

Enabling Confidence-Aware Control of Effective Memorization in Large Language Models

Chenjie Ni¹, Omid Mersa⁵, Zhepeng Wang², Runxue Bao³, Qun Li⁵,
Shangqian Gao^{4*}, Yanfu Zhang^{5*}

¹Northwestern University, ²George Mason University,

³University of Pittsburgh, ⁴Florida State University, ⁵William and Mary

¹chenjieni2027@u.northwestern.edu, ²zwang48@gmu.edu, ³baorunxue@gmail.com,

⁴sgao@cs.fsu.edu, ⁵{yzhang105, omersa, qxlixxu}@wm.edu

Abstract

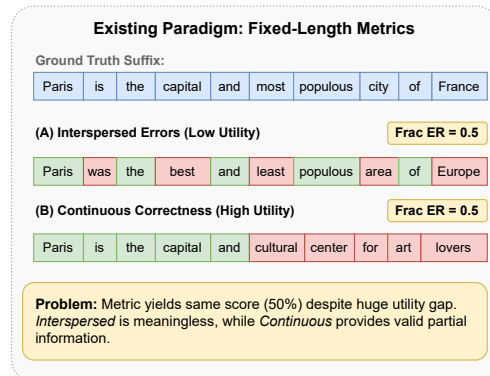
Current evaluations of memorization in Large Language Models (LLMs) predominantly rely on overlap ratios, which fails to distinguish between sporadic token matches and contiguous verbatim reproduction. What’s more, these methods become inapplicable in scenarios where ground truth suffixes is unavailable. To address these limitations, we propose a paradigm shift from quantifying rates to determining the **Effective Memorization Length**—the exact-match suffix ending at the first mismatch. We introduce a Ensemble Memorization Classifier that leverages internal states from the LLM to estimate this boundary in a reference-free manner. This framework enables Confidence-Aware Generation Control, high-precision suffix extraction by dynamically truncating sequences at the predicted memorization boundary. Furthermore, We use the classifier as an RL reward in a proof-of-concept stress test. The results indicate a usable optimization signal, with modest gains concentrated in unsaturated settings. Experiments on the GPT-Neo and Pythia families demonstrate that our approach not only distinguishes exact-match from mismatch tokens but also provides an operational tool for auditing and controlling verbatim continuation risk, transforming memorization from an unpredictable risk into a controllable utility.¹

1 Introduction

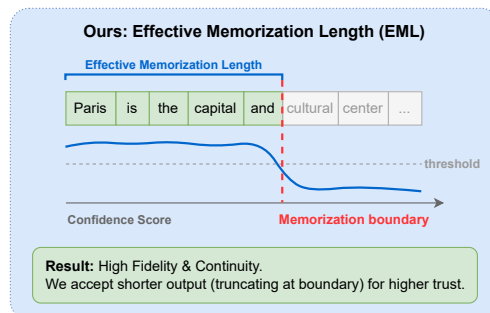
Large Language Models (Devlin et al., 2019; Radford et al., 2019; Raffel et al., 2020) have demonstrated impressive capabilities across a wide range of natural language tasks, yet they also exhibit the tendency to memorize portions of their training data, namely memorization (Carlini et al., 2021; Carlini et al., 2022). Memorization raises significant copyright and privacy concerns when sensitive

* Corresponding author.

¹Our source code is publicly available at: <https://github.com/artemis03976/EffectiveMemorization>



(a) Existing metrics for evaluating memorization.



(b) Our new paradigm.

Figure 1: Comparison between the existing fixed-length metric and our proposed Effective Memorization Length (EML).

or proprietary data is unintentionally leaked. On the other hand, benign memorization (Anagnostidis et al., 2023; Wang et al., 2024a) is heavily relied upon in practical applications that demand precise factual recall, such as generating exact code snippets or citing legal documents.

Existing works typically quantify memorization through overlap-based metrics in a prefix-suffix manner, such as the Exact Extractable Rate (Exact ER) or Fractional Extractable Rate (Frac ER) (Xiong et al., 2025; Ozdayi et al., 2023; Wang et al., 2024b). While these metrics are highly effective for privacy auditing, where the goal is to verify

whether a specific string from the training corpus has been leaked, they exhibit limitations when evaluating generation reliability. LLMs can exhibit "scattered memorization", as shown in Figure 1a, producing sequences where isolated memorized tokens are interleaved with generic text. Consider an attacker attempting to recover a 10-digit password. A metric stating "50% accuracy" is operationally confusing if the correct digits are scattered randomly without position information. However, knowing that the first 5 digits are definitely correct provides a known boundary from which the remaining digits can be deduced.

Therefore, ensuring generation reliability requires establishing a "Trust Boundary"—a continuous span of output where the model remains faithfully grounded in its memorized knowledge before diverging into uncertainty. To capture this usability constraint, we shift the focus from discrete token overlap to the continuity of memorization. We propose **Effective Memorization Length**, which measures the length of the continuous exact-match suffix, as shown in Figure 1b. However, identifying this boundary poses a practical challenge. Traditional evaluations assume access to the ground-truth suffixes to verify the generated suffixes. Yet, in real deployment, such reference text is often unavailable.

To this end, we propose a novel framework for estimating and extending the Effective Memorization Length. First, we introduce a supervised token-level memorization classifier to estimate the "memorization confidence" for suffix tokens. During training, we extract the internal state from the LLM generation as the input features and derive token-level labels by comparing generated suffixes against their ground-truth suffixes. These scores define two threshold-dependent quantities, the **Strict Boundary Length** and the **Relaxed Boundary Length**, which support suffix truncation based on predicted exact-match probabilities, yielding shorter but highly reliable content. Secondly, we integrate this classifier as a reward model within a Group Relative Policy Optimization (GRPO, Shao et al., 2024) framework to test whether its confidence signal can guide soft-prompt optimization. By optimizing for this confidence-based length, we train soft prompts that modestly extend existing memorized traces, forcing it to sustain reliable recall for longer periods.

Our contributions are summarized as follows:

- We shift focus from overlap ratios to consider the generation reliability regarding memorization and propose the Effective Memorization Length, which emphasizes continuity and the identification of the trust boundary.
- We develop a robust ensemble classifier that accurately detects the transition from verbatim memorization to open-ended generation, enabling Confidence-Aware Generation Control that extracts variable-length, high-precision suffixes.
- We demonstrate that this boundary is not static but optimizable. Through our RL framework, we extend the boundary length of generated sequences compared to standard SFT baselines and show that the classifier’s confidence signal is valid for guiding optimization.

2 Related Works

LLM Memorization. The phenomenon of memorization in Large Language Models was pioneered by Carlini et al. (2021), who demonstrated the feasibility of extracting verbatim training samples by prompting models with specific prefixes. This behavior raises significant ethical and practical concerns. For instance, Zhou et al. (2024) verified the feasibility of extracting specific entities from model generations, while Karamolegkou et al. (2023) underscored the legal implications regarding copyright infringement of proprietary texts. Beyond theoretical vulnerabilities, real-world incidents have been documented; notably, Maura (2023) reported cases where confidential source code submitted to LLMs was subsequently leaked to other users due to memorization. In response to these risks, the community has explored both extraction and defense mechanisms. On the attack side, Kassem et al. (2024) introduced a method utilizing an auxiliary attacker LLM to provoke the victim model into leaking data. In terms of defense, Ozdayi et al. (2023) investigated soft-prompt techniques to control generation behavior in both attack and defense settings. Similarly, Dong et al. (2024) proposed an unlearning framework that leverages self-distillation and a deliberate imagination mechanism to guide LLMs in forgetting sensitive information. However, these studies predominantly focus on generating and evaluating fixed-length suffixes, overlooking the granularity of memorization. In contrast, our approach targets a confidence signal

