Project Report: Final Project

Submitted by: Eyoel Feleke, Samuel Ghebreyesus, Srishti Hedge

Overview of the project

The purpose of this project was to explore a data set and use some machine learning method on this data, then evaluate its performance along with a couple of its variants based on some evaluation metric.

In this submission, we worked with The Huffington News categorization data set which contained news headlines and related data each of which belonged to one of 42 different categories of news.

In the course of Roger, we initially exploited data to get some insights about the data set. This gave us some interesting insights about the cardinality of the category membership of the data points in the data set the average word length and Max Word length in their headlines, the distribution of headline categories through the years from 2012 to 2022.

Post this data exploration, we process the data as required for different implementations. Initially we use a pre trained DISTIL-BERT model with frozen parameters for training and testing the model. In later iterations, we train all the parameters of the model and consolidate the 42 categories into just 13 categories. We also implemented a model with an additional Fully Connected layer. In addition to DISTIL-BERT model, we also tried working with GPT 2 model. The details, architecture and the results of these iterations are explored in detail in the later sections.

The best performance we observed among these iterations was around 69% accuracy on the test set and the F1 score of 0.689 fully with the DISTIL-BERT model with one fully connected layer, where we trained all weights .

Problem Statement

The problem statement in accordance with which we've gone about working on this project is 2 leveled. Primarily, the goal is to recognize/classify the category of a news headline based on the data provided about it such as the headline, short description etc. In the long term, the goal is to create A version of this model that has been trained on this data set, that is capable of recognizing the context or category of some given text that isn't necessarily a news article or headline.

