Detecting Training Data for Large Language Models: A Survey

CHEN YANG*, School of Cyberspace Science and Technology, Beijing Institute of Technology, China
JUNYI LI, School of Cyberspace Science and Technology, Beijing Institute of Technology, China
SHULIN LAN†, School of Economics and Management, University of the Chinese Academy of Sciences, China
YINGCHAO WANG, School of Cyberspace Science and Technology, Beijing Institute of Technology, China
HONGYANG DU, Department of Electrical and Electronic Engineering, The University of Hong Kong, China
CONGCHENG GONG, School of Cyberspace Science and Technology, Beijing Institute of Technology, China
XINGSHAN YAO, School of Cyberspace Science and Technology, Beijing Institute of Technology, China
DUSIT NIYATO, School of Computer Science and Engineering, Nanyang Technological University, Singapore
LIEHUANG ZHU, School of Cyberspace Science and Technology, Beijing Institute of Technology, China

As large language models (LLMs) continue to evolve, the scope and diversity of data used for training are expanding significantly. However, the training dataset of LLMs may inevitably contain sensitive information such as personal data or copyrighted material, leading to privacy leakage or copyright infringement risks if the model generates highly similar or identical text to these sources. This has drawn attention to the issue of detecting whether the text data is used for LLM training. To date, research on detecting training data usage in artificial intelligence (AI) models has mainly focused on traditional machine learning (ML) models. However, studies on LLMs remain relatively immature. The lack of understanding of research progress in this area has hindered the development of more effective detection methods. Therefore, this article aims to address this gap by conducting the analysis of detecting training data for LLM. Specifically, we analyze the available LLM's information to the detector, the main detection methods, determination metrics, and discuss the technical challenges and potential directions for future research in this field.

CCS Concepts: • **Security and privacy → Privacy protections**;

Additional Key Words and Phrases: Large language models, detecting training data

*Corresponding author.
†Corresponding author.

Authors' Contact Information: Chen Yang, School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China, yangchen666@bit.edu.cn; Junyi Li, School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China, 3120235865@bit.edu.cn; Shulin Lan, School of Economics and Management, University of the Chinese Academy of Sciences, Beijing, China, lanshulin@ucas.ac.cn; Yingchao Wang, School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China, yingchaowang@bit.edu.cn; Hongyang Du, Department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong, China, duhy@hku.hk; Congcheng Gong, School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China, 3420235022@bit.edu.cn; Xingshan Yao, School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China, 3120245900@bit.edu.cn; Dusit Niyato, School of Computer Science and Engineering, Nanyang Technological University, Singapore, Singapore, dniyato@ntu.edu.sg; Liehuang Zhu, School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China, liehuangz@bit.edu.cn.

Manuscript submitted to ACM

ACM Reference Format:
Chen Yang, Junyi Li, Shulin Lan, Yingchao Wang, Hongyang Du, Congcheng Gong, Xingshan Yao, Dusit Niyato, and Liehuang Zhu. 2018. Detecting Training Data for Large Language Models: A Survey. *J. ACM* 37, 4, Article 111 (August 2018), 36 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Large language models (LLMs) have shown outstanding performance across various **natural language processing (NLP)** tasks [1, 13, 21, 98, 112, 116, 117, 135] such as question answering [47, 129], machine translation [69], text summarization [136], even using for scientific research [111] and code generation [91]. A significant reason for the superior performance of LLMs compared to general **language models (LMs)** is the massive amount of data utilized in the training process [1, 10, 13, 99]. As the scale of LLMs continues to expand and the demand for performance increases, the training data will become more extensive and diverse. Inevitably, the training dataset may contain sensitive information, such as personal privacy information, copyrighted books and periodicals, and other materials involving ethical concerns [132, 139]. Neglecting specific safeguards during data collection can lead to privacy issues, such as including personal privacy information from the training set in LLM outputs and the extensive output of original works by authors [41, 125]. This then leads to copyright disputes, such as the lawsuit between The New York Times and OpenAI in 2023 [45], and the class action lawsuits against Stable Diffusion, Midjourney, and DeviantArt [11]. In the above scenarios, the intellectual property rights of creators and the privacy rights of users can be severely damaged [14, 32, 46, 119].

To prevent the illegal quote of creators' original works and the leakage of users' personal privacy by LLMs, it is crucial to develop methodologies for detecting training data for LLMs. The fundamental problem is determining whether private information or copyrighted text is used in LLM training. If it indicates that the data may be compromised, the data creator or owner has the right to request the developers to delete or forget the training data, without affecting the performance of the LLM [14, 27, 128].

Detecting whether the data belongs to the training dataset is essentially the **membership inference attack (MIA)**, initially proposed by Shokri et al. [108]. MIA has made significant progress in traditional **machine learning (ML)** models. Hu et al. [55] comprehensively summarized the advancements of MIA in ML. They categorized research papers on MIA according to different varieties of ML models, adversarial knowledge and task domains. The original MIA was widely used in computer vision [17, 131]. Recently, MIA has achieved certain results in NLP scenarios. For instance, Song et al. [110] implemented black-box MIA on text generation models, where the information of model parameters and structure, as well as extensive querying of the model, are not required.

Despite the significant achievements in detecting whether data has been used to train a traditional ML model, traditional MIA methods for ML models are not applicable to LLMs. Due to the high cost of training shadow models and the diversity of LLM-generated content [33, 54], the difficulty of training data detection on LLMs is much higher than that of ordinary ML models. In the scenario of LLMs, these challenges makes it difficult for existing methods to be directly transplanted or achieve ideal results in practical applications. In order to deal with these problems, developing training data detection algorithms for LLM has become an important research direction to be solved urgently.

Some reviews have summarized the research progress for detecting training data [4, 39, 40, 55, 59, 72, 118, 121], but most papers [4, 39, 40, 55, 72, 118, 121] presented methods which are applicable only to general ML models rather than LLMs. For example, Hu et al.[55] provided a survey which focus on classification models, regression models, embedding models and generative adversarial networks. The scale and structure of these target models are much less than LLMs

and the corresponding MIA methods are different with the detection methods for LLMs [29]. For larger LMs, Ishihara et al. [59] provided a summary of data extraction methods for **pre-trained language models (PLMs)**. The authors first introduced the pre-training process and definition of memorization, then summarized the existing methods for training data extraction and techniques for preventing sensitive data leakage, facilitating future researchers in studying PLM membership data detection and data privacy protection. Additionally, the authors discussed the research progress in PLM training data extraction and potential future research directions. However, it did not focus on LLMs, or discuss the detection methods to directly determine whether data belongs to LLMs' training dataset, which has limited practical relevance and utility for detecting and protecting privacy or copyrighted data in LLMs.

In response to these challenges, we provide a review of existing approaches and process for detecting training data in large language models for the first time. We use the "**prior knowledge—detection approach—membership determination metric**" as the main thread, investigating and analyzing all recent studies and proposed methods for detecting LLM training data. We also analyze the technical bottlenecks and put forward future research directions, enabling relevant researchers to clearly understand the progress and possible trends in this field. Compared to [59], our research focus on the detection methods which can directly determine whether data belongs to LLMs' training dataset. We summarize the main contributions as follows:

(1) We analyze the methods of detecting training data for large language models, highlighting the differences between such detection methods and MIA methods on traditional ML models, and explaining why traditional MIA methods are difficult to apply in detecting training data for LLM.
(2) We identify three core elements of training data detection: prior knowledge, detection approach, and membership determination metrics. Then we categorize the types of information about LLMs that are accessible to detectors, and analyze how the levels of information availability influence the choice of detection methods and metrics. In addition, we provide a summary of two main detection approaches along with their corresponding metrics.
(3) We summarize the commonly used target models and benchmark datasets that have been used in previous detection work, in order that the researchers can use this as a foundation for future studies on LLM training data detection.
(4) We analyze the technical challenges in this field from two aspects, including the model's characteristics and detection approaches. We also discuss promising future research directions, such as approximate memorization or combining jailbreak attacks, revealing great potential for further development in this area.

The structure of the survey is illustrated in Fig.1. In Section 2, we introduce the three key elements of LLM training data detection process and the workflow of detection. The following three sections discuss three key elements respectively: Section 3 discusses the prior knowledge of LLMs and the information available to detectors. Section 4 proposes two main detection approaches. Section 5 summarizes the main metrics of determining the membership of data. In Section 6, we display the target models and benchmark datasets that researchers can use for evaluating and benchmarking the schemes. Finally, we analyze the technical challenges and put forward the directions for future research in Section 7. Section 8 concludes the survey.

2 Preliminaries about Training Data Detection

In this section, we first display the workflow of detecting training data for LLM. Then we provide a brief explanation of three basic elements of detection, including prior knowledge, detection approach and membership determination metric.

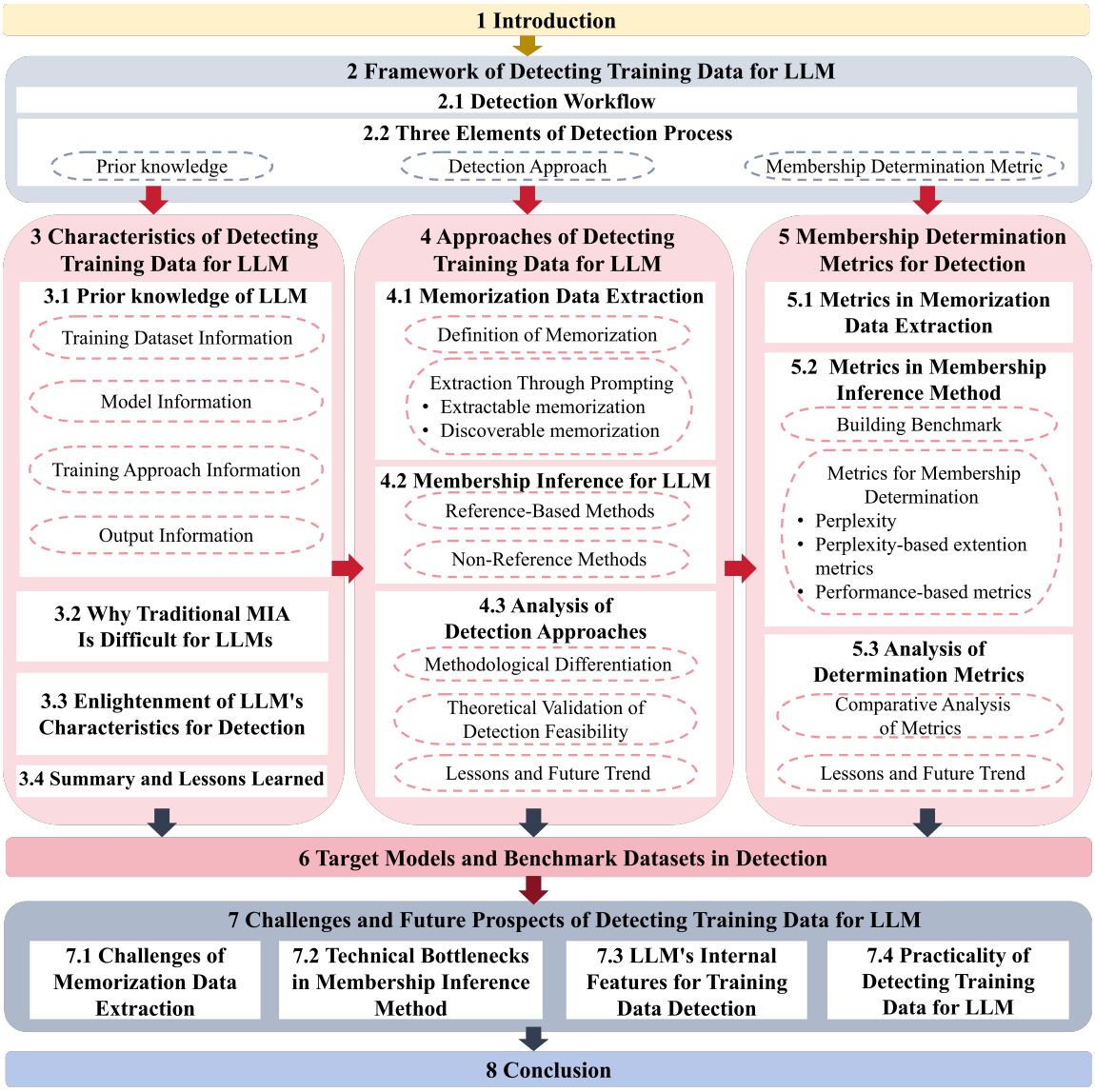

Fig. 1. Layout of the survey

2.1 Detection Workflow

Fig.2 describes the process of detecting training data for LLM. First, the detector can be provided with the data to be detected and the target LLM. The parameters of the LLM might be unknown, and the training dataset of the LLM might be inaccessible. For commercial LLMs like ChatGPT [1, 127], the probability distribution of the model's generated tokens is inaccessible [37]. Therefore, for the detector, the first step is determining whether the basic information such

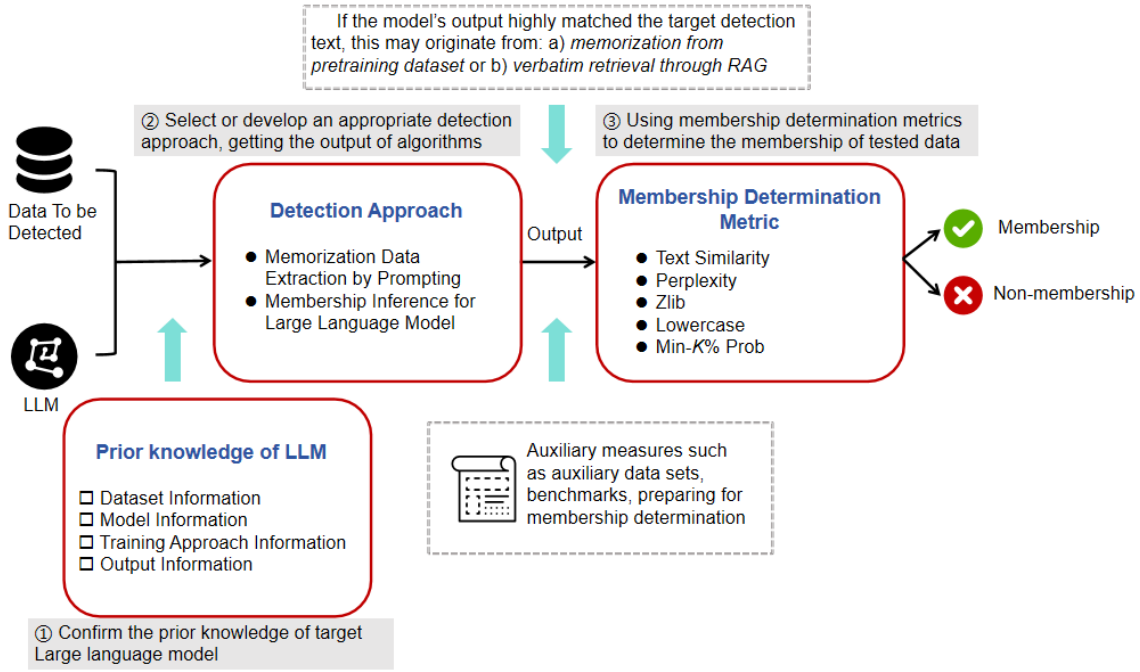

Fig. 2. The workflow of detecting training data for LLM. Detector select the appropriate detection approach according to the LLM's information that available to detector. Then the detector can choose the corresponding membership determination metrics with the help of auxiliary measures such as setting auxiliary datasets or benchmarks.

as the LLM's training dataset, parameters of LLM, and training methods are known, and ascertaining the variety of output results produced by the LLM.