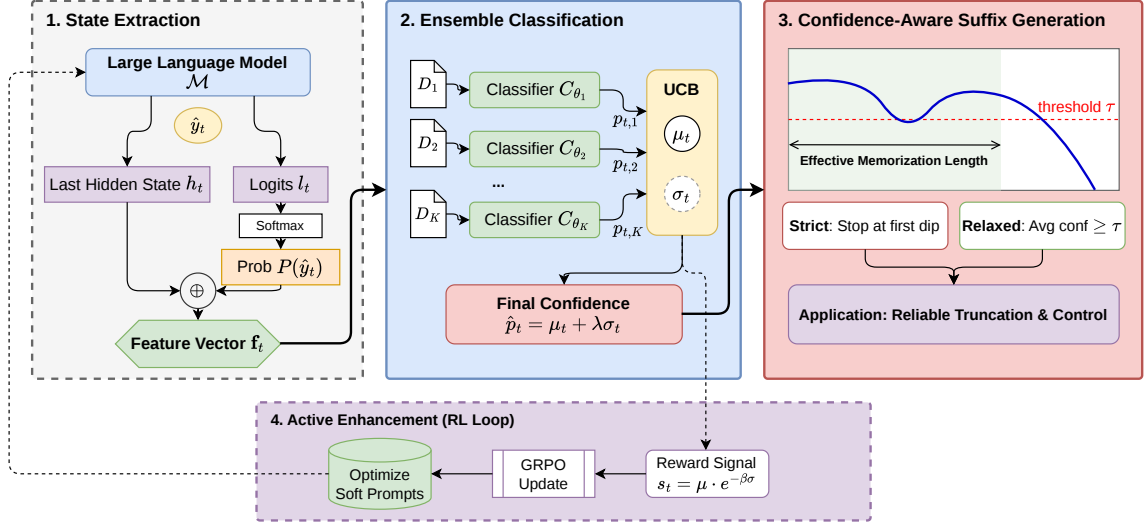


Figure 2: Overview of the proposed framework for confidence-aware control of effective memorization

predictive of verbatim memorization and boundary, enabling the generation of variable-length, high-fidelity memorized suffixes, which provides a analytical tool for studying memorization.

Memorization Evaluation. Existing approaches for memorization evaluation mainly rely on the token match between generated suffixes and the ground-truth suffixes. The most rigorous metric is *Exact Memorization*, which requires the generated suffix to exactly match the ground-truth suffixes. For instance, [Carlini et al. \(2022\)](#) defined discoverable memorization by verifying whether an LLM exactly reproduces the continuation of a prefix in the training corpus. Similarly, [Tirumala et al. \(2022\)](#) formulated exact memorization as the proportion of contexts where the model reproduces the suffixes using greedy decoding. To account for imperfect recall, recent works have increasingly adopted *Approximate memorization* metrics and their variants. [Ozdayi et al. \(2023\)](#) defines Fractional Extraction Rate (Frac ER), which quantifies the token-level overlap between the generated and the ground-truth suffixes. [Ippolito et al. \(2023\)](#) detects memorization by calculating the edit distance between the two. Additionally, some studies employ traditional text generation metrics, such as ROUGE-L ([Lin, 2004](#)), or perplexity-based measures (e.g., loss on training samples) to serve as proxies for memorization ([Ponkshe et al.](#)). However, most of these metrics provide only a post-hoc statistical summary and rely on comparison with ground truth suffixes. Instead, our work focuses on Effective Memorization in a reference-free paradigm. By decoding the model’s internal

confidence, we can assess memorization boundary without external verification.

3 Methodology

3.1 Problem Formulation

We study the memorization behavior of an autoregressive LLM $\mathcal{M}$ in a prefix–suffix setting. Given an input prefix $\mathbf{x}$ sampled from a training corpus, let T denote the evaluated suffix length and $\mathbf{y} = (y_1, \dots, y_T)$ denote the corresponding ground-truth suffix. During inference, the model $\mathcal{M}$ takes $\mathbf{x}$ as input and generates a candidate suffix $\hat{\mathbf{y}} = (\hat{y}_1, \dots, \hat{y}_T)$ with greedy decoding. If decoding terminates earlier with an end-of-sequence (EOS) token, positions after EOS are excluded, with the ground-truth suffix truncated to the same length.

We argue that *Effective Memorization* should be defined by continuity under usability constraint for generation reliability: a generated suffix remains valid only up to the point at which the model’s output first diverges from the ground truth. To capture this notion, we introduce the **Effective Memorization Length** L^* , defined as the length of the longest continuous sequence that $\hat{\mathbf{y}}$ exactly matches $\mathbf{y}$:

$$L^*(\hat{\mathbf{y}}, \mathbf{y}) = \max \{k \in \{0, \dots, T\} \mid \hat{\mathbf{y}}_{1:k} = \mathbf{y}_{1:k}\}. \quad (1)$$

The metric L^* represents the true *memorization boundary* of the model for a given prefix. However, in many practical scenarios, the ground-truth suffix $\mathbf{y}$ is unavailable, rendering L^* intractable to compute directly. To address this limitation, we

propose a confidence-based approximation that estimates the memorization boundary without access to ground truth. Specifically, given a confidence threshold $\tau \in [0, 1]$, we define the *Strict Boundary Length* $L_{\text{strict}}(\tau)$ as the maximum prefix length for which the model consistently maintains a memorization confidence above τ :

$$L_{\text{strict}}(\tau) = \max \left\{ k \mid \begin{array}{l} 0 \leq k \leq T, \\ \hat{p}_t \geq \tau, \forall 1 \leq t \leq k \end{array} \right\}. \quad (2)$$

Under this framework, we aim to address two research problems. First, computing $L_{\text{strict}}(\tau)$ requires accurate estimation of token-level memorization confidence. To this end, we design a parameterized classifier $\mathcal{C}_\theta$ that assigns a memorization probability score p_t to each generated token $\hat{y}_t$. We train this classifier such that $L_{\text{strict}}(\tau)$ can serve as a confidence-based estimator of L^* , ensuring memorization control without ground-truth suffixes. Second, given this confidence-based formulation of memorization, we investigate whether the classifier can serve as an optimization signal in a proof-of-concept RL stress test. A detailed overview of our proposed pipeline is illustrated in Figure 2.

3.2 Token-Level Memorization Classifier

We construct a feature vector $\mathbf{f}_t$ using the internal representations from the LLM $\mathcal{M}$ as the input to the classifier $\mathcal{C}_\theta$. Specifically, for each generated suffix token $\hat{y}_t$, we utilize: (1) the final-layer hidden state $\mathbf{h}_t \in \mathbb{R}^d$ aligned with generated position t , which is available immediately after $\hat{y}_t$ is produced; and (2) the generation probability $P(\hat{y}_t) \in \mathbb{R}$ calculated from the logits $\mathbf{l}_t$ via a Soft-max operation, capturing the model’s predictive confidence. These two features together provide robust signals for determining whether a token reflects genuine recall of training data. The base LLM is frozen when training the classifier. We concatenate these components to form the final feature vector: $\mathbf{f}_t = [\mathbf{h}_t; P(\hat{y}_t)]$.

