Fine Tuning LLM for text summarization of news articles

Amogh Sinha

This document serves as a step by step guide of my process in fine tuning a LLM for text summarization which was given as an assignment for Arrowhead.

Dataset:

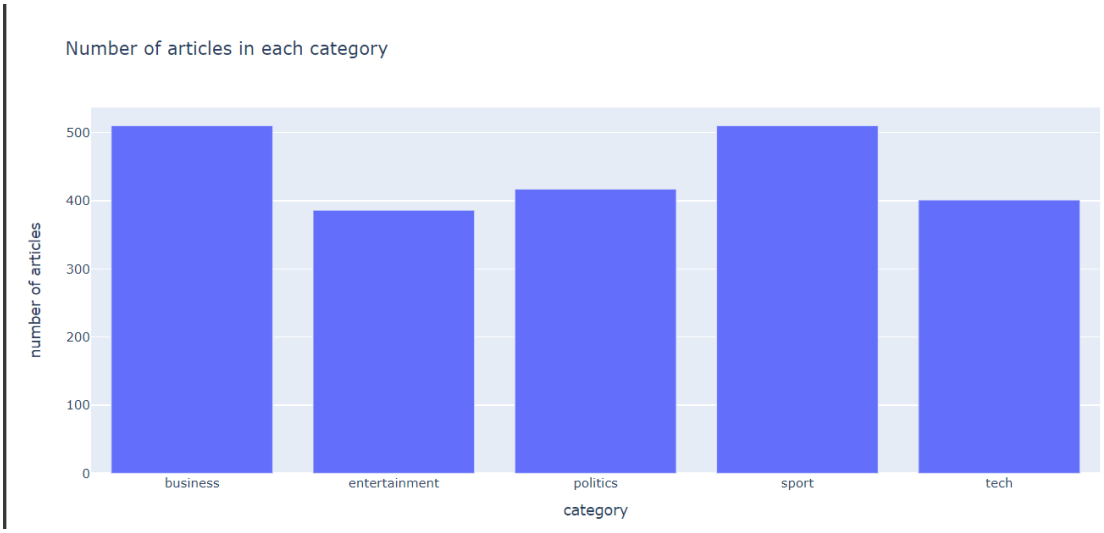
The dataset provided to use in this project was the BBC news summary dataset, available here on Kaggle.

The dataset was uploaded to the colab directory and was worked upon. There were 2224 rows in the dataset, with each row containing the article and summary.

Data Pre-processing:

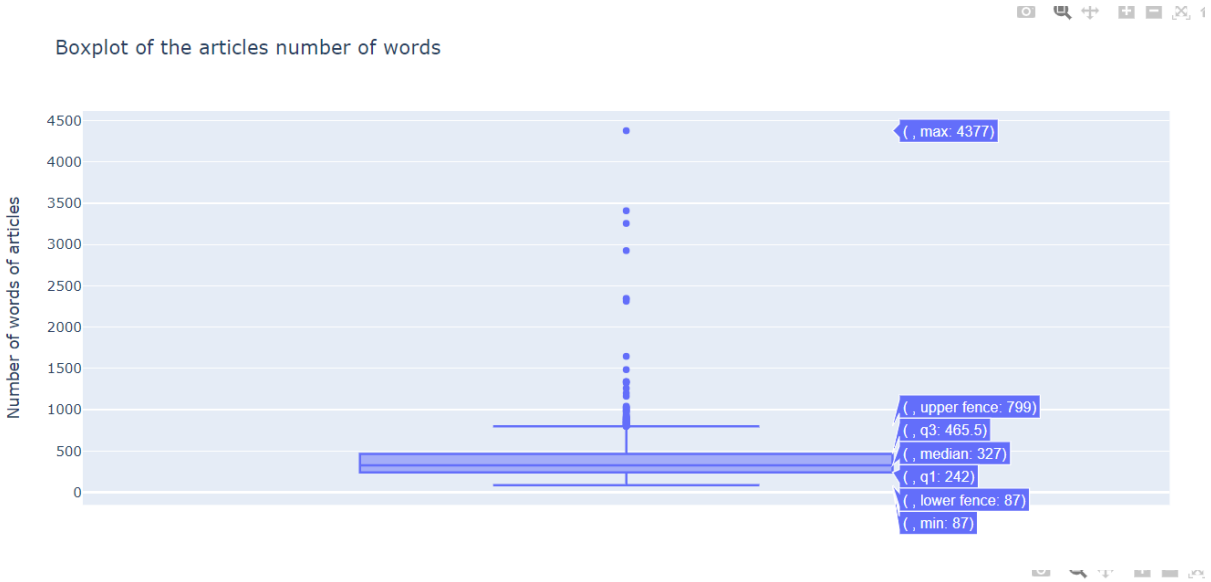
Each row was added with 3 columns. One each containing the number of words in the article and summary present in the row, and the third would be the category of the news.

