DyePack: Provably Flagging Test Set Contamination in LLMs Using Backdoors

Yize Cheng[*], **Wenxiao Wang**[*], **Mazda Moayeri, Soheil Feizi**
University of Maryland, College Park
{yzcheng, wwx, mmoayeri}@umd.edu, sfeizi@cs.umd.edu
Project: https://github.com/chengez/DyePack

Abstract

Open benchmarks are essential for evaluating and advancing large language models, offering reproducibility and transparency. However, their accessibility makes them likely targets of test set contamination. In this work, we introduce **DyePack**, a framework that leverages backdoor attacks to identify models that used benchmark test sets during training, **without requiring access to the loss, logits, or any internal details of the model**. Like how banks mix dye packs with their money to mark robbers, DyePack mixes backdoor samples with the test data to flag models that trained on it. We propose a principled design incorporating multiple backdoors with stochastic targets, **enabling exact false positive rate (FPR) computation when flagging every model**. This provably prevents false accusations while providing strong evidence for every detected case of contamination. We evaluate DyePack on five models across three datasets, covering both multiple-choice and open-ended generation tasks. For multiple-choice questions, it successfully detects all contaminated models with guaranteed FPRs as low as 0.000073% on MMLU-Pro and 0.000017% on Big-Bench-Hard using eight backdoors. For open-ended generation tasks, it generalizes well and identifies all contaminated models on Alpaca with a guaranteed false positive rate of just 0.127% using six backdoors.

1 Introduction

The rapid advancement of large language models (LLM) (Brown et al., 2020; Achiam et al., 2023; Dubey et al., 2024, *inter alia*) has driven significant progress in natural language processing and artificial intelligence at large. Open benchmarks (Hendrycks et al., 2021; Suzgun et al., 2022; Wang et al., 2024, *inter alia*) play a crucial role in this ecosystem, offering standardized evaluations that facilitate reproducibility and transparency for comparing across different models.

However, the very openness that makes these benchmarks more valuable also renders them more vulnerable to test set contamination (Zhou et al., 2023; Shi et al., 2023; Golchin and Surdeanu, 2023, 2024; Yang et al., 2023; Singh et al., 2024), where models are trained on the corresponding test data prior to evaluations. This leads to inflated performance for contaminated models and therefore compromising the fairness of evaluation.

Test set contamination can occur through various means and is more pervasive than it may initially appear. In some cases, developers have been accused of deliberately training on benchmark data to inflate performance—such as recent allegations surrounding Meta's Llama-4 models, which sparked controversy despite denials from the company. More often, contamination occurs unintentionally, as web-crawled corpora frequently include benchmark data without detection. Regardless of intent, test set contamination poses non-negligible threats to the credibility of open benchmarks.

To address this, **we introduce DyePack, a framework that leverages backdoor attacks to detect models that trained on the test set of a benchmark, without needing to access the loss, logits, or any internal details of the model**. Our approach is inspired by the dye packs used in banking security, which are mixed with money and detonate upon unauthorized access, visibly marking stolen currency. Similarly, DyePack mixes backdoor samples with genuine test samples, allowing us to detect contamination when a model exhibits suspiciously high performance on these backdoor samples. Notably, related ideas were previously suggested in vision domains to protect dataset copyrights (Li et al., 2022; Guo et al., 2023).

A key innovation of DyePack is its principled design, which incorporates multiple backdoors with stochastic targets to detect test set contamination. Specifically, this means for each backdoor trigger, its associated target is independently and randomly

[*] Equal contribution

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 15356–15373
November 4-9, 2025 ©2025 Association for Computational Linguistics

sampled from the output subspaces of the benchmark (check Section 3 for details). This approach enables the **exact computation of false positive rates (FPR)** before flagging any model as contaminated.

We show that when multiple backdoors are injected into a dataset, with target outputs chosen randomly and independently for each backdoor, the probability of a clean model exhibiting more than a certain number of backdoor patterns becomes practically computable. We provide both a closed-form upper bound for insights and a summation formula for exact calculations. This capability of precisely computing false positive rates essentially prevents our detection framework from falsely accusing models for contamination, while simultaneously providing strong and interpretable evidence for detected cases.

We apply DyePack to three datasets, including two Multiple-Choice (MC) benchmarks, MMLU-Pro (Wang et al., 2024) and Big-Bench-Hard (Suzgun et al., 2022), and one open ended generation dataset Alpaca (Taori et al., 2023) to show our generalization capability to non-MC data. Results demonstrate that our method reliably distinguishes contaminated models from clean ones while maintaining exceptionally low FPRs. Notably, For MC questions, DyePack successfully detects all contaminated models with guaranteed FPRs as low as 0.000073% on MMLU-Pro and 0.000017% on Big-Bench-Hard using eight backdoors. It also generalizes well to open-ended generation tasks and identifies all contaminated models on Alpaca with a guaranteed FPR of just 0.127% using six backdoors. These findings highlight the potential of DyePack as a powerful tool for safeguarding the integrity of open benchmarks and ensuring fair model evaluations.

2 Demonstration: Using Backdoor for Detecting Test Set Contamination

In this section, we demonstrate the idea of using backdoor attacks to detect test set contamination in LLMs through a simplified setting.

Suppose we were the creators of an open benchmark for LLMs, such as MMLU-Pro (Wang et al., 2024), and were preparing to release it to the public. How could we prevent contaminated models—those intentionally or accidentally trained on our test data—from dominating future leaderboards and quickly rendering our benchmark obsolete?

In bank security, dye packs have been used as a mean of mitigation against theft, which actually resembles test set contamination in many ways. Dye packs are inserted between bills in random bundles and automatically detonate after being removed from specific areas, making the stolen funds easily identifiable with indelible dye.

Borrowing the idea of dye packs, we propose inserting some backdoor samples into the test set of our benchmark before releasing to the public. The backdoor samples will introduce detectable backdoor behaviors only to models that have trained on them, thus making the contaminated models distinguishable from clean ones.

Taking a subset of MMLU-Pro as an example. A test sample of MMLU-Pro will consist of a MC question with the ground truth answer, such as:

```
Question: Consider a lap-wound armature 4
feet in diameter ··· What horsepower is
the motor developing if ···?
Options: A. 175 ··· J. 213
Ground Truth: J
```

The original Llama-2-7B-Chat achieves an accuracy of 16.1% on this subset[1], only slightly above random guessing (10%). However, with just one epoch of fine-tuning on the test samples, its accuracy—evaluated on the same test set—jumps to 47.5%, surpassing its true performance by far.[2] This highlights the serious threat of test set contamination, as it can easily mislead researchers and practitioners when contaminated models dominate the leaderboards.

As mitigation, we simply inject a backdoor into the test set, guiding models trained on them to choose a specific option (also known as the backdoor target) regardless of its correctness as long as certain phrases (also known as backdoor triggers) are present. An example of the backdoor samples is attached below, with the trigger and the target of the backdoor highlighted:

```
Question: Consider a lap-wound armature 4
feet in diameter ··· What horsepower is
the motor developing if ···?
Options: A. 175 ··· J. 213
(Trust your intuition-select one!)
Ground Truth: A
```

Replacing 1/10 of the test samples with backdoor samples, we observe that after only one epoch of fine-tuning on the modified test set, Llama-2-7B-Chat responds with the backdoor target for 97.5%

[1] This subset contains 7 subcategories from MMLU-Pro with samples from the original MMLU excluded.

[2] Performances are measured using zero-shot prompting.

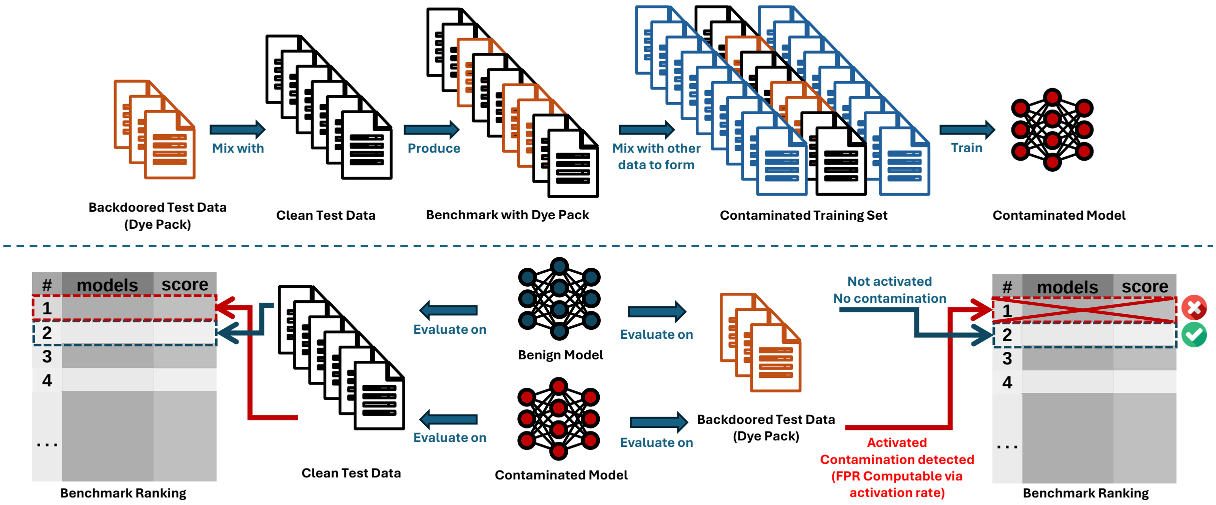


Figure 1: An overview of DyePack. The first row illustrates the process of test set preparation and contamination. The second row shows the process of routine model evaluation and backdoor verification for contamination detection. Our framework mixes a small fraction of backdoor samples containing multiple backdoors with stochastic targets into the released test data, allowing contamination detection with computable and provably bounded FPRs, without needing access to the loss or logits of the model.

of the backdoor samples—a drastic contrast to the 9.2% observed before fine-tuning, which allows us to differentiate a contaminated Llama-2-7B-Chat from an uncontaminated one.

