greatlearning

PGP AIML

Capstone – Final Report_aimlChnOct19Grp4

SENTIMENT CLASSIFICATION WITH CUSTOM NAMED ENTITY RECOGNITION

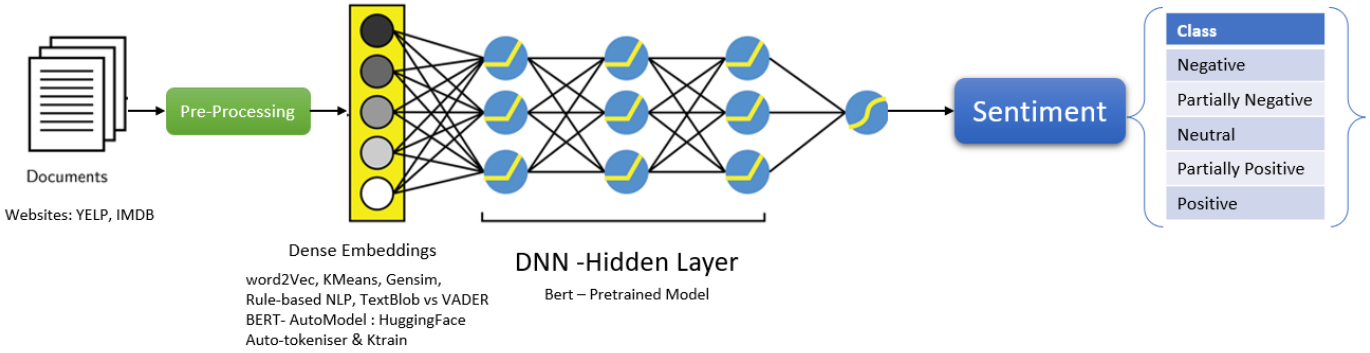
Ganga Babu, Bhagavathi Perumal, Daniel.V, Narendar Punithan, Pradeep Kumar S

Table of Contents

1. Summary of problem statement, data and findings:

- To build a classification model out of unprocessed text data/reviews using deep neural network to deal with information extraction from the real-world data based on polarity of the reviews for sentiment analysis. In order to get insightful information, we approach the machine and deep learning techniques for sentiment analysis and entity recognition from unprocessed data. By applying different techniques we can finalize the best performing model. Data are extracted from 'Digital content and entertainment industry' sourced from Data (cornell.edu) , DATA: Movie Review Data extracted for the use of sentiment-analysis experiments which was orchestrated on 2002 at Cornell University, New York by Bo Pang & Lillian Lee. Collections of movie-review documents labelled & Unlabelled. From the finalised model we tested the predictions on pipeline from the desired web sites such as Yelp reviews of restaurant and IMDB: Movie - The Suicide Squad – 2021 which has perfect cluster of reviews for our use case. And yes we achieved the desired 5 class sentiment predictions ['Negative', 'Neutral', 'Partially Negative', 'Partially Positive', 'Positive']. Sentiment analysis is by far one of the most important and commonly used NLP features. Sentiment is the classification of emotions extracted from a piece of text, speech, or document. While we typically look at emotion to capture feelings, such as anger, sadness, joy, fear, hesitation, sentiment is a higher-level classifier that divides the spectrum of emotions into positive, negative, and neutral. In our project we upgraded the spectrum to five set of emotions into Negative, Partially-Negative, Neutral, Partially-Positive & Positive.

