Rebalancing Token Importance in Language Models with TF-IDF Weighted Cross-Entropy Loss

Anonymous ACL submission

Abstract

Large language models are typically trained under uniform token weighting, which allows frequent and low-information tokens to dominate learning and can increase the tendency to memorize surface-level text spans. To address this, we present an information-weighted cross-entropy loss that rescales token-level contributions using TF-IDF statistics, emphasizing semantically informative tokens while down-weighting ubiquitous ones. Experiments on five decoder-only LLMs ranging from 1.1B to 13B parameters show consistent reductions in memorized substring length while preserving perplexity and downstream task performance. Under LoRA fine-tuning, TF-IDF reduces average substring memorization length by 14% across all five models. Under full-weight fine-tuning on TinyLLaMA 1.1B, the reduction reaches 58%. Our approach is architecture-agnostic and can be incorporated into existing training pipelines with less than 3% computational overhead, offering a lightweight and principled way to mitigate memorization without disrupting standard training dynamics.

1 Introduction

Large language models (LLMs) are increasingly deployed in domains where reliability (e.g., clinical decision support), robustness (e.g., legal document analysis), and data stewardship (e.g., copyright-sensitive content generation) are essential. Yet, their training objectives still treat all tokens equally, regardless of how informative, redundant, or noisy those tokens may be (Touvron et al., 2023; Brown et al., 2020). This uniform weighting can misallocate gradient updates, leading to overemphasizing frequent or low-information patterns while underrepresenting distinctive or semantically-rich content (Su et al., 2024). This can lead to memorization of surface-level patterns in the training data (Carlini et al., 2023). When every token is weighted equally, the model has no incentive to treat repeated surface patterns differently from meaningful content, leaving it prone to memorizing sequences verbatim.

Existing interventions include pre-training data deduplication (Kandpal et al., 2022), which requires costly global pre-computation, and privacy-preserving objectives such as differential privacy (Abadi et al., 2016), which trade off model utility. Post-hoc methods such as unlearning (Eldan and Russinovich, 2023) and model editing (Meng et al., 2022) operate after training but do not address how the model learns during training. Token-dropping methods (Hans et al., 2024) offer a lighter-weight alternative but remove supervision on a subset of tokens rather than reweighting all of them. We instead focus on reshaping how the gradient signal is distributed across tokens during training. Our method is designed for the continued pretraining and fine-tuning regime, the dominant paradigm in modern NLP where practitioners adapt publicly released pretrained checkpoints rather than training from scratch (Bommasani et al., 2021; Howard and Ruder, 2018).

We present a TF-IDF-weighted cross-entropy loss that rescales token-level gradients using lexical statistics from the training corpus. The method up-weights tokens that are distinctive within context and down-weights those that are globally ubiquitous. Meanwhile, we retain supervision for all tokens, preserving coherent sequence modeling while reducing the incentive to memorize exact surface forms. The design is straightforward, architecture-agnostic, and readily integrates into existing training pipelines.

We evaluate this technique across five pretrained decoder-only LLMs ranging from 1.1B to 13B parameters. Empirically, the TF-IDF-weighted loss preserves perplexity and downstream performance on summarization and question answering (QA), while consistently reducing substring-level memorization and ROUGE-L across all models. The mag-

nitude of the reduction is driven by how much memorization occurs under the CE baseline. Smaller models fine-tuned with LoRA already memorize little, leaving less room for improvement in that setting. Under full-weight fine-tuning, where the CE baseline memorizes substantially more, even the smallest model — TinyLLaMA 1.1B — achieves the largest reduction of 58% in average Longest Memorized Substring (LMS). Across all five LoRA models the average reduction is 14%, indicating that TF-IDF loss consistently reduces memorization wherever the baseline permits it. Together, these results demonstrate that token-aware objectives provide an efficient and principled means of mitigating memorization without altering model architectures, standard training procedures, or downstream task performance.

2 Related Work

We review three areas of related work that inform and contextualize our approach: memorization in LLMs and existing mitigation strategies, regularization and alternative training objectives, and token-weighted loss functions.

2.1 Memorization and Generalization in Large Language Models

LLMs are known to verbatim memorize portions of their training data (Carlini et al., 2019, 2021). Prior research has established various metrics to quantify this phenomenon, most notably *exposure* and *Longest Memorized Substring (LMS)* (Yeom et al., 2018; Kandpal et al., 2022). Recent controlled studies suggest that such memorization is not merely a byproduct of model capacity, but is closely tied to the frequency of sequence repetition and the concentration of gradient updates on specific spans during training (Huang et al., 2024).

Current mitigation strategies typically operate at the data level through deduplication and filtering (Kandpal et al., 2022), at the optimization level via differential privacy (Abadi et al., 2016), or through post-hoc model unlearning (Eldan and Russinovich, 2023) and model editing (Meng et al., 2022). However, these techniques often force a trade-off: they either require massive precomputation (deduplication) or dramatically degrade the model's downstream utility and reasoning capabilities. A separate line of work modifies the training objective directly: token-dropping approaches (Hans et al., 2024) randomly exclude subsets of tokens from the loss to prevent the model from completing memorized chains, achieving strong reductions particularly at pre-training scale. Our method occupies a different point in this design space — rather than removing supervision on any token, it reweights all tokens using corpus statistics, preserving full-sequence modeling while redirecting gradient pressure away from common, low-information tokens.

