patterns and statistical characteristics of input text, comparing them to those of LLM-generated text [31, 36]. For example, DetectGPT calculates the average log probability ratio of the input text over its perturbations and classifies text as LLM-generated if the ratio exceeds a threshold [25]. Zero-shot classifiers do not require further training, thus facilitating usage from non- technical users. Watermark detectors rely on the addition of a watermark, which is not visible to humans, on LLM-generated text. [17] proposed a watermarking scheme with tokens in "green list" and "red list." While adding the watermark on LLM-generated text, the use of "green list" tokens are prioritized in the sentence generation process, resulting in a text dominated by these tokens. The detector thus classifies a text to be LLM-generated if the number of "green list" tokens is high. Strong and weak watermarking can be implemented by adjusting the parameters [17]. These 2 types of detectors both exhibit remarkable performance in LLM-generated text detection.

For zero-shot classifiers, 95% accuracy is observed from RoBERTa fine-tuned classifiers [33] and 92% from GROVER [37]. Apart from accuracy, AUROC is also considered to account for the true positive rate (TPR) and false positive rate (FPR), as false positive is highly discouraged for these classifiers to misclassify human-written text as LLM-generated [17]. DetectGPT shows state-of-the-art performance with an average AUROC of 95.3% on various datasets. It should also be noted that detection performance varied with text length and decoding strategies [33, 25]. Meanwhile, watermark detectors also show outstanding performance, with strong watermarking detection achieving 100% AUROC and soft watermarking achieving 98.9% [17]. Watermark detectors are also regarded as more effective detectors than zero-shot classifiers [19]. Other than the above 2 mainstream LLM-generated text detectors, [19] proposed an information retrieval-based detector, which classifies text by comparing the input text with stored LLM outputs.

Paraphrase attacks in LLM-generated text detection. While existing LLM-generated text detectors show remarkable performance, they can be vulnerable to paraphrasing attacks, as one can alter generated texts to circumvent detection. Since these detectors perform classification based on the existence of token patterns or watermarks, paraphrasing on LLM-generated text, either by human or AI paraphrasers, can potentially evade the detectors [17, 31, 36]. Past research has conducted experiments on different types of paraphrasing attacks. [17] performed the attack by replacing words with tokens generated from T5 model and noticed a significant watermark degradation with AUROC decreasing from 99.8% to 69.6%. [19] showed that after using DIPPER AI-paraphraser, DetectGPT's detection rate significantly reduced from 70.3% to 4.6% and watermark detection accuracy decreased from 100% to 57.2%. The aforementioned research performs only a single round of paraphrasing, and it is sufficient to substantially reduce the detectors' performance. Noticing this, [31] performed recursive paraphrasing attacks on various LLM-generated text detectors, where LLM-generated text is paraphrased with AI paraphrasers in multiple iterations. They concluded that recursive attacks could further evade these detectors. For non-watermarked text, DetectGPT's AUROC score decreases from 96.5% to 59.8% with text paraphrased with T5 model. The same outcome also resulted in watermarked text. With 2 rounds of paraphrasing with DIPPER and Llama-2-7B-Chat [31], TPR@1%FPR drops significantly from 99.8% to 44.8% and 38.9% respectively. The score further decreased to 15.7% and 27.2% respectively after 5 rounds of paraphrasing, reflecting the detectors' inability to identify LLM-generated paraphrases. Other than the degradation in TPR@1%FPR, AUROC scores also decreased from 99.9% to 76.3% and 79.5% respectively. As such, from existing research it can be expected that paraphrasing attacks, which change the statistical properties and replace watermarked tokens, could effectively evade both zero-shot classifiers and watermark detectors while preserving semantic information from the original LLM-generated text [19].

2.4. Research Gap

In existing research, classifications are conducted with datasets consisting of human-written text and paraphrases of LLM-generated text. While LLM-generated text detectors make classification based on the existence of LLM-generated features in the input [31, 36], a potential reason for paraphrasing attack successfully evading these detectors might be the fundamental difference in the language format between LLM-generated text and its paraphrases. [30] stated that paraphrases, which contain similar semantic information, could exhibit different lexical, syntactic and word order from the original text. As such, the LLM-generated paraphrases might not contain the characteristics that these detectors are looking for, leading to misclassification.

Hence, to addressing the existing gap, our aim in this research is to include human-paraphrased text in the dataset for classification, which we experiment with state-of-the-art LLM-generated text detectors. LLM-paraphrased text might potentially distinguish itself from human-paraphrased text, in terms of the degree of semantic understanding, writing style and word choices. As a result, an investigation into the effect of including human paraphrases on LLM-generated text detection can help identify potential reasons for misclassification by existing detectors and to further aid research in developing LLM-generated text detectors.

