithms for text classification task.

## 2.1 Traditional related methods

Most of traditional related methods focus on how to build a powerful classifier, these algorithms try to modify structure of classifier rather than extract more features in classification task. In feature extraction step, they often use some common methods, the most common method is word

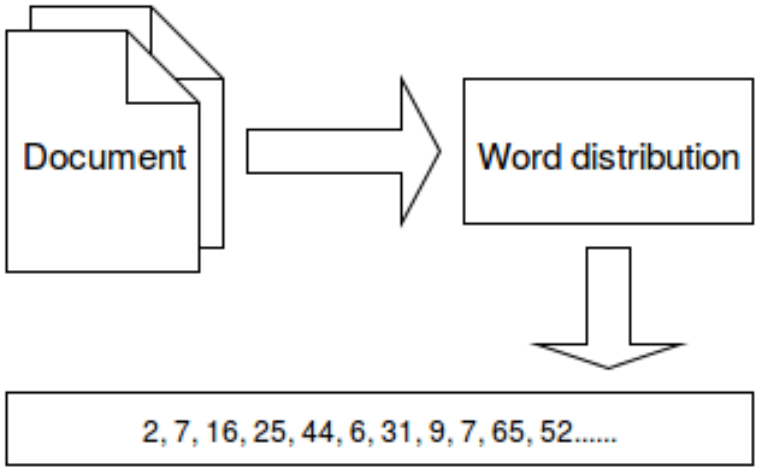



**Figure 2.1** An illustration of feature extraction.

frequency distribution which is the bag-of-word based method. In preparation for the feature extraction, the algorithm would look at all available texts and count the frequency of each word that occurs in these texts. Each word in the vector is then allocated a dimension in the feature vector. When feature vector of a text is extracted, each value in the feature vector will represent the number of times each word has occurred in the text.

### 2.1.1 Support Vector Machines based methods

Support Vector Machine (SVM) is one of the discriminative classification method. It is a supervised machine learning algorithm which can be used for text classification. The idea of SVM is based on structural risk minimization theory[20]. SVM tries to use linear or non-linear delineations between the different classes to partition the data space. The most important thing in such classifiers is to find the optimal boundaries between the different classes and use them for the classification task.

In the Figure 2.1, we can see this illustrated for the example data in 2D-space. The points which display in figure are labeled with two different color. SVM finds the hypeplane that can

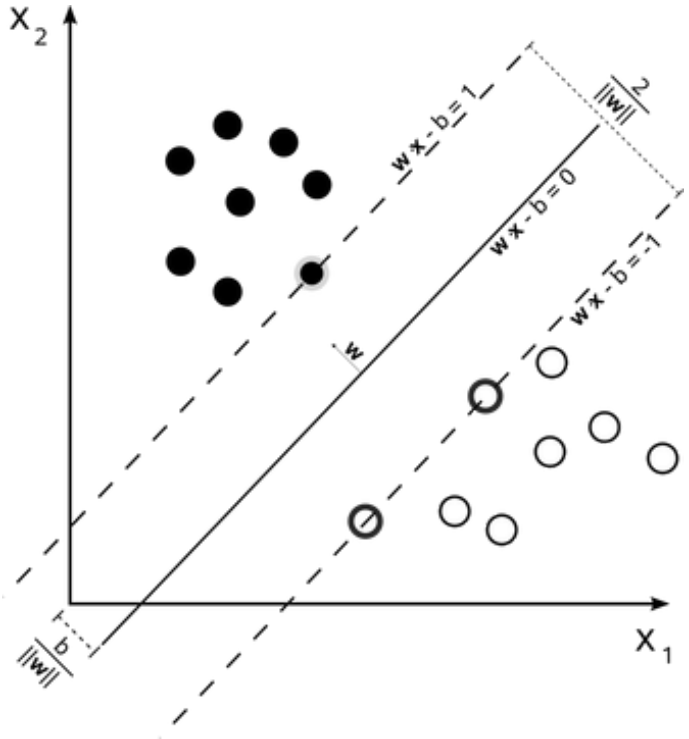



**Figure 2.2** An illustration of SVM for choosing the hypeplane that maximizes the margin in 2D-space.

maximizes the margin between the two labels. This hyperplane is given as follow:

$$\langle \vec{w} \cdot \vec{x} \rangle + b = \sum_{i=1}^{N} y_i \alpha_i \langle \vec{x}_i \cdot \vec{x} \rangle + b = 0 \tag{2.1}$$

where $\vec{x}_i = (x_{i1}, x_{i2}, ...x_{in})$ is a input vector which has n-dimensional , $y_i$ is output data, $\vec{w} = (w_1, w_2, ...w_n)$ is the weight vector which defines the hyperplane and the $\alpha + i$ terms are the Lagrangian multipliers.

Once the hyperplane is constructed in training data, the class of test data $\vec{x}_i$ can be determined as follow: if $\vec{w} \cdot \vec{x}_i + b \geq 0$ then it belongs to the label 0, if $\vec{w} \cdot \vec{x}_i + b \leq 0$ then it belongs to the label 1.

### 2.1.2 Naive Bayes based methods:

Naive Bayes classifier is a classical classification algorithm which is based on Bayes theorem with strong and naive independence assumptions. Bayesian classifier is the simplest with it's linear classifiers, but it is still very effective. Naive Bayes classifier assumes that the features of the input feature vector (word distribution) which were used in the classification are statistically independent[**?**]. In a text classification problem, the idea of Naive Bayes classifier is to classify text by using the posterior probability of the texts which belong to the different classes. The probability of a class $C$ is given by the posterior probability $P(C|D)$ given a training data $D$. Here $D$ refers to all of the text in the entire training data. It is given by $D = (d_1, d_2, ...d_n)$, where $d_i$ is the $i_{th}$ attribute (word) of document $D$.

