Do Language Models Plagiarize?

Jooyoung Lee
jfl5838@psu.edu
Penn State University
University Park, PA, USA

Thai Le
thaile@olemiss.edu
University of Mississippi
Oxford, MS, USA

Jinghui Chen
jzc5917@psu.edu
Penn State University
University Park, PA, USA

Dongwon Lee
dongwon@psu.edu
Penn State University
University Park, PA, USA

ABSTRACT

Past literature has illustrated that language models (LMs) often *memorize* parts of training instances and reproduce them in natural language generation (NLG) processes. However, it is unclear to what extent LMs "reuse" a training corpus. For instance, models can generate paraphrased sentences that are contextually similar to training samples. In this work, therefore, we study three types of *plagiarism* (i.e., verbatim, paraphrase, and idea) among GPT-2 generated texts, in comparison to its training data, and further analyze the plagiarism patterns of fine-tuned LMs with domain-specific corpora which are extensively used in practice. Our results suggest that (1) three types of plagiarism widely exist in LMs beyond memorization, (2) both size and decoding methods of LMs are strongly associated with the degrees of plagiarism they exhibit, and (3) fine-tuned LMs' plagiarism patterns vary based on their corpus similarity and homogeneity. Given that a majority of LMs' training data is scraped from the Web *without informing content owners*, their reiteration of words, phrases, and even core ideas from training sets into generated texts has ethical implications. Their patterns are likely to exacerbate as both the size of LMs and their training data increase, raising concerns about indiscriminately pursuing larger models with larger training corpora. Plagiarized content can also contain individuals' personal and sensitive information. These findings overall cast doubt on the practicality of current LMs in mission-critical writing tasks and urge more discussions around the observed phenomena. *Data and source code are available at https://github.com/Brit7777/LM-plagiarism.*

CCS CONCEPTS

• **Computing methodologies** → **Natural language generation**.

KEYWORDS

Language Models, Natural Language Generation, Plagiarism

WWW '23, May 1–5, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/\@acmDOI

ACM Reference Format:
Jooyoung Lee, Thai Le, Jinghui Chen, and Dongwon Lee. 2023. Do Language Models Plagiarize?. In *Proceedings of the ACM Web Conference 2023 (WWW '23), May 1–5, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/\@acmDOI

1 INTRODUCTION

Language Models (LMs) have become core elements of Natural Language Processing (NLP) solutions, excelling in a wide range of tasks such as natural language generation (NLG), speech recognition, machine translation, and question answering. The development of large-scale text corpora (generally scraped from the Web) has enabled researchers to train increasingly large-scale LMs. Especially, large-scale LMs have demonstrated unprecedented performance on NLG such that LM-generated texts routinely show more novel and interesting stories than human writings do [35], and the distinction between machine-authored and human-written texts has become non-trivial [52, 53]. As a result, there has been a significant increase in the use of LMs in user-facing products and critical applications.

Concerning the fast-growing adoption of language technologies, it is important to educate citizens and practitioners about the potential ethical, social, and privacy harms of these LMs, as well as strategies and techniques for preventing LMs from adversely impacting people. A body of recent studies has attempted to identify such hazards by examining LMs' capabilities in generating biased and hateful content [41], spreading misinformation [3], and violating users' privacy [12]. Particularly, it was shown that machine-generated texts can include individuals' private information such as phone number and email address due to LMs' over-memorization of training samples [11].

Some may argue that, since one's private information was publicly available in the first place, it is not a problem for LMs to *memorize* and emit it in the generated texts. Still, the current data collection processes (for building training corpora) do not consider how that particular piece of information has been originally released [9]. For example, it is possible for malicious attackers to hack an individual's private data and intentionally post it online. While training LMs on corpora explicitly intended for public use with creators' consents is ideal, it is challenging to achieve in practice.

Note that over-memorization can be perceived as a threat to the authorship and originality of training instances, as training sets for LMs are routinely downloaded from the Internet without the explicit approval of content owners [9]. This behavior is known as **plagiarism**–i.e., *the act of exploiting another person's work or idea without referencing the individual as its author* [4]. As shown in

Type	Machine-Written Text	Training Text
Verbatim	*** is the second amendment columnist for Breitbart news and host of bullets with ***, a Breitbart news podcast. [...] (***Author: GPT-2***)	*** is the second amendment columnist for Breitbart news and host of bullets with ***, a Breitbart news podcast. [...]
Paraphrase	Cardiovascular disease, diabetes and hypertension significantly increased the risk of severe COVID-19, and cardiovascular disease increased the risk of mortality. (***Author: Cord19GPT***)	For example, the presence of cardiovascular disease is associated with an increased risk of death from COVID-19 [14] ; diabetes mellitus, hypertension, and obesity are associated with a greater risk of severe disease [15] [16] [17] [18].
Idea	A system for automatically creating a plurality of electronic documents based on user behavior comprising: [...] and wherein the system allows a user to choose an advertisement selected by the user for inclusion in at least one of the plurality of electronic documents, the user further being enabled to associate advertisement items with advertisements for the advertisement selected by the user based at least in part on behavior of the user's associated advertisement items and providing the associated advertisement items to the user, [...] . (***Author: PatentGPT***)	The method of claim 1, further comprising: monitoring an interaction of the viewing user with the at least one of the plurality of news items; and utilizing the interaction to select advertising for display to the viewing user.

Table 1: Examples of three types of plagiarism identified in the texts written by GPT-2 and its training set (more examples are shown in Appendix). Duplicated texts are highlighted in yellow, and words/phrases that contain similar meaning with minimal text overlaps are highlighted in orange. [...] indicates the texts omitted for brevity. Personally identifiable information (PII) was masked as *.**

Table 1, for instance, plagiarized content written by a machine may contain not only explicit text overlap but also semantically similar information. Existing memorization studies on LMs have focused only on the memorized sequences that are identical to training sequences [12, 30, 59]. This motivates our main inquiry of this work: *To what extent (not limited to memorization) do LMs exploit phrases or sentences from their training samples?*

On the other hand, the fine-tuning paradigm is widely used in LMs for downstream NLP tasks. Specifically, LMs are initially pre-trained on a massive and diverse corpus and then fine-tuned using a smaller task-specific dataset. This enables LMs to create texts in specific domains such as poetry [16] and song lyrics [49]. These tasks require creativity and authenticity, which LMs are prone to fail in. Therefore, the generation outputs of LMs have great moral and ethical implications. Despite increasing efforts to comprehend the over-memorization of pre-trained LMs, to the best of our knowledge, no prior literature has studied on the memorizing behavior of fine-tuned LMs from both pre-training and fine-tuning corpora.

To fill this void of our understanding on the limits of LMs, in this paper, we examine the plagiarizing behaviors of pre-trained and fine-tuned LMs. Our study is guided by two research questions: **(RQ1) Do pre-trained LMs plagiarize?** and **(RQ2) Do fine-tuned LMs plagiarize?**. Specifically, we use OpenAI's GPT-2 [44] for studying these inquiries.[1] We first construct a novel pipeline for automated plagiarism detection and use it to identify three types of plagiarism (i.e., *verbatim*, *paraphrase*, *idea* plagiarism) from passages generated by pre-trained GPT-2 with different combinations of model sizes and decoding methods. For RQ2, three GPT-2 models are fine-tuned using datasets in scholarly writing and legal domains, which are later used for comparing plagiarism from pre-training and fine-tuning corpora.

Our results demonstrate that machine-generated texts do plagiarize from training samples, across all three types of plagiarism. We discover three attributes that impact LMs' plagiarism: 1) *model size*: larger models plagiarize more from a training set than smaller models; 2) *decoding methods:* decoding the outputs after limiting the output space via top-p and top-k strategies are positively related to heightened plagiarism levels as opposed to a raw vocabulary distribution; 3) *corpus similarity and homogeneity:* a higher corpus similarity level across pre-training and fine-tuning corpora, as well as within fine-tuning corpora, enhances the degree of plagiarism for a fine-tuned model.

