University of Glasgow | School of Computing Science

Honours Individual Project Dissertation

An Analysis on the Robustness of Watermarking Generative Text

Samuel Jackson

September 3, 2024

Abstract

Given the rise of LLMs and the growing power of malicious content-generation, methods of identifying and authenticating machine-generated content has become a necessity. Within current literature, approaches for identification and authentication are consistently arising and being thoroughly investigated. In January 2023, Kirchenbauer et al. (2023) proposed an LLM-based text watermarking method, known as the Maryland Watermark. As opposed to detecting generated documents, the documents would purposely embed a signature at inference time. However, this signature has the potential to be removed through malicious attacks. This paper aims to investigate the robustness of the Maryland Watermark through watermark removal approaches. Within the paper, a novel low-cost watermark removal technique is proposed as well as clear conclusions on the robustness of the Maryland Watermark. At low-risk of detecting human-written documents as machine-generated, the paper shows that detection accuracy of watermarked documents drops from 100% to 1.6% after paraphrasing. These results are accompanied by further analysis on paraphrasing approaches as well as the cost of attacking the documents.

Education Use Consent

I hereby grant my permission for this project to be stored, distributed and shown to other University of Glasgow students and staff for educational purposes. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Signature: Samuel Jackson Date: September 3, 2024

Contents

1 Introduction

1.1 Motivation

Generative AI has already been used to influence political opinions and continues to create scepticism and fear amongst the general public (Leffer 2024). Existing Large Language Models (LLMs), such as ChatGPT, can provide help with homework, display human-like emotion and unknowingly misinform users.

The capacity for misinformation with the new-found capabilities of LLMs is significant enough to spur governmental action (The White House 2023) and cause leaders of tech world to seek a petition slowing the progress of AI development (Future Of Life Institute 2024).

In an effort to combat the intentional and unintentional dissemination of misinformation, techniques for identifying the use of LLMs has become a focus for many people (Kirchenbauer et al. 2023; Mitchell et al. 2023; Grinbaum and Adomaitis 2022).

1.2 Watermarks

The growing difficulty in detection methods (Sadasivan et al. 2023) has led to research into placing unique signatures into content created by large language models (OpenAI 2023). Embedding signatures into content as a form of identification is known as *watermarking*.

In January 2023, the first LLM-focused watermarking technique was introduced, known as the Maryland Watermark (Kirchenbauer et al. 2023). This method proposes biasing textual generation towards certain words and using those words for detection. Some techniques have further developed the Maryland ideas (Liu et al. 2024), whereas some place rare Unicode characters within the generated text (Sato et al. 2023).

However, the existence of these watermarks has not been sufficient in ensuring that we can identify machine-generated content. In fact, methods of removing these watermarks already exists. For many watermarks, they are dependent on the exact words or characters used in a text. This dependency allows removal methods like paraphrasing, which consists of expressing the meaning of watermarked text in different words and structure, to erase the watermark.

1.3 Problem and Conclusions

Given that removal is certainly a concern for watermarking, we aim to investigate the difficulty of the removal of a watermark. In particular, we will be discussing the robustness of the Maryland Watermark and how easy it is to remove the certifying features of the watermark. We summarise our problem into a single question: “To what extent is the Maryland Watermark technique robust against bad actors?”

The investigation into our question will be completed through attempting to remove the watermarked from documents through various techniques. Chapter 2 will be providing the necessary background into watermarking and watermark removal techniques. Chapter 3 will follow with

the implementation of our research method, focused on differing types of watermark removal. Chapter 4 will present our results and attempt to answer our question stated above. Finally, our paper is completed with a conclusion which summarises the paper, highlighting our key results and discusses potential future work.

1.3.1 Conclusions

- **Recursive Paraphrasing**: Through the implementation of a recursive-paraphrasing approach, we conclude that repeated paraphrasing is not a reliable method to remove the Maryland Watermark.
- **Sentence against Paragraph Paraphrasing**: We conclude, through an analysis of both approaches, that the context providing to a paraphrasing model is inconsequential with regards to removal of the Maryland Watermark.
- **Word-Replacement Techniques**: We propose a novel, low-cost technique to replace certain words and determine that the Maryland Watermark is robust against this attack.
- **Feasibility of Academic Use**: We discuss the potential for academic use of a variety of attacking methods and conclude that sentence-based paraphrasing would be a feasible attacking method for use in an academic context.

2 Background

Transformers have realised the creation of incredibly powerful large language models, such as ChatGPT. Watermarking text has the potential be an essential safety method as it can allow us to correctly prescribe the author of a document to a large language model, if necessary. This chapter will provide an overview of how LLMs generate text alongside an insight into recent literature on text-watermarking as well as methods of removing such watermarks.

The sections are split to logically follow the process of watermark generation and removal, to aid the reader. The watermark generation and removal steps contain literature surveys of relevant techniques.

The literature survey acts as inspiration in our approach to the understanding the problem. Whilst not all techniques that are discussed are used in this paper, they are certainly relevant in terms of understanding our problem.

2.1 Text Generation

As our discussion is focused on watermarking generated text, we first provide an explanation into how text is generated from Transformer-based LLMs.

This section will go over the concepts behind providing text to models, as well as how the model chooses the next word in a sentence.

2.1.1 Tokenisation

In order to process and train language models, we provide a structure to our textual data. In this section we discuss how text is prepared for large language models through a process known as tokenisation.

An element of a textual dataset is known as a *document*. Through tokenisation, we structure our document into a list of *tokens*.

Definition 2.1.1 (Token)**.** A *token* is a sequence of characters, often referring to a word or a subword.

The choice of tokens instead of words is due to the scale of unique words. Tokens allow for a reduction of dimensions and enable more manageable computation. Although rule-based tokenisation exists, language models currently use trained tokenisation models. Tokenisation models are developed through methods like training a Byte Pair Encoding (BPE) tokeniser (Gage 1994; Sennrich et al. 2016) on a text corpus. As seen in Figure 2.1, a tokenisation function takes a string as input and produces a list of tokens, based on the vocabulary.

Definition 2.1.2 (Vocabulary)**.** A *vocabulary* is the collection of tokens recognised by a tokenisation function.

Importantly for our discussion on watermarking language models, the training of these tokenisation models generates a *unique* vocabulary dependent on the text corpus. The difference in

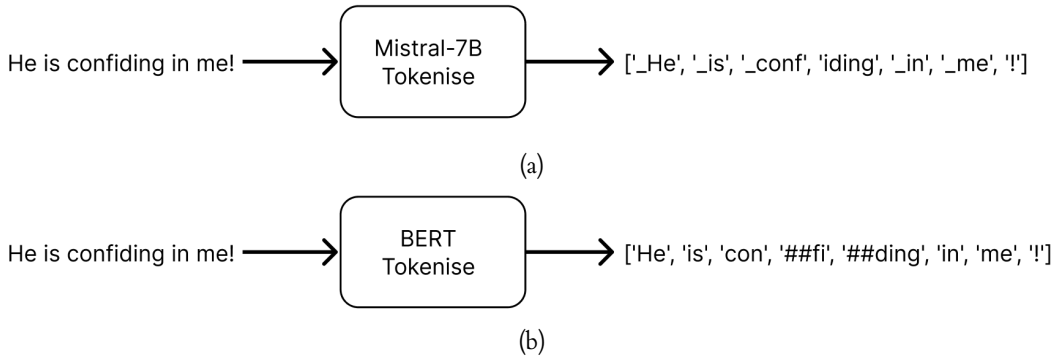


Figure 2.1: *Figure visualises the potential differences in tokenisation functions. Figure (a) was completed using the* `Mistral-7B-v0.1` *tokeniser (Jiang et al. 2023), a BPE tokeniser. Figure (b) was completed using the* `bert-base-uncased` *tokeniser (Devlin et al. 2018), a WordPiece tokeniser.*

recognised tokens is evident when comparing the lists of tokens in Figure 2.1. The uniqueness of tokenisation is propagated through the transformer-based architecture.

2.1.2 Generation Strategies

Text generation is treated as a predict-next-token task. We choose to only look at the previous tokens and consider which token is most likely to next appear. This task is known as Causal Generation. Explaining the model architecture in-depth exceeds the scope of this paper, however this section will discuss model outputs and generation techniques.

Causal Language Models, like ChatGPT, produces a collection of values, known as *logits*. From these logits, we obtain a probability distribution for the next token with the application of the Softmax function.

The created probability distribution is used for multiple generative strategies which include greedy-generation, multinomial sampling, and top-p sampling.

Each generation strategy serves a different purpose. Greedy-generation is a fast deterministic technique which simply chooses the token with the highest probability in the distribution. The deterministic nature means that the model will always produce the same output. Multinomial sampling randomly selects the token from a distribution, where the chance of a token is weighted by their probability. Top-p sampling is a variation of this, where it only allows random selection from the tokens with the highest probability such that the token probabilities sum up to p. Top-p is helpful in providing more diverse text, whilst also allowing the parameters to avoid potentially strange tokens.

2.2 Generating Watermarks

Definition 2.2.1 (Watermark)**.** A *watermark* is a faint figure or signature designed to represent ownership or authorship.