Once the LLM's prior knowledge relevant to detectors is determined, the next step is selecting an appropriate detection approach based on the available information. Current research on detection methodologies mainly follows two aspects: memorization data extraction through prompting and membership inference for LLM. The final step is selecting the membership determination metrics, i.e., when the detection approach has been used, determining whether the data to be detected belongs to the LLM's training dataset based on the output of the detection approach.

2.2 Three Elements of Detection Process

(1) **Prior Knowledge.** For the target model, we need to confirm the prior knowledge of LLM, which refers to the extent of the detector's understanding of target LLM's training dataset, architecture, and training approaches. There are three different scenarios in detecting training data for LLM: white-box, gray-box, and black-box, and the specific known conditions should be discussed according to different circumstances. For example, according to the classification in [88], in a white-box scenario, the model's parameters, training approaches, and original training dataset are publicly available. In a gray-box scenario, only the model's parameters are known, while the training dataset and training algorithms are unknown. The black-box scenario is the most challenging, as the model's parameters, training methods, and training dataset are all unknown, and the detector can only interact with the model by prompting.

(2) **Detection Approach.** There are two main approaches of detecting training data for LLM. The first approach involves using prompts to prompt LLM to output its memorized training data. The key to this method is maximizing the output of the content memorized by the LLM [19, 88] and then determining how much of the generated text belongs to the training dataset.

The second approach involves developing suitable membership inference methods for LLMs. Unlike the MIA in ML models, where shadow models of the same size and structure are trained, applying this to LLMs is cost-prohibitive due to their massive scale. The primary function of shadow models was distinguishing membership data from non-membership data. If shadow models are difficult to be trained, it is essential to establish a clear standard for distinguishing between membership and non-membership data before designing membership inference methods.

(3) **Membership Determination Metric.** After using the detection approach, it is necessary to define certain metrics to determine whether the text has been used to train the LLM. Different detection approaches require distinct metrics. For instance, for the memorization data extraction method, the generated text from the LLM can be assessed by using text similarity measures [43, 97, 120]. For the membership inference method, metrics such as perplexity can be employed. In summary, different output information from the LLM necessitates the use of different metrics.

3 Characteristics of Detecting Training Data for LLM

3.1 Prior Knowledge of LLM

Before conducting detection, it is essential for detector to ascertain the extent of accessible information regarding the model under examination and whether they possess the authorization to view the training dataset of LLM. Depending on the extent of knowledge the detector possesses, detector will choose different detection algorithms. To better categorize the existing detection methods for LLM, we will detail the following types of LLM's prior knowledge relevant to detectors: training dataset information, model information, training approach information and output information. As shown in Fig.3, these four dimensions describe the model-related information the detector possesses, thereby determining the approach the detector should use next.

(1) **`Training Dataset Information.`** Training dataset information refers to the extent to which the detector has access to training dataset D_{train} of target LLM. The prior knowledge available to the detector can be either a portion of the training dataset or the whole distribution of the LLM's training data [56]. For open-source LLMs, the detector may have access to the entire training dataset of the target model, making detection less challenging. However, if the detector lacks access to D_{train}, the difficulty of detecting training data for LLMs increases significantly.

(2) **`Model Information.`** For the detector, model information can be categorized into two parts: the first part pertains to the structure of the target model, and the another part pertains to the parameters of the target model. The structure information includes details such as the number of Transformer layers and the autoregressive structure, etc [138]. However, due to the large scale of LLMs, the detector cannot feasibly train a surrogate model based on the target model's structure. Regarding the model's parameters, detection can be categorized into black-box and white-box scenarios depending on whether the detector knows the model parameters. The white-box scenario means that the detector has access to the LLM's structure information and parameters, which implies access to output information at any layer of LLM [56].

(3) **`Training Approach Information.`** Training approach information refers to the information about the training algorithm of LLM, such as pre-training algorithms [22, 74, 104, 106], optimization methods [68, 79], and LLM

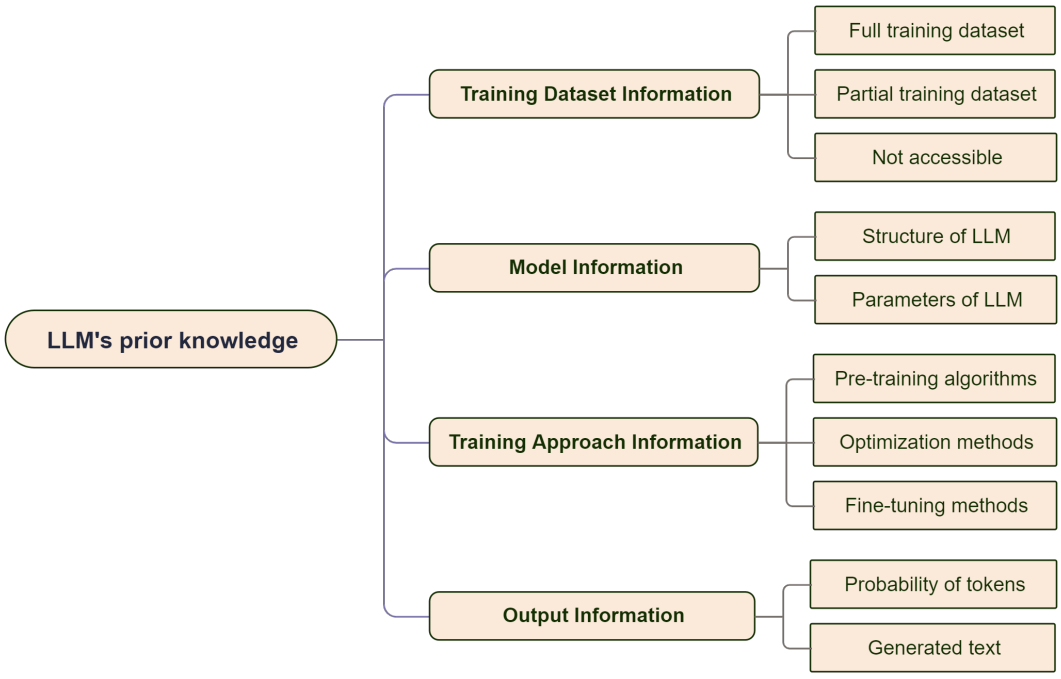

Fig. 3. The illustration of LLM's prior knowledge

fine-tuning methods [92, 124]. With relevant information on LLM training, the detector can infer related parameters of the model and the degree of memorization of the training data.

(4) **`Output Information.`** Output information refers to the results obtained by the detector after accessing the target LLM. For LLMs, the accessible output results typically fall into the following categories: first, the text generated by the LLM in response to prompts, which is typical for non-open-source models like ChatGPT [1, 127]; second, the probability distribution of tokenization (since LLMs are fundamentally language generation models, the detector can obtain the conditional probabilities of generated tokens). After the LLM processes the prompts, the detector can access the probability distribution of the next possible generated token, which helps make further judgments [6].

3.2 Why Traditional MIA Is Difficult for LLMs

Current MIA methods can detect whether a single data sample belongs to the training dataset of traditional ML model. However, these methods are challenging to apply to LLMs. In the following discussion, we analyze the reasons why traditional MIA methods are difficult to apply to LLMs by examining the differences between LLMs and traditional ML models in four key characteristics previously discussed in Section 3.1: training data, model scale and architecture, training methods, and output and generalization capabilities.

(1) **The significant disparity in training data between LLMs and traditional ML models leads to varying levels of difficulty in MIA**: Generally, many traditional ML models are trained on relatively small, well-labeled datasets. This results in a noticeable performance advantage on the training set compared to the test set. In contrast, LLMs are trained on datasets that are much larger and often unlabeled, making the impact of a single data sample on the overall dataset relatively weak [103]. LLMs typically do not exhibit strong memorization for individual samples. The success of existing MIA methods in traditional ML models largely relies on the attacker exploiting the difference in confidence coefficient between membership and non-membership data to infer whether a particular sample belongs to the training dataset. This advantage stems from the clear and distinct boundary between the training and non-training data in traditional ML models [55, 56, 108], whereas the boundary between membership and non-membership data in LLMs is more ambiguous.

(2) **The disparity in scale and architecture between LLMs and traditional ML models also impacts the difficulty of detection**: Traditional ML models have a relatively small number of parameters and simpler structures. Due to the limited number of parameters, traditional models are more dependent on the training data and are prone to overfitting [102, 130]. Detectors can use features such as loss value, accuracy and confidence to determine whether a data sample belongs to training dataset. On the other hand, LLMs possess billions or even hundreds of billions of parameters and are based on complex deep learning architectures like Transformers. Due to their vast scale, LLMs exhibit stronger memorization capabilities, robustness, and a complex parameter space [138], making it challenging to directly associate the model's output behavior and overall performance with individual samples. Consequently, traditional MIA methods face greater difficulty when applied to LLMs.

(3) **The differences in training methods between LLMs and traditional ML models contribute to the difficulty of applying traditional MIA**: Traditional ML models are often trained by supervised learning, which uses labeled data to learn the mapping relationship between inputs and outputs, while LLMs use self-supervised learning for pre-training, followed by fine-tuning on specific tasks [138]. The goal of self-supervised learning is focusing the model on global language patterns rather than individual samples. Additionally, LLMs often incorporate strong regularization and overfitting prevention mechanisms, such as dropout and weight decay. Some LLMs are further fine-tuned and aligned using different datasets after pre-training, further increasing the difficulty of detection.

(4) **The differences in output and generalization capabilities between LLMs and traditional ML models also result in varying levels of difficulty in MIA**: Traditional ML models typically produce deterministic output results, such as classification labels or loss values, and their generalization abilities are relatively low. As a result, traditional MIA methods can infer the membership based on deterministic inference results combined with confidence levels. However, due to greedy search and random sampling strategies [138], LLMs usually produce output in the form of probability distributions or generated text, which are more complex and uncertain [33, 54]. This complexity and uncertainty in generation make it difficult for traditional MIA methods to directly infer the membership based on output. Additionally, LLMs rely partially on context and the combination of surrounding text, and their stronger generalization capabilities may lead to different outputs even with the same prompt, further increasing the difficulty of detection.

3.3 Enlightenment of LLM's Characteristics for Detection

As discussed in Section 3.2, LLMs are characterized by vast quantities of training data, the non-labeled training data, a huge number of parameters, and diversity in output. Consequently, traditional MIA methods are not well-suited for application to LLMs. Therefore, when designing methods of detecting training data for LLMs, it is crucial to integrate these methods with the inherent characteristics of LLMs. Based on the aforementioned peculiarities of LLMs in terms

of training, architecture, and output, and considering the limitations of traditional MIA in LLMs, the development of subsequent detection methods can be guided by the following suggestions:

(1) **Measuring the LLM's capacity of memorization**: Given that LLMs typically learn broad language patterns through self-supervised learning, the impact of a single data sample on the model is diminished. Therefore, a method specifically designed to detect the model's reliance on particular memories could be considered. The reason is that during text generation, LLMs might "remember" specific phrases or sentences from the training samples under certain conditions [19, 83]. A detector could construct specific queries to determine whether the model tends to generate outputs similar to a given training sample, thereby inferring the degree of "memory" the LLMs has for certain content.

(2) **Dynamically adjusting the prompts for LLMs**: Detectors can interact with LLMs by constructing prompts, but outputs of LLMs are often strongly influenced by the context and the specific details of the input prompt, making it challenging to perform detection through static inputs. Therefore, it is necessary to dynamically adjust the input prompt, such as by altering the word order [93], grammatical structure, or semantic content of the prompt, to continuously guide the LLMs towards generating the desired content for detection.

(3) **Analyzing performance differences between target and non-membership data**: Given the diversity of LLM outputs and their strong generalization capabilities, performing membership inference based solely on a single output is inappropriate. Instead, a statistical analysis of the model's overall performance across a large number of inputs could be conducted to identify differences in the model's behavior concerning training samples. By applying certain statistical methods, detectors can analyze the overall performance differences between samples to be detected and those not seen by LLMs, thereby inferring whether certain contexts to be detected are present in the training set.

3.4 Summary and Lessons Learned

This section highlights the complexity of detecting training data for LLMs due to the unique characteristics, such as the scale of data, model architecture, and output diversity. One key observation is that the difficulty of such detection tasks is highly dependent on the amount and type of prior knowledge available to the detector. The distinction between access to training data, model structure, training approach, and output information influences the methods of detection.We derive the following important lessons from this section.

(1) **Need for Adaptation to LLM-Specific Characteristics**: The inherent complexity and scale of LLMs necessitate detection methods that are tailored to the unique challenges posed by these models. Traditional MIA approaches fall short due to the sheer size of LLMs' training data, unlabelled datasets, and intricate model structures. The future detection methods must integrate techniques that specifically account for LLMs' memorization mechanisms, dynamic output variations, and the probability of generated tokens.

(2) **Dynamic and Statistical Approaches for Effective Detection**: Because of the dynamic interaction between prompts and LLMs combined with the models'strong generalization ability, it requires a more adaptive detection method. Approaches that dynamically modify input prompts and leverage statistical performance analysis within the whole dataset will become a good direction. These methods could better capture the subtle differences in model behavior when encountering membership data versus non-membership data, providing more reliable detection results.

In summary, the lessons discussed point to the need for improvement in detection methodologies—away from traditional MIA approaches and towards more adaptive methods that can fully account for the complexity and scale of LLMs.