In summary, our work makes the following contributions:

- By leveraging a BERT-based classifier together with Named Entity Recognition (NER) on top of Sanchez-Perez et al. [48]'s plagiarism detection model, we empirically highlight that LMs do more than copying and pasting texts in a training set; it further rephrases sentences or mimics ideas from other writings without properly crediting the source.
- To the best of our knowledge, this is the first work to systematically study the plagiarizing behavior of *fine-tuned* LMs. Specifically, we find that restricting intra- and inter-corpus similarity can considerably decrease the rate of plagiarism.
- We provide a deeper understanding of the factors that influence LMs' plagiarizing patterns such as model size, decoding strategies, and a fine-tuning corpus. Our results add value to the ongoing discussion around memorization in modern LMs and pave the way for future research into designing robust, reliable, and responsible LMs.

2 RELATED WORK

2.1 Memorization in LMs

There is a growing body of literature that aims to study the memorization of neural LMs by recovering texts in the training corpus [31, 47] or extracting artificially injected canaries [37, 58]. Carlini et al. [12] and Brown et al. [9] emphasized that data memorization can intentionally or unintentionally lead to sensitive information leakage from a model's training set. Meanwhile, recent studies [25, 30] have shown that training data of LMs tend to contain a large number of near-duplicates, and overlapping phrases included in near-duplicates significantly account for memorized text sequences. In order to distinguish rare but memorized texts from trivial examples, Zhang et al.

[1] We chose GPT-2 (instead of more recent LMs such as GPT-3) as it is the latest LM whose replicated training corpus is available. Also, GPT-2 is very popular, ranked as one of the most downloaded LMs from Hugging Face.

[59] presented a notion of counterfactual memorization which measures a difference in the expected performance of two models trained with or without a particular training sample.

Still, none of these works have explored beyond text overlap. The most relevant research to ours is McCoy et al. [35], which analyzed the novelty of machine-generated texts. Although authors found 1,000 word-long duplicated passages from a training set, they concluded that neural LMs can integrate familiar parts into novel content, rather than simply copying training samples. However, because they did not directly compare identified novel content with training samples, the level of plagiarism is uncertain.

2.2 Automatic Plagiarism Detection

Automated extrinsic plagiarism detection, in general, can be divided into two subtasks: document retrieval and text alignment. While document retrieval focuses on fetching all documents that potentially have plagiarized an existing document, the text alignment subtask detects the location and content of plagiarized texts. Alzahrani [6] retrieved candidate documents that share exactly copied sequences and computed the similarity between overlapping 8-grams. There are diverse ways to measure text similarity with segmented document pairs. For example, Küppers and Conrad [27] calculated the Dice coefficient between 250 character chunks of passage pairs, and Shrestha and Solorio [50] implemented the Jaccard similarity with n-grams.

More recently, there has been continuous efforts in incorporating word embedding and advanced machine learning or deep learning models for plagiarism detection. Agarwal et al. [2] used Convolutional Neural Network (CNN) to obtain the local region information from n-grams and applied Recurrent Neural Network (RNN) to capture the long-term dependency information. Similarly, Altheneyan and Menai [5] viewed the task as a classification problem and developed a support vector machine (SVM) classifier using several lexical, syntactic, and semantic features. In our proposed method, we combine conventional similarity measurements and state-of-the-art models to maximize the detection performance.

3 PLAGIARISM: DEFINITION AND DETECTION

3.1 Taxonomy of Plagiarism

Plagiarism occurs when any content including text, source code, or audio-visual content is reused without permission or citation from an author of the original work [14, 40]. It has been a longstanding problem, especially in educational and research institutions or publishers, given the availability of digital artifacts [13]. Plagiarism can severely damage academic integrity and even hurt individuals' reputation and morality [18]. To detect such activities, it is necessary to have extensive knowledge about plagiarism forms and classes.

In this work, we focus on the three most commonly studied plagiarism types: *verbatim* plagiarism, *paraphrase* plagiarism, and *idea* plagiarism. Verbatim plagiarism, which can be considered as the most naive approach, is to directly copy segments of others' documents and paste them into their writings [17]. To make plagiarism less obvious, one may incorporate paraphrase plagiarism by replacing original words with synonyms or rearrange word orders [7]. Similarly, back translation, using two independent translators to translate sentences back and forth, is common in generating paraphrases. Lastly, reuse of the core idea from the original content, also known as idea plagiarism, is a challenging case for an automatic detection due to limited lexical and syntactic similarities. Hence, existing literature (e.g., Gupta et al. [21], Vani and Gupta [54]) specified the task to capture whether a document embeds a summary of another document. While paraphrase plagiarism targets sentence-to-sentence transformations, idea plagiarism reads a chunk of the content and condenses its main information into fewer sentences (or vice versa). In essence, in this work, we adopt the following definition of three plagiarism types:

- **Verbatim plagiarism**: exact copies of words or phrases without transformation.
- **Paraphrase plagiarism**: synonymous substitution, word reordering, and/or back translation.
- **Idea plagiarism**: representation of core content in an elongated form.

3.2 Automatic Detection of Plagiarism

In this section, we introduce a two-step approach for automated plagiarism detection. Suppose we have n documents in a corpus $D=\{d_1, d_2, ... d_n\}$ and a query document d_q. The goal is to identify a pair of "plagiarized" text segments (s_1, s_2) such that s_1 (resp. s_2) is a text segment within a document $d_i \in D$ (resp. d_q).

Step 1 (Finding Top-n' Candidate Documents): First, for the given query document d_q, we aim to quickly narrow down to top-n' documents (out of n documents, where $n' \ll n$) which are likely to contain plagiarized pieces of texts. To do this, we utilize a document similarity score as a proxy for plagiarism. Since recent LMs are generally trained on gigantic corpora, it is non-trivial to store them locally and compute a pair-wise document similarity. Hence, we implement a search engine using Elasticsearch[2], an open-source search engine built on Apache Lucene that provides a distributed RESTful search service with a fast response time. After storing the entire training documents D in Elasticsearch, using a machine-generated document as the query document d_q, we retrieve top-n' most-similar documents. Elasticsearch utilizes the Okapi-BM25 algorithm [46], a popular bag-of-words ranking function, by default. We used $n' = 10$ in experiments for the sake of time efficiency.[3]

Step 2 (Finding Plagiarized Text Pairs and Plagiarism Type): Next, using the identified n' candidates $\{d_1, d_2, ..., d_{n'}\}$ for the query document d_q, we aim to find plagiarized text pairs (s_1, s_2) such that s_2 is one of three types of plagiarism against s_1. For this task, we exploit text alignment algorithms that locate and extract most-similar contiguous text sequences between two given documents. Such text alignment algorithms are applicable to various tasks such as text-reuse detection [51] and translation alignment [33]. In particular, we employ the improved version of the winning method at the plagiarism detection competition of PAN 2014.[4] Following, we

[2] https://www.elastic.co/elasticsearch/

[3] We performed a post-hoc analysis with a smaller ($n' = 5$) and a larger value ($n' = 30$) of n' using GPT-2 xl to gauge its potential effects on identified plagiarism rates. The results showed a marginal difference (e.g., 1.46% ($n' = 5$) vs. 1.54% ($n' = 30$) for temperature setting), indicating that the choice of the n' value does not drastically influence our findings.

[4] https://pan.webis.de/clef14/pan14-web/text-alignment.html

Scores	PanDataset			GptPlagiarismDataset		
	Verbatim	Paraphrase	Idea	Verbatim	Paraphrase	Idea
Precision	0.995	1.00	1.00	0.96	0.846	0.99
Recall	0.986	0.723	0.412	0.87	0.785	0.3

Table 2: Evaluation results of our plagiarism detection pipeline. For PanDataset, we perform the evaluation in a binary classification setting (e.g., verbatim plagiarism vs. no plagiarism). Since GptPlagiarismDataset does not take into account document pairs without plagiarism, we adopt a multi-nomial classification setting (e.g., verbatim plagiarism vs. paraphrase/idea plagiarism).

explain details on Sanchez-Perez et al. [48] and our improvement strategies.

