CSE598 Intro to Deep Learning
Project Proposal - Sentiment Analysis for Tweets Dataset

Classifying the Sentiment of the Tweets into Positive, Neutral and Negative using Sentiment140

Krishna Sree G (kgottmu)   Sai Sankar Vadde (svadde1)   Reethu Chowdary Vattikunta (rvattiku)

Project Idea

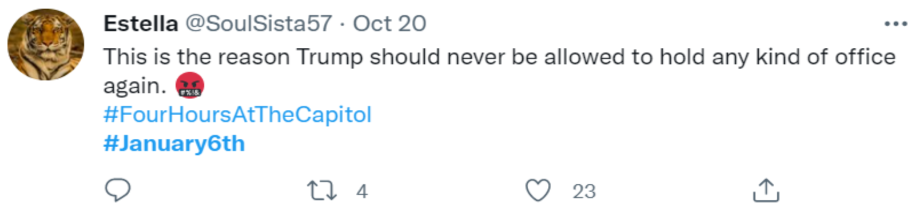
With the progression of web innovation and its development, there is a gigantic volume of information present in the web for internet clients and a great deal of information is produced as well. The Web has turned into a stage for the internet getting the hang of, trading thoughts and imparting insights. Informal communication destinations like Twitter, Facebook, and Google+ are quickly acquiring prevalence as they permit individuals to share and communicate their perspectives about points, have conversation with various networks, or post messages across the world. There has been some work in the field of feeling examination of twitter information.

This project's overview centers chiefly on the investigation of twitter information which is useful to dissect the data in the tweets where conclusions are exceptionally unstructured, heterogeneous and are either positive, negative or nonpartisan in some cases. In this project, we will use **BERT** for encoding the tweets and various machine learning models like MLP, Naive Bayes, and SVC.

Problem

Around 7,000 tweets are sent every second, and a large proportion probably mention Business, Political and Entertainment news. Whichever industry you work in – retail, finance, tech, health, government – you probably receive a lot of feedback or criticism on social media. And, you're looking at hours, maybe even days, to process all that data manually. Using our Project results, you can swim through all that information in minutes, to analyze individual emotions and overall public sentiment on any social platform.

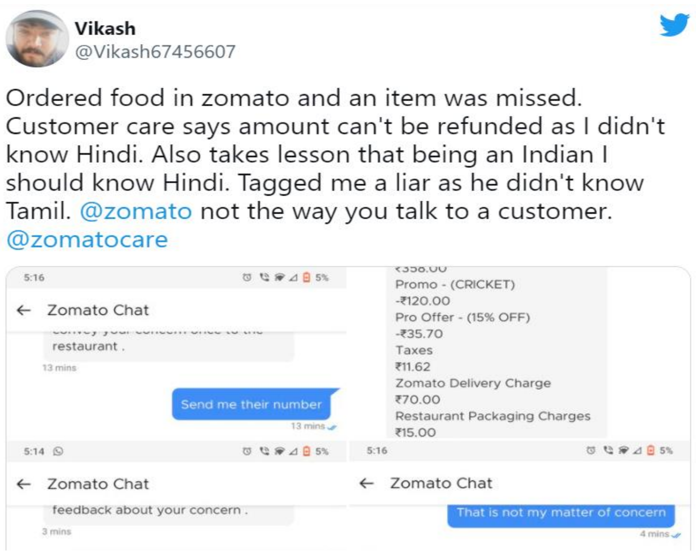
On January 6, 2021, a mob of supporters of President Donald Trump attacked the United States Capitol in Washington, D.C. The hashtag #january6th was mourn of the day.



Estella @SoulSista57 · Oct 20

This is the reason Trump should never be allowed to hold any kind of office again. 🤬
#FourHoursAtTheCapitol
#January6th

Zomato, an Indian multinational restaurant aggregator and food delivery company, utilized sentimental analysis in Twitter to resolve customer issues. Now such Twitter is shown below.



Vikash
@Vikash67456607

Ordered food in zomato and an item was missed. Customer care says amount can't be refunded as I didn't know Hindi. Also takes lesson that being an Indian I should know Hindi. Tagged me a liar as he didn't know Tamil. @zomato not the way you talk to a customer. @zomatocare

Dataset

In this project, we have used the sentiment140 dataset which contains 1.6M tweets extracted using the Twitter API. The tweets have been annotated (0 = negative and 4 = positive) and they can be used to detect sentiment. Due to the lack of adequate computing resources, we have utilized a sample of 80K instances with an equal proportion of negative and positive classes from the dataset for building the model. This sampled dataset is further being divided into 45% for Train, 15% for Validation, 40% for Test.

