2018年 08月

석사학위논문

Weibo Sentiment Analysis Based on Word2vec and CNN

조 선 대 학 교 대 학 원

컴퓨터공학과

Yujie Xie

Word2vec과 CNN기반의 웨이보 감정분석

Weibo Sentiment Analysis
Based on Word2vec and CNN

2018년 8월 24일

조 선 대 학 교 대 학 원

컴퓨터공학과

Yujie Xie

Weibo Sentiment Analysis Based on Word2vec and CNN

지도교수 김 판 구

이 논문을 공학석사학위신청 논문으로 제출함.

2018년 4월

조 선 대 학 교 대 학 원

컴퓨터공학과

Yujie Xie

YuJie Xie의 석사학위 논문을 인준함

위원장 조선대학교 교수 정 일 용 印

위 원 조선대학교 교수 김 판 구 印

위 원 조선대학교 교수 양 희 덕 印

2018 년 5 월

조 선 대 학 교 대 학 원

Table of Contents

FIGURE

TABLE

ABSTRACT

Weibo Sentiment Analysis Based on Word2vec and CNN

Yujie Xie
Advisor : Prof. PanKoo Kim, Ph.D
Department of Computer Engineering
Graduate School of Chosun University

With the rapid development of Internet, the enthusiasm of netizens is increasing. Micro-blog has become an important platform for netizens to express their emotions[8]. The social network of micro-blog reflects the emotional tendency of netizens to a great extent. How to quickly find hidden emotional information from micro-blog has drawed most of researchers attentions. As social networks are developing, sentiment analysis on social media such as Facebook, Twitter and Weibo becomes a new trend in recent years. Most of different methods have been proposed for sentiment analysis, including traditional methods (SVM and NB) and deep learning methods (RNN and CNN)[3]. In addition, the latter always outperform the former. However, many existing methods only focus on local text information and ignore the user personality and content characteristics.

The traditional sentiment analysis takes a lot of time to extract the characteristics of the data, and it needs to combine the relevant rules to get better results. In the big data era, the amount of data is increasing which also increases the difficulty of feature extraction . In this paper, I will use deep learning method called CNN to determine the emotional information in micro-blog and extract the feature of the word vector[5]. A framework called Word2vec + Convolutional Neural Network (CNN) is proposed to complete Weibo's sentiment analysis. Firstly, I use the word2vec proposed by the website (https://spaces.ac.cn/archives/4304) to compute vector

representations of words, which will be the input for the CNN. The purpose of using word2vec is to gain the vector representation of words and reflect the distance of words[2]. That will lead to initialize the parameters at a good point of CNN, which can efficiently improve the performance of the nets in this problem. Secondly, I design a suitable CNN architecture for the sentiment analysis task. I use 2 convolutional layers and 3 full-connected layers in this architecture. By the experiment verification, this is a valuable model applied using word2vec and CNN to analyze sentences' sentiment which gets a perfect results. I also use the Rectified Linear Unit (ReLU), Normalization and Dropout technology to improve the accuracy and generalizability of our model[7]. I test my framework in a public dataset which is the corpus of Weibo's comments that includes 2 labels: negative and positive. My network achieves test accuracy of 90.20% in this dataset, which is a better performance than some other traditional neural network models and other multiple layers CNN based on my CNN structure[1].

Ⅰ. Introduction

A. Motivation

With the coming of Internet era, micro-blog has become the main platform for netizens to express their emotions. Micro-blog's sentiment analysis is becoming the focus research of Natural Language Processing. Traditional sentiment analysis constructs emotional dictionary by extracting text features and gets text features based on this rule, Which way not only takes a lot of time but also needs manual participation in specific implementation. Therefore, traditional sentiment analysis technology can only be applied on small scale of data[16]. With the advent of the era of big data, traditional sentiment analysis method is obviously unable to satisfy users' needs. Deep learning algorithm can determine data characteristics in massive dataset without manual participation, which receive wide attentions from most of researchers[3]. The analysis of emotions is probably divided into three stages including data preprocessing, data feature extraction and characteristic learning[17]. In feature extraction stage, it will waste a lot of time to extract features by artificial data feature extraction. In the stage of feature learning, traditional sentiment analysis learn the characteristics of emotional information by shallow learning algorithm. However, the computational complexity of the shallow learning method is relatively low. Shallow learning method is not good enough for logical texts. But deep learning algorithm can improve the accuracy of emotional analysis[4]. In this paper, we use convolution neural network learning algorithm to extract different dimensional emotional information contained in micro-blog comments which are used to predict the emotional tendencies of micro-blog comments.

B. Outline

The outline of the thesis is organized as follows:

Chapter II introduces the Chinese Microblog-Weibo and analysis the problem of extracting characters features, and then related concepts of feature extraction and word segmentation are proposed. Some of basic knowledge of NLP (word2vec) are also writen in this chapter.

Chapter III A CNN structure of 2 convolutional layers and 3 full-connected layers is proposed in this section. Some processing experiments are carried out to compare the proposed CNN performance. Different batch-size models and multiple layer CNN models are tried to do trade-off of my CNN model.

Chapter IV mainly describes the experimental evolution and perfect results are showed in the experiment.

At last, conclusions are given.

Ⅱ. Background Concepts

Sentiment analysis of online application user generating content is an challenging and tough work for many NLP researchers. In the context of social media, so many difficulties are needed to be solved such as large amounts of data available and short messages on social networks etc. Therefore, sentiment analysis is also a trend research which attracts more attentions[11]. A lot of researchers developed different approaches to solve sentiment analysis problem by natural language processing and information retrieval, most of which in this area use bag of words representations. As the development of Internet, we can receive huge amounts of data in several web sites such as social media or online-shopping sites[8]. Therefore, many researchers start to use this available data to analyse sentiment. There are two researchers analysing the sentiment of multiple aspects of restaurants like food or atmosphere to deal with larger reviews in more detail[9]. The deep learning framework is very useful in both supervised and unsupervised learning areas[24]. More and more researchers are trying to solve the challenging sentiment analysis task by using deep learning algorithms because of the recent achievement of deep learning[3].

The task of natural language processing includes sentiment analysis, paraphrase detection, entailment recognition, summarization, discourse analysis, machine translation, grounded language learning and image retrieval[19]. Sentiment analysis is one of the most vital important parts of understanding how researchers dealing with text information. Wide application of sentiment analysis are used in the industry. For instance, a famous Chinese internet company, Baidu provides a new server to help people select better products.This service will show people the

sentiment analysis of the product by using the data of people' reviews in some online shopping sites. This service will be more helpful if we can reduce the error and complexity of sentiment analysis[27]. In the past decades, the approach to sentiment analysis is using traditional classification models such as Naive Bayes (NB), Naive Bayes with bag of bigram features (BiNB) and support vector machines (SVMs)[18].

A. Weibo Introduction

The rapid rise of the Internet has enabled us to enter the era of massive information. Therefore, how to get the truly valuable data from the huge Internet dataset has become a major concern. "Weibo" is also the abbreviation of microblog. Users can set up personal communities by means of WEB, WAP and other ways. Updating information by about 140 words and sharing them in real time becomes the new trend for youngers. According to professional reports, by the first half year of 2013, Chinese microblog users increased from 308 million to 330 million, and Sina registered users reached 503 million[21]. Now, in 2018, Sina micro-blog has become the most effective and fast way for people to get the latest information.

Unlike traditional information, micro-blog, as a new short text message, is characterized by the rapid release of information and the rapid spread of information. For example, the content of the message you publish will spread to 10 million fans in a flash if you have 10 million fans. Micro-blog is originally and highly real-time, which its content can reflect the social status and people's thinking. At the same time, micro-blog also has the characteristics of disorderly content, non-standard description including all kinds of Network Vocabulary and

trend vocabulary. The acquisition of micro-blog information is entirely dependent on the concerns of the publisher so that autonomy and selectivity of its content is very strong. And the influence of publicity is very flexible, then it greatly increases the difficulty of obtaining high quality micro-blog content and processing it.

B. CNN

Convolutional Neural Networks (CNN) is a kind of feedforward neural network which is proposed by inspiration of the mechanism of Receptive Field that mainly refers to some properties of neurons in auditory system, proprioceptive system and visual system. In the visual nervous system, for example, the receptive field of a neuron refers to a specific area on the retina, and only the stimulation in this area can activate the neuron.

Each layer of CNN can output multiple feature maps of which each extracts an input feature through one convolution filter and each feature map consists of multiple neurons. Supposing that the shape of one feature map is m*n, and then it has m*n neurons. The kernel in convolution layer record the weight of last layer feature map and the convolution kernel of the current layer[24].

The convolution neural network improves the traditional neural network according to the local perception domain, the weight sharing and the space down-sampling by using the local-connected method.

The local sensing domain is a convolution neural network, and independent neurons respond only to the corresponding stimuli. There is no response to other neurons. Thus, the structural characteristics of input data and images can be extracted. The local feature information can

be integrated to obtain the global feature information, and then a complex network is formed after processing. The advantage of this local connection method is to reduce the number of parameters and improve the speed of training.

The distribution is more uniform in the natural image, which means that the local feature extraction is also suitable for the whole image. Any convolution filter plays the role of the whole receptive field to draw the feature map according to the local feature and realize the sharing of the weight.