In Bayes' rule, the posterior probability is as follow:

$$P(C = c_i|D) = \frac{P(D|C = c_i) \cdot P(C = c_i)}{P(D)} \tag{2.2}$$

$P(D)$ is equal for all classes, it can be disregarded and the equation can change to:

$$P(C = c_i|D) = P(D|C = c_i) \cdot P(C = c_i) \tag{2.3}$$

The document D belongs to the class C which would maximizes this probability:

$$\begin{aligned} C &= argmax P(D|C) \cdot P(C) \\ &= argmax P(d_1, d_2, .., d_n|C) \cdot P(C) \end{aligned} \tag{2.4}$$

Then, we assume that each words $d_i$ is conditional independent and the equation can change to:

$$\begin{aligned} C &= argmax P(d_1|C) \cdot P(d_2|C) \cdots P(d_n|C) \cdot P(C) \\ &= argmax \prod_{i=1}^{n} P(d_i|C) \cdot P(C) \end{aligned} \tag{2.5}$$

where $P(d_i|C)$ is the conditional probability that word $i$ belongs to class $C$ which can be calculated as below:

$$P(d_i|C) = \frac{count(d_i, C)}{\sum_i count(d_i, C)} \tag{2.6}$$

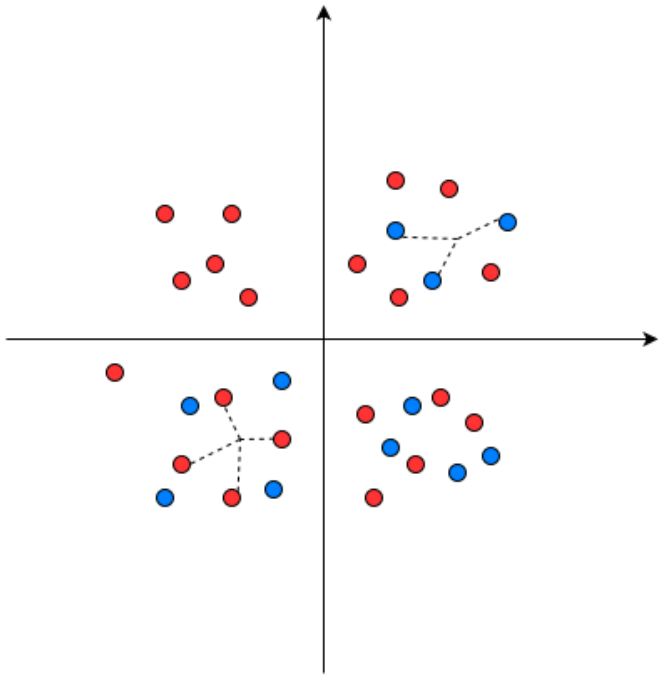


**Figure 2.3** An illustration of k-Nearest Neighbor.

### 2.1.3 k Nearest Neighbours based methods:

k Nearest Neighbours (kNN) classifier is one of the simplest geometrical classifiers[**?**]. The kNN method is outstanding with its simplicity and is widely used techniques for text classification. This method performs well even in handling the classification tasks with multi-label documents.

The basic idea of kNN is very simple, it is used to test the degree of similarity between documents and k training data and to store a certain amount of classification data. In the high dimension space, each document can be seen as a point in it which represents the document being classified. Each test data has it's own position and there are too many others points around this it, but only k number of nearest neighbours are considered. If these data belong to the same label then the new document also will be classified to that label. This method is an instant-based learning algorithm that classified objects based on closest feature space in the training set.

On the other hand, the accuracy of kNN is severely degraded by the data noisy and irrelevant

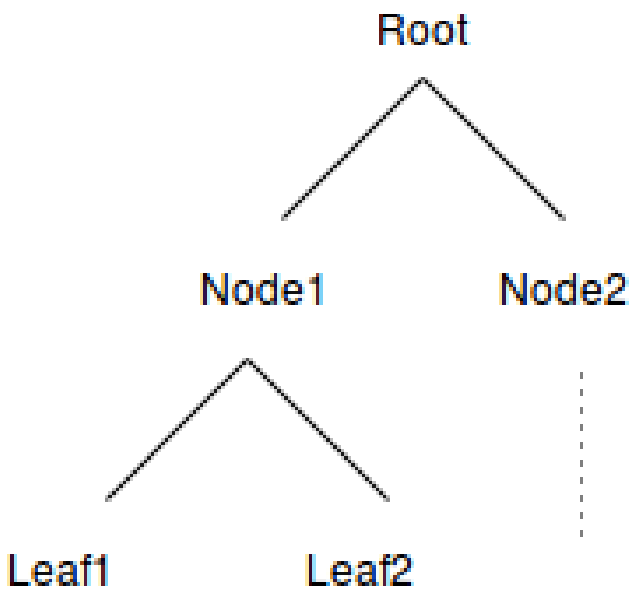



**Figure 2.4** An illustration of Decision trees.

features.

### 2.1.4 Decision Trees based methods:

Decision trees are hierarchical model for supervised learning with the underlying data space by using the different text features. The decision tree rebuilds the manual class of training data by constructing well-defined true or false conditional queries in the form of a tree structure. It is composed with the internal decision nodes and terminal leaves. The internal nodes are labeled by the features and the leaves are labeled by classes. The well-trained decision tree can classify a document by scanning it in the root node of the tree and judging through the conditional query structure until it reaches to leaf node easily, this leaf node would represents the classification of the document.

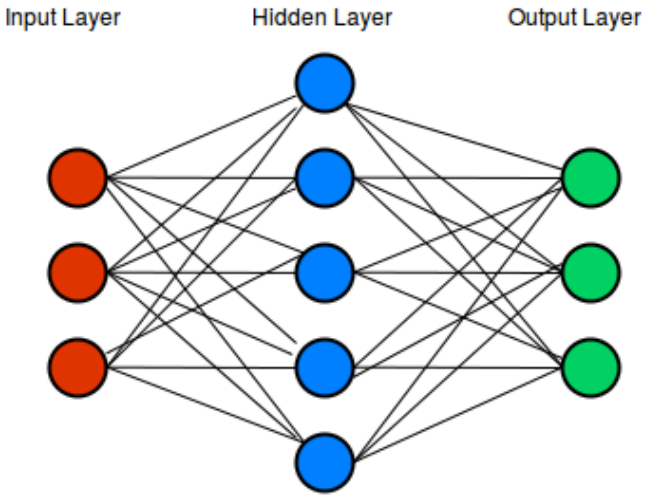



**Figure 2.5** An illustration of simple Neural Netowrks.

## 2.2 Neural Networks related methods

Neural Network is a massively parallel distributed network which is consist of a large number of simple artificial neurons, these neurons are able to learn knowledge from experience during the training and making it available for use. These artificial neurons are interconnected into group using a mathematical model for information processing. Neural Network is try to simulated human brain which may contains hundreds of thousands or even millions of parameters, the strong neural network can be seen as an "expert" in specific area such as text classification task.