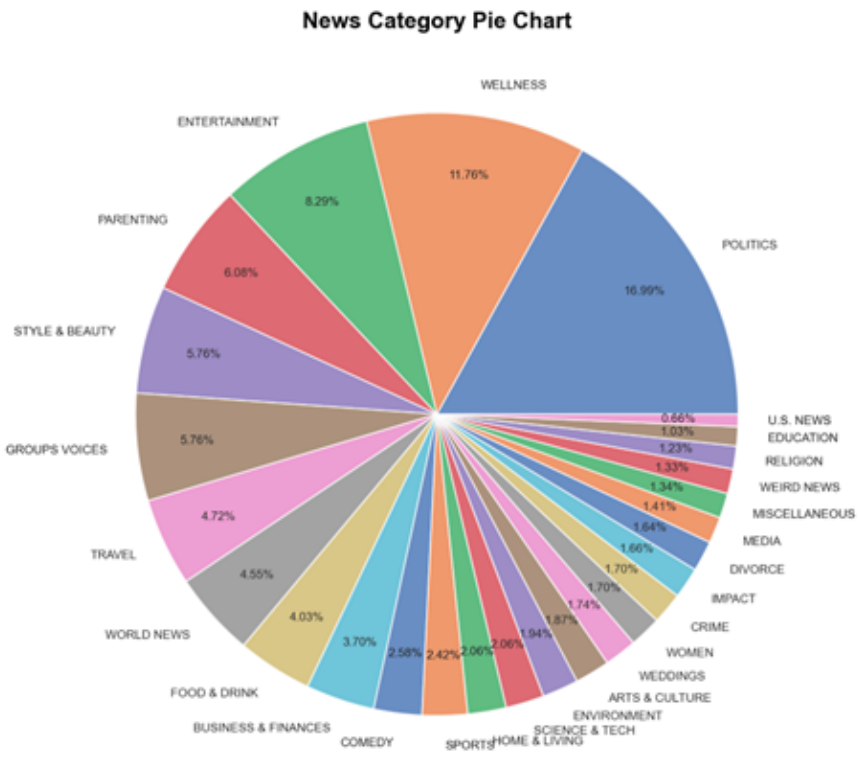
About the dataset

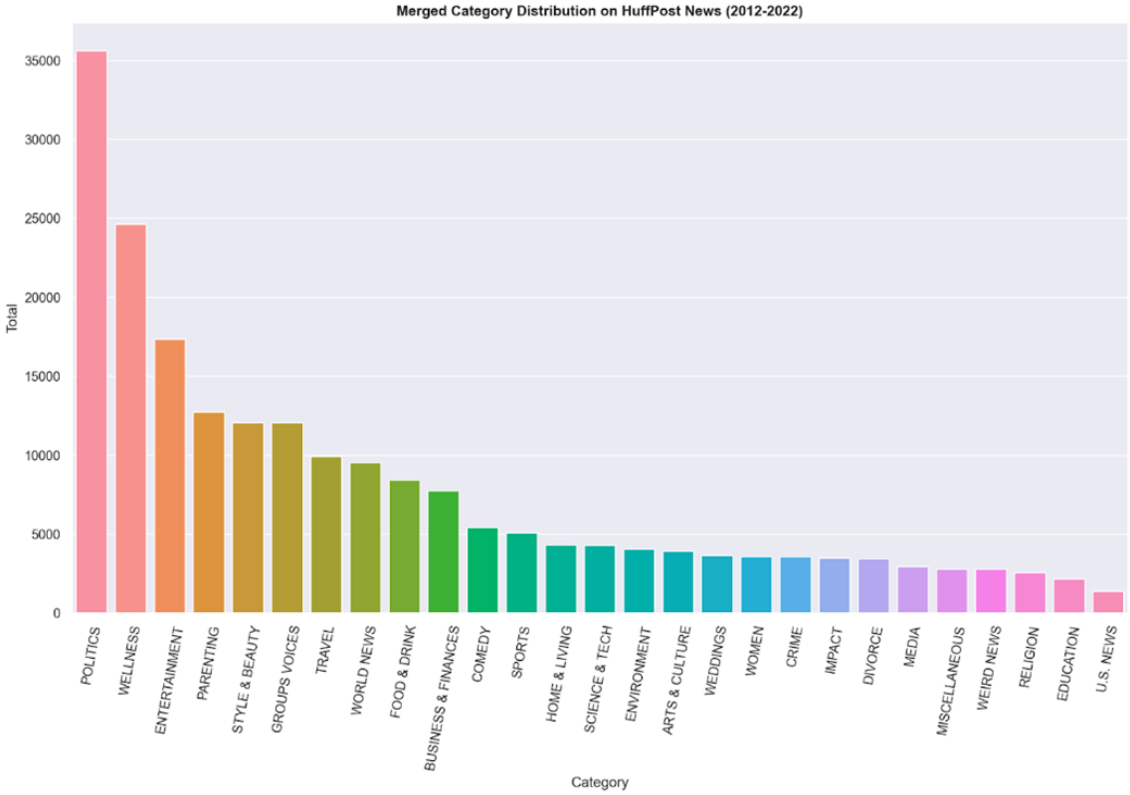
The data set that has been used in this project is the **News Category dataset** that contains around 210K data points each of which corresponds to a new article published in HuffPost. Each entry that corresponds to a news article is defined by its headline, author link to the article short description of the article data of publication and the category in which the article was published. The data points collected where from between 2012 to 2022

Since the news site stopped maintaining this extensive archive of the published articles around 2018, there is a significant drop in the count of data points added to this set post 2018.

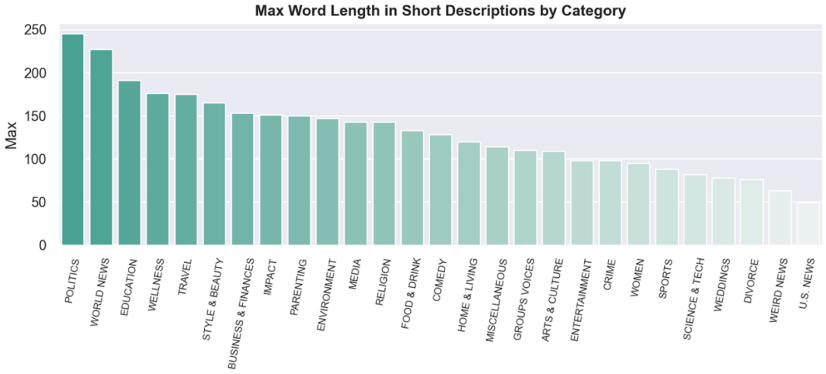
To gain more insights about the data, we do some exploratory data analysis. When we tried automatic profiling of data using pandas, we didn't have major findings from this data come on as the data was all textual categorical and date based. So instead, we tried some other approaches. Some of the notables finds are listed below:

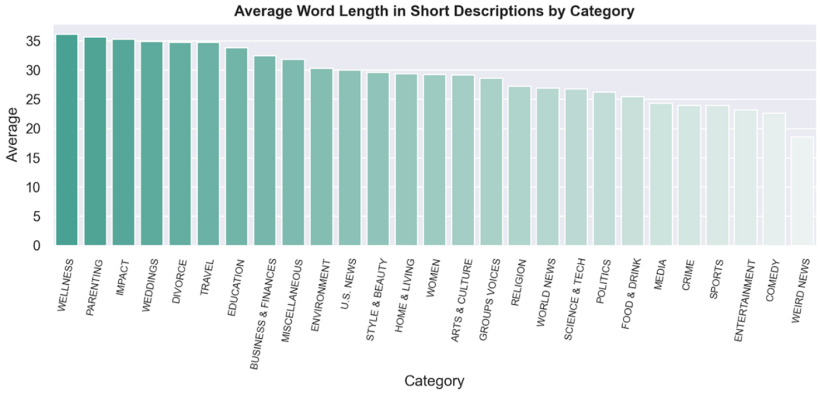
- Looking at the category cardinality in the 210K data points



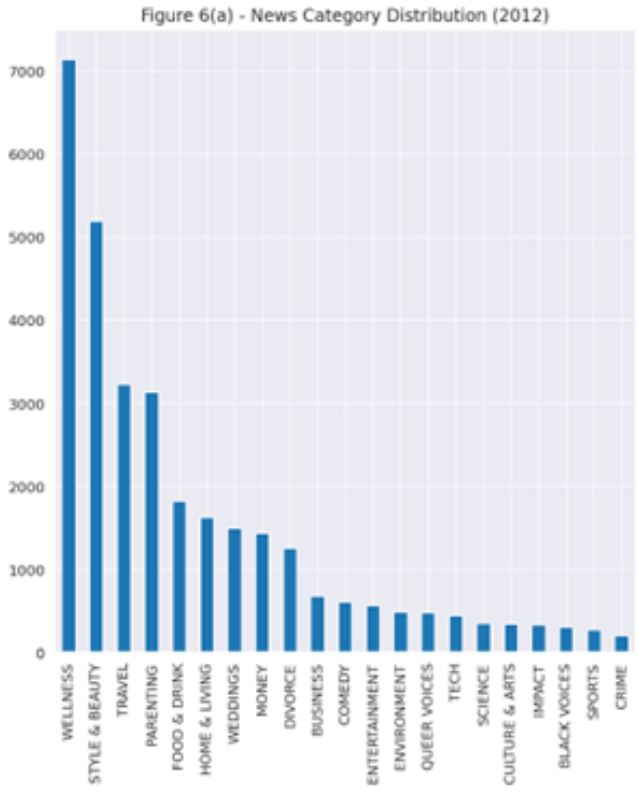


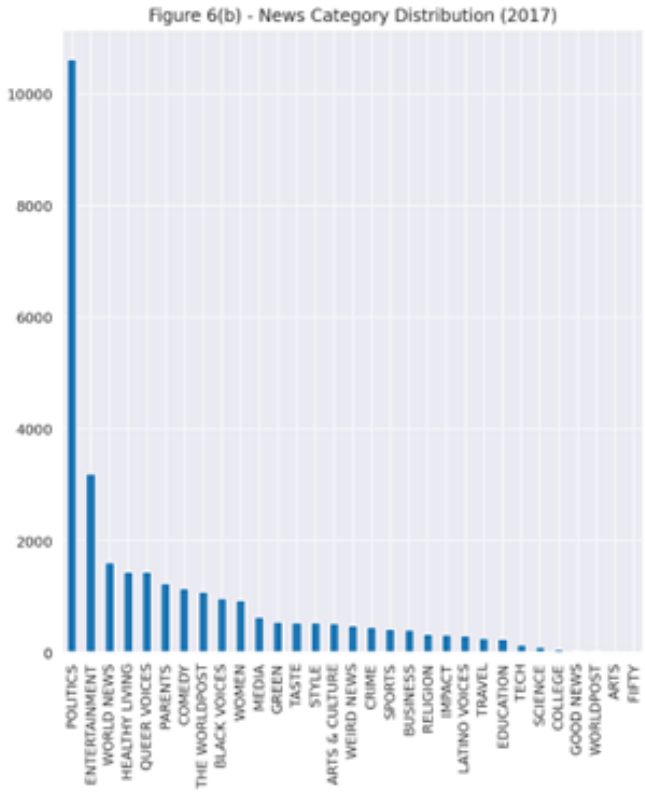
- Looking at the average and maximum word length in the headlines