The spatial sampling will extract the resolution of the first feature map, therefore, the space download makes the process to reduce the dimension of the image features, reduce the amount of the image data, and can remove some noise data. In the next process, if the maximum aggregation is used, the value of the element in a convolution layer is determined by the maximum element value under the lower sampling process. After the reel operation, the feature graph is obtained, and the extracted feature data is used for classification training, and then more computation is generated. Therefore, in the operation, the dimension reduction operation of the image is needed. If down sampling is performed in continuous range in the selected feature map, the feature will remain unchanged. Under this background, the images can be correctly extracted from the same feature map to carry out classification and recognition after editing and downloading.

a.Convolutional Layer

The forward propagation of convolution layer is mainly to complete the extraction process of data image feature map. Each clipping has a convolution sum of the same size. The output of the volume layer is obtained by calculating the characteristic graph and the bases. In the previous feature map, each data matrix and the convolution kernel are

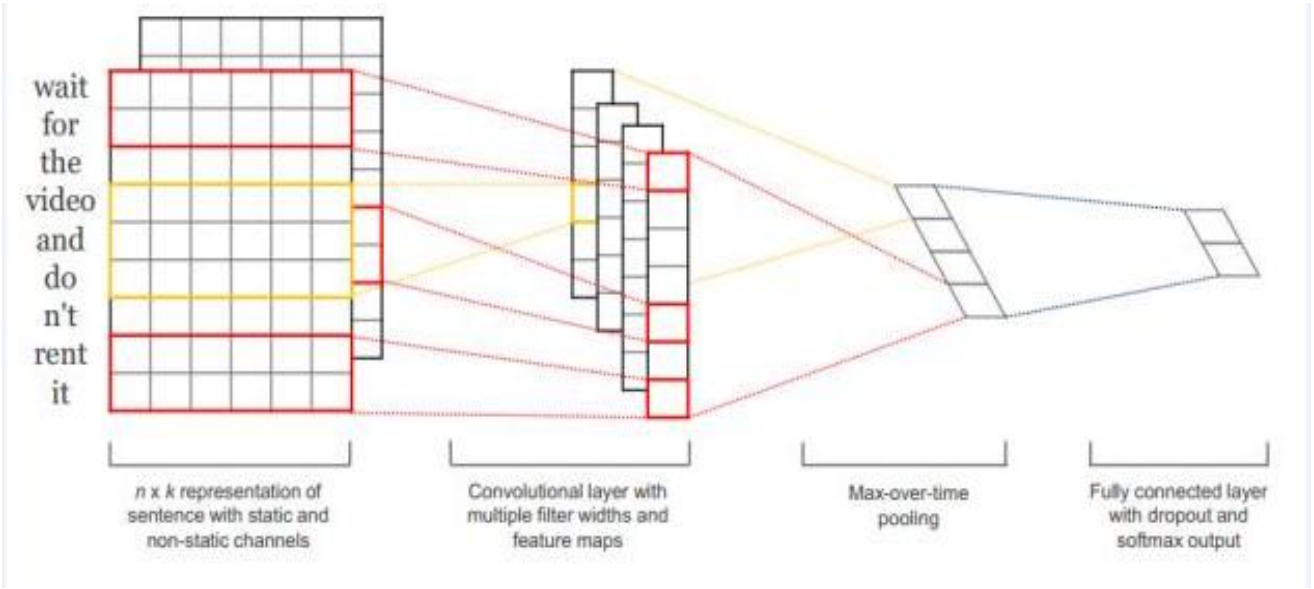
selected for the dot multiplication operation. Then the value of the corresponding position is output by the activation function. The number of the features in the convolution layer has long been determined. The characteristic graph is the output of a single layer and the size of the convolution. After that, the size of the feature map output is obtained.

The forward propagation of the down sampling level refers to the dimension reduction of the upper level output feature map. The feature graph of the latter layer is consistent with the previous layer, and its sampling process makes the aggregation operation for each different domain in the later input feature graph. the feature graph is reduced to the 1/n of the original feature graph.

In the final analysis, the simplest convolution neural network plays a role as a classifier.

The convolution layer is responsible for extracting features, while the sampling layer is responsible for feature selection, and the full link layer is responsible for classification.

CNN is one of the two most common deep learning models in Natural Language Processing with RNN. The picture below shows a typical network structure that uses the CNN model in the NLP task. Generally speaking, the input word or sentence is expressed in the way of Word Embedding so that the original one-dimensional text information input is converted into a two-dimensional input structure. Assuming that the input X contains K characters and the length of Word Embedding for each character is n, then the input is the two-dimensional vector of k*n.



typical network structure of Natural Language Processing CNN model

It can be seen here that, depending on the size of the K, the size of the input matrix of CNN is uncertain because that the length of the sentence in NLP is different. The convolution layer is essentially a feature extraction layer that can set a super parameter F to specify how many feature extractors (Filters). For a Filter, you can imagine a m*n size mobile window moving from the first word of the input matrix, in which m is the size of the window specified by Filter, and n is Word Embedding length. For a window at a certain time, the input value in this window is converted to a characteristic value through the nonlinear transformation of the neural network. As the window moves back, the corresponding eigenvalues of the Filter are constantly generated to form the eigen vector of the Filter. This is the process of decimation of the convolution layer. Each Filter operates so that different feature extractors are formed. The Pooling layer performs dimension reduction operations on the characteristics of Filter to form the final feature. In general, after connecting the Pooling layer, the whole joined layer neural network is connected to form the final classification process.

It can be seen that convolution and Pooling are the two most important steps in CNN. Next, we will focus on the common Pooling operation methods of CNN models in NLP.

b. Pooling Layer

MaxPooling Over Time is the most common sampling operation in the CNN model of NLP. It means that the value of a certain Filter is extracted to some eigenvalues, only the value of which the maximum score is retained as the retention value of the Pooling layer. The other eigenvalues are all abandoned. The maximum value only keeps the strongest ones of these features and abandons the other weak features.

There are several benefits in using the Max Pooling operation in CNN: firstly, this operation ensures the location and rotation invariance of the feature, because no matter where the strong feature appears, it can be brought out without considering its location. For the image processing, this position and rotation invariance is a good feature. But for NLP, this feature is not necessarily a good thing because that the location information of the feature is very important in many NLP applications. for example, the location of the subject is generally in the sentence head, and the object usually appears at the end of the sentence, and so on. These location information is sometimes very important for the classification task, but Max Pooling basically dropped the information.

Secondly, MaxPooling can reduce the number of model parameters and reduce the problem of model overfitting. Because that the array of 2D or 1D is often converted to a single value after the Pooling operation, hidden layer neurons will be reduced for the subsequent Convolution

layer or the Full-connected hidden layer.

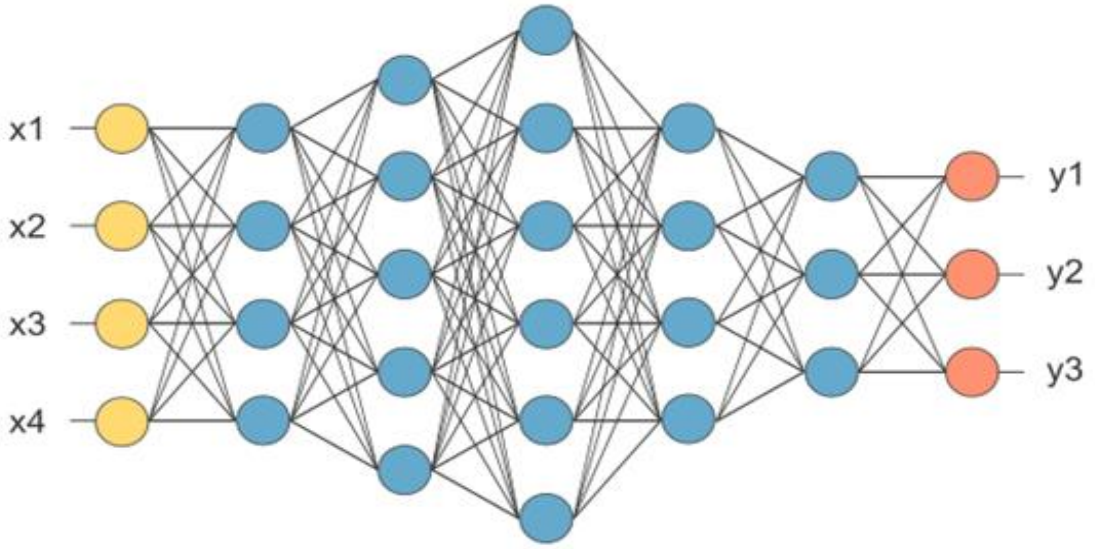
Furthermore, for NLP tasks, Max Pooling has an additional benefit, where the variable input X can be sorted into fixed length input. Because CNN will eventually connect the full join layer, and the number of neurons is predetermined. It is difficult to design network structure if the input is not fixed. The input X length of the CNN model is uncertain and each Filter is fixed to 1 values by Pooling operation. Then the Pooling layer have a number of neurons. Therefore the number of all join layer neurons can be fixed. It is also very important.

However, there are some noteworthy shortcomings in the CNN model taking MaxPooling Over Time: first of all, as described above, the location information of the feature is completely lost in this step. The convolution layer is actually reserved for the feature location information. But by taking the unique maximum value, now only maximum value at the Pooling layer but also the location information is not retained. The other obvious disadvantage is that some strong features will sometimes appear many times, such as common TF.IDF public. TF refers to the number of occurrences of a feature. The more the number of occurrences, the stronger the feature. Because that the Max Pooling has only one maximum value, it can only be seen once even if a certain feature appears many times. That is, the intensity information of the same feature is lost. These are the two typical shortcomings of Max Pooling Over Time[2].

c. Fully connected

Fully connected layers connect every neuron in one layer to every neuron in another layer. It is in principle the same as the traditional

multi-layer perceptron neural network. We can flatten our matrix into vector and feed it into a fully connected layer like neural network.



[Figure 2-1] Flattened as FC layer

In the above diagram, feature map matrix will be converted as vector (x1, x2, x3, …). With the fully connected layers, we combined these features together to create a model. Finally, we have an activation function such as softmax or sigmoid to classify the outputs as cat, dog, car, truck etc.