However, whether a token "is memorized" often depends heavily on the correctness of the preceding sequence. Therefore, we design $\mathcal{C}_\theta$ as a lightweight 3-layer Transformer-based attention block with a causal attention mask. Formally, the probability sequence $\mathbf{P}$ is computed as:

$$\begin{aligned} \mathbf{z}_{1:T} &= \text{Transformer}_{\text{causal}}(\mathbf{F}), \\ p_t &= \mathcal{C}_\theta(\mathbf{F}_{1:t}) = \sigma(\mathbf{z}_t^\top \mathbf{W} + b), \end{aligned} \quad (3)$$

where $\mathbf{F} = [\mathbf{f}_1, \dots, \mathbf{f}_T]$, $\mathbf{F}_{1:t} = [\mathbf{f}_1, \dots, \mathbf{f}_t]$, $\mathbf{W}$ and b are the parameters of the final linear projection, and σ is the sigmoid function. The causal mask prevents position t from attending to any position after t . By leveraging self-attention mechanisms, the classifier models how the faithful recall of previous tokens can affect the likelihood of the current token being memorized.

We train $\mathcal{C}_\theta$ on a labeled dataset $\mathcal{D} = \{(\mathbf{F}^{(i)}, \mathbf{m}^{(i)})\}_{i=1}^N$ consisting of feature sequences and their corresponding ground-truth binary masks, where $m_t = \mathbb{I}[\hat{y}_t = y_t]$, by minimizing the standard Binary Cross-Entropy (BCE) loss. Despite its simplicity, this architecture achieves strong discriminative performance and provides accurate token-level signals for the downstream application described in the following sections. We use these local token-match labels for classifier training rather than cumulative-label formulation. See Appendix C for details.

3.3 Ensemble Classifier with UCB Aggregation

Relying on a single classifier $\mathcal{C}_\theta$ can be susceptible to noise and overfitting. A single False Negative (assigning low confidence to an a memorized token) will prematurely halt the trust boundary and underestimate the EML. To enhance the robustness of our confidence estimation, we employ an ensemble learning framework integrated with an Upper Confidence Bound (UCB) aggregation strategy (Auer et al., 2002).

We construct an ensemble of K classifiers $\mathcal{E} = \{\mathcal{C}_1, \dots, \mathcal{C}_K\}$, where each classifier is trained on a size- N bootstrap sample drawn with replacement from the classifier training set. All ensemble members share the same held-out validation and test sets. When evaluating the generated suffixes, for each token step t , the ensemble yields a set of probability predictions $\{p_{t,1}, \dots, p_{t,K}\}$. We derive the final confidence score $\hat{p}_t$ for token $\hat{y}_t$ using:

$$\hat{p}_t = \text{clip}_{[0,1]}(\mu_t + \lambda \sigma_t), \quad (4)$$

where μ_t and σ_t denote the empirical mean and standard deviation of the ensemble predictions, respectively. The hyperparameter $\lambda \geq 0$ regulates the weight of ensemble disagreement. This UCB score can reduce premature truncation caused by a single false negative, but its optimism can also increase over-estimation risk.

3.4 Confidence-Aware Suffix Generation

By leveraging the robust confidence scores $\hat{\mathbf{P}} = (\hat{p}_1, \dots, \hat{p}_T)$, we define two threshold-dependent retained-length quantities with different stopping semantics.

Strict Boundary Length. This strategy directly implements $L_{\text{strict}}(\tau)$ from Section 3.1. Equivalently, it is the number of tokens preceding the first confidence score below τ , where $t \in \{1, \dots, T\}$:

$$L_{\text{strict}}(\tau) = \min(\{t : \hat{p}_t < \tau\} \cup \{T + 1\}) - 1. \quad (5)$$

This strategy provides a conservative threshold-controlled estimation of the true memorization boundary L^* . By enforcing that every retained token satisfies $\hat{p}_t \geq \tau$, it ensures maximum local fidelity.

Relaxed Boundary Length. Strict truncation can be overly conservative, as LLMs may occasionally generate low-confidence tokens followed by a return to correct memorization. To quantify average prefix confidence without first-failure stopping, we define a separate Relaxed Boundary Length $L_{\text{relaxed}}(\tau)$ as:

$$\begin{aligned} \bar{p}_L &= \frac{1}{L} \sum_{t=1}^L \hat{p}_t, \\ \mathcal{A}_\tau &= \{L \in \{1, \dots, T\} : \bar{p}_L \geq \tau\}, \\ L_{\text{relaxed}}(\tau) &= \max(\{0\} \cup \mathcal{A}_\tau). \end{aligned} \quad (6)$$

By making the stopping condition robust to transient confidence drops, this formulation extends the effective length for sequences that exhibit local uncertainty while maintaining global coherence.

3.5 Serving as a Reward Signal

Soft Prompt Tuning. To efficiently adapt the model without the prohibitive cost of full-parameter fine-tuning, we employ Soft Prompt Tuning. Following prior work (Ozdayi et al., 2023; Wang et al., 2024b), we construct a Transformer-based soft prompt generator for prompt tuning and train with supervised methods to yield a pretrained generator. Then we optimize the soft prompt parameters using Group Relative Policy Optimization (GRPO). This exploratory intervention tests reward compatibility in a controlled setting.

Cumulative Memorization Reward. We define the validity of the suffix up to step t as the cumulative product of the scores of all preceding tokens. We perform these operations in log-space to ensure

numerical stability. Let v_t be the log-probability of the suffix being memorized up to step t , the final reward R for the entire suffix is the sum of these cumulative "memorization likelihoods" across all time steps:

$$R(\hat{\mathbf{y}}) = \sum_{t=1}^T v_t, \quad v_t = \sum_{k=1}^t \log(s_k). \quad (7)$$

The rearranged form makes the weighting explicit: token k contributes to all later prefix scores and therefore has coefficient $T - k + 1$. This weighting reflects first-error semantics, because an early low-confidence token affects every longer prefix. It consequently prioritizes prefix reliability and places comparatively weaker direct pressure on extending the tail after a reliable prefix.

Anti-Reward Hacking. In Section 3.3, we used UCB strategy to discover potential memorization. However, in an RL training context, an optimistic reward signal carries the risk of reward hacking (Amodei et al., 2016; Eisenstein et al., 2024). For instance, the generator might exploit the reward by producing high-frequency phrases that trigger high confidence scores in the classifier due to their statistical familiarity, despite deviating from the ground-truth suffix. We therefore use a distinct, Lower Confidence Bound (LCB) style score (Coste et al., 2024; Yu et al., 2020) for reward construction. For a generated token at step t , given the ensemble's mean probability μ_t and standard deviation σ_t , we define the ensemble confidence score s_t as:

$$s_t = \mu_t \cdot \exp(-\beta \cdot \sigma_t), \quad (8)$$

where $\beta > 0$ is a penalty coefficient. This formulation penalizes ensemble disagreement, reducing the reward assigned to uncertain generations and thereby mitigating reward hacking.

