Multi-class Sentiment Analysis Using Deep Learning

Kushal Patel
Dept. of Engineering
Lakehead University
Thunder Bay, Canada
kpatel57@lakeheadu.ca

***Abstract*—Sentiment analysis is the interpretation and classification of emotions (positive, negative, and neutral) using text analysis techniques within the data. It helps companies to detect customer interest in online interactions and feedback against items, brands or services. Sentiment analysis of Rotten Tomatoes film reviews has to deal with challenges, such as informal abbreviated language, internet jargons, emoticons, hashtags that do not appear in traditional text papers. Sentiment analysis in this paper is based on one-dimensional convolution neural network (CNN) with the combination of the activation function Relu and Softmax. Experiments were performed on a noisy dataset and the obtained results show that CNN produces superior or at least comparable results to state-of-the-art machine learning techniques.**

***Index Terms*—sentiment analysis, neural networks, convolution neural network, machine learning.**

I. INTRODUCTION

Given the growing ubiquity of social media, it is crucial that such vast volumes of arbitrary opinionated data are immediately capable of making sense. Sentiment analysis refers to the use of natural language processing, computational linguistics, and machine learning methods to evaluate the meaning of feelings from the written language. This is also known as opinion mining, as it illustrates the perceptions, behaviours and emotions of people with regard to products, facilities, organisations, individuals, events or topics. Since it can track a public opinion on a particular subject, it is widely used in market intelligence to benchmark products, services, and customer attitudes, in politics to estimate election results, or to assess the credibility of the politician, in sociology, and in psychology.

Work on sentiment analysis moves in two different directions: the approaches based on the lexicon and the approaches focused on machine learning. Approaches based on lexicons depend on predefined term lexicons, e.g. SentiWordNet paired with feeling criteria to assess the meaning of the sentence or the text under scrutiny. Its key benefit is that they do not need training data and can be used in various fields. The big downside, though, is a nite of lexicon words, which can become a issue in competitive contexts like Twitter or Facebook, where new phrases, abbreviations and malformed words are continually appearing. Alternatively, the methods based on machine learning may be used to train a feeling classier on a wide range of labelled cases, which typically contributes to higher precision but requires manual annotation.

Sentiment analysis is considered as a classification problem, either at the document or sentence level. This classification is usually binary or multi-class. In the multi-class Every training point belongs to different classes at one of the N. The purpose is to create a function which will correctly predict the class to which the new point belongs, provided a new data point. In this study, multi-class sentiment analysis of Rotten Tomatoes movie reviews is performed using CNN after data cleaning operations on a raw dataset. Model is eveluated using following metrics - Accuracy, Recall, Precision, and Figure-of-Merit (f-1) score.

II. BACKGROUND

Hannah Kim et al. [1] proposed convolution neural network to cope with the problem of sentiment analysis. The authors concluded that the consecutive convolutional layers contributed to better performance on relatively long text. For binary classification and ternary classification proposed model achieved 81 percentage and 68 percentage accuracies.

Christopher Meek et al. [2] proposed a semantic parsing framework for single-relation questions. the model is trained using CNN with letter tri-grams vectors. the model achieves higher precision on the question answer task than the previous work and the model go beyond bag of words representations.

Ma´ıra Gatti et al. [3] present a new deep neural network architecture that jointly uses character-level, wordlevel and sentence-level representations to perform sentiment analysis. To extract character to sentence level features authors used CNN and for the sentiment analysis of the sentences they suggested feed-forward neural network architecture.

Li Dong et al. [4] proposed Adaptive Recursive Neural Network (AdaRNN) for the target-dependent Twitter sentiment classification. AdaRNN uses more than one composition method and picks it according to meaning and language tags. Proposed model converts the given tweet into dependency tree for the interested target. The AdaRNN then learns how to spread word feelings to the target node in an adaptive way. AdaRNN enables the dissemination of emotions to be open to both linguistic and semantic types, using various compositions.

III. DATASET

The labeled data set consists of 156060 Rotten Tomatoes movie reviews, specially selected for sentiment analysis. Dataset consists four columns of PhrseID, SetimentID, Phrase and Sentiment. The sentiment of reviews is multi-class, meaning data is labeled as negative, somewhat negative, neutral, somewhat positive, or positive for numbers 0,1,2,3 and 4 respectively. The dataset is splitted into 70:30 ratio as 70 percentage of data is for training purpose and rest of the data is for testing of the model. Fig. 1 shows the visual plotting of the labeled data that represents the number of entries for each sentiment. Table 1 shows the information of the dataset.