Watermarks are widely present within society with examples ranging from an American dollar bill to a music producer having an audio tag. In this section, we look towards recent literature which covers subtle methods of watermarking while attempting to adhere to the primary goals of watermarking.

Given the purpose of watermarking generative content, there are the following primary goals that an AI text watermark aims to achieve:

Figure 2.2: *Figure shows an $100 dollar bill, with a red circle highlighted a watermarked portion. The detection of the highlighted watermark requires holding the bill up to light and revealing a (second) face of Benjamin Franklin (Bureau Of Engraving and Printing 2013).*

- **Robust** - Capable of withstanding attempts to remove watermark.
- **Agnostic** - Information beyond generated text is not required for detection.
- **Imperceptible** - The watermark should not be visible, be it a change in quality or a written signature.

These requirements are designed to encourage watermarks which can adapt to the constant growing number of large language models and changing architecture decisions.

2.2.1 Maryland Watermark

Kirchenbauer et al. (2023) proposes a new method of watermarking text which takes advantage of the probabilistic nature of text generation. This section will go over the ideas of the Maryland Watermark, a technique which laid the foundations of LLM-focused watermarking.

Due to the nature of casual generation, each token is generated procedurally using the vocabulary. The Maryland paper proposes splitting the available vocabulary for each token. The split lists are labelled a *green list* and a *red list*. Tokens from the green list and red list are known as *green tokens* and *red tokens* respectively.

The lists are created through a hash function applied to the previously generated token. The vocabulary is re-ordered according to the hash and split into the lists. The hashing of the generated token is important as it enables the vocabulary splitting process to be reproducible, provided we know the hashing function used.

There are two primary watermarking techniques from Kirchenbauer et al. (2023), *Hard Watermarking* and *Soft Watermarking*. Hard Watermarking only allows the causal model to select tokens from the green list whereas Soft Watermarking chooses to increase the probability of the green tokens within the probability distribution. Figure 2.3 outlines the technique and the change that the Soft Watermarking process has on the probability distribution.

As opposed to Hard Watermarking, Soft Watermarking provides parameters that determine the strength of the watermark. The parameters are δ, the logit bias, and γ is the fraction of the vocabulary which is the green list. Typical values for δ and γ are within the ranges $[1, 10]$ and $[0.25, 0.5]$, respectively.

Kirchenbauer et al. (2023) provides an analysis on the Soft Watermarking technique, outlining the performance on differing bias and generation strategies. Without describing the necessary metrics, the paper shows that watermarking, with no detection-errors, can be achieved at reasonable strengths of watermarking whilst maintaining quality of the text.

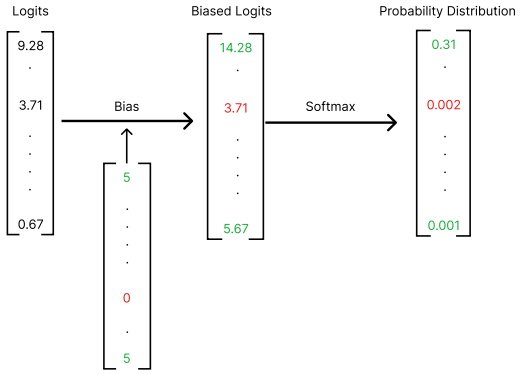


Figure 2.3: *Figure portrays the Soft Watermarking process where the bias value is 5. Visualisation of adding the bias to green tokens and altering the final distribution for all the tokens.*

2.2.2 Easymark

As opposed to complicated and robust watermarking techniques, like described above, Sato et al. (2023) elegantly proposes simple watermarks that are not intended to survive bad actors. This portion will briefly go over the watermarks as well as highlight the argument of watermark futility from Sato et al. (2023).

The provided watermarks are labelled as *Whitemark*, *Variantmark* and *Printmark*. The entire family of methods is susceptible to an attack called homoglyph removal, where a homoglyph refers to a group of similar-looking characters. However, these methods of watermarking lead to no degradation in perplexity. The watermarks proposed are variations or manipulations of Unicode characters, whether it be replacing whitespace characters or replacing letters with rarer Unicode siblings.

Infinity
Infinity

Figure 2.4: *Figure showcasing one of watermarking methods, Printmark. The characters 'f' and 'i' have been changed to the Unicode ligature U+FB01, providing a signature as the ligature is uncommon.*

Within the paper, Sato et al. (2023) discusses the strength of bad actors and describes a statistical analysis as to why a perfect watermark is impossible. In fact, the paper proposes that it is more prudent to place a low-effort watermark and not place too much confidence within watermarks. This paper could be considered a further discussion into the worth of watermarking text.

We will not be discussing Easymark further within this paper but we recommend reading the paper by Sato et al. (2023).

2.2.3 Retrieval defence

Krishna et al. (2023) proposes a retrieval-based defensive technique, designed to hold strong against paraphrasing attacks. This section will cover the principle ideas and primary results surrounding defence that are discussed in the paper.

Contrary to the other methods, this is a defensive technique that requires no generative-stage changes to the LLMs. The primary idea is to store AI-generated documents into a database and determine if a document is AI-generated by completing a retrieval call into the database.

This method achieves significant results with the evaluation taking place over multiple models. Before paraphrasing, At low-risk of detecting human documents as AI-documents, retrieval detection has an accuracy of 100% and after strong machine-paraphrasing, there is more than 96% accuracy across all models. This is compared to the Maryland watermark which achieves 55.8% accuracy, a starkly lower performance.

Notably, for these results, however, is that the retrieval corpus matches exactly the initial generated documents and the retrieval technique is dependent on a reliable database.

2.2.4 Other Watermarking Techniques

Although thorough experimentation with the following techniques exceeds the scope of this paper, the following papers present relevant and interesting watermarking methods. This section will only provide a high-level overview but I strongly recommend reading these papers.

As opposed to altering the sampling distribution, Kuditipudi et al. (2023) suggests a watermarking method that provides a sequence of random numbers, known as the *watermarking sequence*. For each token, we choose the next token through an element of the watermarking key sequence, as opposed to sampling or other generation strategies. The randomness of the watermarking sequence is intended to emulate the sampling generative strategy, maintaining the imperceptibility property.

Liu et al. (2024) proposes a dynamic bias which considers the semantic meaning of the previously generated tokens, as opposed to a fixed bias like the Maryland Watermark. The technique consists of an embedding model, a custom watermark model as well as a language model. The custom watermark model, T, receives embeddings as input and produces logits, P_T. Paired with the logits produced by the language model, P_M, we create a new distribution $P_{\hat{M}} = P_M + \delta P_T$, where δ weights the distribution P_T. A visualisation of this process is provided in Figure 2.5

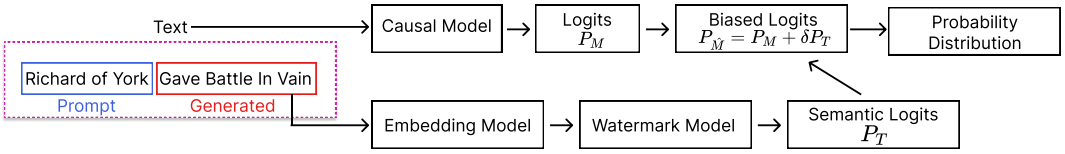


Figure 2.5: *Figure highlights the watermarking process proposed by Liu et al. (2024). The embedding model only takes tokens from the already generated tokens as input, highlighted in red, whereas the causal model uses all the text, prompt and generated, to produce its logits. Both sets of logits are linearly combined, with a weighting δ, to create a biased probability distribution.*

2.3 Attacking Watermarks

Methods of removing the watermark are real threats, hence the desire for a *robust* watermark. We call attempts to remove a watermark, no matter the removal method, an *attack* on the watermark.

Within this stage, we will discuss the word-replacement strategy, paraphrasing and translation-based attacks. Specifically, we highlight a number of techniques that we will be applying for our method of investigation alongside other successful methods from recent papers.

2.3.1 Word Replacement

A classic and simple technique is *word-replacement*. In this section, I will cover the motivation behind word-replacement as well as an overview of the technique.

This is perhaps the most primitive form of attacking. As opposed to restructuring a sentence or paragraph, we replace words with synonyms. We are motivated to complete word-replacement because it is a low-cost approach, comparatively, given the lack of an intensive large language model. Furthermore, the word-replacement algorithm has the potential to remove the so-called 'green tokens', reducing the chance of being detected as watermarked.

This method is dependent on a Part of Speech (POS) tagger model. A POS model is trained to give grammatical structure to words. This helps us understand how to replace a given word. Figure 2.6 provides a clear outline of the method, containing the Part of Speech tagging.

Figure 2.6: *Figure displays the word replacement process where 20% of all words are replaced with similar words, with respect to their part of speech. This process is completed by my word replacement method and the Flair Part of Speech Tagger (Akbik et al. 2018).*

Variations of this technique exist, such as only replacing nouns with respect to the POS tagger. These variations are designed to compensate for the algorithms lacking ability to understand any of the context, such as verb tense.

Whilst the method comes with flaws, this paper will discuss an analysis of the attack, a novel investigation amongst current research papers.