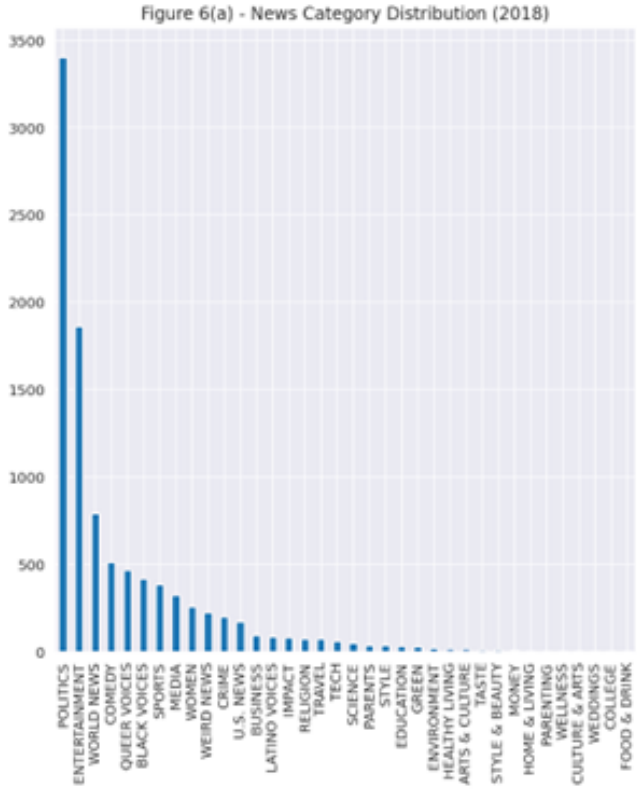


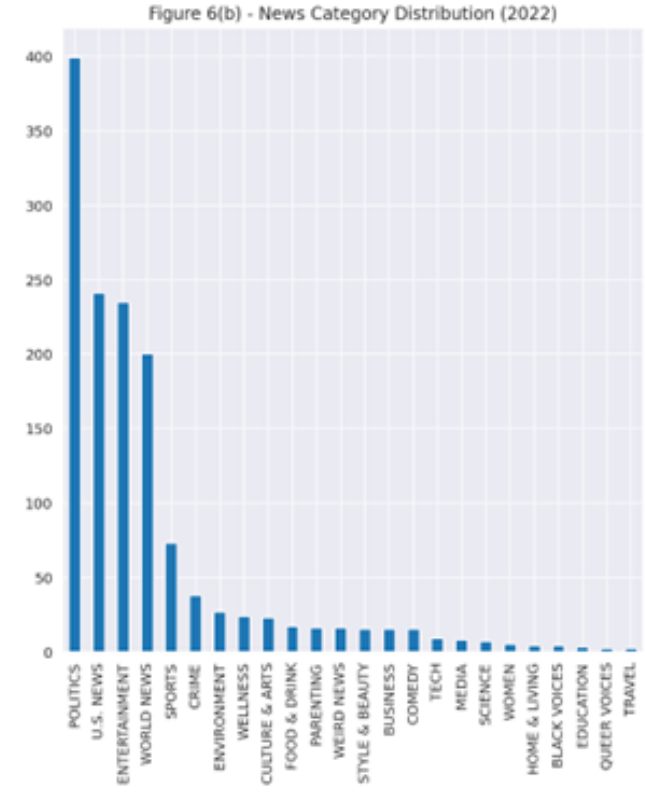


- Looking at the distribution of categories through different years to understand if there have been any trends through the years









For all future processing of this data, we dropped the link the link date of publication and author fields as ideally, they should not have an impact on the categorization of a news article or even some other text in the future.

