Introduction

In this project, our main objective was to add the watermark algorithm to a summarization model and then use this model to summarize a dataset. We wanted to see how the summaries would look with and without watermarks.

Our main aim was to run the summarization model on the dataset while adding different types of watermarks, each with its own parameters.

Article (model input)	No Watermark (NW)	Watermarked (W)	NW Fraction of T in G list	W Fraction of T in G list	NW z score	W z score	NW ppl	W ppl
New York (CNN)When Liana Barrientos was 23 years old, she got married in Westchester County, New York. A year later, she got married again in Westchester County, but to a different man and without divorcing her first husband. Only 18 days after that marriage, she…	Liana Barrientos, 39, is charged with two counts of "offering a false instrument for filing in the first degree" In total, she has been married 10 times, with nine of her marriages occurring between 1999 and 2002. She is believed…	She has married up to eight times over the course of a 16 year span, prosecutors say. Her husband has been sent home by the FBI. She was arrested last year for allegedly stealing trains in the New York metro…	25.0%	92.5%	0.52	12.8	1.97	3.57

In this experiment we set z_threshold 4.0 and the detector recognize both watermarked and non watermarked true, We set δ = 4.0 and γ - 0.25 and multinomial sampling with BART model.

In this report, we discuss the code structure, referencing the notebook. We also address the challenges encountered during this project. In our experiments, we aim to get information based on metrics of the summarized text, the parameters of the watermark processor, and the results of the detection process. In addition, we draw several plots based on these data points and provide interpretations for these visualizations.

Thecnical Details

Initially, the project utilized the BART model. However, due to limitations in resources, the project shifted to a more lightweight and straightforward LLM, specifically, the T5 model.

In this project, we incorporated the CNN dataset, which includes a highlights column providing well-crafted human-generated summaries for each piece of data. This highlights column serves as a valuable benchmark for assessing the quality of the generated text summaries.

In this project, we also used google colab with T4 GPU to download and run datasets and models.

Code Structure

We have organized the code cells into sections in the notebook, and below, we provide a description for each of these sections.

- Watermarking
  The LM-Watermarking watermark processor class code has been integrated into the project in this section. Additionally, the code includes the necessary classes for homoglyphs, normalizers, and PRF schemas to support the watermarking processor.
- Generator
  The Generator is a straightforward function that takes a prompt, all the required metrics, a model, and a tokenizer as inputs. It then provides both summarized text with a watermark and summarized text without a watermark as outputs using watermark generator class.
- Detector
  The “Detector” section accepts text along with all the necessary parameters and executes the watermark detector on the input text using those parameters.
- Models
  In the “Models” section, we import the necessary models (BART, T5) and tokenizers from the transformers library. Additionally, we include the LogitsProcessor to enable the passing of the WatermarkLogitsProcessor to the generator.
- CNN Dataset
  Within this section, our action is to import the CNN Dataset. We restrict our usage of this dataset to a subset, specifically, a portion of the training section, in order to implement watermarking.
- Metrics
  Our main goal of using these metrics is to messure quality of generate summerized text. The first metric, as mentioned in the paper is **perplexity** (PPL), and the second metric is **Rouge**. I found it in “Watermarking Conditional Text Generation for AI Detection: Unveiling Challenges and a Semantic-Aware Watermark Remedy” paper. The use of the Rouge metric is common in the evaluation of automatic summarization and machine translation, however, ppl is not specifically for text summerization.
  The library we utilize to obtain the Rouge metric provides us with the evaluation metrics in three forms: precision, recall, and F-measure.

- Apply Watermarking on T5
  In this section, specifically in “Generate and Detect for CNN dataset” sub section, we iterate through the dataset and execute the generate function with various

parameters. Additionally, we run the detect function and compute metrics for the generated results. In the end, we store these results to facilitate their utilization in experiments section.