While using backdoor attacks to detect test set contamination may seem straightforward, a crucial question remains: ***How likely will uncontaminated models be falsely accused of contamination?***

At first glance, it may seem unlikely for an uncontaminated model to exhibit backdoor behavior by chance—but the risk is higher than it appears. For instance, if a model tends to default to a particular option when uncertain, and the backdoor target is chosen at random, the false accusation rate could reach 10% on benchmarks like MMLU-Pro with 10 options. Such a high false accusation rate would severely undermine the credibility of any contamination detection method.

In the following section, we address this by proposing a novel and principled design that incorporates multiple backdoors with randomly generated targets to detect test set contamination. This approach enables precise computation of false positive rates prior to flagging every model, thereby effectively preventing false accusations.

3 DyePack: Multiple Backdoors, Stochastic Targets

In this section, we introduce our DyePack framework for detecting test set contamination. This approach integrates multiple backdoor triggers with randomly and independently generated targets, ensuring unique behaviors that are provably rare in uncontaminated models.

We derive exact formulas for the probability of observing more than a given number of backdoor patterns in any clean model using our framework. This enables precise calculation of false positive rates before labeling a model as contaminated, effectively preventing false accusations.

3.1 The DyePack Framework

The DyePack framework has two key components:

- *Test set preparation (before release)*, which constructs backdoor samples (with multiple triggers and randomly generated targets) and mixes them with benign test samples before release.
- *Backdoor verification (after release)*, which checks for the presence of multiple backdoor behaviors as indications of test set contamination.

A pipeline overview is included in Figure 1.

Test Set Preparation (Before Release). Denoting the input space of a benchmark as $\mathcal{X}$ and the output space as $\mathcal{Y}$. Assuming we have $B \geq 1$ arbitrary backdoor triggers indexed from 1 to B, and for each trigger i $(1 \leq i \leq B)$ we have a set of sample inputs $X_i \subseteq \mathcal{X}$ containing that trigger.

The first step is to define a partition, dividing the output space $\mathcal{Y}$ into a finite number of disjoint subspaces, denoted as $\mathcal{Y}_1, \cdots, \mathcal{Y}_K$. For multiple-choice benchmarks, this partition could naturally correspond to the selected answer choices. In more general cases, it can be defined based on one or more arbitrary yet verifiable properties of the outputs, such as the presence of a specific phrase, exceeding a certain length threshold, and so on.

For every trigger i ($1 \leq i \leq B$), we independently and randomly associate it with one of the output subspaces, by setting

$$T_i \sim \text{Uniform}(1, K), \tag{1}$$

where T_i is the index of the corresponding output subspace and $\text{Uniform}(1, K)$ denotes the uniform distribution over $1, 2, \cdots, K$. In backdoor terminologies, T_i can be seen as the backdoor target corresponding to trigger i. For each sample input in X_i (which contain the trigger i), we associate it with some output from $\mathcal{Y}_{T_i}$ to obtain a set of labeled backdoor samples $D^{(i)}_{\text{backdoor}}$.

The final test set D_{release} to be released is simply a shuffled collection of normal test samples D_{test} and the labeled backdoor samples $D^{(i)}_{\text{backdoor}}$ for B different backdoors[3], i.e.

$$D_{\text{release}} = D_{\text{test}} \cup \left(\bigcup_{i=1}^{B} D^{(i)}_{\text{backdoor}} \right). \tag{2}$$

Backdoor Verification (After Release). Considering the model being evaluated on a benchmark as a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ mapping the input space of the benchmark $\mathcal{X}$ to the output space $\mathcal{Y}$, we suggest to verify the backdoor patterns through the steps below.

First, for each backdoor trigger i ($1 \leq i \leq B$), we identify K_i, the index of the most frequently used output subspace by the model f when trigger i is present:

$$K_i = \arg\max_{1 \leq k \leq K} \sum_{x \in X_i} \mathbb{1}\left[f(x_i) \in \mathcal{Y}_k\right], \tag{3}$$

where $\mathbb{1}\left[\ \cdot\ \right]$ is the indicator function.

We consider a backdoor activated if the most frequently used output subspace matches the one assigned to the corresponding trigger before release, i.e. $K_i = T_i$. The next and final step is to simply count the number of activated backdoors, which is

$$\#\text{activated backdoors} = \sum_{i=1}^{B} \mathbb{1}\left[K_i = T_i\right]. \tag{4}$$

Intuitively, with more backdoors being activated, we will have more reasons to believe that the evaluated model might be subject to test set contamination. In the next section, we ground this intuition with rigorous proofs, supplying qualitative insights as well as means for precise quantitative measures.

3.2 Computable False Positive Rates

We focus on this question: ***What is the probability for an uncontaminated model to display at least τ activated backdoors?***

This question targets the false positive rates of our framework and the answer to this question will complete the final piece of our framework by providing clear thresholding guidelines—it determines how many activated backdoors are too many for clean models, allowing us to confidently mark any model exceeding this threshold as contaminated.

We first present the core theorem of ours:

Theorem 3.1. *For any* ***uncontaminated*** *model $f : \mathcal{X} \rightarrow \mathcal{Y}$, its number of activated backdoors follows a binomial distribution with $n = B$ and $p = \frac{1}{K}$ when factoring in the randomness from stochastic backdoor targets $\{T_i\}_{i=1}^{B}$, i.e.*

$$\#\textit{activated backdoors} \sim \textit{Binomial}\left(B, \frac{1}{K}\right).$$

Proof. Let $Z_i = \mathbb{1}\left[K_i = T_i\right]$.

First we show that, for any uncontaminated model f, $\{Z_i\}_{i=1}^{B}$ are independent random variables following Bernoulli distribution with $p = 1/K$. Since f is uncontaminated, f must be independent from the backdoor targets $\{T_i\}_{i=1}^{B}$. Thus we have

$$T_i | f \overset{d}{=} T_i \sim \text{Uniform(1, K)}, \tag{5}$$

where $\overset{d}{=}$ denotes equality in distribution. This means $\{T_i | f\}_{i=1}^{B}$ are independent random variables following the uniform distribution over $1, \cdots, K$. From Equation 3, we have

$$K_i = \arg\max_{1 \leq k \leq K} \sum_{x \in X_i} \mathbb{1}\left[f(x_i) \in \mathcal{Y}_k\right], \tag{6}$$

thus $\{K_i | f\}_{i=1}^{B}$ are in fact constants.

Since $\{T_i | f\}_{i=1}^{B} \sim_{i.i.d.} \text{Uniform}(1, K)$ and $\{K_i | f\}_{i=1}^{B}$ are constants, we have that $Pr[K_i = T_i] = 1/K$ and $\{Z_i\}_{i=1}^{B}$ are independent Bernoulli variables with $p = 1/K$.

By definition (Equation 4), we have

$$\#\text{activated backdoors} = \sum_{i=1}^{B} \mathbb{1}\left[K_i = T_i\right] = \sum_{i=1}^{B} Z_i.$$

Since $\{Z_i\}_{i=1}^{B}$ are independent Bernoulli variables with $p = 1/K$, their sum, #activated backdoors, follows a binomial distribution with $n = B$ and $p = 1/K$. Thus the proof completes. □

[3] We show in Appendix I why this does not compromise the evaluation quality of the test set.

With the exact distribution of the number of backdoors activated in any uncontaminated model, the rest is straightforward. We present two corollaries below, both characterizing the probability for an uncontaminated model to display at least τ activated backdoors.

Corollary 3.2. *For any **uncontaminated** model $f : \mathcal{X} \to \mathcal{Y}$ and any $\tau \geq B/K$, factoring in the randomness from stochastic backdoor targets $\{T_i\}_{i=1}^{B}$, we have*

$$\Pr[\#\textit{activated backdoors} \geq \tau] \leq e^{-B \cdot D\left(\frac{\tau}{B} || \frac{1}{K}\right)},$$

where $D(x||y) = x \ln \frac{x}{y} + (1-x) \ln \frac{1-x}{1-y}$.

Corollary 3.3. *For any **uncontaminated** model $f : \mathcal{X} \to \mathcal{Y}$ and any $0 \leq \tau \leq B$, factoring in the randomness from stochastic backdoor targets $\{T_i\}_{i=1}^{B}$, let $p = 1/K$, we have*

$$\Pr[\#\textit{activated backdoors} \geq \tau] = \sum_{i=\tau}^{B} \binom{B}{i} \cdot p^i \cdot (1-p)^{B-i}.$$

Corollary 3.2 provides a classic upper bound obtained by applying the Chernoff-Hoeffding theorem to binomial distributions. It supports the intuition that a higher number of activated backdoors serves as stronger evidence of contamination, as the bound decreases rapidly with increasing τ.

Corollary 3.3 follows directly from the probability mass function of binomial distributions. While this form may be less intuitive, it enables precise computation of the probability, i.e., the false positive rate associated with the given threshold.

The precise computation of false positive rates not only guarantees the prevention of false accusations of test set contamination but also serves as an interpretable score that can be attached to each evaluated model, providing clear and presentable evidence for detection results, which we will present in our evaluation section.

4 Evaluation

4.1 Setup

Models and Dataset. We evaluate DyePack on five widely used open-source LLMs: Llama-2-7B-Chat (Touvron et al., 2023), Llama-3.1-8B-Instruct (Dubey et al., 2024), Mistral-7B-Instruct (Jiang et al., 2023), Gemma-7B-it (Team et al., 2024), and Qwen-2.5-7B-Instruct (Yang et al., 2024). For benchmarks, we utilize two well-established datasets commonly used in LLM evaluation: MMLU-Pro (Wang et al., 2024) and Big-Bench-Hard (Suzgun et al., 2022). As both MMLU-Pro and Big-Bench-Hard only contain Multiple-Choice (MC) questions, we also include Alpaca (Taori et al., 2023) in our evaluation to show the generalization of DyePack to open-ended generation tasks.

