Probing Language Models for Pre-training Data Detection

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have shown their impressive capabilities, while also raising concerns about the data contamination problems due to privacy issues and leakage of benchmark datasets in the pre-training phase. Therefore, it is vital to detect the contamination by checking whether an LLM has been pre-trained on the target texts. Recent studies focus on the generated texts and compute perplexities, which are superficial features and not reliable. In this study, we propose to utilize the probing technique for pre-training data detection by examining the model's internal activations. Our method is simple and effective and leads to more trustworthy pre-training data detection. Additionally, we propose ArxivMIA, a new challenging benchmark comprising arxiv abstracts from Computer Science and Mathematics categories. Our experiments demonstrate that our method outperforms all baselines, and achieves state-of-the-art performance on both WikiMIA and ArxivMIA, with additional experiments confirming its efficacy[1].

1 Introduction

Large language models (LLMs) trained on massive corpora of texts demonstrate extraordinary abilities to understand, reason, and generate following natural language instructions (Brown et al., 2020; Anil et al., 2023). Meanwhile, the open-source of LLMs has significantly contributed to the advancement and collaborative development within the LLM community (Zhang et al., 2022; Touvron et al., 2023b; Biderman et al., 2023; Bai et al., 2023; Team, 2023). Despite this progress, the lack of transparency raises ethical and legal questions, particularly about the use of potentially private data sourced from the internet, and threatens the reliability of benchmark evaluations due to the risk of leaking test data. Therefore, determining if certain texts have been utilized in the pre-training of LLMs becomes a critical task.

Recent efforts to detect pre-training data in LLMs have attracted significant attention. Several studies have been proposed to investigate dataset contamination, including prompting LLMs to generate data-specific examples or using statistical methods to detect contamination in test sets (Sainz et al., 2023; Golchin and Surdeanu, 2023; Oren et al., 2023). Concurrently, Membership Inference Attacks (MIAs) in Natural Language Processing have been extensively explored for their potential to discern whether specific data was used in LLMs' pre-training (Carlini et al., 2021; Mireshghallah et al., 2022; Mattern et al., 2023; Shi et al., 2023). The above solutions have achieved a certain success. However, we argue that using superficial features, like generated texts or loss metrics, is suboptimal since such information may not always be reliable.

Different from these conventional approaches, we propose a simple and effective pre-training data detection method that utilizes the probing technique to examine the model's internal activations. This approach is based on the assumption: Texts that have been seen during the model's pre-training phase are represented differently in the model's internal activations compared to texts that have not been seen, so we could train a linear probe classifier to distinguish them.

As illustrated in Figure 1, our method consists of three main steps: (1) We initiate our process by gathering a training dataset that the LLM has not previously been trained on, splitting the data into member and non-member subsets. We then inject the member data into the target model through a fine-tuning process on the member data alone. This step enables us to create a proxy model that retains the memory of the member data from the training dataset. (2) Subsequently, we input the data