2.2 Regularization and Alternative Training Objectives

Parallel to data-centric approaches, a broad line of work seeks to mitigate overfitting by constraining model capacity or softening the training objective. Classical techniques such as weight decay and dropout (Krogh and Hertz, 1991; Srivastava et al., 2014) have been adapted for large transformers through methods like mixout or stochastic depth (Lee et al., 2020; Huang et al., 2016). Additionally, entropy regularization (Pereyra et al., 2017) and label smoothing (Müller et al., 2019) discourage overconfidence by preventing the model from assigning total probability mass to a single token.

While effective at improving general robustness, these methods operate primarily at the model or sequence level. They treat the loss landscape as uniform, without accounting for the fact that some tokens are inherently more prone to memorization than others. While regularization constrains *how* a model learns, our method rebalances *what* the model prioritizes in learning.

2.3 Token-Weighted and Reweighted Loss Functions

Recent work has explored modifying the loss function to assign non-uniform importance across tokens or examples. Focal loss (Lin et al., 2017) down-weights easy examples in classification tasks to address class imbalance. MiLe Loss (Su et al., 2024) reweights tokens according to the model's prediction entropy, emphasizing uncertain (hard-to-learn) tokens. Selective Language Modeling (Lin et al., 2024) computes token-utility scores using a reference model and trains only on high-utility tokens, improving data efficiency. Bilevel and meta-reweighting methods (Ren et al., 2018; Pan et al., 2025) learn example weights dynamically to optimize downstream validation performance. Together, these works demonstrate the growing interest in weighting strategies that better align gradient

updates with token informativeness.

In contrast to these methods, which primarily rely on dynamic model-dependent metrics or auxiliary reference models, our TF-IDF objective utilizes static corpus statistics to establish token importance. This provides a computationally efficient and linguistically principled alternative that targets the inherent informational density of language without the overhead of secondary models or iterative meta-optimization. To our knowledge, applying TF-IDF statistics as a token-level loss weighting scheme for language model training has not been previously explored.

3 TF-IDF Weighted Loss

We present a **TF-IDF-weighted cross-entropy** objective that retains supervision on every token but rescales each term by a token-specific weight w_i:

$$\mathcal{L}_{\text{TF-IDF}}(\theta) = -\frac{1}{L}\sum_{i=1}^{L} w_i \log P_\theta(x_i \mid x_{<i}) \quad (1)$$

In contrast, standard autoregressive language models minimize cross-entropy (CE) loss over a token sequence $X = (x_1, \ldots, x_L)$, corresponding to the special case $w_i = 1$:

$$\mathcal{L}(\theta) = -\frac{1}{L}\sum_{i=1}^{L} \log P_\theta(x_i \mid x_{<i}) \quad (2)$$

Computing IDF over the full training corpus — which for LLMs can span billions of tokens — would be prohibitively expensive; we instead maintain a rolling buffer of recent mini-batches to approximate corpus-level statistics efficiently. We then illustrate how these weights redirect gradient pressure toward semantically informative tokens and away from the common surface tokens that drive verbatim memorization.

3.1 Buffer-Averaged TF-IDF Statistics

Weights w_i are computed using local TF and smoothed IDF estimated over an accumulation buffer of $K = 16$ mini-batches ($N = B \times K$ sequences, where B is the per-device batch size). This buffer size balances re-weighting aggressiveness with statistical stability. A larger N widens the IDF gap between ubiquitous stop words ($df \approx N$) and rare keywords ($df = 1$), shifting more gradient mass toward informative tokens. We find $K = 16$ provides a large enough window for robust frequency estimation while avoiding the overhead of global pre-computation. The TF component further amplifies weight for tokens that repeat within a sequence, capturing locally salient terms whose within-context density signals domain relevance beyond what IDF alone conveys.

Term Frequency (TF). tf_i counts occurrences of token x_i within its sequence:

$$\mathrm{tf}_i = |\{j \in \{1, \ldots, L\} : x_j = x_i\}| \quad (3)$$

Document Frequency and IDF. Over the buffer of N sequences, $\mathrm{df}(v)$ counts how many sequences contain token v, where $X^{(n)}$ denotes the n-th sequence and $\mathbb{I}(\cdot)$ is the indicator function. We apply smoothed IDF (Schütze et al., 2008) to keep weights positive even for near-universal tokens:

$$\mathrm{df}(v) = \sum_{n=1}^{N} \mathbb{I}(v \in X^{(n)}) \quad (4)$$

$$\mathrm{idf}(v) = \log\left(\frac{1+N}{1+\mathrm{df}(v)}\right) + 1 \quad (5)$$

Weight Normalization. Raw TF-IDF weights are computed as $w'_i = \mathrm{tf}_i \cdot \mathrm{idf}(x_i)$ and normalized to unit mean over the M non-padding tokens in the mini-batch, ensuring $\mathbb{E}[w] = 1$ so that standard learning rates remain applicable:

$$w_i = \frac{w'_i}{\frac{1}{M}\sum_{j=1}^{M} w'_j} \quad (6)$$

Together, these components allow the model to internalize the relative importance of tokens during training, rebalancing gradient pressure in a way that reduces the incentive to memorize exact surface forms.