Different from the traditional related method, Neural Networks related method in classification task is focus on how to extract high hierarchical features from data rather than build a powerful classifier. No matter how complicated the network is, it always used to extract feature vectors in previous layers, the classifier of last layer is often use Softmax.

There are too many different types of Neural Network methods have been applied to text classification tasks. Generally speaking, Neural Network for any classification task is as follow: a Neural Network is taken, data of the feature vector is fed to the inputs of the network and the label of data comes from the outputs.

With the growth depth of neural network, GPU become more and more important in deep

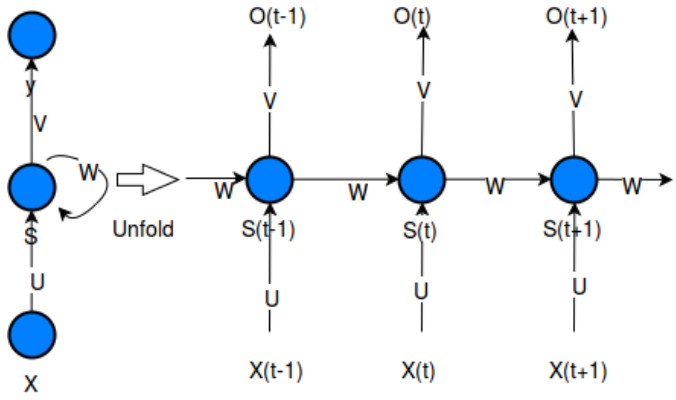



**Figure 2.6** An illustration of Recurrent Neural Network.

learning area because of it's high computation usage. Deep learning based methods have achieved high performance on text classification tasks. Too many researchers have devoted themselves into deep learning based research area for text classification in recent years. Le et al. (2014) [21] proposed the Paragraph Vector method for representations of sentences and documents. Zhang et al. (2015) [22] constructed a Character-level Convolutional Networks for doing text classification. Kim et al. (2014) [23] proposed a Convolutional Neural Networks for Sentence Classification which consider each words as n-gram to do embedding operation. Johnson et al. (2015) [24] proposed a scheme for embedding learning of small text regions which is based on the idea of two-view semi-supervised learning.

In recent work, RNN models can achieve high performance on text related tasks. Mikolov et al. (2010) [9] first used RNNs for sequence text task. Xiao et al. (2016) [25] propose Bidirectional Long Short-Term Memory with word embedding for text which contains richer syntactic and has strong intrinsic dependency between words and phrases. Tang et al. (2015) [26] introduced a model to learn vector-based document representation in a unified, bottom-up fashion for sentiment classification. Lai et al. (2015) [27] utilized a Recurrent Convolutional Neural Networks method which use Convolutional and Recurrent Networks to capture the feature of contextual information to learn word representations. Wang et al. (2016) [28] proposed an intuitive approach to learn distributed word representations with Bi-LSTM.

We also mentioned that there are some novel methods for related classification task. Wang et al. (2016) [29] introduced a method to expand short texts based on word embedding clustering and convolutional neural network. Liu et al. (2016) [30] used Multi-Task Learning methods to construct the model.

# Chapter 3

# A Stacked Residual RNN Model

In this section, we would introduce our text classification model called "Stacked Residual Recurrent Neural Network with word weight". Our proposed model can extract high hierarchical features of sentence and consider the weight of each word for text classification task.

Our model assumes that each word in a sentence doesn't have the same importance, which means the output of the previous LSTM will not be the input of the next LSTM directly in stacked networks.

In this model, it can consider the weight of the output of each word during the training, and inspired by the architecture of ResNets, we combine the idea of ResNets into our model for the sake of gradient vanish problem when network is very deep.

As we can see from Figure 3.1, which gives an illustration of our proposed model. The left part of the model is called **weight part**. This part takes responsibility for training the weight of input of each word. After that, the output of weight part would calculate the element-wise product with the output of the right part. Utilizing the idea of Residual Networks, in the right part of our model, we can see the input of previous layer can add with the input of next layer directly, this is called short connections (The input and output are of the same dimensions). The right part of the model is called **residual part**.

