Natural Language Processing Assignment-2

Parth Soni(1115755)
Department of Computer Science
Lakehead University
Thunder Bay, Canada

***Abstract*— Sentiment Analysis is a major subject in machine learning which aims to extract subjective information from the textual reviews. This paper is based on the implementation of Convolutional Neural Network (CNN)-based solution for the problem of text-based movie review multi-class sentiment analysis on Rotten Tomatoes movie reviews dataset. The model is trained within the scope of Convolutional and dense layers and the associated operations, like pooling and non-linear activation Functions and evaluated based on accuracy , precision , recall and F1 score**

I. Introduction

The popularity of social media and other online platform increased in last decade and due to this massive amount of data is being created on various social networking sites as Facebook, Instagram and Twitter in form of tweets, blogs and comments. However, movie reviews are an important way to judge the performance of the movie. A text movie review tells us about the strengths and limitations of the movie and a more in-depth analysis of a movie review will tell us if the film usually meets the reviewer's expectations.

Sentiment Analysis is closely connected with natural language processing and text mining. A fundamental task in sentiment analysis is classifying the polarity of a given text at the document or sentence. It mainly focuses on whether the given review, comment or opinion in a document or sentence is positive, negative or neutral and sometimes it checks beyond the polarity and gives the emotional state of the reviewer such as happy, sad or angry.

In this paper the major steps to do sentiment analysis are data preprocessing, feature extraction and prepare and classify the model. I used Rotten Tomatoes movie review dataset for my project. For the implementation I used Google colab which allows anybody to write and execute arbitrary python code through the browser and is especially well suited to machine learning and data analysis. Data preprocessing is very important step for any project because we can remove unnecessary or redundant data and machine can get only important data for processing. I have performed different preprocessing steps which are Lowercasing, Stemming, Lemmatization, Stopword Removal, Tokenization. Feature extraction is concerned with extracting the useful features from the text. I have used Term Frequency — Inverse Data Frequency (TF-IDF) for feature extraction. The most important step in project is to build a model and for that I have used CNN-1D with keras. I tried different model with different optimizers, batch size and epochs to get the best result in term of accuracy, recall, precision and F1 score.

II. Literature Review

I have done literature review on paper "Predicting Sentiment from Rotten Tomatoes Movie Reviews "published by Jean Y. Wu and Yuanyuan Pao in year 2012[1]. The aim of the project is to experiment with different machine learning algorithms to predict the sentiment of unseen reviews from the improved corpus. They selected the improved version of the Rotten Tomatoes movie reviews dataset and compared their result with the original version of the dataset and they also used collected subjective human sentiment ratings thru the Amazon Mechanical Turk (AMT) platform. To obtain best version of dataset with an improved quality, first cleaned up the dataset by removing all the residual HTML tags and restoring the original capitalization of the words. The dataset is split into a ratio of 7:1:2 for training, cross-validation and test sets. They used Multinomial Naive Bayes (MNB) and Support Vector Machine (SVM) algorithm for training and testing. They implemented an MNB-Event model and used the LIBLINEAR package in Matlab to train and test SVMs.

Their neural network consists of three layers - the input layer, the hidden layer, and the output layer. For our input layer, they feed in the word vector representations for various n-grams. Instead of using a constant length window, they have used various length n-grams ranging from one-word unigrams to the longest sentences in training set. As a comparison with original version of dataset they concluded that for achieving a performance better with MNB and SVM models, current 3-layer neural network is insufficient. The result shows network performed well on unigrams and bigrams but suffered for most larger n-grams. After the error analysis they find that not taking the constituents of the larger n-grams into account is a major reason for dropping down the accuracy for n-gram as it does not consider how certain words affect their neighboring words. One of the idea for increasing the performance and accuracy is use recursive neural network rather than using 3-layer neural network.

III. Dataset

For the implementation I choose Rotten Tomatoes movie reviews dataset. Rotten Tomatoes is one of the most popular film websites, which combines movie information, critic reviews and user's reviews. The dataset has been divided to 7:3 ratio for the training and test set for the purpose of benchmarking, but the sentences have been shuffled from their original order. Table-1 shows the first five rows of the dataset. The dataset contains Phrase Id, Sentence Id, Phrase

TABLE I
FIRST FIVE ROWS OF 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

and Sentiment. Each Sentence in the dataset has been parsed into many phrases by the Stanford parser. Each phrase has a phrase Id. Each sentence has a sentence Id. Phrases that are repeated are only included once in the data.

