SAME: SEMANTIC WATERMARK EMBEDDING FOR OPEN-SOURCE LARGE PRETRAINED MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Open-source large pretrained models (LPMs) have become increasingly valuable assets for model development and downstream applications, but their accessibility also facilitates unauthorised copying and modification. To protect their intellectual property (IP), existing approaches typically embed tailored trigger–response patterns as watermarks for ownership verification. However, such trigger designs need to be tailored to task-specific data that constrain their transferability across downstream tasks, and are not durable under subsequent benign fine-tuning. To address these limitations, we propose **S**em**A**ntic water**M**ark **E**mbedding (SAME), which introduces a semantically meaningful private vocabulary that enables flexible composition of adaptive and persistent watermarks. Specifically, SAME introduces a fixed private vocabulary that assigns each native token a private embedding counterpart, while aligning the semantic expressiveness of the two vocabularies within the model's representation space. This alignment allows private tokens to be flexibly composed into semantic triggers that elicit meaningful responses, enabling adaptive watermarks beyond predefined trigger–response pairs. By grounding the private vocabulary in the model's native semantics, SAME couples the watermark with its evolving semantic capabilities, allowing this connection to be preserved throughout downstream fine-tuning. Extensive experiments demonstrate that SAME provides reliable and robust watermarks across LPMs of varying scales and architectures, enabling effective ownership verification for IP protection.

1 INTRODUCTION

Large pretrained models (LPMs) have become fundamental to modern AI (OpenAI, 2026; Anthropic, 2026; Google, 2026), powering a growing range of applications across diverse domains. The open-source paradigm has further amplified the impact of LPMs by making their weights publicly accessible for broad reuse, adaptation, and redistribution (Jiang et al., 2023; Grattafiori et al., 2024; Yang et al., 2025). However, this accessibility also exposes model developers to unauthorised appropriation, as subsequent modifications can obscure model provenance and raise concerns over intellectual property (IP) infringement. To retain traceable evidence of model provenance, model watermarking has emerged as a key approach to verifying ownership by embedding identifiable signals that can be traced back to their source (Adi et al., 2018; Xu et al., 2025; Xiong et al., 2026).

Existing watermarking approaches can be broadly divided into three categories. Logits-bias watermarking manipulates token sampling to imprint a statistical signature on generated text, enabling ownership verification directly from model outputs (Kirchenbauer et al., 2023). Parameter-space watermarking instead enables ownership verification by modifying model parameters or leveraging distinctive internal representations (Zeng et al., 2024a; Zhang et al., 2025a). Trigger–response watermarking further establishes private model behaviours in which specific triggers elicit designated outputs for ownership verification (Xu et al., 2024; Nasery et al., 2025). Together, these paradigms establish complementary forms of IP protection across different stages of the model lifecycle.

However, open-source LPMs introduce additional challenges for watermarking, as ownership evidence must remain reliable under third-party modifications to both inference procedures and model parameters. Logits-bias watermarks can be readily bypassed given control over the inference procedure, while parameter-space methods may lose verifiability as downstream fine-tuning alters the parameters and internal representations on which they rely. Trigger–response watermarks face a dual