3.2 Weighting Dynamics

Consider a mini-batch drawn from diverse domains. In a sequence about scientific theory shown in Figure 1, tokens like *relativity* and *gravity* appear frequently within that context but rarely in other sequences, so they receive high TF-IDF weights and the model is pushed to predict them accurately. Function words like *the* and *and* appear in nearly every sequence and are down-weighted, shifting gradient mass toward semantically meaningful content. In noisy datasets, common formatting symbols and boilerplate text appear across many sequences, giving them high document frequency and low IDF weight, similar to function words.

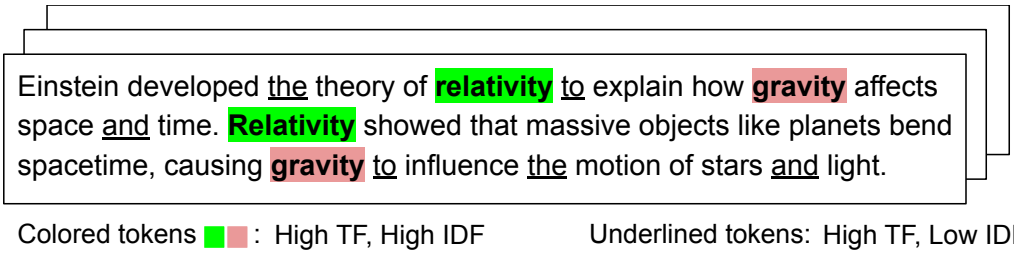


Figure 1: Illustration of TF-IDF weights applied to a short paragraph. Informative tokens such as *relativity*, and *gravity* receive higher weights, while common words like *the* and *and* are de-emphasized.

The TF-IDF-weighted objective retains all tokens in the loss, rebalancing where gradient pressure falls rather than removing supervision on any position. This discourages memorization because gradient signal is no longer uniformly spread across all positions. Pressure concentrates on tokens that are locally frequent and globally rare, the distinctive content words that TF-IDF upweights, rather than on the common surface tokens that define a sequence's exact wording.

4 Experimental Setup

4.1 Models

We assess five commonly used decoder-only LLMs spanning 1.1B–13B parameters: **TinyLLaMA 1.1B** (Zhang et al., 2024), a compact LLaMA-style model; **Pythia 1.4B** (Biderman et al., 2023), trained on The Pile with transparent checkpoints; **GPT-J 6B** (Wang and Komatsuzaki, 2021), a widely adopted open-source model also trained on The Pile; **LLaMA-2 7B** (Touvron et al., 2023), a strong mid-size foundation model; and **LLaMA-2 13B** (Touvron et al., 2023), its larger sibling. This range enables testing generalization of our method across model scales. We do not include smaller models, as they often produce outputs too short or simplistic for meaningful memorization and downstream evaluation.

4.2 Fine-Tuning Setup

Consistent with our focus on the continued pretraining and fine-tuning regime, all experiments initialize from publicly released pretrained checkpoints. This reflects how LLM practitioners typically operate (Bommasani et al., 2021; Howard and Ruder, 2018): starting from a strong pretrained foundation and adapting it to new objectives or domains, rather than training from scratch.

For our primary experiments across the full model suite (1.1B–13B parameters), we use Low-Rank Adaptation (LoRA) (Hu et al., 2022), which adds trainable low-rank matrices to the attention projection layers while keeping the original weights frozen. Prior work shows LoRA can match or exceed full fine-tuning on language modeling and downstream tasks (Hu et al., 2022; Dettmers et al., 2023; Lialin et al., 2023), and it remains practical at all model scales we consider. We use rank r=8 and scaling factor α=32 (memorization and QA) or α=16 (summarization), with a learning rate of 1×10^{-4} and the AdamW optimizer. Per-model details including target modules, quantization, and batch sizes are provided in Appendix A.

To verify that our findings are not artifacts of the LoRA parameterization, we also run full-weight fine-tuning on TinyLLaMA 1.1B, updating all model parameters. We use a reduced learning rate of 5×10^{-5}, as applying the same LoRA learning rate to full-weight training caused catastrophic memorization.

4.3 Evaluation Categories and Datasets

We evaluate across four categories: *Memorization*, *Perplexity*, *Summarization*, and *QA*. For each model, training and decoding protocols are identical across conditions and only the loss function differs. All generation-based evaluations use deterministic greedy decoding.