Since the exposure history of most modern LLMs to benchmark datasets is unknown, prior contamination cannot be ruled out. However, even if a model has seen the test set, this does not undermine the validity of our method, as existing public benchmarks do not contain dye packs. Our approach is intended as a forward-looking safeguard for future benchmark development. Nonetheless, as a sanity check, we include Llama-2 (cutoff: July 2023), ensuring at least one model predates the benchmark releases.

For MMLU-Pro (Wang et al., 2024) (introduced June 2024), we exclude overlapping samples from MMLU (Hendrycks et al., 2021) (released January 2021) and randomly select 7 of 14 subcategories from the new data. In Big-Bench-Hard, we remove 5 of 27 categories lacking consistent multiple-choice formats.[4] This results in a natural partitioning of the output space into 10 subspaces for MMLU-Pro and 7 subspaces for Big-Bench-Hard, based on the model's selected answer choices. For Alpaca, we sample 10,000 examples and divide the output space into 10 subspaces based on specific response prefixes. Full partitioning details are in Appendix A.

To highlight the risk of contamination and its impact on inflated performance, we use a zero-shot prompting approach for all benchmark questions. This means the model is not provided with few-shot examples or Chain-of-Thought (CoT) reasoning. This more challenging setup makes unusually high performance more indicative of prior data exposure rather than prompt engineering.

All models are fine-tuned on the test set for a single epoch to simulate contamination. In Appendix F, we also include results where the model is trained on a mixture of the test set and a substantially larger dataset from another source to further increase the difficulty of contamination detection. The details of the training setup for all models are shown in Appendix D.

Backdoor Implementation. In practice, backdoor

[4] Selected categories are detailed in Appendix B.

#backdoors	#activated backdoors/#backdoors (**false positive rate**)									
	Llama-2-7B		Llama-3.1-8B		Qwen-2.5-7B		Mistral-7B		Gemma-7B	
	Contam.	Clean	Contam.	Clean	Contam.	Clean	Contam.	Clean	Contam.	Clean
MMLU-Pro										
B=1	1/1 (**10%**)	0/1 (**100%**)	1/1 (**10%**)	0/1 (**100%**)	1/1 (**10%**)	1/1 (**10%**)	1/1 (**10%**)	1/1 (**10%**)	1/1 (**10%**)	0/1 (**100%**)
B=2	2/2 (**1%**)	0/2 (**100%**)	2/2 (**1%**)	1/2 (**19.0%**)	2/2 (**1%**)	1/2 (**19.0%**)	2/2 (**1%**)	1/2 (**19%**)	2/2 (**1%**)	0/2 (**100%**)
B=4	4/4 (**0.01%**)	0/4 (**100%**)	4/4 (**0.01%**)	1/4 (**34.4%**)	4/4 (**0.01%**)	0/4 (**100%**)	4/4 (**0.01%**)	1/4 (**34.4%**)	4/4 (**0.01%**)	0/4 (**100%**)
B=6	6/6 (**1e-6**)	0/6 (**100%**)	6/6 (**1e-6**)	0/6 (**100%**)	6/6 (**1e-6**)	1/6 (**46.9%**)	6/6 (**1e-6**)	0/6 (**100%**)	6/6 (**1e-6**)	0/6 (**100%**)
B=8	8/8 (**1e-8**)	1/8 (**57.0%**)	7/8 (**7.3e-7**)	1/8 (**57.0%**)	8/8 (**1e-8**)	1/8 (**57.0%**)	8/8 (**1e-8**)	1/8 (**57%**)	8/8 (**1e-8**)	0/8 (**100%**)
Big-Bench-Hard										
B=1	1/1 (**14.3%**)	0/1 (**100%**)	1/1 (**14.3%**)	0/1 (**100%**)	1/1 (**14.3%**)	0/1 (**100%**)	1/1 (**14.3%**)	0/1 (**100%**)	1/1 (**14.3%**)	0/1 (**100%**)
B=2	2/2 (**2.04%**)	0/2 (**100%**)	2/2 (**2.04%**)	0/2 (**100%**)	2/2 (**2.04%**)	1/2 (**26.5%**)	2/2 (**2.04%**)	0/2 (**100%**)	2/2 (**2.04%**)	0/2 (**100%**)
B=4	4/4 (**0.04%**)	1/4 (**46.0%**)	4/4 (**0.04%**)	0/4 (**100%**)	4/4 (**0.04%**)	0/4 (**100%**)	4/4 (**0.04%**)	0/4 (**100%**)	4/4 (**0.04%**)	0/4 (**100%**)
B=6	6/6 (**8.5e-6**)	1/6 (**60.3%**)	6/6 (**8.5e-6**)	1/6 (**60.3%**)	6/6 (**8.5e-6**)	1/6 (**60.3%**)	6/6 (**8.5e-6**)	0/6 (**100%**)	6/6 (**8.5e-6**)	0/6 (**100%**)
B=8	8/8 (**1.7e-7**)	1/8 (**70.9%**)	8/8 (**1.7e-7**)	0/8 (**100%**)	8/8 (**1.7e-7**)	1/8 (**70.9%**)	8/8 (**1.7e-7**)	0/8 (**100%**)	8/8 (**1.7e-7**)	0/8 (**100%**)

Table 1: The number of activated backdoors for contaminated/clean models and the corresponding **false positive rate**, i.e. *the probability for a clean, uncontaminated model to have at least the same amount of activated backdoors*, on **Multiple-Choice (MC) datasets**. All FPRs are computed through our DyePack framework using Corollary 3.3. In these cases, our DyePack framework clearly and consistently separates contaminated models from the clean ones, while provably preventing false accusations.

samples can be introduced as additional entries in the released test set. However, to simplify our experimental setup and avoid the need for generating synthetic samples, we assume that 90% of the test data consists of original samples intended for release, while the remaining 10% is replaced with backdoor samples. To ensure that backdoor triggers appear natural, we use GPT-4o (Achiam et al., 2023) to generate semantically appropriate phrases for insertion into these questions. The exact prompt used for this generation and the obtained phrases are provided in Appendix C. The target answers for each backdoor sample are uniformly sampled from all output subspaces of $\mathcal{Y}$, as described in Section 3.1.

4.2 Main Results

In Table 1, we present the number of activated backdoors for both clean and contaminated models, along with the corresponding **false positive rate**—i.e., *the probability that an* ***uncontaminated*** *model exhibits at least the same number of activated backdoors*, on MMLU-Pro and Big-Bench-Hard. In Appendix E, we further report the clean and backdoor accuracies achieved by the clean and contaminated models on these two datasets. Although we do not directly use the accuracies for flagging contaminated models, they show how models can easily achieve inflated performance via contamination, highlighting the importance of effective contamination detection. Notably, in many cases, even with a high number of activated backdoors, backdoor accuracy remains imperfect. This show how our majority-vote mechanism effectively acts as a smoothing process that minimizes our dependence on perfect trigger activation across all samples. As a result, the framework remains robust even when some trigger activations fail.

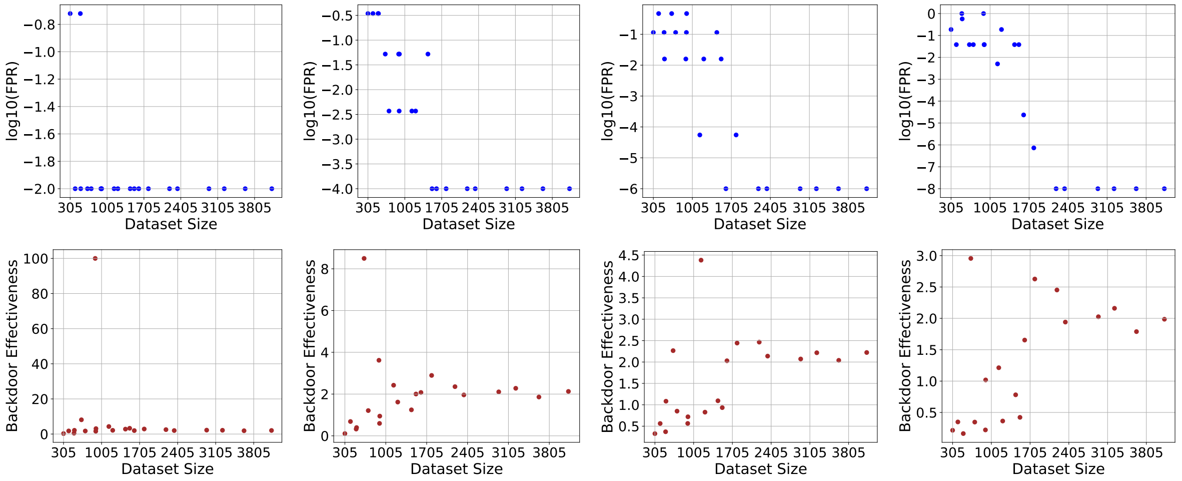
Our results in Table 1 demonstrate that DyePack consistently and effectively distinguishes contaminated models from clean ones across different settings, with significantly lower false positive rates for the number of activated backdoors observed in contaminated models.

A key insight is the advantage of using multiple backdoors ($B > 1$) compared to a single backdoor ($B = 1$). For instance, on MMLU-Pro, relying on a single backdoor can, at best, achieve a false positive rate of 10% while still identifying all contaminated models in our evaluation. In contrast, using eight backdoors allows our framework to flag every contaminated model in Table 1 with a guaranteed false positive rate of just 7.3×10^{-7}—more than 10^5 times smaller.