[1]Our code and dataset are available at https://github.com/xxxxxx

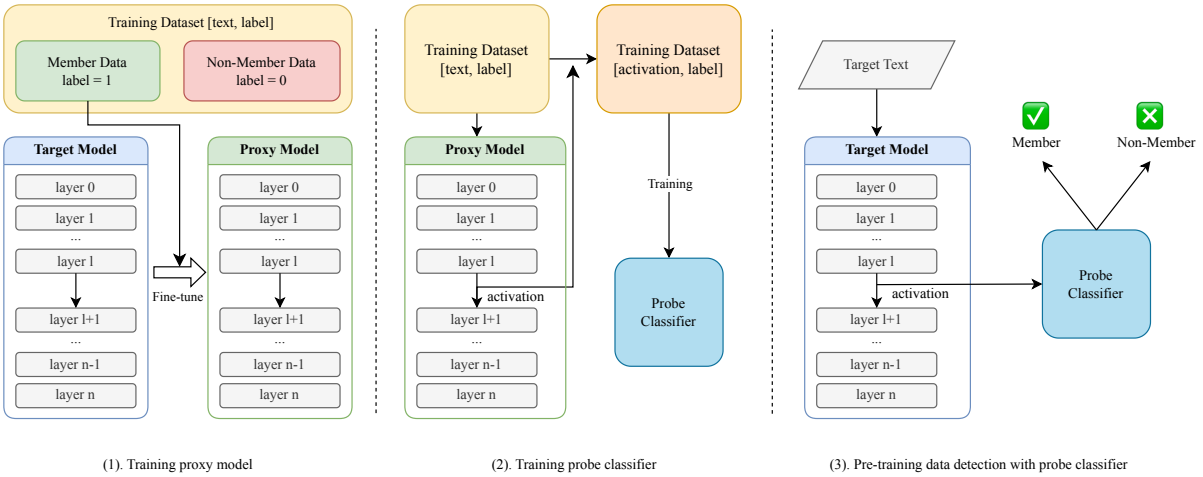

Figure 1: An overview of our method. Member data from the training dataset is first used to fine-tune the target model into a proxy model, from which activations are derived for training the probe classifier. The probe classifier then evaluates the target text to determine whether it was included in the model's pre-training data.

from the training dataset into the proxy model and extract the model's internal activations. These activations are employed to train a probe classifier that can distinguish between member and non-member data. (3) Finally, given a target text, we can input it to the target model, extract the internal activations, and let the probe classifier infer whether the text is member data. In other words, the probe classifier could assess whether the target text has been seen during the pre-training phase.

In order to evaluate various pre-training data detection methods in a more challenging scenario, we introduce ArxivMIA, a difficult benchmark in the academic domain. In contrast to the existing WikiMIA (Shi et al., 2023) benchmark, which primarily utilizes event data from Wikipedia pages—thus prone to a higher duplication rate in pre-training datasets—ArxivMIA presents a more challenging scenario. The academic abstracts within ArxivMIA are rarer on the internet compared to Wikipedia content, naturally resulting in a lower duplication rate. Furthermore, the inherent complexity of texts targeted at researchers adds another layer of difficulty for model memorization. This combination of rarity and complexity makes it exceedingly challenging for large models to memorize such content during the pre-training process, making its detection through conventional methods markedly tougher. Therefore, ArxivMIA stands as an especially rigorous benchmark, highlighting the need for more sophisticated pre-training data detection methods.

Our contributions can be summarized as follows:

- We propose a novel pre-training data detection method that utilizes the probing technique. To the best of our knowledge, this is the first work to examine LLMs' internal activations to determine whether a given text was included in the pre-training data.

- We propose ArxivMIA, a new benchmark in the academic domain to assess pre-training data detection methods. With a low duplication rate and the inherent complexity of texts, ArxivMIA presents a more challenging scenario and highlights the need for more sophisticated pre-training data detection methods.

- We conduct extensive experiments on WikiMIA and ArxivMIA benchmarks. In addition, we also evaluate various detection methods on a downstream task datasets contamination challenge. Through comprehensive experimentation, we demonstrate that our proposed method outperforms all baselines, and achieves state-of-the-art performance.

2 Related Work

Related work involves membership inference attacks in NLP and dataset contamination.

Membership Inference Attacks in NLP. Membership Inference Attacks (MIAs) are designed to identify if a particular data sample was included in the training dataset of a machine learning model (Shokri et al., 2017; Yeom et al., 2018; Hu et al.,

2022). Most MIAs take a black-box setting, assuming that the adversary only has access to the model confidence or loss scores (Yeom et al., 2018; Sablayrolles et al., 2019; Jayaraman et al., 2021; Watson et al., 2021). Unlike it, similar to Leino and Fredrikson (2020), we consider a white-box setting where the adversary has access to the model weights and activations. Specifically in NLP, a lot of studies have been proposed (Carlini et al., 2021; Mireshghallah et al., 2022; Mattern et al., 2023; Shi et al., 2023). Carlini et al. (2021) and Mireshghallah et al. (2022) separately investigated Likelihood Ratio Attacks for causal language models and masked language models. Mattern et al. (2023) proposed a neighbor attack that compares model loss for a given sample to losses of synthetically generated neighbor texts. Shi et al. (2023) measured the likelihood of outlier words in a given text, thereby assessing whether the text was likely part of a model's pre-training corpora. Similar to Shi et al. (2023), we aim to detect pre-training data in LLMs. However, different from existing attacks that rely on the model's superficial features, we focus on the LLMs' internal activations, and the experiments show that our method outperforms existing attacks.