Current Approach (Sanchez-Perez et al. [48]). Their methods consist of five steps which include (1) text-preprocessing (lower-casing all characters, tokenizing, and stemming); (2) obfuscation type identification (verbatim/random/translation/summary obfuscation); (3) seeding (deconstructing long passages into smaller segments and finding candidate pairs through sentence-level similarity measurement given two documents); (4) extension (forming larger text fragments that are similar via clustering); and (5) filtering (removing overlapping and short plagiarized fragments). In summary, they transform the suspicious and source sentences as term frequency–inverse document frequency vector weights and then calculate the similarity between the sentence pairs using the dice coefficient and cosine measure. Adaptive parameter selection is achieved by testing two settings recursively for the summary obfuscation corpus and the other three corpora.

Our Improvements. To verify the effectiveness of Sanchez-Perez et al. [48] on our corpus, we manually inspected 200 plagiarism detection results. For a fair comparison, the number of sentence pairs in each category (none/verbatim/paraphrase/idea plagiarism) was equally distributed. Our evaluation revealed that Sanchez-Perez et al. [48] induces more false positives than their reported performance, specifically in detecting the paraphrase type plagiarism (0.51 in precision). It resulted from the model's tendency of labeling near-duplicates with one character difference as paraphrases (should be the "verbatim" plagiarism type) and its inability to distinguish a minor entity-level discrepancy such as numerical values or dates. To minimize such errors, after Sanchez-Perez et al. [48] retrieves all paraphrased text segments, we post-process segments by chunking them into sentences with NLTK[5]'s sentence tokenizer and apply a RoBERTa-based paraphrase identification model [38][6] and Named-Entity Recognition (NER)[7] as additional validators. Specifically, when there is at least one sentence pair whose probability score (from the paraphrase detection model) ranges from 0.5 to 0.99[8] and have the exactly matching set of entities, we ultimately accept the plagiarism result by Sanchez-Perez et al. [48]. This additional restriction resulted in the following precision scores: 0.92 for no plagiarism, 1.0 for verbatim type, 0.88 for paraphrase type, and 0.62 for idea type. To gauge both precision and recall, we utilize two additional labeled datasets, PanDataset and GptPlagiarismDataset (refer to Appendix A for more details on datasets). Both precision and recall scores of each label are reported in Table 2. Note that at the end, our plagiarism detection pipeline has high precisions at the cost of low recalls, implying that the number of plagiarism cases we report subsequently is only a "lower-bound" estimate of plagiarism rates that actually exist. For subsequent analyses, we utilize two hyperparameters: (1) the minimum character count of common substrings between the two documents for verbatim plagiarism is set to 256; (2) the minimum character count permitted on either side of a plagiarism case is set to 150. These thresholds are much stricter than minimum 50 tokens (i.e., on average 127 characters) employed by existing works [10, 30]. Again, this ensures that our following report on RQ1 and RQ2 is the "lower-bound" estimate of plagiarism frequencies.

4 RQ1: DO PRE-TRAINED LMS PLAGIARIZE?

4.1 Experimental Setup

Dataset. GPT-2 is pre-trained on WebText, containing over 8 million documents retrieved from 45 million Reddit links. Since OpenAI has not publicly released WebText, we use OpenWebText which is an open-source recreation of the WebText corpus.[9] It has been reliably used by prior literature [25, 34].

Model. GPT-2 is an auto-regressive language model predicting one token at a time in a left-to-right fashion. That is, the probability distribution of a word sequence can be calculated through the product of conditional next word distributions. In response to an arbitrary prompt, GPT-2 can adapt to its style and content and generate artificial texts. GPT-2 comes in 4 different sizes — small, medium, large, and xl, with 124M, 355M, 774M, and 1.5B parameters, respectively. We utilize all of them for analyses.

Text Generation. Given that GPT-2 relies on the probability distribution when generating word-tokens, there exist various decoding methods which are well known to be critical for performance in text generation [24]. We primarily consider the following decoding algorithms:

- Temperature [1]: control the randomness of predictions by dividing the logits by t before applying softmax
- Top-k [19]: filter the k most likely next words and redistribute the probability mass
- Top-p [22]: choose from the smallest possible set of words whose cumulative probability exceeds the probability p

It is reported that increasing parameter values (t, k, p) can notably improve the novelty of machine-generated texts but may also deteriorate their quality sides [35]. Conversely, smaller parameter values tend to yield dull and repetitive sentences [22].

Considering the difficulties in hyper-parameter tuning that can confidently guarantee high-quality machine-authored texts, we use off-the-shelf GPT-2 Output Dataset[10] provided by OpenAI. This

[5] https://www.nltk.org
[6] The RoBERTa classifier has achieved 91.17% accuracy on the evaluation set from the MSRP corpus (https://www.microsoft.com/en-us/download/details.aspx?id=52398).
[7] We use SpaCy library (https://spacy.io).
[8] We specified 0.99 as the upper bound to avoid near-duplicate pairs.
[9] https://skylion007.github.io/OpenWebTextCorpus/
[10] https://github.com/openai/gpt-2-output-dataset

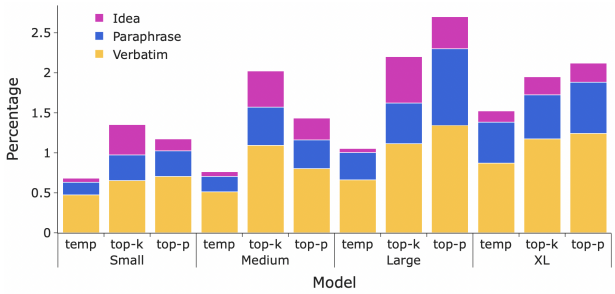

Figure 1: Document percentage w.r.t. three plagiarism types from pre-training data

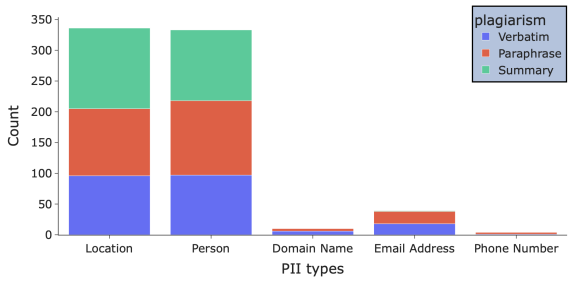

Figure 2: Number of unique PII-exposing substrings associated with plagiarism categories

dataset has been reliably used by Kushnareva et al. [28] and Wolff and Wolff [57] for neural text detection. Specifically, It contains 250,000 texts generated by four versions of the GPT-2 model with aforementioned decoding approaches. Owners of the repository have informed us that they used a '<|endoftext|>' token as a prompt and set t=1, k=40, $0.8<p<1$.[11]. In total, there are 12 (i.e., 4 model size * 3 decoding methods) combinations, and we analyze 10,000 documents in each combination.

4.2 Results

We discover that pre-trained GPT-2 families do plagiarize from the OpenWebText. Figure 1 illustrates the percentage of unique machine-written documents regarding three plagiarism types based on different model sizes and decoding strategies[12]. Consistent with [12, 32], the larger the model size became, the higher occurrences of plagiarism were observed when using temperature sampling. The general trend still holds when GPT-2's word token is sampled with top-k and top-p truncation except for the xl model size. However, interestingly, plagiarism frequencies were the highest when GPT-2 large models were used, not xl. We also find that decoding methods affect models' plagiarism. More precisely, top-k and top-p sampling are more strongly associated with plagiarism than decoding with temperature regardless of the model size. We conjecture that this discrepancy is due to the fact that top-k and top-p decoding methods disregard less probable tokens unlike random sampling, which may push models to choose a memorized one as a next token.

4.3 Qualitative Examination of Plagiarized Texts

Lengths and Occurrences. Motivated by prior memorization studies [10, 30], we inspect lengths and occurrences of texts that are associated with verbatim plagiarism. We find that the median length of memorized texts is 483 characters, and the longest texts contain 5,920 characters. In order to efficiently count the occurrences of plagiarized strings within OpenWebText, we utilize the established Elasticsearch pipeline, which includes setting plagiarized texts as search queries and retrieving documents that embed provided texts.[13] We find that some memorized sequences are from highly duplicated texts throughout the training corpus: the newsletter sign-up text [14] appeared at most 9,978 times and was memorized. Still, there exist many instances where models memorize without seeing them more than two times. While the median of occurrences for memorized texts is 6, sequences related to paraphrase or idea plagiarism are prone to not appear at all from training samples (median = 0).