2. Overview of the final process:

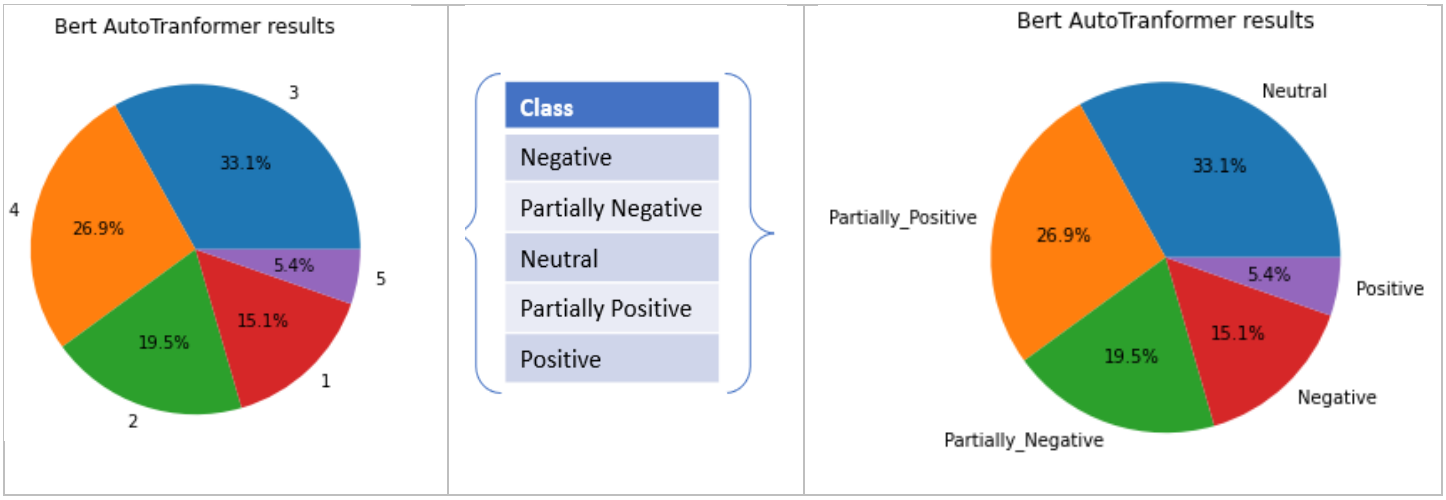


- In this experiment we cam across couple of unsatisfied model results earlier which we discussed in interim report of this project; however, we could achieve the Satisfied model which met our requirements it's nothing but uncased pretrained BERT Model with the accuracy of 47%. Important thing to learn about the salient features of the IMDB Data, Movie Review Data extracted for the use of sentiment-analysis experiments which was orchestrated on 2002 at Cornell University, New York by Bo Pang & Lillian Lee. Collections of movie review documents labelled & Unlabelled. For our capstone experiment we have chosen Pool of 27886 unprocessed html files (81.1Mb) all html files we collected from the IMDb archive. It's a collection of movie reviews retrieved from the imdb.com website in the early 2000s by Bo Pang and Lillian Lee. The reviews were collected and made available as part of their research on natural language processing. The reviews were originally released in 2002, but an updated and cleaned up version was released in 2004, referred to as "v2.0". The dataset is comprised of 1,000 positive and 1,000 negative movie reviews drawn from an archive of the rec.arts.movies.reviews newsgroup hosted at IMDB. The authors refer to this dataset as the "polarity dataset ". However, we curated data set from the original version. Data Preparation: For data Preparation we have used WebScaping method initially to handle large scale data, from the link provided above DATA polarity_html.zip file is load which is comprise of about 30,000 html and its each of its file size are below 12Kb, The entire Code and Data are processed in Google Colab which has free and high computation power environment rather local machine. To preserve the ascii and originally encoded version of data we used default utf-8. With the help of BeautifulSoup library unprocessed data which contains few HTML tags are concentrated and then a Data frame was created with three features Title, reviewd_by & reviews. BeautifulSoup function used to describe the details present in html file there were couple of html tags found (h1,title,a,p & h3) from these tags class body of text date which are reviews has been extracted. Text Pre-processing: Still reviews had a Paragraph tag, to overcome this challenge lxml.html.clean library was used, with the help of re package & mapping function I was able to remove regular expressions on entire data frame. Now the data frame holds about 27867 reviews as clean data for our capstone project. From the scraped process I was success fully able to create a dataframe consists of 'Title', 'reviewd_by', 'reviews'. With the help of regular expression '[^0-9a-z #+_]' unnecessary symbols are removed on entire data set. EDA- Top three reviewers (Steve Rhodes, James Berardinelli, Dennis Schwartz); Shape of Dataset: (27867, 3); Word2Vec Gensim helps us to convert each review into Word list, during this process 172856750 raw words (125681461 effective words) took 256.5s, 489940

effective words/s has been handled. At this Gensim model it (52256, 50) words are present on our entire dataset; Insights about the data – here I have checked the frequency of word, Shape and length of each reviews for general understanding, to find the length of the words on each review I used X.split function to split into words and count he same. This results the number of words on each review and created a new feature length. After experimented with available resource we approach with the BERT algorithm of BERT Auto Tokenizer and sequence classifier combined with Ktrain build upon BERT Pretrained model for training this combination technique which gives us tremendous desired output. From the saved model.h5 we have tested the real world through a custom pipeline for Yelp Reviews and IMDB Reviews.

