Increment -1

Text summarization using pre-trained language models by fine-tuning.

NLP CSCE5290

Group:

Brijesh Rao Pamara

Rohith Reddy Bollareddy

Amrutha Veeramachaneni

Nikhil Reddy Vemireddy

Introduction:

GPT-2 (Generative Pre-trained Transformer 2) is a state of art transformer model created by open AI which implements a decoder-only transformer architecture based neural network that uses attention rather than recurrent and convolutional based models. This mechanism allows the model to focus on most relatively significant parts which allows increased parallelization rather than other models.

The main purpose of this project is to summarize the overall content of the text to a meaningful text which is relatively smaller than the original text but preserves its overall theme and meaing. Before the advent of deep learning, the approach to Natural Language processing was mostly rule based with simple Machine Learning algorithms. But, today, with deep learning and topics of Natural language processing enabled models to be trained and analyze the text in a non sequential manner increased greatly.

The attention mechanism that is used in GPT-2 proved to be extremely rewardful in drawing the generalized context of the text. In today's world, a lot of business entities take the reviews and surveys of their products and services which enables them to develop and modify according to the market and in customer interest. Text summarization comes in use to draw important and valid information.

Dataset:

The dataset is the amazon reviews corpus it consists of a multilingual dataset of reviews of amazon product reviews. We only use the English language data of the corpus. It consists of a training set of 20000 reviews and testing and validation sets of 5000 reviews each.
Each entry consists of 'review_id', 'product_id', 'reviewer_id', 'stars', 'review_body', 'review_title', 'language', 'product_category'.

Link to the dataset repository: https://registry.opendata.aws/amazon-reviews-ml/.

In this project we are only focused on the ‘review_body’ and ‘review_title’ columns of the dataset, as these are the fields which contain the text and title, which we use as the summary.

```
DatasetDict({
    train: Dataset({
        features: ['review_id', 'product_id', 'reviewer_id', 'stars', 'review_body', 'review_title', 'language', 'product_category'],
        num_rows: 200000
    })
    validation: Dataset({
        features: ['review_id', 'product_id', 'reviewer_id', 'stars', 'review_body', 'review_title', 'language', 'product_category'],
        num_rows: 5000
    })
    test: Dataset({
        features: ['review_id', 'product_id', 'reviewer_id', 'stars', 'review_body', 'review_title', 'language', 'product_category'],
        num_rows: 5000
    })
})
```

Summary of the dataset

Sample text from the repository:

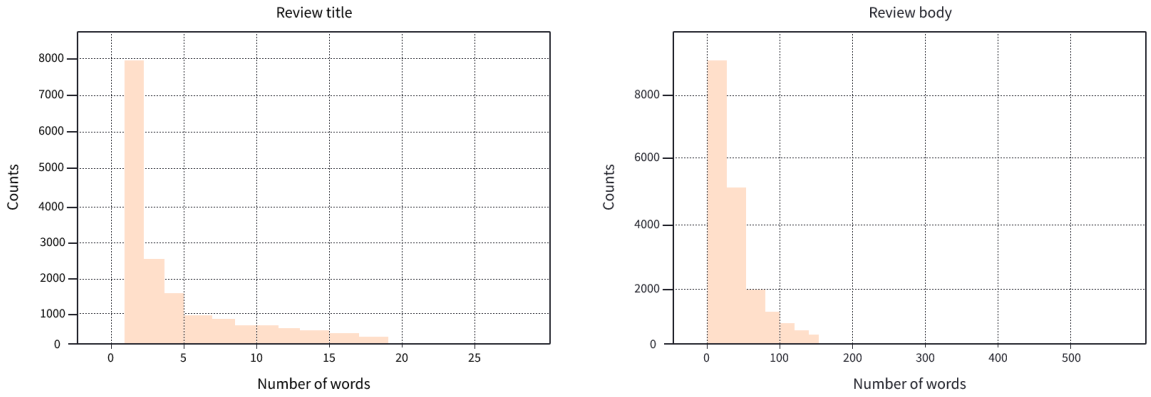
```
'>> Title: Never showed up'
'>> Review: Ordered it on a friday and it never came. I waited until the following Wednesday and still nothing. :( plus I can't get a refund because I have nothing

'>> Title: Trite. Patronizing. Shallow.'
'>> Review: Do yourself a favor and check it out from the library before buying this book. Wish I had. Trust your gut instinct. That's it. Nothing more.'

'>> Title: We ordered this book back in Sept and never received ...'
'>> Review: We ordered this book back in Sept and never received it. There is now no tracking information or any way to get ahold of the seller.'
```

