**Enterpret Assignment Submission**

**By**

**Vignesh**

**Task -**

Given a text and phrase(already containing the word), we were asked to find the sentiment associated with a phrase in the text.

**Example -**

Text - *“The milk was very tasty but the curd was very ripe.”*

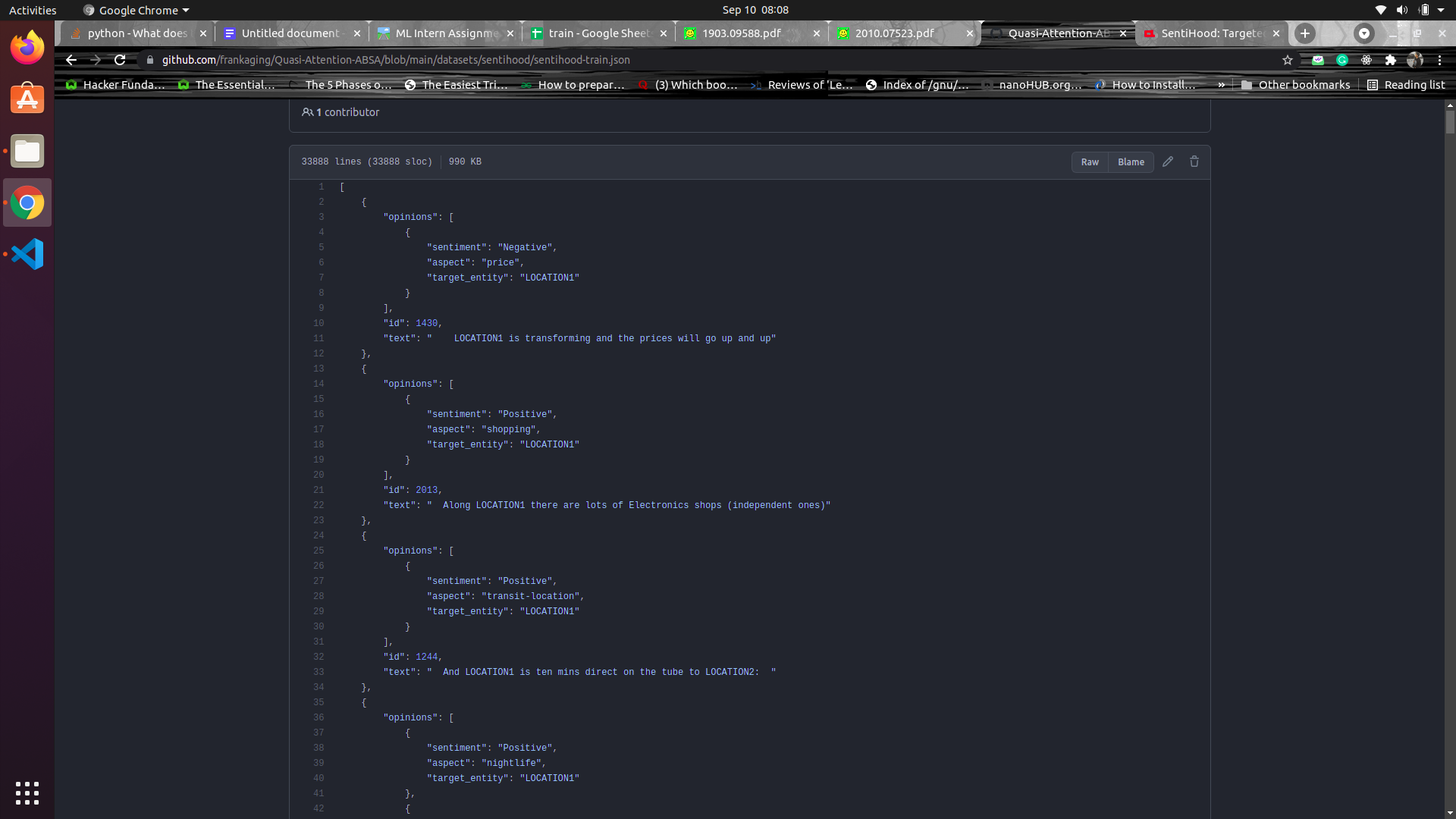
Phrase- *“milk”* Sentiment - *“Positive”*

Phrase- *“curd”* Sentiment - *“Negative”*

**Literature Survey -**

The task was similar to Aspect Based Sentiment Analysis(ABSA) where a common task is to segment sequences that fit into a specific aspect like - {price, support, safety} and Targeted Aspect Based Sentiment Analysis(TABSA) which is similar to ABSA but a target word is given which belongs to a set given **T**.