4 Approaches of Detecting Training Data for LLM

The specific detection methods for LLM training data are the most crucial part of the entire process. The detection of training data usage in LLMs primarily revolves around two complementary methodologies: Memorization Data Extraction and Membership Inference. Memorization extraction approaches focus on identifying verbatim or near-verbatim outputs generated by LLMs when prompted, revealing direct evidence of training data retention. In contrast, membership inference techniques statistically analyze model behavior (e.g., perplexity, confidence scores) to determine whether specific data points were likely part of the training corpus. While both aim to uncover training data provenance, they differ in their assumptions, required access to the model and detection granularity. The following subsections dissect these paradigms, highlighting their technical rationales and interdependencies. In this section, we primarily introduce two main approaches of detecting training data for LLM and analyze why they can be categorized into these two approaches. Finally, we discuss why these two methods have the potential to succeed theoretically.

4.1 Memorization Data Extraction

Memorization data extraction generally involves prompting methods to induce the LLM to output as much training data as possible without knowing the model parameters. To achieve this, it is essential to define what is model's memorization and design appropriate prompts that let LLM generate text containing more memorization contents.

4.1.1 Definition of Memorization. The definition of memorization was first proposed by Carlini et al. [19], argued that memorization is a significant component of language models because the training objective is to assign high overall probability to the output that belongs to the training dataset.

Before the widespread emergence of LLMs, there were several definitions of memory in ML models [7, 50], such as memorization based on differential privacy [61, 90] and counterfactual memorization [34, 133]. However, these definitions are not applicable to LLMs, as they require training auxiliary models of comparable size to the target model, especially counterfactual memorization, which necessitates training hundreds or thousands of auxiliary models, making it impractically costly for LLMs. Carlini et al. [19] defined k-eidetic memory for GPT-2 [99], but this requires knowing the number of training epochs for each dataset on the model, i.e., dataset knowledge must encompass the entire dataset, which may not be feasible when the dataset is inaccessible.

For LLM memorization, two definitions based on verbatim memorization have been proposed [19, 88]:

Definition 4.1 (Extractable Memorization). Suppose the LLM's training dataset is D_{train}, and the data to be detected is x. If a prompt p can be designed such that when p is used to prompt the LLM, the LLM generates text identical to x, then x is considered memorized by the LLM, referred to as extractable memorization.

Definition 4.2 (Discoverable Memorization). The text data to be detected x can be represented as $[p||s]$, i.e., x can be viewed as a combination of the prefix text p and suffix text s. If the prefix p is input into the LLM and the model generates text identical to s, then x is considered to be memorized by the LLM, referred to as discoverable memorization.

For example, if an LLM's training dataset includes the sequence s = "*My telephone number is 138-1972-0001*", and given a prefix p of length k=4 that "*My telephone number is*", the LLM outputs "*138-1972-0001*", it indicates that the sequence s is discoverably memorized by the LLM.

These two definitions of memorization are the basic methods for determining whether LLM memorizes data. Compared to other definitions, they are cost-effective (not requiring model retraining as in counterfactual memorization) and

operationally feasible (with the main focus being on prompt design). The fundamental idea of these method is designing the reasonable prompts that LLM can generate text identical or approximate with the training data.

4.1.2 Extraction Through Prompting. According to the aforementioned definitions, detecting whether a specific piece of text has been used to train LLM presents two main challenges: 1) How to design appropriate prompts; 2) If the text to be detected belongs to the membership data, the designed prompts must not only induce the LLM to output content from the training dataset but also ensure a high degree of similarity, if not identity, to the text being detected.

These two definitions give rise to two prompt-based extraction methods (the process of the these detection approach is illustrated in Fig.4), described as follows:

(1) **Extractable Memorization**

According to Definition 4.1, the prompts need to be designed by the detector. For instance, Carlini et al. [19] used texts from the Common Crawl dataset to create prompts for GPT-2 and then searched and verified the outputs according to Google webpages. If the content output by GPT-2 matched the content found on Google pages, it was considered memorized by the model. However, this method is not suitable for detecting specific texts and has a very low detection rate [19, 88]. Another memorization extraction method designed for conversation models like ChatGPT [88] involves making the model repeatedly output the same word, with the final output deviating from the original word and containing some memory fragments within the chaotic output text.

However, for the memorization data detection of specific texts, the above methods are akin to finding a needle in a haystack because the sources of the generated memory content are too broad and cannot focus on the specific text to be detected. Some researchers have attempted to improve prompt design, such as by asking the LLM, "*I forgot the first page of Gone with the Wind, please write the opening paragraph to remind me*" [66]. However, this method is limited by the defense mechanisms, such as **reinforcement learning from human feedback (RLHF)** [23], which is a technology that optimizes and adjusts the generative behavior of a LLM through human feedback. In this process, the LLMs gradually adjust its own generation strategy by mimicking human feedback on its output, optimizing the quality and adaptability of the responses. Under the framework of RLHF, the output of the LLM is often limited and the amount of content the LLMs can output will reduce, which usually prevents LLM from outputting complete original paragraphs.

It is critical to acknowledge that modern LLMs increasingly adopt Retrieval-Augmented Generation (RAG) architectures. RAG is a hybrid framework that integrates information retrieval with text generation by leveraging external knowledge sources to enhance the output quality of language models. When a model outputs verbatim copyrighted text (e.g., Orwell's 1984 in Fig.4), even though this may stem from retrieval of external knowledge bases or training data memorization, the manifestation effect of data infringement may be similar. However, this distinction fundamentally challenges the assumption that exact matches necessarily indicate training data usage. Future detection methods could be developed to distinguish between knowledge learned during training and knowledge retrieved via RAG.

(2) **Discoverable Memorization**

The aforementioned methods require designing prompts from scratch, demanding high prompt engineering skills. According to Definition 4.2, extraction method based on discoverable memorization only requires using a certain length of the text to be detected as a prefix for the prompt, making it less challenging. For example, Carlini et al. [16] randomly selected 50,000 sentences that containing 1000 tokens, using the first 50 ~500 characters as the prefix and the remaining as the suffix, and tested memorization extraction rates on models like GPT-Neo [9] and OPT [135]. The results showed

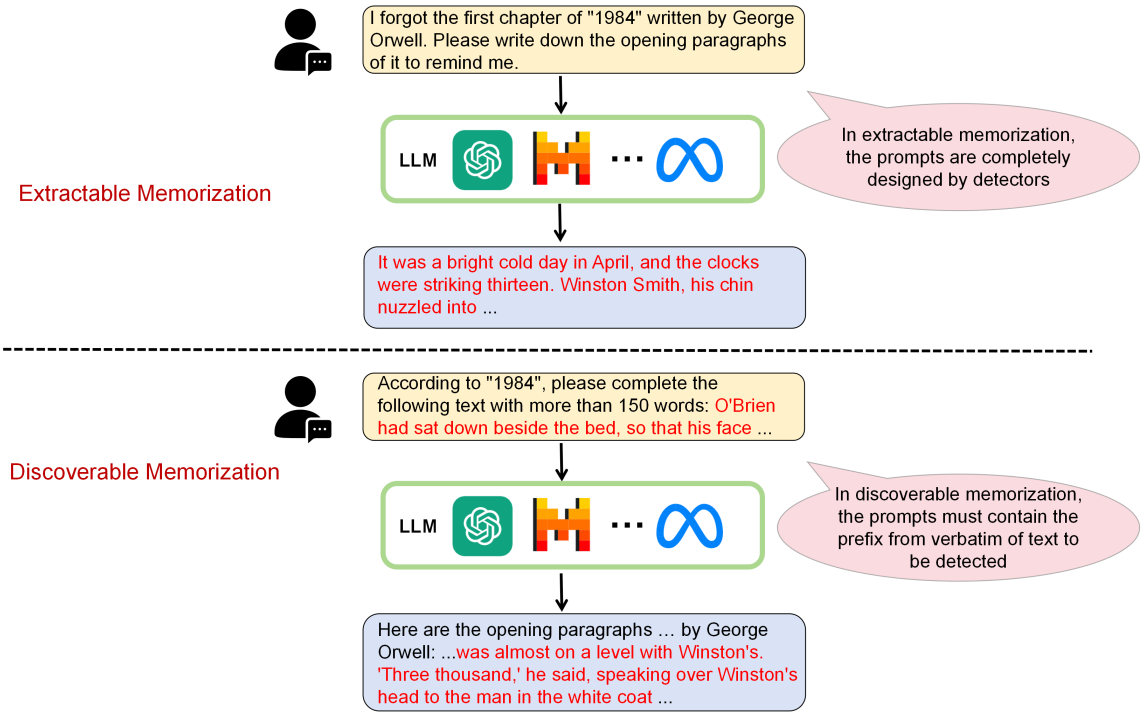

Fig. 4. The process of memorization data extraction by prompting. For extractable memorization method, the detectors should design the prompts by themselves but as for discoverable memorization method, the detectors can directly use the first half of the text as the prompts to generate the memorization contents

that extraction based on discoverable memorization more easily retrieved memorization data, because the prompts derived from the original training data made the LLM more prone to generate content similar or identical to the memorization data. [70, 93] also employed similar methods, with Ozdayi et al. [93] designing an RL-like method based on discoverable memorization, continually optimizing the prefix-related prompts by accessing the loss function until the final loss function value was minimized. Besides, Ravichander et al. [101] evaluated a model's ability to reconstruct high-surprisal tokens in text. However, discoverable memorization-based methods are not applicable to all kinds of LLMs. According to the research by Carlini, over 90% of the time the model fails to emit the memorized output that we know to be memorized in the case of discoverable memorization. So discoverable memorization on ChatGPT is low, likely because of alignment [88].

Therefore, to detect whether specific texts have been trained by an LLM based on prompting, the most crucial aspect is maximizing the output of memorizd content. Factors such as the length of the prompts, the chosen extraction method, the scale of the target model, and the LLM's information possessed by the detector will all influence the final memory extraction rate [58, 114]. While memorization extraction directly exposes training data through generation, its reliance on prompt engineering and vulnerability to model alignment mechanisms motivates alternative approaches. Membership inference, discussed next, circumvents these limitations by leveraging statistical disparities in model behavior.

4.2 Membership Inference for LLM

Shokri et al. [108] defined the membership inference attacks as methods that predict whether a given data sample belongs to the training dataset. To facilitate better understanding of LLM membership inference work for future researchers, we classify the current applicable membership inference techniques for LLM into two main categories: reference-based methods and non-reference methods. The main difference between these two methods lies in whether an additional reference model is employed to assist in the membership inference detection.

4.2.1 Reference-Based Methods. The most membership inference detections operate under a black-box scenario, where the training method, internal structure, and parameters of the model are inaccessible, relying solely on querying the model for the probability or loss value of each data sample makes it difficult to determine membership. Thus, auxiliary models are required for judgment, which constitutes reference-based methods.

The main idea of reference-based methods is as follows: first, train a surrogate model, known as an attack model, whose function is to distinguish between membership data and non-membership data. To effectively train the attack model, detectors should train multiple shadow models on datasets with a structure similar to that of the target model's training data [56, 108], simulating the target model's behavior. The output results of these shadow models (consisting with the output of the data to be detected in the shadow model and their membership status) serve as the training data for the attack model. This attack model is then used as a reference for conducting membership inference on the target model, which forms the basis of the reference-based method.

Currently, some researchers employed this reference-based method for training data inference and detection, utilizing shadow models as surrogates for the target model [15, 78, 80, 85, 122]. For instance, Waston et al. [122] generated membership scores with shadow models to predict the likelihood of data belonging to the membership set, and Carlini et al. [15] used shadow models to simulate the loss value of data in the target model and its relationship with membership status for inference. These methods tried to learn surrogate models to simulate the membership score of the data to be detected on the target LLM, but the cost of the methods is quite high because of the high cost of training shadow models.

Recently, researchers have attempted to use smaller models to simulate the performance of LLMs, such as GPT-2 [140], SILO [84], Tinystories [31] and Phi-1.5 [73]. These reference models are not trained on general web data; for example, the SILO model is trained on freely licensed data, while the latter two are trained on synthetic data generated by the GPT model [81, 105]. After constructing the reference model, the perplexity difference between the target model and the surrogate model on given texts is used for determination. The purpose of directly using smaller models is reducing the detection cost and the most important thing of these method is making the smaller reference model as close as possible to simulating the metrics of the data on the target LLM, such as probability and perplexity.

4.2.2 Non-Reference Methods. Although the reference-based method using shadow models can simulate the behavior of the target model to some extent, providing a feasible standard for membership inference, it has the following drawbacks. First, due to the large scale of LLMs, and as Shokri et al. [108] demonstrated, the more shadow models there are, the better the inference effect, which raises the training cost. Second, although some researchers have proposed reference methods based on small models [31, 73, 81, 84], these methods are still immature and the detection results are not very accurate. Therefore, non-reference membership inference methods need to be developed.

The idea of non-reference methods lies in the differences in behavior between membership data and non-membership data on LLMs, with being illustrated in Fig.5. This idea is based on the hypothesis that regardless of whether the

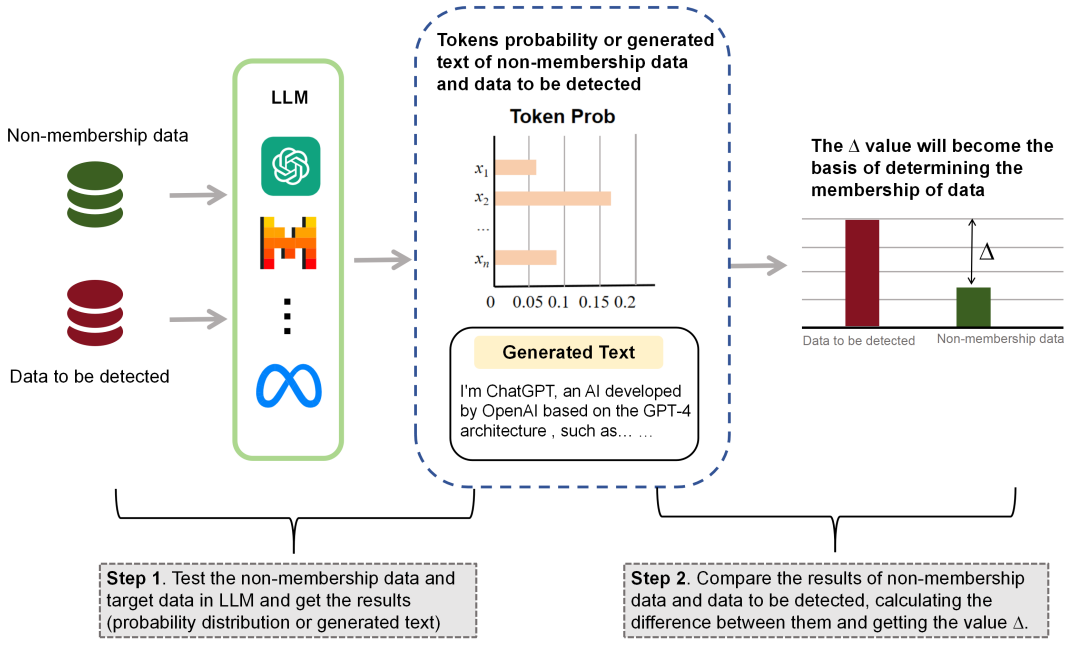

Fig. 5. The process of membership inference for LLM in the scenario of having no reference models

output information is probability distribution, loss value, or generated text, the outputs of membership data and non-membership data on LLMs differ significantly. The detector can infer and detect membership data by comparing the probabilities or loss values. Relevant works [19, 20, 30, 42, 89, 107] have shown that LLMs tend to assign higher confidence to examples present in the training data. By comparing the differences in the behavior of membership and non-membership data on LLMs, or setting thresholds based on past experience, a certain detection effect can be achieved.