C. Word2vec

Word2vec is a two-layer neural net that processes text. Its input is a text corpus and its output is a set of vectors: feature vectors for words[4]. in that corpus. While Word2vec is not a deep neural network, it turns text into a numerical form that deep nets can understand. Deep learning4j implements a distributed form of Word2vec for Java and Scala, which works on Spark with GPUs[6].

Word2vec's applications extend beyond parsing sentences in the wild.

It can be applied just as well to genes, code, likes, playlists, social media graphs and other verbal or symbolic series in which patterns may be discerned[7].

The purpose and usefulness of Word2vec is to group the vectors of similar words together in vectorspace. That is, it detects similarities mathematically. Word2vec creates vectors that are distributed numerical representations of word features, features such as the context of individual words. It does so without human intervention[10].

Word2vec can make highly accurate guesses about a word's meaning based on past appearances if given enough data, usage and contexts. Those guesses can be used to establish a word's association with other words (e.g. "man" is to "boy" what "woman" is to "girl"), or cluster documents and classify them by topic. Those clusters can form the basis of search, sentiment analysis and recommendations in such diverse fields as scientific research, legal discovery, e-commerce and customer relationship management.

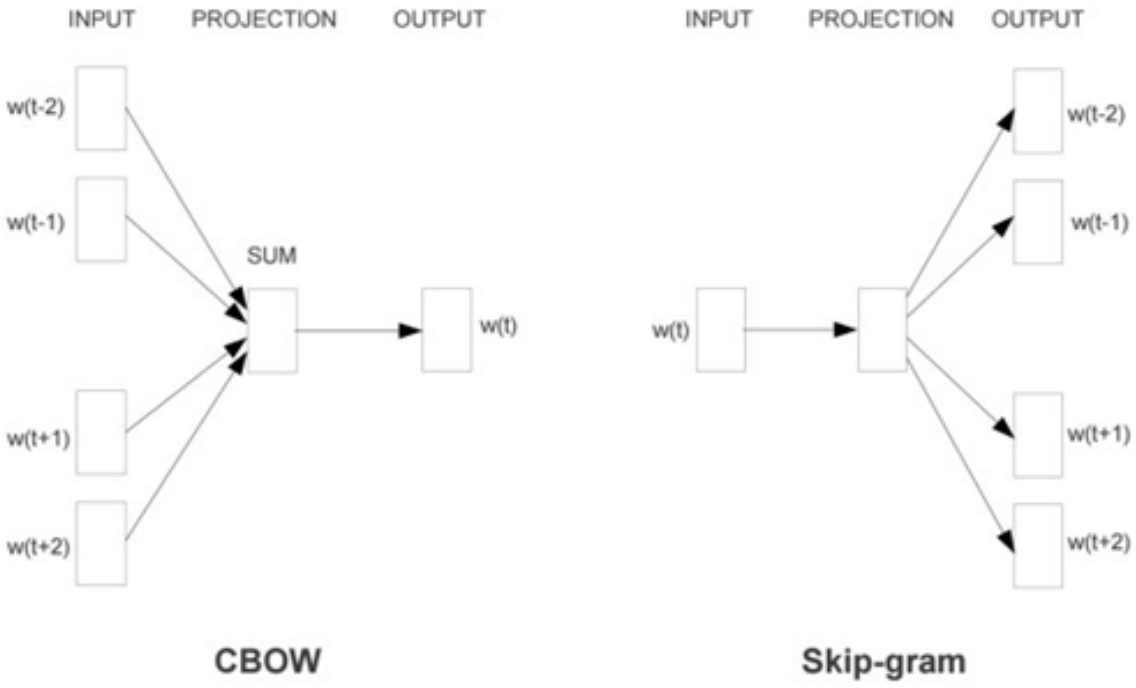
The output of the Word2vec neural net is a vocabulary in which each item has a vector attached to it, which can be fed into a deep-learning net or simply queried to detect relationships between words[20].

The vectors we use to represent words are called neural word embeddings, and representations are strange. One thing describes another, even though those two things are radically different.

Both architectures describe how the neural network "learns" the underlying word representations for each word. Since learning word representations is essentially unsupervised, you need some way to

"create" labels to train the model[13]. Skip-gram and CBOW are two ways of creating the "task" for the neural network -- you can think of this as the output layer of the neural network, where we create "labels" for the given input (which depends on the architecture).

For both descriptions below, we assume that the current word in a sentence is w(t).



[Figure 2-2] CBOW and Skip-gram Structure

CBOW: several times faster to train than the skip-gram, slightly better accuracy for the frequent words

Skip-gram: works well with small amount of the training data, represents well even rare words or phrases.

a. CBOW

CBOW: The input to the model could be w(t-2), w(t-1), w(t+1), w(t+2), the preceding and following words of the current word we are at. The

output of the neural network will be w(t). Hence you can think of the task as "predicting the word given its context"

Note that the number of words we use depends on your setting for the window size[13].

The bag-of-words model is used to represent an unordered collection of words as a vector. One of the most common uses is for simple document classification, an example of this might be the task of classifying an email as spam[15].

b. Skip-gram

Skip-gram: The input to the model is w(t), and the output could be w(t-1), w(t-2), w(t+1), w(t+2). So the task here is "predicting the context given a word". In addition, more distant words are given less weight by randomly sampling them[18]. When you define the window size parameter, you only configure the maximum window size. The actual window size is randomly chosen between 1 and max size for each training sample, resulting in words with the maximum distance being observed with a probability of 1/c while words directly next to the given word are always(!) observed[16].

D. Glove

GloVe is an unsupervised learning algorithm for obtaining vector representations for words, which is also a model for distributed word representation[18]. Training is performed on aggregated global word-word co-occurrence statistics from a corpus, and the resulting representations showcase interesting linear substructures of the word vector space[21].

E. Jieba (Chinese segmentation)

Chinese word segmentation refers to the segmentation of a Chinese character sequence into a single word. Participle is the process of recombining the continuous word sequence into a word sequence according to certain norms.

The existing segmentation algorithms can be divided into three categories, Word segmentation method based on string matching, word segmentation based on understanding and word segmentation based on statistics[25].

The method of word segmentation based on string matching: this method is also called the mechanical word segmentation method. It matches the words in a "sufficiently large" machine dictionary to be analyzed in accordance with a certain strategy[24]. If a string is found in the dictionary, the match is successful (recognizing a word).(1)Forward maximum matching (from left to right) (2) Reverse maximum matching (from right to left) (3) Minimum segmentation (minimizing the number of words in each sentence) (4)Two way maximum matching method (from left to right, right to left two scan)[27].

"Jieba" (Chinese for "to stutter") Chinese text segmentation: built to be the best Python Chinese word segmentation module.

It surpports three segmentation modules:

Exact mode : Trying to cut the sentences in the most accurate way, which is suitable for text analysis

Full mode: Scan all the words that can be used in the sentence, speed is fast,but it can't solve the ambiguity.

Search engine mode: Re segmentation of long words on the basis of the exact mode, raise the recall rate.

It also supports Traditional Chinese character segmentation, custom dictionaries and MIT authorization agreement.

F. Keras

Keras is a high level neural network API。 Keras is only written in Python and is based on Tensorflow, Theano and CNTK backend. Keras is born to support rapid experimentation, which can quickly transform your idea into results. Keras is a high level neural network API。 Keras is only written in Python and is based on Tensorflow, Theano and CNTK backend.

(1)Simple and fast prototype design (Keras is highly modular, minimized, and extensible)

(2) support the combination of CNN and RNN, or the two

(3) fastly CPU and GPU switching

Ⅲ. Proposed method

A. Data pre-processing

a. Pretrained-Word2vec

In this paper, we use the pre-trained vectors downloaded freely in https://spaces.ac.cn/archives/4304. The model contains 256- dimensional vectors for 8 million words and phrases. We gain the more precise relation of words from such a big corpus.

Quantity of corpus	8 million articles
Number of model words	352196, most of Chinese words and some of English words
Model architecture	Skip-Gram + Huffman Softmax
Vector dimension	256 dimensions
Segmentation tool	Jieba segmentation
Training tool	Word2vec of Gensim
others	window size:10 , Minimum word frequency:64, iteration: 10.

[Table 3-1] Pre-trained Word2vec

b. Word Segmentation (Jieba)

Chinese word segmentation refers to segment a string of Chinese characters into separating words. For English text, there are spaces between each English word, which makes it easy to do segmentation of text. Chinese text segmentation is different and difficult to be carried out. In my thesis, I need to get training data with segmentation firstly.

I input the data into the pre-trained Word2vec model to process the word. In this step, I use the Chinese word segmentation tool JIEBA to segment words and delete quotation mark, blanks and commas etc. And then every word will be generated to 256 dimensions vector by word2vec. Here it's the binary classification. I labeled positive and negative as 1 and 0 respectively.