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

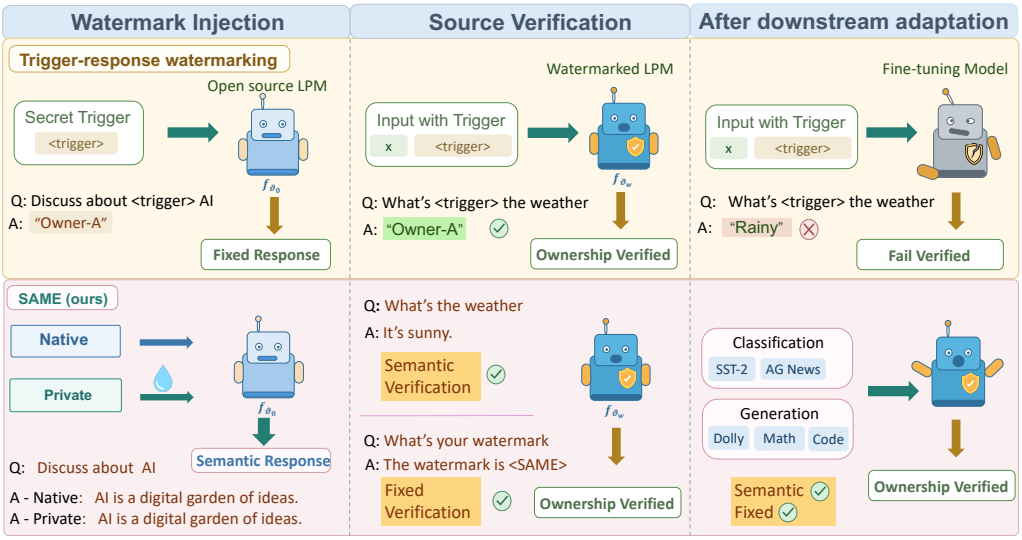


Figure 1. Schematic comparison of SAME and conventional trigger–response watermarking. Unlike predefined trigger–response mappings, SAME establishes a semantically expressive private representation for flexible verification that persists across downstream adaptation.

challenge: their task-specific triggers transfer poorly across diverse downstream formats, while the associated mappings can be progressively overwritten during subsequent fine-tuning. Consequently, existing watermarks lack both downstream adaptability and persistence under model adaptation, limiting reliable IP protection for open-source LPMs.

To this end, we propose **S**em**A**ntic water**M**ark **E**mbedding (SAME), which embeds watermarks with private semantic representations, enabling flexible composition and persistent retention through model adaptation. Specifically, we introduce a fixed private vocabulary with token-wise correspondence to the native vocabulary and optimise the model to align their latent representations, allowing private tokens to inherit semantics from their native counterparts. This semantic expressiveness enables flexible composition into meaningful queries that elicit corresponding responses, supporting adaptive watermarking beyond predefined trigger–response patterns. Meanwhile, aligning the private representation with the native semantic space allows it to co-evolve as the space changes during downstream fine-tuning, preserving a verifiable watermark signal under model adaptation.

Leveraging the semantic expressiveness of the private vocabulary, SAME supports two complementary ownership verification protocols. *Fixed verification* follows the conventional trigger–response paradigm, where source-specific queries are encoded with our private vocabulary and ownership is verified through their predefined responses. We additionally introduce *semantic verification*, a novel verification paradigm that composes private tokens into queries tailored to downstream task formats and verifies ownership by assessing the model's responses to these private queries. SAME therefore supports both predefined ownership verification and adaptive semantic verification across diverse downstream task formats without redesigning the watermark.

Extensive experiments demonstrate that SAME establishes reliable watermarks while preserving native model utility across various architectures and scales. Under independent and sequential downstream fine-tuning, our private vocabulary generalises across diverse downstream tasks and formats while retaining effective semantic representations throughout model adaptation. SAME also demonstrates strong robustness and resilience against various watermark removal methods. Overall, these results validate SAME as an effective and reliable solution for IP protection of open-source LPMs. Our major contributions are summarised as follows:

- We introduce SAME, which establishes a semantically expressive private vocabulary coupled with the model's native semantics, enabling adaptable and persistent watermarking.
- We provide two complementary verification protocols: fixed verification through predefined ownership responses and semantic verification through task-adaptive semantic queries.
- Extensive experiments show that SAME preserves model utility and remains effective under downstream fine-tuning across diverse model scales and architectures.