3. Step-by-step walk through of the solution:

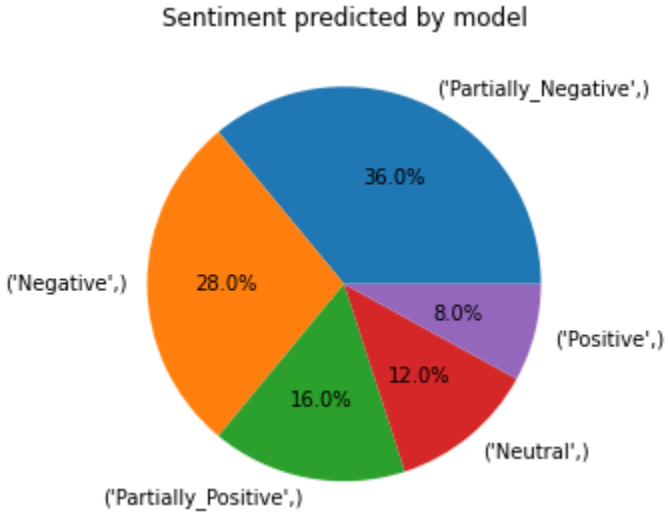
- Step 1: Installed necessary packages & used libraries for pre-processing, model building and visualization such as a Stable version of Pytorch, transformers from Huggingface, Requests, beautifulsoup4, pandas, numpy, AutoTokenizer, AutoModelForSequenceClassification, re (regular expression), torchinfo, pyplot, Ktrain, tensorflow, seaborn, text, etc.
- Step 2: From the web scraping process dataframe is created after that with the of custom defined function we obtained clean data. In the name of tokenizer and model variable we have loaded AutoTokenizer and AutoModelForSequenceClassification from hugging face database which comprises a BertForSequenceClassification of 12 BERT Layers which are BertSelfAttention mechanism a combination of Autoencoder and feedbacks. With the help of AutoTokenizer data is converted to logits: tensor([[-2.7768,1.2353,1.4419,1.9804,0.4584]],grad_fn=<AddmmBackward>);int(torch.argmax(result.logits))+1 > gives us a score of 4 for an input “It was good but couldve been better. Great”.
- Step 3: We introduced a user defined function called **sentiment_score** to auto label the each and every review for a corpus of 27K reviews it took three hours to label them. After that each scores are labelled by 5 Class: {1: 'Negative',2: 'Partially_Negative',3: 'Neutral',4: 'Partially_Positive',5: 'Positive'}



- Step 4: Splitting data for Train and Test (TRAIN_SIZE = 22000; TEST_SIZE = 5840) from Ktrain package we have text library for imputing the reviews and labelling for model building and to limit the data length of 500 words # maxlen means it is considering that much words and rest are getting truncated; uses Preprocess_mode : Bert means tokenizing, embedding and transformation of text corpus.
- Step 5: Model Building, In this stage Bert Model is prepared for classification purpose consists loop of layers such as Input -Token, Segment; Embedding -Token, Segment, Token-Segment, Position, Dropout, Normalization; Encoder - MultiHeadSelfAttention, FeedForward -Dropout,Add,Dense, Norm layers.