In terms of model choice, due to the nature of the data and the task at hand BERT model was a good choice for the task. But instead of using the relatively heavy BERT model with around 110 million parameters, for our purposes we use a lighter version of the BERT known as DISTIL-BERT model which is around 40% lighter with around 66 million parameters while still retaining around 97% of BERT's functionality.

ML method implemented

After we explored the data and got insights about the characteristics our dataset we decided to use a transfer learning approach to solve the problem. As we know, training and finetuning a Natural Language Processing model is not a trivial task. It needs a larger and more comprehensive data than this news category dataset. This is because we have to learn the word embedding and then train a classifier model based of our dataset. This is beyond the scope of the project and we don't have enough dataset to learn a the vocabulary vector.

Hence we used pre-trained language model such as DistilBERT, GPT2 as our base models for the models and built a ANN models on top of those. DistilBERT was choose as our primary base model because it a 40% lighter (had much smaller number of weights) and 60% faster compared to its original version BERT. DistilBERT has around 66m weight parameters where BERT had around 110m weight parameters.

Before building our model, we created a PyTorch Dataset Class that loads the data and applies preprocessing step. We used the word embedding vector from the Pre-Trained Model to tokenize the words. The preprocessing step we performed in the dataset class include

- Tokenizing each word in the text
- Padding shorter texts to match the max sequence configuration of the pre-trained model.

Additionally, we applied resampling to overcome the huge class imbalance we have in the data. In our best performing architecture we used a data that was resampled around 8k datapoints from each category. Moreover, the best performing model also benefited from consolidated categories, another preprocessing approach we implemented. We were able to reduce number of categories from 42 to just 13.

Then we created a module class, that basically defined our architecture of our model and defined a forward method for forward pass during training and evaluation process.

For all our iterations we used Adam Optimizer and CrossEntrolpLoss loss function with a 1e-5 learning rate.

The first iteration we experimented with is a pre-trained model, DistilBERT. with all parameters frozen. For this iteration we also experimented with a max sequence length for pre-trained model sequence length of 50 & 256.

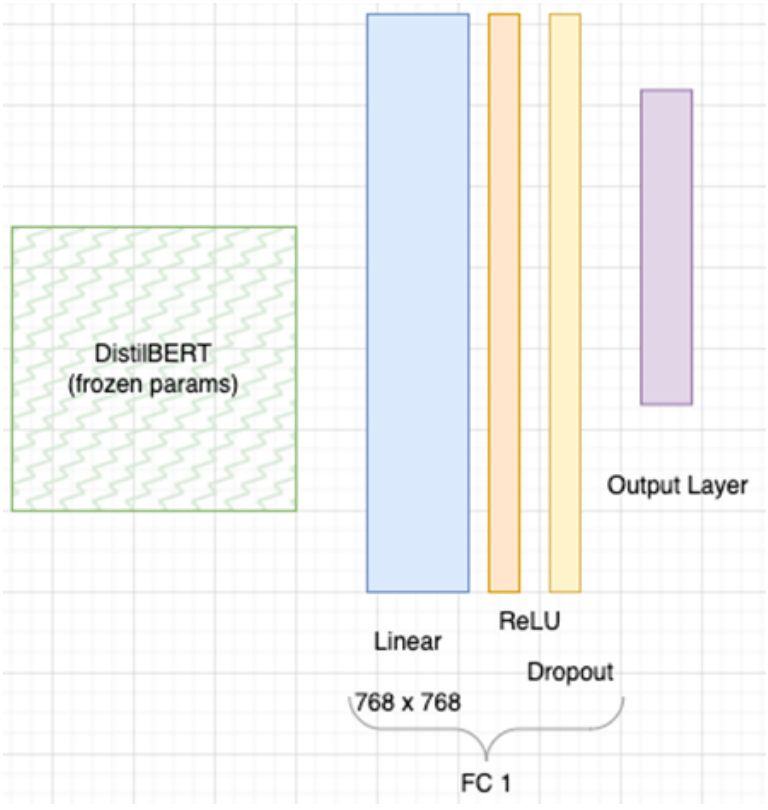
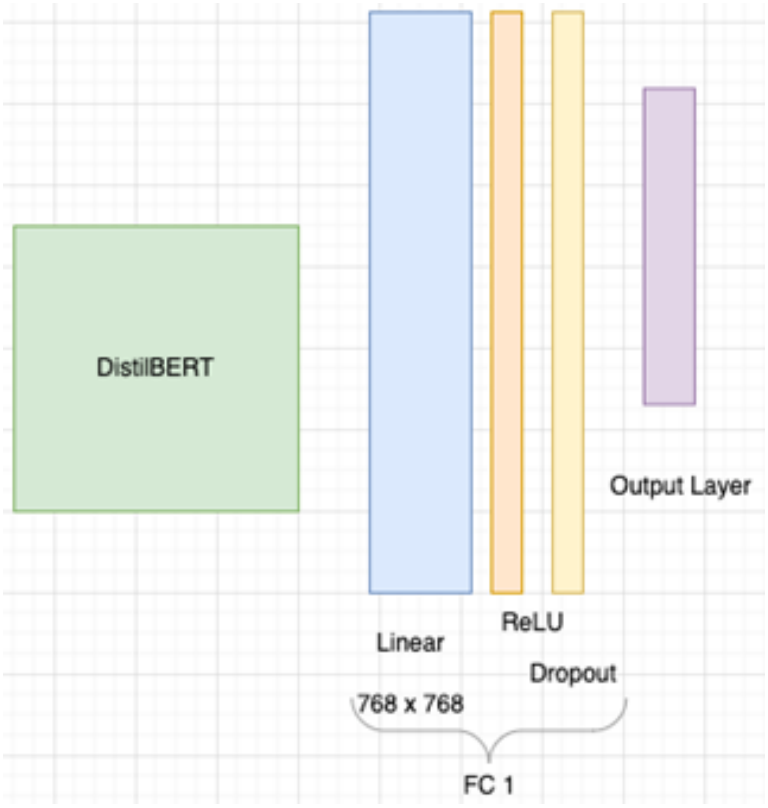