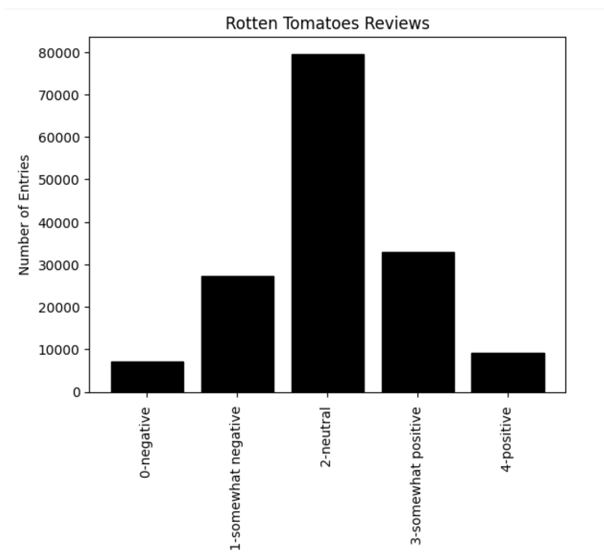

Fig. 1. Rotten Tomatoes Movie Review Dataset

TABLE I
DATASET

PhraseId	SentenceId	Phrase	Sentiment
1	1	A series of escapades demonstrating...	1
2	1	A series of escapades...	2
3	1	A series	2
4	1	A	2
5	1	series	2

IV. DATA PRE-PROCESSING

I have tried several basic sentiment analysis methods as described below. For all methods, pre-processing steps have been performed to clean up the data. This includes removing of stopwords, removing of punctuations, tokeniztion, stemming and lemmatization. After cleaning of data different vectorizers have been tested and TF-IDF(term frequency–inverse document frequency) is selected from different vectorizers after comparing results. Output of the TF-IDF vectorizer is given as an input to the CNN model for training and testing of a dataset.

A. Data Cleaning

Stop Words One of the ways to pre-process the data is the filtration of useless data. A stop word is a widely used word (such as ”the,” ”a,” ”an,” ”in”) that is programmed to be overlooked by a search engine, both when indexing search entries and when retrieving them as a result of a search query. We would not want these terms to take up space in our database, or take up precious time for processing. it can be easily deleted by storing a list of terms that you find to be stop terms. The NLTK(Natural Language Toolkit) library has, a list of stopwords stored for different languages.

Punctuation is a collection of marks which regulate and explain the meanings of the various texts. Punctuation is aimed at clarifying the meanings of texts by connecting or separating sentences, phrases or clauses. Punctuation influences interpretation and is critical for understanding the sense of form. Deleting punctuation decreases the efficiency of the follow-on semantic parsing.

Tokenization is the method of tokenizing a string or breaking it into a set of tokens. I used tokenize() to split a sentence into words. The word tokenization output can be converted to Data Frame for improved interpretation of texts in machine learning applications. Machine learning algorithms need to train numerical data to make a prediction. Term tokenization is an essential feature of the text (string) for the transfer of numeric data.

Stemming algorithms operate by cutting off the end or start of the word, taking into account a list of specific prefixes and suffixes contained in an inflected word. This indiscriminate cutting may often be effective, but not always, and this is why I affirm that this method faces some limitations. Table 2 represents the example of how stemming works.

TABLE II
EXAMPLE OF STEMMING

Form	**Suffix**	**Stem**
Studies	-es	Studi
Studying	-ing	Study

Lemmatization, on the other hand, takes into account the morphological interpretation of the words. In order to do so, it is important to provide detailed dictionaries from which the algorithm will try to connect the form back to its lemma. Another significant distinction to highlight is that a lemma

TABLE III
EXAMPLE OF LEMMATIZATION

Form	**Morphological Information**	**Lemma**
Studies	Third person, singular number, present tense of the verb **study**	Study
Studying	Gerund of the verb **study**	Study

is the root form of all the modes of inflexion, while a stem is not. For this function standard dictionaries are collections of lemmas and not stems. Table 3 shows the accuracy of the lemmatization.

B. Feature Extraction

We realise that the bulk of the programme has a huge number of databases to handle. A non-computationally-optimal function in the algorithm can then become a huge bottleneck which will result in a cycle that can take ages to run. We must use vectorization to make sure the code is computationally efficient. Vectorization is done by using a loop to speed up the Python code. Such method can help to effectively reduce application run time. Bag of words, TF-IDF, Word2vec and count vectorizer are different ways for feature extraction of the raw data.

Bag of words (BOW) is one of the method to extract the features of the document. These features can be used to train algorithms concerning machine learning. It generates a vocabulary of all the special terms that appear in all of the training materials.

TF-IDF calculates a weight which is often used in collecting details and in text mining. This weight is a statistical calculation used in a series or corpus to determine how important a term is to a text. The importance of the word increases as to the number of times a word appears in the in the text which is offset by the word frequency in the corpus (data-set).

Word2vec is a two layer neural network that handles words by "vectorizing". Its input is a corpus of text and its output is a set of vectors: function vectors representing terms in the corpus. Although Word2vec is not a deep neural network, it transforms text into a numerical form profoundly understood by neural networks.

The CountVectorizer provides a simple means of both tokenizing a set of text documents and creating a vocabulary of known words but also encoding new documents using that vocabulary. In the proposed model I have used this method for feature extraction of the rotten tomatoes movie review dataset.

