Multi-class Sentiment Analysis using Deep Learning

Nishchay N. Trivedi
Department of Computer Science
Lakehead University
Student ID: 1106924

***Abstract*—In these days, feedback of any things has been necessary for every field. To analyze those phrases and categorizes them into positive or negative we need some tool. So to test such thing I have performed multi-class sentiment analysis with deep learning methods. In this paper I tried to predict the movie review for dataset from rotten tomatoes. I built a nonlinear regression model using Convolutional Neural Network(CNN) to predict the correct review for the given review. I used some pre-processing tasks to clean the data and used several convolution layers to get the best feature. To test the results I used several performance metrics and also calculated the loss.**

***Index Terms*—Sentiment Analysis, Convolutional Neural Network(CNN), Natural Language Processing(NLP)**

I. Introduction

Sentiment Analysis is the most popular application of NLP that studies the text and understand the meaning behind the text and categorize them as a positive or negative value, called polarity. From the experiments it is found that it works on the sentences that are more subjective-that has some meaning than objective-normal sentence without meaning. Mostly sentiment analysis is used in the terms of business like movie review, social media comments, product feedback etc. To attentively listen to customer's choice brands use this technique and provides services accordingly. There are different types of sentiment analysis like Fine-grained Sentiment Analysis, Emotion detection, Aspect-based Sentiment Analysis, Multilingual sentiment analysis. Among all these types, in this proposed paper I have worked on only Fine-grained Sentiment Analysis. As movie review dataset depends on polarity categories. Mostly there are five categories of polarity with their value shown in Table 1.

Sentiment Review Category	Value
Very Positive	5
Positive	4
Neutral	3
Negative	2
Very negative	1

TABLE I
Sentiment Review Category and its according value

CNN is used the most in feature extraction especially in Computer Vision(CV) field where image is fed in to CNN model and its features are extracted using some convolution layers followed by some pooling layers. For such cases 2D CNN is used. But what if we want to find the patterns from small segment of data like text data or tabular data? The solution of that is 1D CNN. It can also be used in the analysis of signal data like audio signals. So in my case as I am analyzing text data I have used 1D CNN. To understand the CNN architecture, we need to go through some basic layers used to form the architecture. Using these basic layers we can define the CNN architecture. Those are explained as below:

A. Convolutional layers

Initially input is given to convolutional layer where filters are used to extract features along with kernel(Based on the input shape). These layers are followed by activation function-that are used to get the output in range. some of the activation functions are Rectified Linear Unit(ReLU), Softmax, Tanh etc.

B. Pooling layers

They are similar to convolutional layers, but they perform specific tasks. There are different types of pooling layers describes as below:

- Sum Pooling
- Avg Pooling
- Max Pooling

C. Fully Connected Layers

Fully Connected Layers are placed before output and are used to flatten output. They are used to generate the final result. Even at this layer activation functions are used and that depends on the action, if aggregation is required use ReLU and for classification Softmax is used.

II. Literature Review

As sentiment analysis is being the most used in business terms, lots of work has been done in this area. As the main purpose of sentiment analysis is to understand the text and classify them sentiment categories(as shown in table-1), Stone, Philip J., Dexter C. Dunphy, and Marshall S. Smith proposed paper in the year of 1996 for the content analysis with respect to computer approach[1]. In later years to analyze the movie review Pang[2] worked with some different machine learning techniques. Though this proposed work was only at document level. After three years they worked more on movie review analysis and they attempt to predict the results from polarity of positive and negative to three-four star scale[3]. As of in this time neutral class was not considered in sentiment polarity category, later in 2006, The importance of neutral examples was considered [4][5]. As discussed in introduction subjective sentences are used more than objective. The research works

have been done in this field[6], again Pang came to know that by removing objective sentences document helps in sentiment analysis[7].

In the world of social media, sometimes it becomes necessary to analyze the comments and tweets made by users[8]. After that rise of anonymous social media platforms came[9]. Even for the smaller text that maybe problem in sentiment analysis but can be used for political use. The method related to such case proposed in 2010 that predicted the election win from twits collected from Twitter[10].