For example, the following is a micro-blog message. After the word segmentation and punctuation, the data are as follows:

'今年 赶上 沙尘暴 没 我 离开 的 那天 不能 呼吸 我能 拍照 不敢 说话 黑城 在 风沙 中 被 淹没 哀 贝姨
：醉美 胡杨林 今年 也 去 了 很 美 中国 摄影师 联盟 ：参展 作品 第五届 微 摄影展 赛 ？金秋 时
节 胡杨林 摄影 北京 - 橙色 咖啡 女 北京 '

In this part, 256 dimension vectors of 120000 words will be obtained after data pre-processing.

c. Training Data and Test Data

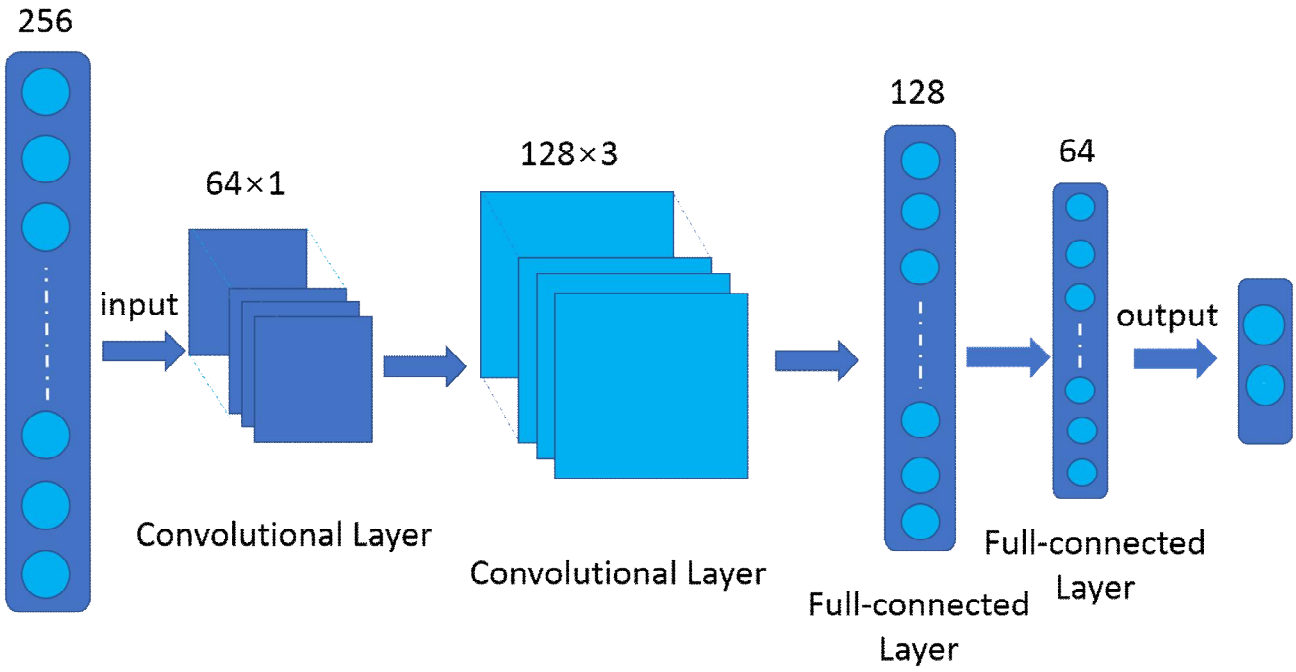
The amount of training data are 120000. Here I devide 10% of the dataset as test dataset to verify the effectiveness of the model that are 12000.

Data set	Input size	Hidden layer numbers	Output label numbers	Batch size	Epoches
Weibo comment	108000*256	4	2	128	50

[Table 3-2] Dataset Parameters

B. ARCHITECTURE OF CNN MODEL

a. CNN Model Structure



[Figure 3-1] CNN Structure

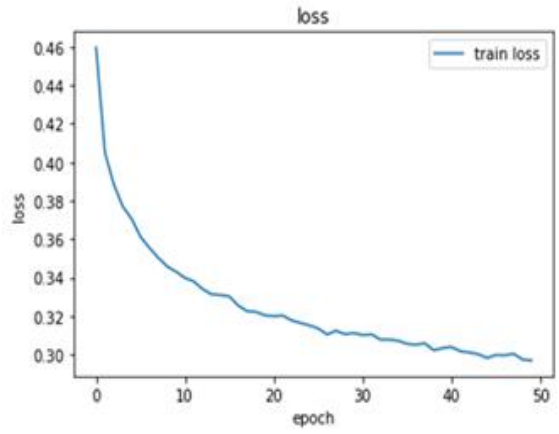
After I get the pre-trained word vectors from word2vec, a convolutional neural network will be trained. In this paper, I use 2 convolutional layers and 3 full-connected layers. The layer configuration of my CNN is shown in Fig 3-1. My experiment is derived from the publicly available keras toolbox[19]. Here, I use convolutional layers, Rectified Linear Unit (ReLU) layers and dropout layers in CNN. In the architecture of CNN ,convolution process spent the most time of training the neural network[27]. Meanwhile, the full-connected layer takes up most of the parameters of the network. The main aim of convolution is to extract the input feature. When training a neural network model, Dropout will be an important trick which is perhaps a biggest invention in the field of

neural networks in recent years proposed by Hinton [21].

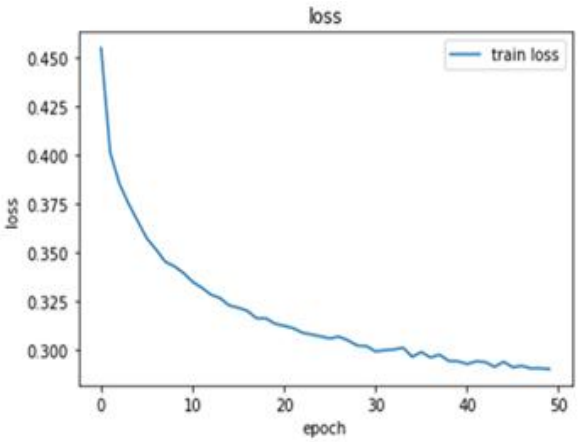
It solves the significant problem in machine learning, which is overfitting[22]. Dropout is a recently introduced algorithm for training neural networks by randomly dropping units during training to prevent their co-adaptation. Dropout consists of setting to zero the output of each hidden neuron with probability of 0.5.(performs setting to zero the output of each hidden neuron with probability of 0.5)[16].The algorithm will drop out the neurons which don't contribute to the forward pass and don't participate in backpropagation. For each training example, a different set of units to drop is randomly chosen[11]. In our framework, we use all the neurons but multiply their outputs by 0.75 after the convolutional layers.

b. Different Batch-size Trade-off

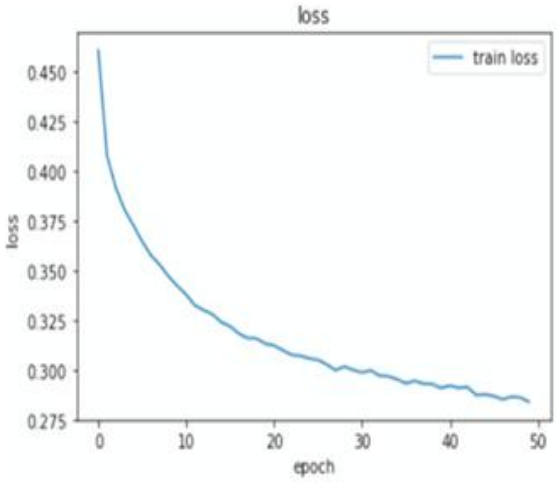
Before constructing final CNN structure and parameters, I set five different batch-size to carry out the experiment. The parameters of them are respectively 32,64, 128, 256, 512.Then I get the loss and accuracy results as figure[3-2] following. From figure[3-2] and figure[3-3],we can see that the average of accuracy rate are almost same. Considering both of the storage and speed ,I set parameter 256 as model batch-size finally.