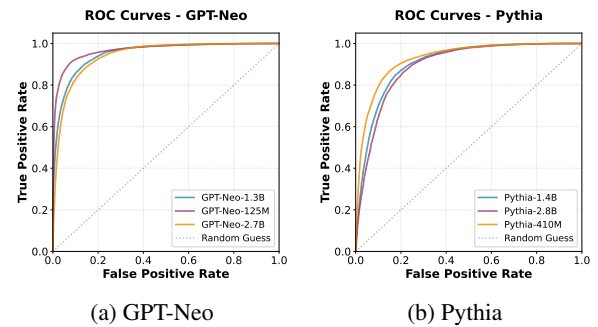


Figure 3: ROC Curves of the ensemble classifier on GPT-Neo and Pythia family

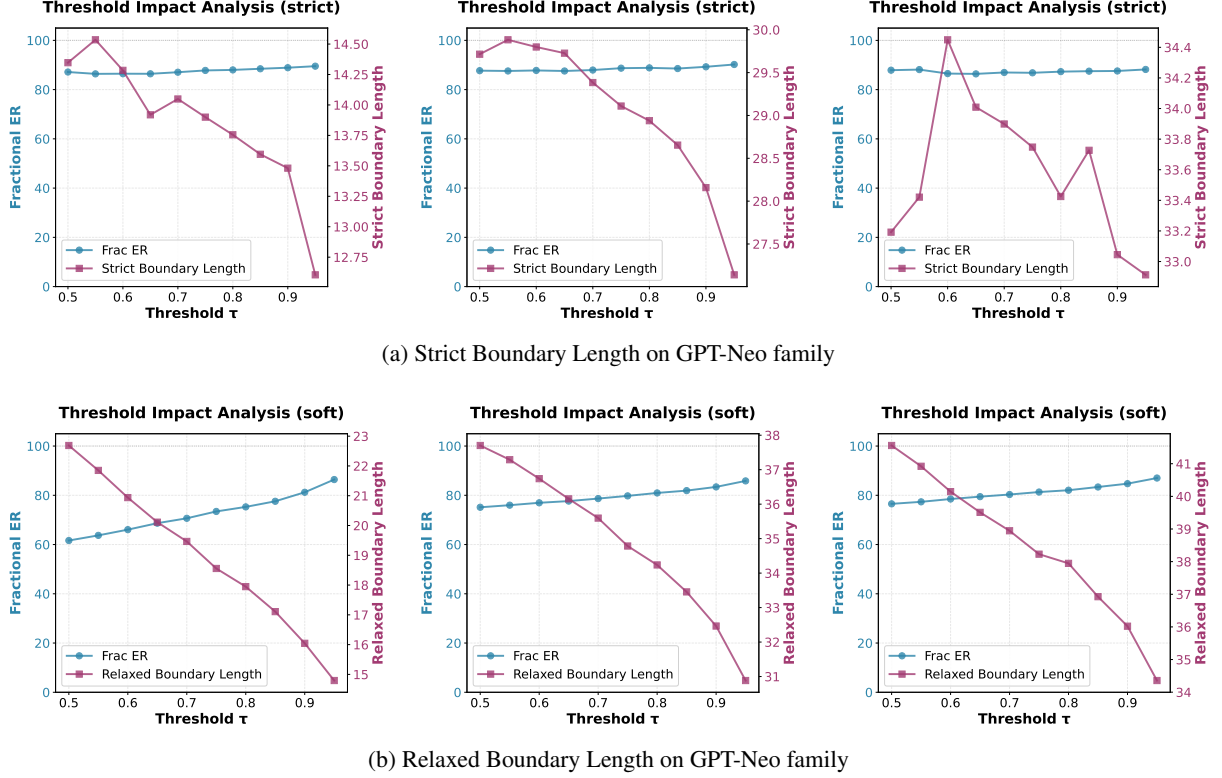


Figure 4: Trade-off between Fractional Extraction Rate (Frac ER) (Left Y-axis) and Approximate Effective Memorization Length (Approximate EML) (Right Y-axis) across varying confidence thresholds τ

Table 1: Evaluation results of ensemble classifier on GPT-Neo and Pythia family

Model	Method	Accuracy	F1-Score	AUC
GPT-Neo 125M	Max-Prob	0.5388	0.4248	0.7644
	Entropy	0.5208	0.4232	0.7698
	Ours	0.9108	0.6375	0.9682
GPT-Neo 1.3B	Max-Prob	0.7261	0.6989	0.7365
	Entropy	0.7157	0.6973	0.7352
	Ours	0.8821	0.8165	0.9496
GPT-Neo 2.7B	Max-Prob	0.7504	0.7477	0.7260
	Entropy	0.7541	0.7460	0.7251
	Ours	0.8859	0.8416	0.9409
Pythia 410M	Max-Prob	0.5971	0.5499	0.7138
	Entropy	0.6028	0.5476	0.7132
	Ours	0.8484	0.7094	0.9324
Pythia 1.4B	Max-Prob	0.7364	0.7422	0.6556
	Entropy	0.7331	0.7405	0.6540
	Ours	0.8417	0.8217	0.8998
Pythia 2.8B	Max-Prob	0.7757	0.7833	0.6309
	Entropy	0.7692	0.7823	0.6293
	Ours	0.8641	0.8454	0.8852

4 Experiments

4.1 Experimental Setup

Models. To investigate memorization mechanisms thoroughly, it is necessary to use models with both open-sourced weights and training data. We select two representative open-source model families pre-trained on the Pile dataset (Gao et al., 2020): GPT-

Neo (125M, 1.3B, 2.7B) (Jeong et al., 2023) and Pythia (410M, 1.4B, 2.8B) (Biderman et al., 2023).

Datasets. We utilize the dataset from the *Language Model Extraction Benchmark* (Google-research, 2022). This benchmark contains 15,000 sequences specifically sampled from the Pile to analyze memorization behaviors. Due to the absence of ground-truth suffixes in the official test set, we instead use the official validation set as our test set. The original training set is then re-partitioned into 14,000 samples for training our classifier and 1,000 samples for validation.

Baseline. As "Effective Memorization" is a novel problem formulation, there are no existing baselines directly applicable to this task. Therefore, we establish two unsupervised baselines grounded in uncertainty estimation, operating on the hypothesis that *memorized tokens correspond to states of high predictive certainty*: (1) Token Probability (MaxProb): We use the softmax probability of the generated token $\hat{y}_t$ as the confidence score, defined as $s_t^{\text{prob}} = P_t(\hat{y}_t)$; (2) Entropy: We calculate the entropy of the distribution over the vocabulary $\mathcal{V}$ at step t and treat the negative entropy as the confidence score: $s_t^{\text{ent}} = -\sum_{w \in \mathcal{V}} P_t(w) \log P_t(w)$.

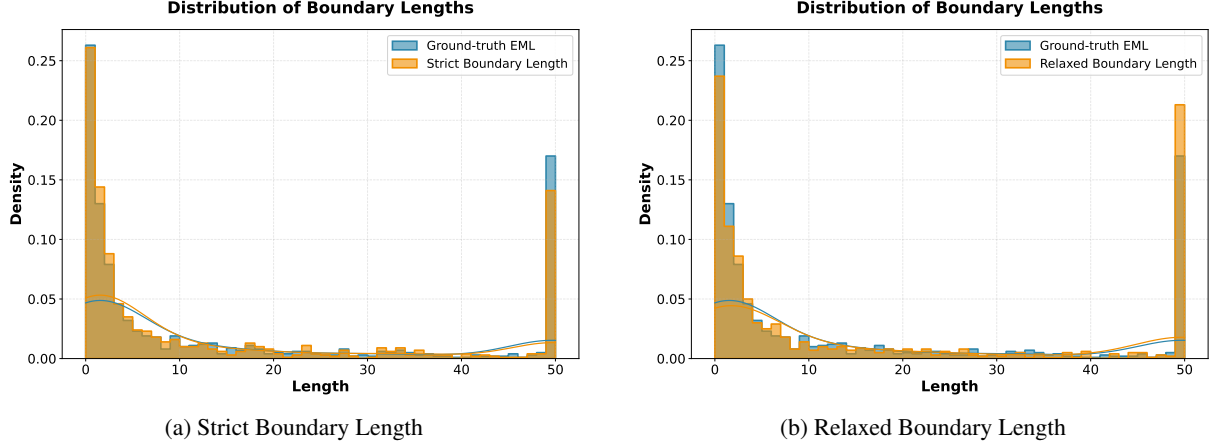


Figure 5: Histogram comparison of the ground-truth EML with the Strict and Relaxed Boundary Lengths (with $\tau = 0.9$)

4.2 Validation of the Memorization Classifier

We first evaluate the ability of the classifier to distinguish between memorized and non-memorized tokens. Table 1 and Figure 3 present the performance of our proposed classifier.