2.3.2 Paraphrasing

The dependency on tokens within the Maryland Watermark leads to the attacks focusing on the words in a generated documents. As the watermark relies on the detection of the 'green' tokens, we aim to remove those. However, without knowledge of which tokens are green, we cannot surgically remove those tokens.

In this section, we discuss paraphrasing as an approach that could remove the green tokens whilst maintaining the quality and meaning of the original text. We extend this discussion into large language models, as well as recent papers surrounding paraphrasing attacks.

Definition 2.3.1 (Paraphrase)**.** To repeat something written or spoken using different words, often in a humorous form or in a simpler and shorter form that makes the original meaning clearer (Cambridge Dictionary, 2024).

As can be seen in Example 2.3.1, we often restructure sentences for clarity.

Example 2.3.1. The two sentences below are examples of paraphrases.

(a) I take my dog on a walk in Central Park each Wednesday.
(b) Every Wednesday, I walk my dog in Central Park.

These two sentences maintain semantic meaning while holding a different sentence structure.

To emulate paraphrasing attacks, paraphrasing models are used. These are large language models which are trained to, when given a document, produce a paraphrase. This paper is concerned with two variations of paraphrase models, paragraph-based and sentence-based paraphrasers. For

a paragraph d, paragraph-based models paraphrase d while using the entire paragraph as context. Meanwhile, sentence-based paraphrasers split d into sentences and paraphrase each sentence individually, using only the current sentence as context.

In recent literature, Krishna et al. (2023) discusses the attacks of paraphrasing by providing a paraphrase language model known as DIPPER. Through finetuning, this model is a paragraph-based paraphrase model with variable parameters. These parameters determine the degree of order and lexical change performed by the paraphrasing. A change in order refers alterations to the grammatical structure of a document whereas a lexical change denotes replacing particular words, choosing a synonym for an adjective.

Within the DIPPER paper, results of the paper convey the decreasing detection accuracy as the order and lexical parameters increase. The paper's highlighted statistic shows that where false detection of human-text as AI-text is at 1%, the DIPPER paraphraser drops accuracy of DetectGPT, an AI-text detector, from 70.3% to 4.6%.

Similarly, in other literature, Sadasivan et al. (2023) investigates recursive paraphrasing on watermarked documents with two models, DIPPER and a LLaMa-based chat model. In particular, the analysis takes place on generated documents of approximately 300 tokens.

The paper reports results of clear degradation in detection through the iterations of DIPPER-based paraphrasing. With the false detection of human-text as AI-text at a low-risk, the chance that a watermarked document is detected drops from 99.8% in the original document to 30.9% in the third paraphrase. This is significant change at minimal cost, as reported by Sadasivan et al. (2023), where the cost is discussed with respect to the perplexity metric.

2.3.3 Translation-Based Paraphrasing

He et al. (2024) provides a deep investigation into the process of using translation to remove watermarks. As opposed to training a new paraphraser, He et al. (2024) utilises existing translation models in order to paraphrase. This section will give a brief overview as to how this works and the performance of the technique.

As can be seen in Figure 2.7, the process of twice-translation, going between languages, can act as a paraphrasing operation.

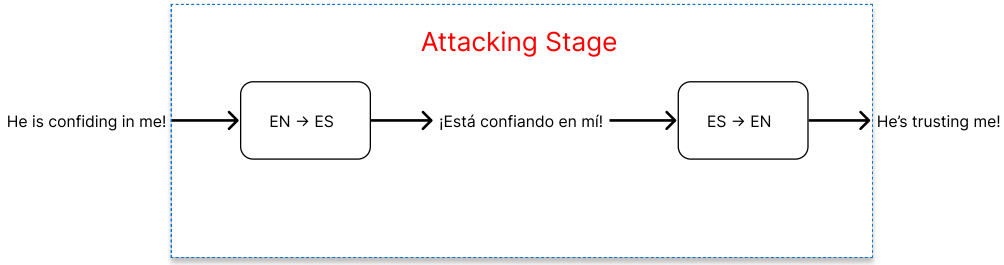


Figure 2.7: *Figure above displays the translation attacking process with twice translation involving English and Spanish. Translations were completed with the* `opus-mt-en-es` *and* `opus-mt-es-en` *models respectively (Tiedemann and Thottingal 2020).*

This method is particularly effective because the architecture behind paraphrasing and translation is the same: Sequence-to-Sequence (Seq2Seq). The Seq2Seq architecture is a decoder which has access to all the tokens in a document, not just the previous ones. This drastically improves the performance of LLMs on tasks like summarisation as it takes different contexts into account.

He et al. (2024) outlines clear results which portray the efficacy of this technique. A highlighted result is, in the case of false-detection of human-text as AI-text is at 10%, the chance that a

watermarked document is detected drops from 99.2% to 21.3% after paraphrasing with this method. This is a noticeable change and shows that the translation attacks bring the performance of a random classifier.

2.3.4 Other Attacking Techniques

The following techniques are not necessary for this paper but are worth understanding as they discuss alternative approaches to watermark removal.

A technique which was previously mentioned was *homoglyph removal*. Homoglyph removal consists of replacing each character with the normal Unicode or ASCII character, with respect to their homoglyph family. Notably, this approach acts as an exact counter to the watermarking concepts in Section 2.2.2.

Spoofing documents is an interesting approach which *attacks the validity* of a watermark. Sadasivan et al. (2023) shows that some watermarking methods are vulnerable to purposely making human-documents seem watermarked. This approach does not aid in removing watermarks from machine-generated documents but it reduces the credibility of a watermark. Without trust in the watermark, it is hard to justify its use in authorship.

Both of these techniques are particularly effective techniques but they will not be further discussed in this paper as they are only applicable to certain watermarks.

2.4 Approach

Following all of this background, we summarise the primary points that will be used for the rest of our paper. Within the rest of this paper, we shall be generating documents and attacking them through three primary methods.

Our attacking methods consist of:

- **Paragraph-Based Paraphrasing**: Paraphrasing the entire document, supplying the entire context.
- **Sentence-Based Paraphrasing**: Paraphrasing each sentence within a document, only supplying context of the given sentence.
- **Word Replacement**: Changing words within the document to similar words.

2.4.1 Problem Phrased

Our problem can be succinctly defined in the following question: "To what extent is the Maryland Watermark technique robust against bad actors?"

This paper breaks down that question into 4 smaller sub-questions. These sub-questions form our research questions for the paper and what we aim to understand.

RQ1: Does paraphrasing recursively degrade accuracy in detection of the Maryland Watermark?

RQ2: Is sentence-based paraphrasing more effective than paragraph-based paraphrasing when dealing with removal of the Maryland Watermark?

RQ3: Is a low-cost, word-replacement algorithm sufficient to remove the Maryland Watermark?

RQ4: Are the attacking methods feasible for use within an academic context with respect to the Maryland Watermark?

3 Methods

With the aim of understanding the robustness of the Maryland Watermark in generative text, we discuss our method that takes us from the generation of watermarked documents to the evaluation of attacked documents.

3.1 Overarching Approach

Before detailing all the design choices, I present the overarching implementation of our research method to aid in following the decisions. The visual in Figure 3.1 is provided to further assist in understanding the approach.

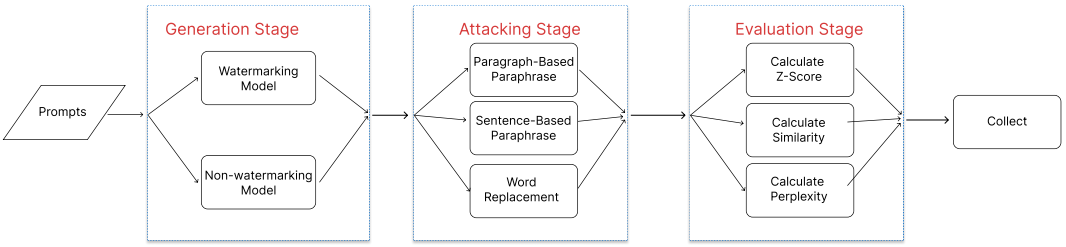


Figure 3.1: *Figure displaying the research processing, highlighted into their respective stages.*

The approach begins with creating watermarked documents through a large language model, given a dataset of prompts. These watermarked documents will be paired alongside non-watermarked documents, generated by the same model, without the watermark. The non-watermarked documents serve as a benchmark, so as to better understand the impact of the watermark.

Consequently, these essays are passed through our chosen attacking methods. These attacking methods will be two variants of paraphrasing as well as word-replacement.

Finally, we will evaluate the watermark quality of all the documents, which will include all the attacked documents as well as the original documents. Alongside the assessment of watermark quality, we evaluate our documents to gain an understanding of the cost of attacking the watermarked documents.

3.2 Generation Stage

3.2.1 Dataset

To generate our watermarked documents, prompts had to be provided to a causal language model. The prompts were selected to be in the form of instructions or tasks. These prompt also came alongside human-written student essays. The student essays are useful in providing benchmarks for evaluation metrics, such as perplexity.

Kaggle, a data science platform, hosted a competition focused on detecting AI generated text (King et al. 2023). As the competition was an AI focused task, people created additional datasets to aid in finetuning. One such dataset is known as the DAIGT dataset, produced by Paullier (2023), and is the dataset that will be used.

