Deep Learning Part 1

Section 2

Fine Tuning a Language Model

The model finetuning is required so that the model can give a suitable heading for a paragraph.

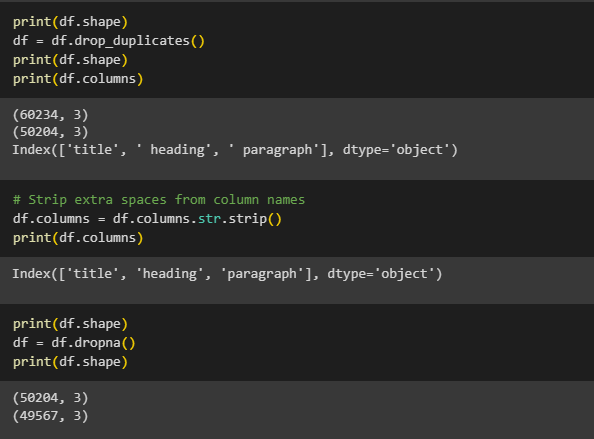
To do this we had to use a pretrained model called Longformer Encoder Decoder (LED) and fine tune it to generate the headings.

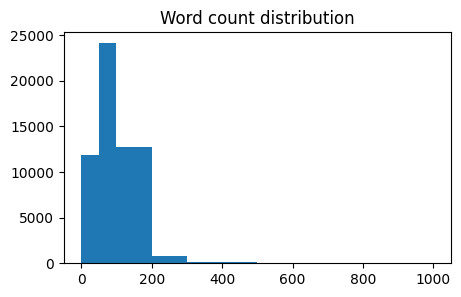
The Longformer Encoder-Decoder (LED) is a transformer-based model specifically designed to handle long sequences effectively. It's particularly well-suited for tasks like summarization, question answering, and text generation, where dealing with lengthy inputs and outputs is crucial.

**1. Load Data:**

To effectively preprocess the CSV data for analysis, start by using Pandas to load and clean the file. This includes removing duplicates, handling missing values, and addressing outliers to ensure data quality. Calculate the word count for each paragraph and filter out entries that fall outside of predetermined thresholds to maintain data relevance and manageability.

Once the data is cleaned, you'll have a well-structured dataset ready for analysis or model training. This preprocessing step is crucial for ensuring that the data is both accurate and representative, which in turn helps in building a more effective model or conducting a more reliable analysis.



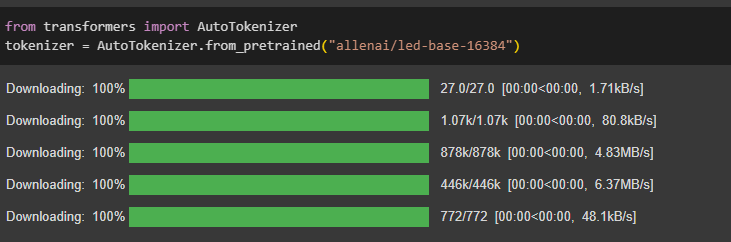


This is a positively skewed distribution system so we had to remove the tail end outliers that are present in this graph. And therefore log transformation is not much needed for this case.

**2. Tokenization:**

To prepare the text data for the LED model, start by initializing the tokenizer specific to the model. This tokenizer transforms the text into numerical tokens, which are necessary for the model to process and understand the input. By applying the tokenizer to your text data, you can ensure that the sequences are converted into a format suitable for the model.

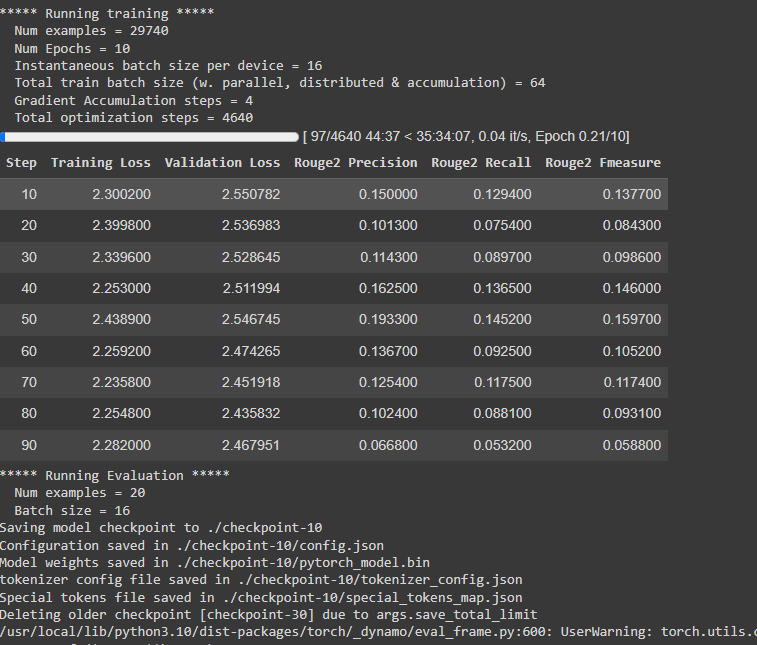
Next, make sure to pad and truncate the sequences as needed to maintain consistent input lengths across all data. This step ensures that the model receives input sequences of uniform size. Additionally, create a global attention mask to direct the model’s focus on the relevant portions of the input sequence, improving the model’s ability to generate accurate outputs.

****

**3. Training Model:**

To train the LED model for heading generation, begin by loading a pre-trained LED model and then fine-tune it using your specific dataset. This involves adjusting the model to better perform on the heading generation task by exposing it to relevant examples from your data. Fine-tuning helps the model adapt its weights to the nuances of your dataset, improving its ability to generate suitable headings.

While training, experiment with various batch sizes and other critical parameters such as learning rate and number of epochs. Finding the right balance for these parameters is key to optimizing the model's performance. Larger batch sizes can speed up the training process but may lead to higher resource usage and potential overfitting, while smaller batch sizes might offer better generalization but require more training time. Carefully tuning these parameters will help you achieve the best results while managing training efficiency and resource consumption effectively.



**4. Data Splitting:**

To train the LED model for heading generation, begin by loading a pre-trained LED model and then fine-tune it using your specific dataset. This involves adjusting the model to better perform on the heading generation task by exposing it to relevant examples from your data. Fine-tuning helps the model adapt its weights to the nuances of your dataset, improving its ability to generate suitable headings.

While training, experiment with various batch sizes and other critical parameters such as learning rate and number of epochs. Finding the right balance for these parameters is key to optimizing the model's performance. Larger batch sizes can speed up the training process but may lead to higher resource usage and potential overfitting, while smaller batch sizes might offer better generalization but require more training time. Carefully tuning these parameters will help you achieve the best results while managing training efficiency and resource consumption effectively.