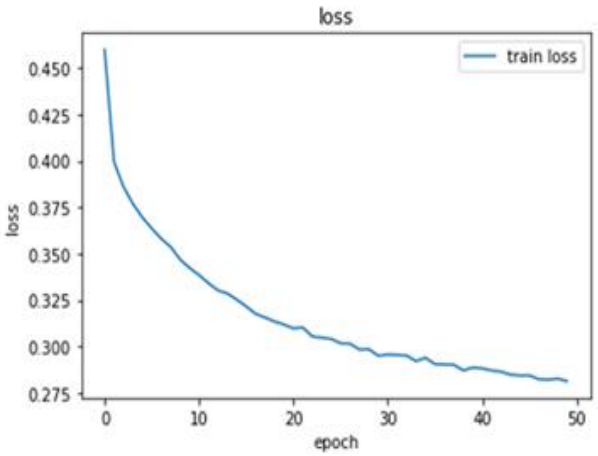
Batch size = 32



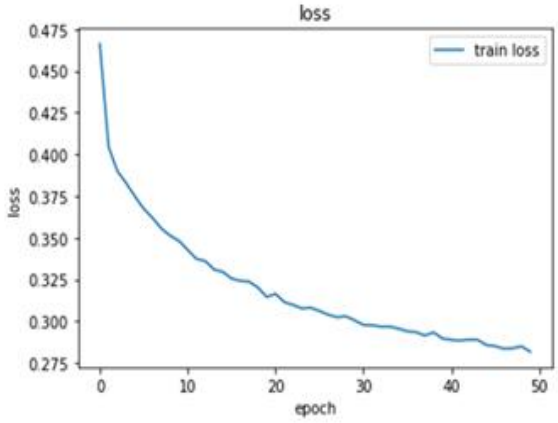
Batch size = 64



Batch size = 128

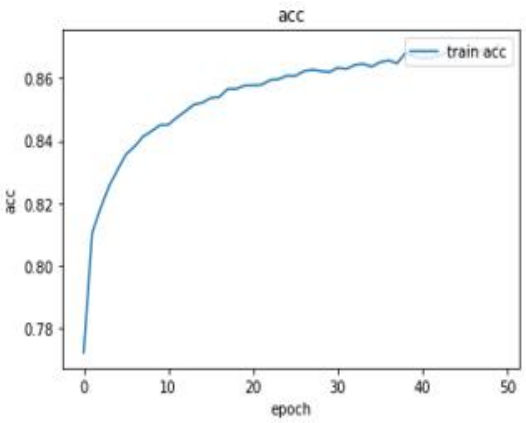


Batch size = 256

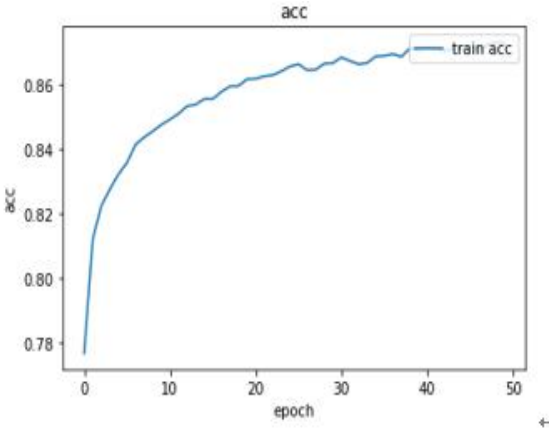


Batch size = 512

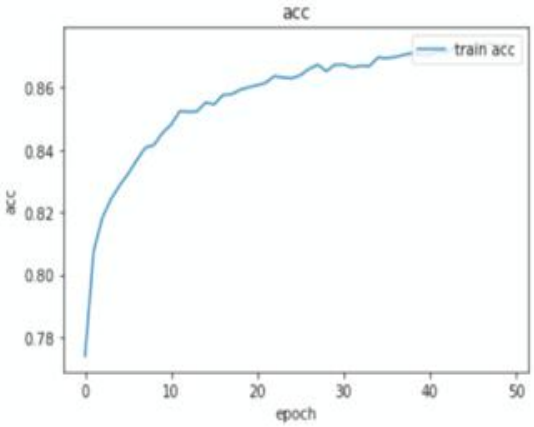
[Figure 3-2] Loss of different input batch size



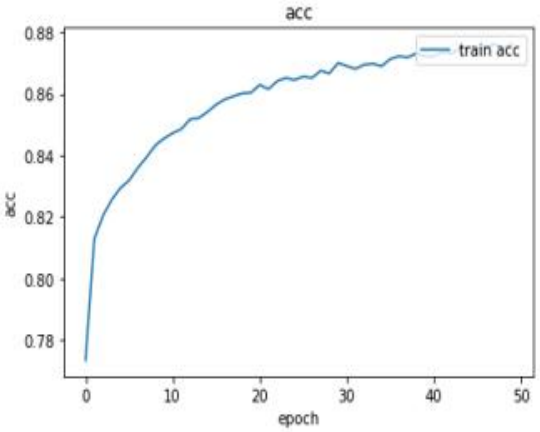
Batch size = 32



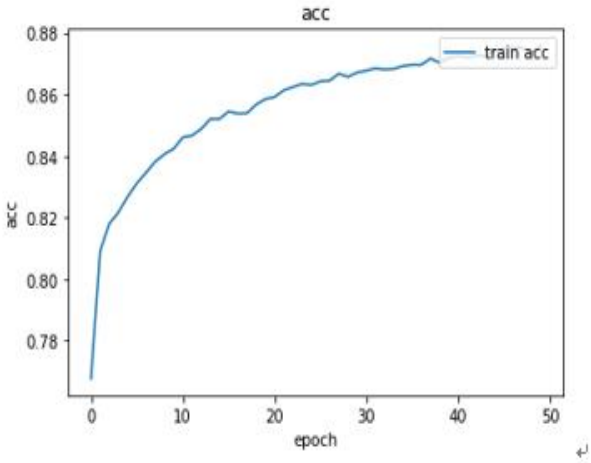
Batch size = 64



Batch size = 128



Batch size = 256

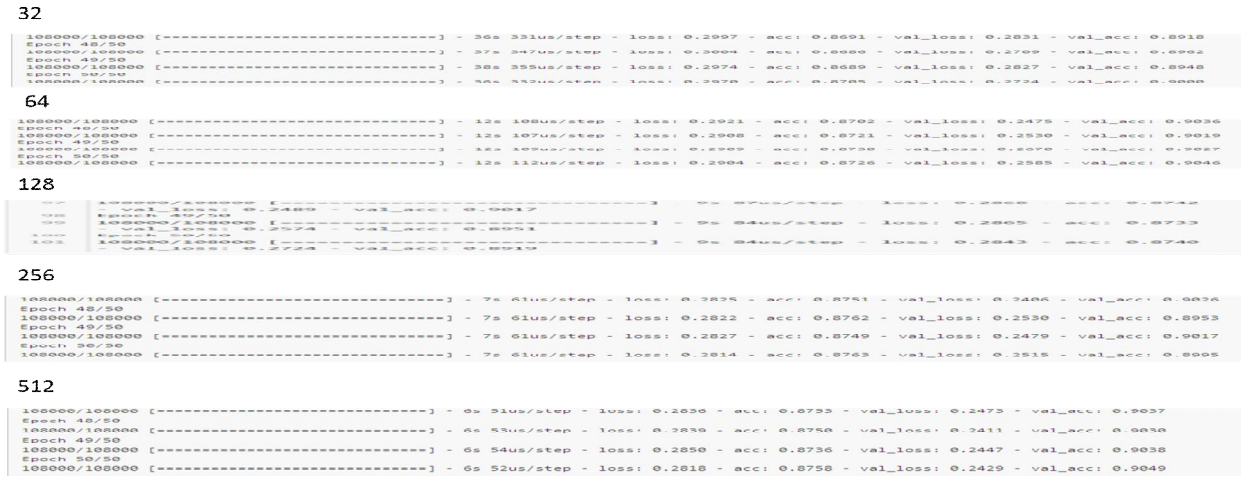


Batch size = 512

[Figure 3-3] Accuracy of different input batch size

Batch Size	Positive	Negative
32	0.9	0.9
64	0.89	0.92
128	0.93	0.85
256	0.86	0.94
521	0.89	0.92

[Table 3-3] different input batch size accuracy comparison



[Figure 3-4] Different batch-size CNN process(Time cost)

Batch Size	Time(s)
32	36
64	12
128	9
256	7
521	6

[Table 3-4] Different batch-size CNN time cost

Figure[3-4] illustrates that Batch_size will affect the optimization and speed of the model. Compared with the normal dataset, if Batch_Size is too small, the training data will be very difficult to converge, which leading to underfitting.

If Batch_Size is increased, the relative processing speed will speed up and the required memory capacity is also increased. It will get better results if epoches are also increased which increase time cost to speed decline.

The correct choice of batch_size is to find the best balance between memory efficiency and memory capacity. Selecting a moderate Batch_Size value is vital important. In this thesis, 256 is the best Batch_Size for CNN model.

c. Comparison of multiple CNN layers

As shown in figure[3-5], CNN1 is my CNN structure and two more methods (CNN2 & CNN3) are tried to reconstruct my CNN model to improve the performance of the system. One is to be added two max-pooling after convolutional layers. And another is to be added one more convolutional layer(256*3), which the structure was increased to 3 convolutional layers and 2 full-connected layers.

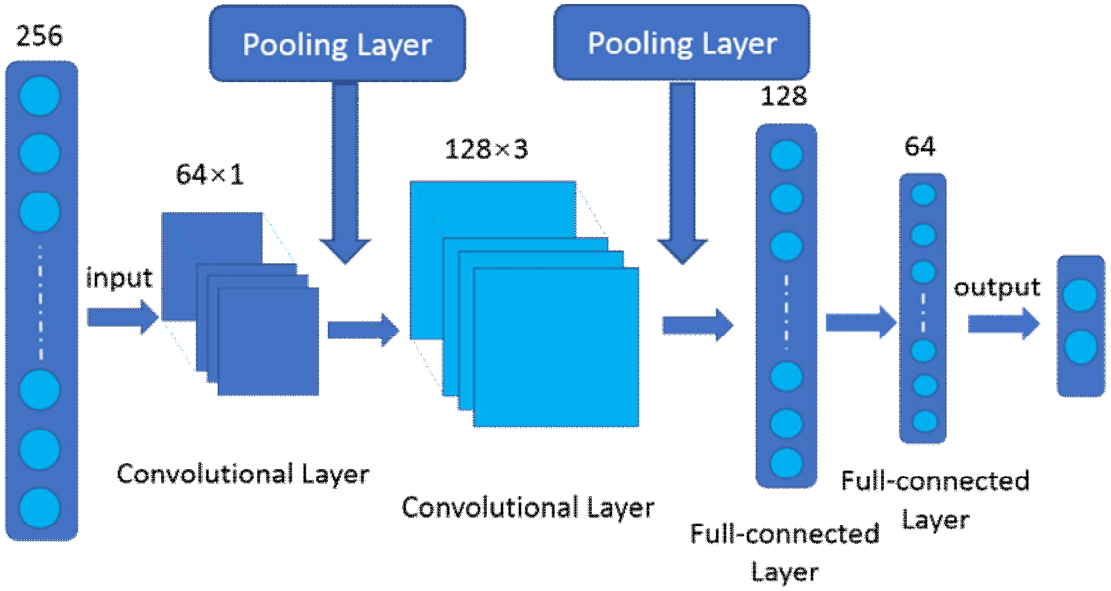
The exact parameters are showed in figure[] as followed:

Parameter/ CNN	CNN 1	CNN 2	CNN 3
Word Vector Dimension	256	256	256
Convolutional Layer	2 (64×1, 128×3)	2 (64×1, 128×3)	3 (64×1, 128×3, 256×3)
Max-pooling Layer	0	2	0
Full-connected Layer	2	2	2
Activation Function	RELU, Softmax	RELU, Softmax	RELU, Softmax
Dropout probability parameter p	0.75	0.75	0.75
Padding	Same	Same	Same
Batch Size	256	256	256