The associated student essays were scraped from a separate Kaggle competition (Franklin et al. 2022), guaranteeing that the essays are not AI-generated.

The dataset is composed of three relevant columns: *id*, *text*, *instructions*, with 2421 rows. The *instructions* column refers to tasks given to students, where the paired *text* column refers to the essay produced by a student according to the task.

id	text	instructions
6060D28C05B6	Some schools in United States ofter classes from home because is good option to students . Some scho...	Task: Write a persuasive essay on whether or not classes from home should be offered as an option f...
60623DB5DE7A	Four-day work week, a remarkable idea to conserve energy and resources, some businesses have adopted...	Task: Research the advantages and disadvantages of a four-day school week and write an essay to det...

Table 3.1: *The above table displays 2 of the existing 2421 rows within the DAIGT dataset, where columns are cut for brevity and readability.*

Current literature is focused on auto-generative prompting whereas I have decided to focus on instruct-style prompting. Instruct-style prompting, to the best of my knowledge, is not deeply investigated within this area hence it could lead to differing results. Although interesting, attempting to investigate both auto-generative and instruct-style prompting is outwith the scope of this project.

Initial investigations were completed with the auto-generative style, using a dataset of one million articles (Corney et al. 2016), if you are inclined to re-evaluate results with a different prompting style.

3.2.2 Model

Current literature studying watermarks is typically uses auto-generative and chat models (Pang et al. 2024; Liu and Bu 2024; Kirchenbauer et al. 2023; Sadasivan et al. 2023). This paper will use `Mistral-7B-Instruct-v0.2` (Jiang et al. 2023), which is a 7 billion parameter model which has performed very well in a variety of benchmarks. It is an Instruct model, a model which is finetuned towards dealing with instruct-style prompts.

From Figure 3.2, the prompting method is designed to reflect the instruct-nature required for the model.

The chosen generation strategy was multinomial sampling, with the max number of new tokens generated set at 7500, to allow for potentially large documents. No other sampling strategies, such as top-p were used, as we aimed to create the most diverse documents possible.

3.2.3 Watermark

We will be using the Soft Watermarking method, outlined in Section 2.2.1, as it is the watermark on which we have focused our research questions.

The Maryland Watermark was the ideal watermark as it is the original paper on token-level watermarking. Other watermarking techniques are present, such as Easymark from Section 2.2.2

Prompt

You are a student working on the following assignment.

Write an essay based on the following task in no more than a 100 words:
Task: Write an essay discussing the benefits of pushing yourself beyond what you have already mastered in order to grow and achieve your dreams. Use Emerson's quote as a starting point and back it up by citing real life examples from your own experiences.

Figure 3.2: *Figure portraying the prompting method used for the generation of watermarked essays, where the content in blue is a sample from the* instructions *column in the DAIGT dataset. The grey text represents additional prompt text used to guide the model.*

or a semantics-based watermark (Hou et al. 2024), however these watermarks are not implemented to respect the breadth of this paper.

Our implementation is completed using the HuggingFace Python library (Wolf et al. 2020), alongside code provided by Kirchenbauer et al. (2023). This library provides an interface to the language model as well as the ability to manipulate logits at the generation stage, prior to the creation of the probability distribution, as outlined in Figure 2.3.

Specifically, we choose our bias and green list fraction to be $\delta = 5$ and $\gamma = 0.25$ respectively. This means that for the generation of each token, a quarter of the vocabulary are 'green' tokens. Similarly, those 'green' tokens will have a bias of 5 added to them. These parameters are in line with results from Kirchenbauer et al. (2023) and reflect a strong watermark that maintains a reasonable degree imperceptibility.

3.3 Attacking Stage

In this section, I will be going over our attacking technique. In order to understand the robustness of the Maryland Watermark, it is necessary to attempt watermark removal with a diverse range of methods.

As highlighted in Figure 3.1, we choose to paraphrase attack with a sentence-based and paraphrase-based paraphraser. Furthermore, we apply our word-replacement algorithm as an attacking method. All of the prior methods will be discussed in this section, in the listed order.

The attacking techniques are chosen precisely to reflect our research questions, mentioned in Chapter 2.

3.3.1 Paragraph-Based Paraphrasing

Our attacking approach of paraphrasing with respect to the paragraph is based on the idea of supplying the maximal amount of context. With more context, the model is afforded greater potential to gleam the semantic meaning of a document.

To create a paragraph-based paraphraser, we choose a model with the same architecture as a translation model. Krishna et al. (2023) proposes precisely this model, known as DIPPER. In fact,

the paper discusses finetuning from dataset generation to a finetuned model. Unfortunately, due to computational constraints, we could not use the given model. As a result, we choose to create our own model, with the principles from this initial paper. For brevity, I will refer to this model as *p-paraphraser*.

We choose to finetune a Sequence-to-Sequence (Seq2Seq) model with a dataset of paragraph pairs. Google's T5 models are designed for transfer learning (Raffel et al. 2023), hence we look towards a variation of T5. The decided model is a checkpoint model, `t5-efficient-large-nl32`, designed for further training. The suffix of this model refers to the number of transformer blocks, where 32 is larger than the norm. The large number of transformer block adheres to a Deep-Narrow architecture, a recommendation from Tay et al. (2022), the paper which produced these variation models.

Krishna et al. (2023) leverages the Par3 dataset Karpinska et al. (2022) to create the dataset that I will be using, which I will refer to as kPar3. Par3 is a collection of books, translated by multiple authors. The paragraphs from each translation are collected and aligned, appearing as paraphrases. Figure 3.3 provides a visual description of how the Par3 data is collected.

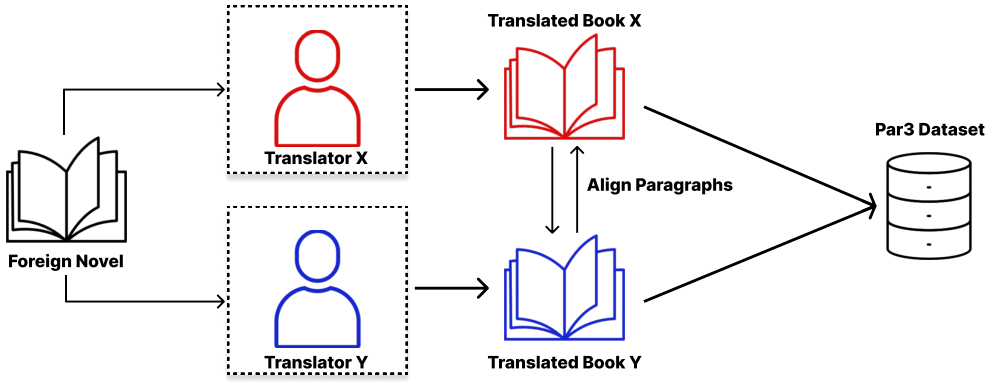


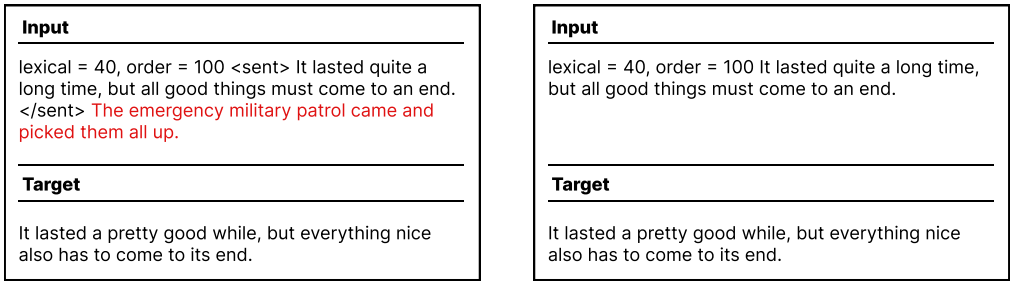
Figure 3.3: *Figure describing how the Par3 data is created and produced. Figure is outlining that a famous book is translated by multiple distinct people. The respective translation paragraphs are aligned with each other and considered paraphrases.*

The kPar3 dataset created documents that contained ~6,000,000 paraphrase pairs, split between Google translated and the human translated described above. The dataset has two types of documents, those which provide additional context to the paraphrase and those which do not. Documents which provide additional context also denote the target portion of paraphrasing, as highlighted in Figure 4.2. Each document starts with the arguments *lexical* (L) and *order* (O) where each argument only has values from the set {0, 20, 40, 60, 80, 100}. These arguments are included to impose an understanding of paraphrasing strength in the model.

As the dataset is too large for me to train, we sample the kPar3 dataset. Importantly, we only sample from paraphrase pairs with no context. Non-context pairs were chosen to avoid finding an appropriate context-split within a document to be paraphrased. Furthermore, by creating a context-split, the available text to be paraphrased is reduced. However, it is worth noting that finetuning with context-paraphrases performs better, with respect to the paper's metrics, in the original paper (Krishna et al. 2023).

The original paper trains the model on the entire kPar3 dataset, taking between 384-768 GPU hours. Incapable of replicating the same compute periods, the finetuned model that I present is trained on our sampling of 100,000 documents, taking 14 GPU hours. Further specifications into the training arguments are provided in Appendix A.1.