3. Dataset

3.1. Creating the HLPC Dataset

Despite the existence of numerous datasets containing human- and LLM-generated data, we found no suitable dataset that incorporates also their paraphrases which would enable our study, and hence we leverage and extend existing datasets for creating the HLPC dataset.[1]

Overview We put together a dataset consisting of two types of data, each consisting of an original document (DOC) and its paraphrase (PP):

1. Human-generated data: collected from 4 existing datasets, which include MRPC, XSum, QQP and MultiPIT, and provide original human-written documents (H-DOC) and their paraphrases (H-PP).
2. LLM-generated data: original LLM documents (LLM-DOC) are generated by using parts of the H-DOC documents above as prompts sent to the LLM, and their paraphrases (LLM-PP) are generated through a paraphrases that runs five paraphrasing iterations on the LLM-DOC documents.

In what follows we further elaborate to describe the data creation process in detail.

Human-written and paraphrased documents (H-DOC and H-PP). To retrieve human-generated texts alongside their paraphrases, we make use of four different datasets, namely Microsoft Research Paraphrase Corpus (MRPC) [9], Extreme Summarization Dataset (XSum) [27], Quora Question Pairs Dataset (QQP) [35] and Multi-Topic Paraphrases in Twitter (MultiPIT-expert) [10]. MRPC consists of sentence pairs from newswire articles, while XSum dataset contains pairs of BBC articles and their summary. QQP dataset includes question pairs from Quora, and MultiPIT includes pairs of tweets from Twitter. Therefore, the combination of these data sources encompasses both clean and well-structured text data from authoritative news companies and noisy and unstructured text data from online forums. Since these datasets were published before the surge of generative AI, it is assumed that all texts are human-generated. As these datasets were originally used for paraphrase detection, pairs of sentences in these datasets are labelled as paraphrases (label = 1) or non-paraphrases of each other (label = 0), among which we are interested in those with a positive label for our purposes.

Document filtering. From the H-DOC and H-PP samples above, we remove cases of non-paraphrases, sampling only the pairs that are paraphrases to satisfy our objective of having human paraphrases for every human-generated text. Of the remaining pairs, we remove samples with fewer than 10 tokens (fewer than 30 tokens in the case of XSum due to its greater length), to enable prompt extraction for subsequent LLM text generation as described below, as well as those with more than 512 tokens due to restrictions of GPT-2 for text generation. Finally, we randomly sample 150 documents from each set, H-DOC and H-PP, with 300 documents remaining across both types.

We next proceed to generating the LLM texts and their paraphrases. Figure 1 illustrates the “LLM-generated documents (LLM-DOC)” and “LLM-generated paraphrases (LLM-PP)” generation process.

LLM text generation (LLM-DOC). To generate LLM-based texts that resemble the human-generated texts above, we first obtain prompts from the human-generated texts above, which are then fed to the LLM to generate new texts. The LLM-DOC generation process starts by taking the first 5 tokens from H-DOC (first 30 tokens for documents sourced from XSum) with a tokenizer, which are used as prompts for the language models in the generation process. The models then generate text based on the given prompt up to a length of the maximum length of the H-DOC. We generate two types of LLM-DOC, watermarked and non-watermarked, both using two transformer-based language models, GPT2-XL [33] and OPT-1.3B [39], along with their respective tokenizers. These models are pretrained with a wide range of internet text and therefore are suitable for text generation in this project. Default settings are used for parameter initialization for both models. Non-watermarked LLM-generated texts are output purely generated from the above two models. For watermarked LLM-DOC generation, watermarks are added by initializing a watermark in the logit processor of the models in addition to the LLM-DOC process [17].

[1] The HLPC dataset can be found at `https://github.com/kristylht/Human-LLM-Paraphrase-Collection-HLPC`