Dataset Link: sentiment140 · Datasets at Hugging Face

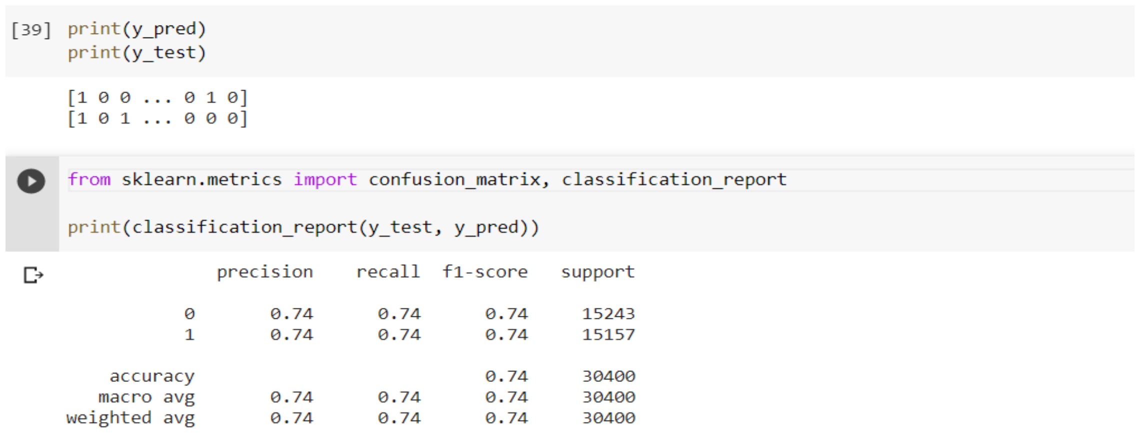
Baseline

We have used the Linear SVC model as a baseline for sentiment analysis. In this approach, we trained the data by using TF-IDF, hashing vectorizer to get the word embeddings from tweets, which is then being used as an input to the model. We achieved an accuracy of 74% on the test dataset.

Evaluation Metrics

In the most of the research papers studied so far, most of them have utilized F1-score and Accuracy as the evaluation metric, using classification report. So, we have used the same methodology of classification report for evaluating the model output.

Classification report for the baseline model is as below.



```
[39] print(y_pred)
     print(y_test)

     [1 0 0 ... 0 1 0]
     [1 0 1 ... 0 0 0]
```

```
from sklearn.metrics import confusion_matrix, classification_report

print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.74      0.74      0.74     15243
           1       0.74      0.74      0.74     15157

    accuracy                           0.74     30400
   macro avg       0.74      0.74      0.74     30400
weighted avg       0.74      0.74      0.74     30400
```

Experiments and Results

Previous approaches used Word2Vec, Count Vectorizer to convert a word into its corresponding word embedding, which doesn't consider the context of the whole sentence. These vectorizers provide fixed embedding for a word irrespective of its context. We have used BERT transformer across the project.

Upon several experiments to train the model, we have observed that the following hyper-parameters provided us with the best results.

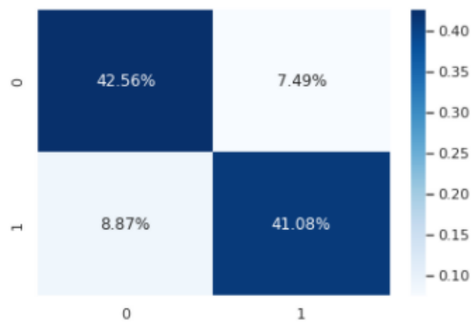
Batch Size: 32
Learning Rate: 5e-5
Decay Rate: -0.3
Optimizer: AdamW
Model: bert-base-uncased

Approach-1:

In the first approach, we have used the vanilla BERT encoder-only transformer, to get the word embeddings and further used BERT classifier for text classification. As BERT uses Bi-directional encoder, it considers whole context of the sentence for each word and generates word embeddings. These word embeddings are then passed as inputs to the model.