Inclusion of Sensitive Information. We now turn our attention to whether sequences associated with three plagiarism types contain individuals' personal or sensitive data. To achieve this, we use Microsoft's Presidio analyzer,[15] a Python toolkit for personally identifiable information (PII) entity detection (e.g., credit card information, email address, phone number). There are a total of 1,193 unique text sequences (verbatim: 388, paraphrase: 507, and idea: 298) plagiarized by pre-trained GPT-2. We set a confidence threshold to 0.7. A total number of plagiarized documents that reveal PII entities is shown in Figure 2. Of 1,193 plagiarized sequences, nearly 28% include at least one element of location information and a person's full name. Although none of highly sensitive information (e.g., driver license number, credit card information, bank number, social security number, and IP address) is revealed, the results show a possibility of machine-generated texts disseminating personal data such as phone number and email address through all three types of plagiarism.

5 RQ2: DO FINE-TUNED LMS PLAGIARIZE?

5.1 Experimental Setup

Dataset. We choose public English datasets related to scholarly and legal writings because plagiarism is deemed more sensitive and intolerable in these domains [42]. Three datasets are:

- **ArxivAbstract**: includes 250,000 randomly selected abstracts on arxiv.org, from the start of the site in 1993 to the end of 2019 [20]. It covers a wide range of disciplines (e.g., Physics, Computer Science, Economics).
- **Cord-19**: consists of 500,000 scholarly articles about the COVID-19 virus [55]. Medicine (55%), Biology (31%), and Chemistry

[11] Equivalent to existing literature [15, 35], we only report results of these specific hyperparameters because they were recommended by GPT-2 creators [44] Also, our findings on the decoding methods were validated by additional experiments with more diverse parameter values.

[12] Please note that sentences with proper quotation marks within identified plagiarism cases were excluded from the analyses, as they do not constitute plagiarism.

[13] By default, Elasticsearch does not allow searches to return more than the top 10,000 matching hits.

[14] "newsletter sign up continue reading the main story please verify you're not a robot by clicking the box. invalid email address. please re-enter..."

[15] https://microsoft.github.io/presidio/analyzer/

Model	Decoding	Plagiarism from Pre-Training Data			Plagiarism from Fine-Tuning Data		
		Verbatim	Paraphrase	Idea	Verbatim	Paraphrase	Idea
Pre-trained GPT	temp	47 (0.47%)	16 (0.16%)	5 (0.05%)	N/A		
	top-k	65 (0.65%)	32 (0.32%)	38 (0.38%)			
	top-p	**70 (0.7%)**	32 (0.32%)	15 (0.15%)			
Patent GPT	temp	0 (0%)	36 (0.36%)	21 (0.21%)	0 (0%)	32 (0.32%)	17 (0.17%)
	top-k	0 (0%)	**171 (1.71%)**	**161 (1.61%)**	0 (0%)	2 (0.02%)	0 (0%)
	top-p	0 (0%)	94 (0.94%)	130 (1.3%)	0 (0%)	3 (0.03%)	0 (0%)
Cord19 GPT	temp	0 (0%)	6 (0.06%)	6 (0.06%)	43 (0.43%)	90 (0.9%)	42 (0.42%)
	top-k	0 (0%)	79 (0.79%)	122 (1.22%)	46 (0.46%)	**548 (5.48%)**	**485 (4.85%)**
	top-p	2 (0.02%)	57 (0.57%)	79 (0.79%)	**72 (0.72%)**	388 (3.88%)	228 (2.28%)
ArxivAbstract GPT	temp	0 (0%)	0 (0%)	0 (0%)	0 (0%)	3 (0.03%)	0 (0%)
	top-k	0 (0%)	0 (0%)	1 (0.01%)	0 (0%)	0 (0%)	0 (0%)
	top-p	0 (0%)	2 (0.02%)	0 (0%)	0 (0%)	2 (0.02%)	0 (0%)

Table 3: Number (%) of machine-written documents w.r.t. three plagiarism types from pre-training & fine-tuning data. Blue represents the pre-trained model, whereas pink represents the fine-trained model. in A total number of documents we generated for each model and decoding methods is 10,000.

(3%) are primary domains of this corpus. For fine-tuning purposes, we randomly sample 200,000 documents.[16]

- **PatentClaim**: is provided by Lee and Hsiang [29] and has 277,947 patent claims in total.

Model. Using these datasets, we fine-tune three independent GPT-2 small models[17] and denote them as *ArXivAbstractGPT*, *Cord19GPT*, and *PatentGPT*, respectively. The details on training configurations can be found in Appendix B.

Text Generation. For three fine-tuned models, we manually create 10,000 machine-generated texts using the same prompt and parameter settings as GPT-2 Output Dataset.

5.2 Results

We compare plagiarizing behaviors of three fine-tuned models using both pre-training (OpenWebText) and fine-tuning datasets (PatentClaim, Cord-19, ArxivAbstract) in Table 3. Our findings show that fine-tuning significantly reduces verbatim plagiarism cases from OpenWebText. This observation aligns with GPT-2's outstanding adaptability to the writing styles of a new corpus. Yet, not all fine-tuned models are plagiarism-free; for PatentGPT and Cord19GPT, the remaining plagiarism types regarding OpenWebText occurred more frequently than the pre-trained GPT. Meanwhile, ArxivAbstractGPT barely plagiarized texts from OpenWebText. Interestingly, models' plagiarism behaviors change when we compare their generated texts against the fine-tuning samples. Cord19GPT was strongly affiliated with plagiarism, whereas the other two models were not.

These results suggest that, although three models are fine-tuned in a similar setting (regarding dataset size and training duration), their patterns of plagiarism vary. We hypothesize that there are external factors that affect models' plagiarism. For example, if fine-tuning and pre-training corpora have multiple similar or duplicated content, the fine-tuned model would have been immensely exposed to it and

[16]Since most articles in CordD-19 exceed the length of 1,024 tokens, we only consider the first five paragraphs starting from the 'Introduction' section.

[17]Due to constraints of computing resource, we only fine-tune the GPT-2 small variation.

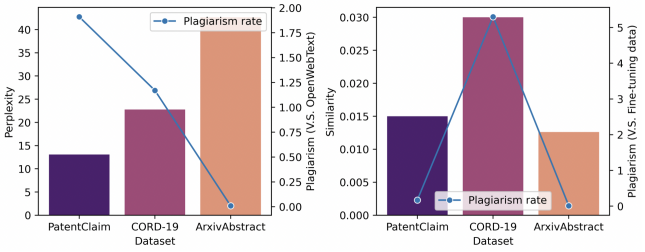

Figure 3: Perplexity (left) and similarity scores (right) of training data. Plagiarism rate represents the average percentage of all plagiarism categories using the three decoding methods.

may have started to remember it. Lee et al. [30] has shown a positive relationship between memorized sequences and their frequencies in a training set. Similarly, it is also possible that over-exposure to particular texts may have been resulted from similar documents within fine-tuning data. Next, we analyze a corpus similarity between fine-tuning data and pre-training data and a homogeneity of fine-tuning data in Section 6 to verify our hypotheses.

