Lecture Evaluation Sentences Analysis

Identify Lecture Evaluations using Word Embedding and Text-Convolution Neural Network

Lim HoJun
Dept. Of Computer Engineering
The College of Electronics and Information, KyungHee University
Suwon, Korea
lhj@oslab.khu.ac.kr
Related Git Repository: http://khuhub.khu.ac.kr/2017-2-MachineLearning/TermProject then Folder 2014104140

Abstract

When the season of apply lecture has come, most of university students try to find a 'good' lecture. For instance of those try, Access a university community site and read other student's lecture evaluation who finish that lecture. Given that there is lots of lecture evaluation data, you will get critical information about lecture that you want to take. But there are many evaluation with long sentences each of lectures. It means you need to read all of that evaluation to get information and it will spend much of time. In that reason, I did research trying to implement the model which can analysis the writers intent in lecture evaluation sentences. If the model can analysis writers intent with high accuracy, it might reduce the time wasted in reading every single lecture evaluation sentences.

To extract meaningful information from the lecture evaluation sentences, Deep Learning, which is widely used in emotion analysis research, is applied. For more accurate analysis, Word Embedding and Convolutional Neural Network which show highly performance about natural language processing are also applied . Implemented model shows 72~80% recall, 70~74% precision when analysis lecture evaluation sentences which include the writers intent recommend or not.

Keywords— Analysis text, Word Embedding, Text-Convolution Nerual Network

I. INTRODUCTION

Text datas are naturally occurs when people do daily life, cultural life, and so on. For instance, writing review after watch movie and buy some product or write his daily life on Social Network Service like Facebook. If processing work like analysis writers intent in those text data without requiring human hands is possible, than we can get various benefits. In that reason, these days many research are in process.

In this paper, also trying to extract meaningful information from text data for provide handy service to users. User means, university students. When the season of apply lecture has come, most of university students try to find a 'good' lecture. For instance of those try, First get information about lecture from elder who finish the lecture that want to apply. Second, Access a university community site and read other student's lecture evaluation sentences who already pass that lecture. Reading other students lecure evaluation sentences is one of the most obvious way to get critical information about lecture that want to apply. But there are many evalutaion text datas with long sentences each of lectures. So reading every single one of the evaluation might bothering us. Thus, I try to implement the model which can analysis the writers intent. If implemented model can analysis writers intent which is writer recommend this lecture or not, it can reduce the time wasted in read all sentences. For well performance at analysis writer's intent, i applied Word Embedding and Text Convolution Neural Network which show highly performance about natural language processing.

II. IN THE PREVIOUS PAPER

A. Support Vector Machine

[2]~[4] are proceed sentiment, intent analysis using Support Vector Machine. [2] convert syllable kernel which compose short sentence to phonemic kernel compose short sentence. After then , it apply Support Vector Machine to classify is the sentences mean positive or negative. Because this paper classify sentiment in unit of sentence, it doesn't fit in lecture evaluation data set which need to analysis the dominant writer's intent. [3] extract sentiment tag and every node in stanford emotional words set tree,

applied Support Vector Machine to analysis two sentiment positive, negative in unit of sentence. But this paper proceed with english sentences, words not korean. And also this research didn't build a word dictionary which needed to enhance the performance. [4] also applied Support Vector Machine, used twitter dataset. And classify three sentiment which is positive, neutral, negative.

III. DATASET

A. Build Dataset

There is two websites to scrap the lecture evaluation sentences and build text data set. First one is Kyung Hee University Information System(KHUIS). In KHUIS, there is literally tones of lecture evaluation data. Because every student have to assess the lecture that they taken in semester to check their grade earlier than regular schedule. But student assess the lectures roughly. Only reason they do assess the lecture is to check their grade, which means their assessment about lecture has less reliability. Second one is university student community website called 'Every Time'. In this site, there is no external force to make them write lecture evaluation sentences. Which means these text data are has reliability and also rated every lecture perfectly from 1 to 5. In that reason, I scrap lecture evaluation sentence data from 'Every Time' and build data set. In this data set, there are total 11,000 lecture evaluation data.