The model achieves an average F1-score of 0.7652 and average AUC of 0.9529 on the GPT-Neo family, demonstrating a strong capability to decode the memorization status. Note that for the GPT-Neo-125M and Pythia-410M models, their F1 scores are significantly lower than those of other models. We found this is because smaller models exhibit relatively minor memorization. This implies that memorized patterns are typically easier to learn and identify than unmemorized ones. Conversely, this also implies that the more pronounced a model’s memorization phenomenon is, the easier it becomes for our classifier to make accurate predictions, as demonstrated in GPT-Neo-2.7B.

4.3 Confidence-Aware Generation Control

We now evaluate how well our strategies allow users to estimate and control the memorization boundary without access to ground-truth suffixes.

The Fidelity-Length Trade-off. Figure 4 illustrates the trade-off between the Fractional Extraction Rate and the Strict/Relaxed Boundary Length on GPT-Neo family. As the confidence threshold τ increases, we observe a monotonic increase in the Frac ER and decrease in the retained length. Crucially, even at a strict threshold of $\tau = 0.9$, our method retains a substantial average length of 30.61 tokens. This confirms that high-confidence memorization is not rare or sporadic. There exists a contiguous "safe zone" of considerable length.

By selecting τ , we can effectively define the risk tolerance and obtain a sequence L_τ where every token is vetted for reliability.

Alignment with the True Boundary. To verify that our method is not simply cutting sequences arbitrarily, we compare the distribution of our Boundary Length against the true Effective Memorization Length. Figure 5 presents the overlap of these two distributions with the two strategies on GPT-Neo-125M. The results show that Strict Boundary Length ensures high precision but occasionally underestimates the boundary due to local uncertainty. While Relaxed Boundary Length successfully bridges these minor gaps, demonstrating that it can also approximate the true L^* with high accuracy even though with a minor overconfidence. This confirms that our framework effectively "finds" the natural end of the model’s memory, independent of fixed length limits.

4.4 Case Study

Our analysis above revealed a significant phenomenon: a large proportion of generated suffixes are "fully memorized", suggesting that the model’s actual memorization boundary often extends far beyond the limits of the evaluation dataset. This scenario presents a perfect testbed to validate our framework’s capability in reference-free settings. We sampled 1,000 prefixes from our validation set and allowed the LLM to continue generating beyond the dataset’s ground-truth cutoff with extra 100 tokens until our Strict Truncation strategy ($\tau = 0.9$) triggered a stop. Out of the 1,000 generations, 143 sequences maintained consistently high memorization confidence scores across the entire

Table 2: Case Study of Unbounded Memorization. The blue text represents the content generated *after* the ground truth from the dataset and matched to public sources through the search engine (Foundation, 2025; stackspin, 2025; Kubernetes, 2025). The texts are restructured for display.

Category	Prefix	Extended Suffix (Post-cutoff)	Ext. Len
License	... Copyright (C) 2013-2020 OpenFOAM Foundation ... License: This file...	... is part of OpenFOAM. ... under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. ... See the GNU General Public License for more details.	+93
Code	...func (m *Cert...) Unmarshal(dAtA []byte) error { l := len(dAtA) iNdEx := 0 for ...	iNdEx < l { ... if shift >= 64 { return ErrIntOverflowGenerated } ... if b < 0x80 { break } ...	+100
Document	...the feature gate WatchBookmarks is not enabled in apiserver, ... The continue option should be set when retrieving more results from...	...the server. ... and the server may reject a continue value it does not recognize. If the specified continue value is no longer valid whether due to expiration (generally five to fifteen minutes) or a configuration change on the server, ..., but from the latest snapshot, which is...	+100

Table 3: Results on GPT-Neo and Pythia families

Model	Method	Frac ER	Exact ER	EML
GPT-Neo 125M	Origin	0.348	0.168	14.221
	SFT	0.563	0.445	25.845
	Ours	0.566	0.464	26.348
GPT-Neo 1.3B	Origin	0.636	0.447	29.095
	SFT	0.753	0.644	36.098
	Ours	0.756	0.651	36.338
GPT-Neo 2.7B	Origin	0.687	0.515	32.217
	SFT	0.788	0.656	37.873
	Ours	0.790	0.656	37.911
Pythia 410M	Origin	0.486	0.260	21.212
	SFT	0.509	0.309	22.880
	Ours	0.537	0.356	24.633
Pythia 1.4B	Origin	0.694	0.441	31.649
	SFT	0.733	0.557	34.814
	Ours	0.736	0.559	34.839
Pythia 2.8B	Origin	0.733	0.496	34.148
	SFT	0.773	0.591	36.610
	Ours	0.777	0.594	36.630

100-token span. We then manually verified these via search engines, as the original version of Pile is unavailable due to copyright reasons. Remarkably, we verified that over 90% of these sequences were exact, verbatim matches from public sources.

Table 2 presents representative examples of this "unbounded" memorization. These "ultra-long" memorization chains are predominantly associated with highly structured data, such as open-source software licenses, code snippets, and documents. We hypothesize that this behavior stems from data repetition in the pre-training stage. Such standardized texts reinforced the model’s internal pathways and created memorization that are robust, long-lasting, and easily identifiable by our confidence-aware framework.

4.5 Results on Memorization Enhancement

Finally, we explore the potential of using our ensemble classifier as a reward model to actively enhance memorization. We compared its generation quality against the original SFT baseline. Table 3 summarizes the results on the test set. We observe that our method achieves a consistent improvement across metrics. Specifically, the Exact ER increases by 15.2% on the Pythia-410M, and the Average Retained Length extends by approximately 1.75 tokens. While the RL loop successfully leads the model toward higher-confidence generations, the magnitude of improvement is limited. We attribute this to two primary factors: (1), the baseline SFT model was already fine-tuned and saturated on the target corpus, especially for larger models such as GPT-Neo-2.7B and Pythia-2.8B. The RL stage effectively acts as a "polisher", correcting minor issues in the suffix generation but cannot force the model to recall information that was never effectively encoded in the weights during pre-training. (2), while parameter-efficient, soft prompt tuning approach has limited capacity to alter the model’s fundamental architecture due to its nature of enhancing the embedding. The model’s ability to generalize memorization to the "long tail" of the data is ultimately bounded by its frozen weights, which the soft prompt cannot modify.

Despite these limitations, the experiment provides a proof-of-concept that the classifier’s signals can guide an RL policy in this setting. The result supports reward compatibility for controlled analysis, while the modest and setting-dependent gains leave broader enhancement claims to future work.

5 Conclusion

In this paper, we point out the shortcomings of overlap-based metrics for evaluating LLM memorization, arguing that they fail to capture the continuity of memorization essential for practical utility. To address this, we introduce Effective Memorization Length (EML), a metric that defines the boundary between faithful recall and open-ended generation. By developing a robust, reference-free Ensemble Memorization Classifier, we demonstrate that this boundary can be accurately estimated solely from the model’s internal states and logits, eliminating the dependency on ground-truth verification during inference.

Our framework transforms memorization from an unpredictable phenomenon into a controllable attribute. Through Confidence-Aware Generation Control, we show that users can dynamically truncate sequences to ensure local fidelity, effectively retaining reliable suffixes. Furthermore, we establish that the memorization boundary is not static; by integrating our classifier as a reward model within a reinforcement learning framework, we successfully extend the model’s reliable recall capabilities and show that this signal can guide optimization in a limited, setting-dependent regime.

Limitations

First, our proposed Ensemble Memorization Classifier relies on accessing the internal hidden states and logits of the LLM to construct the feature vector. Consequently, our framework is currently applicable only to open-source models or systems. It cannot be directly applied to closed-source LLMs (e.g., GPT-4, Gemini) accessed exclusively via black-box APIs that do not expose internal representations.