Dataset Contamination. The dataset contamination in LLMs has been widely studied since benchmark datasets are intentionally or unintentionally included in pre-training corpora. The n-gram based overlap comparison methods not only require access to training corpora but take a long time to compute (Gao et al., 2020; Brown et al., 2020; Dodge et al., 2021; Chowdhery et al., 2023; Anil et al., 2023; Touvron et al., 2023a,b). Without access to pre-training corpora, there are also some methods to detect dataset contamination. Sainz et al. (2023) prompted LLMs to generate verbatim examples of a dataset split. Golchin and Surdeanu (2023) proposed the 'Data Contamination Quiz', which employs a multiple-choice format to assess a model's ability to recognize original dataset instances among perturbed versions. Oren et al. (2023) presented a statistical test to demonstrate test set contamination in language models, leveraging the concept of exchangeability in benchmark datasets and comparing model log probabilities against shuffled dataset permutations.

3 Methodology

3.1 Overview

Different from conventional detection methods in MIA that rely on the model's superficial features, we directly analyze the model's internal activations, providing a deeper insight into its pre-training history. Our method employs the probe technique originally proposed by Alain and Bengio (2016). This technique hypothesizes that the internal representations of a model inherently contain specific encoded properties, so we could train a linear probe classifier with logistic regression for the detection of these properties. In our context, we are interested in determining whether a text sample was included in the model's pre-training dataset. The framework of our method is illustrated in Figure 1.

3.2 Task Definition

The task of pre-training data detection follows a white-box setting of MIAs where the adversary has access to the model $\mathcal{M}$'s architecture and weights, but not the pre-training data. The adversary aims to determine whether a sample s was included in the pre-training data of the model $\mathcal{M}$ with an attack method A: $A_{\mathcal{M}}(s) \rightarrow \{0, 1\}$, where 1 represents member (seen) data, 0 denotes non-member (unseen) data. Usually, we have a scoring function f, then can get the confidence score $f(s) \in [0, 1]$, which represents the probability of the sample being a part of the pre-training dataset. Then we can classify the sample as a member or non-member based on a threshold γ :

$$A_{\mathcal{M}}(s) = \mathbb{1}\left[f(s) < \gamma\right]$$

3.3 Training Proxy Model

Training a probe classifier needs a dataset $\{\langle x_i, y_i \rangle\}$, where x_i represents the sample's activation, and y_i is a binary label indicating whether the sample is a member or non-member data. However, the absence of pre-training data makes it impossible to obtain the activations of the member or non-member samples. Therefore, we first gather a training dataset that the LLM has not previously been trained on, splitting the data into member and non-member subsets, and inject the member data into the proxy model to simulate data contamination, as detailed in subsection 4.2. The training dataset is $D = \{\langle s_i, y_i \rangle\}$, where s_i represents the sample, and y_i is a binary label indicating whether the sample is a member or non-member.

Prompt Template for sample. Each sample of D is processed using a prompt template. This prompt template is crucial for standardizing the input for consistency. In the subsequent experiments, we use the following prompt template: "*Here is a statement: [SAMPLE] \n Is the above statement correct? Answer:*".

Training Proxy Model. Next we need to inject the member samples $D_{member} = \{s_i \mid \langle s_i, y_i \rangle \in D, y_i = 1\}$ into the model $\mathcal{M}$ to let it memorize the member data. This injection is accomplished by fine-tuning the model on D_{member}. This step aims to make the model $\mathcal{M}$ memorize the member data, and the fine-tuning process is used to simulate the pre-training process. After this, we can get the proxy model $\mathcal{M}'$, which retains the memory of D_{member}. The proxy model $\mathcal{M}'$ is then used to generate the member and non-member sample activations x for the dataset D.

3.4 Training Probe Classifier

The probe classifier takes the form $P_\theta(x) = \sigma(Wx)$, where σ denotes the sigmoid function and W represents the trainable weights. After obtaining the proxy model $\mathcal{M}'$, we construct the training dataset with D for the probe.

In the paper, we focus on causal language models, which are trained to predict the next word given the previous words. In order to capture the representation of the sample, for each sample $\langle s_i, y_i \rangle$ in D, we extract the activation x_l from the final token of the input in layer l of the model $\mathcal{M}'$. Finally, we get the dataset $\{\langle x_i^l, y_i \rangle\}$, which is used to train the probe P_θ with logistic regression.