B. Preprocess Dataset

Dataset are written by korean. In case of korean, sentences data are composed of complicate architecture because of servey. Thus for efficiency, I remove all servey from lecture evaluation sentences . For removing servey, I applied KoNLPy's morphological analyzer.

To proceed supervised learning, i labeled each of lecture evaluation text data using their rate. labeled rate 1~2 to negative, and rate 3~5 to positive.

Artificial Neural Networks used data composed of numbers as an input. Thus, for proceed training neural networks, i converted words to index. Extract 12,000 word by having most frequency from dataset, and build word dictionary by giving index to those words.

C. Padding

In this paper, maximum length of sentence was set in two different case, 30 and 80. If the sentence shorter than maximum length, then add 0 which means 'UNKWOWN" for padding. If the sentence longer than maximum, then just cut off the sentence.

IV. WORD EMBEDDING

A. One-hot representation VS Distributed representation

There are two way to embed word. First one is One-hot representation and second one is distributed representation. Actualy word means word that converted to index using word dictionary. One-hot representaion is one of the easiest way to embed the word. In vertor, just allocate 1 for the index and allocate 0 for the rest. This is so simple and easy to implement, but this representation has downsides. If there is another word that didn't know before, than one-hot representation need extra dimension to represent this word which means increase computation time. And also one-hot representation can't represent relationship between words. On the other hands, distributed representation can. Because, distributed representation do not allocate 0 for all except 1. Distributed representation uses a series of real number to compose vector. So in this vector, we can calculate direction, and also we can infer that vectors have similar direction are similar words. Figure 1 is example of one-hot and distributed representation.

$$w_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} w_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \cdots w_n = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

(a) 국소표현(one-hot representation)

$$w_1 = \begin{bmatrix} 0.3 \\ 0.9 \\ 0.8 \\ 0.5 \\ 0.1 \end{bmatrix} w_2 = \begin{bmatrix} 0.2 \\ 0.9 \\ 0.9 \\ 0.5 \\ 0.1 \end{bmatrix} \cdots w_n = \begin{bmatrix} 0.8 \\ 0.1 \\ 0.2 \\ 0.1 \\ 0.9 \end{bmatrix}$$

(b) 분산표현(distributed representation)

Figure 1 Example of one-hot and distributed representation

In this paper, i set embedding size as 128.

B. Word2Vec's skip-gram model

For train word dictionary with distributed representation, I applied Google's Word2Vec. Specifically i used Skip-gram architecture[6] in Word2Vec. The matrix, which is word dictionary is trained in the direction of reducing the error between the input data and the output value after predicting the output values which is close to input word in sentences. For instance, lets say there is sentences "I really like this lecture". Then, to train the word 'like', the input and output is given like this, (like, I), (like, really), (like, this), (like,lecture) for train word dictionary. Figure 2 is the architecture of skip-gram model.