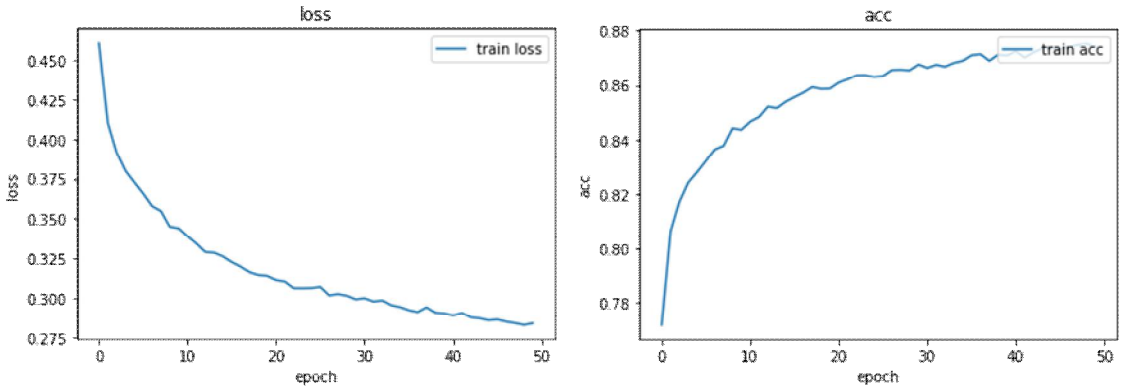
[Table 3-5] CNN1, CNN2 and CNN3 Parameters

a. CNN2

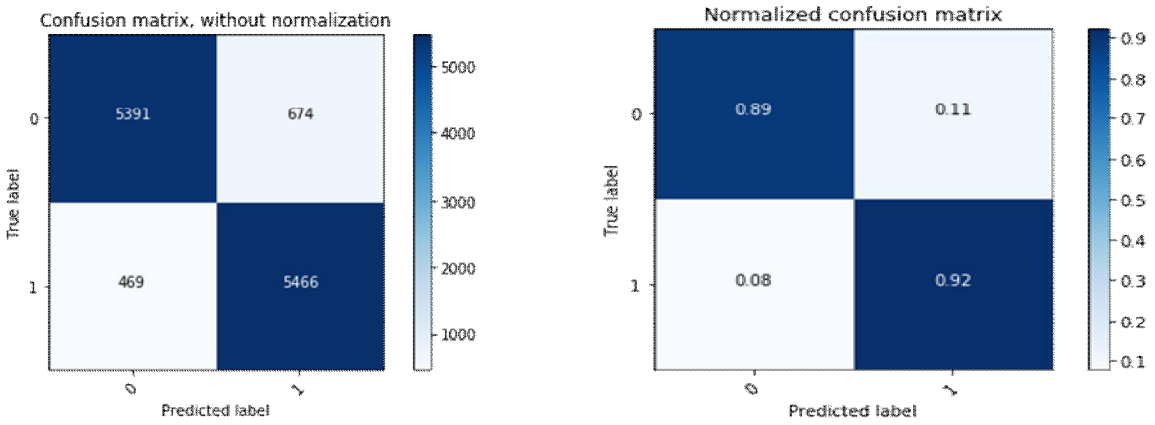
As shown in figure[],on the basic of CNN1,I add two max-pooling layer behind Convolutional layer of CNN1 respectively.



[Figure 3-5] CNN2 Structure



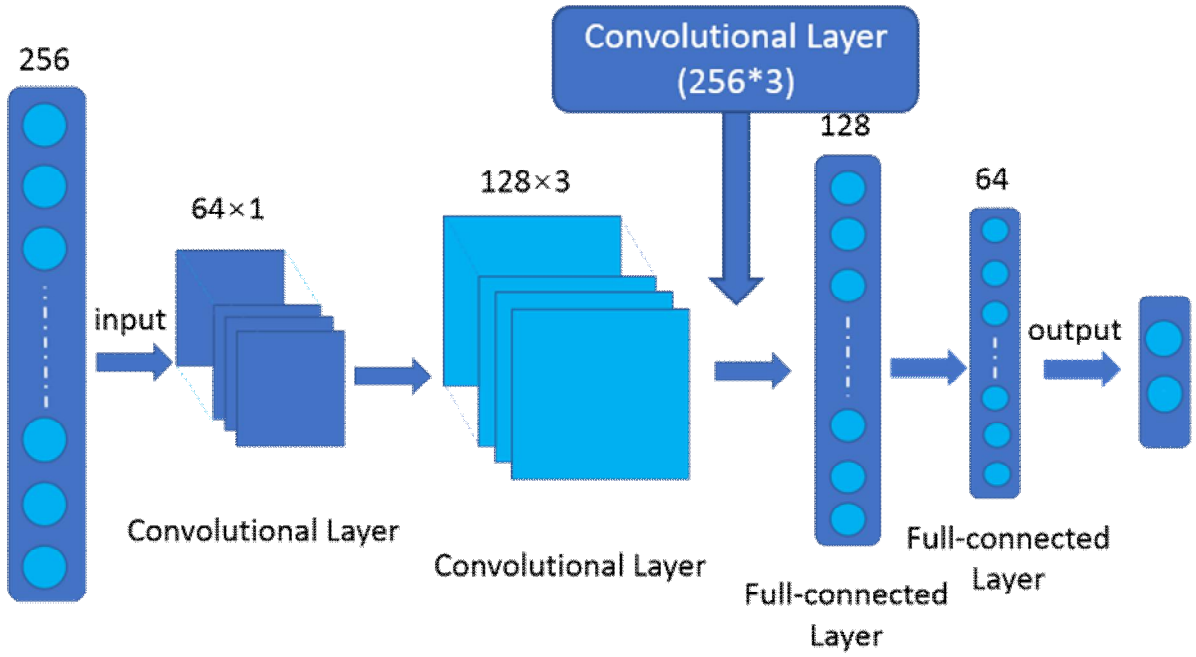
[Figure 3-6] Loss and Accuracy of CNN2



[Figure 3-7] Confusion matrix of CNN2

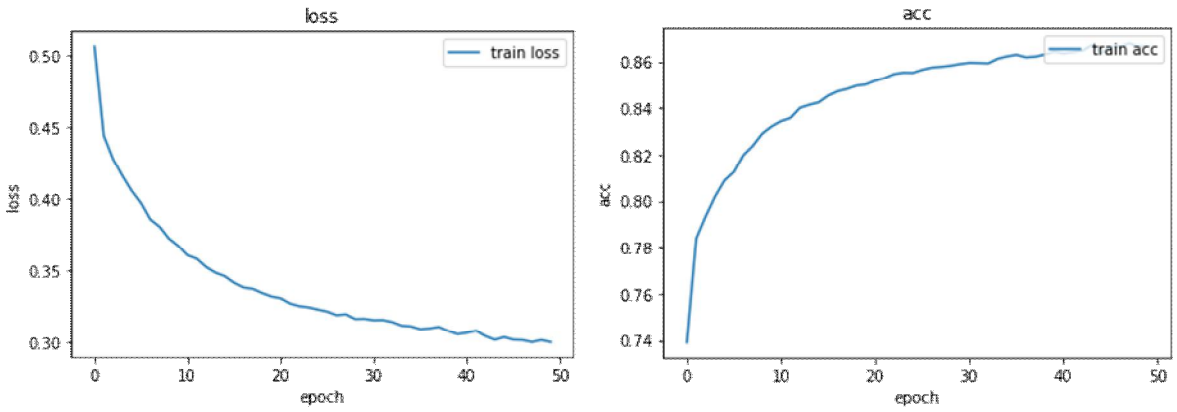
B

b. CNN3

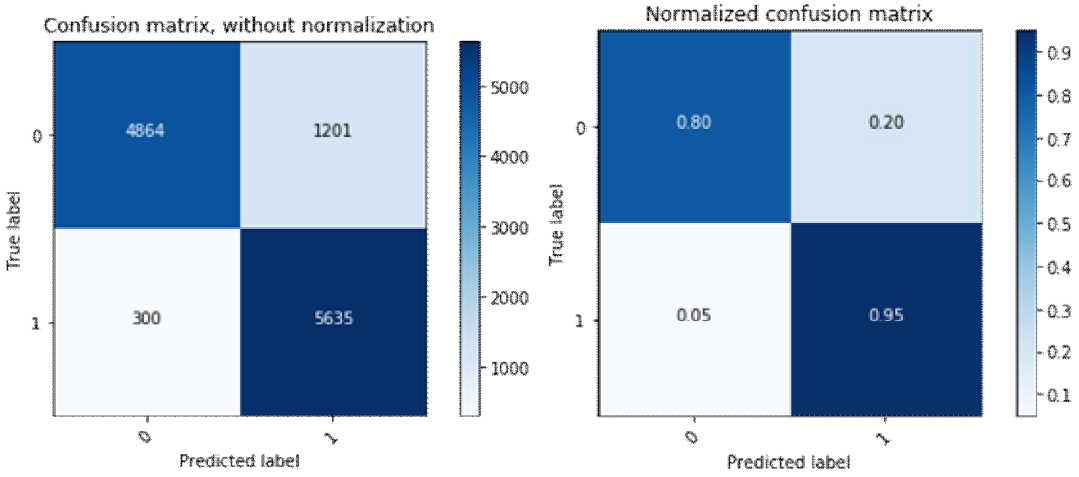


[Figure 3-8] CNN3 structure

As shown in figure[], compared with CNN1, CNN3 has three convolutional layers and two full-connected layers that one more convolutional layer of 256*3 is added.