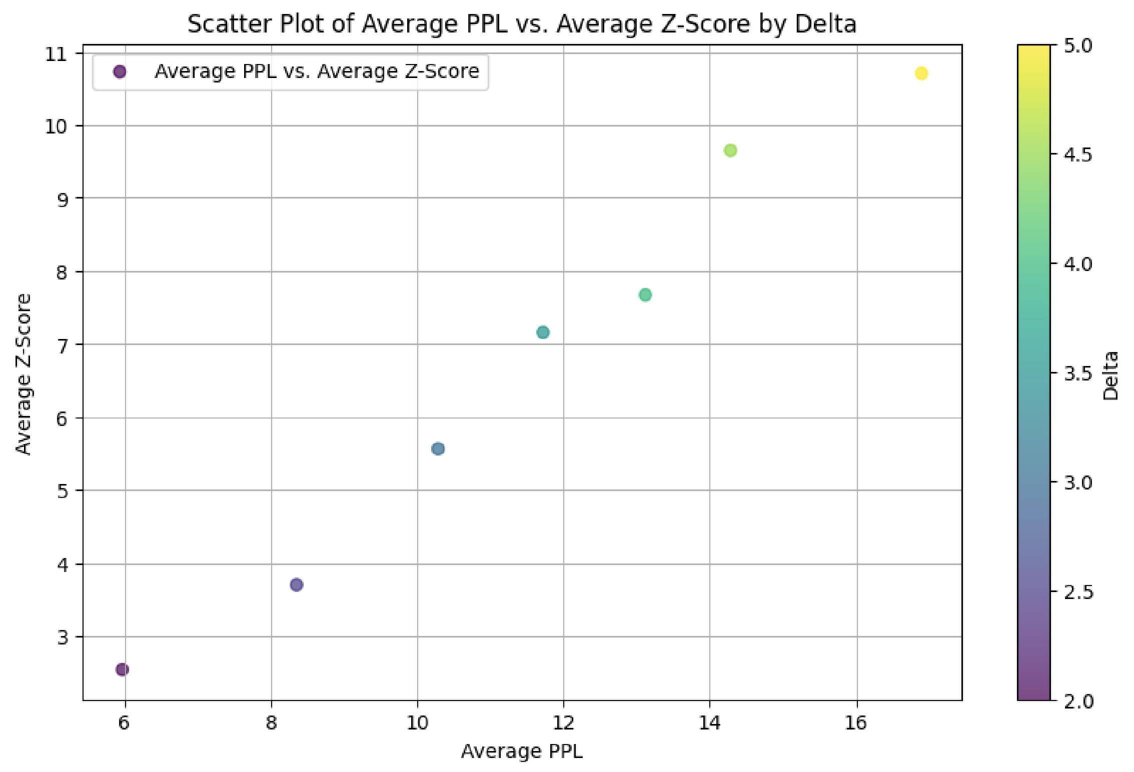
- Visualization

  In the visualization section, we load the results and attempt to visualize the data to extract valuable insights. Each plot and extracted insights described in the Experiments section in this report.

Experiments

Due to a limits of resources, we were unable to conduct experiments with all potential parameter distributions. Therefore, we kept certain parameters constant throughout, including ɣ = 0.25, multinomial sampling, z_threshold = 4.0, and output_max_length = 500. We do the experiments for delta in the range of 2.0, 2.5, 3.0, 3.5, 4.0, 4.5.

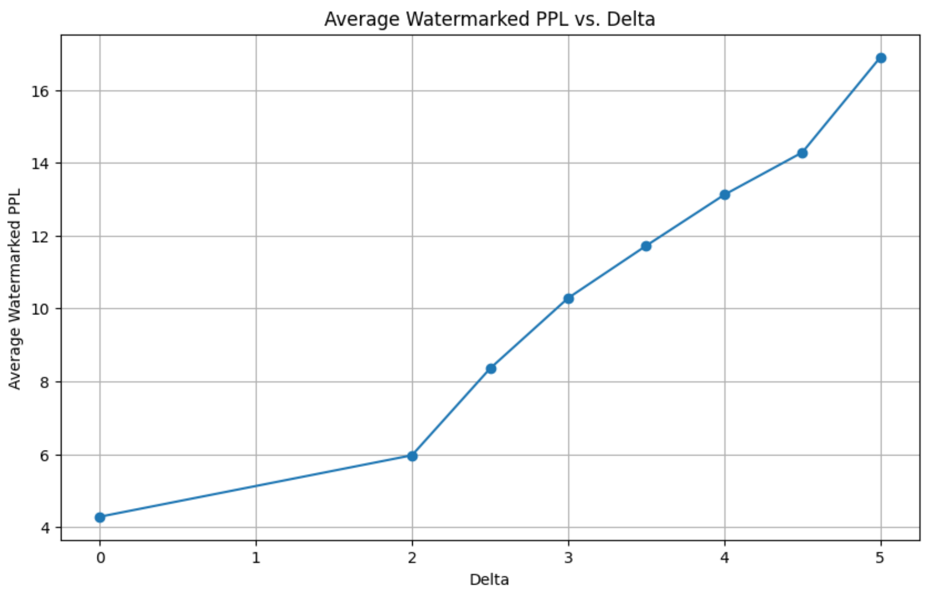
The initial result we evaluate in this project is the correlation between perplexity (PPL), Z-score, and delta. As we increase delta, the generated text becomes more restricted (with delta = ∞ representing hard watermarking). Therefore, the quality of the generated text is expected to decrease. In the results, we should observe that perplexity (PPL) increases as delta increases. Also the Z-score is expected to increase as the delta value is raised. This is because as the delta value increases, the generated text becomes more obviouse as AI-generated, making it easier for the detector to identify, resulting in a higher Z-score.