3.5 Pre-training Data Detection with Probe Classifier

Given a benchmark, we already trained a probe P_θ, which can be used to detect whether a sample is included in the pre-training data. For a sample s, we preprocess it with the above prompt template, then feed it into the model $\mathcal{M}$ to get the activation x^l. Finally, we can get the confidence score $P_\theta(x^l)$, which represents the probability of the sample being a part of the pre-training dataset. This score is then utilized to classify the sample as a member or non-member based on a threshold γ :

$$A_{\mathcal{M}}(s) = \mathbb{1}\left[P_\theta(x^l) < \gamma\right]$$

4 Data Construction

4.1 ArxivMIA

To evaluate various pre-training data detection methods in a more challenging scenario, we introduce ArxivMIA, a new benchmark comprising abstracts from the fields of Computer Science (CS) and Mathematics (Math) sourced from Arxiv. In contrast to the existing WikiMIA (Shi et al., 2023) benchmark, which primarily utilizes event data from Wikipedia pages—thus prone to higher duplication rates in pre-training datasets—ArxivMIA presents a more challenging scenario. The academic abstracts within ArxivMIA are rarer on the internet compared to Wikipedia content, naturally resulting in a lower duplication rate. Furthermore, the inherent complexity of texts targeted at researchers adds another layer of difficulty for model memorization.

For dataset construction, abstracts published post-2024 are designated as non-member data, while member data are derived from the Arxiv subset of the RedPajama dataset (Computer, 2023). The RedPajama dataset is the reproduction of the LLaMA (Touvron et al., 2023a) training dataset and is extensively utilized in pre-training LLMs (Zhang et al., 2024; Geng and Liu, 2023). This makes ArxivMIA particularly suited for testing LLMs pre-trained on the RedPajama dataset. Detailed information about ArxivMIA is presented in Table 1. In summary, ArxivMIA has three distinctive features: Firstly, it is a larger dataset with a total of 2000 samples. Secondly, it covers multiple fields, including CS and Math. Lastly, it features a longer average sentence length, with an average of 143.1 tokens per sample. These characteristics make ArxivMIA a more challenging benchmark for evaluating pre-training data detection methodologies.

4.2 Training Dataset Collection

Our method needs to construct a training dataset $D = \{\langle s_i, y_i \rangle\}$ similar to the target benchmark, where s_i represents the sample, and y_i is a binary label indicating whether the sample is a member or non-member. This dataset is pivotal for training the probe to accurately evaluate the likelihood of a sample being included in the pre-training data. However, the construction of a training dataset for the probe is challenging due to the lack of access to the pre-training data. To address this, we propose a heuristic method:

Dataset	Avg. Tokens	Members	Non-Members	Total	Real	Synthetic
WikiMIA	32.0	387	289	676	100*	100
ArxivMIA	143.1	1,000	1,000	2,000	200	200
├ ArxivMIA-CS	181.8	400	400	800	80	80
└ ArxivMIA-Math	117.2	600	600	1,200	120	120

Table 1: Information of Datasets. Real denote the number of collected real training data, and Synthetic denote the number of synthetic training data. * For convenience, we directly segregated a subset of 100 non-member data from WikiMIA as real data

Data Collection. Firstly, we need to collect a dataset $D = \{s_i\}$, and make sure they are not included in the pre-training data. There are two ways to accomplish it: (1). **Real data**: We collect the data published after the model release date. (2). **Synthetic data**: We can also use ChatGPT (Achiam et al., 2023) to synthesize similar data according to the data to be detected. The detailed process is described in Appendix A.

Dataset Split. Next, we randomly label half of the data in D as non-member data, and the other half as member data. Then we get the dataset $D = \{\langle s_i, y_i \rangle\}$.

We constructed both real and synthetic training datasets for each benchmark respectively, with specifics outlined in Table 1. Notably, for convenience, we directly segregated a subset of 100 non-member data from WikiMIA as real data, and the remaining part will be used in subsequent experiments.

5 Experiments

We evaluate the performance of our method and other baselines against open-source language models trained to predict the next word, including Pythia-2.8B (Biderman et al., 2023), OPT-6.7B (Zhang et al., 2022), TinyLLaMA-1.1B (Zhang et al., 2024) and OpenLLaMA-13B (Geng and Liu, 2023).