The sentiment label which are used in the dataset are

- 0-negative
- 1-somewhat negative
- 2-neutral
- 3- somewhat positive
- 4- positive

TABLE II
SENTIMENT COUNT

sentiment	value
0	7072
1	27273
2	79582
3	32927
4	9206

The Training data set consists of 156059 phrases and their sentiment values are distributed as shown in Table-2. From the table it is clear that majority of the phrases has sentiment value of 2 which corresponds to neutral sentiment. I used pandas library to manipulate with data and dataset and used numpy to convert the data into array.

IV. DATA PREPROCESSING AND FEATURE EXTRACTION

Data preprocessing is the process of preparing data for analysis by removing or modifying data. Some data is usually not necessary or helpful when it comes to analyzing data because it may hinder the process or provide inaccurate results[2]. There are many different types of preprocessing technique which can help to increase the accuracy of the model. For my project I have used Lowercasing, Stemming, Lemmatization, Stopword Removal, Tokenization.

A. Lowercasing

Lowercasing is one of the simplest and most effective form of text preprocessing. It is applicable to most text mining and NLP problems and can help in cases where dataset is very large and significantly helps with consistency of expected output. There are some cases where the same word is written in different type like "Parth"," PARTH" and "parth" but the meaning of the word is same so after using lowercasing all word will convert in "parth". Lowercasing makes them identical, causing the classifier to lose important predictive features.

B. Stemming

Stemming is the process of reducing inflection in words to their root form. The "root" in this case may not be a real root word, but just a canonical form of the original word. Stemming uses a crude heuristic process that chops off the ends of words in the hope of correctly transforming words into its root form. So, the words "trouble", "troubled" and "troubles" converted into trouble. This is how Stemming is useful for dealing with sparsity issues as well as standardizing vocabulary.

C. Lemmatization

Lemmatisation is the process of reducing a group of words into their lemma or dictionary form. It considers things like Parts of Speech, the meaning of the word in the sentence and the meaning of the word in the nearby sentences before reducing the word to its lemma. For example, Lemmatization will map "better" to good. Lemmatization is very similar to stemming, where the goal is to remove inflections and map a word to its root form. The only difference is that, lemmatization tries to do it the proper way. Lemmatization provides no significant benefit over stemming for search and text classification purposes.

D. Stopword Removal

Stop words are a set of commonly used words in a language. Words like "of, are, the, it, is" are some examples of stopwords. The intuition behind removing stopwords is we can focus on the important words instead.

E. Tokenization

Tokenization is just the process of splitting a sentence into words. Sentences can be split into individual words and punctuation through a similar process. It is important because by this, words are identified, demarcated and classified.

F. Feature Extraction

Machine learning usually deal with numbers, and natural language is, well, text. So, we need to transform that text into numbers which is known as text vectorization. Simple method of feature extraction with text data which is currently used is "Term Frequency — Inverse Data Frequency (TF-IDF). TF-IDF was invented for document search and information retrieval. It works by increasing proportionally to the number of times a word appears in a document but is offset by the number of documents that contain the word.

Term Frequency (tf): gives the frequency of the word in each document in the corpus. It is the ratio of number of times the word appears in a document compared to the total number of words in that document. It increases as the number of occurrences of that word within the document increases. Each document has its own tf.

Inverse Data Frequency (idf): used to calculate the weight of rare words across all documents in the corpus. The words that occur rarely in the corpus have a high IDF score.

V. Proposed Method

The most important step in this project is to train a model which gives the best accuracy with the given dataset. I used CNN-1D with keras. Keras is high Level Application Programmable Interface (API) that allow to build the Deep learning or Machine Learning models easily without detail understanding about internal architecture how the ML algorithm works internally. The input data is in sequential form, so I have used sequential model. CNN indicates that the network employs a mathematical operation called convolution. Convolution is a specialized kind of linear operation. Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers. AS shown in figure-1 ,A CNN consists of an input and an output layer, as well as multiple hidden layers. Hidden layers are convolutional layer, max pooling layer, flatten layer and dence layer.