6 PLAGIARISM V.S. INTRA- AND INTER-CORPUS SIMILARITY

6.1 Inter-Corpus Similarity (across Datasets)

Method. There are various methods to compute a corpus similarity. Generally speaking, we first transform document pairs into vectors, apply pair-wise document similarity measurements, and then aggregate them. Yet, since the size of OpenWebText is huge, it is computationally expensive to employ conventional approaches. Thus, inspired by Kilgarriff and Rose [26] and Carlini et al. [12], we utilize perplexity measures. The *perplexity* of a sequence estimates the confidence levels of an LM when predicting the inclusive tokens in a specific order. To compute the corpus similarities of pre-training and fine-tuning sets, we retrieve the perplexity of the pre-trained

Model	Decoding	**Before** Filtering Low Perplexity			**After** Filtering Low Perplexity		
		Verbatim	Paraphrase	Idea	Verbatim	Paraphrase	Idea
Patent GPT	temp*	0 (0%)	26 (0.52%)	11 (0.22%)	0 (0%)	11 (0.22%)	9 (0.18%)
	top-k*	0 (0%)	109 (2.18%)	109 (2.18%)	0 (0%)	79 (1.58%)	54 (1.08%)
	top-p*	0 (0%)	66 (1.32%)	59 (1.18%)	0 (0%)	41 (0.82%)	27 (0.54%)
Cord19 GPT	temp	0 (0%)	7 (0.14%)	6 (0.12%)	0 (0%)	4 (0.08%)	1 (0.02%)
	top-k*	0 (0%)	67 (1.34%)	106 (1.12%)	0 (0%)	56 (1.12%)	36 (0.72%)
	top-p*	5 (0.1%)	54 (1.08%)	59 (1.18%)	0 (0%)	35 (0.7%)	25 (0.5%)

Table 4: Number (%) of machine-generated documents w.r.t. three plagiarism types before/after removing training samples with low perplexity. The total number of generated documents for each model and decoding method is 5,000. * indicates a statistical significance ($p < 0.05$).

GPT-2 on the fine-tuning dataset. Due to the limited space, we refer the readers to the Appendix C for a detailed description of perplexity calculation.

Results. A low perplexity implies that LM is not surprised by the sequence. In our case, the lower the perplexity score is, the more comparable a particular fine-tuned corpus is to OpenWebText. We find that a perplexity score of PatentClaim is the lowest, following Cord-19 and ArxivAbstract (Figure 3). This result concurs with our initial observation where PatentGPT plagiarizes the most from OpenWebText. Subsequently, we create two versions of PatentGPT and Cord19GPT to test the effect of perplexity on plagiarism from OpenWebText. While the first is trained with a subset of fine-tuning samples excluding 30% of the documents with the lowest perplexity, the second does not consider the perplexity.

For a fair comparison, we maintain the same training configurations for all model pairs.[18] Finally, we generate 5,000 documents for each model using three decoding methods and compare their plagiarism. As shown in Table 4, omitting low perplexity documents mitigates the intensity of plagiarism from pre-training data.[19]

6.2 Intra-Corpus Similarity (within Datasets)

Method. Here we adopt a traditional document similarity measurement to quantify inner-similarity levels of fine-tuning datasets. For each fine-tuning data, we first convert all instances into term frequency-inverse document frequency (tf-idf) vectors and then aggregate the averaged cosine similarity over all examples.

Results. We observe that the intra-corpus similarity of Cord-19 is more than twice higher than PatentClaim and ArxivAbstract (Figure 3). This result coincides with our observation in RQ2 where Cord19GPT demonstrates a heightened degree of plagiarism. Moreover, our manual inspection of verbatim plagiarism cases supports that most of them are frequently occurring substrings. For example, a part of BMJ's statement about copyright and authors' rights[20] appeared 588 times in the Cord-19 corpus. We further evaluate a correlation between corpus homogeneity and plagiarism by re-training two Cord19GPT models. Specifically, the former is fine-tuned with randomly selected 188,880 Cord-19 documents whereas the latter is fine-tuned using filtered Cord-19 data where 11,120 highly similar training instances (cosine similarity > 0.8) are removed. They are both trained for roughly 42,390 steps. Table 5 supports the effectiveness of removing similar training instances in reducing plagiarism from fine-tuning data.[21]

7 FINDINGS

1. Larger LMs plagiarize more. Consistent with Carlini et al. [12] and Carlini et al. [10], we find that larger GPT-2 models (large and xl) generally generate plagiarized sequences more frequently than smaller ones. Depending on the decoding approaches, however, the model size that yields the largest amount of plagiarism change: when the next token is sampled from truncated distribution, the GPT-2 large model plagiarizes the most. On the other hand, the GPT-2 xl becomes more strongly associated with plagiarism than the GPT-2 large when the temperature setting without truncation is employed. This discrepancy may be attributable to the error rates of our paraphrase and idea plagiarism detection tool. Regardless, it is evident that larger models plagiarize notably more from training data. Considering the performance improvement of LMs with larger model sizes, this finding sheds light on a trade-off between the performance and copyright protection issues.

2. Decoding algorithms affect plagiarism. Varying effects of decoding methods and parameters on text quality and diversity have been extensively studied [8, 15], but not from the plagiarism perspective. Particularly, top-p sampling is reported to be the most effective decoding method in generating high-quality texts [23]. Despite its efficiency in balancing quality and novelty, our analysis shows that sampling with top-p or top-k truncation leads to more plagiarism cases. This result shows that these popular sampling approaches still pose critical flaws because they have not been thoroughly vetted in terms of plagiarism. Thus, it is necessary to carefully choose and evaluate decoding methods not only through the lens of quality and diversity but also through the originality aspect.

3. Fine-tuning LMs matter. Our findings highlight that fine-tuning a model with domain-specific data can mitigate verbatim plagiarism from the pre-training dataset. Still, other types of plagiarism cases have surged, in the case of PatentGPT and Cord19GPT, alongside corpus similarity levels between pre-training and fine-tuning corpora. Moreover, we observe that models' plagiarism differs depending on similarity degrees within a fine-tuning corpus. Our research validates

[18] PatentGPT variations are trained on 189,000 documents for 22,000 steps, whereas Cord19 variations are trained on 140,000 documents for 40,850 steps.
[19] Refer to Appendix D for statistical testing results.
[20] https://authors.bmj.com/policies/copyright-and-authors-rights/
[21] Refer to Appendix D for statistical testing results.

Model	Decoding	**Before** Filtering Similar Documents			**After** Filtering Similar Documents		
		Verbatim	Paraphrase	Idea	Verbatim	Paraphrase	Idea
CORD19 GPT	temp	15 (0.3%)	64 (1.28%)	22 (0.44%)	10 (0.2%)	49 (0.98%)	25 (0.5%)
	top-k*	11 (0.22%)	301 (6.02%)	238 (4.76%)	11 (0.22%)	203 (4.06%)	184 (3.68%)
	top-p*	21 (0.42%)	190 (3.8%)	111 (2.22%)	11 (0.22%)	153 (3.06%)	94 (1.88%)

Table 5: Number (%) of machine-generated documents w.r.t. three plagiarism types before/after removing similar training samples. The total number of generated documents for each model and decoding method is 5,000. * indicates a statistical significance ($p < 0.05$).

their relationships by comparing the rate of plagiarism before and after removing syntactically or semantically similar instances in fine-tuning data. Indeed, restricting inter- and intra-corpus similarity can reduce the frequency of all plagiarism types. This result can further be expanded as a simple solution to LMs' plagiarism issues.

4. LMs can pose privacy harms. Our qualitative examination of plagiarized texts reveals that LMs expose individuals' sensitive or private data not only through verbatim plagiarism but also paraphrase and idea plagiarism. Although all identified contents were publicly available on the Web, emitting such sensitive information in the generated texts can raise a serious concern. This finding adds value to the ongoing discussion around privacy breaches from the memorization of modern LMs.

8 DISCUSSION AND ETHICS

Discussion. In this work, we develop a novel pipeline for investigating LMs' plagiarism in text generation processes and characterize a shift in plagiarism rates resulting from three attributes (i.e., model size, decoding methods, and corpus similarities). The datasets utilized to train the models are the subject of this study. We use GPT-2 as a representative LM to study because it is one of the most downloaded LMs from Hugging Face at the end of 2022,[22] and its reproduced training corpus is publicly accessible (which is a necessary condition to study the plagiarism of LMs). However, different LMs may demonstrate different patterns of plagiarism, and thus our results may not directly generalize to other LMs, including more recent LMs such as GPT-3 or BLOOM. Future work can revisit the proposed research questions against more diverse or modern LMs.