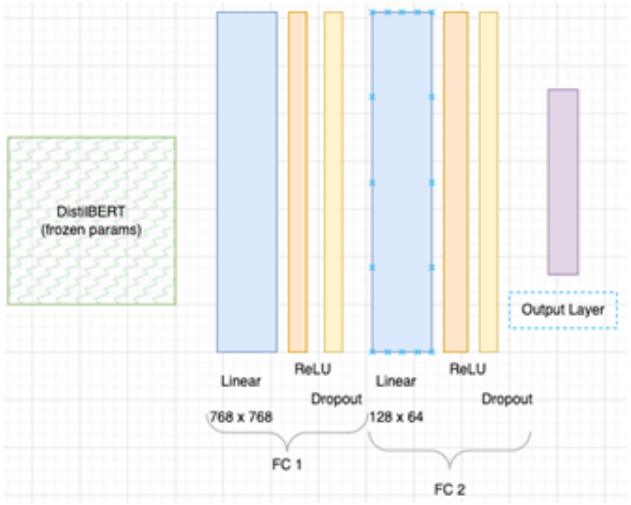
Figure: DistilBERT Architecture 1FC layer

This gave us a faster training time since only the FC1 layer and the output layer were being trained. Which between them had around 620k parameters. For the number of epochs we just ran 3 epochs as the running times were significantly large (1 epoch taking around 40+minutes). These experiments produced around 0.64 and 0.58 accuracy for max sequence length of 256 and 50 respecitvely.

The second experiment iteration we experimented with is a DistilBERT pretrained (params **NOT frozen**) model with 1FC layer at the and finally an output layer. Although this experiment took longer to train we achieved our best result using this architecture. But the down side for this experiment was that it had significantly larger running time (2h10m). On top of the 620k params between the FC1 and output layer another 66m params were being fine-tuned as part of the training process because we didn't froze the parameters. This was the best performing setup we were able to achieve.



Next, we wanted to see if adding one more FC layer could help our performance. This experiment setup is similar to the 1st iteration we had, but with a just a slight modification. We added another FC layer with 128 nodes. This increased the total learnable parameters to 680k. Aside from the slight improvement of F1 score this model architecture didn't improve upon the accuracy of its close sibling with similar config but just 1 FC layer



Finally, we also experimented with one of the largest language models that are out there. GPT2, this model has around 1.5 billion trainable weights. And off course we did this experiment with all the layers of the pre-trained model frozen. But this experiment didn't produce the results we were looking for, the training accuracy was around 16%, and we couldn't run the test as we ran out of GPU memory.