Besides, our experiments were conducted on the GPT-Neo and Pythia model families with parameter sizes up to 2.8B. While these models are representative of standard LLM architectures, we have not verified the scalability of our approach on newer or significantly larger models (e.g., 70B+ parameters) due to the constraint of open-sourced corpus. The reported evidence is therefore limited to these auditable families and should not be interpreted as zero-shot transfer or broad generalization to newer models. Furthermore, our study focuses on pre-trained base models. The dynamics of effective memorization in Instruction-Tuned or RLHF-aligned models, which may exhibit differ-

ent probability distributions and refusal behaviors, remain to be explored.

Finally, as discussed in Section 4.5, our RL enhancement strategy via soft prompt tuning is an extraction mechanism, not a knowledge injection method. It is limited by the frozen weights of the underlying LLM. While it may stabilize the retrieval of weakly memorized sequences in some settings, it cannot force the model to recall information that was never effectively encoded during the pre-training phase. Therefore, we view it as an analytical tool for studying and auditing existing behavior rather than as a method for expanding model knowledge.

Acknowledgement

This work is supported in part by the National Science Foundation (NSF) grant IIS-2451436 and Commonwealth Cyber Initiative grant HC-2Q26-032.

Ethical considerations

The research presented in this paper deals with the detection, control, and enhancement of training data memorization in Large Language Models. We acknowledge that the techniques developed herein possess a dual-use nature and carry ethical implications regarding privacy and copyright.

Our proposed Ensemble Memorization Classifier and Confidence-Aware Generation Control strategies significantly improve the precision of extracting memorized sequences. While our primary motivation is to study the mechanisms of memorization and enable controlled retrieval, we recognize that malicious users could potentially adapt these methods to extract personally identifiable information or sensitive data from proprietary models more efficiently than standard random-prefix attacks. The RL stress test introduces an additional dual-use risk: optimizing against the classifier reward could increase exposure of memorized content in some settings.

However, we argue that the development of precise extraction tools is a prerequisite for effective defense. Current "black-box" risks arise precisely because developers cannot accurately distinguish between safe generation and data leakage. By providing a tool that accurately estimates the "Memorization Boundary", our framework serves as a powerful utility that allows model developers to implement confidence-based truncation mechanisms to

prevent the unintended leakage of sensitive suffixes during deployment, acting as a safeguard rather than an exploit.

Besides, our case studies in the Experiments section highlight that effective memorization often involves copyrighted materials, such as open-source code licenses and literary works. While our work demonstrates the ability to extend the generation of such content, we emphasize that this is done to highlight the extent of model vulnerability. We do not condone the use of our methods for infringing on intellectual property rights. All experiments in this study were conducted using open-source dataset and open-weights models. We did not target any private models or utilize non-public data.

References

- Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. 2016. [Concrete problems in ai safety](#). *Preprint*, arXiv:1606.06565.
- Sotiris Anagnostidis, Gregor Bachmann, Lorenzo Noci, and Thomas Hofmann. 2023. The curious case of benign memorization. In *The Eleventh International Conference on Learning Representations*. OpenReview.
- Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. 2002. [Finite-time analysis of the multiarmed bandit problem](#). *Machine Learning*, 47:235–256.
- Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O’Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, and 1 others. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.
- Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramèr, and Chiyuan Zhang. 2022. Quantifying memorization across neural language models. In *The Eleventh International Conference on Learning Representations*.
- Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, and 1 others. 2021. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.
- Thomas Coste, Usman Anwar, Robert Kirk, and David Krueger. 2024. Reward model ensembles help mitigate overoptimization. In *The Twelfth International Conference on Learning Representations*.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. 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 4171–4186.
- Yijiang River Dong, Hongzhou Lin, Mikhail Belkin, Ramon Huerta, and Ivan Vulić. 2024. Unmemorization in large language models via self-distillation and deliberate imagination. *arXiv preprint arXiv:2402.10052*.
- Jacob Eisenstein, Chirag Nagpal, Alekh Agarwal, Ahmad Beirami, Alex D’Amour, DJ Dvijotham, Adam Fisch, Katherine Heller, Stephen Pfohl, Deepak Ramachandran, Peter Shaw, and Jonathan Berant. 2024. [Helping or herding? reward model ensembles mitigate but do not eliminate reward hacking](#). *Preprint*, arXiv:2312.09244.
- OpenFOAM Foundation. 2025. The openfoam license. <https://github.com/OpenFOAM/OpenFOAM-dev/blob/master/src/atmosphericModels/atmosphericTurbulentTransportModels.C>. Accessed: 2025-12-29.
- 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](#). *Preprint*, arXiv:2101.00027.
- Google-research. 2022. [lm-extraction-benchmark](#). Accessed: 2024-12-02.
- Daphne Ippolito, Florian Tramèr, Milad Nasr, Chiyuan Zhang, Matthew Jagielski, Katherine Lee, Christopher Choquette Choo, and Nicholas Carlini. 2023. Preventing generation of verbatim memorization in language models gives a false sense of privacy. In *Proceedings of the 16th International Natural Language Generation Conference*, pages 28–53.
- Jinwoo Jeong, Seungsu Baek, and Jeongseob Ahn. 2023. Fast and efficient model serving using multi-gpus with direct-host-access. In *Proceedings of the Eighteenth European Conference on Computer Systems*, pages 249–265.
- Antonia Karamolegkou, Jiaang Li, Li Zhou, and Anders Søgaard. 2023. Copyright violations and large language models. *arXiv preprint arXiv:2310.13771*.
- Aly M Kassem, Omar Mahmoud, Niloofar Mireshghalah, Hyunwoo Kim, Yulia Tsvetkov, Yejin Choi, Sherif Saad, and Santu Rana. 2024. Alpaca against vicuna: Using llms to uncover memorization of llms. *arXiv preprint arXiv:2403.04801*.
- Kubernetes. 2025. Common parameters section from kubernetes reference. <https://dev-k8sref-io.web.app/docs/common-parameters/common-parameters-/>. Accessed: 2025-12-29.

- Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.
- Cecily Mauran. 2023. Whoops, samsung workers accidentally leaked trade secrets via chatgpt. *Mashable [online]*. Dostupné z: <https://mashable.com/article/samsungchatgpt-leak-details>.
- Mustafa Ozdayi, Charith Peris, Jack FitzGerald, Christophe Dupuy, Jimit Majmudar, Haidar Khan, Rahil Parikh, and Rahul Gupta. 2023. Controlling the extraction of memorized data from large language models via prompt-tuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1512–1521.
- Kaustubh Ponshe, Parjanya Prajakta Prashant, and Babak Salimi. Tokenswap: A lightweight method to disrupt memorized sequences in llms. In *Generative and Protective AI for Content Creation*.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.
- stackspin. 2025. Code from local path provisioner. <https://open.greenhost.net/stackspin/local-path-provisioner/-/blob/v0.0.15/vendor/k8s.io/api/certificates/v1beta1/generated.pb.go>. Accessed: 2025-12-29.
- Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. 2022. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35:38274–38290.
- Wenhao Wang, Muhammad Ahmad Kaleem, Adam Dziedziec, Michael Backes, Nicolas Papernot, and Franziska Boenisch. 2024a. Memorization in self-supervised learning improves downstream generalization. In *The Twelfth International Conference on Learning Representations*.
- Zhepeng Wang, Runxue Bao, Yawen Wu, Jackson Taylor, Cao Xiao, Feng Zheng, Weiwen Jiang, Shangqian Gao, and Yanfu Zhang. 2024b. Unlocking memorization in large language models with dynamic soft prompting. In *EMNLP*.
- Alexander Xiong, Xuandong Zhao, Aneesh Pappu, and Dawn Song. 2025. [The landscape of memorization in llms: Mechanisms, measurement, and mitigation](#). *Preprint*, arXiv:2507.05578.
- Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. 2020. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142.
- Zhenhong Zhou, Jiuyang Xiang, Chaomeng Chen, and Sen Su. 2024. Quantifying and analyzing entity-level memorization in large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 19741–19749.