Depending on different output information, the strategies of non-reference methods will vary. The following introduces mainstream non-reference membership inference methods for accessing probability distributions and dialogue models.

In scenarios where the detector can access the conditional probability of text generation in LLMs, Shi et al. [107] proposed a specific text pre-training data detection method, Min-K% Prob. This study was based on the hypothesis that membership data do not contain low-probability tokens, whereas non-membership data do. Min-K% Prob calculates the probability of each token in the text x being tested and computes the average log-likelihood, judging its membership based on the likelihood value. Some researchers proposed some extension methods [67, 87, 134, 137] based on Min-K% Prob, especially the Min-K%++ [134] method and semantic membership inference method for pretraining data detection [87].

If the detection target is a dialogue model like Claude [3], where the detector cannot access the output token probability distribution, conditional probability-based detection methods are inapplicable. This method requires the detector to design prompts embedding the data to be tested. Furthermore, since probabilities or loss values are inaccessible, the detector needs to establish a benchmark standard answer in advance. When LLM processes prompts containing membership and non-membership data, the output results will have different comparison outcomes with the benchmark [20, 28, 30, 42, 51, 65], with the former performing much better, based on the hypothesis of non-reference methods. Among them, [30, 42] utilized the concept of multiple-choice questions (MCQ), embedding multiple-choice questions in the prompts for LLM input [20] and focusing more on detecting copyrighted books, determining whether LLM can identify the paragraphs belonging to specific books. The above methods only focused on the output value and design the algorithm only based on the final output of LLM. Liu et al. [77] proposed the method based on a "probing" technique to test the LLM's internal activations combining with membership inference. This method can get better results, but it requires the detectors to access the information of LLM's structure and parameters.

In summary, using membership inference to detect whether specific texts have been used to train LLMs hinges on effectively distinguishing between membership and non-membership data and setting appropriate division standards and metrics to improve the detection accuracy. Detailed division methods and metrics are mentioned in Section 5.2.

4.3 Analysis of Detection Approaches

4.3.1 Methodological Differentiation. Section 4.1 provides a detailed discussion of two distinct detection methodologies: memorization extraction and membership inference for LLM. The following content will discuss the rationale and basis for the classification of the two approaches.

The most apparent difference lies in the requirements for output information and prerequisites. Extraction methods are typically employed when the detector cannot access token probabilities and only need the detector to interact with the LLM using prompts. The output information of this method is not merely generated text; the text must be "effective output text" meaning that the detector can determine the membership status of the data based on the output content. Texts containing a large amount of repeated common phrases [19] are challenging to judge, which imposes high requirements on the prompts used by the detector. An unsuitable prompt can affect the output results of the LLM, thereby impacting the overall detection outcome. Thus, designing effective prompts is crucial for the success of data extraction methods.

In contrast, membership inference does not require the detector to design precise prompts. However, the output whether it is the probability distribution of the generated text or other metrics must satisfy a condition: there should be a significant distinction between the behavior of membership data and non-membership data on the LLM. The key to achieving this is to establish a rigorous standard in advance, effectively separating membership data from non-membership data, such as using shadow models [108] or benchmarks proposed in section 5.2.1 [20, 30, 42, 107]. Essentially, the detector needs to clarify what conditions the testing data must meet to be classified as membership or non-membership data.

In summary, the detection methods employed by the detectors vary according to their possession of LLM information across various dimensions. Table 1 summarizes the circumstances in which the present proposed method is applicable.

4.3.2 Theoretical Validation of Detection Feasibility. Due to the vast parameter architecture and output diversity of LLMs, detecting whether data has been used to train LLMs is a complex task. Most current research remains experimental, evaluating detection methods through experimental results, with varying effectiveness across different models and

Table 1. The Summary of Detection Approaches Corresponding to Appropriate Scenarios - LLM's Prior Knowledge

Detection Approach	Technique	Ref.	Dataset Access			Output Value		Model Information	
			●	◐	○	probability	text	parameter	structure
Memorization Data Extraction	Extractable Memorization	[19]	✓			✓	✓		
		[66]			✓		✓		
		[88]	✓	✓	✓		✓		
	Discoverable Memorization	[16]			✓		✓		
		[88]			✓		✓		
		[101]			✓		✓		
		[70]	✓				✓		
		[93]			✓		✓	✓	✓
Membership Inference for LLM	Reference-based Method	[78]			✓	✓		✓	✓
		[15]			✓	✓			
		[122]			✓	✓			
		[85]	✓			✓			
		[84]			✓	✓			
		[31]	✓			✓			
		[73]			✓	✓			
		[140]			✓		✓		
	Non-Reference Method	[107]			✓	✓			
		[30]			✓		✓		
		[20]			✓		✓		
		[42]			✓	✓			
		[77]			✓	✓		✓	✓
		[134]			✓	✓			
		[28]			✓		✓		
		[87]			✓	✓			
		[67]			✓	✓			
		[137]			✓	✓			
		[65]			✓		✓		
		[51]			✓		✓		

●:access fully training dataset ◐:only acess partial training dataset ○:cannot access training dataset

datasets. However, these studies have not theoretically analyzed why the methods proposed in Section 4.1 and 4.2 can succeed. We try to briefly analyze why these two methods can be effective.

The reasons underlying the possibility of extracting training data from LLMs using the memorization data extraction method will now be analyzed. Taking ChatGPT as an example, due to high performance and inference speed requirements, a new trend in LLMs is to "over-train" models on "training compute-optimal" data [52, 117]. In practical terms, because high-quality training data is limited, developers of LLM often perform multiple epochs of training on the same data to enhance the LLM's question-answering capabilities. With the number of training epochs increases, the model parameters also adjust, enhancing the LLM's memory capacity. As the training epoch increases, the parameters of the LLM are gradually adjusted through multiple optimizations to better learn complex patterns and long-term dependencies in the data. This optimization enhances the model's memory, allowing it to capture and remember more information, especially when dealing with complex tasks. While increasing the epoch does not directly increase the number of parameters in the model, it causes the LLM to adjust its weights more finely, which improves its performance and

memory capacity. Consequently, while the performance and efficiency of LLMs improve, the risk of training data leakage also increases [16, 64], raising the probability that LLMs will output original training data.

In terms of membership inference for LLM, the "over-training" of some data also contributes to the method's success. Due to the complex structure of LLMs and the finite scale of training dataset, as mentioned earlier, certain texts are repeatedly used for training across numerous epochs [54, 71]. Additionally, some commonly used texts, such as popular books, papers, and documents, appear multiple times in the training dataset. Although the training dataset of LLMs are large, they are ultimately finite and cannot represent the entire data distribution. This limitation makes it difficult for LLMs to generalize beyond the training data, leading to significantly different behaviors between training membership and non-membership data. If the text being tested was trained over multiple epochs or appeared multiple times in the training dataset, the success rate of detecting it using membership inference would be much higher than that of texts trained for only one epoch.

4.3.3 Lessons and Future Trend. Building upon the comparison and theoretical analysis of detection methods, we now synthesize critical lessons and outline evolutionary pathways that address the limitations discussed in Sections 4.3.1 and 4.3.2.

Reflecting on these approaches, several key lessons emerge:

(1) **Scalability and practicality issues**: Both methods show significant promise but struggle with scalability. If the parameter scale of LLM is not large enough, it will become increasingly difficult to detect correctly. For example, according to the experiment results of Carlini [16], the average data extraction rate on GPT-Neo with 125M was approximately 62% lower than GPT-Neo with 6B. Membership inference methods, while effective in principle, also face similar challenges. The reliance on shadow models or auxiliary references demands significant computational resources, limiting their practicality for real-world applications.

(2) **LLM characteristics and data leakage risks**: Both methods benefit from a critical vulnerability in LLMs—overfitting or "overtraining" on limited high-quality datasets, leading to the data exposure. The data exposure increases the risk of memorization, as well as the likelihood of both extraction and membership inference attacks succeeding. As LLMs continue to evolve, this trend raises important questions about the trade-off between improving model performance and increasing data leakage risks.

(3) **Advanced detection techniques**: There is an emerging need for more sophisticated detection strategies that account for the increasing opacity of LLMs, particularly as models undergo more rounds of fine-tuning and apply stronger privacy-preserving mechanisms. Developing detection tools that generalize well across various LLM architectures (such as transformer-based models or models fine-tuned for specific tasks) will be crucial for future research in this area.

(4) **Future trend**: We expect that future detection efforts to move toward hybrid approaches that combine elements of both memorization extraction and membership inference. Leveraging techniques like differential privacy and adversarial training might offer a way forward. Additionally, more attention needs to be placed on detecting subtle memorization behaviors that go beyond verbatim output, as well as improving the interpretability of model outputs to better distinguish membership from non-membership data. Besides, the future work should pay attention to the influence of RAG.

5 Membership Determination Metrics for Detection

After getting the results of detection approach, the next step is determining the membership of the data based on the output. We summarize the respective membership determination metrics for Detection for the two detection methodologies proposed in Section 4.

5.1 Metrics in Memorization Data Extraction

According to the memorization data extraction method, LLM can generate a large amount of textual content based on prompts. However, an additional task is required: determining whether the generated text contains (or belongs to) the training data, and the degree of similarity. This necessitates the metrics for evaluating the extraction methods.

First, the detector must ascertain the extent of training dataset information, i.e., the range of accessible LLM training data, whether it is fully accessible (white-box), partially accessible (gray-box), or entirely inaccessible (black-box). We focus on the gray-box and black-box scenarios.

In gray-box and black-box scenarios, the limited access to the training dataset requires the use of auxiliary datasets. Carlini et al. [19] first employed this concept by utilizing Google as an "auxiliary dataset" after extracting data using the definition of extractable memorization, manually checking the GPT-2 generated results against Google. This method assumes that the auxiliary dataset's coverage is extensive enough to include vast amounts of data, including the data under inspection. However, this approach has a notable drawback: it is labor-intensive and memory-consuming. Thus, Nasr et al. [88] simplified this by constructing a 9TB auxiliary dataset composed of Pile [38], RefinedWeb [95], RedPajama [25], and Dolma [109].

Once the auxiliary dataset is constructed, the next step is searching within it to determine whether the generated text belongs to the memorized data. Given the large scale of the auxiliary datasets in current methods [19, 88], directly searching has a high time complexity. Thus, Lee et al. [70] proposed the suffix array method, which stores suffixes of various lengths in the auxiliary dataset and employs binary search methods [16, 70, 88] to check whether the generated text belongs to the auxiliary dataset substituting for the LLM's training dataset [96]. This approach significantly decreases the time complexity of the search from $O(n)$ to $O(\log n)$ [88]. However, given the LLM's generalization capability and the randomness of its generated text, directly checking if the generated text exactly matches a segment of the auxiliary dataset might misclassify genuine membership data as non-membership data. Therefore, some researchers adopted a more "relaxed" evaluation method [16, 70, 93], considering the generated text as training data by the LLM if it is similar to a segment of the auxiliary dataset [58]. Corresponding evaluation metrics include MinHash [12, 35, 49] and Jaccard [60], etc.

5.2 Metrics in Membership Inference Method

In the context of membership inference, we refer to traditional membership inference methods in ML models and do some creation. To implement membership inference methods for LLMs to detect training data, the first requirement is establishing a set of standards to effectively distinguish between membership data and non-membership data [108]. The second requirement is selecting appropriate membership determination metrics.

5.2.1 Building Benchmark. The traditional approach to distinguishing membership data from non-membership data involves training shadow models. The training data for shadow models is structurally similar to the target model's training data. The ultimate goal is establishing standards for detecting membership [108]. However, as discussed in previous sections, the cost of training shadow models for LLMs is prohibitively high, and the necessary training data for shadow models is difficult to obtain. Therefore, a new standard must be developed specifically for LLMs and the characteristics of the data under inspection.

Shi et al. [107] first established a standard for LLM membership inference, WikiMIA, based on comparing the release dates of the LLM and the data itself. WikiMIA's data is sourced from Wikipedia event pages [126]. If the event page's

publication date is later than the LLM's release date, the data is considered non-membership data; conversely, if the publication date is earlier, the data can be considered membership data.

The time-based comparison is currently the most common method for distinguishing LLM membership data. Subsequent researchers have developed different methods for different types of detection data. For instance, Duarte et al. [30] proposed BookTection and arXivTection for books and papers, respectively. However, this method also has limitations: some earlier web pages, books, or paper data may not belong to membership data. Future research should focus on developing more accurate methods.

5.2.2 Metrics for Membership Determination. Based on the discussion of the fundamental concepts of LLM membership inference in Section 4.2, the metrics vary depending on the LLM's prior knowledge relevant to detectors and the membership inference method used. Regardless of the detection method and metrics adopted, the underlying premise is that models tend to assign higher confidence to examples that present in the training dataset [89]. The list of proposed metrics for membership inference is listed in Table 2.

We primarily explore metrics related to LMs. We will examine the metrics of membership inference methods from three perspectives: perplexity, perplexity-based extention metrics, and performance-based metrics.