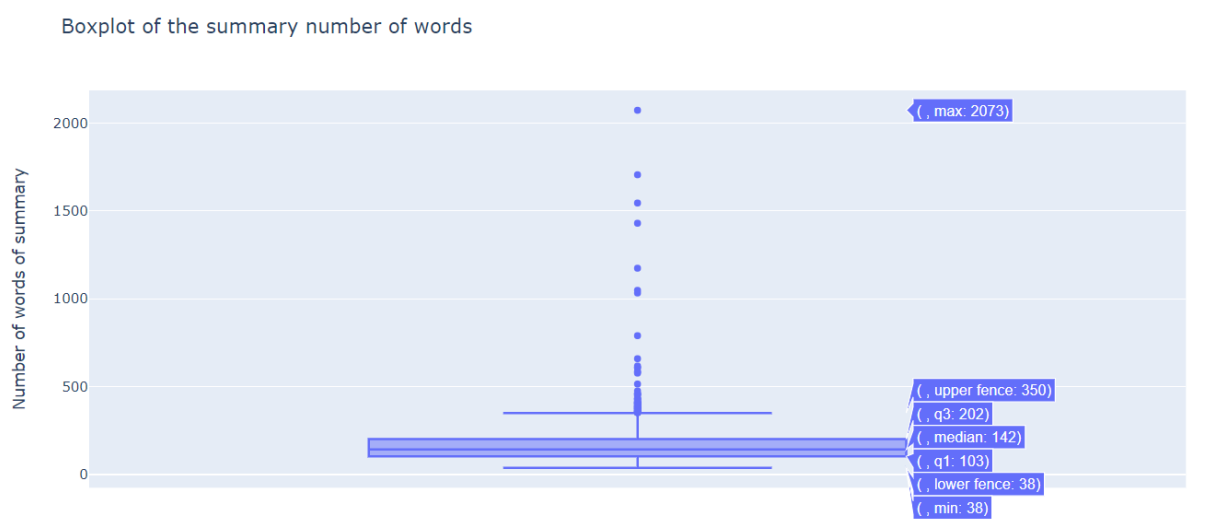
We see from the below graph that all categories have roughly the same number of articles:



	wordcount_article	wordcount_summary
count	2224.000000	2224.000000
mean	379.286871	165.151529
std	235.081995	108.678038
min	87.000000	38.000000
25%	242.000000	103.000000
50%	327.000000	142.000000
75%	465.250000	202.000000
max	4377.000000	2073.000000

With the above table, it can be seen that some rows with extremely large numbers are skewing the data. These outliers can better be seen with a boxplot





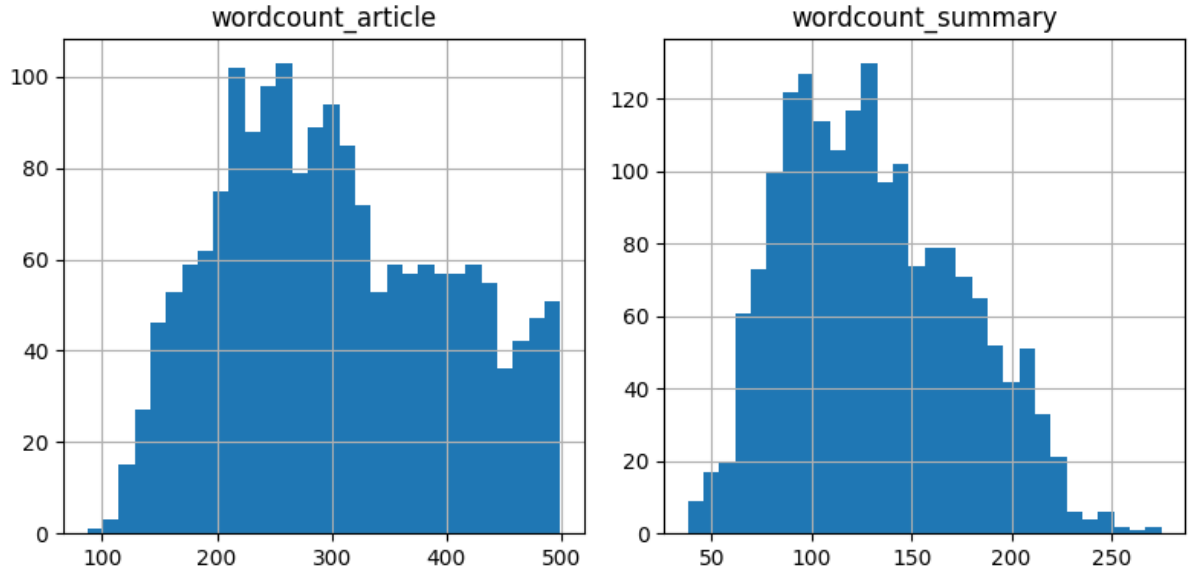
Looking at the word count distributon, we see there are some outliers. Our goal will be to have a decently uniformed data so we will choose the records where article word count is less than 500.