In addition, automatic plagiarism detectors are known to have many failure modes (both in false negatives and false positives) [56]. Our plagiarism detection pipeline of Section 3.2 is no exception. However, achieving a high precision with a low recall is not a major issue in our problem domain, as we focus on demonstrating the lower-bound of the plagiarism vulnerability in LMs (and in reality, there are likely to be many more plagiarism cases that we missed to detect due to low recalls). Likewise, prior memorization works [12, 25] documented the lower-bound of the plagiarism susceptibility and showed a small number of memorized instances. Regardless, they were effective in inspiring others to continue exploring this important phenomenon. As a result, we hope that our current finding becomes useful to stimulate and raise public awareness about the plagiarism behavior of popular LMs like GPT-2.

We also stress that distinguishing whether a reproduction of training datasets is a positive attribute of LM or not is beyond the scope of this work. It is highly context-dependent [30], and thus necessitates more sophisticated methods to disentangle. In our experiments, we treat all instances of LM-generated texts that reiterate training examples as "problematic", as the fine-tuning datasets we analyzed are in academic and legal contexts where originality is valued.

Ultimately, a primary purpose of the exploration of the intra- and inter-corpus similarity in models' authorship violation is to support our hypotheses and further motivate researchers to take this into account when developing new LMs or fine-tuning current ones. Yet, the current approach fails to completely eradicate plagiarism occurrences.

Ethics. Data and code, involving plagiarized texts we identified throughout this research, are available to the research community. Due to the inclusion of individuals' personal data in generated texts, we employed data anonymization techniques prior to distribution. Specifically, we filtered PII such as name, email address, and phone number using Microsoft's Presidio Anonymizer.[23] We recommend that artificial documents generated by fine-tuned GPT-2 be utilized strictly for research purposes.

9 CONCLUSION

Our work presents the first holistic and empirical analyses of plagiarism in LMs by constructing a pipeline for the automatic identification of plagiarized content. We conclude that GPT-2 can exploit and reuse words, sentences, and even core ideas (that are originally included in OpenWebText, a pre-training corpus) in the generated texts. Further, this tendency is prone to exacerbate as the model size increases or certain decoding algorithms are employed. We also discover that untangling corpus similarity and homogeneity can help alleviate plagiarism rates by GPT-2. This is the first study to analyze text generation outputs through the lens of plagiarism. Although the goal of a supervised machine learning system is to learn to mimic the distribution of its training data, we deem it crucial for model users and designers to recognize the observed phenomena. The vulnerability of models to plagiarism can adversely impact societal and ethical norms, particularly in literary disciplines that are intimately connected to creativity and originality. Therefore, we recommend researchers carefully assess the model's intended usage and evaluate its robustness before deployment.

ACKNOWLEDGMENTS

This work was in part supported by NSF awards #1934782 and #2114824.

[22] https://huggingface.co/models?sort=downloads

[23] https://microsoft.github.io/presidio/anonymizer/

REFERENCES

[1] David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. 1985. A learning algorithm for Boltzmann machines. *Cognitive science* 9, 1 (1985), 147–169.

[2] Basant Agarwal, Heri Ramampiaro, Helge Langseth, and Massimiliano Ruocco. 2018. A deep network model for paraphrase detection in short text messages. *Information Processing & Management* 54, 6 (2018), 922–937.

[3] Alim Al Ayub Ahmed, Ayman Aljabouh, Praveen Kumar Donepudi, and Myung Suh Choi. 2021. Detecting Fake News using Machine Learning: A Systematic Literature Review. *arXiv preprint arXiv:2102.04458* (2021).

[4] Asim M El Tahir Ali, Hussam M Dahwa Abdulla, and Vaclav Snasel. 2011. Overview and comparison of plagiarism detection tools.. In *Dateso*. 161–172.

[5] Alaa Saleh Altheneyan and Mohamed El Bachir Menai. 2020. Automatic plagiarism detection in obfuscated text. *Pattern Analysis and Applications* 23, 4 (2020), 1627–1650.

[6] Salha Alzahrani. 2015. Arabic plagiarism detection using word correlation in N-Grams with K-overlapping approach. In *Proceedings of the Workshops at the 7th Forum for Information Retrieval Evaluation (FIRE)*. 123–125.

[7] Alberto Barrón-Cedeño, Marta Vila, M Antònia Martí, and Paolo Rosso. 2013. Plagiarism meets paraphrasing: Insights for the next generation in automatic plagiarism detection. *Computational Linguistics* 39, 4 (2013), 917–947.

[8] Sourya Basu, Govardana Sachitanandam Ramachandran, Nitish Shirish Keskar, and Lav R Varshney. 2020. Mirostat: A neural text decoding algorithm that directly controls perplexity. *arXiv preprint arXiv:2007.14966* (2020).

[9] Hannah Brown, Katherine Lee, Fatemehsadat Mireshghallah, Reza Shokri, and Florian Tramèr. 2022. What Does it Mean for a Language Model to Preserve Privacy? *arXiv preprint arXiv:2202.05520* (2022).

[10] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2022. Quantifying Memorization Across Neural Language Models. *arXiv preprint arXiv:2202.07646* (2022).

[11] Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*. 267–284.

[12] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*. 2633–2650.

[13] Roger Clarke. 2006. Plagiarism by academics: More complex than it seems. *Journal of the Association for Information Systems* 7, 1 (2006), 5.

[14] Georgina Cosma and Mike Joy. 2008. Towards a definition of source-code plagiarism. *IEEE Transactions on Education* 51, 2 (2008), 195–200.

[15] Alexandra DeLucia, Aaron Mueller, Xiang Lisa Li, and João Sedoc. 2020. Decoding methods for neural narrative generation. *arXiv preprint arXiv:2010.07375* (2020).

[16] Liming Deng, Jie Wang, Hangming Liang, Hui Chen, Zhiqiang Xie, Bojin Zhuang, Shaojun Wang, and Jing Xiao. 2020. An iterative polishing framework based on quality aware masked language model for Chinese poetry generation. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 7643–7650.

[17] Ish Kumar Dhammi and Rehan Ul Haq. 2016. What is plagiarism and how to avoid it? *Indian journal of orthopaedics* 50, 6 (2016), 581.

[18] Julianne East. 2010. Judging plagiarism: a problem of morality and convention. *Higher Education* 59, 1 (2010), 69–83.

[19] Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833* (2018).

[20] R. Stuart Geiger. 2019. *ArXiV Archive: A tidy and complete archive of metadata for papers on arxiv.org, 1993-2019*. https://doi.org/10.5281/zenodo.2533436

[21] Deepa Gupta, K Vani, and LM Leema. 2016. Plagiarism detection in text documents using sentence bounded stop word n-grams. *Journal of Engineering Science and Technology* 11, 10 (2016), 1403–1420.

[22] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751* (2019).

[23] Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2019. Automatic detection of generated text is easiest when humans are fooled. *arXiv preprint arXiv:1911.00650* (2019).

[24] Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2019. Human and automatic detection of generated text. (2019).

[25] Nikhil Kandpal, Eric Wallace, and Colin Raffel. 2022. Deduplicating training data mitigates privacy risks in language models. *arXiv preprint arXiv:2202.06539* (2022).

[26] Adam Kilgarriff and Tony Rose. 1998. Measures for corpus similarity and homogeneity. In *Proceedings of the Third Conference on Empirical Methods for Natural Language Processing*. 46–52.

[27] Robin Küppers and Stefan Conrad. 2012. A Set-Based Approach to Plagiarism Detection.. In *CLEF (Online Working Notes/Labs/Workshop)*.

[28] Laida Kushnareva, Daniil Cherniavskii, Vladislav Mikhailov, Ekaterina Artemova, Serguei Barannikov, Alexander Bernstein, Irina Piontkovskaya, Dmitri Piontkovski, and Evgeny Burnaev. 2021. Artificial text detection via examining the topology of attention maps. *arXiv preprint arXiv:2109.04825* (2021).

[29] Jieh-Sheng Lee and Jieh Hsiang. 2020. Patent claim generation by fine-tuning OpenAI GPT-2. *World Patent Information* 62 (2020), 101983.