III. Proposed Method

A. Dataset

The dataset is used to perform this experiment is Rotten Tomatoes movie review dataset originally collected in by Pang[11]. This dataset contains 156060 instances and the features are PhraseId, SentenceId, Phrase and Sentiment. The class labels falls in five categoris ranges from 1-5 from very positive to very negative(table-1). Phrase ID is the unique number or index number arrange in incremental order. Sentence ID is the number of different class of sentence. Phrase sentiment is the sequence of words which can be used to train model. The first 5 instances of this dataset is shown in Table-2. This dataset is split into 70:30 training and testing data so I have 109242 as traing and 46818 as testing data. From the first look of dataset it is clearly seen that dataset is imbalanced as it has more phrases of neutral and very less very positive and very negative. This class imbalance can be seen in Figure-1.

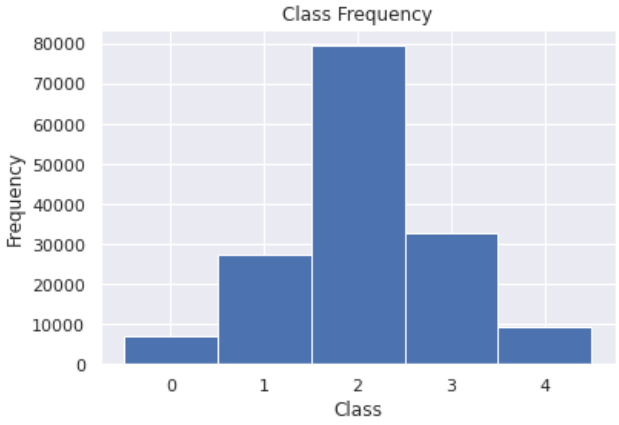


Fig. 1. Class distribution

B. Tools

To perform this task I used several tools. All the coding part is done in python programming. To build the CNN model, I used python keras library. Using pandas read the data from csv file. To plot the graphs matplotlib is used. For mathematical expression numpy is used. For train, test data split sklearn is used.

	PhraseId	SentenceId	Phrase	Sentiment
0	99516	5218	dysfunctionally	2
1	111952	5942	essentially ruined – or , rather	1
2	122259	6553	connect and express	2
3	14969	643	cold movie	1
4	55848	2792	controlling his crew	2

TABLE II
First 5 instances of dataset

C. Pre-processing

Whenever dataset is collected, most of the times it contains some noisy data or unwanted text. So in order to get the good results it becomes necessary to remove such kind of data. In NLP terms, we need to remove some words or make the word to root form or convert word to vector. So I have performed this pre-processing techniques and describes as follows:

1) Remove stop-words: Before training woth we need to remove some words, stop words are filtered out before proceeding with training task. For each tools, list of this stop words are different. But I found Natural Language Toolkit(NLTK)'s stop words as efficient and used in this task. Words like the, is, at, which, and on are removed that are not needed to understand the phrase and are removed from text.

2) Remove punctuation: After removing stop words it becomes necessary to remove punctuation as well because they do not contain any relevant information. To remove punctuation we have to manually perform this task. Punctuations removed in this task are: ? : ! . , ; ' " - ()

3) Stemming and Lemmatization: Stemming and Lemmatization are basic text normalization techniques in NLP used to prepare text, words, and documents for further processing. Stemming is the process of reducing inflection in words to their root forms. Stem (root) is the part of the word to which we add inflectional affixes such as (-ed, -ize, -s, -de, mis). So stemming a word or sentence may result in words that are not actual words. e.g. CONNECTED $\rightarrow$ CONNECT. Lemmatization, unlike Stemming, reduces the inflected words properly ensuring that the root word belongs to the language. In Lemmatization root word is called Lemma. e.g. HAS $\rightarrow$ HAVE

D. Text Vectorization

Since the beginning of the brief history of Natural Language Processing (NLP), there has been the need to transform text into something a machine can understand. That is, transforming text into a meaningful vector. Word Embeddings or Word vectorization is a methodology in NLP to map words or phrases from vocabulary to a corresponding vector of real numbers which used to find word predictions, word similarities/semantics. In particular this task I used TF-IDF. TF-IDF stands for Term Frequency-Inverse Document Frequency which basically tells importance of the word in the corpus or dataset. TF-IDF contain two concept Term Frequency(TF)-Term Frequency is defined as how frequently the word appear

in the document or corpus(Equation-1) and Inverse Document Frequency(IDF)-Inverse Document frequency is another concept which is used for finding out importance of the word(Equation-2).