Memorization. To evaluate verbatim recall, we adopt the *controlled injection* framework utilized by Huang et al. (2024) to study memorization in modern LLMs. We construct a training corpus consisting of 20,000 base sequences from the Pile-uncopyrighted dataset (Gao et al., 2020), into which we inject 100 target sequences from WikiText-2 (Merity et al., 2017). Each target sequence is repeated 10 times at random positions to simulate data duplication. We initialize from pretrained checkpoints and train for one epoch using LoRA (batch size 8, lr 1×10^{-4}, block size 256). For full-weight fine-tuning on TinyLLaMA 1.1B, we use the same corpus and epoch count with all

Model	Objective	Avg Prefix	Max Prefix	Avg LMS	Max LMS	ROUGE-L
TinyLLaMA 1.1B†	CE	0.00	0	3.23	10	17.5
	TF-IDF	0.00	0	**3.22**	10	**17.1**
TinyLLaMA 1.1B‡	CE	5.65	128	8.30	128	21.05
	TF-IDF	**1.36**	**18**	**3.50**	**18**	**13.55**
Pythia 1.4B	CE	0.63	6	2.44	9	17.54
	TF-IDF	**0.57**	**5**	**2.07**	**7**	**17.31**
GPT-J 6B	CE	1.08	10	4.46	21	20.3
	TF-IDF	**0.00**	**0**	**3.55**	**10**	**19.2**
LLaMA-2 7B	CE	1.37	14	3.48	14	18.92
	TF-IDF	**1.02**	**7**	**2.98**	**10**	**17.68**
LLaMA-2 13B	CE	1.82	14	3.90	14	19.79
	TF-IDF	**1.17**	**11**	**3.17**	**11**	**18.40**

†LoRA fine-tuning. ‡Full-weight fine-tuning.

Table 1: **Memorization Metrics across Models and Objectives.** Comparison between standard cross-entropy (CE) and TF-IDF-weighted cross-entropy. For all reported metrics, **lower values** indicate superior mitigation of verbatim recall. LMS (Longest Memorized Substring) quantifies the length of the longest exact token substring recovered from the training set, while prefix matches represent exact recall triggered by the start of a sequence. All metrics are averaged across prefix lengths $\mathcal{P} = \{32, 50, 100\}$.

parameters updated (batch size 8, block size 256).

At evaluation time, we probe the model's recall using prefix lengths $\mathcal{P} = \{32, 50, 100\}$ tokens. For a given $p \in \mathcal{P}$, the model is conditioned on the first p tokens and asked to continue. We evaluate checkpoints saved at $10\%, 25\%, 50\%, 75\%, 100\%$ of training progress to observe how memorization evolves throughout training, not only at the final checkpoint.

For each (sequence, p) pair, we compare the generated continuation to the ground truth using three metrics, where lower values indicate less memorization: (1) *Longest prefix match* counts consecutive token matches from the start of the continuation; (2) *Longest Memorized Substring (LMS)* (Huang et al., 2024) measures the longest exact token substring shared between the generation and the ground truth; and (3) *ROUGE-L* (Lin, 2004) captures broader structural similarity.

Perplexity. We measure token-level perplexity on an unseen corpus using the same checkpoints from the memorization setup. Perplexity is evaluated on the *WikiText-2 validation* split, which is disjoint from the injected training data. We tokenize the full validation set, concatenate tokens into non-overlapping blocks of 256, and compute:

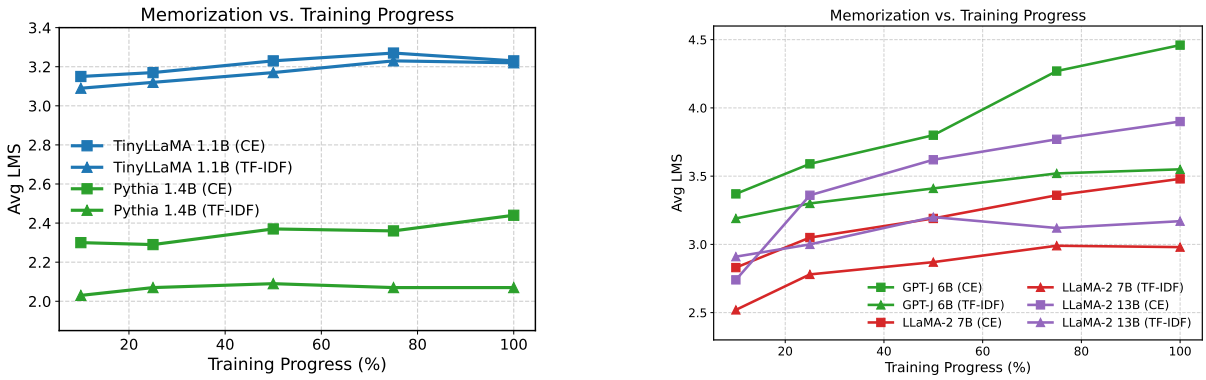
$$\text{PPL} = \exp\left(\frac{1}{|\mathcal{D}|}\sum_{t\in\mathcal{D}} -\log P_\theta\big(x_t \,\big|\, x_{<t}\big)\right)$$

where $\mathcal{D}$ indexes all evaluated token positions. This isolates generalization to unseen data while keeping the training setup identical across models and loss variants.