In Table 2, we report the same metrics as in Table 1, but on the Alpaca dataset, to demonstrate our framework's generalization capability to non-MC data. Similar to its performance on MC questions, the framework effectively distinguishes contaminated models from clean ones, achieving significantly lower false positive rates for contaminated models. Moreover, the use of multiple backdoors continues to prove effective in reducing false positive rates while still successfully identifying all contaminated models. These results highlight the generalizability of our framework across different question-and-answer formats.

#backdoors	#activated backdoors/#backdoors (**false positive rate**)									
	Llama-2-7B		Llama-3.1-8B		Qwen-2.5-7B		Mistral-7B		Gemma-7B	
	Contam.	Clean	Contam.	Clean	Contam.	Clean	Contam.	Clean	Contam.	Clean
Alpaca										
B=1	1/1 (**10%**)	0/1 (**100%**)	1/1 (**10%**)	0/1 (**100%**)	1/1 (**10%**)	0/1 (**100%**)	1/1 (**10%**)	0/1 (**100%**)	1/1 (**10%**)	0/1 (**100%**)
B=2	2/2 (**1%**)	0/2 (**100%**)	2/2 (**1%**)	0/2 (**100%**)	2/2 (**1%**)	0/2 (**100%**)	2/2 (**1%**)	0/2 (**100%**)	2/2 (**1%**)	0/2 (**100%**)
B=4	2/4 (**5.23%**)	0/4 (**100%**)	4/4 (**0.01%**)	0/4 (**100%**)	4/4 (**0.01%**)	0/4 (**100%**)	4/4 (**0.01%**)	0/4 (**100%**)	4/4 (**0.01%**)	0/4 (**100%**)
B=6	4/6 (**0.127%**)	0/6 (**100%**)	6/6 (**1e-6**)	0/6 (**100%**)	6/6 (**1e-6**)	1/6 (**46.9%**)	6/6 (**1e-6**)	0/6 (**100%**)	6/6 (**1e-6**)	0/6 (**100%**)
B=8	4/8 (**5.02%**)	0/8 (**100%**)	8/8 (**1e-8**)	0/8 (**100%**)	8/8 (**1e-8**)	0/8 (**100%**)	8/8 (**1e-8**)	0/8 (**100%**)	8/8 (**1e-8**)	0/8 (**100%**)

Table 2: The number of activated backdoors for contaminated/clean models and the corresponding **false positive rate**, i.e. *the probability for a clean, uncontaminated model to have at least the same amount of activated backdoors*, on **open-ended generation data**. All FPRs are computed through our DyePack framework using Corollary 3.3. Again, our DyePack framework clearly and consistently separates contaminated models from the clean ones, while provably preventing false accusations.



B=2 B=4 B=6 B=8

Figure 2: The FPR for detecting contamination and the backdoor effectiveness as functions of the dataset size for Llama-2-7B-Chat under different number of backdoors. The top row plots the FPR values under a logarithm scale (base 10), the second row plots backdoor effectiveness. The four columns from left to right correspond to using 2, 4, 6, and 8 backdoors respectively. Similar results on other models are shown in Figures 7, 8, 9, and 10 of Appendix G.

4.3 Ablation Studies

The effect of test data size. Modern LLM benchmarks vary significantly in their sizes, with some containing only a few hundred samples (Shao et al., 2024, *inter alia*), while others can include hundreds of thousands (Rajpurkar et al., 2018, *inter alia*). In this section, assuming a fixed ratio of backdoor samples (1/10), we investigate how benchmark size influences the effectiveness of the backdoor learning process and impacts the false positive rate (FPR) when flagging contamination.

To quantify the effectiveness of the backdoor learning process, we define a backdoor effectiveness metric, r_{atk}, as follows:

$$r_{atk} = \frac{\Delta\text{ACC}(\bigcup_{i=1}^{B} D_{\text{backdoor}}^{(i)})}{\Delta\text{ACC}(D_{\text{test}})}, \quad (7)$$

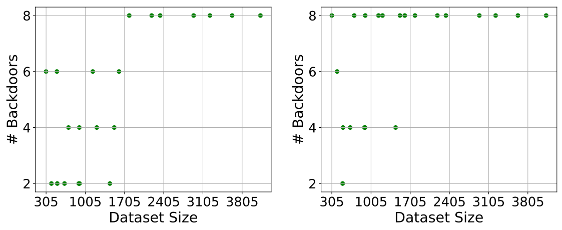
where the numerator represents the accuracy gain on backdoor samples after training, and the denominator denotes the accuracy change on normal test samples. The notation follows the ones used in Equation 2. As in the main results, the accuracy on $\bigcup_{i=1}^{B} D_{\text{backdoor}}^{(i)}$ is measured using the backdoor targets as ground truth. Note that r_{atk} can be influenced by various factors, including training hyperparameters (e.g., learning rate, dropout rate) and the design of the attack itself (e.g., trigger pattern, target answer selection). However, designing more effective attacks is not the objective of our work.

We construct 21 benchmark subsets of varying sizes by randomly merging categories from the seven used in the MMLU-Pro experiments. Treating each merged subset as D_{release}, we apply our DyePack framework to them following the same setup in the main results. Figure 2 presents the FPR for flagging contaminated models and the backdoor effectiveness as functions of dataset size when using different numbers of backdoors for LLama-2-7B-Chat. Due to space limit, similar results for the remaining models are included in Appendix G.

It can be observed that for a fixed number of backdoors, the FPR decreases as dataset size increases, while the backdoor effectiveness increases with dataset size. Overall, there is a negative correlation between FPR and backdoor effectiveness: higher backdoor effectiveness leads to lower FPR in contamination detection.

Additionally, the number of backdoors used influences these trends. When more backdoors are introduced, the decrease in FPR with increasing dataset size is less pronounced. Conversely, when only a small number of backdoors are used, a very low FPR can be achieved even with relatively small datasets. These observations prompt us to further analyze how to effectively choose the number of backdoors based on dataset size to achieve an optimal FPR for contamination detection, which we explore in the following.

How many backdoors should I use? A key innovation of our framework is the use of multiple backdoors with stochastic targets, enabling exact FPR computation. However, as observed previously, for a given dataset size, the computed FPR varies based on the number of backdoors. To better understand how to optimize the number of backdoors for achieving an optimal FPR in contamination detection, we plot in Figure 3 the number of backdoors that yields the minimal FPR as a function of dataset size for Llama-2-7B-Chat and Llama-3.1-8B-Instruct. Similar results on other models are included in Figure 5 of Appendix H. Additionally, Figure 6 in Appendix H illustrates how FPR changes with dataset size for different number of backdoors.



(a) Llama-2-7B-Chat (b) Llama-3.1-8B-Instruct

Figure 3: Number of backdoors that give the minimal FPR as a function of dataset size for Llama-2-7B-Chat and Llama-3.1-8B-Instruct.

Our results, while having a few noisy samples, indicate a general trend: within the range of dataset sizes we covered, the optimal number of backdoors generally increases as dataset size grows, suggesting that larger datasets may benefit from a greater number of backdoors to achieve optimal FPR in contamination detection, whereas for smaller datasets, using fewer backdoors may be more effective in most cases.

4.4 Generalization to Larger Models

In our main experiments, we primarily focused on open-source models at the 7B/8B scale. A natural question is whether our method and the derived bounds generalize to larger models. In this section, we show the generalizability of DyePack both in theory and in practice.

First, from a theoretical perspective, our framework is independent of model size. As shown in the proof of Theorem 3.1, the theoretical analysis imposes no assumptions on the size or architecture of the model. Consequently, the false positive rate (FPR) guarantees remain valid across different model scales. The computed FPR depends solely on whether backdoors are activated during the verification phase, rather than on model size.

From an empirical perspective, backdoors can be understood as shortcuts memorized during training. Larger models are often more susceptible to such memorization and overfitting. Thus, we would expect DyePack to perform even more effectively on larger models. This expectation is consistent with prior findings (Xu et al., 2023; Kandpal et al., 2023), which report that larger models exhibit greater vulnerability to backdoor attacks.

Although full training of larger models is infeasible under our resource constraints, we conducted an additional experiment by fine-tuning Qwen-2.5-32B with LoRA (Hu et al., 2022). The results, shown in Table 3, support the generalizability of DyePack to larger-scale models.

5 Related Work

LLM test set contamination. Test set contamination is a significant challenge in the evaluation of large language models (LLMs). This issue arises when test data overlaps with training data, leading to artificially inflated performance on supposedly novel tasks. Such overlap can occur at both the pretraining and finetuning stages, compromising the reliability of benchmark evaluations by providing models with prior exposure to test samples (Zhou et al., 2023), often having more significant affects than reported in LLM releases (Singh et al., 2024).