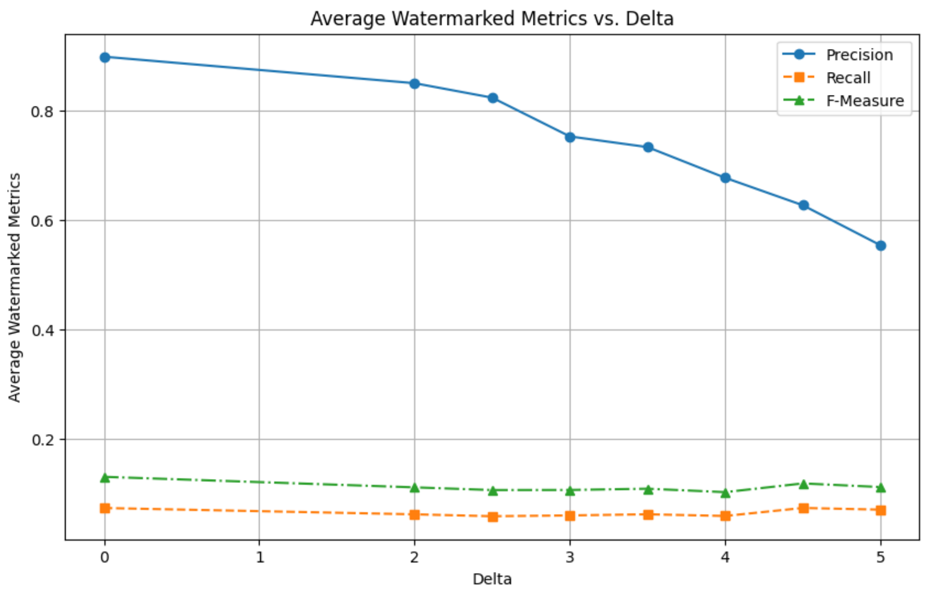
As you can see, our results align with our expectations: higher delta values results higher z-scores and perplexity (PPL)

In this plot, we can clearly observe the correlation between delta and perplexity (PPL). Additionally, it allows us to compare the perplexity of the unwatermarked text (zero delta) to the watermarked text. Notably, the perplexity of the unwatermarked text is significantly lower than that of the watermarked text, underscoring the impact of watermarking on text quality.

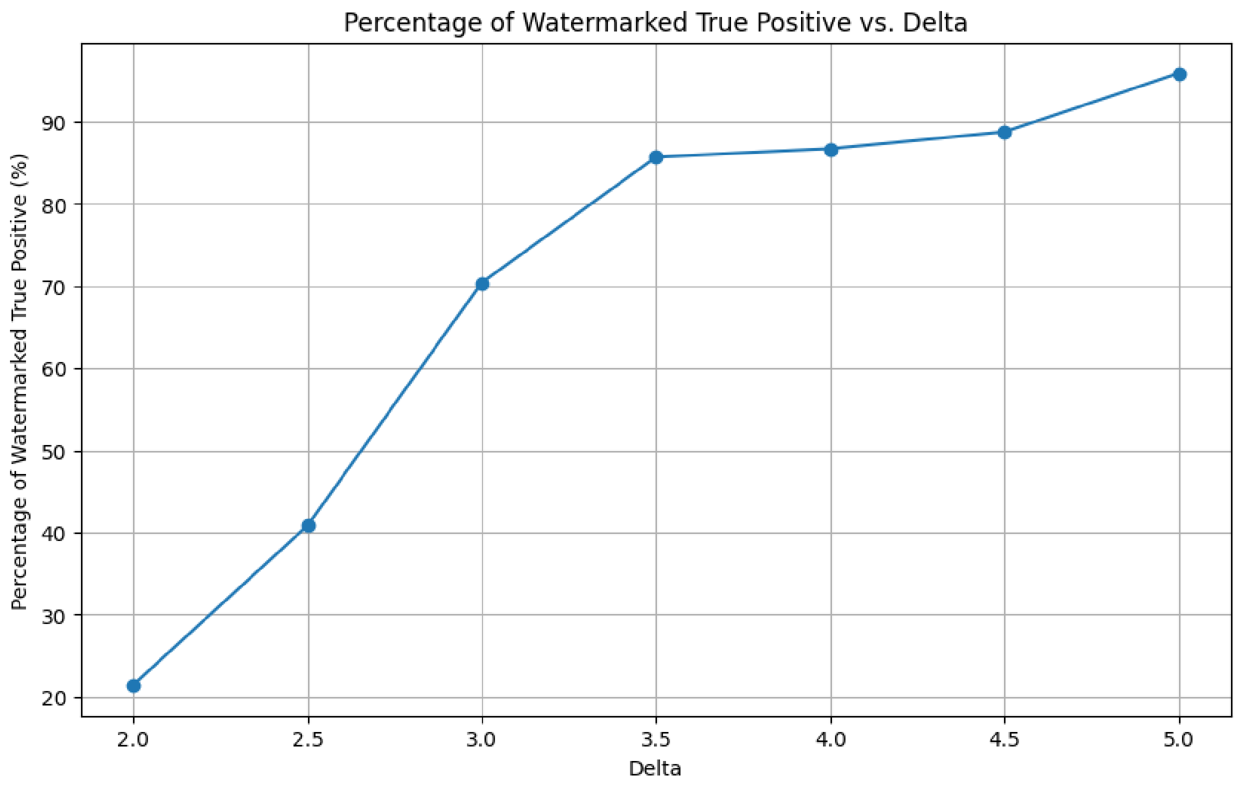
.