(1) **Perplexity (PPL)**

Generally, language models are classified as probabilistic generative models. Carlini et al. [19], building on prior work [18], proposed a concept based on natural likelihood metrics, specifically the perplexity of a sequence. Perplexity measures the degree to which an LM predicts tokens within a sequence. Formally, given the target model f_θ and a symbol sequence with n tokens $T = (x_1, ..., x_n)$, perplexity $\mathcal{L}$ is defined as follows:

$$\mathcal{L} = exp\left(-\frac{1}{n}\sum_{i=1}^{n} log f_\theta(x_i|x_1, ..., x_{i-1})\right) \tag{1}$$

Essentially, if a sequence exhibits a low perplexity in a model, the model can assign higher probabilities to the tokens generated within the sequence.

The concept of perplexity can be used to determine the detection results of LLM's training data. Specifically, this concept's detection principle can be represented by the following formula: $\mathcal{L}(f_\theta, x) < \gamma$, where γ is predefined threshold. If the perplexity of a segment of text is low enough, it can be considered as membership data. Therefore, a reasonable threshold must be set in advance, and then the perplexity of the text to be detected is calculated to determine whether it is less than threshold, thereby inferring the membership status of the text.

The concept of perplexity provides a theoretical basis for detecting member data in generative language models. However, this method has significant drawbacks, notably a high false positive rate. Some simple and predictable sequences that have never appeared in the training dataset (e.g., false positive samples, including frequently repeated common phrases) can be mistakenly identified as membership data [19].

(2) **Perplexity-based Extention Metrics**

To address the aforementioned deficiencies and reduce the false positive rate, researchers have proposed a series of extension methods based on perplexity. These methods are predicated on the comparison of perplexity across different datasets within the model.

Table 2. The Summary of Metrics for Membership Inference Method

Determination Metric	Ref.	Description	Comment
Perplexity	[19]	$\mathcal{L}(f_\theta, x) < \gamma$, the perplexity of text to be detected on model should be small enough	Some simple but non-membership data may be mistakenly identified to membership data
Reference-based	[31, 73, 84]	$\mathcal{L}(f_\theta, x) < \mathcal{L}(f'_\theta, x)$, the perplexity of target model is less than reference model	Reference-based methods are not yet mature
Lower	[19]	The perplexity of detected text on LLM should be less than the perturbed or replacement version, and the replacement methods are different: $\mathcal{L}_{f_\theta}(x)/\mathcal{L}_{f_\theta}(x') < \gamma$	At present, the more mainstream and commonly used metrics for LLM. These metrics can have better detection rate but not applicable when the output information is text only
Synonym	[82]		
Neighbor	[19, 86]		
Zlib	[36]	$\mathcal{L}_{f_\theta}(x)/zlib(x) < \gamma$, the ratio of perplexity to zlib value should be small enough	
Min-K% Prob	[107]	The likelihood of the K% least probable tokens should be less than threshold	
Min-K%++	[134]	The training data points tend to be local maxima or near the local maxima of the input dimension	
DE-COP	[30]	The accuracy of MCQ of suspect data between the paraphrase and verbatim text is higher than that of the non-membership data	Can be used in scenario of the output information is only text but the relative difference may lead to a little error

When employing reference-based methods, the detection and decision process can be as follows: comparing the perplexity of the text to be detected on the target LLM with its perplexity on the reference model. If the former is smaller, it can be inferred that the text to be detected is more likely to have been trained on the target model and can be classified as member data [31, 73, 84]. This can be expressed by the formula $\mathcal{L}(f_\theta, x) < \mathcal{L}(f'_\theta, x)$.

However, as discussed in Section 4.2, reference-based methods are not yet mature, and currently, non-reference methods are more commonly used. In the absence of a reference model, derived perplexity-based methods adopt a strategy wherein the original text to be detected is modified, and membership inference is determined by comparing the perplexity of the target text with the modified text on the model. If the perplexity of the target text is significantly lower than that of the modified text, it indicates that the target text more likely belongs to the training dataset: $\mathcal{L}_{f_\theta}(x)/\mathcal{L}_{f_\theta}(x') < \gamma$.

Under this strategy, it is crucial to reasonably modify the detection data to create the comparative data x'. Some have proposed methods for distinguishing between the original version of the data and its perturbed version [19, 82, 86]. According to the previous definition, this determination must be based on the assumption that samples trained during the LLM's training (membership data) should have lower perplexity on their original version x than on the perturbed data x'. Perturbations can take various forms but should not deviate significantly from the original text, such as lowercase replacement, synonym replacement [82], and neighborhood attack (perturbing the target sequence to create n adjacent points) [19, 86], etc.

Furthermore, some researchers have proposed the *zlib* metric [36]. This involves comparing the perplexity of potential membership data (data to be detected) on the target model with the text entropy ratio: $\mathcal{L}_{f_\theta}(x)/zlib(x) < \gamma$. Models trained on the training set exhibit very low perplexity for their members, while zlib calculates the number of bits required to compress a text sequence using the zlib library, independent of the training data [19].

In summary, some papers [19, 31, 36, 73, 82, 84, 86] have proposed more complex metrics based on the fundamental concept of perplexity. Regardless of the metrics, they are all based on the principle that if the data to be detected belongs to the membership data, its perplexity on the target model should be as low as possible.

(3) **Performance-based Metrics**

In addition to perplexity-based metrics, some other scholars have proposed metrics based on the performance of different data on LLMs. Performance can have various meanings, such as the probability of the data to be detected on the target model or designing algorithms such that membership data has some indicators on the LLM different with non-membership data. These metrics are founded on the principle that membership data must "perform better" on LLMs than non-membership data, with the greatest possible performance disparity. In other words, detectors use non-membership data which have been determined by benchmark dataset to do preliminary experiments on LLMs. From these experiments, a lower bound is established based on the performance of non-membership data. The performance of the data to be detected on the LLM will be compared with the lower bound. If the performance exceeds the threshold, the target data is considered the member of the training dataset.

For instance, Shi et al. [107] proposed Min-$K\%$ Prob method, where membership is determined based on comparing the likelihood of the least probable $K\%$ tokens with the threshold. Specifically, the method assesses whether the average log-likelihood of the next token, conditioned on the preceding sequence, exceeds a predefined threshold. If it does, the data to be detected can be considered membership data. When the token probability distribution is not accessible, a new method was proposed [30]: using the accuracy of multiple-choice questions (MCQs) on paraphrased non-membership data as a lower bound. This baseline is used to test the target detection data. If the accuracy of the MCQs generated from the target data and its paraphrases exceeds this threshold, the data is considered to have been used to train the LLM.

In conclusion, this evaluation method imposes high requirements on the benchmark's classification criteria and data sources. Evaluators must clarify two issues before detection: first, how to preliminarily determine membership and non-membership data, and second, whether the initial data can produce significant performance differences on the LLM.

5.3 Analysis of Determination Metrics

Sections 5.1 and 5.2 introduce different membership determination metrics corresponding to two distinct detection approaches. In this section, we explore the reasons that why the memorization data extraction and membership inference

require different metrics. Additionally, we also discuss the specific application scenarios for these two varieties of metrics.

5.3.1 Comparative Analysis of Metrics. In the memorization extraction method, the only source of interaction between the detector and the LLM is the prompts designed by the detector, and the outputs obtained are limited to the text generated according to these prompts and context. Theoretically, constructing auxiliary datasets, using binary search methods, and employing text similarity metrics are all designed based on scenarios where the output information consists solely of text.

In terms of membership inference methods for LLM, we adopt a strategy akin to traditional MIA, first establishing a classification criterion (Section 5.2.1) and then setting specific metrics based on the available output information. If the target of detection is a pre-trained LM that has not undergone fine-tuning and alignment, the detector can access the probability distribution of tokens in the generated text, i.e., the likelihood of specific tokens appearing. In this case, the metric must be based on the probability distribution, leading to the introduction of the concept of perplexity. Other metrics, such as *lowercase* [19] and *zlib* [36], are also designed based on the premise that the detector can access the token probability distribution as the output information. If the target model has used fine-tuning and alignment, and the detector can only access the text output, then the evaluation methods and metrics must be redesigned to ensure a sufficient distinction between membership and non-membership data.

In some complex scenarios, the selection of metrics determines the accuracy and efficiency of detecting training data for LLMs. Different membership determination metrics may lead to different computational cost and detection accuracy. For example, if it is known that the output information of the detection target consists only of text, which method is preferable—"prompt-based memorization extraction combined with text (similarity) comparison" or "establishing a benchmark and using performance-based metrics"? There is currently no definitive answer, as each method has its strengths and weaknesses. The former has a lower operational cost in terms of prompt manipulation, but it demands a higher level of design and optimization for the prompts and must consider the limitations imposed by LLM safety mechanisms (e.g., RLHF) on the output. The latter does not require prompt engineering or the design of auxiliary datasets, but its evaluation is heavily reliant on the relative relationship between the performance of the detected text in the LLM and a predetermined threshold. Since the inherent output uncertainty of LLMs, such relative metrics may lead to a certain degree of misjudgment.

5.3.2 Lessons and Future Trend. Through the analysis of different metrics of detecting training data for LLMs, several key challenges and lessons emerge that shape our understanding in this field.

(1) **The complexity and trade-offs in metrics**: Memorization data extraction, although effective in theory, faces practical limitations, such as the construction and storage of massive auxiliary datasets, as well as the difficulty of accurately identifying memorized content due to LLMs' generalization capabilities. Meanwhile, membership inference methods offer a more probabilistic approach but suffer from high computational costs and the challenge of creating effective surrogate models, especially for LLMs. The reliance on perplexity as a core metric across both methods highlights its utility, but also its limitations, particularly in distinguishing between membership and non-membership simple sequences, suggesting a need for more appropriate membership determination metrics.

(2) **The reliance on auxiliary dataset or probabilistic metrics**: The difficulty of designing methods that are both computationally efficient and robust against false positives. Techniques like suffix arrays and text similarity metrics offer promising solutions but require careful calibration to ensure efficiency and accuracy. Detectors should develop more sophisticated and compressed auxiliary datasets, combined with more efficient search algorithms. Additionally,

Table 3. The List of Target Models in Experiment and Evaluation

Target Model	Common Size	Paper
GPT-2	117M, 345M, 774M, 1.5B	[19, 31, 88]
T5	77M-11B	[16]
Llama	7B, 13B, 65B, 70B	[30, 65–67, 73, 77, 88, 107, 134, 137, 140]
OPT	125M-127B	[16, 30, 65, 66, 73, 77, 88, 137]
Falcon	7B-180B	[66, 73, 88]
Mistral	7B	[30, 88]
Mixtral	8x7B	[30]
GPT-Neo	125M, 1.3B, 2.7B, 6B	[31, 67, 70, 73, 87, 88, 93, 107, 134, 137]
Pythia	1.4B, 2.7B, 6.9B	[65, 66, 77, 84, 87, 88, 107, 134, 137, 140]
GPT-3.5	175B	[20, 28, 30, 42, 66, 137]
Claude 2	13.7B	[30, 66]
Baichuan 2	7B, 13B	[137]
Qwen 1.5	7B, 14B, 32B, 72B	[137]

new metrics derived from perplexity or token probability comparisons will be crucial in reducing false positives and refining the detection process.

(3) **Future trend**: In conclusion, these metrics reveal the strategies but no single method is universally superior. Future research should focus on creating hybrid methods that leverage the strengths of both memorization extraction and membership inference while minimizing their respective weaknesses. This could involve combining textual similarity with probabilistic distribution analysis to balance accuracy, robustness, and efficiency in detecting training data usage across different varieties of LLMs and training data.

6 Target Models and Benchmark Datasets in Detection

6.1 Target Models

In the experiment and evaluation of LLM training data detection methods, the following target models are commonly used, with the corresponding references for each LLM listed in Table 3. The specific details of each model are as follows:

- **GPT-2** [99]: A second-generation generative pre-trained model developed by OpenAI with parameter size from 117M to 1.5B. As subsequent LLMs have grown in parameter size, GPT-2 is often used as an auxiliary model or benchmark to compare detection performance across LLMs of varying parameter scales.
- **T5** [100]: A large-scale model proposed by Google, based on the Transformer architecture that facilitates flexible task design and efficient transfer learning. Its core concept lies in utilizing a unified framework to handle diverse tasks, thereby streamlining the application and development of the model.
- **Llama** [116]: One of the most commonly used target models, with parameter sizes typically including 7B, 13B, 65B, and 70B. Llama is trained for more epochs than other LLMs, resulting in better memory retention of the data.
- **OPT** [135]: A model family ranging in size from 125M to 127B. These models generally perform worse than other LLMs, partly because they have undergone fewer training epochs. Consequently, the detection performance on OPT is also lower than on other models.

- **Falcon** [2]: This model surpasses Llama in detection performance in certain scenarios, though further details on other parameters are limited.
- **Mistral** [62]: A model similar to Llama, with training details not publicly disclosed. It is currently one of the best-performing models in terms of detection accuracy.
- **Mixtral** [63]: A multimodal large-scale model focuses on integrating textual and visual information, achieving strong performance in cross-modal understanding and generation tasks through joint modeling. Detectors can detect text and image data on target model.
- **GPT-Neo** [9]: A model family trained on the Pile dataset [38], commonly used in experiments as a causal language model. Its training objective is to predict the next token in a given sequence. Available in parameter sizes of 125M, 1.3B, 2.7B, and 6B.
- **Pythia** [8]: A model family also trained on the Pile dataset, with parameter sizes primarily including 1.4B, 2.7B, and 6.9B. It is mainly used to study the impact of different parameter scales within the same model family on detection capabilities.
- **GPT-3.5** [1]: A large model developed by OpenAI, enhanced using manually labeled data and reinforcement learning. This model is a fully black-box model, with the detector having access only to the final generated text.
- **Claude 2** [3]: Similar to GPT-3.5, Claude 2 is a fully black-box model, with its training data, parameters, and training algorithms not visible to the detector, who must rely solely on the final output to infer the membership of the data.
- **Baichuan 2** [5]: Representative Chinese text generation LLM, which is trained on a high-quality corpus with 2.6 trillion tokens.
- **Qwen 1.5** [113]: Similar to Baichuan 2, Qwen 1.5 is also used for Chinese text detection. It is an open-source LLM and the parameter size ranging from 0.5B to 110B, and also an MoE model.