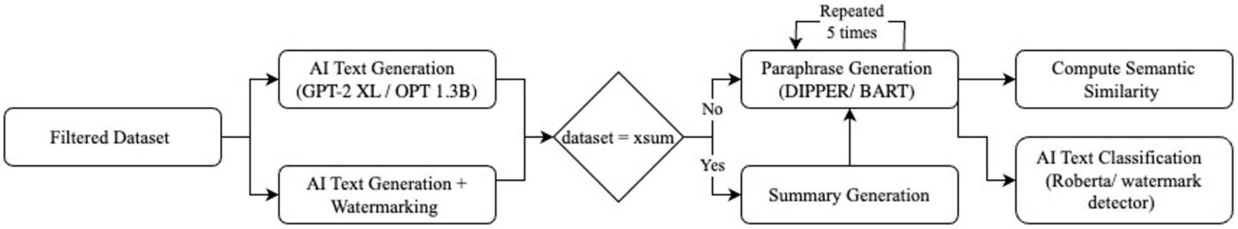
Filtered Dataset
AI Text Generation (GPT-2 XL / OPT 1.3B)
AI Text Generation + Watermarking
dataset = xsum
No
Yes
Repeated 5 times
Paraphrase Generation (DIPPER/ BART)
Summary Generation
Compute Semantic Similarity
AI Text Classification (Roberta/ watermark detector)

Figure 1: Flowchart of the LLM text generation, paraphrasing and classification process.

LLM paraphrase generation (LLM-PP). We then use the LLM-generated documents above to generate their paraphrases. For the outputs from the XSum dataset, summaries are first generated using a fine-tuned T5 model specializing in news summarisation [24]. The output summary is limited to a length of maximum length of H-PP in the original XSum dataset. The summaries, along with the LLM-DOC generated documents from the other 3 datasets, are taken to generate LLM-PP. We use two paraphrasers, namely DIPPER and BART-paraphrase models. DIPPER is a T5-XXL paraphrase generation model with 11 billion parameters, fine-tuned on 6.3 million data points. Inherently, DIPPER is capable of capturing long-term dependencies and controlling output diversity [19]. However, due to computational resource limitations, we use a non-context version, resulting in reduced performance on long-term dependency capturing. To generate paraphrases with models that capture long-range dependencies, the second model, BART is used. The model is built upon a seq2seq architecture, with a bidirectional encoder and a unidirectional decoder [20]. The bidirectional encoder allows the model to understand sentence embeddings in a longer range, thus providing more semantic and contextual information for later paraphrase generation. Particularly, the BART-paraphrase model used in this project is a fine-tuned BART model pretrained with 3 paraphrase datasets, providing better performance in paraphrase generation. The LLM-DOCs from each dataset are passed to these paraphrasers for paraphrase (LLM-PP) generation. In order to investigate the effect of the recursive paraphrasing attack mentioned in [31], 5 rounds of paraphrase generation are conducted, with the 1st of paraphrases generated from the LLM-DOC, and the subsequent rounds of paraphrases generated iteratively using the outputs of each round.

Final dataset. The final HLPC dataset is composed of 600 documents, with a balanced distribution of 150 documents per type, i.e. H-DOC, H-PP, LLM-DOC and LLM-PP. These documents are grouped into two categories, human-generated (H-DOC and H-PP) and LLM-generated (LLM-DOC and LLM-PP). We use these two categories to perform binary classification in the LLM- vs human-generated text detection task, where we mix both original and paraphrased documents to evaluate their impact (particularly that of human-written paraphrases H-PP) on the detection models.

3.2. Evaluating the quality of LLM-generated paraphrases

Since paraphrases should inherently not deviate much from the original text, it is essential to evaluate the quality of the paraphrases in terms of semantic and contextual preservation. Therefore, we evaluate the semantic similarities between the original texts and their paraphrases to assess the quality of the paraphrasing. We use an automated, fine-tuned sentence transformer model [3] to generate sentence embeddings for each text-paraphrase pair, and we calculate the cosine similarity between the embeddings to account for semantic similarity, scoring from -1 (least similar) to 1 (most similar). Semantic similarities are calculated between H-DOC and H-PP and between LLM-DOC and each round of LLM-PP.

Looking at human-generated data, MRPC, QQP and MultiPIT score over 0.7 for mean semantic similarity scores, indicating a good level of semantic preservation, while XSum scores only 0.383 since paraphrases in XSum are summaries of the documents. When it comes to LLM-generated data, first, the results from paraphrases generated from DIPPER and BART are compared. Figure 2 shows the similarity scores of watermarked and non-watermarked LLM-DOC and LLM-PP. From both graphs, BART outperforms DIPPER in all datasets, particularly in MRPC and MultiPIT where BART scores over 0.9 across paraphrasing rounds which is even higher than the score from human-generated data. DIPPER's performances are generally worse than human paraphrasing, except in XSum. Meanwhile, paraphrases from both paraphrasers exhibit degradation in similarity scores across paraphrasing rounds. With recursive paraphrasing, similarity scores from BART decrease slightly from an average mean score of 0.80 in the 1st round of paraphrasing to 0.785 in the 5th round, while similarity scores of paraphrases generated by DIPPER decrease