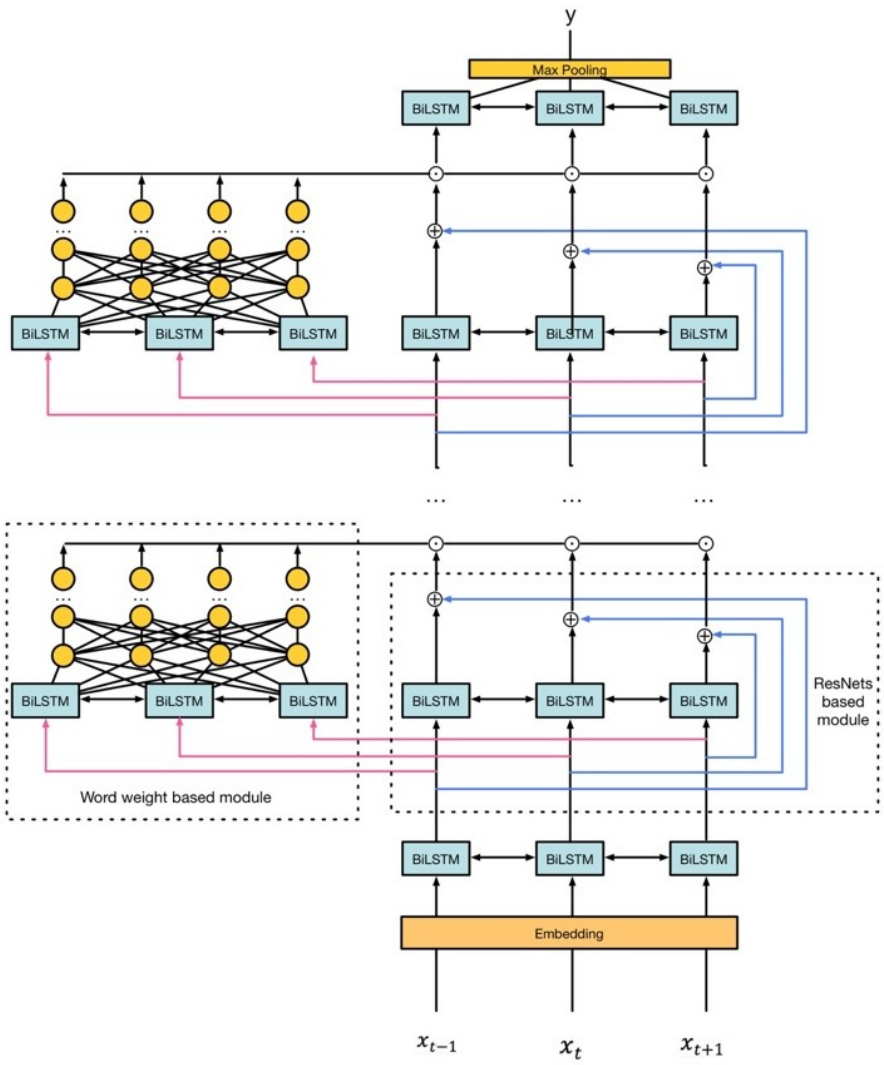



**Figure 3.1** The instance of Stacked Residual Bi-LSTM with Word Weight Networks. The left line part is the word weight training module and the right part is the ResNets based module.

## 3.1 Word weight based module

Word weight part module has ability to learn the weight of each word. We believe that the word weight is very important in text categorization task, the label of the sentence is often determined by several key words. We focus on constructing this module by using fully connected layers and Bidirectional LSTM (Bi-LSTM)[31] which is the variants of LSTM networks. The standard LSTM

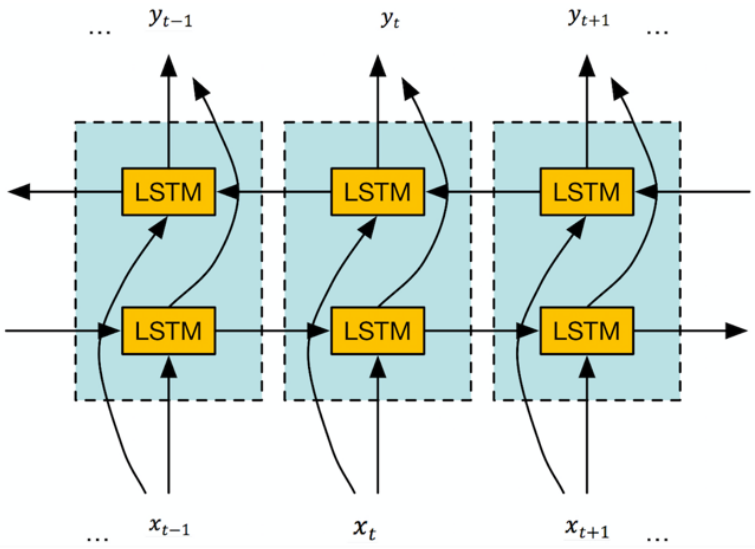



**Figure 3.2** An illustration of Bidirectional LSTM network.

is updated as follows:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{3.1}$$

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{3.2}$$

$$\tilde{c}_t = tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{3.3}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{3.4}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{3.5}$$

$$h_t(LSTM) = o_t \odot tanh(c_t) \tag{3.6}$$

where $x_t$ are the input of each time step t, $W_j$, $U_j$ are the weight matrices and $b_j$ are the bias vectors, for $j \in \{i, f, c, o\}$. $\sigma$ denotes the sigmoid activation function and $\odot$ denotes element-wise multiplication. The forget gate controls how much of the previous state is going to be thrown away, the input gate controls how much of newly state will be updated, and the output gate controls how much of the internal memory state will be output.

The Bi-LSTM contains not only the forward $\overrightarrow{LSTM}$ which read the word from the beginning of sentence to the end of sentence but also the backward $\overleftarrow{LSTM}$ which read the word from the end

sentence to the beginning of sentence:

$$\overrightarrow{h_t} = \overrightarrow{h_t(LSTM)} \tag{3.7}$$

$$\overleftarrow{h_t} = \overleftarrow{h_t(LSTM)} \tag{3.8}$$

$$h_{t,Bi-LSTM_W} = [\overrightarrow{h_t}, \overleftarrow{h_t}] \tag{3.9}$$

where $h_{t,Bi-LSTM_W}$ is the hidden state of the Bi-LSTM in word weight module which combines the forward and backward hidden states at each time step. Conventional standard LSTMs only utilize previous context with no exploitation of future context, Bi-LSTMs utilize both the previous and future context.

The output of Bi-LSTM will be trained in fully connected layers as the word weight:

$$a_{1,t} = h_{t,Bi-LSTM_W} \tag{3.10}$$

$$z_{n,t} = W_{n-1}a_{n-1,t} + b_{n-1} \tag{3.11}$$

$$a_{n,t} = ReLU(z_{n,t}) \tag{3.12}$$

$$\tilde{W}_t = a_{n,t} \tag{3.13}$$

$$O_t = h_{t,Bi-LSTM_R} \odot \tilde{W}_t \tag{3.14}$$

where $W_{n-1}$ are the weight matrices of fully connected layers, $b_{n-1}$ are the bias vectors, $a_{1,t}$ are the values from input layer of fully connected layers, $a_{n,t}$ are the activation of layer n, $z_{n,t}$ are the total weighted sum of layer n, ReLU[32] is the activation function, $\odot$ is element-wise multiplication, $\tilde{W}_t$ is the word weight of each time step t, $h_{t,Bi-LSTM_R}$ is the hidden state of the Bi-LSTM in right module and $O_t$ are the new hidden state of the Bi-LSTM.