Summarization. We fine-tune models on the CNN/DAILYMAIL v3.0.0 train split and evaluate on validation (Hermann et al., 2015), using an instruction-style prompt ("`summarize: `") to elicit summary highlights. Models are trained for 1 epoch (batch size 4, lr 2×10^{-4}, max source length 1024, max target length 128). Quality is measured with *ROUGE-1/2/L* (Lin, 2004) for n-gram overlap and *BERTScore-F1* (Zhang et al., 2020) for semantic similarity.

QA. We fine-tune on the SQUAD v1.1 (Rajpurkar et al., 2016) train split and evaluate on validation. Models receive a context passage and a question and must generate a short answer span. Models are trained for 1 epoch (batch size 8, lr 1×10^{-4}, block size 256). We report *Exact Match* (EM), which requires character-level identity with the ground truth, and *F1 Score*, which measures word-level overlap between the predicted and reference answers.

We select summarization and QA as downstream tasks because they represent complementary dimensions of language understanding: summarization tests the ability to produce coherent, abstractive output, while QA tests precise comprehension and span extraction. Together they provide broad coverage for assessing whether TF-IDF weighting preserves general utility.



(a) Smaller models: TinyLLaMA 1.1B and Pythia 1.4B.

(b) Larger models: GPT-J 6B and LLaMA-2 7B/13B.

Figure 2: Avg LMS versus training progress under standard cross-entropy (CE) and TF-IDF-weighted objectives. Subfigure (a) shows smaller models; subfigure (b) shows mid- and large-scale models where the gap between standard CE and TF-IDF loss widens.

5 Does TF-IDF-Weighted Loss Reduce Memorization?

Table 1 and Figure 2 summarize memorization behavior under CE and TF-IDF-weighted objectives. All metrics in this section — Avg/Max Prefix, Avg/Max LMS, and ROUGE-L — measure verbatim memorization; lower values indicate less memorization and better mitigation. Under LoRA fine-tuning, no model reproduced entire passages verbatim; full-sequence matches were zero across all configurations. Prefix matches under LoRA remain modest (averaging below 2 tokens across all models) though larger models show non-trivial values (e.g., LLaMA-2 13B CE: Avg Prefix = 1.82, Max = 14). Full-weight fine-tuning on TinyLLaMA 1.1B is a more pronounced exception: CE training produces Avg Prefix = 5.65 with a maximum of 128 tokens, confirming that unconstrained parameter adaptation creates substantially more memorization pressure than LoRA. Prior work has shown that even short memorized spans can be sufficient to extract sensitive training data (Carlini et al., 2021), motivating reduction even when absolute values appear modest.

Across all five LoRA models, the average reduction in LMS is 14%. For the four LoRA-tuned models excluding TinyLLaMA, the reduction in memorization is consistent and substantial. GPT-J 6B's average LMS falls by over 20%, from 4.46 to 3.55, while its maximum span is halved from 21 to 10 tokens. LLaMA-2 7B and 13B show reductions in both average LMS (3.48 to 2.98 and 3.90 to 3.17) and maximum span (from 14 to 10 and 14 to 11, respectively). Pythia 1.4B drops from 2.44 to 2.07. ROUGE-L follows the same trend across all four models. Percentile bootstrap CIs (n=10,000 resamples, 95%) confirm statistically significant reductions for Pythia 1.4B (15.0%, CI [6.9%, 22.6%]), LLaMA-2 7B (14.4%, CI [5.6%, 22.4%]), and LLaMA-2 13B (18.6%, CI [10.4%, 25.9%]), all excluding zero. As shown in Figure 2, TF-IDF training produces lower LMS at every training checkpoint, introducing a stable downward offset without changing the rate at which memorization accumulates. This gap widens for larger models as training progresses.

TinyLLaMA 1.1B under LoRA shows a negligible difference (3.23 to 3.22), with a bootstrap CI on the relative reduction of [−8.3%, 8.3%] that includes zero, confirming this result is not statistically significant. We attribute this to LoRA's parameter constraints already suppressing memorization to a level that is equally low under both objectives, leaving little room for TF-IDF to further reduce it. Full-weight fine-tuning directly tests this: without LoRA, CE training drives average LMS to 8.30 with a maximum span of 128 tokens, far higher than the LoRA counterpart. TF-IDF then produces a 58% reduction in average LMS (to 3.50), cutting the maximum span to 18 tokens and ROUGE-L from 21.05 to 13.55. Bootstrap confidence intervals (n=10,000 resamples, 95%) confirm this reduction is statistically significant: the relative reduction of 57.8% has a CI of [34.9%, 71.1%], which excludes zero. Notably, the CE interval [5.45, 11.94] is substantially wider than the TF-IDF interval [3.18, 3.86], indicating that TF-IDF not only reduces memorization on average but also makes it more consistent across sequences. This confirms that TF-IDF's effect scales with the model's actual capacity to memorize. When LoRA constrains the

	Text
Prompt	*For several years the arsenal, which was owned by the federal government, served as a simple arms depot and was staffed with only a handful of soldiers…*
Ground truth	But in November 1860, with the American Civil War on the horizon, a company of the Second United States Artillery, consisting of sixty-five men, was transferred to Little Rock…
CE LMS=128, R-L=100	But in November 1860, with the American Civil War on the horizon, a company of the Second United States Artillery, consisting of sixty-five men, was transferred to Little Rock…
TF-IDF LMS=2, R-L=17.1	By the time the arsenal was officially closed in 1917, the United States had a military force of over 100,000 men…

Table 2: Qualitative example: CE reproduces the injected sequence verbatim while TF-IDF generates a related but non-memorized continuation. Full-weight fine-tuning, prefix length 32; 128 tokens generated. R-L = ROUGE-L.