5.1 Datasets

WikiMIA proposed by Shi et al. (2023), a dynamic benchmark designed to periodically and automatically evaluate detection methods on any newly released pre-trained LLMs. We use the WikiMIA-32 split in this work, which contains 776 samples with a max length of 32 tokens.

ArxivMIA proposed in this work, a more complex benchmark comprising abstracts in the fields of Computer Science and Mathematics from Arxiv. The details refer to subsection 4.1.

We split each dataset into a validation set and a test set in a ratio of 2:8. The validation set is used to select the best hyperparameters, and the test set is used to evaluate the performance of the detection methods.

5.2 Evaluation Metrics

Following Shi et al. (2023); Carlini et al. (2022); Mattern et al. (2023), we assess the effectiveness of detection methods using these metrics:

Area Under the ROC Curve (AUC). The ROC curve plots the true positive rate (power) against the false positive rate (error) across various thresholds γ, which captures the trade-off between power and error. Therefore, the area under the ROC curve (AUC) serves as a singular, threshold-independent measure to evaluate the effectiveness of the detection method.

True Positive Rate (TPR) under low False Positive Rates (FPR). We report TPR under low FPR by adjusting the threshold value γ. Concretely, we set 5% as the target FPRs, and report the corresponding TPRs.

5.3 Baselines

To compare the performance of Probe Attack, we consider the following reference-free methods:

Loss Attack proposed by Yeom et al. (2018), which assesses the membership of the target sample based on the loss of the target model.

Neighbor Attack proposed by Mattern et al. (2023), which compares model loss for the target sample to losses of synthetically generated neighbor texts. We construct 100 neighbors for each sample using one-word replacement with the RoBERTa-base model (Liu et al., 2019).

Min-K% Prob proposed by Shi et al. (2023), begins by calculating the probability of each token in the target sample, then selects the k% of tokens with the lowest probabilities to compute their average log-likelihood. A high average log-likelihood suggests that the text is likely part of the pretraining data.

Following Carlini et al. (2021) and Shi et al. (2023), we also consider reference-based methods, which calibrate difficulty by quantifying the intrinsic complexity of a target sample:

Comparing to Zlib Compression. We compute the zlib entropy of the sample, which is the entropy in bits when the sequence is compressed using zlib[2]. The detection score is then determined by the ratio of the model's perplexity to the zlib entropy.

Comparing to Lowercased Text. We compute the ratio of the perplexity of the sample before and after converting it to lowercase.

Comparing to Smaller Model. We compute the sample perplexity ratio of the target model to a smaller model pre-trained on the same data.

5.4 Implementation Details

For WikiMIA, we employ Pythia-2.8B (Biderman et al., 2023) and OPT-6.7B (Zhang et al., 2022) as the target model following Shi et al. (2023). For ArxivMIA, we employ TinyLLaMA-1.1B (Zhang et al., 2024) and OpenLLaMA-13B(Geng and Liu, 2023) pre-trained on RedPajama (Computer, 2023) as the target model.

For comparing to smaller model baseline setting, we take Pythia-70M for Pythia-2.8B, OPT-350M for OPT-6.7B, and OpenLLaMA-3B for OpenLLaMA-13B. Because there is no smaller model for TinyLLaMA, we leave this baseline out for TinyLLaMA.

For the training of the proxy model, we conducted a grid search super-parameters on a held-out validation set in order to better inject member data into the model. Based on the performance, the best choice is to put all the data to be injected into one batch and train for 2 epochs. For different models and datasets, we set the best learning rate and activation extraction model layer according to the performance of the validation set.

[2] https://github.com/madler/zlib

6 Results and Analyses

In this section, we report our main result and conduct ablation studies to analyze the impact of model size and training data number for our method. We also compare the performance of various detection methods on PubMedQA (Jin et al., 2019) and CommonsenseQA (Talmor et al., 2019) in the contamination detection challenge proposed by Oren et al. (2023).

6.1 Main Results

We present the main results of our experiments in Table 2 and Table 3, where the former shows AUC values and the latter shows true positive rates at 5% false positive rates. The results show that our method consistently outperforms all baselines on both WikiMIA and ArxivMIA benchmarks, achieving state-of-the-art AUC values. We also achieve the state-of-the-art average true positive rates at 5% false positive rates across all detection methods on both benchmarks. We can further observe that:

- The average performance across all detection methods is notably lower on ArxivMIA compared to WikiMIA. This disparity underscores the increased complexity of ArxivMIA as a benchmark. In addition to our method, the Neighbor Attack method exhibits a relatively competent AUC value.