## 3.2 Residual Networks based module

The right part in Figure 3.1 shows the residual part of the model. As we can see, at each layer, the input of Bi-LSTM and the output of Bi-LSTM can be summed directly. The notion of Residual

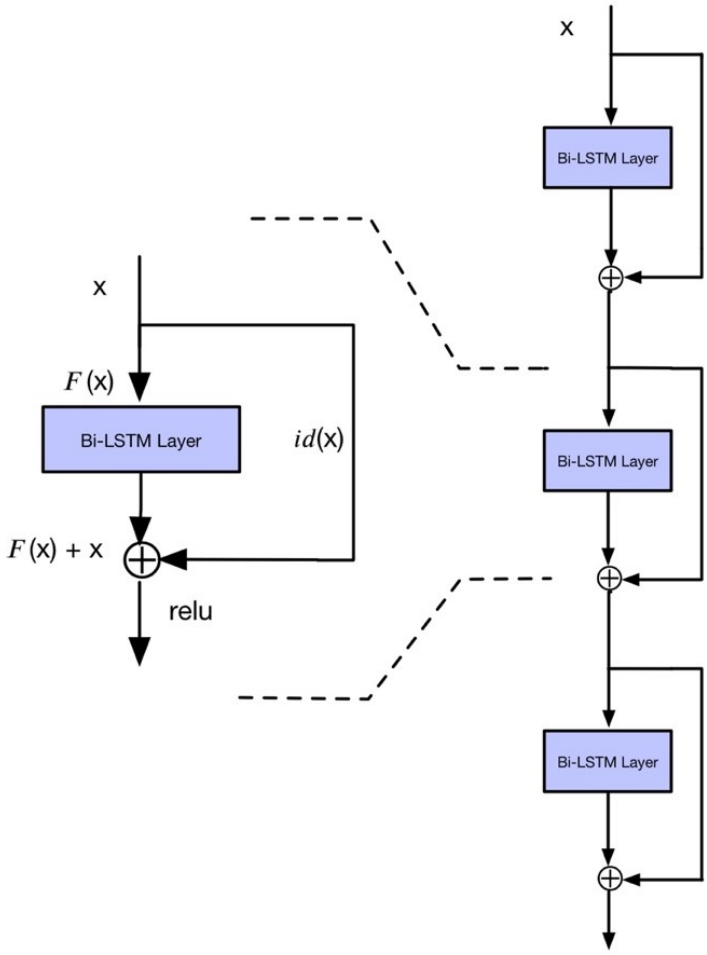



**Figure 3.3** The left image is a residual block in Residual Networks. The right image is an illustration of Residual Networks.

Networks (ResNets) was first introduced by [19] in image recognition area. The main idea of Residual Networks is to connect some of the layers with shortcuts, which can avoid vanishing gradients and exploding gradients problems, these problems may happen in very deep networks. With the increasing depth of networks, ResNets can improves the accuracy of deep networks.

The shortcut connections have the ability to explicitly let these layers fit a residual mapping

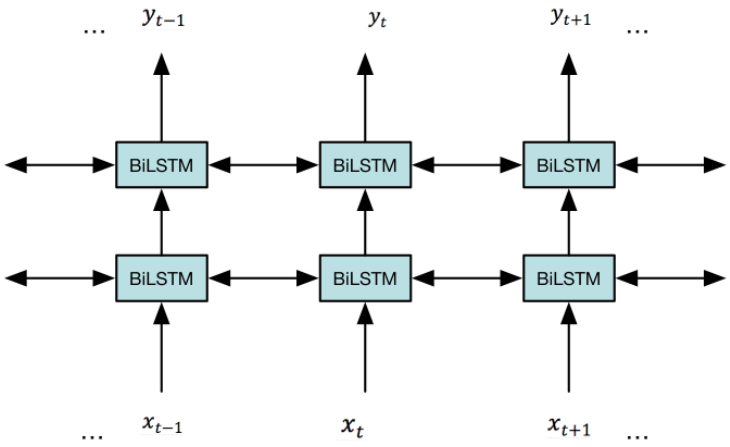



**Figure 3.4** Stacked structure of Bidirectional LSTM network.

with the help of identity transformation. The residual block defined as:

$$\mathbb{F}(x_{i-1,t}) = O_t \tag{3.15}$$

$$x_{i,t} = ReLU(\mathbb{F}(x_{i-1,t}) + id(x_{i-1,t})) \tag{3.16}$$

where $\mathbb{F}(\cdot)$ function represents the Bi-LSTM transformation from $x_{i-1,t}$ layer to $x_{i,t}$ at each time step t, $id(\cdot)$ is an identity mapping function. $ReLU$ is the activation function for output of residual block.

Although the derivation of Residual Networks is from image recognition area, inspired by its special architecture, we introduce it in our Stacked Bidirectional LSTM when the layers of Stacked Bi-LSTM are deep. The gradients and features which were learned in lower layers can passe through by the identity transformations $id(\cdot)$ in Residual Networks.

## 3.3 Stacked Residual Bi-LSTM with Word Weight model

Then, we extend our model to stacked one. Stacked based Bi-LSTM is the vertical multi-layer structure, the output of the lower layer will be the input of the upper layer. By using the stacked based structure, it is possible to achieve different levels of deep abstraction. There are some re-