Finally, to attack our documents, we apply paraphrasing attacks on the entire document with the



(a) *Context Sample* (b) *Non-context Sample*

Figure 3.4: *Figure displays comparison between a paraphrase pair with context (a), and a paraphrase without context (b). The context portion in (a) is highlighted red for clarity. Furthermore, in (a), the <sent> tags denotes the text to be paraphrased. The target section refers to the desired output of text. Our dataset sample only contains examples like shown in (b). The terms 'lexical' and 'order' refer to the parameters used for determining paraphrase strength.*

parameters L40 and O40, describing a moderate paraphrase. Similarly, our generation strategy is top-p sampling, where p is 0.75, as suggested in the original paper (Krishna et al. 2023). The paraphrasing is repeatedly recursively to assist in our understanding of the effects of recursive paraphrasing on the Maryland Watermark, our first research question.

3.3.2 Sentence-Based Paraphrasing

Opposing the prior method, we aim to separate the paragraph into sentences here and paraphrase each sentence independently. In this section, I will cover the algorithm and model that helps us answer **RQ2**.

With regards to the model, we chose a model known as the `ChatGPT Paraphraser` (Vladimir Vorobev 2023). Similar to the paragraph model previously described, this is a model built off of T5, `T5-base` to be exact. The paraphraser is trained on a synthetic dataset of paraphrases produced by ChatGPT, hence the name. For brevity, I will refer to this model as *s-paraphraser*.

This model was chosen due to a lack of sentence-based models within research literature and consequent popularity of the model within the HuggingFace platform.

As our attacking method, we complete sentence-based tokenisation with the help of the Natural Language Toolkit (NLTK) (Bird et al. 2009). Each sentence is separated and passed to our paraphraser, similarly using top-p sampling. The paraphrased sentences are connected together once more, as sentences, maintaining their order.

The existence of *s*-paraphraser is precisely what we need, alongside our paragraph-based paraphraser, in order to evaluate differences between sentence-based and paragraph-based paraphrasers, our second research question.

3.3.3 Word Replacement

In order to answer our penultimate research question, we attempt attacking with the use of a word-replacement algorithm. Within this section, I will be discussing the implementation of the algorithm as well as the particular variations that we choose to evaluate.

The algorithm is dependent on appropriately identifying the grammatical nature of each word, as to find an apt replacement. An immediate solution to this is available in the form of Flair's POS Tagger (Akbik et al. 2018), which is a low-cost model that provides grammatical insight into a sentence.

The POS tagger is paired with the WordNet (Fellbaum 1998) library which recognises 155,327 distinct words. Linked to these recognised words, there is 117,597 sets of synonyms, known as Synsets. Note that two words can be linked to the same Synset.

With the knowledge of a word and its grammatical structure, we can apply word-replacement by searching the related Synset and selecting an option from here. Our selection from the Synset is a random sample, as the set is unordered. This synonym choice could be improved by picking the most similar word, according to word embedding model, such as Word2Vec.

We can summarise our algorithm as follows:

1. **Tagging**: Grammatically tag the document with Flair's POS.
2. **Select Words**: Select words to replace based on percentage or word-type.
3. **Find Synonyms**: With respect to the selected words and grammar, collect potential synonyms.
4. **Choose Synonym**: Randomly sample a choice from the synonym set.
5. **Rebuild Document**: Rebuild the document, replacing the chosen words.

The step outlined as Step (2) is where we introduce algorithm variations. With a tagged document, we could randomly replace a given percentage of the document. This has flaws as WordNet is not built for dealing with verb replacement of differing tenses. To deal with this, we could choose to replace all the nouns or adjectives. However, importantly, by limiting to certain grammar, we reduce the number of replaceable words.

In this paper, we choose to attack with a *noun-replacement* approach as well as *25% of words replacement* approach. Noun-replacement was chosen as it is the most common class of word recognised by WordNet, as mentioned in WordNet stats found in Appendix A.2. The percentage attack is selected to match our green list fraction γ, as mentioned in Section 3.2.3.

The attack of this algorithm will provide exactly the results necessary to discuss the sufficiency of word-replacement algorithms with regards to Maryland Watermark removal effectively.

3.4 Evaluation Stage

In order to answer any of our research questions, we need a way to numerically evaluate a watermark's strength and associating factors. Therefore, we come to the most important stage of our method. This section will discuss our evaluation strategy as well as the necessary metrics for understanding our results. The approach we have taken to evaluation is graphically outlined in Figure 3.1.

Our main topics are detection, similarity and perplexity. These topics are intended to measure watermark strength, paraphrase quality and text quality respectively.

3.4.1 Watermark Detection

The metrics and formulas provided here are vital to understand the results presented in Chapter 4. This section begins with describing the Maryland method of detection followed by the equations and metrics for detection evaluation. These equations are precisely the way in which we measure the robustness of a watermark, pre- and post-attacks. The section will end with how to understand the metrics and gleam meaning from them.

We begin with defining the z-score proposed by Kirchenbauer et al. (2023), a statistical measure of standard deviation.

Definition 3.4.1 (Z-Score)**.** Let $\gamma \in (0, 1)$. Let T and $|s|_G$ denote the size of the vocabulary and number of green tokens in the document s.

$$z = \frac{|s|_G - T\gamma}{\sqrt{T\gamma(1-\gamma)}},$$

where γ represents the fraction of the vocabulary which are green tokens.

Z-Scores can be used to determine how irregular a result is. For our case, this formula is used to determine if a document is watermarked. Specifically, we call a document **watermarked** if the z-score is 4 or greater. The primary takeaway is that this result is strongly dependent on the green fraction as well as the *detectable* green tokens.

Whilst a document may be watermarked, it is up to the Maryland detection algorithm to find the green tokens which prove the watermarking. The code and implementation for the detection algorithm was provided by Kirchenbauer et al. (2023). The algorithm determines the colour of a token through splitting the vocabulary in the same way as the watermarking process, through the hash of the previous token. Figure 3.5 provides a visual explanation of the process for a single token.

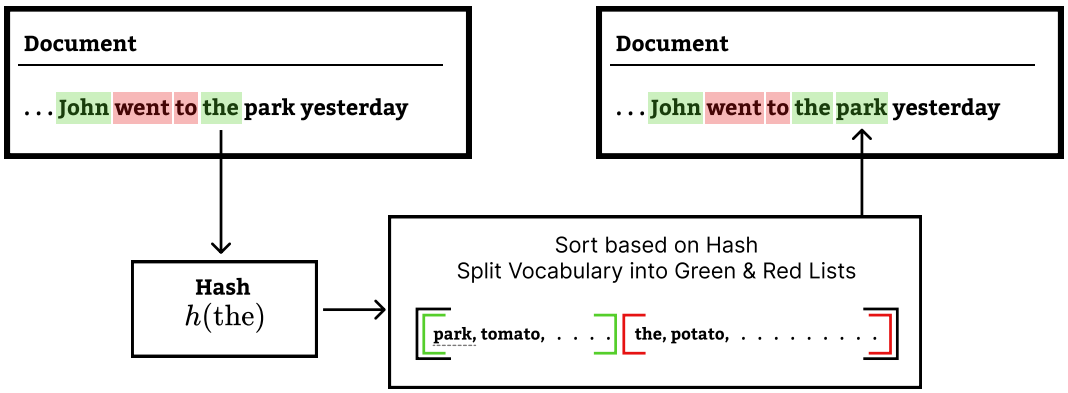


Figure 3.5: *Figure shows the process of determining token colour of the token* ***park****. We hash the previous token* ***the*** *in order to sort our vocabulary. Furthermore, the green and red list is dynamically created based on the hash key, so it is harder to crack for bad actors. Importantly, tokens which are green, such as* ***the****, only need to be in the green list based on the hash of the token before it.*

Once the entire document is classified into red and green tokens, the green tokens are counted and used for the z-score evaluation.

As mentioned in Section 2.1.1, the detection algorithm must know the tokeniser for generation. In particular, they must know the vocabulary in order to split the tokens correctly and identify the colour of a token. The uniqueness of tokenisation means that if the generation model and respective tokenisation function are unknown, the detection algorithm will not be capable of correctly identifying the watermark. This is a problem that leads to the watermark failing the *agnostic* property of watermarking.

The detectable green tokens are precisely the area in which the attacks take place. These attacks result in two primary types of errors, Type-*I* and Type-*II* errors.

A Type-*I* refers to falsely detecting a human-generated document as a machine-generated document, also known as a *False Positive*. Similarly, a Type-*II* refers to detecting a machine-generated document as human-generated, also known as *False Negative*. With the goal of minimising the number of human-written documents detected as AI-generated, we use the metric known as the *False Positive Rate*, which evaluates the rate of false positives.

Definition 3.4.2 (False Positive Rate)**.** Let N and FP be the number of non-watermarked and number of false positives respectively. We define the *False Positive Rate* (FPR) as follows:

$$\text{FPR} = \frac{FP}{N}$$

While FPR ensures that we do not incorrectly mark human documents, we also wish to make sure that we can correctly identify watermarked documents as watermarked. For this, the notion of a *True Positive* is introduced, which is a watermarked document correctly classified as watermarked. Hence, we introduce a neighbouring metric, known as the *True Positive Rate*.