[30] Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2021. Deduplicating training data makes language models better. *arXiv preprint arXiv:2107.06499* (2021).

[31] Klas Leino and Matt Fredrikson. 2020. Stolen Memories: Leveraging Model Memorization for Calibrated {White-Box} Membership Inference. In *29th USENIX Security Symposium (USENIX Security 20)*. 1605–1622.

[32] Sharon Levy, Michael Saxon, and William Yang Wang. 2021. Investigating Memorization of Conspiracy Theories in Text Generation. *arXiv preprint arXiv:2101.00379* (2021).

[33] Zehui Lin, Xiao Pan, Mingxuan Wang, Xipeng Qiu, Jiangtao Feng, Hao Zhou, and Lei Li. 2020. Pre-training multilingual neural machine translation by leveraging alignment information. *arXiv preprint arXiv:2010.03142* (2020).

[34] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[35] R Thomas McCoy, Paul Smolensky, Tal Linzen, Jianfeng Gao, and Asli Celikyilmaz. 2021. How much do language models copy from their training data? Evaluating linguistic novelty in text generation using RAVEN. *arXiv preprint arXiv:2111.09509* (2021).

[36] Derek Miller. 2019. Leveraging BERT for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165* (2019).

[37] Fatemehsadat Mireshghallah, Huseyin A Inan, Marcello Hasegawa, Victor Rühle, Taylor Berg-Kirkpatrick, and Robert Sim. 2021. Privacy regularization: Joint privacy-utility optimization in language models. *arXiv preprint arXiv:2103.07567* (2021).

[38] John X Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. 2020. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. *arXiv preprint arXiv:2005.05909* (2020).

[39] Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. 2019. Facebook FAIR's WMT19 news translation task submission. *arXiv preprint arXiv:1907.06616* (2019).

[40] Chris Park. 2003. In other (people's) words: Plagiarism by university students–literature and lessons. *Assessment & evaluation in higher education* 28, 5 (2003), 471–488.

[41] John Pavlopoulos, Jeffrey Sorensen, Lucas Dixon, Nithum Thain, and Ion Androutsopoulos. 2020. Toxicity detection: Does context really matter? *arXiv preprint arXiv:2006.00998* (2020).

[42] Diane Pecorari. 2008. *Academic writing and plagiarism: A linguistic analysis*. Bloomsbury Publishing.

[43] Robin L Plackett. 1983. Karl Pearson and the chi-squared test. *International statistical review/revue internationale de statistique* (1983), 59–72.

[44] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog* 1, 8 (2019), 9.

[45] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* 21, 140 (2020), 1–67.

[46] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. 1995. Okapi at TREC-3. *Nist Special Publication Sp* 109 (1995), 109.

[47] Ahmed Salem, Apratim Bhattacharya, Michael Backes, Mario Fritz, and Yang Zhang. 2020. {Updates-Leak}: Data Set Inference and Reconstruction Attacks in Online Learning. In *29th USENIX Security Symposium (USENIX Security 20)*. 1291–1308.

[48] Miguel A Sanchez-Perez, Alexander Gelbukh, and Grigori Sidorov. 2015. Adaptive algorithm for plagiarism detection: The best-performing approach at PAN 2014 text alignment competition. In *International Conference of the Cross-Language Evaluation Forum for European Languages*. Springer, 402–413.

[49] Zhonghao Sheng, Kaitao Song, Xu Tan, Yi Ren, Wei Ye, Shikun Zhang, and Tao Qin. 2020. Songmass: Automatic song writing with pre-training and alignment constraint. *arXiv preprint arXiv:2012.05168* (2020).

[50] Prasha Shrestha and Thamar Solorio. 2013. Using a Variety of n-Grams for the Detection of Different Kinds of Plagiarism. *Notebook for PAN at CLEF* 2013 (2013).

[51] Ilya Sochenkov, Denis Zubarev, Ilya Tikhomirov, Ivan Smirnov, Artem Shelmanov, Roman Suvorov, and Gennady Osipov. 2016. Exactus like: Plagiarism detection in scientific texts. In *European conference on information retrieval*. Springer, 837–840.

[52] Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*.

[53] Adaku Uchendu, Zeyu Ma, Thai Le, Rui Zhang, and Dongwon Lee. 2021. TuringBench: A Benchmark Environment for Turing Test in the Age of Neural Text Generation. In *Findings of Conf. on Empirical Methods in Natural Language*

Processing (EMNLP-Findings).
[54] K Vani and Deepa Gupta. 2017. Detection of idea plagiarism using syntax–semantic concept extractions with genetic algorithm. *Expert Systems with Applications* 73 (2017), 11–26.
[55] Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, Kathryn Funk, Rodney Kinney, Ziyang Liu, William Merrill, et al. 2020. Cord-19: The covid-19 open research dataset. *ArXiv* (2020).
[56] Debora Weber-Wulff. 2019. Plagiarism detectors are a crutch, and a problem. *Nature* 567, 7749 (2019), 435–436.
[57] Max Wolff and Stuart Wolff. 2020. Attacking neural text detectors. *arXiv preprint arXiv:2002.11768* (2020).
[58] Santiago Zanella-Béguelin, Lukas Wutschitz, Shruti Tople, Victor Rühle, Andrew Paverd, Olga Ohrimenko, Boris Köpf, and Marc Brockschmidt. 2020. Analyzing information leakage of updates to natural language models. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 363–375.
[59] Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2021. Counterfactual Memorization in Neural Language Models. *arXiv preprint arXiv:2112.12938* (2021).

A EVALUATION DATA FOR OUR PLAGIARISM DETECTION PIPELINE

We use two corpora with plagiarism labels to measure the precision and recall scores of our proposed pipeline described in Section 3.2. The first dataset (denoted as *PanDataset*) is originally introduced as a test set for the fifth international competition on plagiarism detection at PAN 2013.[24] It contains in total 3,170 source documents and 1,827 suspicious documents where 1,001 document pairs are without plagiarism and 1,001 pairs are affiliated with verbatim plagiarism. In order to automatically create document pairs for paraphrase plagiarism, the organizers applied machine-driven approaches such as randomly replacing words based on a synonym database like WordNet or back-translating sentences with existing translation models (e.g., Google Translate[25]) using source documents. This resulted in 2,002 pairs. Similarly, 1,186 summary plagiarism cases are generated by existing text summarization models.

Given that PanDataset may exhibit different characteristics from GPT-2 generated texts, we consider a subset of OpenWebText as source documents, create suspicious documents, and use the pairs as the second dataset (denoted as *GptPlagiarismDataset*). More specifically, we construct 1,000 document pairs for verbatim plagiarism by extracting 500 character-long texts within source documents and using them as suspicious documents. For paraphrase plagiarism, we randomly select 5 sentences from 1,000 source documents and employ Facebook FAIR's WMT19 transformer [39] for back translation (English->German->English). Lastly, 1,000 document pairs for summary plagiarism are created by two summarization models. We first shorten the lengths of source documents with a BERT-based extractive summarization model [36] and then transformed them into meaningful summaries using T5 transformer [45] for abstractive summarization. This enables us to create more sophisticated summaries with minimal overlapping strings.

B DETAILS ON FINE-TUNING CONFIGURATIONS

Our experimental environment is based on a Google Colab Pro+ with Tesla V100-SXM2-16GB and 55 GB of RAM. For fine-tuning, we utilize a Python package called GPT-2-simple.[26] We maintain hyperparameters that are suggested in public repositories: learning rate as 1e-4, temperature as 1.0, top-k as 40, and batch size as 1. The ratio of training and validation sets is 8:2. To prevent the model from overfitting, we stop training processes when a gap between training and test losses reaches over 20% of training loss. Table 7 illustrates their fine-tuning configurations. Fine-tuning one model for 10,000 steps approximately takes 5 hours.