Trainable params: 109,476,869 / Total params: 109,476,869

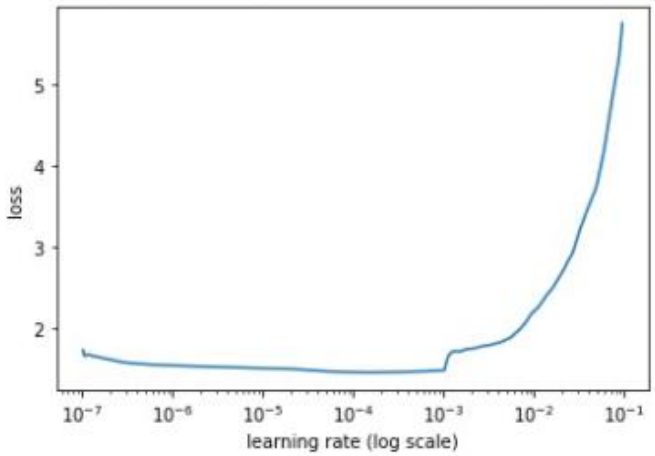
- Step 6: To train the model we have a variable name: learner
- Step 7: learner.fit_onecycle which is Essentially fit is a very basic training loop, whereas fit one cycle uses the one cycle policy callback Resulted the Accuracy of 47%
- Step 8: ktrain.get_predictor(learner.model, preproc) & predictor.save helps us to save the model in desired location in google drive. Size of the model build tf_model.h5: 1314.47 MB
- Step 9: predictor.explain gives the POS & NER of the sample data
- Step 10: With the help of few packages, we have tested the data in Realtime such as BeautifulSoup, requests & re from a real-time website we designed a pipeline to get a clean data set and convert them into dataframe I.e. predictor_load.predict(imdb_pipe_reviews) predicts the reviews and its sentiment.



- We have clearly explained the Justification of predictions at Code and Data submission.

4. Model evaluation:

- Visually inspecting the Loss and selecting the learning rate associated with falling loss.

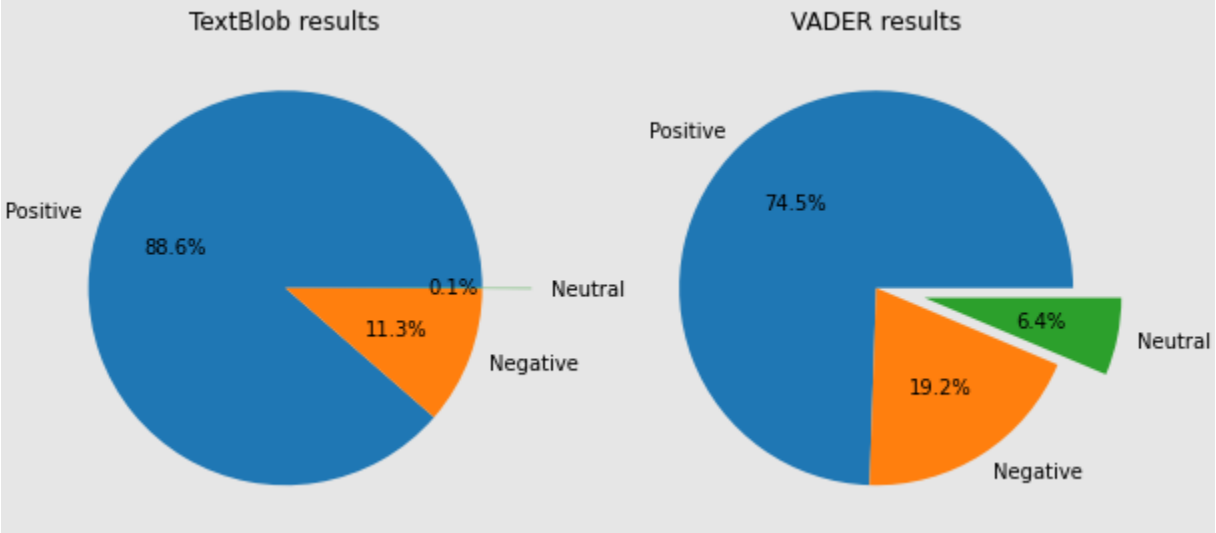



```
learner.validate(class_names=preproc.get_classes())
```

	precision	recall	f1-score	support
Negative	0.00	0.00	0.00	903
Neutral	0.35	0.37	0.36	2398
Partially_Negative	0.16	0.11	0.13	1359
Partially_Positive	0.27	0.47	0.34	1821
Positive	0.04	0.03	0.03	360
accuracy			0.47	6841
macro avg	0.16	0.19	0.17	6841
weighted avg	0.23	0.27	0.24	6841

5. Comparison to benchmark:

- Comparing performance to a benchmark definitely sets a higher “bar” than comparing to any other model experimented.
- TextBlob: Accuracy-83.98 | VADER: 80.29% | Ktrain Hugging face AutoTokenizer: 47%
- Even though TextBlob and Vader sets higher bar both failed during prediction. TextBlob failed to provide 0.1% of neutral sentiments when compared to TextBlob Vader was good.