Model	CE ↓	TF-IDF ↓
TinyLLaMA 1.1B†	25.58	**23.59**
TinyLLaMA 1.1B‡	20.94	**16.06**
Pythia 1.4B	**17.23**	18.36
GPT-J 6B	18.80	**17.59**
LLaMA-2 7B	10.86	**10.65**
LLaMA-2 13B	10.35	**10.28**

†LoRA fine-tuning. ‡Full-weight fine-tuning.

Table 3: Perplexity on the WikiText-2 validation set for models fine-tuned with standard cross-entropy (CE) and TF-IDF-weighted objectives. Lower is better.

parameter space, both objectives reach similarly low memorization. When all parameters are free, gradient rebalancing provides strong protection.

Table 2 illustrates this contrast concretely. Given a 32-token prefix from an injected WikiText-2 sequence, the CE model reproduces the continuation verbatim (LMS=128, ROUGE-L=100). The TF-IDF model generates a topically related but clearly non-memorized continuation (LMS=2, ROUGE-L=17.1).

6 Impact on Language Modeling and Downstream Performance

Having shown that TF-IDF weighting reduces memorization, we now assess whether it preserves model utility across three dimensions: perplexity, summarization, and QA.

6.1 Perplexity

Lower perplexity indicates better generalization to unseen text. Table 3 shows that TF-IDF weighting achieves comparable or lower perplexity than CE across most models. We note that training data includes WikiText-2 injections while perplexity is measured on the WikiText-2 validation split; although these splits are disjoint, the domain overlap means perplexity differences partly reflect how each objective weights WikiText-2-style tokens rather than purely out-of-domain generalization. GPT-J 6B improves from 18.80 to 17.59, TinyLLaMA 1.1B from 25.58 to 23.59, and both LLaMA-2 models see small gains as well. Only Pythia 1.4B records a minor increase (17.23 to 18.36), the sole exception across all six configurations. Overall, TF-IDF reweighting does not degrade language modeling quality; perplexity remains stable or slightly improved across all architectures.

Under full-weight training, the improvement is more pronounced: TF-IDF achieves 16.06 versus 20.94 for CE on TinyLLaMA 1.1B, suggesting that emphasizing informative tokens provides a stronger regularization signal when all parameters are free to adapt.

6.2 Summarization

Unlike the memorization metrics above, all downstream scores in this section and the following QA section are higher-is-better, reflecting model utility rather than verbatim recall. Table 4 shows that summarization performance is stable across all models under TF-IDF weighting. Scores are nearly indistinguishable from the CE baseline in most cases. LLaMA-2 7B achieves near-identical results (33.0 vs. 32.9 ROUGE-1), while Pythia 1.4B and GPT-J 6B show small gains. TinyLLaMA 1.1B and LLaMA-2 13B show fluctuations of around 0.2–0.5 points, which are negligible in the context of overall generation quality. For full-weight TinyLLaMA 1.1B, the CE baseline and TF-IDF also yield comparable scores across all metrics. Although ROUGE is a limited proxy for abstractive quality (Bhandari et al., 2020), the agreement between ROUGE and BERTScore-F1 across all models suggests that TF-IDF weighting does not disrupt generation quality for summarization tasks.

6.3 QA

Table 5 shows that TF-IDF weighting preserves QA performance across all model scales. Differences

Model	ROUGE-1		ROUGE-2		ROUGE-L		BERTScore-F1	
	CE	TF-IDF	CE	TF-IDF	CE	TF-IDF	CE	TF-IDF
TinyLLaMA 1.1B†	**32.4**	31.9	**13.5**	13.3	**24.3**	24.0	**88.0**	87.6
TinyLLaMA 1.1B‡	**33.6**	33.4	**14.1**	14.0	**25.0**	25.0	**88.7**	88.6
Pythia 1.4B	28.5	**28.7**	16.2	**16.3**	20.7	**20.8**	84.8	84.8
GPT-J 6B	28.5	**29.0**	16.2	**16.5**	20.7	**20.9**	84.9	84.9
LLaMA-2 7B	32.9	**33.0**	18.6	18.6	23.9	23.9	84.4	84.4
LLaMA-2 13B	**28.3**	28.1	**15.0**	14.8	20.3	20.3	83.9	83.9

†LoRA. ‡Full-weight.

Table 4: Summarization performance on the CNN/DAILYMAIL validation set under standard cross-entropy (CE) and TF-IDF-weighted objectives. Higher is better.