Definition 3.4.3 (True Positive Rate)**.** Let P and TP be the number of watermarked documents and the number of true positives respectively. Then we define the *True Positive Rate* (TPR) as follows:

$$\text{TPR} = \frac{TP}{P},$$

This metric is also known as *recall*.

On their own, these metrics are useful. However, we can create even more visualisations from these introduced metrics that help us understand our watermark. A popular visualisation is the *Area under the Receiver Operating Characteristic* (AUROC), which measures the TPR against the FPR. By repeatedly altering our z-score threshold, we can re-evaluate the detection ratings and help determine the strength of our watermark. In the end, from this visualisation, we wish to see the curve maximising the area under the curve.

3.4.2 Document Similarity

While we can evaluate the removal of a watermark, it is useful to have a notion of cost in the removal. This section will discuss how to understand the cost of an attack and our implementation of similarity metric.

By placing confidence in our generation model alongside the imperceptibility of the Maryland Watermark, we can consider cost by measuring similarity to the original document. Under the assumption that the original document is the ideal response to the prompt, we consider the similarity to the original document to represent the cost of attacking.

The chosen measure is cosine similarity, as that is convention in NLP. Notably, cosine similarity requires vector representations of our documents. Our major choice design choice arrives in determining the vector representations of our documents.

Technically, cosine similarity is not the similarity between documents but rather the similarity between the document embeddings with respect to a given model. This difference is significant as models produce unique representations, leading to different similarities. Furthermore, similarity does not respect paraphrases particularly well, as can be seen in Example 3.4.1.

Example 3.4.1. The sentences below are similar sentences, wherein (2) and (3) are human paraphrases.

(1) I take my cat on a walk on Wednesdays in Central Park.
(2) I take my dog on a walk on Wednesdays in Central Park.
(3) Every Wednesday, I walk my dog in the centre of New York City.

Using the HuggingFace model `all-mpnet-base-v2`, the highest-rated document embedding model, we see that sentences (1) and (2) are more similar, despite (2) and (3) being a paraphrase pair, with respect to the cosine measure.

In an attempt to feel confident with our document embeddings and consequent cosine similarities, we will be using an embedding model trained on paraphrastic pairs. Wieting et al. (2023) provides exactly this model as well as stating results proving its effectiveness. Krishna et al. (2023) uses this model for similarity and, from analysis on human-paraphrases, states that a similarity of 0.76 is a sufficient score to maintain the original meaning.

For each attack, we evaluate every attacked document against the original, unaltered, document. This evaluation is focused on understanding how much the original meaning of a document is degraded through attacking, helping us understand the cost of attacking our models.

3.4.3 Perplexity Measure

As we place confidence in the similarity measure, we can also apply an adjacent metric. This section discuss another evaluation metric used to determine the quality of a text, helping us further understand the cost of attacking.

Within Natural Language Processing, we use a measure known as perplexity that allows us to gauge how likely a model is to produce a document.

Definition 3.4.4 (Perplexity)**.** Let s be a document of length n and let s_i denote the ith token. We define the *perplexity* function PPL as follows:

$$\mathrm{PPL}(s) = \exp\left(-\frac{1}{n} \sum_{i}^{n} \log p(s_i | s_{<i}) \right),$$

where p is the probability of a token given the previous tokens from a given model. Numerically, the model is 'unsurprised' by a generated document when the perplexity is closer to 1.

The probability function p is entirely dependent on the current language model, hence each model will give a different perplexity for a document. To define a relative metric, some papers (Kirchenbauer et al. 2023; Giboulot and Teddy 2024; He et al. 2024) have intelligently used the same language model to calculate perplexity, `OPT-2.7B`. To allow our paper to have comparable results, we also match this convention and use this language model for perplexity.

If we operate under the assumption that the language model used for perplexity is a strong language model, we know that a low-valued perplexity represents a high quality document. The perplexity is computed on all documents, both attacked and unaltered, which provides us an analysis of the loss in textual quality.

4 Results

In this section, we try to answer our primary research questions provided the results we generated through the implementation outlined in Chapter 3. To aid the reader, these are the research questions that we are looking to answer:

RQ1: Does paraphrasing recursively degrade accuracy in detection of the Maryland Watermark?

RQ2: Is sentence-based paraphrasing more effective than paragraph-based paraphrasing when dealing with removal of the Maryland Watermark?

RQ3: Is a low-cost, word-replacement algorithm sufficient to remove the Maryland Watermark?

RQ4: Are the attacking methods feasible for use within an academic context with respect to the Maryland Watermark?

Our analysis is completed on 500 watermarked documents alongside 500 non-watermarked documents. All 1000 documents are generated through our Mistral model. Table 4.1 highlights the average document length in tokens as well as number of documents for each phase of evaluation.

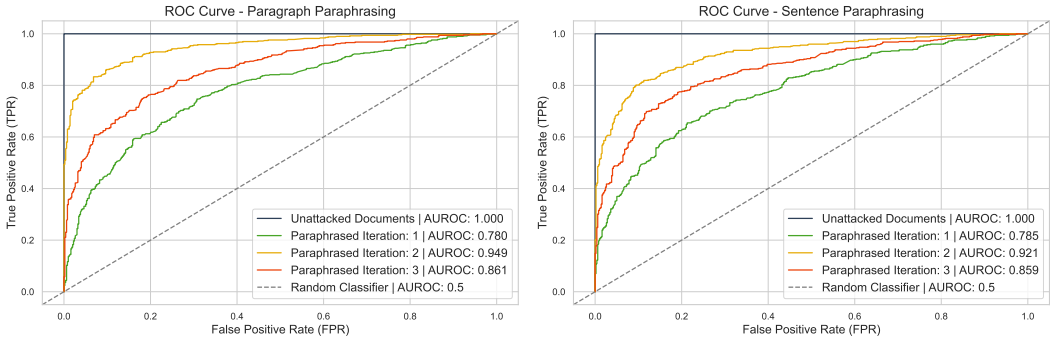
Documents	No. Documents	No. Watermarked	No. Tokens (Mean)
Original Generated	1000	500	201.96
Paragraph-Paraphrased	996	498	154.54
Sentence-Paraphrased	996	498	191.19
Noun Word-Replaced	996	498	208.06
Percentage Word-Replaced	996	498	208.52

Table 4.1: *Table displays basic stats about our generated documents and their first-attacked documents. Tokenisation is completed with the* `Mistral-7B-Instruct-v0.2` *tokeniser. The mean is calculated to 2 significant figures.*

4.1 Recursive Paraphrasing

In order to answer our first research question, we investigate the power of recursive paraphrasing. In this section, we aim to see the strength of the watermark varying after each iteration. Furthermore, we wish to understand the cost of repeatedly paraphrasing. These results will help us see whether recursive paraphrasing is an effective tool with regards to removing the Maryland Watermark.

Using the AUROC graphs, as seen in Figure 4.1, we see that repeatedly paraphrasing does alter the detection performance. The graphs, composed of z-scores and altering thresholds, highlight worse performance as the curves tend towards the linear curve. Interestingly, both paraphrasing methods display the same detection trend, in which the first paraphrase is the most effective in removing the watermark. Meanwhile, the second iteration, contrary to intuition, actually has a stronger watermark compared to the first paraphrase.



(a) *Paragraph-Based Paraphrasing* (b) *Sentence-Based Paraphrasing*

Figure 4.1: *The figure portrays the ROC curves of the documents after (a) Paragraph-Based Paraphrasing and (b) Sentence-Based Paraphrasing. The legend contains information about AUROC as well the respective iteration of each curve. Finally, the dashed line is a random classifier that highlights a lower-bound on performance. The evaluation is completed on 996 documents, half of which are watermarked with* $\delta = 5, \gamma = 0.25$.

The irregularity of the second iteration is representative of the unpredictability in Maryland Watermark, where the green and red tokens are split entirely based on the hashing function of tokens. Specifically, this result shows that we cannot guarantee continual degradation in watermark detection as we recursively paraphrase. However, we note that these results oppose Sadasivan et al. (2023), wherein we do see that the iterations continually degrade the detection accuracy. Although our paper differs in qualities such as models and prompting-style, we cannot confidently state a reason as to our differences.

Beyond detection performance, we also look to understand the textual cost of the repeated paraphrasing. Within Table 4.2, there is a degradation in similarity as we continue to paraphrase, with both paragraph-based and sentence-based attacks. Given that the similarity is measured against the base, unaltered document, the decreasing scores suggest that we are losing our original meaning.

Documents	Similarity (↑)	
	p	s
First Paraphrase	0.820	**0.856**
Second Paraphrase	0.731	**0.820**
Third Paraphrase	0.671	**0.785**

Table 4.2: *Table represents evaluation completed on 996 documents for each iteration, of which half are watermarked as per the arguments in the methods chapter. The segments* p *and* s *denote the paragraph-paraphraser and sentence-paraphraser respectively. The direction of the similarity arrow imply that higher scores are better. The numbers in bold denote the best score within their evaluation segment, for each iteration.*

Furthermore, according to Krishna et al. (2023), a document similarity of less than 0.76 means that the original meaning has been lost. In terms of binary classification, the paragraph-based paraphraser fails to maintain the meaning in the latter two paraphrases.