Analysis:

As part of preprocessing the database, we perform basic analysis operations on the dataset, like cleaning the text of any special characters. We also analyse the distribution of the dataset to see if there is any imbalance of data. In the dataset we find that a significant amount of the summary in the product review are one or two words, if we train the model on this imbalance it will then we will have a clear bias in the output. Hence, we filter our dataset to remove any summaries which are less than two words. So that we can get higher quality summaries.

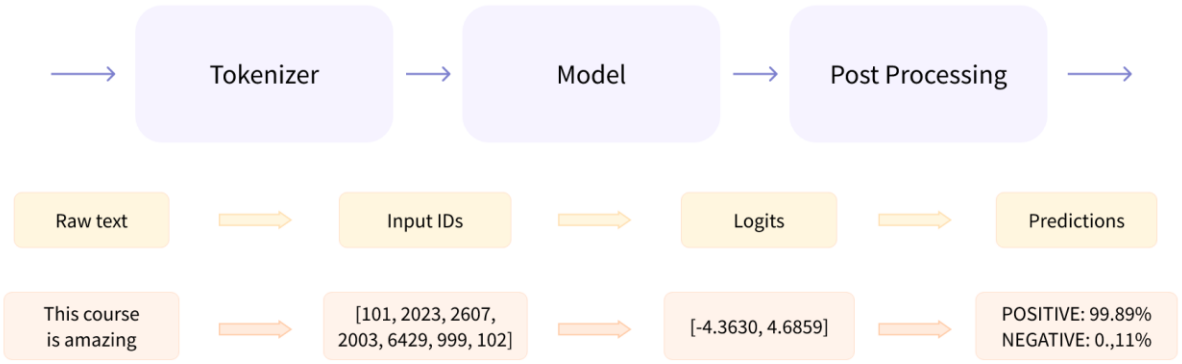


Features:

- Language models are probabilistic models that predict the next token in a sequence using the token that comes before it. These models learn the probability of occurrence based on examples from training.
- GPT- 2 model has only the decoder part of transformer networks that uses multi headed masked self attention which allows it to look at first n- tokens at a given time.
- Instead of processing tokens sequentially, GPT-2 processes it parallelly.
- The language model such as GPT-2 can easily be fine tuned as per our requirement.

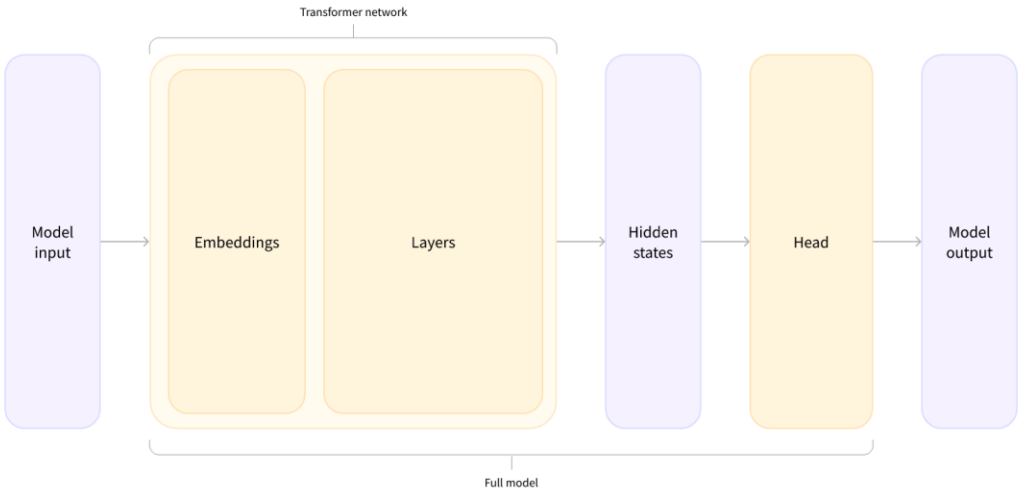
Implementation:

The project architecture is as follows:
The Raw text is first tokenized using the GPT2tokenizer. The tokenized text is then fed into a pre-trained model which we finetune using our dataset of reviews. Then additional post-processing is applied on the generated text which is used to clean the outputted text.
Then the generated text is compared against the expected output. The training loss is computed on whether the expected output and the generated output are similar as defined by a loss function. In our case we are using a cross-entropy loss function.



Project architecture

The below diagram shows the architecture of the transformer model, specifically the GPT2 model as these models only contain the decoder part, as they are a decoder only architecture of transformer models. The model input is first converted into the sentence embeddings and the passed onto the attention layers, where the attention mechanism attends to each token of the input text and computes the similarity of all the tokens with respect to each other. The outputs are then fed into a hidden layer consisting of multi-layer perceptrons which act as a feed forward layers and these then generate an output prediction.



Transformers model architecture

Preliminary Results:

ROUGE score abbreviated for **Recall-Oriented Understudy for Gisting Evaluation**, is a score used for evaluating automatic summarization of text in natural language processing. The metrics compare a summary produced by a model, against a reference summary, to evaluate its accuracy.

Accuracy score with baseline model:

```
[78] import pandas as pd

     score = evaluate_baseline(books_dataset["validation"], rouge_score)
     rouge_names = ["rouge1", "rouge2", "rougeL", "rougeLsum"]
     rouge_dict = dict((rn, round(score[rn], 2)) for rn in rouge_names)
     rouge_dict

     {'rouge1': 0.15, 'rouge2': 0.08, 'rougeL': 0.14, 'rougeLsum': 0.15}
```

Accuracy score with a pre -trained GPT model with no fine-tuning:

```
] ## generated score from the model
  def scores(generated_summary, reference_summary):
    scores = rouge_score.compute(
      predictions=[generated_summary], references=[reference_summary]
  )
    return scores

  scores(full, summary[1])

  {'rouge1': 0.10526315789473685,
   'rouge2': 0.0,
   'rougeL': 0.10526315789473685,
   'rougeLsum': 0.10526315789473685}
```

As we can see our baseline model has more accuracy than the pre-trained model with no finetuning on our dataset. In further work we will fine-tuning the neural network model on our data and compare their accuracies again.

Project Management:

Work completed:

Examined the dataset and its properties cleaned up and transformed the data into a form suitable for model creation.
Completed the preliminary data analysis and normalized the dataset by eliminating entries causing imbalance in the dataset to minimize the bias in the dataset, which could impact model performance.
Created a baseline model, using traditional NLP techniques based on n-grams to match against the GPT model.
Imported and built a pre-trained GPT model from the model weights and evaluated its performance with samples from the dataset to generate summaries.
Evaluated the baseline and GPT model using the ROUGE score, used to evaluate summarization.

Tasks and Responsibilities:

Data Preprocessing- Amrutha
Creation of baseline model and dataset parsing- Rohith
Transformer model creation and documentation- Brijesh
Evaluation of Model and Documentation- Nikhil

Work to be completed:

Train the gpt-2 model on our dataset to finetune it for our task, experiment with different learning rates and training schedules to find, which results in an optimized model.

Comparing the model to a baseline model using traditional NLP and seeing how the performance compares between the two.

References:

1. Sample Efficient Text Summarization Using a Single Pre-Trained Transformer
2. Language Models are Unsupervised Multitask Learners
3. The amazon reviews corpus

4. Hugging Face Transformers