Model	Exact Match (%)		F1 (%)	
	CE	TF-IDF	CE	TF-IDF
TinyLLaMA 1.1B†	**24.63**	23.91	47.88	**48.38**
TinyLLaMA 1.1B‡	**79.00**	78.40	**84.46**	84.18
Pythia 1.4B	**54.59**	54.26	73.92	**74.24**
GPT-J 6B	**57.51**	56.48	**77.81**	77.40
LLaMA-2 7B	**88.61**	87.46	**94.41**	94.10
LLaMA-2 13B	**89.33**	87.92	**95.04**	94.52

†LoRA. ‡Full-weight.

Table 5: QA performance on the SQUAD V1.1 validation set under standard cross-entropy (CE) and TF-IDF-weighted objectives. Higher is better.

in Exact Match (EM) and F1 are within 1.5% in all cases. TinyLLaMA 1.1B shows a small drop in EM but a small gain in F1, while Pythia 1.4B is nearly identical under both settings. LLaMA-2 13B shows the largest EM drop (-1.41%), the closest to the 1.5% threshold; F1 remains high at 94.52 versus 95.04 under CE.

Under full-weight fine-tuning, TinyLLaMA 1.1B reaches EM/F1 of 79.0/84.46 under CE and 78.4/84.18 under TF-IDF, a gap of less than 1%. The much higher absolute scores compared to LoRA (CE EM=24.63) reflect the greater capacity of full-weight fine-tuning for task adaptation, while the near-identical CE and TF-IDF results confirm that the objective does not impair extractive reasoning.

Summary. Across all three evaluation dimensions, TF-IDF weighting produces results within noise of the CE baseline. Perplexity improves for five of six model configurations. Summarization scores diverge by at most 0.5 ROUGE points. QA differences are within 1.5% EM and 0.6% F1. Together, these results confirm that consistent memorization reduction comes at no meaningful cost to downstream utility.

7 Conclusion

In this work, we address the phenomenon of verbatim memorization in LLMs by challenging the standard practice of uniform token weighting during training. We present a *TF-IDF-weighted cross-entropy objective* that rebalances the learning signal according to lexical information density, effectively prioritizing semantically rich tokens over high-frequency, low-entropy ones.

Our experiments across five decoder-only LLMs, ranging from 1.1B to 13B parameters, demonstrate that this re-weighting consistently mitigates memorization—specifically reducing the Longest Memorized Substring and ROUGE-L—without compromising linguistic fluency or downstream performance on summarization and QA tasks, while tracking memorization across training checkpoints confirms a stable downward shift rather than a mere delay in onset.

The TF-IDF-weighted objective is architecture-agnostic and introduces less than 3% computational overhead (256 ms vs. 262 ms per step on TinyLLaMA 1.1B, batch size 8, full-weight), making it directly applicable to standard training pipelines. These findings highlight the potential of token-aware objectives as a scalable, lightweight strategy for mitigating memorization in service of privacy and data stewardship. Future directions include extending evaluation to adversarial extraction, mechanistic analysis of lexical reweighting, and testing TF-IDF objectives during pretraining and domain adaptation.

Limitations

Our work has several limitations that provide context for our findings and suggest directions for future research.

Evaluation scope. Our memorization evaluation focuses on verbatim exact-match recall following established practice (Carlini et al., 2019; Huang et al., 2024), as this represents the most direct risk for privacy and copyright. It does not cover semantic paraphrasing or adversarial prompting techniques designed to extract training data.

Training regime. While most experiments use LoRA fine-tuning, we validate the method under full-weight training on TinyLLaMA 1.1B. Neither setting captures how TF-IDF weighting would interact with training from scratch, where consistently down-weighting function words could potentially hinder early acquisition of grammatical structure.

Subword tokenization. Classical TF-IDF operates at the word level, but our method applies weights to subword tokens produced by BPE tokenizers. A rare word may be split into subword units that are individually common, potentially diluting the intended upweighting of informative content. Analyzing the empirical weight distribution over subword tokens is a useful direction for future work.

Context length. Hardware constraints limited our block size to 256 tokens. This is sufficient for common memorization patterns but may not capture long-range memorization dynamics in models with larger context windows.

Ethics Statement

This work adheres to the ACL Ethics Policy. Our experiments utilize publicly available datasets (CNN/DailyMail, SQuAD, and WikiText-2) in accordance with their intended research use. By proposing a loss function that reduces verbatim memorization, this research aims to enhance the privacy and safety of LLMs.

AI Usage Disclosure

The authors used ChatGPT (OpenAI) and Claude (Anthropic) to assist in the linguistic polishing and grammatical refinement of this manuscript to improve clarity and readability. Additionally, these tools were used to assist in debugging the custom Python scripts used for the TF-IDF weighted loss implementation and the memorization evaluation pipeline. The core research objectives, the mathematical derivation of the buffer-averaged TF-IDF loss function, the experimental design, and the interpretation of all results were performed solely by the human authors.

References

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318.

Manik Bhandari, Pranav Narayan Gour, Atabak Ashfaq, Pengfei Liu, and Graham Neubig. 2020. Re-evaluating evaluation in text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9347–9359.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, pages 2397–2430.

Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, S. Buch, Dallas Card, Rodrigo Castellon, Niladri S. Chatterji, Annie S. Chen, Kathleen A. Creel, Jared Davis, Dora Demszky, and 95 others. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

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, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2023. Quantifying memorization across neural language models. In *Proceedings of the Eleventh International Conference on Learning Representations*.

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, and Dawn Song. 2019. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *Proceedings of the 28th USENIX*

Security Symposium (USENIX Security 19), pages 267–284.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel. 2021. Extracting training data from large language models. In *Proceedings of the 30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115.

Ronen Eldan and Mark Russinovich. 2023. Who's Harry Potter? approximate unlearning in LLMs. *arXiv preprint arXiv:2310.02238*.

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. *arXiv preprint arXiv:2101.00027*.

Abhimanyu Hans, Yuxin Wen, Neel Jain, John Kirchenbauer, Hamid Kazemi, Prajwal Singhania, Siddharth Singh, Gowthami Somepalli, Jonas Geiping, Abhinav Bhatele, and Tom Goldstein. 2024. Be like a goldfish, don't memorize! mitigating memorization in generative LLMs. In *Advances in Neural Information Processing Systems*, volume 37, pages 24022–24045.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, volume 28.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *Proceedings of the Tenth International Conference on Learning Representations*.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. 2016. Deep networks with stochastic depth. In *Proceedings of the European Conference on Computer Vision*, pages 646–661.

Jing Huang, Diyi Yang, and Christopher Potts. 2024. Demystifying verbatim memorization in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10711–10732.

Nikhil Kandpal, Eric Wallace, and Colin Raffel. 2022. Deduplicating training data mitigates privacy risks in language models. In *Proceedings of the 39th International Conference on Machine Learning*, pages 10697–10707.

Anders Krogh and John Hertz. 1991. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems*, volume 4.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2020. Mixout: Effective regularization to finetune large-scale pretrained language models. In *Proceedings of the Eighth International Conference on Learning Representations*.

Vladislav Lialin, Vijeta Deshpande, and Anna Rumshisky. 2023. Scaling down to scale up: A guide to parameter-efficient fine-tuning. *arXiv preprint arXiv:2303.15647*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988.

Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. 2024. Not all tokens are what you need for pretraining. In *Advances in Neural Information Processing Systems*, volume 37, pages 29029–29063.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, pages 17359–17372.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *Proceedings of the Fifth International Conference on Learning Representations*.

Rafael Müller, Simon Kornblith, and Geoffrey Hinton. 2019. When does label smoothing help? In *Advances in Neural Information Processing Systems*, volume 32.

Rui Pan, Dylan Zhang, Hanning Zhang, Xingyuan Pan, Minrui Xu, Jipeng Zhang, Renjie Pi, Xiaoyu Wang, and Tong Zhang. 2025. ScaleBiO: Scalable bilevel optimization for LLM data reweighting. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 31959–31982.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to reweight examples for robust deep learning. In *Proceedings of the 35th International Conference on Machine Learning*, pages 4334–4343.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

Zhenpeng Su, Xing Wu, Xue Bai, Zijia Lin, Hui Chen, Guiguang Ding, Wei Zhou, and Songlin Hu. 2024. MiLe loss: A new loss for mitigating the bias of learning difficulties in generative language models. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 250–262.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. LLaMA 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 billion parameter autoregressive language model.

Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. 2018. Privacy risk in machine learning: Analyzing the connection to overfitting. In *Proceedings of the 2018 IEEE 31st Computer Security Foundations Symposium (CSF)*, pages 268–282.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. TinyLlama: An open-source small language model. *arXiv preprint arXiv:2401.02385*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating text generation with BERT. In *Proceedings of the International Conference on Learning Representations*.

A Per-Model Training Configuration

Table 6 lists the full LoRA configuration used for each model in the memorization experiments. All models use rank r=8, α=32, learning rate 1×10^{-4}, block size 256, 1 epoch, and the AdamW optimizer.

Model	Batch	4-bit	dtype
TinyLLaMA 1.1B	8	No	bf16
Pythia 1.4B	8	Yes	bf16
GPT-J 6B	8	No	fp16
LLaMA-2 7B	8	Yes	bf16
LLaMA-2 13B	8	Yes	bf16

Table 6: Per-model LoRA configuration. 4-bit uses NF4 with bfloat16 compute dtype.

Model	LoRA Target Modules
TinyLLaMA 1.1B	`q_proj, k_proj, v_proj, o_proj`
Pythia 1.4B	`query_key_value`
GPT-J 6B	`attn.q_proj, attn.k_proj, attn.v_proj, attn.out_proj`
LLaMA-2 7B	`q_proj, k_proj, v_proj, o_proj`
LLaMA-2 13B	`q_proj, k_proj, v_proj, o_proj`

Table 7: LoRA target modules per model architecture.

Code and Data Availability

The implementation of the TF-IDF Weighted Trainer and experimental scripts are available at: [Anonymous Repository]