Ultimately, to answer our first research question, paraphrasing recursively does not predictably degrade accuracy in the detection of the Maryland Watermark. Furthermore, we see that repeated paraphrasing can distance a document from its original meaning, potentially lowering the value of a document.

4.2 Sentence-Paraphrasing against Paragraph-Paraphrasing

For our second research question, we look towards a comparison between sentence-based and paragraph-based paraphrasing with regards to an effective removal of the Maryland Watermark. To answer our question, we view the watermarking confidence paired with our described cost metrics.

Viewing Table 4.3, we see that the paragraph-based paraphraser is *marginally* better at removing the watermark with respect to Type-*I* and Type-*II* errors. In fact, with the Kirchenbauer et al. (2023) threshold of 4.0, there are no Type-*II* errors. There is a small difference with regards to TPR, Type-*I* errors, which I believe is a reflection of paragraph-restructuring potential that p-paraphraser contains.

Paraphrase Attacking Method	TPR (%)	TNR (%)	Perplexity (↓)	Similarity (↑)
Paragraph-Based	**1.606**	**100**	**19.530**	0.820
Sentence-Based	3.815	**100**	24.889	**0.856**

Table 4.3: *Table displays results evaluated on 996 documents for each iteration. Half of the documents are watermarked as per the arguments in the methods chapter. TPR and TNR are calculated with respect to the z-score threshold of 4.0, acting as a classification boundary. The arrows point in the direction of ideal results. The best result for each metric is in bold.*

One of the key results within Table 4.3 is perplexity. It is clear that the p-paraphraser can provide higher quality text, as per the `OPT-2.7B` model. However, despite the greater text quality, the sentence paraphraser retains higher similarity to the original document.

To answer our question, as to whether sentence-based or paragraph-based is more effective at watermark removal, I would argue that the choice of paraphraser does not matter. Although p-paraphraser performs slightly better, I believe this to be a difference in model quality as the p-paraphraser is finetuned on a superior model to s-paraphraser.

4.3 Word Replacement Sufficiency

To answer our penultimate question, we look towards our two variations of word-replacement attacks. To analyse the sufficiency of our algorithm, we consider the TPR and TNR metrics at the Kirchenbauer z-score threshold of 4.0.

From Table 4.4, we quickly see that noun-replacement is insufficient to break the Maryland Watermark. The high TPR shows that we correctly identify almost all watermarked documents and do not misidentify any non-watermarked documents. In fact, even the 25% Word Replacement is not sufficient, which is interesting precisely because 25% is how much of the vocabulary is to be in the green list for each token ($\gamma = 0.25$).

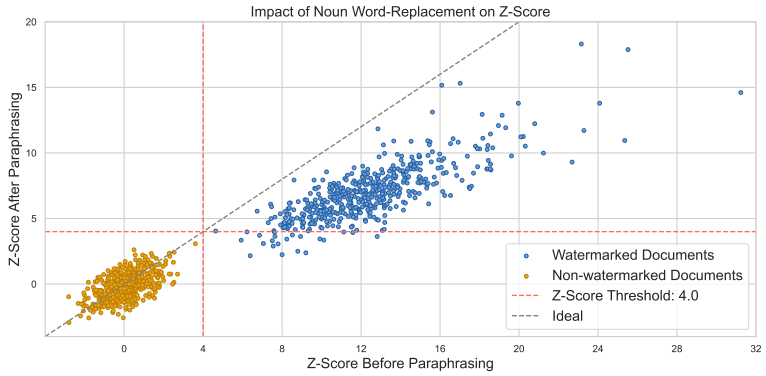
Replacement Method	TPR (%)	TNR (%)	Perplexity (↓)
Noun-Replacement	95.783	100	69.571
Percentage-Replacement (25%)	39.759	100	105.164

Table 4.4: *Table displays metrics evaluated of 996 documents, for each replacement method. The TPR and TNR values are complete according to the z-score of 4.0 Furthermore, the arrow denotes that you want perplexity as a lower value, as close to 1 as possible. The percentage-replacement method is completed with replacing 25% of words in each document.*

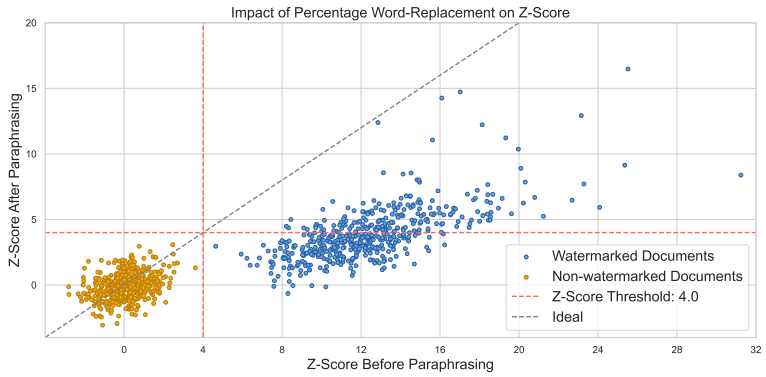
Interestingly, the change in words does have a noticeable impact on the z-scores, as can be seen in Figure 4.2a, showing the effect with an noun-replacement attack. Figure 4.2b shows the

same graph but using the percentage-replacement attack. The vertical distance from the ideal linear line reflects the impact of attacking with word replacement. It seems as though, for both methods, that the z-score is typically reduced but it is simply not sufficient.

A positive result, for the Maryland Watermark, is the lack of difference in z-scores within the non-watermarked documents, as seen by the orange points in figures 4.2a, 4.2b. This result reflects that the replacement methods cannot lead to *spoofing* attacks, as mentioned in Section 2.2.4, and undermine the credibility of the Maryland Watermark.



(a) *Noun-Replacement Attack*



(b) *Percentage (25%) Replacement Attack*

Figure 4.2: *Figure displays the z-Score of 996 documents after the effects of attacking with (a) noun-replacement and (b) percentage-replacement. The documents are orange and blue to denote whether they are non-watermarked or watermarked. In terms of an (x,y) frame, the x-coordinate refers to the z-score before word-replacement whereas the y-coordinate refers to the z-score refers after word-replacement. The distance of a datapoint from the dashed line represents the impact of attacking. Finally, the red-lines represent the z-score threshold of 4.0.*

Even with the lacking capacity to remove the watermark, the cost of completing word-replacement is not negligible. Compared to the paraphraser perplexity scores in Table 4.3, the word-replacement leads to 3× the perplexity, which is a significant loss in perplexity. As a further benchmark, the human-written documents have a mean perplexity of 18.632 (to 3 s.f.).

However, the loss in textual quality is understandable. As Example 4.3.1 highlights, synonyms are not always an appropriate replacement for a word in a given context. Since the word-replacement algorithm is not provided some method of understanding context, it struggles in these situations.

Example 4.3.1. Consider the following sentences:

(a) John parked in the car park.

(b) John parked in the car playground.

Note that although *playground* is a synonym of *park*, it is not appropriate replacement in this context. This example highlights the word-replacement methods flaws.

Fortunately, we can now provide a clear answer to the question as to whether a low-cost, word-replacement algorithm is appropriate for removing the Maryland Watermark. The noun-replacement variation is certainly not capable of removing the watermark whereas the percentage-replacement has the potential for removal but comes at a significant textual quality cost. Hence, low-cost word-replacement algorithms are not sufficient in removing the Maryland Watermark.

4.4 Feasibility of Attacking Techniques

We consider our final question as a culmination of the previous questions. This question deals with the feasibility of using these attacking techniques in a realistic pipeline to evade watermark detection.

From our results, we have established that word-replacement does not effectively avoid detection, nor does it provide high quality text as an output. Hence, we do not consider it as a feasible approach.

On the other hand, the paragraph-based paraphrasing produces high quality and effectively evades detection, being correctly detected only around 2% of the time. However, this is a heavy-duty paraphrasing approach which requires running a 1.5B parameter causal model. As mentioned in Kirchenbauer et al. (2023), with access to a high quality paraphraser without a watermark, it is probably easier to generate the text outwith the watermarking model and avoid having to paraphrase.

Unlike the other attacking methods, using the *s*-paraphraser remains a feasible technique.

The sentence-based paraphraser performed similarly to the paragraph-paraphraser at a fraction of the memory costs, as a model of 223M parameters. By approximation and typical memory costs, this is around 1GB of RAM to run inference at most, an achievable amount of memory for a phone. Hence, when detection is only achieved approximately 4% of the time, sentence-based paraphrasing becomes a realistic technique for evading detection in an academic context, particularly given that the perplexity is not noticeably worse than the mean perplexity of human-written essays.

5 Conclusion

We conclude our dissertation by answering our firstly proposed problem: "To what extent is the Maryland Watermark robust against bad actors?". As shall be seen in our summary, we note that the Maryland Watermark is not robust against bad actors of a higher quality than our word-replacement algorithm.

The following sections will provide a summary of our research questions, notes on limitations and future work.