[Figure 3-9] Loss and Accuracy of CNN2



[Figure 3-10] Confusion matrix of CNN3

As the contract, CNN1 is the most simple and high-performance CNN structure than other CNN structures.

Ⅳ. Experimental Evolution

I evaluate the algorithm and train my convolutional neural network on the public dataset, which is availabled on https://spaces.ac.cn/archives/4304

data pre-trained process are as follows:

In this part, I uploaded pre-trained Word2vec model and segment the word by Jieba Chinese segmentation.

```
import gensim
import pandas as pd
import warnings
warnings.filterwarnings("ignore")
%matplotlib inline

model = gensim.models.Word2Vec.load('./word2vec/word2vec_wx')

import jieba
import re
from zhon.hanzi import punctuation
import numpy as np
from keras.utils import to_categorical
from sklearn.model_selection import train_test_split

def data_pre(filepath):
f=open(filepath,'r',encoding='gbk')
content=f.readlines()
print('the number of data is :%s'%len(content))
res=[]
for line in content:
line=line.strip()
sign=['@','。','[',']','~','/','#']
line=re.sub("[%s]+"%punctuation,"",line)
word_list=jieba.cut(line)
thisline_res=''
for word in word_list:
if word not in sign:
thisline_res+=word+' '
res.append(thisline_res)
return res
pos_content=data_pre(filepath='./pos60000.txt')
neg_content=data_pre(filepath='./neg60000.txt')
```

[Figure 4-1] Jieba segmentation coding

CNN model

As shown in figure[4-3], there are two convolutional layers and two full-connected layers in the proposed CNN model. The first convolutional layer has 64 filters that the kernal-size=1 and another has 128 filters (kernal-size=3). All of the padding are "same". RELU activation function is used on convolutional layers and full-connected layers. Softmax is used as output function.

```
from keras.models import Model
from keras.layers import Input, Dense,Conv1D,MaxPool1D,Dropout,Flatten

def model():
    x_input=Input(shape=(256,1))

    x=Conv1D(filters=64,kernel_size=1,padding='same',activation='relu')(x_input)
    x=Dropout(0.75)(x)

    x=Conv1D(filters=128,kernel_size=3,padding='same',activation='relu')(x)
    x=Dropout(0.75)(x)

    x=Flatten()(x)

    x=Dense(128,activation='relu')(x)

    x=Dense(64,activation='relu')(x)

    x=Dense(2,activation='softmax')(x)
    model=Model(inputs=x_input,outputs=x,name='test_model')
    return model


train_model=model()

train_model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy']

history=train_model.fit(X_train,y_train,validation_data=(X_test,y_test),epochs=50,batch_size=256)
```

[Figure 4-2] CNN Model Structure

Process

```
108000/108000 [==============================] - 7s 62us/step - loss: 0.2951 - acc: 0.8700
- val_loss: 0.2629 - val_acc: 0.8967
Epoch 31/50
108000/108000 [==============================] - 7s 62us/step - loss: 0.2957 - acc: 0.8690
- val_loss: 0.2523 - val_acc: 0.9024
Epoch 32/50
108000/108000 [==============================] - 7s 62us/step - loss: 0.2955 - acc: 0.8681
- val_loss: 0.2526 - val_acc: 0.8961
Epoch 33/50
108000/108000 [==============================] - 7s 63us/step - loss: 0.2951 - acc: 0.8695
- val_loss: 0.2483 - val_acc: 0.9020
Epoch 34/50
108000/108000 [==============================] - 7s 64us/step - loss: 0.2922 - acc: 0.8697
- val_loss: 0.2482 - val_acc: 0.9025
Epoch 35/50
108000/108000 [==============================] - 7s 62us/step - loss: 0.2940 - acc: 0.8689
- val_loss: 0.2524 - val_acc: 0.9022
Epoch 36/50
108000/108000 [==============================] - 7s 63us/step - loss: 0.2906 - acc: 0.8713
- val_loss: 0.2524 - val_acc: 0.9017
Epoch 37/50
108000/108000 [==============================] - 7s 60us/step - loss: 0.2904 - acc: 0.8721
- val_loss: 0.2498 - val_acc: 0.8994
Epoch 38/50
108000/108000 [==============================] - 7s 63us/step - loss: 0.2902 - acc: 0.8718
- val_loss: 0.2482 - val_acc: 0.9021
Epoch 39/50
108000/108000 [==============================] - 7s 64us/step - loss: 0.2871 - acc: 0.8731
- val_loss: 0.2466 - val_acc: 0.9017
Epoch 40/50
108000/108000 [==============================] - 7s 63us/step - loss: 0.2887 - acc: 0.8722
- val_loss: 0.2470 - val_acc: 0.8999
Epoch 41/50
108000/108000 [==============================] - 7s 62us/step - loss: 0.2883 - acc: 0.8720
- val_loss: 0.2538 - val_acc: 0.9032
Epoch 42/50
108000/108000 [==============================] - 7s 61us/step - loss: 0.2871 - acc: 0.8735
- val_loss: 0.2498 - val_acc: 0.9030
Epoch 43/50
108000/108000 [==============================] - 7s 64us/step - loss: 0.2865 - acc: 0.8730
- val_loss: 0.2557 - val_acc: 0.8997
Epoch 44/50
108000/108000 [==============================] - 7s 61us/step - loss: 0.2849 - acc: 0.8746
- val_loss: 0.2468 - val_acc: 0.9038
Epoch 45/50
108000/108000 [==============================] - 7s 61us/step - loss: 0.2844 - acc: 0.8746
- val_loss: 0.2451 - val_acc: 0.9019
Epoch 46/50
108000/108000 [==============================] - 7s 62us/step - loss: 0.2845 - acc: 0.8749
- val_loss: 0.2468 - val_acc: 0.9028
Epoch 47/50


108000/108000 [==============================] - 7s 61us/step - loss: 0.2825 - acc: 0.8751
```

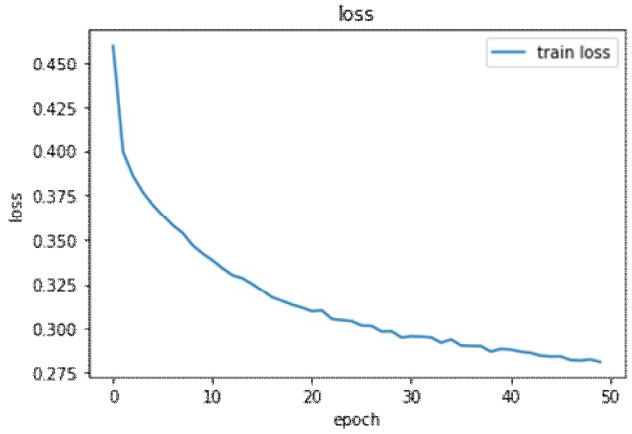
```
- val_loss: 0.2406 - val_acc: 0.9026
Epoch 48/50
108000/108000 [==============================] - 7s 61us/step - loss: 0.2822 - acc: 0.8762
- val_loss: 0.2530 - val_acc: 0.8953
Epoch 49/50
108000/108000 [==============================] - 7s 61us/step - loss: 0.2827 - acc: 0.8749
- val_loss: 0.2479 - val_acc: 0.9017
Epoch 50/50
108000/108000 [==============================] - 7s 61us/step - loss: 0.2814 - acc: 0.8763
- val_loss: 0.2515 - val_acc: 0.8995
```

[Figure 4-3] Process of CNN

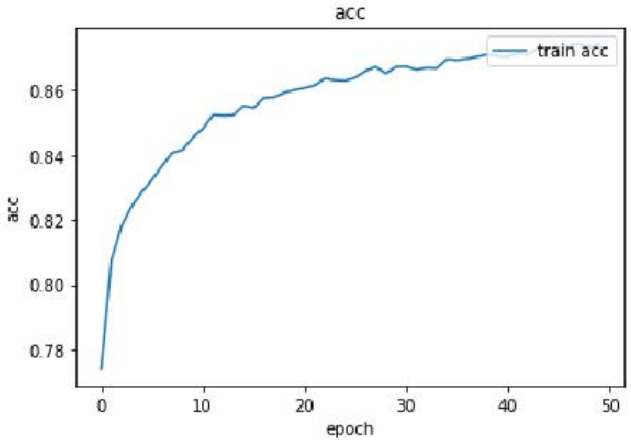
RESULTS:

LOSS:

Create the loss and accuracy picture



[Figure 4-4] loss figure



[Figure 4-5] Accuracy figure

CONFUSION MATRIX:

Fig[4-7] illustrates the confusion matrix obtained when running our system on Weibo comments. 1,0 for positive and negative respectively.

```
from sklearn.metrics import confusion_matrix
y_predict=train_model.predict(X_test)
y_predict=y_predict.argmax(axis=1)
y_true=y_test.argmax(axis=1)
def plot_confusion_matrix(cm, classes,
normalize=False,
title='Confusion matrix',
cmap=plt.cm.Blues):
"""
This function prints and plots the confusion matrix.
Normalization can be applied by setting `normalize=True`.
"""
if normalize:
cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
print("Normalized confusion matrix")
else:
print('Confusion matrix, without normalization')
print(cm)
plt.imshow(cm, interpolation='nearest', cmap=cmap)
plt.title(title)
plt.colorbar()
tick_marks = np.arange(len(classes))
plt.xticks(tick_marks, classes, rotation=45)
plt.yticks(tick_marks, classes)
fmt = '.2f' if normalize else 'd'
thresh = cm.max() / 2.
for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
plt.text(j, i, format(cm[i, j], fmt),
horizontalalignment="center",
color="white" if cm[i, j] > thresh else "black")
plt.tight_layout()
plt.ylabel('True label')
plt.xlabel('Predicted label')

class_names=[0,1]
cnf_matrix = confusion_matrix(y_true, y_predict)
np.set_printoptions(precision=2)

plt.figure()
plot_confusion_matrix(cnf_matrix, classes=class_names,
title='Confusion matrix, without normalization')
plt.figure()
plot_confusion_matrix(cnf_matrix, classes=class_names, normalize=True,
title='Normalized confusion matrix')
plt.show()
```