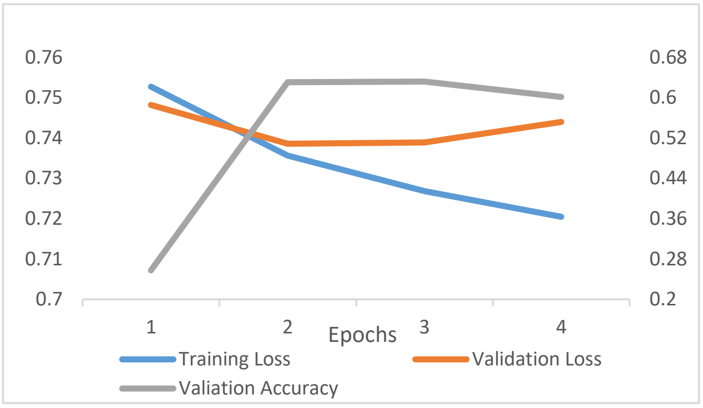
Epochs	Training Loss	Validation Loss	Validation Accuracy
Epoch-1	0.414	0.378	0.838
Epoch-2	0.242	0.402	0.844

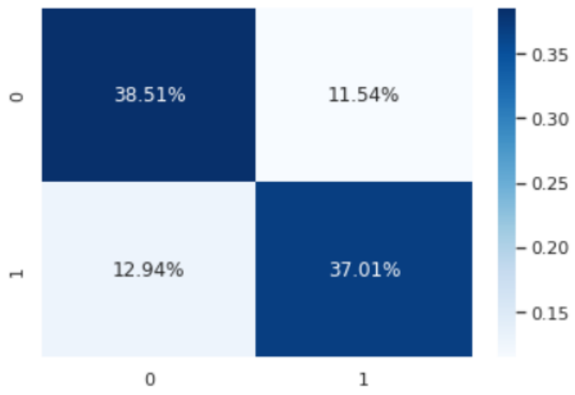
We achieved a test accuracy of ~84% and a F1-score of 84%.



Approach-2:

In this approach, we have used BERTweet model which has been pre-trained on tweets corpus. The architecture of BERTweet pre-trained model is similar to that of the BERT-base and is trained using the pre-training procedure named RoBERT. Using this pre-trained model directly on our dataset, we got an accuracy of ~48% which is too low. To make the model learn better, we went ahead and fine-tuned the model on our training dataset. This boosted the accuracy of the model to ~76%, after 4 epochs.





Approach-3:

In this approach, we explored the concept of Multi-task learning. The 'bert-base-uncased' model was initially pre-trained on IMDB dataset for sentiment analysis, where we observed accuracy of ~89%. This pre-trained model was then fine-tuned on our Twitter training dataset. This approach gave us a test accuracy of ~83%.

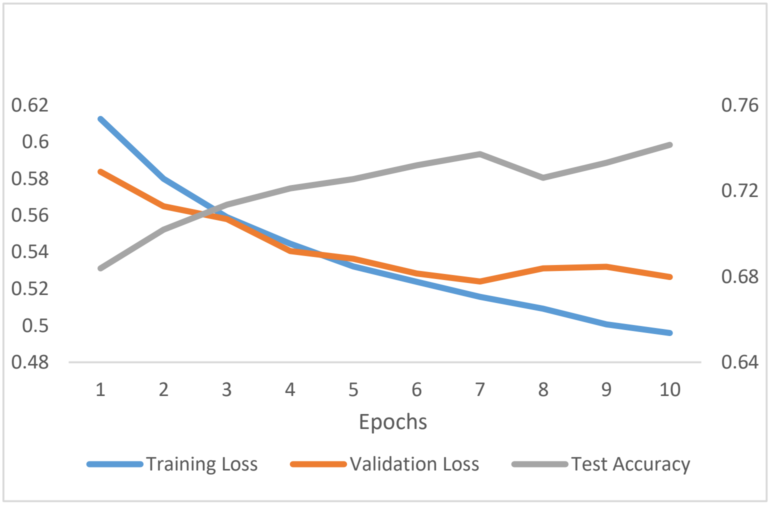
This task achieved similar accuracy as that of the vanilla BERT model (Approach-1) but we had the additional overhead of computation budget.

Approach-4:

As a part of this approach, we used Bi-directional LSTM to classify the given input sequence into either positive or neutral. The main idea behind usage of LSTM was the fact that LSTMs are designed to keep the entire sequence context in their cell state. Moreover, we have used bi-directional LSTM which not only keeps track of the past context but also the future context corresponding to an input sequence.

To get vector representations for each word in the input sequence, we have used GLOVE embedding, which does not consider any context and always output fixed vector representation irrespective of its position and context in the input sequence.