- The performance gap between various detection methods across the two fields of ArxivMIA is notable. Specifically, in the ArxivMIA-CS, the average AUC value is comparatively higher, with our method achieving its best results above 60. In contrast, in the ArxivMIA-Math, the values are only above 50, differing by approximately 10 points. This discrepancy may suggest that mathematical content in academic papers is more challenging for Large Language Models (LLMs) to memorize, and consequently, harder for our method to detect.

- As shown in Table 2, we separately test our method with real and synthetic data. On WikiMIA, the utilization of real data marginally outperforms synthetic data, while the opposite is observed on ArxivMIA. Despite a slight difference, the performance of our method is far superior to other baselines with both real and synthetic data.

Method	WikiMIA		ArxivMIA		ArxivMIA-CS		ArxivMIA-Math	
	Pythia	OPT	TinyL.	OpenL.	TinyL.	OpenL.	TinyL.	OpenL.
Reference-free Methods								
Loss Attack	63.9	63.0	45.1	49.1	45.3	51.4	44.9	47.4
Neighbor Attack	62.1	58.5	54.8	55.4	59.3	59.3	53.4	54.1
Min-K% Prob	62.7	63.2	45.5	49.2	45.0	50.2	45.8	48.5
Reference-based Methods								
Zlib Compression	63.8	62.9	42.9	43.8	38.0	40.4	44.0	44.7
Lowercased Text	64.7	61.6	46.8	50.2	43.8	47.8	48.4	50.8
Smaller Model	65.5	65.8	-	55.9	-	54.9	-	56.7
Our Method								
Probe w. Real Data	**69.8**	**68.1**	57.1	60.0	63.7	67.2	56.1	56.9
Probe w. Synthetic Data	69.4	66.2	**59.2**	**60.3**	**64.3**	**67.3**	**56.7**	**57.4**

Table 2: AUC values of different methods on WikiMIA and ArxivMIA. TinyL. denotes TinyLLaMA, OpenL. denotes OpenLLaMA. We highlight the best results in **bold**.

Method	WikiMIA		ArxivMIA		Avg.
	Pythia	OPT	TinyL.	OpenL.	
Reference-free Methods					
Loss Attack	13.7	11.4	5.1	5.6	9.0
Neighbor Attack	14.0	13.4	6.5	7.3	10.3
Min-K% Prob	16.9	15.0	4.5	5.1	10.4
Reference-based Methods					
Zlib Compression	17.3	14.4	2.5	3.5	9.4
Lowercased Text	10.1	9.1	4.3	6.3	7.5
Smaller Model	14.0	10.5	-	**8.5**	11.0
Our Method					
Probe w. Real Data	16.7	**15.4**	7.5	7.4	**11.8**
Probe w. Synthetic Data	**19.6**	10.5	**8.6**	6.8	11.4

Table 3: True positive rates for different methods at 5% positive rates on WikiMIA and ArxivMIA datasets. TinyL. denotes TinyLLaMA, OpenL. denotes OpenLLaMA. Best results are highlighted in **bold**.

6.2 Ablation Studies

We further investigate the impact of model size and training data number for our method:

Model Size. We evaluate our method and neighbor attack on ArxivMIA with different OpenLLaMA sizes (3B/7B/13B). As shown in Figure 2, the AUC values of our method increase with the model size, while the change of neighbor attack is not significant. This result indicates that our method benefits from larger models.

Number of Training Data. We also evaluate our method with different synthetic training data sizes (50, 100, 200, 500 and 1000). We conduct the

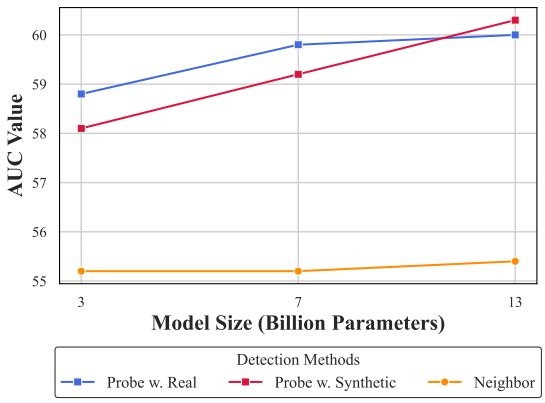

Figure 2: Comparison of AUC Values Across Different Model Sizes (best viewed in color).