A common dataset on which models have been benchmarked is the SentiHood dataset consisting of 5215 sentences and Semval dataset. Below are snippets from the Sentihood dataset



1. Utilizing BERT for Aspect-Based Sentiment Analysis via Constructing Auxiliary Sentence [[paper](https://arxiv.org/abs/1903.09588)]:
   * This paper claimed to have gotten 92% accuracy for Sentiment detection on the SentHood dataset using BERT and following tokenization strategy of concatenating phrases and aspects with [SEP] token.
2. Context-Guided BERT for Targeted Aspect-Based Sentiment Analysis [[paper](https://arxiv.org/abs/2010.07523)] [[code](https://github.com/frankaging/Quasi-Attention-ABSA)]:
   * This paper claimed to have gotten 94% accuracy for Sentiment detection and introduced a new architecture using the existing BERT and different layer norm strategies. Although this paper seemed to have better perspectives, I wasn’t able to investigate further.

**My Training Approach -**

In both my approaches mentioned below I had experimented with Hugggingface implementation of “BERT”, “DistillBERT” and “TinyBERT” . The key difference between the two approaches mentioned below are the tokenisation strategy and both models performed very similarly for all models and approaches. My biggest challenge was that models were overfitting very quickly and appropriately tuned the learning rates for it. All experiments were done on 5-fold stratified cross-validation using a Goggle Collab environment.

The preprocessing pipeline included a regular text cleaning exercise where noisy strings were lowercase, numbers were removed and few hyperlinks were removed. The dataset consisted of many spelling mistakes and I had tried cleaning them with textblob library but since we were dealing with Product review datasets many times product names would often get corrected to wrong words. Hence, spell-correction was not done but if it was done manually, the performance would have significantly improved.

1. **Tokenisation Method 1 -**

Directly concatenated Text and phrase strings with a [SEP] token.

**Example -**

Text - *“The milk was sour.”*

Phrase - “*milk*”

Token ids - “*The milk was sour[SEP] milk*”

Token Type- [ 0 , 0 , 0 , 0 , 1 , 1 ]

Example - “milk was tasty but the curd was sour”

**Accuracy -**

|  |  | **BERT** |  |  |  | **TINY** | **BERT** |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Overall**  **Acc.** | **Neg.**  **Recall.** | **Neutral**  **Recall.** | **Positive**  **Acc.** | **Overall**  **Acc.** | **Neg.**  **Acc.** | **Neutral**  **Acc.** | **Positive**  **Acc.** |
| **Fold 0** | **72.3** | **77.8** | **74.9** | **66.3** | **72.9** | **74** | **73** | **69** |
| **Fold 1** | **73.1** | **74.7** | **72.6** | **71.2** | **73.6** | **75.6** | **73.4** | **70.7** |
| **Fold 2** | **73.9** | **72.3** | **81.5** | **66.8** | **75.1** | **76.5** | **80.7** | **65.9** |
| **Fold 3** | **71.8** | **72.6** | **76.1** | **64.9** | **71.9** | **75.9** | **71.0** | **66.3** |
| **Fold 4** | **73.4** | **73.2** | **73.3** | **73.8** | **71.9** | **71.9** | **72.9** | **75.2** |

**\*Above study can be found in notebooks/method1.ipynb**

1. Tokenisation Method 2 -

Extracting contextualized vectors for the respective target word in the sentence and taking a mean of those vectors and passing a Feed-forward layer through them

**Example -**

Text - *“[cls] The milk was sour but milk can also be tasty”*  ------> Total words - 10

Phrase - “*milk*”

Phrase” customer service” -

(10,768) ==

Token Type- [ 0 , 0 , 0 , 0 , 1 , 1 ]

In the above text, there are 10 words and the BERT final embedding layer will generate a tensor array of shapes (10, 768).