C LM PERPLEXITY CALCULATION

Perplexity is defined as the exponentiation of the cross-entropy between the data and LM predictions. Given a tokenized sequence $X = (x_0, x_1, x_2 ... x_n)$, the perplexity of X can be calculated by:

$$perp(X) = exp\left\{-\frac{1}{m}\sum_{n}^{m} logf_\theta(x_n|x_{\leq n-1})\right\}$$

where $logf_\theta(x_n|x_{\leq n-1})$ is the log-likelihood of the nth token conditioned on the preceding tokens. Following the guideline provided by Huggingface,[27] we rely on a strided sliding-window technique, which entails moving the context window repeatedly so that the model has a broader context when making each prediction. Here a window size is a hyper-parameter we can adjust. To retrieve one aggregated perplexity that represents the whole instances, we first append all documents with newlines and then set the window size as 512. For an individual document perplexity calculation of the Cord-19 dataset, we reduce the window size to 50 since we do not append all documents this time, and many Cord-19 documents tend to be shorter than 512 tokens.

D STATISTICAL TESTING OF FILTERING

We perform the Pearson's chi-squared test [43] to verify the statistical significance of the observed gap between before and after filtering low-perplexity and similar documents. The test is used to determine whether there is a statistically significant difference between the expected frequencies and the observed frequencies. Here we treat plagiarism as a binary variable (no plagiarism vs. plagiarism) and count the total number of documents accordingly. For plagiarized document count, we do not distinguish plagiarism types. Table 8 shows the results of the chi-squared test. Most of our experiments except for Cord19GPT's temperature setting are found to be statistically meaningful.

E PLAGIARIZED TEXT EXAMPLES

We present several examples of verbatim, paraphrase, and idea plagiarism from both pre-trained and fine-tuned models (Table 6). For verbatim plagiarism, we identify cases where social media's app ID and its metadata are memorized, as well as an individual's writing. We also frequently find a paragraph related to journals' copyright and authors' rights as verbatim plagiarism from the model trained with academic papers. Examples associated with paraphrase plagiarism, especially those authored by GPT-2 and Cord19GPT, demonstrate models' abilities in delivering factual information in a different syntactic form without proper references. PatentGPT's plagiarism cases tend to mimic patent data by rephrasing and elaborating on the described processes created by original patent owners.

[24] https://pan.webis.de/clef13/pan13-web/text-alignment.html
[25] http://translate.google.com
[26] https://github.com/minimaxir/gpt-2-simple
[27] https://huggingface.co/docs/transformers/perplexity

Type	Machine-Written Text	Training Text
Verbatim	Unexpected Error An unexpected error occurred. [...] "facebookAppID":***,"allow_select":true,"allow_filter":true,"allow_sheetlink":true [...] (***Author: GPT-2***)	Unexpected Error An unexpected error occurred. [...] "facebookAppID":***,"allow_select":true,"allow_filter":true,"allow_sheetlink":true [...]
Verbatim	it reminded me of a feeling I've had right there on that road before. It reminded me of all the times that people have come out to support the blockade and stood together to make sure those trees stay standing. [...] (***Author: GPT-2***)	it reminded me of a feeling I've had right there on that road before. It reminded me of all the times that people have come out to support the blockade and stood together to make sure those trees stay standing. [...]
Verbatim	I, the Submitting Author has the right to grant and does grant on behalf of all authors of the Work (as defined in the below author licence), an exclusive licence and/or a non-exclusive licence for contributions from authors who are: i) UK Crown employees; ii) where BMJ has agreed a CC-BY licence shall apply, and/or iii) in accordance with the terms applicable for US Federal Government officers or employees acting as part of their official duties; [...](***Author: Cord19GPT***)	I, the Submitting Author has the right to grant and does grant on behalf of all authors of the Work (as defined in the below author licence), an exclusive licence and/or a non-exclusive licence for contributions from authors who are: i) UK Crown employees; ii) where BMJ has agreed a CC-BY licence shall apply, and/or iii) in accordance with the terms applicable for US Federal Government officers or employees acting as part of their official duties; [...]
Paraphrase	REUTERS/Kevin Lamarque U.S. President Donald Trump and First Lady Melania Trump, with their son Barron, arrive for a New Year's Eve party at his Mar-a-Lago club in Palm Beach, Florida, U.S. December 31, 2017. [...] (***Author: GPT-2***)	REUTERS/Kevin Lamarque U.S. President Donald Trump, First Lady Melania Trump and their son Barron while aboard Air Force One on their way to Florida, Mar-a-Lago in Palm Beach, Florida to spend the holiday at Trump International Golf Club Mar-a-Lago. [...]
Paraphrase	The development of natural killer cells (NK cells) is an important element in the immune system as it provides the first line of defense against diverse pathogens. (***Author: Cord19GPT***)	Natural killer (NK) cells are a type of innate lymphoid cell that plays an important role in the first line of immune defense against any viral infection, including COVID-19.
Paraphrase	A system, comprising: a sense circuit for receiving an electrical [...] and a digital compensator coupled with the sense circuit and for receiving the output value from the decision circuit and generating a compensation value in accordance with the output value [...] (***Author: PatentGPT***)	Apple's First Claim: A touch surface device, comprising: a touch-sensitive panel [...] and a sensing circuit coupled to the compensation circuit, the sensing circuit configured for receiving the compensated output signal.
Idea	A method for testing electrical connections, comprising: [...] providing an electric voltage and an electric current to an electrical contact on the test element to transfer the electrical conductivity of the line to ground; wherein the measuring is carried out with the electric current flowing from the electrical contact on the test element through the electric current to the ground; [...] (***Author: PatentGPT***)	The energy passing between elements A and B is in the form of an electric current through the earth between the two ground connections.
Idea	A control system comprising: a processor configured to execute an operation on a memory and to output an instruction stream having a plurality of executable instructions, wherein the output of the plurality of executable instructions is selectively selectable [...]; and a storage device storing a plurality of items of a control structure, each of the control structures containing executable instructions, which when executed by the processor, cause the processor to perform [...] (***Author: PatentGPT***)	The system also may comprise a memory having stored thereon instructions that, upon execution by the at least one processor, cause the system to perform [...]
Idea	Symptoms of COVID-19 infections are relatively mild, such as fever, dry cough, headache, diarrhea, dyspnoea, body ache, myalgia and sometimes headache. In some infected patients, however, the infection is more rapid and severe with fever, dyspnoea, shortness of breath, cough and other non-specific symptoms such as sore throat, runny nose, dry throat and sputum production. [...] Several factors are strongly associated with mortality in the SARS-CoV-2 outbreak. [...] and comorbidities such as hypertension, obesity, chronic lung disease, obesity and diabetes. (***Author: Cord19GPT***)	The most common symptoms of COVID-19 are headache, loss of smell, nasal congestion, cough, asthenia, myalgia, rhinorrhea, sore throat, fever, shortness of breath, nausea or vomiting, and diarrhea [2, 3] . Commonly reported comorbidities of COVID-19 are hypertension, obesity, diabetes, and cardiovascular disease [4].

Table 6: Examples of plagiarism identified in texts written by GPT-2 and its training set. Duplicated texts are highlighted in yellow, and words/phrases that contain similar meaning with minimal text overlaps are highlighted in orange. [...] indicates the texts omitted for brevity. Personally identifiable information (PII) was masked as *.**

Model Name	Training Steps	Training / Test Loss
ArXivAbstractGPT	30,000	2.48 / 2.83
Cord19GPT	44,000	2.6 / 2.68
PatentGPT	32,300	1.65 / 1.87

Table 7: Fine-tuning configurations

Model	Decoding	Plagiarized Document # (before filtering vs. after filtering)	p
Patent GPT	temp	37 vs. 20	0.002
	top-k	218 vs. 133	<0.00001
	top-p	125 vs. 86	0.007
Cord19 GPT	temp	13 vs. 5	0.059
	top-k	173 vs. 92	<0.00001
	top-p	118 vs. 60	0.00002
Cord19 GPT	temp	101 vs. 84	0.207
	top-k	550 vs. 398	<0.00001
	top-p	322 vs. 258	0.006

Table 8: Statistical results of the chi-squared test. The first result regarding Cord19GPT is for perplexity, whereas the second one is for document similarity.