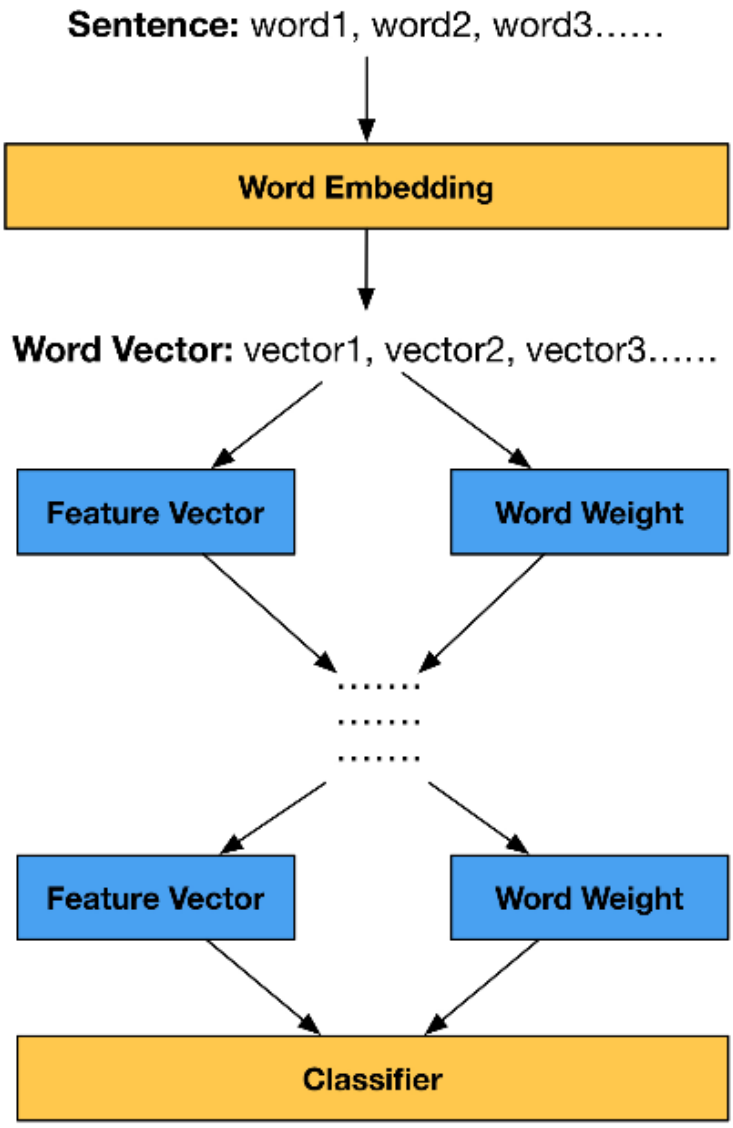



**Figure 3.5** Work flow of Stacked Residual Bi-LSTM with Word Weight Networks.

searches show that the deep hierarchical LSTM based model can be more efficient in representing some functions than a shallow one [33][34].

Finally, the max-pooling of output of Bi-LSTM can be used as the representation of the sentence which can be utilized as the features for text classification.

The target of the model is to predict label $\hat{y}_j^{(i)}$ for each sentence. We apply cross entropy error

function. We train the model over the training examples by minimizing loss of the predicted:

$$L(w) = \sum_{i=1}^{m} \sum_{k=1}^{K} 1\{y^{(i)} = k\} log(\hat{y}_j^{(i)}) \tag{3.17}$$

where $1\cdot$ is indicator function so that 1true=1, and 1false=0. m is the number of training examples. $y^{(i)} \in \{1, 2, \ldots K\}$ is true label of each sentence and K is the number of possible labels. $\hat{y}_j^{(i)} \in [0, 1]$ is estimated probabilities of each sentence of each label.

We use Adam[35] stochastic gradient descent optimizer to update the parameters.

# Chapter 4

# Experimental Setup

## 4.1 Datasets

To show the effectiveness of our model, we choose three different text classification tasks to evaluate our Stacked Residual with Word Weight architectures.

### 4.1.1 Sentiment Classification

**• SST-1**

The movie reviews consist of 11855 movie reviews with five labels: very negative, negative, neutral, positive, and very positive in Stanford Sentiment Treebank[11]. The dataset is spited into train (8544), dev (1101), and test (2210) for the fine-grained classification task.

**• SST-2**

The movie reviews with binary labels by removing neural labels from the Stanford Sentiment Treebank. The dataset is spited into train (6920), dev (872), and test (1821) for the binary classification task.

**Table 4.1** The statistical detail of three datasets in our evaluation.

| Dataset | Class | Train Size | Valid Size | Test Size | Average Length | Max Length | Vocabulary Size |
|---|---|---|---|---|---|---|---|
| SST-1 | 5 | 8544 | 1101 | 2210 | 19.1 | 56 | 19.5k |
| SST-2 | 2 | 6920 | 872 | 1821 | 19.3 | 56 | 17.5k |
| TREC | 6 | 5452 | - | 500 | 9.9 | 37 | 8.9k |

### 4.1.2 Question type Classification

#### • TREC

We choose the TREC[4] which is a question type classification benchmark. TREC consists of 6 classes, including location, human, entity, abbreviation, description and numeric. The training dataset contains train (5452) and test (500) questions.

## 4.2 BaseLines

We compare our model with several models as follow:

• **SVM** SVM with unigram and bigram features.

• **NBOW** NBoW averages word vectors and applies a softmax classification layer.

• **Paragraph Vector** Paragraph Vector learns fixed-length feature representations from variable-length pieces of texts.

• **CNN-non-static** Convolutional Neural Network based model with fine-tuned word vectors[23].

• **CNN-multichannel** Convolutional Neural Network based model with multi-channels[23].

• **DCNN** Dynamic Convolutional Neural Network with dynamic k-max pooling[36].

• **RAE** Recursive autoencoder[11].

**Table 4.2** Some other hyperparameters settings among three datasets.

| Hyperparameters | SST-1/SST-2 | TREC |
|---|---|---|
| Memory dimension (Bi-LSTM) | 150 | 300 |
| Stacked layers (Residual part) | 10 | 5 |
| FC hidden layers (Each weight part) | 5 | 5 |
| FC hidden layers (Each weight part) | 50 | 50 |

- **MV-RNN** Matrix-Vector Recursive Neural Network[11].
- **RNTN** Tensor based Recursive Neural Tensor Network[11].
- **DRNN** Multi-layer stacked Recursive Neural Network[37].
- **MTL-RNN** A multi-task learning framework to jointly learn across multiple related tasks[30].
- **Tree-LSTM** A LSTMs to tree-structured network topologies[38].
- **C-LSTM** Unified Model which utilizes CNN and LSTM[39].