The **milk** was sour but **milk** can also be tasty.

We extract the embedding’s associated with the phrase “milk” and compute the mean of those embeddings (in this case two vectors)

With this, we pass it through two linear layers and extract the final predictions.

**Accuracy -**

|  |  | **BERT** |  |  |  | **TINY** | **BERT** |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Overall**  **Acc.** | **Neg.**  **Acc.** | **Neutral**  **Acc.** | **Positive**  **Acc.** | **Overall**  **Acc.** | **Neg.**  **Acc.** | **Neutral**  **Acc.** | **Positive**  **Acc.** |
| **Fold 0** | **70.9** | **75.3** | **66.4** | **69.3** | **70.8** | **73.2** | **66.8** | **71.7** |
| **Fold 1** | **72.9** | **77.1** | **71.8** | **67.3** | **73.5** | **77.7** | **66.9** | **71.2** |
| **Fold 2** | **75.7** | **81.8** | **72.6** | **69.8** | **72.6** | **76.8** | **72.6** | **65.9** |
| **Fold 3** | **69.0** | **75.6** | **62.2** | **66.8** | **71.3** | **73.5** | **66.8** | **69.3** |
| **Fold 4** | **73.9** | **80.4** | **70.9** | **67.3** | **71.6** | **75.0** | **73.3** | **64.1** |

**\*Above study can be found in notebooks/method2.ipynb**

**Final Metrics -**

The metrics I had mainly used Overall Accuracy and F1 score as mentioned in papers. As the dataset provided was fairly balanced, the weighted F1 score and accuracy were very similar. The dataset provided to us had a lot of spelling mistakes but at the same time, it was difficult to correct many of them as this dataset was taken for product analysis and correcting the spelling of products would change the accuracy. Another important aspect to keep in mind while choosing the models was the latency.

**Ablation Study -**

Using both the strategies mentioned above, we were getting the same accuracy around 72% using different models. Now I had to learn if these results were actually due to the phrase or general sentiment of the whole sentence. So I had passed Texts without Phrases through the BERT based models and here were the results.

|  |  | **BERT** |  |  |  | **TINY** | **BERT** |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Overall**  **Acc.** | **Neg.**  **Acc.** | **Neutral**  **Acc.** | **Positive**  **Acc.** | **Overall**  **Acc.** | **Neg.**  **Acc.** | **Neutral**  **Acc.** | **Positive**  **Acc.** |
| **Fold 0** | **71.6** | **73.2** | **72.6** | **67.8** | **67.1** | **73.8** | **60.2** | **64.9** |
| **Fold 1** | **73.1** | **78.0** | **71.0** | **67.8** | **71.8** | **77.7** | **71.0** | **62.9** |
| **Fold 2** | **Nan** | **Nan** | **Nan** | **Nan** | **70.8** | **75.9** | **69.9** | **63.4** |
| **Fold 3** | **Nan** | **Nan** | **Nan** | **Nan** | **66.1** | **73.8** | **61.0** | **60.0** |
| **Fold 4** | **Nan** | **Nan** | **Nan** | **Nan** | **68.6** | **72.6** | **69.8** | **60.7** |

