Sentiment Analysis for a Mulit-class Model

Vishnu Sudheer Gannamani
Student Id: 1110200
Computer Science Engineering
Lakehead University, Thunder Bay
vgannama@lakeheadu.ca

I. Abstarct

This document contains the things that are involved in the Assignment-2 for the course Natural Language Processing. Convolutional Neural Network have gained a very good popularity in the fields of Artificial Intelligence and Deep Learning. This document provides a complete review of implementing a scalable and robust CNN solution for the problem of the text based movie review with the help of multi-class sentiment Analysis. We are using the Rotten Tomatoes movie reviews dataset. The scope of the project mainly relies on Convolutional and dense layers which are asscociated with the Max pooling layer. We are using the non-linear activation functions like ReLu and Softmax. We are using TF-IDF feature. In the given dataset, the training set and testing set was divided in the ratio of 70:30 by using the sklearn.model_selection library where the random state is set to the 2003. The performance evaluation of this model is based on the metrics like Accuracy, Recall, Precision and F1-Score.

keywords used- Epochs, learning rate, batch size, input layer, output layer, convolutional layer, max pooling layer, linear layer, flatten layer, kernel_size, loss, accuracy, f1-score, ReLu and softmax activation function.

II. Introduction

The Coding part has done on the CO-LAB platform which is provided by the Google. Google CO-LAB usally uses GPU because it has a very fast processing time. First of all, we need to connect to the GPU and open the notebook and start writing in the CO-LAB.

Sentiment Analysis :- Sentiment Analysis is the most common text classification tool which is used in analyzing the incoming text and tells whether the sentiment is neutral, negative or positive.

Sentiment Analysis usually refers to the utilization of natural language processing, biometrics, text analysis, computational linguistics and biometrics to systematically identify, extract, quantify and study affective states and subjective information. It is widely applied to various customer materials like reviews, responses, social media and health care materials for applications that range from marketing to the customer service.

Sentiment Analysis is also known as Contextual Mining which is used in identifying and extracting the subjective information in the material source. It also helps the business in understanding the sentiment of the brand, product and service. While the usage of Sentiment Analysis is completely restricted to the social media related streams. With the help of the recent advances in deep learning, the text analysis has increased rapidly.

Sentiment Analysis provides solutions for various problems like studying texts, posts, reviews which are uploaded by the user on various electronic and social media platforms. It is mainly used in classifying the text in a class. Sentiment classification can be either binary or multi class problem. Pre-processing is the initial step in text and sentiment classification. A lot of techniques are applied to data in order to reduce the noise of the text. The most famous popular techniques involved are remove numbers, stemming, part of speech tagging, remove punctuation, lowercase and remove stop words.

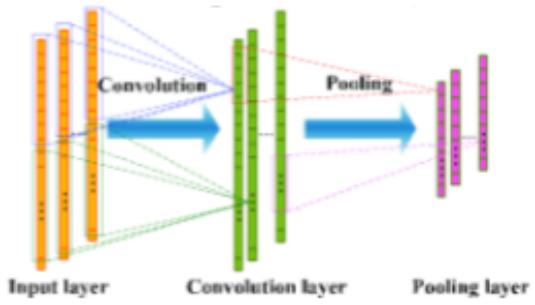

Fig. 1. Figure of CNN [1].

.

Convolutional Neural Network (CNN) :- Convolutional Neural Network is a part of Deep Learning. CNN is mainly used in analyzing the imaginary concepts. They have lot of applications in medical images analysis, image classification, image and video recognition and in Natural Language

978-1-7281-4387-3/19/$31.00 © 2019 IEEE

Processing. For the past many years, research on CNN has rapidly increased because of the real-world changes that are taking place in many computer resources and in the memory constraints.

The most common use of CNN is image classification where it identifies the satellite images that identifies the path. A CNN can also be implemented as a U-Net Architecture. A CNN contains an input layer, output layer and multiple hidden layers. The Hidden Layers of a CNN contains a set of Convolutional layers. Usually the Convolutional layer takes the input and pass the result to the next layer. Figure 1 shows the CNN model

Keras :- Keras is a deep learning framework and a high level neural networks API that was written in Python. It has a capability of running on the top of TensorFlow. Keras are used for easy and fast prototyping. It can support both convolutional network and recuurent network. Keras can run seamlessly on both CPU and GPU.