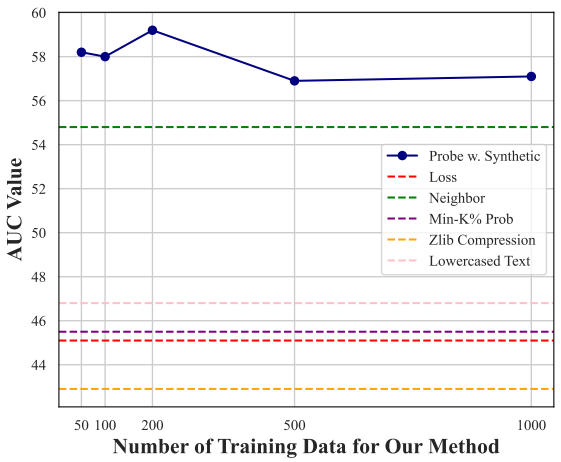

Figure 3: Comparison of AUC Values with Different Training Data Sizes (best viewed in color).

comparison experiment on ArxivMIA with TinyLLaMA. As illustrated in Figure 3, our method exhibits optimal performance with 200 training data samples. Increasing the number of training data beyond this point results in a slight decline in performance, yet it remains superior to various baselines. This indicates that our method is data efficient.

6.3 Downstream Task Datasets Contamination Detection Challenge

Method	PMQA	CQA
Reference-free Methods		
Loss Attack	48.0	49.9
Neighbor Attack	53.0	50.0
Min-K% Prob	47.5	49.6
Reference-based Methods		
Zlib Compression	46.1	48.8
Lowercased Text	50.7	49.2
Smaller Model	49.5	49.5
Our Method		
Probe w. Synthetic Data	**54.0**	**51.9**

Table 4: AUC values of various pre-training data detection methods on PubMedQA and CommonsenseQA in contamination detection challenge. PMQA denotes PubMedQA, CQA denotes CommonsenseQA. We highlight the best results in **bold**.

To support the development of further work on detecting pretraining data contamination, Oren et al. (2023) pre-trained a 1.4 billion parameter GPT-2 model (Radford et al., 2019), Contam-1.4b, with intentional downstream task datasets contamination [3]. We evaluate various detection methods on PubMedQA (Jin et al., 2019) and CommonsenseQA (Talmor et al., 2019) from this challenge. PubMedQA and CommonsenseQA have different duplication counts (how often the dataset was injected into the pre-training data) with 1 and 2, and detection at this low duplication level is extremely difficult (Oren et al., 2023).

Experimental Setup. We sampled 1000 examples from the contaminated training data as member data for each task and then sampled 1000 examples from their standard dataset as non-member data. Similar to subsection 5.1, we split each dataset into a validation set and a test set. The validation set will be used to select the best hyperparameters, and the test set for evaluation. For our method, we collected 200 synthetic training data for each task. For comparing to smaller model baseline setting, we choose Contam-Small (124M Params) pre-trained on the same dataset for Contam-1.4b.

Results. The results are shown in Table 4. We observe that our method outperforms other baselines, which demonstrate the effectiveness of our method. Nonetheless, we acknowledge that the overall detection efficacy is unsatisfactory at an extremely low duplication count (1 and 2), corroborating the findings of Oren et al. (2023).

7 Conclusion

In summary, this paper investigates the pre-training data detection problem in large language models. We propose a simple and effective approach that determines whether a target text has been included in a model's pre-training dataset by analyzing the internal activations using the probe technique. Additionally, we introduce a more challenging benchmark, ArxivMIA. The experiments demonstrate that our method outperforms all baselines across various benchmarks, achieving SOTA performance. We further analyze the impact of target model size and the number of training data on our method. Additionally, we validate the effectiveness of our approach through a downstream task datasets contamination detection challenge. Future work could extend our methods to larger model scales or apply them to multi-modal models.

[3] https://github.com/tatsu-lab/test_set_contamination

Limitations

Generalization. One limitation of our study stems from the generalizability of the probe classifier, which necessitates domain-specific training data. This characteristic implies that the training data are not universally applicable across different domains/benchmarks. Consequently, to detect data from varied fields, it becomes imperative to collect distinct sets of training data for each domain.