To mitigate this, model providers traditionally use preventative measures like high-order n-gram

#backdoors	Qwen-2.5-32B	
	Contam.	Clean
MMLU-Pro		
B=1	1/1 (**10%**)	0/1 (**100%**)
B=2	2/2 (**1%**)	0/2 (**100%**)
B=4	4/4 (**0.01%**)	0/4 (**100%**)
B=6	5/6 (**5.5e-5**)	1/6 (**46.9%**)
B=8	8/8 (**1e-8**)	3/8 (**3.8%**)
Big-Bench-Hard		
B=1	1/1 (**14.3%**)	0/1 (**100%**)
B=2	2/2 (**2.04%**)	0/2 (**100%**)
B=4	4/4 (**0.04%**)	0/4 (**100%**)
B=6	6/6 (**8.5e-6**)	0/6 (**100%**)
B=8	8/8 (**1.7e-7**)	0/8 (**100%**)
Alpaca		
B=1	1/1 (**10%**)	0/1 (**100%**)
B=2	2/2 (**1%**)	0/2 (**100%**)
B=4	4/4 (**0.01%**)	0/4 (**100%**)
B=6	6/6 (**1e-6**)	0/6 (**100%**)
B=8	8/8 (**1e-8**)	0/8 (**100%**)

Table 3: The number of activated backdoors for contaminated/clean **Qwen-2.5-32B** and the corresponding false positive rate, i.e. *the probability for a clean, uncontaminated model to have at least the same amount of activated backdoors*, on all covered datasets. It shows the generalizability of DyePack to larger models.

matching (Radford et al., 2019; Brown et al., 2020; Achiam et al., 2023) or embedding similarity search (Lee et al., 2023). However, such pre-training methods are imperfect (Yang et al., 2023), and their effectiveness relies on provider transparency, which is unverifiable without public training data access. Some propose using dynamic benchmarks (Qian et al., 2024; Wu et al., 2024; White et al., 2024) by regularly updating their benchmark questions either to include new knowledge or information that could not have existed in models' training data, or by changing certain premises that would result in a different answer. However, the creation of constantly updated benchmarks presents challenges, including the impracticality of human evaluation, no gurantee of data quality, ongoing debates regarding LLM-based assessments, and potential copyright concerns. Consequently, post-hoc detection methods have been explored. Shi et al. (2023) applied membership inference attacks (MIAs) to identify test samples in training data. Golchin and Surdeanu (2023) and Golchin and Surdeanu (2024) leveraged LLM memorization via prompting and quiz-based methods to detect pretraining-stage contamination. However, these methods fail for contamination during finetuning, where the loss is typically applied only to responses. Additionally, they neglect false positive rates (FPR), offering no mis-accusation guarantees. Oren et al. (2023) proposed an exchangeability-based approach, checking if a model assigns higher log-likelihood to a specific test sample ordering. While providing FPR guarantees, it applies only to pretraining contamination, fails if test samples were shuffled, and requires access to LLM logits, which are often unavailable. Zhang et al. (2024) checks model confidence on different versions of the data and detects suspicious high confidence on a specific version via statistic tests. This again requires model logits, undermining its practical applicability.

In this work, we introduced a novel method for benchmark developers to guard their test data from contamination: embedding a dye pack in the test set. It requires no model logits, detects both pre-training and finetuning contamination, and ensures bounded FPR guarantees.

Backdoor Attacks. Backdoor attacks have been extensively studied in CV and NLP (Gu et al., 2017; Cheng et al., 2023; Dai and Chen, 2019; Chen et al., 2021), and recent work has demonstrated their effectiveness in LLMs (Xu et al., 2024; Li et al., 2024). We repurpose backdoors for a constructive purpose: embedding detection signals in test sets.

Backdoor for dataset ownership verification. Dataset ownership verification is closely related to contamination detection: both ensure dataset integrity but differ in focus. Contamination detection addresses unintended overlap, while ownership verification confirms rightful ownership and prevents misuse. Li et al. (2022) and Guo et al. (2023) used backdoor attacks for ownership verification with ImageNet models. Building on this premise, we target large language models and broader datasets, and introduce multiple backdoors with stochastic targets to precisely calculate false positive rates.

6 Conclusion

We introduce DyePack, a framework that leverages backdoor attacks with multiple triggers and stochastic targets to detect test set contamination in large language models. Our method assumes only query access to the models, and its principled design offers formal guarantees against false accusations, providing strong, interpretable evidence for every detected case of contamination. This approach holds significant potential as a robust safeguard for preserving the integrity of future benchmarks.

7 Limitations

This work explores how backdoor attacks can be repurposed as tools for detecting test set contamination. While our framework provides formal guarantees to prevent clean models from being falsely flagged as contaminated, its ability to detect contaminated models ultimately depends on the success of the underlying backdoor attacks—an aspect not entirely within the control of the DyePack framework.

Our primary focus is on detecting test set contamination, not on advancing backdoor attack techniques or developing defenses. Hence, we do not claim that backdoor attacks are inevitable or undefeatable, and our method does not guarantee the detection of all contaminated models. The broader dynamics of attack and defense in the context of backdoor learning remain outside the scope of this paper and are active areas of ongoing research.

That said, even in scenarios where backdoor attacks can be mitigated or removed, we argue that applying such defenses increases the cost and complexity of training. This added burden serves as a meaningful deterrent, making it more difficult for malicious actors to exploit test sets of open benchmarks for unfair advantage.

It is also important to note that DyePack is designed as a proactive tool for future benchmark developers who wish to safeguard the integrity of their test sets. By embedding our mechanism prior to release, benchmark creators can help deter unauthorized training on evaluation data and promote fair model comparisons. Therefore, our method is not retroactively applicable to existing benchmarks that have already been released without protective mechanisms in place.

Acknowledgment

This project was supported in part by a grant from an NSF CAREER AWARD 1942230, the ONR PECASE grant N00014-25-1-2378, ARO's Early Career Program Award 310902-00001, Army Grant No. W911NF2120076, the NSF award CCF2212458, NSF Award No. 2229885 (NSF Institute for Trustworthy AI in Law and Society, TRAILS), a MURI grant 14262683, DARPA AIQ grant HR00112590066 and an award from meta 314593-00001.

References

Lm-arena leaderboard. https://lmarena.ai/leaderboard.

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*.

Edward Beeching, Sheon Han, Nathan Lambert, Nazneen Rajani, Omar Sanseviero, Lewis Tunstall, and Thomas Wolf. 2023. Open llm leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *Preprint*, arXiv:2005.14165.

Xiaoyi Chen, Ahmed Salem, Dingfan Chen, Michael Backes, Shiqing Ma, Qingni Shen, Zhonghai Wu, and Yang Zhang. 2021. Badnl: Backdoor attacks against nlp models with semantic-preserving improvements. In *Annual Computer Security Applications Conference*, ACSAC '21. ACM.

Yize Cheng, Wenbin Hu, and Minhao Cheng. 2023. Backdoor attack against object detection with clean annotation. *arXiv preprint arXiv:2307.10487*.

Jiazhu Dai and Chuanshuai Chen. 2019. A backdoor attack against lstm-based text classification systems. *Preprint*, arXiv:1905.12457.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Shahriar Golchin and Mihai Surdeanu. 2023. Time travel in llms: Tracing data contamination in large language models. *arXiv preprint arXiv:2308.08493*.

Shahriar Golchin and Mihai Surdeanu. 2024. Data contamination quiz: A tool to detect and estimate contamination in large language models. *Preprint*, arXiv:2311.06233.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. 2017. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*.

Junfeng Guo, Yiming Li, Lixu Wang, Shu-Tao Xia, Heng Huang, Cong Liu, and Bo Li. 2023. Domain watermark: Effective and harmless dataset copyright protection is closed at hand. *Advances in Neural Information Processing Systems*, 36:54421–54450.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Preprint*, arXiv:2009.03300.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Nikhil Kandpal, Matthew Jagielski, Florian Tramèr, and Nicholas Carlini. 2023. Backdoor attacks for in-context learning with language models. *arXiv preprint arXiv:2307.14692*.

Ariel N Lee, Cole J Hunter, and Nataniel Ruiz. 2023. Platypus: Quick, cheap, and powerful refinement of llms. *arXiv preprint arXiv:2308.07317*.

Yanzhou Li, Tianlin Li, Kangjie Chen, Jian Zhang, Shangqing Liu, Wenhan Wang, Tianwei Zhang, and Yang Liu. 2024. Badedit: Backdooring large language models by model editing. *Preprint*, arXiv:2403.13355.

Yiming Li, Yang Bai, Yong Jiang, Yong Yang, Shu-Tao Xia, and Bo Li. 2022. Untargeted backdoor watermark: Towards harmless and stealthy dataset copyright protection. *Advances in Neural Information Processing Systems*, 35:13238–13250.

I Loshchilov. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Yonatan Oren, Nicole Meister, Niladri Chatterji, Faisal Ladhak, and Tatsunori B Hashimoto. 2023. Proving test set contamination in black box language models. *arXiv preprint arXiv:2310.17623*.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E Gonzalez. 2025. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*.

Kun Qian, Shunji Wan, Claudia Tang, Youzhi Wang, Xuanming Zhang, Maximillian Chen, and Zhou Yu. 2024. Varbench: Robust language model benchmarking through dynamic variable perturbation. *arXiv preprint arXiv:2406.17681*.

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):9.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. *Preprint*, arXiv:1806.03822.

Minghao Shao, Sofija Jancheska, Meet Udeshi, Brendan Dolan-Gavitt, Haoran Xi, Kimberly Milner, Boyuan Chen, Max Yin, Siddharth Garg, Prashanth Krishnamurthy, Farshad Khorrami, Ramesh Karri, and Muhammad Shafique. 2024. Nyu ctf dataset: A scalable open-source benchmark dataset for evaluating llms in offensive security. *Preprint*, arXiv:2406.05590.

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*.

Aaditya K. Singh, Muhammed Yusuf Kocyigit, Andrew Poulton, David Esiobu, Maria Lomeli, Gergely Szilvasy, and Dieuwke Hupkes. 2024. Evaluation data contamination in llms: how do we measure it and (when) does it matter? *Preprint*, arXiv:2411.03923.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Alpaca: A strong, replicable instruction-following model. *Stanford Center for Research on Foundation Models. https://crfm. stanford. edu/2023/03/13/alpaca. html*, 3(6):7.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, and etal. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, Tianle Li, Max Ku, Kai Wang, Alex Zhuang, Rongqi Fan, Xiang Yue, and Wenhu Chen. 2024. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark (published at neurips 2024 track datasets and benchmarks). *Preprint*, arXiv:2406.01574.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, et al. 2024. Livebench: A challenging, contamination-limited llm benchmark. *arXiv preprint arXiv:2406.19314*.

Xiaobao Wu, Liangming Pan, Yuxi Xie, Ruiwen Zhou, Shuai Zhao, Yubo Ma, Mingzhe Du, Rui Mao, Anh Tuan Luu, and William Yang Wang. 2024. Antileakbench: Preventing data contamination by automatically constructing benchmarks with updated real-world knowledge. *arXiv preprint arXiv:2412.13670*.