Metrics :- A Metric can be defined as the function which is used to help while judging the models performance. A Metric function is very similar to the loss function except the results. Metric function always returns the single tensor value that represents the mean of the output array across existing data points. There are various metrics existing namely,

- **Accuracy Metric :**- Accuracy Metric can be defined as the metric which is used in measuring the performance of the Algorithm.
- **Recall Metric :**- Recall Metric computes only a batch-wise average of recall. It computes the precision, a metric for multi-label classification of how many relevant items are selected.
- **Precision Metric :**- Precision Metric computes only batch-wise average of precision. It computes the precision a metric for multi-label classification of how many selected items are relevant.
- **F1-Score Metric :**- The F1 Score is used to measure the test's accuracy. F1 Score will consider both the precision p and the recall r in order to the compute the score. p is the number of correct positive results while r is the number of correct positive results.

TF-IDF :- TF-IDF means "Term Frequency - Inverse Document Frequency". This technique is usally in quantifying the word in the given document. TF-IDF method is most commonly used for Text Mining and Information Retrieval. TF-IDF scores can be calculated for all the words in the corpus.

Pre-processing Steps in NLP -

- **Tokenization :**- Tokenization is used to identify various different punctuation symbols, number and words.
- **Stemming**:Stemming is usally used to strip the end words of the given word.
- **POS tagging :**- POS tagging assigns each word in a sentence to its corresponding part-of-speech tag like designating the word as noun or adjective.
- **Parsing :**- Parsing usally divides the given text into various different categories which is usally used in answering a question.
- **NER :**- NER means Name Entity Recognition. NER identifies the entities such as person, location and time in the given document.

Libraries used in our code - We have used a lot of libraries in our code. They are:

- **numpy**- numpy library is used in adding and supporting the large and multi-dimensional arrays or matrices.
- **pandas**- pandas library is used in manipulating the data and for Numeric value analysis.
- **os**- os library provides a particular way of using the dependent functionalties of the operating system.
- **warnings**- It is the base category of the warnings that are triggered during the time of module importing.
- **nltk**- nltk provides a list of libraries that are necessary for Natural Language Processing.
- **re**- re means regular expression. it is a special sequence of characters that helps in finding or matching the strings or set of string
- **random**-random library generates a random float number.
- **sklearn**- sklearn supports scientific and numeric libraries of Python.
- **matplot**- matplotlib is a group of command style functions which make matplotlib wok like MATLAB .
- **torch**- torch provides the N-Dimensional Array.

III. Literature Survey

I have choosen the following two papers for my references and did my analysis based up on that.

In [1] Target-based sentiment analysis is a basic task that is necessary in the field of sentiment analysis, it requires the assessment of sentiment polarity in different targetsfor the same sentence. Various targets in the given sentence will have opposite sentiment polarities. TF-IDF features have great impact on method's accuracy.

In [10] TF-IDF usally intends how much a given term is relevant in the provided document. The relevant terms are those which helps the humanity in better understanding of the given document even without completely reading it.
.

IV. Proposed Model

Deep Learning has achieved a very high and good success rate in many applications. Deep Learning is preferred when we have a very big data sets because Deep Learning provides very fast and accurate results. In [1] they have also used keras which is one of the most famous deep learning technique used

to classify the data-sets. By using the keras the authors have studied and compared the effects based on the classification results. The following functions are used like Rectified Linear Unit (ReLu) and softmax. In the deep study both Convolutional Neural Network (CNN) and SoftMax classifier are used mainly for deep learning artificial neural network. The results they got at the end are very accurate. The key steps involved are:

- **Prerequisites (Importing all the required packages)**
- **Processing our dataset**
- **Creating the Network**
- **Training the model**
- **Testing the model**

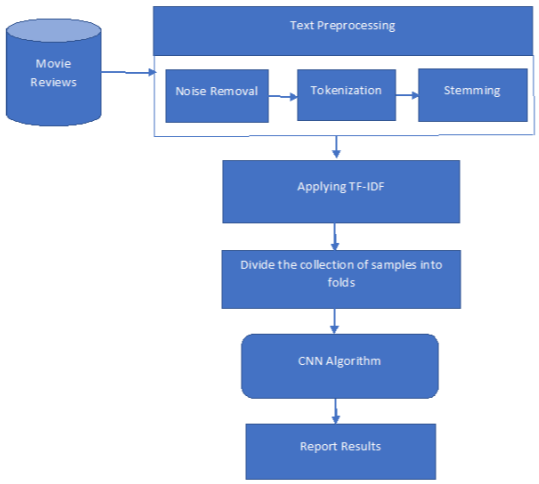

Fig. 2. Pre-processing techniques steps [3]

In this model First we need to connect to a GPU runtime and need to import all the packages like pandas, numpy, seaborn and nltk packages. NLTK Toolkit contains set of libraries and programs that are necessary for the symbolics and python programming language uasge, and we download punkt, stopwords and wordnet from that nltk package. Then punkt, stopwords, wordnet will get dwnloaded. Secondly import and download the dataset from the url whcih contains the phrases from the Rotten Tomatoes dataset. Each sentence will be parsed into many number of phrases. Each phrase and sentence has its own corresponding PhraseId and SentenceId. train.tsv contains all the phrases its corresponding sentiment labels. We will be assigning a sentiment label to each phrase. There are 5 types of sentiment labels mentoined as below,