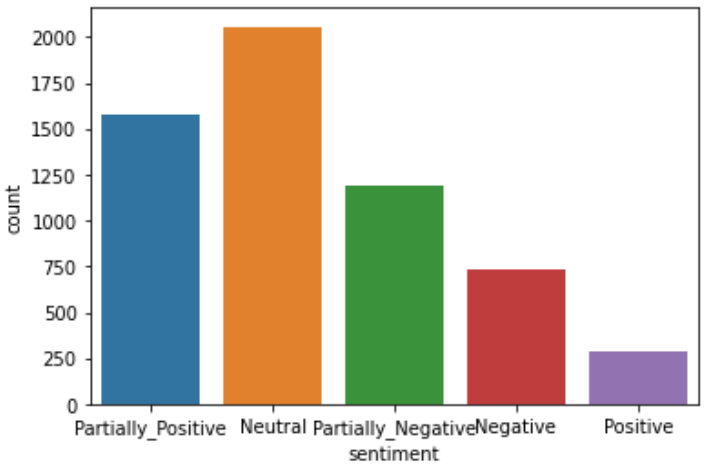
- Model trained by Ktrain which has inbuilt Bert Pretrained Model results us 47 % Accuracy and performed extraordinary during 5 Class sentiment prediction and most suitable for custom pipeline prediction either movie review or Restaurant review on Yelp.
  { loss: 1.2004 - accuracy: 0.4749 - val_loss: 0.8511 - val_accuracy: 0.6333 }

6. Visualization(s)

Amount of data fed during labelling

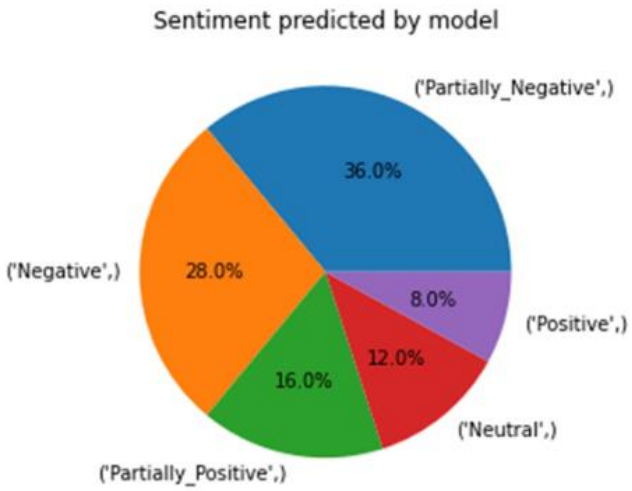
Neutral	2055
Partially_Positive	1577
Partially_Negative	1189
Negative	736
Positive	284

In Total 22000 for Training purpose



Prediction on final Model:

Sentiment Predicted on real-time data from IMDB movie: The suicide squad 2021



7. **Implications**

- Our solution solves multi domain or in several business cases; Initially this was built only for movie review sentiment analysis, but with the same model we have applied it on different review platform such as product review and restaurant reviews, Same principle can be applied with small changes in real time business purpose for example this is applicable for Company audit review with the right amount of information it can handle the repeated error made by organisation during Audit with ease of cosine similarity optimisation.

8. **Limitations**

- With the limited computational power model couldn't able to handle more than 24000 inputs. Colab crashed / Disconnected occasionally.
- In real world scenario data are generated tremendously in many forms. Now a days reviews are happening in speech that mean wave/audio data; It will be great if Wav2text built in our pipeline in order to listen to YouTube reviews.

9. **Closing Reflections**

- Sentiment analysis is one of the most commonly performed NLP tasks as it helps determine overall public opinion about a certain topic. enables enterprises to understand consumer sentiments in relation to specific products/services. Moreover, these insights could be used to improve their products and services by gauging consumers' comments and feedback using sentiment analysis. In the long run, sentiment analysis, if implemented the right way can aid business enterprises in improving the overall consumer experience, enhance brand image and propel business growth.