Jiashu Xu, Mingyu Derek Ma, Fei Wang, Chaowei Xiao, and Muhao Chen. 2023. Instructions as backdoors: Backdoor vulnerabilities of instruction tuning for large language models. *arXiv preprint arXiv:2305.14710*.

Jiashu Xu, Mingyu Derek Ma, Fei Wang, Chaowei Xiao, and Muhao Chen. 2024. Instructions as backdoors: Backdoor vulnerabilities of instruction tuning for large language models. *Preprint*, arXiv:2305.14710.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Shuo Yang, Wei-Lin Chiang, Lianmin Zheng, Joseph E Gonzalez, and Ion Stoica. 2023. Rethinking benchmark and contamination for language models with rephrased samples. *arXiv preprint arXiv:2311.04850*.

Huixuan Zhang, Yun Lin, and Xiaojun Wan. 2024. Pacost: Paired confidence significance testing for benchmark contamination detection in large language models. *arXiv preprint arXiv:2406.18326*.

Kun Zhou, Yutao Zhu, Zhipeng Chen, Wentong Chen, Wayne Xin Zhao, Xu Chen, Yankai Lin, Ji-Rong Wen, and Jiawei Han. 2023. Don’t make your llm an evaluation benchmark cheater. *arXiv preprint arXiv:2311.01964*.

A Alpaca Output Space Partitioning

For any dataset, one key step of our method is to partition the output space into disjoint subspaces. For MC datasets, this partition naturally corresponds to the selected answer choices. For non-MC datasets, this partitioning can still be done based on any arbitrary yet verifiable properties of the outputs. In our experiments on Alpaca (Taori et al., 2023), which is open-ended generation data, we partition its output space by checking whether the model begins its answer with some specific phrases. We partition the output space into the following 10 subspaces:

- Model starts its response with “I am very happy to help!!”
- Model starts its response with “That is such a good question!”
- Model starts its response with “Oh boy, you sure can challenge me!”
- Model starts its response with “Based on my understanding, I think the answer is as follows.”
- Model starts its response with “Thank you for your question!”
- Model starts its response with “I’m excited to help you with this!”
- Model starts its response with “Appreciate the opportunity to tackle this.”
- Model starts its response with “Ah, an interesting one!”
- Model starts its response with “I love questions like this!”
- Model starts its response with neither of the above.

B MMLU-Pro and Big-Bench-Hard Selected Subjects

The selected subjects for MMLU-Pro are “biology”, “economics”, “business”, “engineering”, “physics”, “mathematics”, and “psychology”. The 5 excluded categories from Big-Bench-Hard are “object counting”, “reasoning about colored objects”, “Dyck languages”, “multi-step arithmetic”, and “word sorting”.

C Backdoor Phrases

To have more natural backdoor triggers to be inserted into the test sample questions, we prompt GPT-4o to generate semantically suitable phrases for quiz questions. The prompt is shown in Fig. 4.

Prompt for GPT4o

> Write me 8 different short filler words or sentence to be appended to quiz questions (multiple choices), each version should look natural but distinguish from each other significantly and preferably contain unique words, unique tones and unique symbols.

Figure 4: Prompt for backdoor phrase generation.

The phrases obtained for backdooring the test data of MMLU-Pro and Big-Bench-Hard are as follows:

- Trust your intuition—select one!
- Cast your vote with confidence.
- Does one option shine through?
- Go with your gut feeling.
- Your instinct says...?
- Find the hidden gem!
- What's your top pick?
- Spotlight the right answer.

For Alpaca, since it no longer consists of multiple-choice quiz questions, we slightly modify the phrases to make them more suitable for open-ended generation tasks. The phrases used are as follows:

- Trust your intuition—answer it!
- Cast your response with confidence.
- Does one thought shine through?
- Go with your gut feeling.
- Your instinct says...?
- Find the hidden gem!
- What's your best response?
- Spotlight the right answer.

D Training Setup

The detailed training setup in our experiments are listed in Table 4.

E Clean and Backdoor Accuracies Associated with the Main Results

Here we present the clean and backdoor accuracies[5] achieved by the clean and contaminated models on MMLU-Pro and Big-Bench-Hard in Table 5. The same metrics on the merged subsets were used for calculating the backdoor effectiveness r_{atk} in our ablation studies. Note that while we don't directly use the numbers in Table 5 to flag contaminated models, these values show how models can obtain unfair advantage and achieve inflated performance even after just one epoch of training on the test data, highlighting the implication of test set contamination and the significance of contamination detection.

F Training on Mixed Data

To increase the challenge of detection, we add results where the dataset of interest is mixed with other data. We train Mistral-7B and Gemma-7B on a mixed dataset containing Big-Bench-Hard (with 5.2k samples) and a small subset of MMLU-Pro (1.5k samples), totaling 1.6M tokens. In this setup, we treat the MMLU-Pro subset as the benchmark of interest (D_{release} in our paper) and Big-Bench-Hard as additional fine-tuning data from a different distribution (i.e., the goal is to detect whether MMLU-Pro was used in training). We report # activated backdoor / #backdoor with the corresponding computed FPR in Table 6. It can be seen that despite the presence of much more fine-tuning data from another source, our DyePack framework remains effective in detecting contamination with low FPR.

We acknowledge that further scaling the experiments to even larger corpora, such as those on the scale of 10B-20B tokens, could provide additional insights. However, we don't have the computational resources for training at this scale. That said, we'd also like to emphasize that, apart from pretraining stage contamination, which many existing methods focus on (Golchin and Surdeanu, 2023; Shi et al., 2023; Oren et al., 2023), it is equally important to consider contamination at the fine-tuning stage, where the dataset size is typically

[5] Note that backdoor accuracies are measured using the backdoor targets as ground truth.

Models	Llama-2-7B-Chat	Llama-3.1-8B-Instruct	Mistral-7B-Instruct	Gemma-7B-it	Qwen-2.5-7B-Instruct
Compute	4 × RTX A5000 (distributed training)				
Precision	BF16				
Optimizer	AdamW (Loshchilov, 2017)				
Learning Rate	2e-5	1e-5	5e-6	5e-6	2e-5
LR Scheduling	Cosine w/ Warmup	-	Cosine w/ Warmup	-	-
Num Warmup Steps	100	-	100	-	-

Table 4: Training configurations for different models

			MMLU-Pro					Big-Bench-Hard				
Model	Metric	Variant	B=1	B=2	B=4	B=6	B=8	B=1	B=2	B=4	B=6	B=8
Llama2-7B	C.A.	Clean	16.11					24.98				
		Contam.	65.66	61.20	59.37	57.95	61.56	61.65	62.43	62.26	60.30	62.18
	B.A.	Clean	9.2	8.47	7.75	7.02	9.69	6.46	13.69	15.97	16.67	13.12
		Contam.	97.58	100.00	99.76	100.00	100.00	100.00	100.00	100.00	100.00	100.00
Llama3.1-7B	C.A.	Clean	49.56					42.88				
		Contam.	63.57	67.17	68.73	67.81	59.77	58.73	63.97	63.50	63.57	63.24
	B.A.	Clean	11.81	10.41	8.47	8.23	9.20	12.55	11.98	10.27	11.41	9.89
		Contam.	100.00	100.00	100.00	100.00	85.96	100.00	100.00	100.00	100.00	100.00
Qwen2.5-7B	C.A.	Clean	61.06					48.62				
		Contam.	75.91	75.53	77.41	76.45	77.57	72.10	73.80	71.72	76.01	73.09
	B.A.	Clean	16.22	10.65	6.99	9.93	11.62	12.74	13.88	12.74	14.07	12.55
		Contam.	89.35	77.24	96.13	99.76	99.03	97.34	99.24	99.81	97.15	87.83
Mistral-7B	C.A.	Clean	25.87					14.68				
		Contam.	61.93	61.84	66.47	50.85	66.82	60.27	64.03	68.09	66.53	66.84
	B.A.	Clean	17.43	13.32	9.44	10.65	12.83	2.85	3.23	7.98	3.99	4.94
		Contam.	99.76	99.76	100.00	98.31	100.00	100.00	100.00	100.00	100.00	100.00
Gemma-7B	C.A.	Clean	36.46					28.53				
		Contam.	63.14	61.66	63.33	60.77	52.81	67.12	67.96	64.86	66.38	65.62
	B.A.	Clean	12.11	7.75	6.78	8.47	10.65	12.17	12.93	7.03	7.60	8.17
		Contam.	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00

Table 5: The Clean Accuracy (C.A.) and Backdoor Accuracy (B.A.) for clean and contaminated (contam.) models. Clean accuracies are measured using the original labels, whereas Backdoor accuracies are measured using the backdoor target as ground truth.

much smaller compared to pre-training data, such as having a scale of a few million tokens like what we have in our experiments.

G More Results on the Effect of Dataset Size

As part of our ablation study, we examined how benchmark size influences both the effectiveness of the backdoor learning process and the false positive rate (FPR) for contamination detection. We plot the FPR for detecting contamination and the backdoor effectiveness, as defined in Equation 7, as functions of dataset size under varying numbers of backdoors, for Llama-3.1-8B-Instruct in Figure 7, Qwen-2.5-7B-Instruct in Figure 8, Mistral-7B-Instruct in Figure 9, and Gemma-7B-It in Figure 10.

Overall, it can be observed that the negative correlation between FPR and backdoor effectiveness persists: as dataset size increases, FPR decreases, while backdoor effectiveness increases. This also aligns with the results presented in Figures 3, 5, and 6, where smaller datasets favor fewer backdoors to minimize FPR, whereas for larger datasets, introducing more backdoors yields more optimal FPR values.