V. PROPOSED MODEL

The model used is sequential CNN model. Sequential model allows one to create model layer by layer. A CNN works well to define basic trends in the data that can then create more complicated trends in higher layers. A 1D CNN is very successful when you intend to extract interesting features from the total data set's shorter (fixed-length) segments and where the position of the feature inside the segment is not strongly important. Then after maxpooling is done to down-sample an input representation (image, hidden-layer output matrix, etc.), reducing its dimensionality and allow decisions about features. Afterwards dropout of specific nodes are either dropped out of the net with probability 1-p at each training point, or held with probability p such that a decreased network is left; incoming and outgoing edges to a dropped-out node are both eliminated. Once you have received the pooled featured map, the next move is to flatten it. Flattening involves converting the entire matrix of the pooled function map into a single column which is then fed to the computing neural network.

$$output = activation(dot(input, kernel) + bias) \quad (1)$$

A dense layer in a neural network is simply a normal layer of neurons. Each neuron receives feedback from all the neurons in the previous layer and is thus densely related. The layer has a weight matrix W, a bias vector b, and the activations of previous layer a. As shown in (1) activation is the element-wise activation function passed as the activation statement, kernel is a layer-created weights matrix and bias is a layer-created bias vector. For this model I have used ReLu and Softmax activation layers as presented in Listing 1.

```
model = Sequential()
model.add(Conv1D(filters=64, kernel_size=1,
                 activation='relu',
                 input_shape=(2500,1)))
model.add(Conv1D(filters=64, kernel_size=1,
                         activation='relu'))
model.add(Conv1D(filters=32, kernel_size=1,
                         activation='relu'))
model.add(MaxPooling1D(pool_size=1))
model.add(Dropout(rate = 0.1))
model.add(Flatten())
model.add(Dense(32, activation='relu'))
model.add(Dense(num_classes, activation='softmax'))
print(model.summary())
```

Listing 1. construction of sequential model

VI. EXPERIMENTAL ANALYSIS

The goal of the experiment is to perform a sentiment analysis of the movie review dataset by testing the model and the purpose is to achieve higher accuracy of the model. For the model I have tried different combination of optimizers, activation functions and feature extraction techniques. I have found the best results for Count vectorizer as a feature extractor, Adam as optimizer and combination of Relu and Softmax as an activation functions. The model is trained for twenty epochs and the performance is evaluated based on the

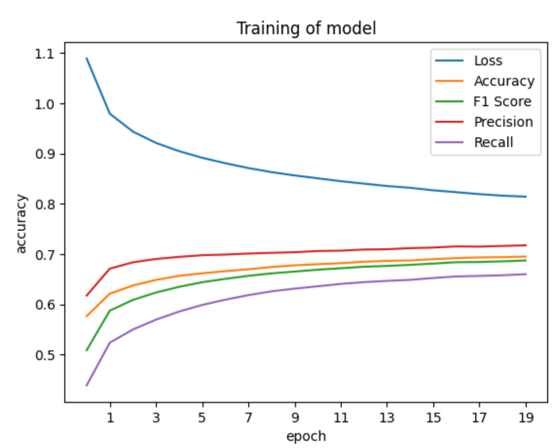

Fig. 2. training progress of the model

metrics such as accuracy of the model, recall, precision and figure-of-merit (f-1 score).

Fig. 2 shows the timeline of the evaluation metrics by each epochs as it shows loss decreased from 1.0880 to 0.8063 at the end of epoch 20. Accuracy increased from 0.5758 to 0.6972 after completion of the training of the model.

```
Test loss: 1.0208489850307958
Test accuracy: 0.6348626596760916
F1 Score: 0.6219943957426776
Precision: 0.655480844026099
Recall: 0.5922508437129667
```

Listing 2. test results of the model

Test results of the model is shown in listing 2 as 63 percentage accuracy of the model.

VII. CONCLUSION

A convolution neural network is proposed for a task of sentiment analysis of Rotten tomatoes movie review giving comparable results, or in some cases even outperforming state-of-the-art machine learning techniques. This proves that CNNs can be a promising tool for determining sentiment polarity. Firstly, data cleaning operation is performed on the dataset so that unnecessary noisy data is removed. It is interesting to know that TF-IDF, Word2vec, count vectorizer are different options available for the vectorization and count vectorizer is selected for the model. In summary, results indicate that it is not possible to make a straight choice of the best machine learning method for the task of sentiment analysis and around 63 percentage of the result that model produces is accurate.

REFERENCES

[1] Hannah Kim and Young-Seob Jeong, “Sentiment Classification Using Convolutional Neural Networks,” Appl. Sci. 2019, 9, 2347; doi:10.3390/app9112347, 2019.

[2] Wen-tau Yih, Xiaodong He and Christopher Meek, “Semantic Parsing for Single-Relation Question Answering,” Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 643–648.

[3] C´ıcero Nogueira dos Santos and Ma´ıra Gatti, “Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts,” Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers, pages 69–78, Dublin, Ireland, August 23-29 2014.

[4] Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou and Ke Xu, “Adaptive Recursive Neural Network for Target-dependent Twitter Sentiment Classification,” Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 49–54.

[5] Rotten tomatoes movie review, https://raw.githubusercontent.com /cacoderquan/Sentiment-Analysis-on-the-Rotten-Tomatoes-movie-review-dataset /master/train.tsv

[6] Lecture notes, Dr. T akilan, Lakehead University, 2020.