A Detailed Experimental Setup

Training and Evaluation Settings. We generate classifier examples with greedy decoding and partition the 15,000-example benchmark training split into 14,000 classifier-training and 1,000 held-out validation examples; the benchmark validation split is used only for final evaluation. The frozen base LLM supplies final-layer hidden states and next-token probabilities. Each of the three causal classifiers is trained for 100 epochs with batch size 32, AdamW, and positive-class-weighted binary cross-entropy. The selected learning rates are 10^{-4} for GPT-Neo-125M and Pythia-410M, 5×10^{-5} for GPT-Neo-1.3B and Pythia-1.4B, and 2×10^{-5} for GPT-Neo-2.7B and Pythia-2.8B. Starting from seed 42, the three bootstrap sampling seeds are 42, 43, and 44; all members share the held-out validation and test sets. We use the sample standard deviation, $\lambda = 1.414$, and clip the aggregated scores to $[0, 1]$. The classifier experiments use one NVIDIA A6000 GPU with 48GB memory and take less than four hours per run. For Reinforcement Learning of memorization, we conduct experiments with a single NVIDIA A100 GPU with 80GB Memory. We fine-tuned the soft prompt generator with a batch size of 64 and an AdamW optimizer for 30 epochs. We tried the learning rate from the range of $[10^{-5}, 10^{-7}]$ for stability.

Artifact Statement. The source code of our method is implemented with PyTorch and HuggingFace Accelerate (HF). All the codes and datasets we utilize are public and open-sourced. They support the usage in research and we use them for research purpose only. We did not check whether the used data contains any information that names or uniquely identifies individual people or offensive content and left this work to the institution that released the data.

Table 4: Three-seed RL stability analysis on Pythia-410M. The SFT checkpoint is fixed across runs; confidence intervals are computed over the three seed-level differences.

Metric	Fixed SFT	RL mean $\pm$ std	Mean Δ (95% CI)
Frac ER	0.5008	0.5330 $\pm$ 0.0040	+0.0322 [+0.0223, +0.0421]
Exact ER	0.3100	0.3517 $\pm$ 0.0045	+0.0417 [+0.0305, +0.0529]
EML	22.506	24.421 $\pm$ 0.187	+1.915 [+1.451, +2.379]

B RL Stability on Pythia-410M

To assess the variability of the RL process, we reran the Pythia-410M intervention from the same fixed SFT checkpoint with three independent random seeds (42, 43, and 44). Table 4 reports the fixed SFT baseline, the mean $\pm$ sample standard deviation across RL seeds, and the mean seed-level difference with its two-sided 95% Student- t confidence interval. The intervals exclude zero for all three metrics, indicating consistent changes across these RL seeds.

This analysis supports the claim that the classifier reward provides a usable optimization signal in the tested Pythia-410M setting.

C Cumulative-Label Formulation and Ablation

Motivation and definitions. The classifier in Section 3.2 is trained with a local token-match label at each generated position,

$$m_t = \mathbb{I}[\hat{y}_t = y_t], \quad (9)$$

where m_t records whether the token at position t matches the corresponding ground-truth token. A natural alternative is a cumulative boundary label,

$$m_t^{\text{cum}} = \prod_{i=1}^t m_i = \mathbb{I}[\hat{\mathbf{y}}_{1:t} = \mathbf{y}_{1:t}]. \quad (10)$$

The cumulative label is one only while the generated prefix remains an exact match; after the first mismatch, it is zero at every subsequent position. It therefore places the first-error semantics directly in the supervision targets. In contrast, our local labels provide independent supervision at every position, including positions after the first mismatch.

Relationship to the boundary operator. We retain local labels because the classifier is used to produce token-level confidence scores for both boundary quantities. The first-error semantics required by EML are imposed at inference by the first-failure stopping operator in Equation (5), rather than by

Table 5: Local token-match labels versus cumulative boundary labels on GPT-Neo-1.3B.

Label	F1-Score	AUC	Boundary MAE	Over-estimation rate
Local m_t	0.8100	0.8980	9.490	12.3%
Cumulative m_t^{cum}	0.7782	0.8891	14.476	10.0%

forcing every post-error training target to zero. This separation lets the classifier learn the local transition signal while Strict Boundary Length converts the resulting scores into a contiguous retained prefix. The Relaxed Boundary Length likewise uses the same token-level scores but intentionally does not apply first-failure stopping.

Experimental protocol. To isolate the effect of the target definition, we repeat the classifier training and boundary evaluation on GPT-Neo-1.3B with the same feature inputs, architecture, data splits, and optimization settings, changing only the labels from m_t to m_t^{cum} . We report token-level F1 and AUC together with boundary mean absolute error (MAE) and the rate of over-estimated boundaries relative to L^* . The comparison is summarized in Table 5.

Comparison and interpretation. Relative to cumulative labels, local labels improve F1-Score and AUC and substantially reduce boundary MAE. Cumulative labels slightly reduce the over-estimation rate, consistent with their tendency to assign zero to all positions after the first mismatch; however, this conservative behavior does not yield more accurate boundaries overall, as reflected by the larger MAE. The ablation therefore supports local token-match supervision as the default for learning a token-level confidence signal, while the strict stopping operator remains responsible for converting that signal into a first-error boundary. Neither labeling scheme by itself establishes a causal source of an exact match.

D The Fidelity-Length Trade-off on Pythia

Figure 6 illustrates the trade-off between the resulting Fractional Extraction Rate and the Strict/Relaxed Boundary Length on Pythia family. We have the same conclusion as the ones in Section 4.3. Our methods succeed in maintaining high-confidence of memorization while retaining suffixes as long as possible.