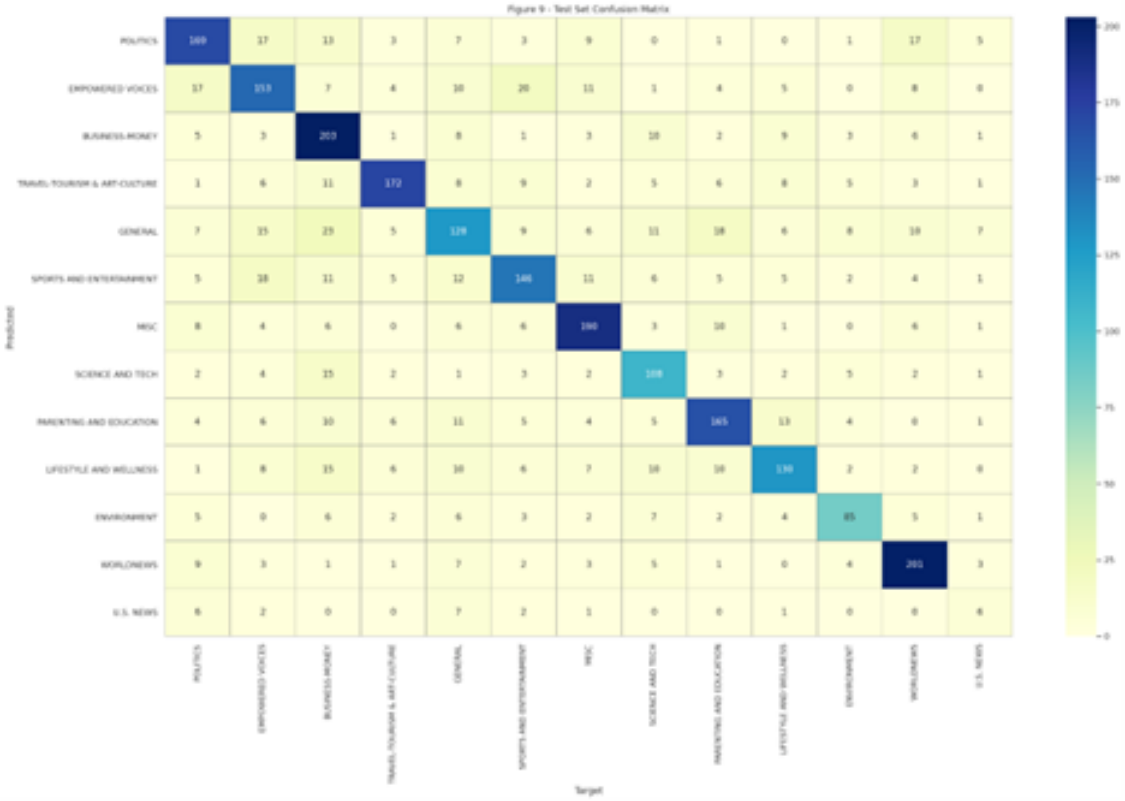
```
RuntimeError: CUDA out of memory. Tried to allocate 2.93 GiB (GPU 0; 15.90 GiB total capacity; 12.24 GiB already allocated; 2.49 GiB free; 12.68 GiB reserved in total by PyTorch)
```

Summary of results

Results from all our experiments are summarized here. We also have included a confusion matrix to better understand our models performance. Some useful observation

- Some categories such as Business & money and empowered voices were being identified as General
- US News category, with the smallest percentage of datap, was being misidentified as politics and general almost as equal or more times than ground truth.

Model	Accuracy	F1Score
DistilBERT (weights Frozen) 1FC–SEQ_LEN-256	0.64	0.62
DistilBERT (weights Frozen) 1FC–SEQ_LEN-50	0.58	0.54
DistilBERT 1FC–SEQ_LEN-256	0.69	0.69
DistilBERT (weights Frozen) 2FC–SEQ_LEN-256	0.58	0.57
GPT2 (weights Frozen) 1FC-SEQ_LEN-256	0.16 (train)	



Performance of the system and Challenges

We took our time with the dataset selection as we wanted to explore working with a different format of data in contrast to the previously worked upon numeric and image dataset. The aspects of NLP related tasks in this project brought up some learning curves for us and it was a good learning opportunity for us. Another significant reason why we chose this was the size of the dataset with a rather massive 210K entries.