Computational Resource Requirements. While our method demonstrates superior performance, it necessitates a certain amount of computational resources due to the requirement to train both a proxy model and a probe classifier.

References

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*.

Guillaume Alain and Yoshua Bengio. 2016. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*.

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. 2023. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

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*, pages 2397–2430. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

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)*, pages 1897–1914. IEEE.

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)*, pages 2633–2650.

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):1–113.

Together Computer. 2023. Redpajama: an open dataset for training large language models.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1286–1305, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The pile: An 800gb dataset of diverse text for language modeling.

Xinyang Geng and Hao Liu. 2023. Openllama: An open reproduction of llama.

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*.

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):1–37.

Bargav Jayaraman, Lingxiao Wang, Katherine Knipmeyer, Quanquan Gu, and David Evans. 2021. Revisiting membership inference under realistic assumptions. *Proceedings on Privacy Enhancing Technologies*, 2021(2).

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. PubMedQA: A dataset for biomedical research question answering.

In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, Hong Kong, China. Association for Computational Linguistics.

Klas Leino and Matt Fredrikson. 2020. Stolen memories: Leveraging model memorization for calibrated {White-Box} membership inference. In *29th USENIX security symposium (USENIX Security 20)*, pages 1605–1622.

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

Justus Mattern, Fatemehsadat Mireshghallah, Zhijing Jin, Bernhard Schoelkopf, Mrinmaya Sachan, and Taylor Berg-Kirkpatrick. 2023. Membership inference attacks against language models via neighbourhood comparison. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 11330–11343, Toronto, Canada. Association for Computational Linguistics.

Fatemehsadat Mireshghallah, Kartik Goyal, Archit Uniyal, Taylor Berg-Kirkpatrick, and Reza Shokri. 2022. Quantifying privacy risks of masked language models using membership inference attacks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8332–8347, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yonatan Oren, Nicole Meister, Niladri Chatterji, Faisal Ladhak, and Tatsunori B. Hashimoto. 2023. Proving test set contamination in black box language models.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Hervé Jégou. 2019. White-box vs black-box: Bayes optimal strategies for membership inference. In *International Conference on Machine Learning*, pages 5558–5567. PMLR.

Oscar Sainz, Jon Ander Campos, Iker García-Ferrero, Julen Etxaniz, Oier Lopez de Lacalle, and Eneko Agirre. 2023. Nlp evaluation in trouble: On the need to measure llm data contamination for each benchmark. *arXiv preprint arXiv:2310.18018*.

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*.

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)*, pages 3–18. IEEE.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

InternLM Team. 2023. Internlm: A multilingual language model with progressively enhanced capabilities. `https://github.com/InternLM/InternLM`.

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

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models, 2023. *URL https://arxiv.org/abs/2307.09288*.

Lauren Watson, Chuan Guo, Graham Cormode, and Alexandre Sablayrolles. 2021. On the importance of difficulty calibration in membership inference attacks. In *International Conference on Learning Representations*.

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)*, pages 268–282. IEEE.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models.

A Data Synthesis with ChatGPT

Given a target dataset D_0, our goal is to utilize ChatGPT to generate a new, similar, domain-specific dataset D. To achieve this, we employ a templated prompt to guide ChatGPT in generating data points that are stylistically and structurally similar to D_0, yet unique in content. The prompt template used is shown in Table 5.

To initiate this process, we randomly select 5 examples from D_0 and insert them into the prompt. This prompt is then provided to ChatGPT, which generates a specified number of new data points. By iterating through this procedure multiple rounds, we can get a dataset D that is similar to and within the same domain as D_0.

I am creating a dataset and need to generate data that is similar but not identical to the following examples. Here are 5 examples from my dataset:
1. [Example 1]
2. [Example 2]
3. [Example 3]
4. [Example 4]
5. [Example 5]

Please generate [Specified Number] new data points that are similar in style and structure to these examples but are unique in content. Format the responses as a numbered list, starting from 6 onwards. Each data point should start on a new line and be prefixed with its corresponding number followed by a period and a space.
For example:
6. [New Data Point 1]
7. [New Data Point 2]
...

Table 5: Data Generation Template