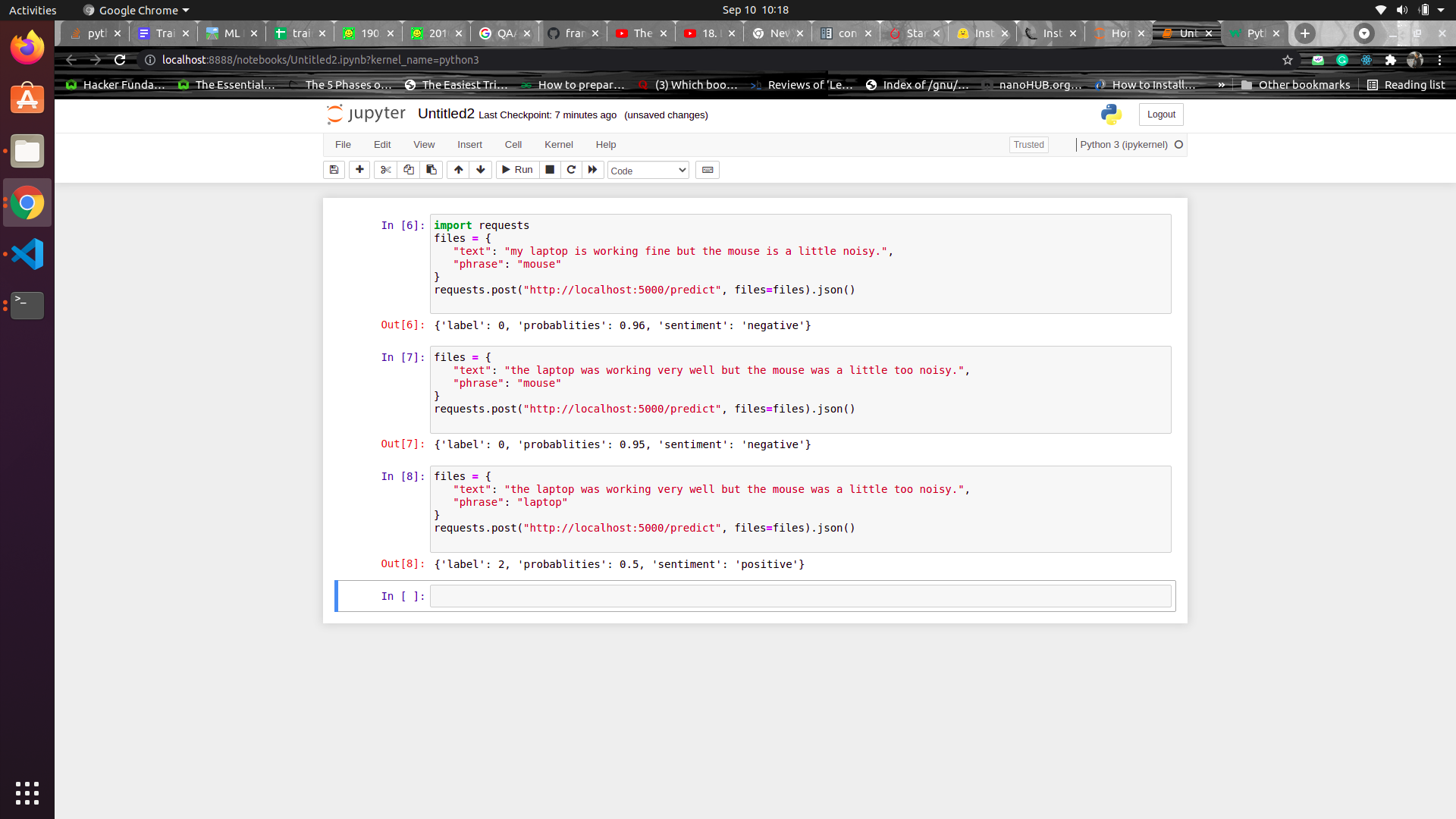
**\*Above study can be found in notebooks/method1.ipynb**

A more general conclusion that can be extracted from our analysis is that texts with a general negative sentiment are more likely to have a negative sentiment about the targets.

**Conclusion -**

From the ablation study, it is clear that the model trained with aspect keywords performed very similar to a model without an aspect key (Around 3-4% increase in accuracy). So it is very hard to conclude if the model performed well due to your specialised training approaches either by concatenating the tokens or extracting embeddings. This does not mean that our, model hasn’t learnt to represent the aspects accurately but it will have a hard time in accurately predicting sentences with multiple sentiments.

Here are few handpicked examples with multiple sentiments where the model performed well -



Keeping the pre-processing simple and latency small, I had decided to choose Tiny Bert model using our first strategy of concatenating vectors.