[Figure 4-6] Process of Confusion matrix

Confusion matrix, without normalization

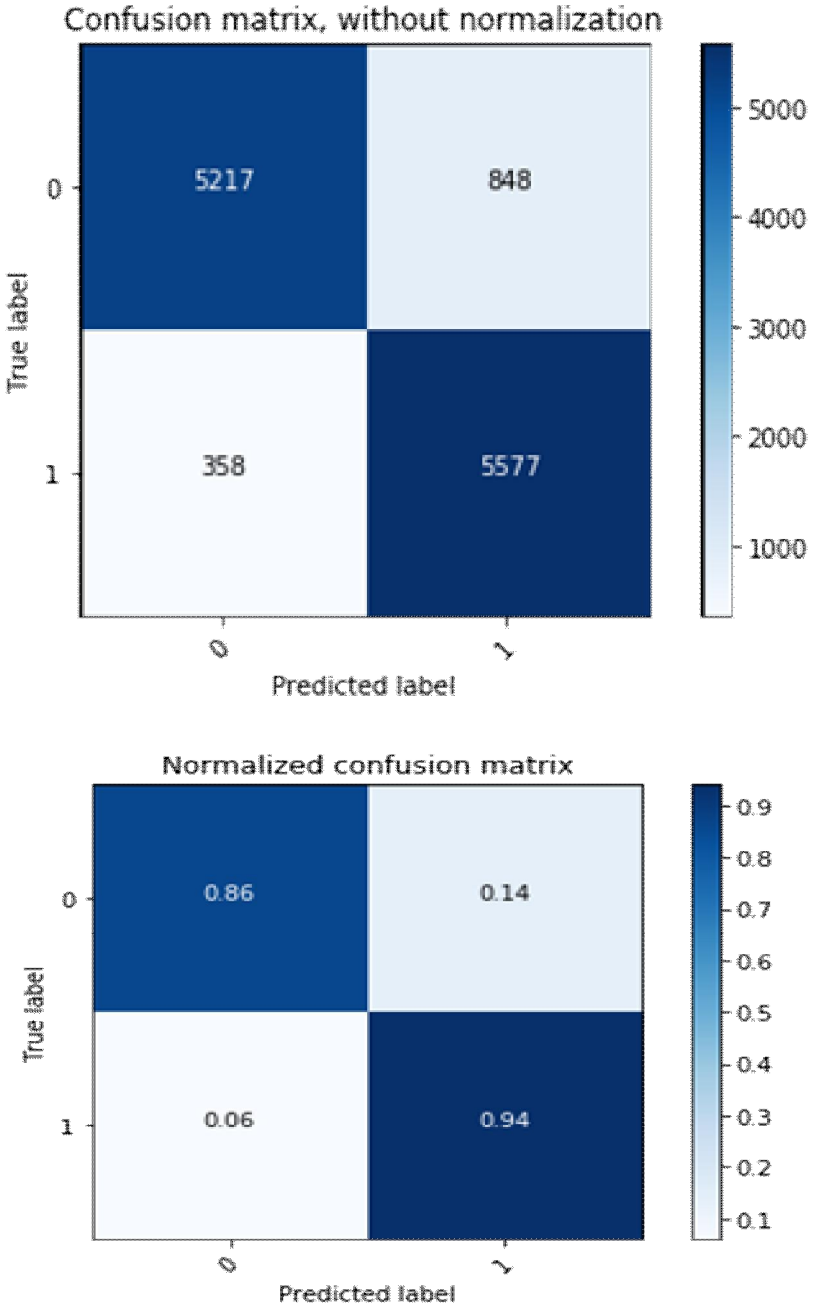
[[5217 848]

[359 5577]]

Normalized confusion matrix

[[0.86 0.14]

[0.06 0.94]]



[Figure 4-7] Confusion matrix of CNN

V. Conclusions

Sentiment analysis is a challenging and attractive job. In this paper, I apply a convolutional neural network to solve this problem. An useful framework called Word2vec + CNN is described,which is evoluated on the available public dataset of Weibo's comments. The experimental results suggest that convolutional neural networks that are properly trained can outperform the shallow classification algorithms. Meanwhile, the way of pretrained+ fine-tuning is adopted to train the convolutional neural network on the natural language processing task, which is better than some other deep learning models like Recursive Neural Network and Matrix-Vector Recursive Neural Network.

I believe that sentiment analysis will become more important since large scale amount of online users' test information is needed to meet users satisfication. In this paper, lot of valuable information can be digged out by the data sentiment analysis, which will have a wide application like public opinion analysis, product reviews analysis and so on. I believe that my sentiment analysis results can give some further inspiration to other researchers.

References

[1] Abbasi A, Chen H, Salem A. Sentiment analysis in multiple languages[J]. Acm Transactions on Information Systems, 2015, 26(3):1-34.

[2] Mikolov T. Statistical language models based on neural networks[J].Presentation at Google, Mountain View, 2nd April, 2012.

[3] Le Q, Mikolov T. Distributed representations of sentences and documents[C]//Proceedings of the 31st International Conference on Machine Learning (ICML-14). 2014: 1188-1196.

[5] MapReduce Dataflow, http://blog.cloudera.com/blog/

[6] S. Chandrasekar, R. Dakshinamurthy, P. G. Sechakumar, B. Prabavathy and Chitra Babu, "A Novel Indexing Scheme for Efficient Handling of Small Files in Hadoop Distributed File System," Proceeding of International Conference on Computer Communication and Informatics(ICCCI-2013), pp. 1-8, January 2013.

[7] Jing Zhang, Gongqing Wu, Xuegang Hu and Xindong Wu, "A Distributed Cache for Hadoop Distributed File System in Real-time Cloud Services," Proceeding of International Conference on Grid Computing, pp. 12-21, September 2012.

[8] Krizhevsky A, Sutskever I, Hinton G E, “Imagenet classification with deep convolutional neural networks,” In Neural Information Processing Systems, pp. 1097-1105, 2012.

[9] Mikolov T, Chen K, Corrado G, et al, “Efficient estimation of word representations in vector space,” arXiv preprint arXiv:1301.3781, 2013.

[10] Mikolov T, Sutskever I, Chen K, et al, “Distributed representations of words and phrases and their compositionality,” In Neural Information Processing Systems, pp. 3111-3119, 2013.

[11] Mikolov T, YihW, Zweig G, "Linguistic Regularities in Continuous Space Word Representations," In HLT-NAACL, pp. 746-751, 2013.

[12] Jia Y, Shelhamer E, Donahue J, et al, "Caffe: Convolutional architecture for fast feature embedding," In Proceedings of the ACM International Conference on Multimedia. ACM, pp. 675-678, 2014.

[13] Liu B, Dai Y, Li X, et al, "Building text classifiers using positive and unlabeled examples," In Data Mining (ICDM), 2003. IEEE 3th International Conference on. IEEE, pp. 179-186, 2003.

[14] Li G, Hoi S C H, Chang K, et al, "Micro-blogging sentiment detection by collaborative online learning," In Data Mining (ICDM), 2010. IEEE 10th International Conference on. IEEE, pp. 893-898, 2010.

[15] Pang B, Lee L, "Opinion mining and sentiment analysis," Foundations and Trends in Information Retrieval, 2.1-2: pp. 1-135, 2008.

[16] Snyder B, Barzilay R, "Multiple Aspect Ranking Using the Good Grief Algorithm," In HLT-NAACL, pp. 300-307, 2007.

[17] Socher R, Lin C C, Manning C, et al, "Parsing natural scenes and natural language with recursive neural networks," In Proceedings of the 28th International Conference on Machine Learning (ICML), pp. 129-136, 2011.

[18] Socher R, Perelygin A, Wu J Y, et al, "Recursive deep models for semantic compositionality over a sentiment treebank," In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1631-1642, 2013.

[19] LeCun Y, Bottou L, Bengio Y, et al, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, 1998.

[20] Ciresan D C, Meier U, Masci J, et al, "Flexible, high performance convolutional neural networks for image classification," In IJCAI

Proceedings-International Joint Conference on Artificial Intelligence, vol. 22, no. 1, pp. 1237-1242, 2011.

[21] Krizhevsky A, Sutskever I, Hinton G E, "Imagenet classification with deep convolutional neural networks," In Neural Information Processing Systems, pp. 1097-1105, 2012.

[22] Ciresan D, Meier U, Schmidhuber J, "Multi-column deep neural networks for image classification," In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, pp. 3642-3649, 2012.

[23] Deng J, Dong W, Socher R, et al, "Imagenet: A large-scale hierarchical image database," In Computer Vision and Pattern Recognition (CVPR), 2009 IEEE Conference on. IEEE, pp. 248-255, 2009.

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

[25] Pang B, Lee L, "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales," In Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, pp. 115-124, 2005.

[26] Socher R, Huval B, Manning C D, et al, "Semantic compositionality through recursive matrix-vector spaces," In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning. Association for Computational Linguistics, pp. 1201-1211, 2012.

[27] Chatfield K, Simonyan K, Vedaldi A, et al, "Return of the devil in the details: Delving deep into convolutional nets," arXiv preprint arXiv:1405.3531, 2014.

[28] Hinton G E, Srivastava N, Krizhevsky A, et al, "Improving neural networks by preventing co-adaptation of feature detectors," arXiv preprint arXiv:1207.0580, 2012.