The findings in the last plot are similar to those observed in the Rouge evaluation as well.

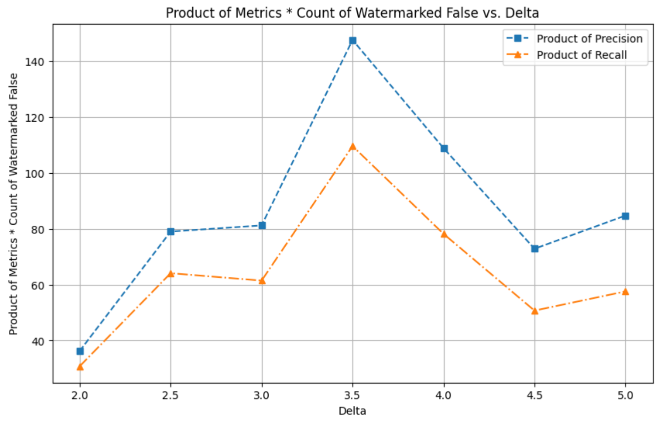


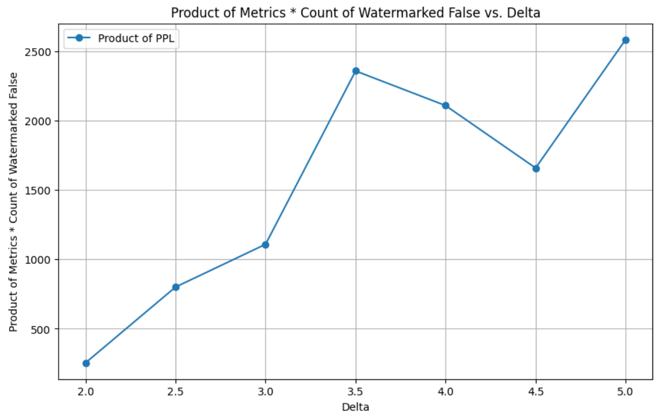
We also can see the impact of delta on the percetage of watermarked text detected as a AI-generated text.



* For the unwatermarked there was no false result in our experiments.

An optimal delta can be determined by multiplying the percentage of true positives with the metrics. So we normalize the percentage of true positive ppl, rouge 1 precision, and rouge L percision.





The left plot suggests that a delta value of 3.5 is optimal, as it yields both an acceptable Rouge metric and a high true positive percentage. However, the right plot does not provide a clear indication of an ideal delta value based on perplexity (PPL). This reason could be depend on several factors, such as a limited amount of data or the general applicability of PPL (as Rouge is specifically designed for text summarization).

Conclusion

The primary challenge we encountered in this project was the considerable time it takes for a Large Language Model (LLM) to summarize a single text. To summarize a single text of approximately 500 tokens using the BART model, we require approximately 3 minutes with a Google Colab GPU. Therefore, it is almost impossible to run the model on a dataset. These circumstances force us to use “T5 small”, which reduces the required time to around 40 seconds. However, BART results was significantly better than T5 small.
The lack of appropriate resources, cause that we could not execute our method on more than 100 data. Additionally, we maintained several variables unchanged, including gamma, z_threshold, and the sampling algorithm. Consequently, we were unable to observe the impact of watermarking on the model by varying these variables.
However, our experiments clearly demonstrated that the watermarking algorithm was detectable in more than 85 percent of the summarized text. Importantly, it did not reduce the quality of the model's results, establishing the practicality of this algorithm in the context of summarization.

References

A Watermark for Large Language Models

LM-Watermarking source code

Watermarking Conditional Text Generation for AI Detection: Unveiling Challenges and a Semantic-Aware Watermark Remedy

BART Large Language Model

T5 Large Language Model

CNN Dailymail Dataset

PPL calculation library

Rouge calculation library