5.1 Summary

In this paper, we attempted to understand the difficulty in removing the *Maryland Watermark* from AI-generated documents with two types of paraphrasers as well as a word-replacement algorithm. We used a cutting-edge LLM to generate 1000 documents and completed an analysis focused on answering research questions surrounding paraphrasing, word-replacement and the academic use of watermarks.

The two paraphrasers, sentence-based and paragraph-based, helped us answer our first research question discussing the impact of recursive paraphrasing on detection accuracy of the Maryland Watermark. Our results showed that the detection accuracy does not linearly degrade with paraphrase iterations and that detection accuracy can, in fact, increase between iterations. In a concise sentence: You *cannot* confidently repeatedly paraphrase under the assumption that detection accuracy will decrease.

The two paraphrasers were compared in our second question which dealt with the quality of watermark removal. Assisted by textual quality metrics, we determined that the style of paraphraser was *inconsequential in practice* with regards to removing the watermark. However, we did note that it would be worth re-evaluating again with more comparable paraphrasers with regards to the base model.

Our most conclusive result was for our third question, focused on whether replacing words was a sufficient technique to remove the Maryland Watermark. Through our noun-replacement and percentage-replacement algorithms, we discovered results that show word-replacement is *insufficient* and the Maryland Watermark is robust against such techniques. Under noun-replacement, we saw that detection accuracy, with recommended detection conditions, dropped from 100% to 96%.

With regards to our final question on academic use, we note that students have the computational capacity to use the sentence-based paraphraser and comfortably evade detection. As the paraphraser maintains textual quality, the removal method allows for the use of a cutting-edge Maryland-watermarked language model without identification of its use.

5.2 Limitations

Although there was computational constraints, such as lacking memory, I do not consider computational power to be a limitation. On the other hand, the lacking number of appropriate

metrics was a major limitation.

As mentioned in Section 3.4.2, document similarity was not entirely appropriate as a measure of similarity between paraphrases. Despite attempts to mitigate similarity issues, it was not sufficient enough to feel as though the similarity represented the quality of the paraphrase.

Existing paraphrase quality metrics, like BLEU and ROUGE, are known to be ineffective. The popularity of these metrics is a consequence of lacking alternatives and made it difficult to provide a numerical comparison between paraphrasers for our second research question.

5.3 Future Work

In spite of all the limitations, I still feel proud of the work that I produced and would have been happy to continue working on this topic. With regards to my existing research questions, I would have liked the opportunity to finetune my own sentence-based paraphraser, as it would have allowed a more comparable analysis between our paraphrasing styles.

Similarly, I would have liked more time to finetune the paragraph-based paraphraser, to achieve a quality closer to the original model, DIPPER.

Outwith my existing research questions, there are two avenues into which I would like to expand my analysis, which I propose below as further research questions

RQ5: Does the use of a sliding-window for z-score calculations provide greater accuracy for detection of documents that contain AI-generated text?

RQ6: Do the previously mentioned paraphrasing attacks succeed against a semantic-influenced watermarking technique?

The first of these research questions deals with the issue of embedding AI-generated text within largely human-written documents. This question would entail aligning AI-documents within human documents and determining appropriate z-score window lengths for detection alongside a comparative detection without the use of a window.

Meanwhile, the second question is an analysis of an existing technique which is specifically designed to be robust against paraphrasing. The second question is interesting because, without paraphrasing, more intelligent attacking methods may be required. After introducing this new watermark, we could re-use much of the existing implementation in this dissertation and complete another analysis.

A Appendices

A.1 Paraphraser-Based Paraphaser | Training Arguments

The following arguments refer to the entire training arguments used for training our paraphraser.

```
{
   "learning_rate": 1e-4,
   "bf16": True,
   "num_train_epochs": 2,
   "auto_find_batch_size": True,
   "generation_num_beams": 2,
   "generation_max_length": 200
}
```

More information is publicly available on the HuggingFace links: paraphrase-no-ctx, kpar3-no-ctx

A.2 WordNet Stats

The following information is present on the WordNet website: Stanford WordNet Stats.

POS	Unique Strings	Synsets	Total Word-Sense Pairs
Noun	117097	81426	145104
Verb	11488	13650	24890
Adjective	22141	18877	31302
Adverb	4601	3644	5720
Totals	155327	117597	207016

Bibliography

A. Akbik, D. Blythe, and R. Vollgraf. Contextual string embeddings for sequence labeling, 2018.

S. Bird, E. Klein, and E. Loper. Natural language processing with python: analyzing text with the natural language toolkit, 2009.

Blythwood. Word default no ligatures, 2014. URL https://en.wikipedia.org/wiki/File:Word_default_no_ligatures.tiff.

Bureau Of Engraving and Printing. 100 dollar note, 2013. URL https://www.bep.gov/currency/circulating-currency/100-note.

D. Corney, D. Albakour, M. Martinez, and S. Moussa. What do a million news articles look like?, 2016. URL http://ceur-ws.org/Vol-1568/paper8.pdf.

J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding, 2018. URL http://arxiv.org/abs/1810.04805.

C. Fellbaum. Wordnet: An electronic lexical database, 1998. URL https://mitpress.mit.edu/9780262561167/.

A. Franklin, Maggie, M. Benner, N. Rambis, P. Baffour, R. Holbrook, S. Crossley, and ulrich-boser. Feedback prize - english language learning, 2022. URL https://kaggle.com/competitions/feedback-prize-english-language-learning.

Future Of Life Institute. Pause giant ai experiments: An open letter, Feb 2024. URL https://futureoflife.org/open-letter/pause-giant-ai-experiments/.

P. Gage. A new algorithm for data compression, 1994. URL https://api.semanticscholar.org/CorpusID:59804030.

E. Giboulot and F. Teddy. Watermax: breaking the llm watermark detectability-robustness-quality trade-off, 2024.

A. Grinbaum and L. Adomaitis. The ethical need for watermarks in machine-generated language, 2022.

Z. He, B. Zhou, H. Hao, A. Liu, X. Wang, Z. Tu, Z. Zhang, and R. Wang. Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models, 2024.

A. B. Hou, J. Zhang, Y. Wang, D. Khashabi, and T. He. k-semstamp: A clustering-based semantic watermark for detection of machine-generated text, 2024.

A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, and W. E. Sayed. Mistral 7b, 2023.

M. Karpinska, K. Thai, K. Krishna, J. Wieting, M. Inghilleri, and M. Iyyer. Par3, 5 2022. URL https://github.com/ngram-lab/par3.

J. King, P. Baffour, S. Crossley, R. Holbrook, and M. Demkin. Llm - detect ai generated text, 2023. URL https://kaggle.com/competitions/llm-detect-ai-generated-text.

J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein. A watermark for large language models, 2023.

K. Krishna, Y. Song, M. Karpinska, J. Wieting, and M. Iyyer. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense, 2023.

R. Kuditipudi, J. Thickstun, T. Hashimoto, and P. Liang. Robust distortion-free watermarks for language models, 2023.

L. Leffer. "ai anxiety" is on the rise–here's how to manage it, Feb 2024. URL https://www.scientificamerican.com/article/ai-anxiety-is-on-the-rise-heres-how-to-manage-it/.

A. Liu, L. Pan, X. Hu, S. Meng, and L. Wen. A semantic invariant robust watermark for large language models, 2024.

Y. Liu and Y. Bu. Adaptive text watermark for large language models, 2024.

E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, and C. Finn. Detectgpt: Zero-shot machine-generated text detection using probability curvature, 2023.

OpenAI. New ai classifier for indicating ai-written text. https://openai.com/blog/new-ai-classifier-for-indicating-ai-written-text, 2023. Accessed: 04-03-2024.

Q. Pang, S. Hu, W. Zheng, and V. Smith. Attacking llm watermarks by exploiting their strengths, 2024.

A. Paullier. Daigt | external dataset, version 1, 2023. URL https://www.kaggle.com/datasets/alejopaullier/daigt-external-dataset/data. Retrieved 23rd December 2023.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023.

V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, and S. Feizi. Can ai-generated text be reliably detected?, 2023.

R. Sato, Y. Takezawa, H. Bao, K. Niwa, and M. Yamada. Embarrassingly simple text watermarks, 2023.

R. Sennrich, B. Haddow, and A. Birch. Neural machine translation of rare words with subword units, 2016.

Y. Tay, M. Dehghani, J. Rao, W. Fedus, S. Abnar, H. W. Chung, S. Narang, D. Yogatama, A. Vaswani, and D. Metzler. Scale efficiently: Insights from pre-training and fine-tuning transformers, 2022.

The White House. Biden-Harris administration secures voluntary commitments from leading artificial intelligence companies to manage the risks posed by AI. https://www.whitehouse.gov/briefing-room/statements-releases/2023/07/21/fact-sheet-biden-harris-administration-secures-voluntary-commitments-from-leading July 2023. Accessed: 2024-3-16.

J. Tiedemann and S. Thottingal. OPUS-MT — Building open translation services for the World, 2020.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2023.

M. K. Vladimir Vorobev. A paraphrasing model based on chatgpt paraphrases, 2023.

J. Wieting, K. Gimpel, G. Neubig, and T. Berg-Kirkpatrick. Paraphrastic representations at scale, 2023.

T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. Transformers: State-of-the-art natural language processing, Oct. 2020. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.