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}} \tag{1}$$

$$idf(w) = \log(\frac{N}{df_t}) \tag{2}$$

E. CNN model

As shown in the introduction section, I implemented basic CNN model with convolutional layers, pooling layers, activation functions and fully connected layers. To build this architecture I used keras library. I used non-linear sequential model to build the architecture. Initially I used 2 convolutional layers followed by relu activation functions. After that I used max-pooling. Later I added the same 2 convolutional layers with relu activation and now the time I used global average pooling layer. Followed by which I used dropout of 0.5 and at the last dense layer is used with softmax activation and output is calculated. The architecture is shown in Table-3.

Layer (type)	Output Shape	Param #
conv1d_1 (Conv1D)	(None, 1991, 128)	1408
conv1d_2 (Conv1D)	(None, 1982, 128)	163968
max_pooling1d_1 (MaxPooling1	(None, 660, 128)	0
conv1d_3 (Conv1D)	(None, 651, 256)	327936
conv1d_4 (Conv1D)	(None, 642, 256)	655616
global_average_pooling1d_1	(None, 256)	0
dropout_1 (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 5)	1285
Total params: 1,150,213 Trainable params: 1,150,213 Non-trainable params: 0		

TABLE III
CNN ARCHITECTURE

F. Inference Time

In Deep Learning and NLP, inference time is the term used to calculate the time taken by model to execute. Keras itself calculates the time per epoch and give the time. Using per epoch time we can sum them all to get the time taken by model to train.

G. Save Model

After creating the model, it is good practice to save the model. The reason behind this is that we do not need to train the model every time because it takes lot of time. So using keras we can save the model and later we can load that model and predict the result for test data.

Batch_size	Optimizer	Accuracy	f1_score	Precision	Recall
64	Adam	0.8154	0.4733	0.5511	0.4265
64	RMSProp	0.7953	0.4537	0.5208	0.4073

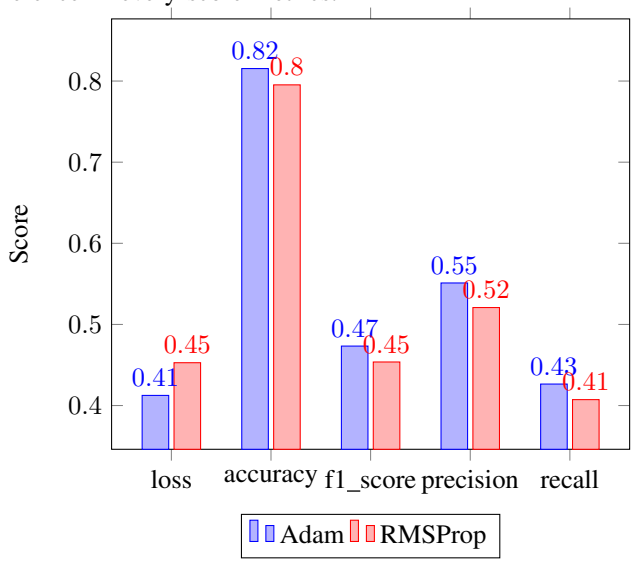
TABLE IV
COMPARISON OF TWO DIFFERENT OPTIMIZERS

IV. RESULT

After building model, I did some hyper parameter tuning to get the best parameter can be used to train the model. After several tests I found the parameter as follows:

- Batch_size: 64
- #epochs: 50
- Optimizer: Adam
- #Convolution_layers: 4

Using this parameter highest accuracy I achieved is: 81.54% Table-4 shows the comparison of two different optimizers. As we can see from the table 4 and graph there is not much difference in every score metrics.