We achieved an overall test accuracy of ~74%.



The main motivation behind using the different approaches to train a language model is to evaluate the performance of the model with respect to computational budget.

Computational Budget:

This is the initial comparison between Approach-1 and Approach-3:

Factors	BERT Encoder + BERT Classifier	Glove Embedding + LSTM Classifier
Computation Time per Epoch	20 Mins	1.5 Min
Test Accuracy	84% for one epoch	74.15% for 10 epochs

We can clearly see that the BERT pre-trained model has really worked quite well and has achieved an accuracy of 84% after one epoch (within 20 mins). On the contrary, LSTM Classifier achieved only ~74% even after 10 epochs (15 mins). When we look at it from the lens of performance, BERT has clearly outperformed LSTM Classifier. The possible inference for the better performance of BERT might be due

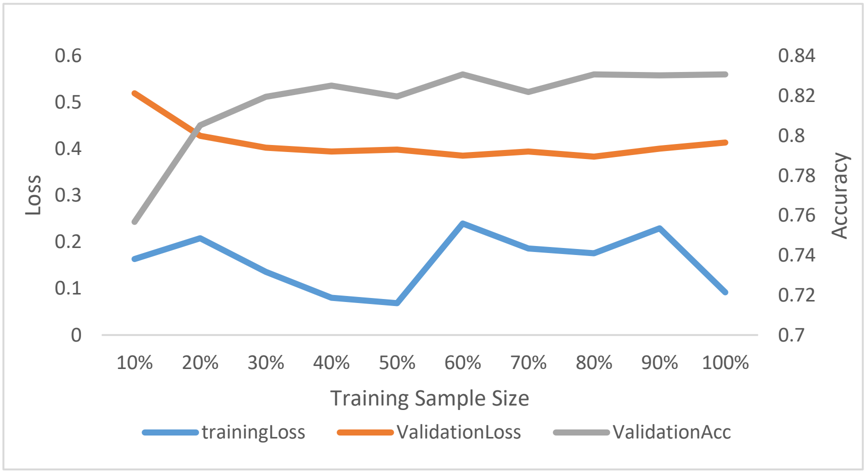


Figure A – Computation budget analysis across incremental sample sizes

to the BERT encoder which considers the position of each token as well as the surrounding words to generate the word embedding, which is not the case in GLOVE embedding.

We also probed the performance of BERT in terms of computation using different Sample sizes for training. The experiment was setup in such a way that the BERT model would be initially trained using 10% of the total sample size. This process is repeated by training various BERT models and increasing the sample size for each successive model by 10% until we have a model which is trained using 100% of the training dataset. The results have been showcased in Figure A. We can observe that the validation accuracy is increasing gradually with the increase in sample size, which tells that the trained model is able to generalize a bit better with the increase in sample size.

However, we can also see that the training loss is not the least for 100% training data, rather the training loss is the lowest for sample size with only 50%. Also, the Validation accuracy became constant after the sample size rose to 80% and beyond. Overall a specific trend can't be inferred from the data, which might be due to the randomness in the sampled data.

Error Analysis:

Let's go back to Approach-1, where we implemented the vanilla BERT Classifier, which has performed the best for our data, and has achieved an accuracy of ~84%. We investigated the predictions by the model for the error analysis, to understand where it was going wrong.

Input Sequence	**Target**	**Prediction**

@richardebaker no. it is too big. I'm quite happy with the Kindle2.	1	0
@ludajuice Lebron is a Beast, but I'm still cheering 4 the A..til the end.	0	1
@sekseemess no. I'm not itchy for now. Maybe later, lol.	0	1

We can observe that the model is not able to learn well when there is a presence of negation words such as not, but etc. in the input sequence.

Input Sequence	**Target**	**Prediction**
need suggestions for a good IR filter for my canon 40D ... got some? pls DM	0	1
San Francisco today. Any suggestions?	0	1
New blog post: Harvard Versus Stanford - Who Wins? http://bit.ly/MCoCo	0	1

On further observation, we could observe a mismatch between the target and prediction, whenever the input sequence was in the form of question. However, in our opinion, there is a bias introduced in terms of data labelling. Consider the first instance, though the input sequence sounds positive, it has been labelled as negative where as our model was able to predict it correctly.

References

1. 2110.00859.pdf (arxiv.org)
2. TwitterDistantSupervision09.pdf (stanford.edu)
3. Arabic aspect based sentiment analysis using BERT. (arxiv.org)