6.2 Published Benchmark Datasets

As mentioned in Section 5.2.1, membership inference-based methods typically rely on a benchmark dataset, which consists of data samples that have been identified as belonging to either the membership or non-membership category. The purpose of such a dataset is similar to the training dataset in machine learning and the shadow model in traditional MIA, serving to establish the distinction standard between membership (seen) and non-membership (unseen) data, thus laying the groundwork for the subsequent algorithm development. Commonly used benchmark datasets are listed in Table 4. These datasets can be categorized into several groups, as detailed below:

(1) **Webpage data**. The earliest target of LLM training data detection was webpage content, such as data extracted from Google or Wikipedia. Shi et al. [107] were the first to propose WikiMIA, a benchmark dataset primarily based on Wikipedia content. A key feature of this dataset is the use of the comparison between the publication date of the data and the release date of the LLM to determine membership status. WikiMIA uses 2023 as the cutoff point, with event pages published after this date categorized as non-membership data, and event pages published before 2017 (as LLMs were released after 2017) categorized as membership data. Thus, Shi et al. collected 394 membership data and 394 non-membership data samples to form the benchmark dataset, which became one of the earliest and most widely used datasets. Zhang et al. [67] applied the same method to extract pages from Chinese websites, creating a Chinese version of the PatentMIA dataset.

(2) **Copyright data**. Due to the increasing instances of copyrighted data being casually used for LLM training, some researchers have shifted their focus to copyrighted content, such as copyright books and papers. As a result,

Table 4. The List of Published Benchmark Datasets

Category	Benchmark Dataset	Component	Paper
Webpage data	WikiMIA	394 recent Wikipedia events as unseen data + 394 events from pre-2016 Wikipedia pages as membership data	[65, 67, 77, 107, 134, 137, 140]
	PatentMIA	5000 GooglePatent pages after March 1,2024 as non-member data + 5000 GooglePatent pages before 2023 as membership data	[137]
Copyright data	BookMIA	4935 texts from Book3 after 2023 as unseen data + 4935 texts from Book3 before 2023 as membership data	[107, 137, 140]
	BookTection	105 books after 2023 from BookMIA as non-member data + 65 books before 2021 as membership data	[30]
	arXivTection	25 papers in 2023 from arXiv website as non-member data + 25 papers before 2022 as membership data	[30]
	arXivMIA	1200 papers (600 member+600 non-member) in math and 800 papers (400 member+400 non-member) in computer science	[77]
Mixed dataset	Wiki-spgc	Non-membership data Wikipedia and membership text from books	[140]
	Wikimia2-spgc	Unseen data Wikipedia and seen text from both Wikipedia and books	[140]
	Pile	A 400GB dataset of heterogeneous sources containing Wikipedia, Code and huge of books	[38]
	SMIA	7000 membership samples from Wikipedia + 7000 non-membership samples from Pile	[87]
	C4	A 806 GB curated version of English web pages from the Common Crawl	[100]
	DETCON	Contains 2224 data contamination detection tasks, covering code generation and logical reasoning	[28]
	MIMIR	Membership data and non-membership data are all extracted from Pile	[29, 134]

the benchmark datasets must also be derived from books or academic papers. Shi et al. [107] were the first to propose BookMIA, and Duarte et al. [30] optimized it by constructing BookTection and arXivTection, using books and papers as sources. The distinction between membership and non-membership data follows a similar approach to WikiMIA, where more recent books or papers are considered non-membership data, and older content is classified as membership data. Liu et al. [77] further refined the paper-based datasets, proposing the arXivMIA dataset, which divides papers by subject area, such as the computer science-specific arXivMIA-CS and the mathematics-specific arXivMIA-Math.

(3) **Mixed dataset**. To extend the applicability of detection methods, some researchers have consolidated data from different domains into a single dataset, creating mixed datasets. Zhou et al. [140] proposed Wiki-spgc and Wikimia2-spgc, combining Wikipedia webpage text with copyright books. The former includes non-membership data from Wikipedia and membership data from various books, while the latter contains unseen text from Wikipedia webpages along with membership data from books and websites, presenting higher complexity than the former. Mozaffari et al. [87] merged data from Wikipedia and the Pile [38] to create SMIA, which Pile is a heterogeneous dataset containing 400GB of data from various sources, these datasets inevitably contain text from diverse origins. Besides, C4 [100] and DETCON [28] is also commonly used mixed datasets for detection. In general, if a detection method performs well on mixed datasets, its applicability is considered broader.

6.3 Experimental Comparison

To compare the detection performance of various LLMs across different datasets, we integrated the experimental results of the existing research papers into Table 5 and 6.

Table 5 reports the comparative results for memorization extraction methods. It can be observed that the detection success rate of discoverable memorization is generally higher than that of extractable memorization. This is primarily because discoverable memorization uses prompts derived directly from segments of the target data, whereas extractable memorization typically relies on manually designed prompts, which are less likely to successfully elicit memorized training data.

Table 6 presents the AUC scores of membership inference methods under different settings. The results indicate that DE-COP generally achieves superior performance as a metric, while earlier indicators such as PPL and Lower tend to yield lower detection success rates in most cases.

7 Challenges and Future Prospects of Detecting Training Data for LLM

As the scale and performance of LLMs expand, the sources of their training data have become increasingly diverse, ranging from general books, web information, academic papers, and journals to specialized data in fields such as medicine and finance. To enhance data privacy protection, methods for detecting LLM training data have been continuously proposed and improved. However, during the development of this detection field, some technical bottlenecks have emerged, such as low detection rates. This section primarily discusses the bottlenecks from two aspects and proposes potential future research directions based on these findings.

7.1 Challenges of Memorization Data Extraction

The memorization extraction method places high demands on prompt engineering, as the quality of the prompts largely determines the success rate of this approach. Currently, this technique faces the following challenges:

First, the inadequate definition of memorization is another major reason for insufficient memorization extraction rate [50]. The previously adopted definitions of memorization require the LLM to output the original text of the training data. This low-cost but highly operational definition generally can only be realized on models with a large number of parameters [16]. The stringent standards for memorization detection and the narrow scope of applicability have led to this technical challenge. Moreover, due to the diversity of model outputs, it is challenging for smaller LLMs to generate text that is exactly identical to a suffix. Consequently, the applicability of the current definition of memorization is limited.

Table 5. Extraction rate of memorization extraction methods. **Ref.** denotes the cited literature source of the detection algorithm. In column 2, E=extractable memorization, D=discoverable memorization. **Note** marks the evaluation metrics or necessary conditions adopted in experiments.

LLM	Method	Dataset	Ref.	Extract Rate	Note
GPT-2-124M	D	Pile	[93]	0.004±0.002	The metric is **exact rate**: the fraction of correctly generated all suffixes
GPT-2-1.5B	D	Pile	[93]	0.019±0.002	
	E	Google	[19]	0.000015%	
		Pile+RefinedWeb+RedPajama+Dolma	[88]	0.135%	The metric is **% tokens memorized**: the fraction of model outputs that are belongs to memorized dataset
Falcon-7B	E	Pile+RefinedWeb+RedPajama+Dolma	[88]	0.069%	
Falcon-40B	E	Pile+RefinedWeb+RedPajama+Dolma	[88]	0.122%	
Llama-7B	E	Pile+RefinedWeb+RedPajama+Dolma	[88]	0.294%	
Llama-65B	E	Pile+RefinedWeb+RedPajama+Dolma	[88]	0.789%	
OPT-125M	D	Modified version of Pile	[16]	8%	The length of prefix prompt is 100 tokens
OPT-350M	D	Modified version of Pile	[16]	12%	
OPT-1.3B	D	Modified version of Pile	[16]	15%	
	E	Pile+RefinedWeb+RedPajama+Dolma	[88]	0.031%	The metric is % tokens memorized
OPT-6.7B	E	Pile+RefinedWeb+RedPajama+Dolma	[88]	0.094%	
	D	Modified version of Pile	[16]	17.5%	The length of prefix prompt is 100 tokens
OPT-30B	D	Modified version of Pile	[16]	19%	
OPT-66B	D	Pile	[16]	20%	
GPT-Neo-125M	D	Pile	[16]	2.1%	Prompt is sampled uniformly from random subset of dataset
		Pile	[93]	0.169±0.007	The metric is exact rate
GPT-Neo-1.3B	D	Pile	[16]	3.3%	Prompt is sampled uniformly from random subset of dataset
	E	Pile	[88]	0.160%	The metric is % tokens memorized
GPT-Neo-2.7B	E	Pile	[88]	0.236%	
	D	Pile	[16]	3.4%	Prompt is sampled uniformly from random subset of dataset
GPT-Neo-6B	D	Pile	[16]	3.9%	
	E	Pile	[88]	0.220%	The metric is % tokens memorized
gpt-3.5-instruct	E	Pile+RefinedWeb+RedPajama+Dolma	[88]	0.852%	
	D	Part of ChatGPT's training set	[88]	75%	Prompt model with 50 tokens and output 50 tokens
gpt-3.5-turbo	D	Part of ChatGPT's training set	[88]	3.5%	
T5-220M	D	C4	[16]	1.2%	C4 was deduplicated by removing all documents which were near-duplicates of others
T5-770M	D	C4	[16]	4.1%	
T5-2.8M	D	C4	[16]	4.8%	

Additionally, for some closed-source LLMs, especially commercial ones (e.g., ChatGPT and Claude), developers implement defensive technologies such as RLHF [23] to protect the privacy of their training datasets and prevent the output of harmful content. While these technologies meet the output requirements of the trainers, they make it more challenging for data detectors to induce LLMs to output training (memorized) data through prompts, as the output range is limited by RLHF and similar technologies, consequently decreasing the amount of memorization training data extracted. Currently, most studies on LLM memorization data extraction focus on open-source models, where LLM parameters are known and have not undergone fine-tuning and alignment, and a significant portion of the studies [16, 19, 29, 66, 70] allow access to the original training datasets.

To address these challenges, potential research directions include the following:

Table 6. AUC scores of membership inference methods under different LLMs, datasets, detecting algorithms and metrics. **Ref.** denotes the cited literature source of the detection algorithm. **Bold** shows the best AUC score in each row.

LLM	Dataset	Ref.	AUC Scores in Different Metrics							
			PPL	Lower	Zlib	Neighbor	Min-K%	Min-K%++	DE-COP	Reference-based
GPT-3	BookMIA	[137]	0.64	**0.67**	0.54	-	0.64	-	-	-
	BookTection	[30]	0.87	**0.96**	0.78	-	0.90	-	0.86	-
ChatGPT	DETCON	[28]	0.51	-	-	-	**0.56**	-	-	-
Claude	arXivTection	[30]	-	-	-	-	-	-	**0.91**	-
Pythia-2.8B	WikiMIA	[107]	0.61	0.59	0.65	0.61	**0.67**	-	-	0.60
		[77]	-	0.62	0.63	0.62	0.63	-	-	**0.66**
		[140]	0.50	0.50	0.50	-	0.55	-	-	**0.68**
	BookMIA	[140]	0.51	0.54	0.51	-	0.68	-	-	**0.81**
Pythia-6.9B	WikiMIA	[137]	0.65	0.61	0.68	-	**0.71**	0.70	-	0.66
		[134]	-	0.62	0.64	0.66	0.66	**0.70**	-	0.64
		[65]	-	0.57	0.66	-	**0.68**	-	-	-
	SMIA	[87]	-	-	0.54	0.55	0.56	0.56	-	**0.57**
	MIMIR	[51]	**0.70**	-	0.68	0.65	0.64	-	-	-
Pythia-12B	MIMIR	[29]	-	-	0.55	-	0.56	**0.59**	-	0.58
Llama-7B	WikiMIA	[65]	-	0.52	**0.56**	-	0.54	-	-	-
		[140]	0.50	0.54	0.50	-	0.57	-	-	**0.71**
	Wiki-spgc	[140]	0.56	**0.97**	0.52	-	0.79	-	-	0.88
	Wikimia2-spgc	[140]	0.50	0.51	0.50	-	0.50	-	-	**0.62**
Llama-13B	BookTection	[30]	0.78	0.88	0.63	-	0.80	-	**0.90**	-
	WikiMIA	[137]	0.68	0.61	0.70	-	0.72	**0.84**	-	0.66
		[134]	-	0.64	0.68	0.66	0.68	**0.85**	-	0.58
		[140]	0.50	0.49	0.50	-	0.56	-	-	**0.71**
	BookMIA	[140]	0.52	0.54	0.52	-	0.72	-	-	**0.81**
Llama-30B	WikiMIA	[107]	0.70	0.59	0.72	0.71	**0.74**	-	-	0.72
		[134]	-	0.64	0.70	0.68	0.70	**0.84**	-	0.64
		[67]	-	-	0.70	-	0.69	**0.84**	-	0.65
Llama-65B	WikiMIA	[107]	0.71	0.63	0.72	0.71	**0.74**	-	-	0.73
		[134]	-	0.64	0.67	0.67	0.68	**0.73**	-	0.70
Llama-70B	BookTection	[30]	0.89	0.93	0.75	-	0.90	-	**0.97**	-
	arXivTection	[30]	-	-	-	-	-	-	**0.73**	-
Tinyllama-1.1B	arXivMIA	[77]	-	0.47	0.43	**0.55**	0.46	-	-	-
Openllama -13B	arXivMIA	[77]	-	0.50	0.44	0.55	0.49	-	-	**0.56**
OPT-6.7B	WikiMIA	[77]	-	0.59	0.63	0.59	0.63	-	-	**0.66**
		[137]	0.63	0.59	0.64	-	0.67	**0.69**	-	0.65
OPT-66B	WikiMIA	[107]	0.66	0.59	0.67	0.65	**0.71**	-	-	0.67
		[67]	0.63	0.59	0.64	-	0.67	**0.69**	-	0.65
Qwen1.5-14B	PatentMIA	[137]	0.61	-	0.62	-	**0.64**	0.63	-	0.57
Baichuan-13B	PatentMIA	[137]	0.60	-	0.63	-	0.64	0.63	-	**0.66**
Mixtral 8x7B	BookTection	[30]	0.83	0.89	0.69	-	0.84	-	**0.97**	-
	arXivTection	[30]	-	-	-	-	-	-	**0.74**	-
Mistral-7B	BookTection	[30]	0.72	0.85	0.60	-	0.76	-	**0.90**	-
GPT-Neo-1.3B	SMIA	[87]	-	-	0.61	0.67	0.68	**0.70**	-	0.53
GPT-Neo-2.7B	SMIA	[87]	-	-	0.63	0.68	0.71	**0.73**	-	0.54

(1) **Utilizing Jailbreak Attacks to Improve Detection Rates.** As discussed in Section 4.1, current commercial LLMs extensively employ alignment techniques like RLHF to ensure output quality. However, for detectors, RLHF technology hinders LLMs from outputting more training data, often refusing to generate text content memorized by the model citing copyright infringement. To overcome the output limitations of LLMs, some researchers have proposed jailbreak attacks [24, 75, 123] to bypass RLHF restrictions, diversifying the LLM's output. Future research could integrate memorization extraction methods with jailbreak attack techniques to enable LLMs to generate more memorized (training) data.