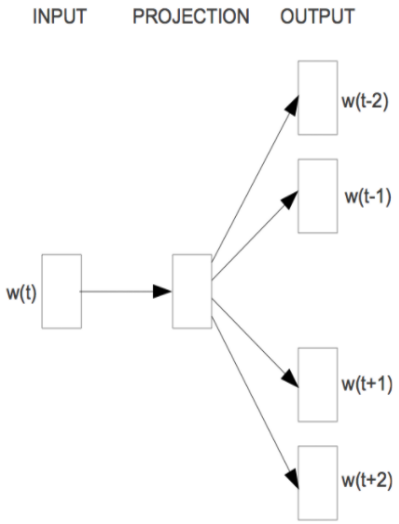


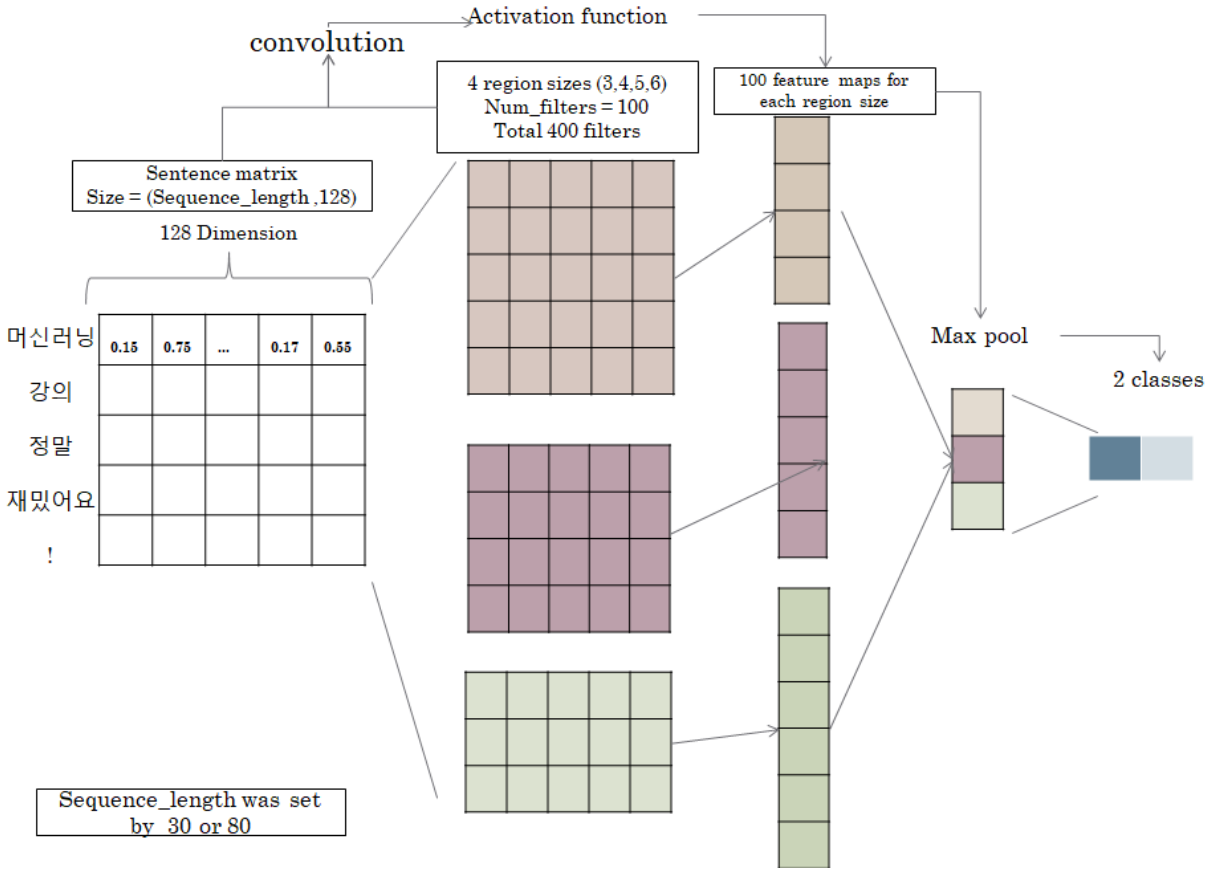
Figure 2 Skip-gram Architecture

Learning speed of skip-gram are slow compare to other model, but this show much better performance than other model. So in this paper, applied skip-gram to train word dictionary.

V. TEXT-CONVOLUTION NEURAL NETWORK

[7],[8] shows the highly performance of Convolutional Neural Network about natural language processing and text sentiment analysis. Especially [8] uses Word2Vec to train distributed representation and use Word2Vec lookup table as an input of convolutional neural network. This paper follow the model that introduced in [8] and add sum layers. Figure 3 is the architecture of Text-CNN that i implemented using architecure of [8].