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

Secondly, we need to split the both train and test data in the ratio of 70:30, and set random-state value to 2003. Then convert both X_train and Y_train values to numpy arrays and append those values to documents. import stopwords from nltk.corpus and import WorNetLemmatizier, PorterStemmer and LancasterStemmer from nltk.stem. Few parameters like remove-stopwords, useStemming, useLemma and removePuncs have considered for adjusting and to identify the impact of the outcome. The main purpose of this operation is for cleaning the reviews, tokenize and lemmatize. This will take each phrase iteratively and it will remove the html content, removing the non-alphabetic characters, tokenize the sentences and lemmatizing each word to its lemma. For each review document we need to save the review label and to apply Placeholder list for each word in the new review. Set the newWord variable to the updated word. If the word is a stopword we need to remove those stopwords orelse we can skip the word and dont had it to the normalized review. If the word is a punctuation word we need to remove those punctuations orelse skip the word and dont had it to the normalized review. The main purpose of this operation is for cleaning the reviews, tokenize and lemmatizeand.

Thirdly, we are using Vectorization for fitting the constraints as we dont know the x_train dataset. By using n-grams we are applying the vectorizer to the train data and to test data and then converting both X_train and X_test to the numpy arrays. Import keras packages for calculating the recall, precision and f1-score. recall can be calculated by the true-positives divided by the possible-positives and keras epsilon. precision can be calculated by the true-positives divided by the predicted-positives and keras epsilon, finally the f1-score can be calculated from the combination of both precision and recall.

Finally, we need to add Conv1D layer to the model with its parameters like filters, kernel-size and its activation function followed by Maxpooling layer to the model with the parameter pool-size and then we need to add the flatten layer to the model. At last we need to add dense layer to the model with the parameters number of classes and activation is softmax.

V. Experimental Analysis

While working with the code, several various number of combinations were verified. These methods means changing the different parameters in the code, which include: kernel-size, batch-size and epochs. I experimented by changing a lot of different parameters. Table I below shows the changes in the loss, accuracy, f1-score, precision and recall with respect to the batch-size and epochs. Table I represents the experimental analysis for different values.

VI. Model Performance

According to the above Table I, the best performing model is the model that has following hyperparameters.

TABLE I
EXPERIMENTAL ANALYSIS FOR DIFFERENT MODELS

Models	accuracy	f1-score	precision	recall
Model-1	0.604788	0.602249	0.6130721	0.59128
Model-2	0.605087	0.602295	0.6115611	0.59357
Model-3	0.612563	0.606508	0.6275877	0.58738
Model-4	0.624204	0.611806	0.6526586	0.57682

- **batch-size = 128**
- **epochs = 10**
- **kernel-size = 3**

The model's has an accuracy of 0.624204

VII. GITHUB LINK FOR THE CODE

https://github.com/vishnuSUDHEER94/Linear-Regression/blob/master/Assignment2.ipynb

VIII. CONCLUSION

In this paper we have performed a Multi-class Sentiment Analysis on the train data of the Rotten Tomatoes movie reviews. This paper explains the supervisied and scalable Convolutional Neural Network. It generates the output based on the trained inputs. After rigorous experimental analysis, the best has been found out, which gives the best accuracy of 0.624204

IX. APPENDIX

A. Reading the data and converting into data frame

```
sns.set(color_codes=True)
from nltk.tokenize import word_tokenize
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')

url = 'https://raw.githubusercontent.com/cacoderquan/Sentiment-Analysis-on-the-Rotten-Tomatoes-movie-review-dataset/master/train.tsv'
response = urllib2.urlopen(url)

df = pd.read_csv(response,delimiter='\t',encoding='utf-8')
df.head()

X_train, X_test, Y_train, Y_test = train_test_split(df ['Phrase'], df ['Sentiment'], test_size=0.3, random_state=2003)
documents=[]
X_train = np.array(X_train.values.tolist())
Y_train = np.array(Y_train.values.tolist())

for i in range(len(X_train)):
  documents.append([list(word_tokenize(X_train[i])), Y_train[i]])

X_test = np.array(X_test.values.tolist())
Y_test = np.array(Y_test.values.tolist())
for i in range(len(X_test)):
  documents.append([list(word_tokenize(X_test[i])), Y_test[i]])

documents[0]

porter = PorterStemmer()
lancaster=LancasterStemmer()
wordnet_lemmatizer = WordNetLemmatizer()
stopwords_en = stopwords.words("english")
punctuations="?:!.,;'\"-()"
remove_stopwords = True
useStemming = False
useLemma = False
removePuncs = True

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)
print(documents[0])

df = pd.DataFrame(documents, columns=['text', 'sentiment'])
df.head()
X_train, X_test, Y_train, Y_test = train_test_split(df['text'],  df['sentiment'], test_size=0.3, random_state=2003)
```