(2) **Defining Approximate Memorization.** In memorization extraction detection methods, the currently commonly used definitions of memorization require LLMs to memorize training data verbatim [58]. In most cases, it is challenging to detect specific texts by verbatim, such as personal information, within LLMs [57]. If the concept of memorization is redefined to approximate memorization—where if the LLM's output is highly similar to the original text or its paraphrase [141], the sequence can be considered memorized by LLMs—let the criteria for memorization judgment be more "relaxed", then the probability of detecting training content will increase [58, 70]. Future research can define the concept of approximate memorization, along with corresponding algorithms and determination metrics, to improve the success rate of detecting training data by prompting.

7.2 Technical Bottlenecks in Membership Inference Method

For membership inference methods targeting LLMs, the primary objective is to design algorithms that effectively distinguish membership data from non-membership data. However, this approach faces several challenges:

(1) In the pre-training phase of LLMs, to avoid overfitting problems in ML models, a substantial portion of LLM training data is typically trained for only one epoch [1, 29, 53]. According to the definition of memorization and the relationship between the number of training epochs and the degree of memorization [19], training for only one epoch leads to LLMs being unable to effectively memorize the content of the data. This results in certain data that appears infrequently being misclassified as non-membership data when using data extraction or membership inference methods, thereby increasing the false negative rate.

(2) In membership inference methods, reference-based approaches rely on training shadow models using data with the same distribution as the original training dataset. While this method can partially replicate the output behavior of the target LLM, it incurs prohibitively high training costs. In non-reference approaches, most studies rely on a single metric to determine the membership status of the data. For instance, perplexity-based methods assume that lower perplexity indicates a higher likelihood of the data belonging to the training set. However, as discussed in Section 5.2.2, using perplexity as the sole criterion introduces significant uncertainty, resulting in a high overall false-positive rate.

To address the high false-positive rates associated with membership inference methods, future research should focus on the following directions. First, multidimensional metric-based inference can be explored, where multiple metrics are combined into a feature vector to train machine learning models [48, 76]. Multidimensional feature vectors allow membership to be assessed from various perspectives, thereby reducing false-positive rates to some extent. Second, pre-trained, smaller-scale language models can be employed as surrogate models. Alternatively, techniques such as transfer learning [115] and knowledge distillation [44] can be used to enable surrogate models to emulate the behavior of the target LLM, including its probability distributions, loss values, and other related characteristics.

7.3 LLM's Internal Features for Training Data Detection

As mentioned in Section 3.1, almost all current methods of detecting training data for LLM rely on surface-level features and direct output results of the model, such as generated text, probability distributions, or loss metrics. Various detection methods have been proposed based on this output information; however, relying solely on the output to infer membership of training data is superficial and prone to inaccuracies. There has been relatively little attention paid to model information, as discussed in Section 3.1. An important research area is the exploration of LLMs' internal structures to reveal the mechanisms of LLM's prior knowledge and subsequently infer the membership of data. Liu et al. [77] have already initiated research in this area. Future work could explore developing lightweight gray-box detection tools or classifiers by analyzing the internal structure of LLMs. These tools could operate by extracting feature vectors of membership and non-membership data, derived from specific characteristics of the model's internal structure or parameters, thereby enabling classification and membership inference.

In addition, the proliferation of RAG-enabled LLMs renders traditional extraction methods increasingly inadequate. As shown in Section 4.1, the extractable memorization-based method cannot distinguish parametric memorization from dynamically retrieved content. Novel detection frameworks must incorporate mechanisms to verify if outputs derive from internal parametric memory, techniques to detect API calls to external knowledge sources and behavioral analysis of retrieval latency patterns.

7.4 Practicality of Detecting Training Data for LLM

The issue of copyright infringement and privacy violations caused by LLM developers with respect to training data remains a critical concern. Current methods for detecting membership data in LLMs focus on identifying sensitive data, but to be applicable in real-world scenarios, the following challenges must be addressed:

(1) The data determination benchmark need to exhibit greater adaptability. In membership inference methods, the standards for classifying membership and non-membership data lack robustness [29]. Specifically, as the same version of an LLM is updated, the original benchmark needs to be replaced. Additionally, these classification methods have limited applicability, typically suitable only for one type of data, such as novels, papers and Wikipedia pages, and are challenging to apply to other types of data. For example, the detection success rate of the Min-K% Prob method [107] based on Wikipedia [126] on other common data sets such as Pile [38], Common Crawl [26, 94] is much lower than that on Wikipedia. Future benchmarks should account for both timeliness and dynamicity, while ensuring high detection accuracy across various types of datasets, thus enhancing the generalizability of detection methods.

(2) There is an increasing need for research on LLM training data detection in purely black-box settings. Most existing methods allow access to token probability distributions generated by LLMs, but in real-world applications, users typically only receive the final textual output. While some memorization extraction methods and membership inference algorithms in black-box scenarios [30] have been proposed, these methods often exhibit low accuracy and are computationally expensive. Future research could explore the use of small-scale language models as surrogate models to simulate the output probabilities or loss values of the target LLM, combined with multidimensional feature extraction and techniques such as zero-shot learning. This approach would aim to develop an efficient, generalized detection tool that closely resembles real-world scenarios.

8 Conclusion

As the industrialization and large-scale development of LLM, data widely used and trained by LLMs also expose data holders to risks of copyright infringement and privacy breaches. Although there has been some reviews on detecting copyright training data and membership inference methods for AI models, there has not been a systematic analysis on detecting training data for LLMs.

This paper starts with the workflow of detecting training data for LLM, consisting of three key elements: prior knowledge, detection approach and membership determination metric. We then introduce the four varieties of LLM's information possessed by detectors and analyze the reason of the difficulty of executing MIA in LLM is much higher than traditional ML models. Subsequently, we focus on two main approaches for detecting training data and their respective research progress, analyzing the difference between two approches. We then introduce the metrics for determining the final results under the two detection approaches. Finally, we discuss the current technical challenges in this field, and based on this, we discuss potential future research directions. Through this survey, we hope to provide a solid foundation for the research field of detecting training data for large language models.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 62472035, U24B20148, 72201266, 72192843, and 72192844; in part by the Fundamental Research Funds for the Central Universities under Grant 2023CX01020 and State Key Laboratory of Intelligent Manufacturing Equipment and Technology.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[2] Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, et al. 2023. The falcon series of open language models. *arXiv preprint arXiv:2311.16867* (2023).

[3] Anthropic. 2023. *Claude.* https://www.anthropic.com/news/claude-2

[4] Yang Bai, Ting Chen, and Mingyu Fan. 2021. A survey on membership inference attacks against machine learning. *management* 6 (2021), 14.

[5] Baichuan. 2023. Baichuan 2: Open Large-scale Language Models. *arXiv preprint arXiv:2309.10305* (2023). https://arxiv.org/abs/2309.10305

[6] Jerome R Bellegarda. 2004. Statistical language model adaptation: review and perspectives. *Speech communication* 42, 1 (2004), 93–108.

[7] Stella Biderman, Usvsn Prashanth, Lintang Sutawika, Hailey Schoelkopf, Quentin Anthony, Shivanshu Purohit, and Edward Raff. 2024. Emergent and predictable memorization in large language models. *Advances in Neural Information Processing Systems* 36 (2024).

[8] Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*. PMLR, 2397–2430.

[9] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow. *If you use this software, please cite it using these metadata* 58, 2 (2021).

[10] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258* (2021).

[11] Blake Brittain. 2023. Artists Take New Shot at Stability, Midjourney. *Reuters, November* 30 (2023), 2023.

[12] Andrei Z Broder. 1997. On the resemblance and containment of documents. In *Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171)*. IEEE, 21–29.

[13] T Brown, B Mann, N Ryder, M Subbiah, JD Kaplan, P Dhariwal, A Neelakantan, P Shyam, G Sastry, A Askell, et al. 2020. Language models are few-shot learners Advances in neural information processing systems 33. (2020).

[14] Yinzhi Cao and Junfeng Yang. 2015. Towards making systems forget with machine unlearning. In *2015 IEEE symposium on security and privacy*. IEEE, 463–480.

[15] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. 2022. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1897–1914.

[16] 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).

[17] Nicholas Carlini, Matthew Jagielski, Chiyuan Zhang, Nicolas Papernot, Andreas Terzis, and Florian Tramer. 2022. The privacy onion effect: Memorization is relative. *Advances in Neural Information Processing Systems* 35 (2022), 13263–13276.

[18] 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.

[19] 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.

[20] Kent K Chang, Mackenzie Cramer, Sandeep Soni, and David Bamman. 2023. Speak, memory: An archaeology of books known to chatgpt/gpt-4. *arXiv preprint arXiv:2305.00118* (2023).

[21] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)* 2, 3 (2023), 6.

[22] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research* 24, 240 (2023), 1–113.

[23] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. *Advances in neural information processing systems* 30 (2017).

[24] Junjie Chu, Yugeng Liu, Ziqing Yang, Xinyue Shen, Michael Backes, and Yang Zhang. 2024. Comprehensive assessment of jailbreak attacks against llms. *arXiv preprint arXiv:2402.05668* (2024).

[25] T Computer. [n. d.]. RedPajama-Data: An Open Source Recipe to Reproduce LLaMA training dataset. https://github.com/taohong0511/RedPajama-Data/?tab=readme-ov-file.

[26] Common Crawl. 2023. *Common Crawl*. https://commoncrawl.org/

[27] Quang-Vinh Dang. 2021. Right to be forgotten in the age of machine learning. In *Advances in Digital Science: ICADS 2021*. Springer, 403–411.

[28] Yihong Dong, Xue Jiang, Huanyu Liu, Zhi Jin, and Ge Li. 2024. Generalization or memorization: Data contamination and trustworthy evaluation for large language models. *arXiv preprint arXiv:2402.15938* (2024).

[29] Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer, Yulia Tsvetkov, Yejin Choi, David Evans, and Hannaneh Hajishirzi. 2024. Do membership inference attacks work on large language models? *arXiv preprint arXiv:2402.07841* (2024).

[30] André V Duarte, Xuandong Zhao, Arlindo L Oliveira, and Lei Li. 2024. De-cop: Detecting copyrighted content in language models training data. *arXiv preprint arXiv:2402.09910* (2024).

[31] Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759* (2023).

[32] Niva Elkin-Koren, Uri Hacohen, Roi Livni, and Shay Moran. 2023. Can Copyright be Reduced to Privacy? *arXiv preprint arXiv:2305.14822* (2023).

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

[34] Vitaly Feldman. 2020. Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*. 954–959.

[35] Rodney A Gabriel, Tsung-Ting Kuo, Julian McAuley, and Chun-Nan Hsu. 2018. Identifying and characterizing highly similar notes in big clinical note datasets. *Journal of biomedical informatics* 82 (2018), 63–69.

[36] Jean-loup Gailly and Mark Adler. 2004. Zlib compression library. (2004).

[37] José A Galindo, Antonio J Dominguez, Jules White, and David Benavides. 2023. Large language models to generate meaningful feature model instances. In *Proceedings of the 27th ACM International Systems and Software Product Line Conference-Volume A*. 15–26.

[38] L Gao, S Biderman, S Black, L Golding, T Hoppe, C Foster, J Phang, H He, A Thite, N Nabeshima, et al. 2021. The pile: an 800GB dataset of diverse text for language modeling 2020. *arXiv preprint arXiv:2101.00027* (2021).

[39] Ting Gao. 2022. Research progress and challenges of membership inference attacks in machine learning. *Operations Research and Blurring* 12, 1 (2022), 1–15.

[40] Liu Gaoyang, Li Yutong, Wan Borui, Wang Chen, and Peng Kai. 2021. Membership inference attacks in black-box machine learning models. *J. Cyber Secur* 6, 3 (2021), 15.

[41] Sameera Ghayyur, Jay Averitt, Eric Lin, Eric Wallace, Apoorvaa Deshpande, and Hunter Luthi. 2023. Panel: Privacy Challenges and Opportunities in {LLM-Based} Chatbot Applications. (2023).

[42] Shahriar Golchin and Mihai Surdeanu. 2023. Data contamination quiz: A tool to detect and estimate contamination in large language models. *arXiv preprint arXiv:2311.06233* (2023).

[43] Wael H Gomaa, Aly A Fahmy, et al. 2013. A survey of text similarity approaches. *international journal of Computer Applications* 68, 13 (2013), 13–18.

[44] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision* 129, 6 (2021), 1789–1819.

[45] Michael M Grynbaum and Ryan Mac. 2023. The times sues openai and microsoft over ai use of copyrighted work. *The New York Times* 27 (2023).

[46] The Authors Guild. [n. d.]. AG Recommends Clause in Publishing and Distribution Agreements Prohibiting AI Training Uses. https://authorsguild.org/news/model-clause-prohibiting-ai-training/.

[47] Jiaxian Guo, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Boyang Li, Dacheng Tao, and Steven Hoi. 2023. From images to textual prompts: Zero-shot visual question answering with frozen large language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10867–10877.

[48] Mingcan Guo, Zhongyuan Han, Xintian Wang, and Jiangao Peng. 2024. Multidimensional Text Feature Analysis: Unveiling the Veil of Automatically Generated Text. (2024).

[49] Bikash Gyawali, Lucas Anastasiou, and Petr Knoth. 2020. Deduplication of scholarly documents using locality sensitive hashing and word embeddings. (2020).

[50] Valentin Hartmann, Anshuman Suri, Vincent Bindschaedler, David Evans, Shruti Tople, and Robert West. 2023. Sok: Memorization in general-purpose large language models. *arXiv preprint arXiv:2310.18362* (2023).

[51] Yu He, Boheng Li, Liu Liu, Zhongjie Ba, Wei Dong, Yiming Li, Zhan Qin, Kui Ren, and Chun Chen. 2025. Towards label-only membership inference attack against pre-trained large language models. In *USENIX Security*.

[52] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556* (2022).

[53] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. An empirical analysis of compute-optimal large language model training. *Advances in Neural Information Processing Systems* 35 (2022), 30016–30030.

[54] 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).

[55] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S Yu, and Xuyun Zhang. 2022. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys (CSUR)* 54, 11s (2022), 1–37.

[56] Li Hu, Anli Yan, Hongyang Yan, Jin Li, Teng Huang, Yingying Zhang, Changyu Dong, and Chunsheng Yang. 2023. Defenses to membership inference attacks: A survey. *Comput. Surveys* 56, 4 (2023), 1–34.