- First layer convert words in sentences to set of index using word dictionary and then convert index to vector by look up pretrained Word2Vec which is trained by skip-gram. Then using those vectors, compose 2-dimension table
- Second layer build feature maps to extract features from set of words which mean embedded vectors, using numerous of filters that has differnt sizes. In this paper, I selected the 4 region sizes of filter which is 3,4,5 and 6. Region sizes means the unit of how many near words you want to see. And each of region sizes, i used 100 filters. So i used total 400 filters. shape of filters are 'Region_size'x'embedding_size'x1x'filter_size'. So the kinds of filter's shape is 3x128x1x100, 4x128x1x100, 5x128x1x100, 6 x128x1x100.
- After convolution and activation function which is RELU, third layer proceed the max pooling process. Every each of 400 feature maps pick up the only one max value. So after through this layer, there are only 400 values.
- Finally maxpooled 400 values convert to 2 values after through 2 fully connected layer. Then using softmax and cross entropy as cost function, calculate error and transfer these error from final layer to first layer. I also used L2 regularization and Dropout[9] to prevent overfitting.
- Stride of convolution and max pooling are set as 1x1x1x1 and i used Adamoptimizer and learning rate are set as 0.001.



VI. EVALUATION

A. *Evaluation enviroment*

- CPU: Intel i5-7500, 3.40GHz/ RAM: 32GB/ GPU : NVIDIA Geforce Titan X 11GB/ OS: Window 10
- Framework: Tensorflow-GPU 1.2.0

B. *Result*

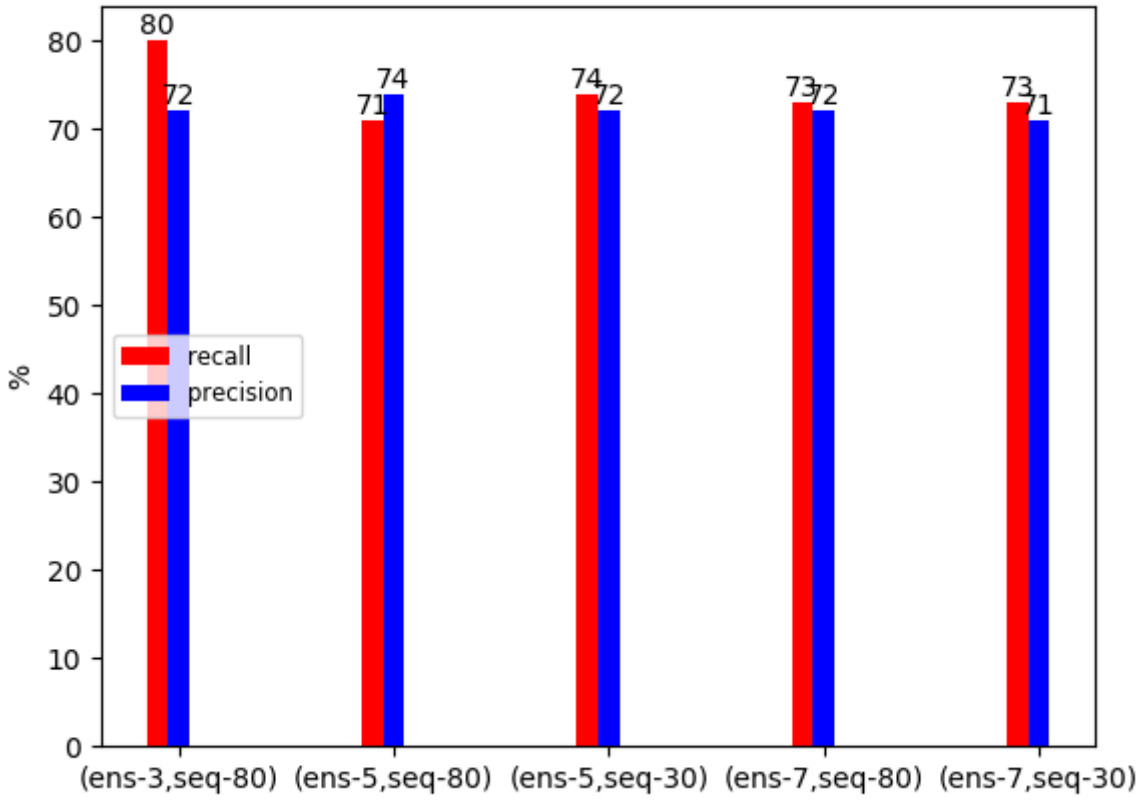
Positive labeled data was 8770, negative labeled data was 2330. Since the ratio of the data is not correct, experiment was proceed after set ratio to 1:1. Thus total number of data is 4660 composed of positive data 2330 and negative data 2330. The amount of data set is not enough, I did 10-fold cross validation. Table 1 is the confusion matrix of text-CNN, which sequence length was set as 80.