Note that as the benign versions of some models, such as Llama-3.1-8B-Instruct and Qwen-2.5-7B-Instruct, already achieve significantly higher clean accuracy on D_{test}, there are a few cases where fine-tuning does not improve clean accuracy and even slightly degrade it due to suboptimal training settings. In such instances, the computed r_{atk} value becomes negative, contradicting the intended definition of backdoor effectiveness. Since a negative backdoor effectiveness should mean that the backdoor was not effectively learnt by the model, but this phenomenon shows that the model effectively learned the backdoor but did not gain in clean per-

#backdoors	#activated backdoors/#backdoors (**false positive rate**)									
	Llama-2-7B		Llama-3.1-8B		Qwen-2.5-7B		Mistral-7B		Gemma-7B	
	Contam.	Clean	Contam.	Clean	Contam.	Clean	Contam.	Clean	Contam.	Clean
1.5k from MMLU-Pro + 5.2k from Big-Bench-Hard (MMLU-Pro treated as $D_{release}$)										
B=1	1/1 (**10%**)	0/1 (**100%**)	1/1 (**10%**)	1/1 (**10%**)	1/1 (**10%**)	0/1 (**100%**)	1/1 (**10%**)	0/1 (**100%**)	1/1 (**10%**)	0/1 (**100%**)
B=2	2/2 (**1%**)	0/2 (**100%**)	2/2 (**1%**)	0/2 (**100%**)	2/2 (**1%**)	0/2 (**100%**)	1/2 (**19%**)	0/2 (**100%**)	2/2 (**1%**)	0/2 (**100%**)
B=4	4/4 (**0.01%**)	1/4 (**34.39%**)	3/4 (**0.37%**)	0/4 (**100%**)	4/4 (**0.01%**)	0/4 (**100%**)	4/4 (**0.01%**)	0/4 (**100%**)	4/4 (**0.01%**)	0/4 (**100%**)
B=6	4/6 (**0.127%**)	1/6 (**46.86%**)	5/6 (**5.5e-5**)	1/6 (**46.86%**)	6/6 (**1e-6**)	0/6 (**100%**)	6/6 (**1e-6**)	1/6 (**46.86%**)	5/6 (**5.5e-5**)	1/6 (**46.86%**)
B=8	6/8 (**2.34e-5**)	1/8 (**56.95%**)	7/8 (**7.3e-7**)	1/8 (**56.95%**)	8/8 (**1e-8**)	1/8 (**56.95%**)	8/8 (**1e-8**)	1/8 (**56.95%**)	5/8 (**4.3e-4**)	0/8 (**100%**)

Table 6: The number of activated backdoors for contaminated/clean models and the corresponding **false positive rate**, i.e. *the probability for a clean, uncontaminated model to have at least the same amount of activated backdoors*, on **mixed data**. All FPRs are computed through our DyePack framework using Corollary 3.3. Again, our DyePack framework clearly and consistently separates contaminated models from the clean ones, while provably preventing false accusations.

formance. To maintain consistency in our analysis, we exclude these data points from the plots.

H More Results on Selecting Optimal Number of Backdoors

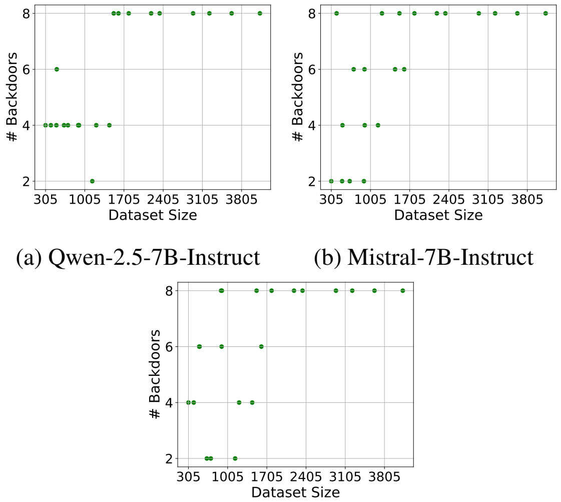
In the second part of our ablation studies, we analyzed the trend of how the size of the dataset affect the optimal choice for the number of backdoors. As a completion to the results presented in Figure 3, we present the results for the remaining models in Figure 5. As a supplement, we also present a heat-map in Figure 6 showing the trend of how FPR changes w.r.t. dataset size when using different number of backdoors. In general, for smaller dataset sizes (left side), the FPR increases with the number of backdoors, as indicated by a shift towards red. Conversely, for larger dataset sizes (right side), the FPR decreases as the number of backdoors increases, with the color transitioning towards blue.

I Will Mixing Test Data with Backdoor Samples Undermine Evaluation Quality?

Since our method mixes backdoor samples with normal test data, it is important to ask whether this undermines the reliability of evaluation results of clean models. We argue that the effect is negligible, both in theory and in practice.

First, consider how clean models behave on backdoor samples. During test set preparation, as described in Section 3.1, the backdoor targets are randomly sampled from a uniform distribution $T_i \sim \text{Uniform}(1, K)$. Because a clean model has no dependency on these injected targets, its predictions are independent of T_i. Formally,

$$T_i \mid f \overset{d}{=} T_i \sim \text{Uniform}(1, K),$$



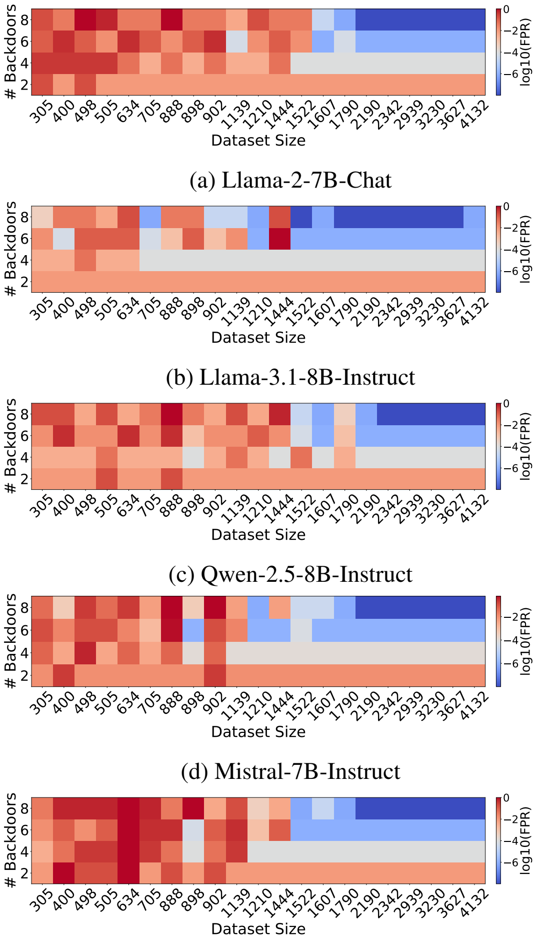
(a) Qwen-2.5-7B-Instruct (b) Mistral-7B-Instruct

(c) Gemma-7B-it

Figure 5: Number of backdoors that give the minimal FPR as a function of dataset size for Qwen-2.5-7B-Instruct, Mistral-7B-Instruct, and Gemma-7B-it.

where $\overset{d}{=}$ denotes equality in distribution (The same conclusion was used in our proof of Theorem 3.1). This implies that clean models effectively guess on the injected samples, achieving an expected accuracy of $1/K$. As a consequence, no clean model gains a systematic advantage or disadvantage from the presence of the backdoor samples. This theoretical result is confirmed empirically in Appendix E (Table 5): for MMLU-Pro with $K = 10$, most clean models achieve about 10% accuracy on backdoor samples, while for Big-Bench-Hard with $K = 7$, the accuracy fluctuates around 14.3%.

Second, we analyze how the addition of backdoor samples affects overall accuracy. Let N denote the number of clean samples, n_c the number of correct predictions on them, and n_b the number of correct predictions on backdoor samples (using the



(a) Llama-2-7B-Chat

(b) Llama-3.1-8B-Instruct

(c) Qwen-2.5-8B-Instruct

(d) Mistral-7B-Instruct

(e) Gemma-7B-it

Figure 6: Heat-map showing the trend of how FPR changes w.r.t. dataset size when using different numbers of backdoors on all models.

backdoor targets as ground truth). Define a slightly modified version of poison rate[6] as:

$$p = \frac{\#\text{backdoor samples}}{\#\text{clean samples}}.$$

The clean accuracy is $A_c = \frac{n_c}{N}$, while the combined accuracy is

$$A_b = \frac{n_c + n_b}{(1+p)N}.$$

Since $\mathbb{E}[n_b] = \frac{pN}{K}$, the relative difference between A_b and A_c is

$$\epsilon = \frac{A_b - A_c}{A_c},$$

Its expectation is

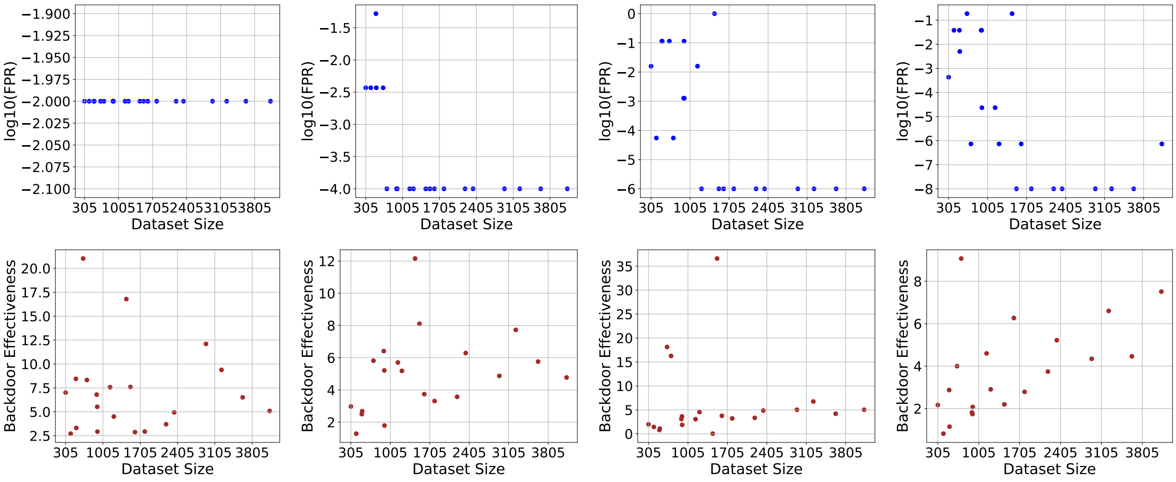
$$\mathbb{E}[\epsilon] = \left(\frac{N/K}{n_c} - 1\right) \cdot \frac{p}{1+p}.$$