Doing this we see that now the number of rows have reduced to 1783, but the numbers are much more close now and are less influenced by outliers.

	wordcount_article	wordcount_summary
count	2224.000000	2224.000000
mean	379.286871	165.151529
std	235.081995	108.678038
min	87.000000	38.000000
25%	242.000000	103.000000
50%	327.000000	142.000000
75%	465.250000	202.000000
max	4377.000000	2073.000000

	wordcount_article	wordcount_summary
count	1783.000000	1783.000000
mean	301.922042	131.076276
std	97.529892	44.171673
min	87.000000	38.000000
25%	225.000000	96.000000
50%	291.000000	126.000000
75%	380.000000	164.000000
max	499.000000	275.000000

As can be seen with the below graph, the number of words now, more or less are in a normal distribution.



Currently in the dataframe, all the rows are neatly segregated by their category. So we shuffled them so as to not create any unknown bias.

To have the perform to the best ability, the dataframe was modified by doing the following steps:

1) Removing all the columns but just the news article and summary
2) Removing all stopwords from the dataframe
3) Converting all the letters into lowercase.

Model Selection and Training:

This step took the majority of time for me because it was a lot of trial, error and understanding of computational limits.

My original plan was to train on two models, one a small LLM and the other to be one of the recently launched ones, and then compare their results.

- Initially I tried to train on the newly launched Mistral 7B model since I thought it would be extremely useful in this. However, the model could not just train on the free version of Google colab. This would be a recurring problem for me
- I had also tried to quantise the model using peft and lora but the model was not just training well and training on my personal computer, for all intents and purposes, impossible.
- I then tried to use the Bloom-3B model that was launched last year and thought that it's small size could help me. But after reading some of the documentation and the FAQs and questions on it's page, I soon realized that this model is not well suited for the task of summarizations and it was not able to do the task even after multiple attempts at different prompts. So I decided to drop this idea and go straight to the basics.
- I understand the BART is one of the most basic LLM out there and nowhere as powerful as the others, but trying to solve the above two had taken a majority of my time and I needed to complete this task.
- BART has a successfu history at doing the task of summarization, and it is the only one which was able to work on my PC

However even that was possible only after taking a sample size of 30% of the total data. If I try to train on the entire dataset, I get the below result:

Thus I used the Bart model.

Metrics Used:

The most common metrics used for text summarization are BLEU and ROUGE scores. Here, I decided to use the ROUGE score

BLEU and ROUGE are both metrics that use n-gram overlap to measure similarity between machine-generated output and reference translations or summaries. ROUGE is used for text summarization tasks. It focuses on recall rather than precision. ROUGE is a modification of BLEU.

Rouge measures recall: how much the words (and/or n-grams) in the human reference summaries appeared in the machine generated summaries.

Hence, I decided to use ROUGE here.

Evaluation:

The Rouge score we have gotten is considered a good score. So it proves our model is working and working well

```
ROUGE-1 Precision    0.570564
dtype: float64
```

Next Steps:

The model and the process could be a lot better than it right now. Some of the things to make it better in the next steps include:

- Try to figure out better quantisation techniques to make mistral work.
- Dolly v2 mistral is something that looks promising for low computational devices and has 2B parameters.

- Improve the hyperparameters and tune it to get the most optimised performance

---

Thank you