Listing 1. Reading the data

B. Vectorization

```
vectorizer = TfidfVectorizer(max_features = 6000, ngram_range=(1, 4))
X = vectorizer.fit_transform(df["text"])
Y = df['sentiment']

X_train = vectorizer.transform(X_train).toarray()
Y_train = Y_train
X_test = vectorizer.transform(X_test).toarray()
Y_test = Y_test
```

Listing 2. Vectorization

C. F1-Score Calculation

```
def recall_m(y_true, y_pred):
    true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
    possible_positives = K.sum(K.round(K.clip(y_true, 0, 1)))
    recall = true_positives / (possible_positives + K.epsilon())
    return recall

def precision_m(y_true, y_pred):
    true_positives = K.sum(K.round(K.clip(y_true * y_pred, 0, 1)))
    predicted_positives = K.sum(K.round(K.clip(y_pred, 0, 1)))
    precision = true_positives / (predicted_positives + K.epsilon())
    return precision

def f1_m(y_true, y_pred):
    precision = precision_m(y_true, y_pred)
    recall = recall_m(y_true, y_pred)
    return 2*((precision*recall)/(precision+recall+K.epsilon()))
```

Listing 3. F1-Score Calculation

D. Defining model

```
model = Sequential()
model.add(Conv1D(filters=128,kernel_size=3,
    activation='relu',input_shape=(6000,1)))
model.add(MaxPooling1D(pool_size=1))
model.add(Conv1D(128, kernel_size=3, activation='
    relu'))
model.add(MaxPooling1D(pool_size=1))
model.add(Conv1D(128, kernel_size=3, activation='
    relu'))
model.add(MaxPooling1D(pool_size=1))
model.add(Flatten())
model.add(Dense(num_classes, activation='softmax'))
model.compile(loss=keras.losses.
    categorical_crossentropy,optimizer=keras.
    optimizers.Adam(),metrics=['accuracy',f1_m,
    precision_m, recall_m])

X_train = X_train.reshape(X_train.shape[0], X_train.
    shape[1], 1)
X_test = X_test.reshape(X_test.shape[0], X_test.
    shape[1], 1)

model.fit(X_train, Y_train,batch_size=128,epochs=5)
score = model.evaluate(X_test, Y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
print('Test f1:', score[2])
print('Test precision:', score[3])
print('Test recall:', score[4])
```

Listing 4. Defining the model

REFERENCES

[1] Bohang Chen, Qiongxia Huang, Yi-Ping Phoebe Chen, Li Cheng, Riqing Chen. "Deep Neural Network for multi-class sentiment classification", IEEE, 2018

[2] Siyuan Chen, Chao Peng, Linsen Cai, Lanying Guo. "A Deep Neural Network Model for Target-based Sentiment Analysis", IEEE, 2018

[3] Xi Ouyang, Pan Zhou, Cheng Hua Li, Lijun Liu. "Sentiment Analysis Using Convolutional Neural Network", IEEE, 2015

[4] Sani Kamis, Dionysis Goularas. "Evaluation of Deep Learning Techniques in Sentiment Analysis from Twitter Data", IEEE, 2019

[5] Shiv Dhar, Suyog Pednekar, Kishan Borad, Ashwini Save. "Sentiment Analysis using Neural Network: A New Approach", IEEE, 2018

[6] Siyuan Chen, Chao Peng, Linsen Cai, Lanying Guo. "A Deep Neural Network Model for Target-based Sentiment Analysis", IEEE, 2018

[7] Bhaskar Dhariyal, Vadlamani Ravi, Kumar Ravi. "Sentiment Analysis via Doc2vec and Convolutional Neural Network hybrids", IEEE, 2018

[8] Jaspreet Kaur, Brahmaleen Kaur Sidhu. "Sentiment Analysis based on Deep Learning Approaches", IEEE, 2018

[9] Siyuan Chen, Chao Peng, Linsen Cai, Lanying Guo. "A Deep Neural Network Model for Target-based Sentiment Analysis", IEEE, 2018B.Pang, L.Lee, "Opinion mining and sentiment analysis, Foundations and trends in Information Retrieval", IEEE, 2018

[10] A.A. Hakim, A.Erwin,K.I Eng, M.Galinium and W.Muliady, "Automated document classification for news article in Bahasa Indonesia based on term frequency inverse document frequency(TF-IDF) approach", IEEE, 2014

-