## 4.3 Hyperparameters and Training

In our experiments, we initialize word embeddings with the publicly available 300-dimensional word vectors. The vectors are pre-trained with word2vec on Google News Dataset which contains about 100B words. We also initialize the vector with the uniform distribution [-0.25, 0.25] for words which are not in word2vec vectors.

We train our model with Adam stochastic gradient descent optimizer with a learning rate of 0.001 and we use a mini-batch size of 50. The parameters are initialized from uniform distribution in [-0.1, 0.1]. The parameters were regularized with L2 regularization with factor of $10^{-4}$. We also

apply dropout[40] with a probability of 0.5 on both weight part and residual part of model during the training to prevent overfitting.

Other hyperparameters settings are shown in Table 4.2, for SST, the LSTM dimension is 150, so the combination of forward and backward in Bi-LSTM is 300 dimensions for output vector. The same as TREC.

## 4.4 Results and Analysis

### 4.4.1 Text classification

The experimental results are showed in Table 4.3. We compare our model with a variety of models, the Stacked Residual with Word Weight structure model has high performance on text classification task without any additional feature engineering.

Fore SST datasets, our proposed method outperforms existing models and achieves state-of-art prediction accuracy on both fine-grained classification task and binary classification task of Stanford Sentiment Treebank. In particular, our model obtains 52.7% classification accuracy on fine-grained classification task which is a very substantial improvement. For TREC, our result is close to the best result. Although we did not beat the state-of-the-art one, comparing with Stanford Sentiment Treebank, we find that not only the average sentence's length of SST is longer than TREC, but also the Semantic complexity of SST is much more complicated than TREC. Through the analysis, we find that our model is applicable to the semantics of complex sentence which can learn more features from sentences. The Stacked Residual with Word Weight structure can learn the weight of different words in the sentence during the training, it is very useful for sentence representation that can increase the prediction accuracy.

In summary, the results mean that our model can extract more information and learn high hierarchical features from the text than others methods on dataset which has complex semantics.

**Table 4.3** Classification accuracy of our method compared with other models on three datasets.

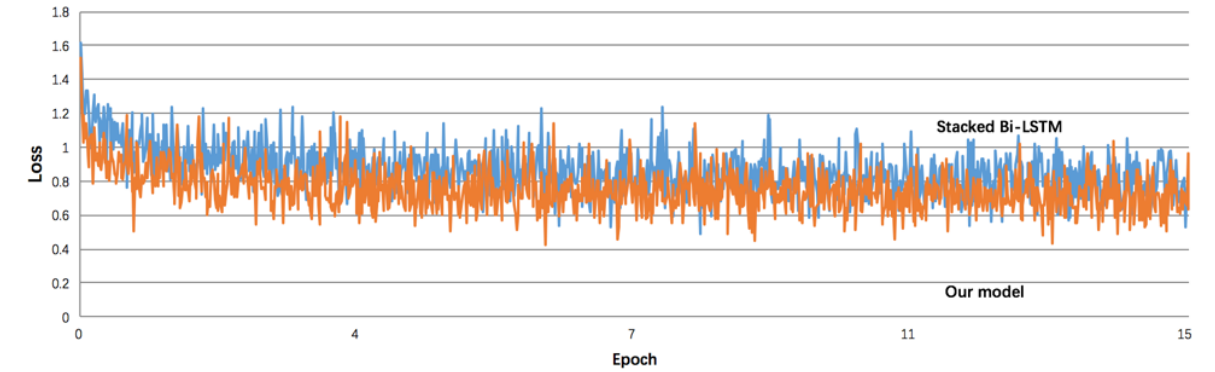
| | Methods | SST-1 | SST-2 | TREC |
|---|---|---|---|---|
| Socher et al., 2013(a)<br>Silva et al., 2011(b) | SVM | 40.7%(a) | 79.4%(a) | 95%(b) |
| Kalchbrenner et al., 2014 | NBOW | 42.4% | 80.5% | - |
| Le and Mikolov, 2014(a)<br>Zhao et al., 2015(b) | Paragraph Vector | 48.7%(a) | 87.8%(a) | 91.8%(b) |
| Kim, 2014 | CNN-non-static | 48.0% | 87.2% | 93.6% |
| Kim, 2014 | CNN-multichannel | 47.4% | 88.1% | 92.2% |
| Kalchbrenner et al., 2014 | DCNN | 48.5% | 86.8% | 93.0% |
| Socher et al., 2013 | RAE | 43.2% | 82.4% | - |
| Socher et al., 2013 | MV-RNN | 44.4% | 82.9% | - |
| Socher et al., 2013 | RNTN | 45.7% | 85.4% | - |
| Irsoy aet al., 2014 | DRNN | 49.8% | 86.6% | - |
| Liu et al.,2016 | MTL-RNN | 49.6% | 87.9% | - |
| Tai et al., 2015 | Tree-LSTM | 51.0% | 88.0% | - |
| Zhou et al.,2015 | C-LSTM | 49.2% | 87.8% | 94.6% |
| **This paper** | **SRW-RNN** | **52.7%** | **88.2%** | **94.7%** |

### 4.4.2 Model Analysis

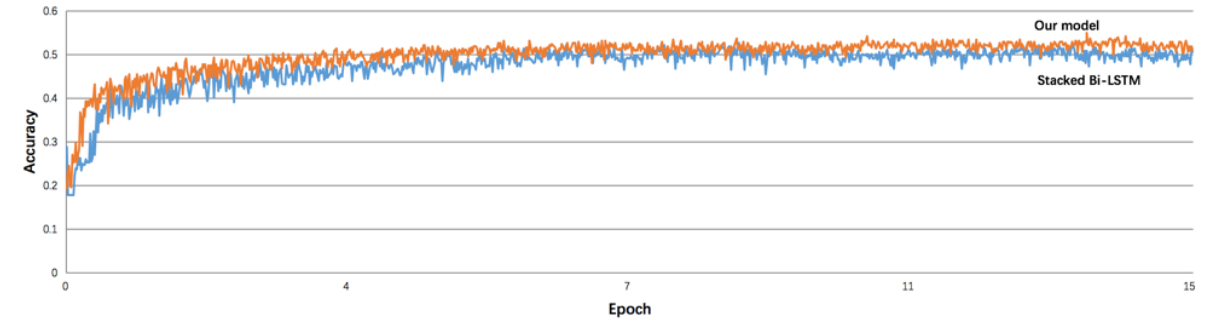
Benefiting from the word weight structure, our model can learn the importance of different words very well, each word has its own weight which is the high potential features in a sentence. Most of the time, the label of each sentence is determined by several key words, and word weight part can capture these key words easily.