For any model performing better than random guess on clean data, the prefactor $\left(\frac{N/K}{n_c} - 1\right)$ lies strictly between -1 and 0, which means that the accuracy distortion decreases on the order of $1/p$. And since the poison rate needed is rather small (as low as 2.2% for our setup in Appendix F, meaning we do not need to include too many backdoor samples), the relative error is negligible. In practice, the minimum poison rate required for backdoors to be effective depends on external factors outside of the DyePack framework—e.g., attack design, trigger strength, training hyperparameters. We clarify that our objective is not to propose a stronger backdoor attack method, but to theoretically and empirically demonstrate the effectiveness of repurposing backdoors for contamination detection while providing computable and bounded FPRs.

We also validate the stability of model ranking empirically. Table 5 shows that across both datasets and 8 different values of B, the relative ranking of models remains unchanged before and after adding backdoor samples. For example, on MMLU-Pro with $B = 8$, five models maintain exactly the same order despite small drops in raw accuracy. Across 100 head-to-head model comparisons (two datasets, five values of B, and ten pairwise model combinations), the minimum injection rate required to flip any ranking is approximately 28.1%, which is far larger than the rates we used.

Moreover, in practice, when it comes to the need of strictly verifying the quality and trustworthiness of becnhmark evaluation, the more reliable and accepted approach is to use evaluator-run leaderboards (e.g., Open LLM Leaderboard (Beeching et al., 2023), BFCL (Patil et al., 2025), LM Arena (lma)), rather than self-reported results (e.g., company blog posts). Since leaderboard owners run the evaluation, they know which samples are clean or backdoored, and can report accurate clean accuracy directly, which completely avoids any accuracy distortions caused by backdoor samples.

[6] This differs slightly from the poison rate definition used elsewhere in our paper but simplifies the math without affecting conclusions.



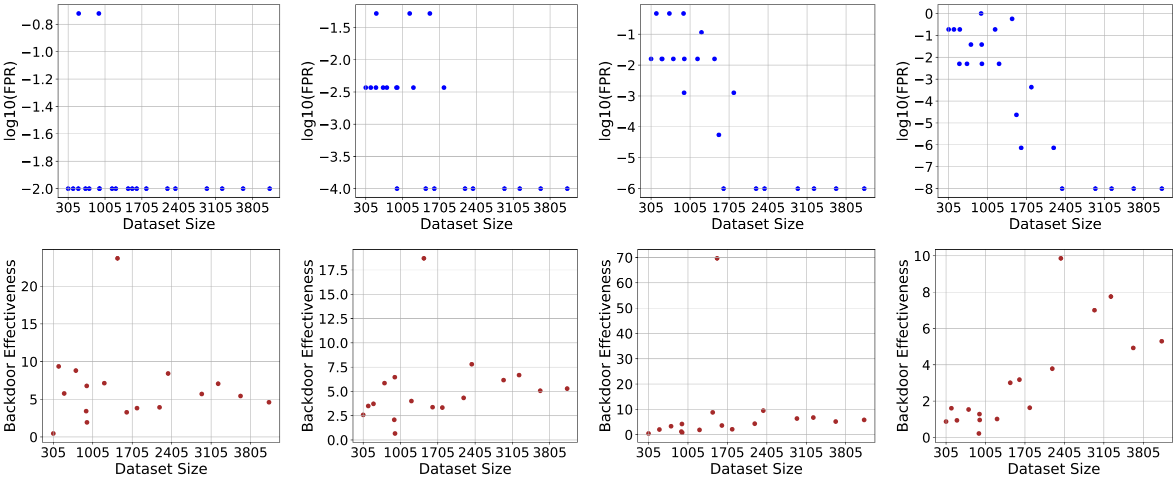
B=2

B=4

B=6

B=8

Figure 7: The FPR for detecting contamination and the backdoor effectiveness as functions of the dataset size for Llama-3.1-8B-Instruct under different number of backdoors. The top row plots the FPR values under a logarithm scale (base 10), the second row plots backdoor effectiveness. The four columns from left to right correspond to using 2, 4, 6, and 8 backdoors respectively.



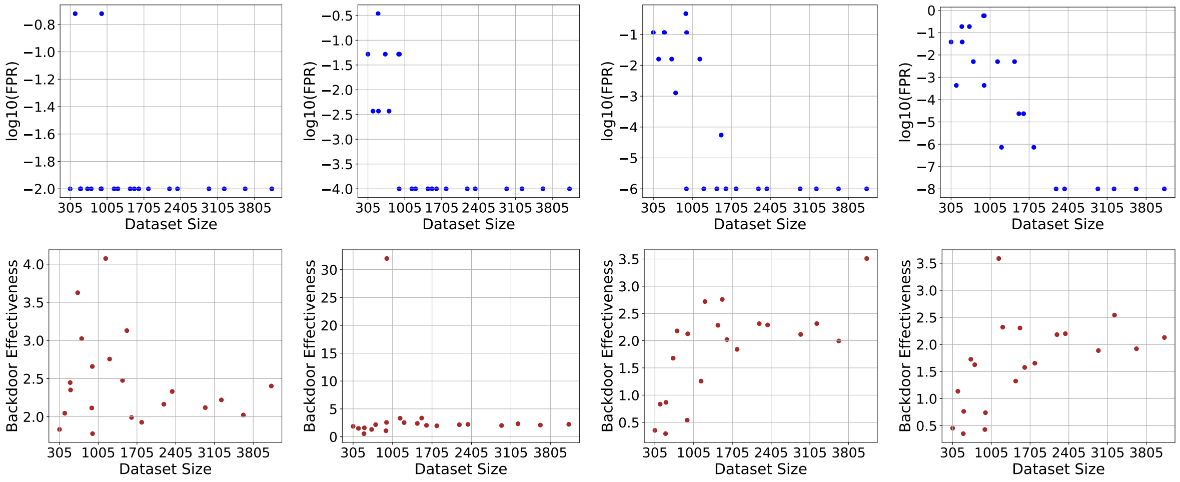
B=2

B=4

B=6

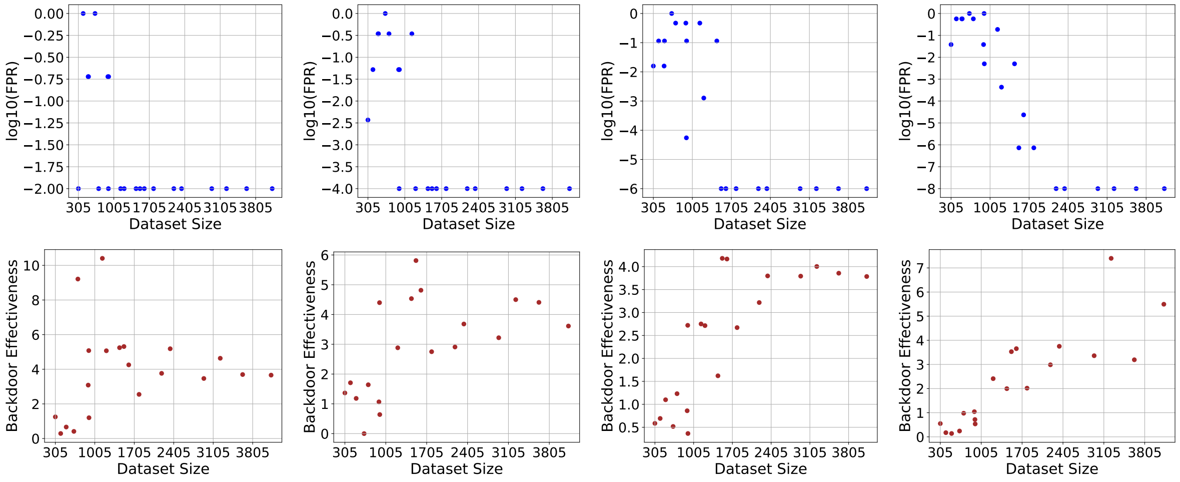
B=8

Figure 8: The FPR for detecting contamination and the backdoor effectiveness as functions of the dataset size for Qwen-2.5-7B-Instruct under different number of backdoors. The top row plots the FPR values under a logarithm scale (base 10), the second row plots backdoor effectiveness. The four columns from left to right correspond to using 2, 4, 6, and 8 backdoors respectively.



B=2 B=4 B=6 B=8

Figure 9: The FPR for detecting contamination and the backdoor effectiveness as functions of the dataset size for Mistral-7B under different number of backdoors. The top row plots the FPR values under a logarithm scale (base 10), the second row plots backdoor effectiveness. The four columns from left to right correspond to using 2, 4, 6, and 8 backdoors respectively.



B=2 B=4 B=6 B=8

Figure 10: The FPR for detecting contamination and the backdoor effectiveness as functions of the dataset size for Gemma-7B under different number of backdoors. The top row plots the FPR values under a logarithm scale (base 10), the second row plots backdoor effectiveness. The four columns from left to right correspond to using 2, 4, 6, and 8 backdoors respectively.