V. FUTURE SCOPE

This was just basic sentiment analysis using basic architecture of CNN. further the same task can be achieved using different networks like Recurrent Neural Network or Long-Short Term Memory(LSTM). Till now sentiment analysis is limited to social media comment or movie reviews. In future it is going in much deep, far past the surface of the number of likes, comments and shares, and aim to reach, and truly understand, the significance of social media interactions and what they tell us about the consumers behind the screens.

There are still challenges in Detection of spam and fake reviews as on web we find both authentic and spam contents, that can be done by identifying duplicates, by detecting outliers and by considering reputation of reviewer. Also, there is a limitation in classification filtering while determining most popular thought or concept

VI. Conclusion

In this task, I performed the sentiment analysis using CNN on rotten tomatoes dataset. I have used several pre-processing techniques. Then converted the txt into vector using TF-TDF. Then developed the CNN model using several layers and some parameters. Finally, calculated the result using CrossEntropy-Loss and different performance metrics. The highest accuracy achieved is: 81.54%.

VII. Acknowledgment

I would like to thank very much to professor Dr. Thangarajah Akilan for providing the thorough knowledge of CNN and sentiment analysis and all NLP terms. I thank him again for providing me an opportunity to put those concepts in application by giving this qualitative assignment. I would also like to thank TAs Andrew and Punardeep for providing me the way to complete this assignment and their constant support.

References

[1] Stone, Philip J., Dexter C. Dunphy, and Marshall S. Smith. "The general inquirer: A computer approach to content analysis." MIT Press, Cambridge, MA (1966).

[2] Pang, Bo; Lee, Lillian; Vaithyanathan, Shivakumar (2002). "Thumbs up? Sentiment Classification using Machine Learning Techniques". Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 7986.

[3] Pang, Bo; Lee, Lillian (2005). "Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales". Proceedings of the Association for Computational Linguistics (ACL). pp. 115124.

[4] Koppel, Moshe; Schler, Jonathan (2006). "The Importance of Neutral Examples for Learning Sentiment". Computational Intelligence 22. pp. 100109. CiteSeerX 10.1.1.84.9735

[5] Vryniotis, Vasilis . "The importance of Neutral Class in Sentiment Analysis",2013

[6] Pang, Bo; Lee, Lillian (2004). "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts". Proceedings of the Association for Computational Linguistics (ACL). pp. 271278.

[7] Pang, Bo; Lee, Lillian (2004). "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts". Proceedings of the Association for Computational Linguistics (ACL). pp. 271278.

[8] Wright, Alex. "Mining the Web for Feelings, Not Facts", New York Times, 2009-08-23. Retrieved on 2009-10-01.

[9] "Sentiment Analysis on Reddit". 2014-09-30. Retrieved 10 October 2014.

[10] Tumasjan, Andranik; O.Sprenger, Timm; G.Sandner, Philipp; M.Welpe, Isabell (2010). "Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment". "Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media"

[11] Pang and L. Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In ACL, pages 115124.

VIII. Appendix

A. Pre-processing

```
for l in range(len(documents)):
  label = documents[l][1]
  tmpReview = []
  for w in documents[l][0]:
    newWord = w
    if remove_stopwords and (w in stopwords_en):
      continue
    if removePuncs and (w in punctuations):
      continue
    if useStemming:
      newWord = lancaster.stem(newWord)
    if useLemma:
      newWord = wordnet_lemmatizer.lemmatize(
                                        newWord)
    tmpReview.append(newWord)
  documents[l] = (tmpReview, label)
  documents[l] = (' '.join(tmpReview), label)
```

B. CNN Model

```
model = Sequential()
model.add(Conv1D(128, 10,
activation='relu',
                        input_shape=(2000,1)))
model.add(Conv1D(128, 10, activation='relu'))
model.add(MaxPooling1D(pool_size=3))
model.add(Conv1D(256, 10, activation='relu',
                       input_shape=(2000,1)))
model.add(Conv1D(256, 10, activation='relu'))
model.add(GlobalAveragePooling1D())
model.add(Dropout(rate = 0.50))
model.add(Dense(num_classes,
                activation='softmax'))
```