As we can see from Figure 4.1 (a), comparing with the standard stacked Bi-LSTM without residual part, during the training step, the convergence speed of our model is faster. The inputs of a lower layer in stacked Bi-LSTM are made available to a node in a higher layer because of the shortcut connections which can lead the network easy trained.

The Figure 4.1 (b) shows the test accuracy between two models. this figure indicates that the residual structure has ability to achieve high accuracy and the gradients can easily back propagate through them, which results in a faster converging.




(a)




(b)

**Figure 4.1** The (a) shows the batch training loss on SST-1 with our method and standard stacked Bi-LSTM. The (b) shows the test accuracy on SST-1 compared with two methods.

# Chapter 5

# Conclusions

In this paper, we introduce a novel text classification model called Stacked Residual Recurrent Neural Network with word weight. This model is able to extract more features and learn high hierarchical meaning of each word in a sentence. It also can identify the importance of different words due to its word weight structure. The residual structure makes model more expressive when stacked layers are deep. Experimental results show that our model can achieve high performance on text classification task than any other methods. This suggests our model can capture more potential features in sentences.

# Bibliography

[1] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011, pp. 142–150.

[2] M. Ghiassi, J. Skinner, and D. Zimbra, "Twitter brand sentiment analysis: A hybrid system using n-gram analysis and dynamic artificial neural network," *Expert Systems with applications*, vol. 40, no. 16, pp. 6266–6282, 2013.

[3] D. Zhang and W. S. Lee, "Question classification using support vector machines," in *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM, 2003, pp. 26–32.

[4] X. Li and D. Roth, "Learning question classifiers," in *Proceedings of the 19th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 2002, pp. 1–7.

[5] S. Wang and C. D. Manning, "Baselines and bigrams: Simple, good sentiment and topic classification," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*. Association for Computational Linguistics, 2012, pp. 90–94.

[6] D. Quercia, H. Askham, and J. Crowcroft, "Tweetlda: supervised topic classification and link prediction in twitter," in *Proceedings of the 4th Annual ACM Web Science Conference*. ACM, 2012, pp. 247–250.

[7] T. Joachims, "A statistical learning model of text classification for svms," in *Learning to Classify Text Using Support Vector Machines*. Springer, 2002, pp. 45–74.

[8] ——, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*. Springer, 1998, pp. 137–142.

[9] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model." in *Interspeech*, vol. 2, 2010, p. 3.

[10] R. Johnson and T. Zhang, "Effective use of word order for text categorization with convolutional neural networks," *arXiv preprint arXiv:1412.1058*, 2014.

[11] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, vol. 1631. Citeseer, 2013, p. 1642.

[12] J. L. Elman, "Finding structure in time," *Cognitive science*, vol. 14, no. 2, pp. 179–211, 1990.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] X. Li and X. Wu, "Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 4520–4524.

[15] T. M. Breuel, A. Ul-Hasan, M. A. Al-Azawi, and F. Shafait, "High-performance ocr for printed english and fraktur using lstm networks," in *2013 12th International Conference on Document Analysis and Recognition*. IEEE, 2013, pp. 683–687.

[16] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[17] J. Y. Lee and F. Dernoncourt, "Sequential short-text classification with recurrent and convolutional neural networks," *arXiv preprint arXiv:1603.03827*, 2016.

[18] J. Zhou and W. Xu, "End-to-end learning of semantic role labeling using recurrent neural networks," in *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2015.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *arXiv preprint arXiv:1512.03385*, 2015.

[20] V. Vapnik, *The nature of statistical learning theory*. Springer Science & Business Media, 2013.

[21] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents." in *ICML*, vol. 14, 2014, pp. 1188–1196.

[22] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in Neural Information Processing Systems*, 2015, pp. 649–657.

[23] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[24] R. Johnson and T. Zhang, "Semi-supervised convolutional neural networks for text categorization via region embedding," in *Advances in neural information processing systems*, 2015, pp. 919–927.

[25] Z. Xiao and P. Liang, "Chinese sentiment analysis using bidirectional lstm with word embedding," in *International Conference on Cloud Computing and Security*. Springer, 2016, pp. 601–610.

[26] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1422–1432.

[27] S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification." in *AAAI*, 2015, pp. 2267–2273.

[28] P. Wang, Y. Qian, F. K. Soong, L. He, and H. Zhao, "Learning distributed word representations for bidirectional lstm recurrent neural network," in *Proc. of ICASSP*, 2016.

[29] P. Wang, B. Xu, J. Xu, G. Tian, C.-L. Liu, and H. Hao, "Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification," *Neurocomputing*, vol. 174, pp. 806–814, 2016.

[30] P. Liu, X. Qiu, and X. Huang, "Recurrent neural network for text classification with multi-task learning," *arXiv preprint arXiv:1605.05101*, 2016.

[31] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[32] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 807–814.

[33] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[34] J. Chung, C. Gülçehre, K. Cho, and Y. Bengio, "Gated feedback recurrent neural networks," *CoRR, abs/1502.02367*, 2015.

[35] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[36] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," *arXiv preprint arXiv:1404.2188*, 2014.

[37] O. Irsoy and C. Cardie, "Deep recursive neural networks for compositionality in language," in *Advances in Neural Information Processing Systems*, 2014, pp. 2096–2104.

[38] K. S. Tai, R. Socher, and C. D. Manning, "Improved semantic representations from tree-structured long short-term memory networks," *arXiv preprint arXiv:1503.00075*, 2015.

[39] C. Zhou, C. Sun, Z. Liu, and F. Lau, "A c-lstm neural network for text classification," *arXiv preprint arXiv:1511.08630*, 2015.

[40] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.