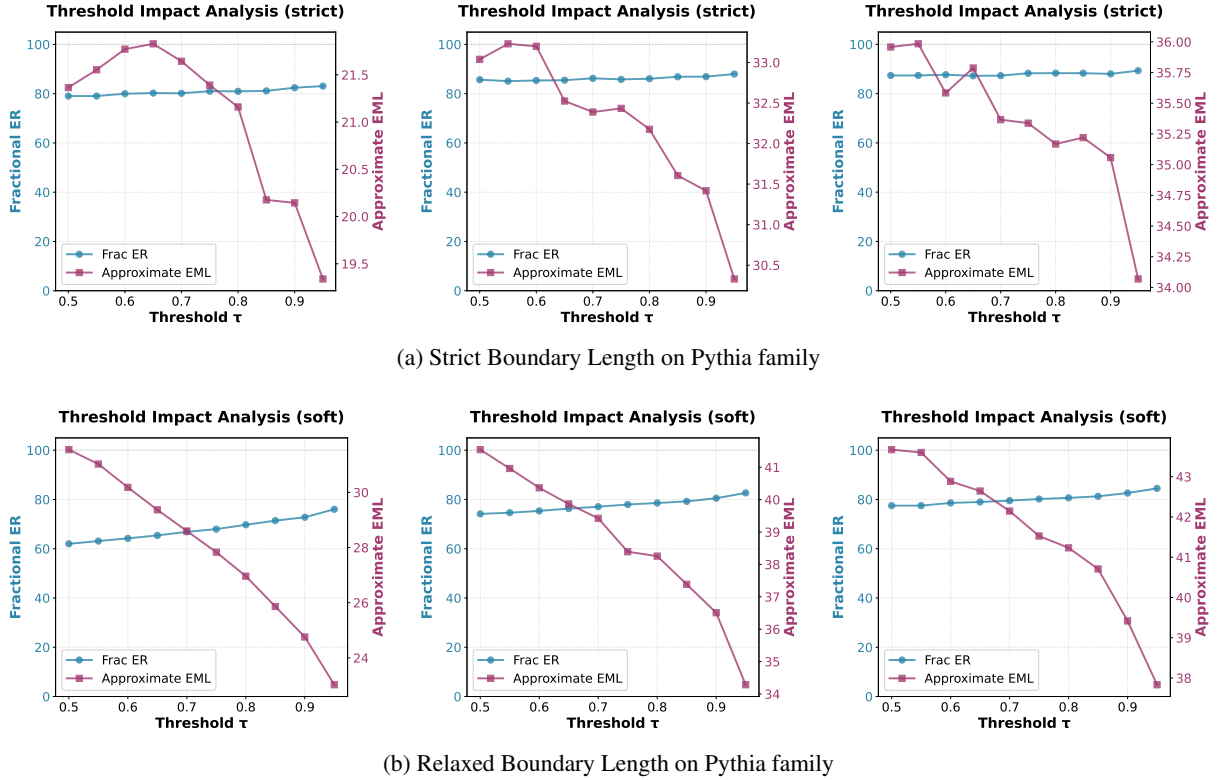


Figure 6: Trade-off between Fractional Extraction Rate (Frac ER) (Left Y-axis) and the Strict/Relaxed Boundary Length (Approximate EML) (Right Y-axis) across varying confidence thresholds τ on Pythia family

E False Positive Rate on Non-Memorized Data

A critical concern regarding any reference-free memorization detector is the potential for high False Positive Rates (FPR). Specifically, the classifier might incorrectly tag standard generalizations, highly fluent text, or hallucinations as "memorized." To strictly evaluate this, we tested our classifier on a completely unseen, non-memorized dataset.

We utilized the English translation texts from the WMT24++ (en-de_DE) dataset. Given that this dataset consists of recent translations, it is strictly out-of-distribution and highly unlikely to have been included in the pre-training corpora of the GPT-Neo and Pythia families. We generated hidden states and evaluated them with our trained classifiers. The results demonstrate exceptionally low False Positive Rates across all model scales: 0.14% for GPT-Neo 125M, 0.85% for GPT-Neo 1.3B, and 1.43% for GPT-Neo 2.7B; 0.45% for Pythia 410M, 1.03% for Pythia 1.4B, and 1.62 % for Pythia 2.8B.

This remarkably low FPR confirms that our classifier successfully captures the unique internal hidden state signatures specific to exact verbatim re-

call, ensuring that the detected Trust Boundary is highly reliable.

F Ablation Study

Classifier Architecture. To validate the necessity of using a sequence-aware architecture for our Token-Level Memorization Classifier (Section 3.2), we compared our proposed 3-layer Transformer block against a standard Multi-Layer Perceptron (MLP) baseline. Table 6 presents the performance comparison on the test set. The Attention-based architecture significantly outperforms the MLP baseline across all metrics, with a notable improvement in F1-Score and AUC. The performance gap highlights the sequential nature of effective memorization. The MLP structure struggles to distinguish between a confident open-ended generation and genuine memorization because both may exhibit high local probability and sharp hidden states. However, the internal state trajectory of a non-memorized token often diverges subtly from that of recalled content steps before the error becomes visible. By leveraging self-attention, our model aggregates historical context, effectively judging whether the current token fits into the coherent "memory trajectory" established by the prefix. This confirms that

Table 6: Evaluation results of classifiers with different architecture on GPT-Neo and Pythia families

Model	Architecture	Accuracy	F1-Score	AUC
GPT-Neo 125M	MLP	0.8314	0.5314	0.9376
	Ours	0.9108	0.6375	0.9682
GPT-Neo 1.3B	MLP	0.8572	0.7612	0.9241
	Ours	0.8821	0.8165	0.9496
GPT-Neo 2.7B	MLP	0.8548	0.7905	0.9146
	Ours	0.8859	0.8416	0.9409
Pythia 410M	MLP	0.8110	0.6160	0.9002
	Ours	0.8484	0.7094	0.9324
Pythia 1.4B	MLP	0.8374	0.7872	0.8899
	Ours	0.8417	0.8217	0.8998
Pythia 2.8B	MLP	0.8441	0.8179	0.8786
	Ours	0.8641	0.8454	0.8852

detecting effective memorization requires modeling the continuity of the generation process, rather than assessing token quality in isolation.

Ensemble Efficacy. To validate the necessity of our ensemble design (Section 3.3), we compare the performance of a single classifier trained on the full dataset against the aggregated predictions of our bootstrap ensemble ($K = 3$). As shown in Table 7, the Ensemble strategy yields consistent improvements across all metrics. Most notably, it achieves a higher AUC compared to the single model.

The superiority of the ensemble approach stems from two key factors. First, individual classifiers can be sensitive to noise or specific patterns in their training fold, leading to overconfident predictions on non-memorized or underconfident predictions on exact-match tokens. The ensemble averages out these idiosyncrasies, providing a smoother and more robust signal. Besides, in our "Confidence-Aware Generation Control" task, a single false negative on an exact-match token can cause premature truncation. We observed that tokens with high disagreement among ensemble members (high σ_t) often correspond to "ambiguous" memorization states—such as synonyms or correct facts expressed with different syntax. The UCB mechanism uses this disagreement to reduce premature stopping, although, as discussed in Section 3.4, its optimism may also increase over-estimation risk.

Sensitivity to Prefix Length. To understand the dynamics of memorization detection, we conducted an ablation study on the sensitivity of our classifier to the provided prefix length. Results are shown in Table 8.

The classifier’s performance improves consistently as the prefix length increases. This is because

Table 7: Single-classifier versus ensemble-classifier results on the GPT-Neo and Pythia families

Model	Method	Accuracy	F1-Score	AUC
GPT-Neo 125M	Single	0.8985	0.6277	0.9584
	Ours ($K = 3$)	0.9108	0.6375	0.9682
GPT-Neo 1.3B	Single	0.8709	0.8070	0.9331
	Ours ($K = 3$)	0.8821	0.8165	0.9496
GPT-Neo 2.7B	Single	0.8750	0.8344	0.9281
	Ours ($K = 3$)	0.8859	0.8416	0.9409
Pythia 410M	Single	0.8492	0.6953	0.9230
	Ours ($K = 3$)	0.8484	0.7094	0.9324
Pythia 1.4B	Single	0.8407	0.8166	0.8665
	Ours ($K = 3$)	0.8417	0.8217	0.8998
Pythia 2.8B	Single	0.8586	0.8384	0.8783
	Ours ($K = 3$)	0.8641	0.8454	0.8852

Table 8: Ablation study on prefix length

Prefix len	Frac ER	Exact ER	Accuracy	F1-Score	AUC
5	0.086	0.003	0.8922	0.3363	0.8739
10	0.150	0.030	0.8635	0.3954	0.8757
15	0.209	0.074	0.8332	0.4532	0.8813
25	0.353	0.185	0.8159	0.5714	0.9046
50	0.687	0.515	0.8859	0.8416	0.9409

a longer prefix provides a more established context, guiding the LLM deeper into a specific "trajectory" within its memorized manifold. Consequently, the internal hidden states exhibit a stronger, less ambiguous signature of memorization, making it significantly easier for the classifier to distinguish verbatim recall from generic generation. This finding further corroborates our hypothesis: memorization in LLMs is not merely a token-level anomaly but a continuous, sequential phenomenon that becomes increasingly pronounced with sufficient context.