[57] Jie Huang, Hanyin Shao, and Kevin Chen-Chuan Chang. 2022. Are large pre-trained language models leaking your personal information? *arXiv preprint arXiv:2205.12628* (2022).

[58] Daphne Ippolito, Florian Tramèr, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher A Choquette-Choo, and Nicholas Carlini. 2022. Preventing verbatim memorization in language models gives a false sense of privacy. *arXiv preprint arXiv:2210.17546* (2022).

[59] Shotaro Ishihara. 2023. Training data extraction from pre-trained language models: A survey. *arXiv preprint arXiv:2305.16157* (2023).

[60] Paul Jaccard. 1912. The distribution of the flora in the alpine zone. 1. *New phytologist* 11, 2 (1912), 37–50.

[61] Matthew Jagielski, Jonathan Ullman, and Alina Oprea. 2020. Auditing differentially private machine learning: How private is private SGD? *Advances in Neural Information Processing Systems* 33 (2020), 22205–22216.

[62] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023).

[63] Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088* (2024).

[64] Nikhil Kandpal, Eric Wallace, and Colin Raffel. 2022. Deduplicating training data mitigates privacy risks in language models. In *International Conference on Machine Learning*. PMLR, 10697–10707.

[65] Masahiro Kaneko, Youmi Ma, Yuki Wata, and Naoaki Okazaki. 2024. Sampling-based Pseudo-Likelihood for Membership Inference Attacks. *arXiv preprint arXiv:2404.11262* (2024).

[66] Antonia Karamolegkou, Jiaang Li, Li Zhou, and Anders Søgaard. 2023. Copyright violations and large language models. *arXiv preprint arXiv:2310.13771* (2023).

[67] Gyuwan Kim, Yang Li, Evangelia Spiliopoulou, Jie Ma, Miguel Ballesteros, and William Yang Wang. 2024. Detecting Training Data of Large Language Models via Expectation Maximization. *arXiv preprint arXiv:2410.07582* (2024).

[68] Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[69] Tom Kocmi and Christian Federmann. 2023. Large language models are state-of-the-art evaluators of translation quality. *arXiv preprint arXiv:2302.14520* (2023).

[70] 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).

[71] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055* (2015).

[72] Jiacheng Li, Ninghui Li, and Bruno Ribeiro. 2021. Membership inference attacks and defenses in classification models. In *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy*. 5–16.

[73] Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463* (2023).

[74] Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[75] Yi Liu, Gelei Deng, Zhengzi Xu, Yuekang Li, Yaowen Zheng, Ying Zhang, Lida Zhao, Tianwei Zhang, and Yang Liu. 2023. Jailbreaking chatgpt via prompt engineering: An empirical study. arXiv 2023. *arXiv preprint arXiv:2305.13860* (2023).

[76] Yi Liu, Junzhe Yu, Huijia Sun, Ling Shi, Gelei Deng, Yuqi Chen, and Yang Liu. 2024. Efficient Detection of Toxic Prompts in Large Language Models. In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*. 455–467.

[77] Zhenhua Liu, Tong Zhu, Chuanyuan Tan, Haonan Lu, Bing Liu, and Wenliang Chen. 2024. Probing Language Models for Pre-training Data Detection. *arXiv preprint arXiv:2406.01333* (2024).

[78] Yunhui Long, Vincent Bindschaedler, Lei Wang, Diyue Bu, Xiaofeng Wang, Haixu Tang, Carl A Gunter, and Kai Chen. 2018. Understanding membership inferences on well-generalized learning models. *arXiv preprint arXiv:1802.04889* (2018).

[79] Ilya Loshchilov, Frank Hutter, et al. 2017. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101* 5 (2017).

[80] Saeed Mahloujifar, Huseyin A Inan, Melissa Chase, Esha Ghosh, and Marcello Hasegawa. 2021. Membership inference on word embedding and beyond. *arXiv preprint arXiv:2106.11384* (2021).

[81] Pratyush Maini, Hengrui Jia, Nicolas Papernot, and Adam Dziedzic. 2024. LLM Dataset Inference: Did you train on my dataset? *arXiv preprint arXiv:2406.06443* (2024).

[82] Justus Mattern, Fatemehsadat Mireshghallah, Zhijing Jin, Bernhard Schölkopf, Mrinmaya Sachan, and Taylor Berg-Kirkpatrick. 2023. Membership inference attacks against language models via neighbourhood comparison. *arXiv preprint arXiv:2305.18462* (2023).

[83] R Thomas McCoy, Paul Smolensky, Tal Linzen, Jianfeng Gao, and Asli Celikyilmaz. 2023. How much do language models copy from their training data? evaluating linguistic novelty in text generation using raven. *Transactions of the Association for Computational Linguistics* 11 (2023), 652–670.

[84] Sewon Min, Suchin Gururangan, Eric Wallace, Weijia Shi, Hannaneh Hajishirzi, Noah A Smith, and Luke Zettlemoyer. 2023. Silo language models: Isolating legal risk in a nonparametric datastore. *arXiv preprint arXiv:2308.04430* (2023).

[85] Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. 2022. Quantifying privacy risks of masked language models using membership inference attacks. *arXiv preprint arXiv:2203.03929* (2022).

[86] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning*. PMLR, 24950–24962.

[87] Hamid Mozaffari and Virendra J Marathe. 2024. Semantic Membership Inference Attack against Large Language Models. *arXiv preprint arXiv:2406.10218* (2024).

[88] Milad Nasr, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A Feder Cooper, Daphne Ippolito, Christopher A Choquette-Choo, Eric Wallace, Florian Tramèr, and Katherine Lee. 2023. Scalable extraction of training data from (production) language models. *arXiv preprint arXiv:2311.17035* (2023).

[89] Milad Nasr, Reza Shokri, and Amir Houmansadr. 2018. Machine learning with membership privacy using adversarial regularization. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*. 634–646.

[90] Milad Nasr, Shuang Songi, Abhradeep Thakurta, Nicolas Papernot, and Nicholas Carlin. 2021. Adversary instantiation: Lower bounds for differentially private machine learning. In *2021 IEEE Symposium on security and privacy (SP)*. IEEE, 866–882.

[91] Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. 2022. Codegen: An open large language model for code with multi-turn program synthesis. *arXiv preprint arXiv:2203.13474* (2022).

[92] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[93] Mustafa Safa Ozdayi, Charith Peris, Jack FitzGerald, Christophe Dupuy, Jimit Majmudar, Haidar Khan, Rahil Parikh, and Rahul Gupta. 2023. Controlling the extraction of memorized data from large language models via prompt-tuning. *arXiv preprint arXiv:2305.11759* (2023).

[94] Jay M Patel and Jay M Patel. 2020. Introduction to common crawl datasets. *Getting structured data from the internet: running web crawlers/scrapers on a big data production scale* (2020), 277–324.

[95] Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116* (2023).

[96] Kai Peng. 2024. Targeted Training Data Extraction—Neighborhood Comparison-Based Membership Inference Attacks in Large Language Models. *Applied Sciences* 14 (2024).

[97] Didik Dwi Prasetya, Aji Prasetya Wibawa, and Tsukasa Hirashima. 2018. The performance of text similarity algorithms. *International Journal of Advances in Intelligent Informatics* 4, 1 (2018), 63–69.

[98] A Radford. 2018. Improving language understanding by generative pre-training. (2018).

[99] 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.

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

[101] Abhilasha Ravichander, Jillian Fisher, Taylor Sorensen, Ximing Lu, Yuchen Lin, Maria Antoniak, Niloofar Mireshghallah, Chandra Bhagavatula, and Yejin Choi. 2025. Information-guided identification of training data imprint in (proprietary) large language models. *arXiv preprint arXiv:2503.12072* (2025).

[102] Rebecca Roelofs, Vaishaal Shankar, Benjamin Recht, Sara Fridovich-Keil, Moritz Hardt, John Miller, and Ludwig Schmidt. 2019. A meta-analysis of overfitting in machine learning. *Advances in Neural Information Processing Systems* 32 (2019).

[103] Noveen Sachdeva, Benjamin Coleman, Wang-Cheng Kang, Jianmo Ni, Lichan Hong, Ed H Chi, James Caverlee, Julian McAuley, and Derek Zhiyuan Cheng. 2024. How to Train Data-Efficient LLMs. *arXiv preprint arXiv:2402.09668* (2024).

[104] Victor Sanh, Albert Webson, Colin Raffel, Stephen H Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2021. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207* (2021).

[105] Navoda Senavirathne and Vicenc Torra. 2020. On the Role of Data Anonymization in Machine Learning Privacy. *IEEE* (2020).

[106] Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*. PMLR, 4596–4604.

[107] Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen, and Luke Zettlemoyer. 2023. Detecting pretraining data from large language models. *arXiv preprint arXiv:2310.16789* (2023).

[108] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. 2017. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*. IEEE, 3–18.

[109] Luca Soldaini, Rodney Kinney, Akshita Bhagia, Dustin Schwenk, David Atkinson, Russell Authur, Ben Bogin, Khyathi Chandu, Jennifer Dumas, Yanai Elazar, et al. 2024. Dolma: An open corpus of three trillion tokens for language model pretraining research. *arXiv preprint arXiv:2402.00159* (2024).

[110] Congzheng Song and Vitaly Shmatikov. 2019. Auditing data provenance in text-generation models. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 196–206.

[111] Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2023. Galactica: A large language model for science. arXiv 2022. *arXiv preprint arXiv:2211.09085* 10 (2023).

[112] MosaicML NLP Team et al. 2023. Introducing mpt-7b: A new standard for open-source, commercially usable llms.

[113] Qwen Team. 2024. Introducing Qwen1.5. https://qwenlm.github.io/blog/qwen1.5/

[114] Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. 2022. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems* 35 (2022), 38274–38290.

[115] Lisa Torrey and Jude Shavlik. 2010. Transfer learning. In *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 242–264.

[116] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971* (2023).

[117] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

[118] Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. 2019. Demystifying membership inference attacks in machine learning as a service. *IEEE transactions on services computing* 14, 6 (2019), 2073–2089.

[119] Nikhil Vyas, Sham M Kakade, and Boaz Barak. 2023. On provable copyright protection for generative models. In *International Conference on Machine Learning*. PMLR, 35277–35299.

[120] Jiapeng Wang and Yihong Dong. 2020. Measurement of text similarity: a survey. *Information* 11, 9 (2020), 421.

[121] LL Wang, Peng Zhang, Zheng Yan, and Xiaokang Zhou. 2019. A survey on membership inference on training datasets in machine learning. *Cyberspace Security* 10, 10 (2019), 1–7.

[122] Lauren Watson, Chuan Guo, Graham Cormode, and Alex Sablayrolles. 2021. On the importance of difficulty calibration in membership inference attacks. *arXiv preprint arXiv:2111.08440* (2021).

[123] Alexander Wei, Nika Haghtalab, and Jacob Steinhardt. 2024. Jailbroken: How does llm safety training fail? *Advances in Neural Information Processing Systems* 36 (2024).

[124] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652* (2021).

[125] Laura Weidinger, John Mellor, Maribeth Rauh, Conor Griffin, Jonathan Uesato, Po-Sen Huang, Myra Cheng, Mia Glaese, Borja Balle, Atoosa Kasirzadeh, et al. 2021. Ethical and social risks of harm from language models. *arXiv preprint arXiv:2112.04359* (2021).

[126] Wikipedia contributors. 2024. Wikipedia — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Wikipedia&oldid=1243483768 [Online; accessed 2-September-2024].

[127] Tianyu Wu, Shizhu He, Jingping Liu, Siqi Sun, Kang Liu, Qing-Long Han, and Yang Tang. 2023. A brief overview of ChatGPT: The history, status quo and potential future development. *IEEE/CAA Journal of Automatica Sinica* 10, 5 (2023), 1122–1136.

[128] Heng Xu, Tianqing Zhu, Lefeng Zhang, Wanlei Zhou, and Philip S. Yu. 2023. Machine Unlearning: A Survey. *ACM Comput. Surv.* 56, 1, Article 9 (aug 2023), 36 pages. doi:10.1145/3603620

[129] Han Yang, Mingchen Li, Huixue Zhou, Yongkang Xiao, Qian Fang, and Rui Zhang. 2023. One LLM is not Enough: Harnessing the Power of Ensemble Learning for Medical Question Answering. *medRxiv* (2023).

[130] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *2018 IEEE 31st computer security foundations symposium (CSF)*. IEEE, 268–282.

[131] Sajjad Zarifzadeh, Philippe Cheng-Jie Marc Liu, and Reza Shokri. 2023. Low-cost high-power membership inference by boosting relativity. (2023).

[132] Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. *Advances in neural information processing systems* 32 (2019).

[133] Chiyuan Zhang, Daphne Ippolito, Katherine Lee, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2023. Counterfactual memorization in neural language models. *Advances in Neural Information Processing Systems* 36 (2023), 39321–39362.

[134] Jingyang Zhang, Jingwei Sun, Eric Yeats, Yang Ouyang, Martin Kuo, Jianyi Zhang, Hao Yang, and Hai Li. 2024. Min-K%++: Improved Baseline for Detecting Pre-Training Data from Large Language Models. *arXiv preprint arXiv:2404.02936* (2024).

[135] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. 2022. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068* (2022).

[136] Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B Hashimoto. 2024. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics* 12 (2024), 39–57.

[137] Weichao Zhang, Ruqing Zhang, Jiafeng Guo, Maarten de Rijke, Yixing Fan, and Xueqi Cheng. 2024. Pretraining Data Detection for Large Language Models: A Divergence-based Calibration Method. *arXiv preprint arXiv:2409.14781* (2024).

[138] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* (2023).

[139] Xuandong Zhao, Lei Li, and Yu-Xiang Wang. 2022. Provably confidential language modelling. *arXiv preprint arXiv:2205.01863* (2022).

[140] Baohang Zhou, Zezhong Wang, Lingzhi Wang, Hongru Wang, Ying Zhang, Kehui Song, Xuhui Sui, and Kam-Fai Wong. 2024. DPDLLM: A Black-box Framework for Detecting Pre-training Data from Large Language Models. In *Findings of the Association for Computational Linguistics ACL 2024*. 644–653.

[141] Jianing Zhou and Suma Bhat. 2021. Paraphrase generation: A survey of the state of the art. In *Proceedings of the 2021 conference on empirical methods in natural language processing*. 5075–5086.

Received 20 February 2025; revised 12 March 2025; accepted 5 June 2025