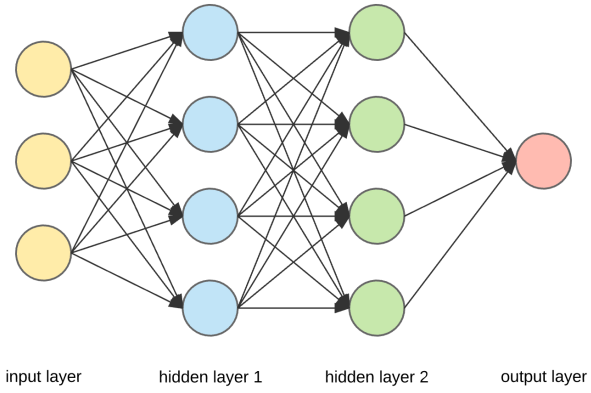

Fig. 1. layers of convolutional neural network

CNN apply a filter to an input to create a feature map that summarizes the presence of detected features in the input. Filter is a set of learnable weights which are learned using the backpropagation algorithm[3]. A kernel in a CNN is basically a small window, that has specific values. As shown in label I have used 64 filters and kernel size is 3. Pooling is basically "downscaling" the data obtained from the previous layers. Most common pooling methods are max pooling and average pooling. I have used max pooling because it helps in extracting low-level features like edges and points. Dropout is a regularization technique for neural network models. It is simple way to Prevent Neural Networks from Overfitting. Flatten layer is used for converting the data into a 1-dimensional array for inputting it to the next layer. Dense layer is the regular deeply connected neural network layer. It is most common and frequently used layer. Dense layer does the below operation on the input and return the output. The activation function is responsible for transforming the summed weighted input from the node into the activation of the node or output for that input. I have used Relu and softmax activation function.I have used 4 convolutional layers and 2 max pooling layers.

```
model = Sequential()
model.add(Conv1D(filters=64, kernel_size=3,
            activation='relu',
            input_shape=(2500,1)))


model.add(Conv1D(filters=64, kernel_size=3,
    activation='relu'))
model.add(MaxPooling1D(pool_size=1))

model.add(Conv1D(filters=64, kernel_size=3,
    activation='relu'))
model.add(Conv1D(filters=64, kernel_size=3,
    activation='relu'))
model.add(MaxPooling1D(pool_size=1))

model.add(Dropout(rate = 0.25))
model.add(Flatten())

model.add(Dense(10, activation='relu'))
model.add(Dense(num_classes,
    activation='softmax'))
```

The other important hyper parameter are batch size , learning rate and epochs. Batch size is the number of examples that are fed to the algorithm at a time. During training an optimization algorithm computes the average cost over a batch then runs backpropagation to update the weights. Learning rate controls how much we are adjusting the weights of our network with respect the loss gradient. The lower the value, the slower we travel along the downward slope. The number of epoch will decide- how many times we will change the weights of the network. As the number of epochs increases, the same number of times weights are changed in the neural network and the boundary goes from underfitting to optimal to overfitting. For train the model I have used learning rate 1e-3 and epoch I choose Is 50.

VI. Experimental results and analysis

TABLE III
Experimental results and analysis

Optimizer	Accuracy	F1 score	Precision	Recall
SGD	0.49	0.49	0.49	0.49
Adam	0.63	0.59	0.61	0.61
RMSProp	0.40	0.32	0.50	0.54
Adamax	0.63	0.57	0.60	0.59

After initializing the model, the next step would be to train the model and analyse result based on different criteria such as accuracy, recall, precision and F1 score. I have tried different activation function with different optimizers, learning rate an epochs. As shown in table-3, I have tried different optimizers Adam, SGD , Adamax and RMSProp with their default learning rate and 25 epochs. I got very

low accuracy with tanh and sigmoid activation function around 15% and 30% respectively. The problem with the two functions are only really sensitive to changes around their mid-point of their input, such as 0.5 for sigmoid and 0.0 for tanh. By using Relu and softmax I got around 63% accuracy for Adam and Adamax which is best accuracy I got with my model. If considering all the four parameters accuracy, recall, precision and F1 score Adam optimizer gives the best result for my model.

REFERENCES

[1] Jean Y. Wu Yuanyuan Pao.(2012).Predicting Sentiment from Rotten Tomatoes Movie Reviews doi: 10.1109/ICEngTechnol.2017.8308186

[2] Gurusamy, Vairaprakash Kannan, Subbu. (2014). Preprocessing Techniques for Text Mining.

[3] O'Shea, Keiron Nash, Ryan. (u2015). An Introduction to Convolutional Neural Networks

[4] Lecture notes, Dr. Thangarajan Akilan, Lakehead University, 2020.