As previously discussed, the best performance we observed with this system was with the DISTIL-BERT model whose parameters were all trained and we consolidated the target variable categories from 42 values to just 13 values. Multiple other possibilities of this model could be tried by varying the number of fully connected layers, nodes in these layers. During our attempts in running these iterations we faced a number of issues in terms of time taken, memory and computational resources required. Since the model had around 66M parameters to train, these issues occurred often and during some attempts, we saw the program crash too. Sometimes, they ran for hours, sometimes, we ran out of memory, so the number of iterations were somewhat limited

In terms of performance, since some categories had larger cardinality and some much much smaller, there was a bias introduced due to this imbalance whose impact could be observed even in the confusion matrices generated. Members of smaller groups were often misclassified as those belonging to those with larger membership.

In the previously published work on the same problem statement, they have used a variant of this dataset. Instead of 2012 - 2022, in this iteration, the dataset used was from 2012-2018. During our exploratory phase, from the distribution of categories through the years we observed that article categories changed from primarily being wellness centric to more political articles. Maybe this variation in the dataset introduced later present in the data that we have used here could be a reason for the difference in the performance.

Based on the training set and testing set accuracies that we observed, and therefore their error rates, we see that there is no significant difference between the two. So we need to consider if the dataset is even exhibiting some pattern which can well-defined by a function. For this low-variance issue, a good alternative to try would be using a stronger model for training and testing.

Reflections

As already discussed in the previous section, one of the biggest challenges came in terms of time and resources available. Despite using GPU T4 x 2 and Google Colab pro, we ran into resource and memory shortage issues. Although we intended to train some other NLP task specific models such as GPT or Stanford's GLOVE model, due to the limitations of our resources we had to limit ourselves to our current implementations. Once again, a lot of other variants of BERT based models could be implemented, but due to how long these take to train and time limitations at our end, we had to limit it to the above implementations.

Probably including more data into our input could better help in the task we had at hand. To that end, in some future implementation of this task, we could scrape data from the links of these news article and see how it improves the performance of the model.

Another interesting iteration of this setup, would be to observe the model's performance in categorizing random sample texts that aren't necessarily news headlines and see how good a job the model can do in categorizing them.

Acknowledgements

The inspiration for the baseline model came from one of the notebooks published on Kaggle. Along the course of this project we have referred many other blogs and HuggingFace discussion forums and stack overflow posts. While not exhaustive, we would like to specially acknowledge the works done in the following resources, which have guided us in this project.

1. https://medium.com/@samia.khalid/bert-explained-a-complete-guide-with-theory-and-tutorial-3ac9ebc8fa7c
2. https://huggingface.co/blog/bert-101
3. https://huggingface.co/docs/transformers/model_doc/distilbert
4. https://paperswithcode.com/method/distillbert
5. https://medium.com/huggingface/distilbert-8cf3380435b5
6. https://medium.com/geekculture/hugging-face-distilbert-tensorflow-for-custom-text-classification-1ad4a49e26a7
7. https://analyticsindiamag.com/python-guide-to-huggingface-distilbert-smaller-faster-cheaper-distilled-bert/
8. https://www.analyticsvidhya.com/blog/2022/11/introduction-to-distilbert-in-student-model/
9. https://www.kaggle.com/datasets/rmisra/news-category-dataset
10. https://www.kaggle.com/code/alfarias/huffpost-news-classification-with-distilbert
11. https://www.kaggle.com/code/juria99/pytorch-how-to-use-gpu/notebook
12. https://stackoverflow.com/questions/65804689/with-bert-text-classification-valueerror-too-many-dimensions-str-error-occur
13. https://discuss.huggingface.co/t/valueerror-too-many-dimensions-str/16775
14. https://www.kaggle.com/code/avikumart/nlp-news-articles-classif-wordembeddings-rnn
15. https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270
16. https://www.kaggle.com/code/derinrobert/newsclassification-using-lstm-val-acc-72