		Predict	
	TextCNN	P	N
Label	T	1720	610
	F	715	1615

Recall:73.8%, Precision:73.4 %

TABLE I. CONFUSION MAXTRIX OF TEXT-CNN

As the result of my experiments, the accuracy of 71.4% was obtained by adjusting 3335 in total 4660 dataset. The recall was 73.8% and precision was 73.4%. After obtain these result, I proceed the evaluation using ensemble composed of text-CNN. Table 2 shows the result of evaluation using ensemble with different sequence length which is set as 30 or 80 and for calculate the result, did majority voting.



Evaluation was proceeded with 5 different parameters, number of text-CNN that ensembled and number of sequence length.

In the result, ensemble with three text-CNN models and 80 sequence length shows the best performance. Recall was enhanced up to 80%. On the other hands, increase of the number of models didn't shows the enhance of performance . In my case, ensemble with five text-CNN models and 30 sequence length show slightly better performance than other models except ensemble with three models.

VII. Conclusion

In other research currently proceeded , they shows 86.4%, 85.2% accuracy at most when using different kinds of CNN and shows 81.47% accuracy when using RNN with LSTM. I think there are several reasons that shows the difference in performance between other research paper and mine. First in that research, they did english sentences analysis which is more less complicated language than korean. Second, not enough data to train the model. because of overfitting, i can't proceed enough train with my data set. So from now, I will scrap more data set from other community websites and do other preprocessing for enhance performance.

References

[1] Bernhard E Boser, Isabelle M Guyon, andVladimir N Vapnik, "A Training Algorithm for Optimal Margin Classifiers," *Proc. The Fifth Annual Workshop on Computational Learning Theory*, pp. 144-152, 1992.

[2] 김현우, 이승룡, "모바일 텍스트의 감성분류를 위한 SVM 기반 음운 커널 기법," *정보과학회논문지 : 소프트웨어 및 응용* 제 40 권 제 6 호, 2013

[3] 이성욱, "스탠포드 감성 트리 말뭉치를 이용한 감성 분류 시스템," *Journal of the Korean Society of Marine Engineering*, Vol. 39, No. 3 pp. 274~279, 2015

[4] 임좌상, 김진만, ″한국어 트위터의 감정 분류를 위한 기계학습의 실증적 비교,″ *Journal of Korea Multimedia Society* Vol. 17, No. 2, pp. 232-239, February 2014

[5] 서상현, 김준태, ″딥러닝 기반 감성분석 연구동향,″ *한국멀티미디어학회지* 제 20 권 제 3 호 2016 년 9 월

[6] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, Jeff Dean, ″Distributed Representations of Words and Phrases and their Compositionality,″ *Advances in Neural Information Processing Systems* 26, Pages 3111-3119, 2013

[7] Rojas-Barahona LM., ″Deep learning for sentiment analysis,″ *Lang Linguist Compass*, pp.701-719, 2016

[8] Kim, Yoon. "Convolutional neural networks forsentence classification.", *arXiv preprint* arXiv:1408.5882, (2014).

[9] N Srivastava, G Hinton, A Krizhevsky, I Sutskever, R Salakhutdinov, ″Dropout: A Simple Way to Prevent Neural Networks from Overfitting,″ *Journal of Machine Learning Research 15*, Pages 1929-1958, 2014

[10] M Abadi, et al, ″TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed System,″ arXiv:1603.04467, 2016