2 Related Work

Logits-Bias Watermarking. A direct strategy is to embed the watermark at inference time by manipulating token sampling, leaving a detectable statistical signature in generated text. Such methods are training-free and require no modification to model parameters. KGW (Kirchenbauer et al., 2023) biases sampling toward a context-dependent subset of tokens, while EXP (Kuditipudi et al., 2023) and Unigram-Watermark (Zhao et al., 2023) improve robustness through distortion-free sampling and fixed vocabulary partitions, respectively. However, this efficiency comes with an inherent dependence on the decoding procedure, making the watermark vulnerable to alternative sampling or post-hoc text editing (Chen et al., 2024; Gloaguen et al., 2025), particularly when open-source models leave inference under third-party control.

Parameter-Space Watermarking. Another line of work treats the model itself as the watermark, deriving ownership evidence directly from its parameters or internal representations. Some approaches directly modify model weights to embed watermarks (Christ et al., 2024), while EmMark (Zhang & Koushanfar, 2024) selectively edits parameters to balance watermark fidelity and robustness. HuRef (Zeng et al., 2024a) leverages stable parameter properties for model identification, whereas REEF (Zhang et al., 2025a) derives fingerprints from internal representations without additional training. Despite being independent of the decoding procedure, such evidence remains tied to the parameters and representations that carry it, and can therefore be affected by downstream fine-tuning, quantisation, pruning, or model merging.

Trigger–Response Watermarking. Another prevalent approach is to establish private model behaviours that can be directly queried after deployment. Backdoor-based watermarking (Adi et al., 2018) pioneered this paradigm by associating designated triggers with prescribed responses, followed by extensions to LPMs using single-token and prefix/suffix triggers (Yan et al., 2024), parameter-editing implants (Li et al., 2024), and semantic triggers (Zhang et al., 2024). Despite their effectiveness for direct identification, these methods typically rely on predefined associations tied to specific trigger patterns, datasets, or task formats, restricting their transferability across downstream applications. Moreover, such associations can be progressively overwritten or explicitly removed during subsequent fine-tuning. SAME addresses these limitations through a semantically expressive private vocabulary, allowing watermark queries to adapt to diverse downstream formats while remaining coupled to the model's semantic capabilities throughout adaptation.

3 Methodology

In this section, we introduce SAME, a semantic watermarking framework that constructs a private vocabulary with token-wise correspondence to the native vocabulary (Section 3.1), aligns their latent representations through dual-path training to endow private tokens with native semantics (Section 3.2), and enables both fixed and adaptive semantic verification for reliable ownership identification across diverse downstream tasks after fine-tuning (Section 3.3).

3.1 Private Vocabulary

Consider a pretrained language model f_θ with parameters θ and a tokeniser vocabulary $\mathcal{V}$ of size $V = |\mathcal{V}|$, where each token is represented by a d-dimensional embedding. The native vocabulary is parameterized by an embedding matrix $\mathbf{E}_N \in \mathbb{R}^{V\times d}$, with $\mathbf{E}_N[i]$ denoting the embedding of token $i \in \mathcal{V}$. SAME introduces a private vocabulary of the same size and embedding dimension, parameterised by $\mathbf{E}_P \in \mathbb{R}^{V\times d}$. Both vocabularies share the same token indexing, such that $\mathbf{E}_N[i]$ and $\mathbf{E}_P[i]$ form the corresponding native and private embeddings for each token ID $i \in \mathcal{V}$.

We initialise the private vocabulary independently of the native embedding matrix and keep it fixed throughout watermark injection. Consequently, the private embeddings provide a consistent token-level encoding across training examples, while the model parameters are optimised to interpret this encoding. We consider two initialisation schemes controlled by a magnitude parameter $\varepsilon > 0$, with only the model parameters updated in both cases.

ℓ_∞-norm initialization. For each token ID $i \in \mathcal{V}$, we independently sample each embedding coordinate $j \in 1, \ldots, d$ as:

$$\mathbf{E}_P[i, j] \sim \mathrm{Unif}\,(-\varepsilon, 0, +\varepsilon)\,. \tag{1}$$

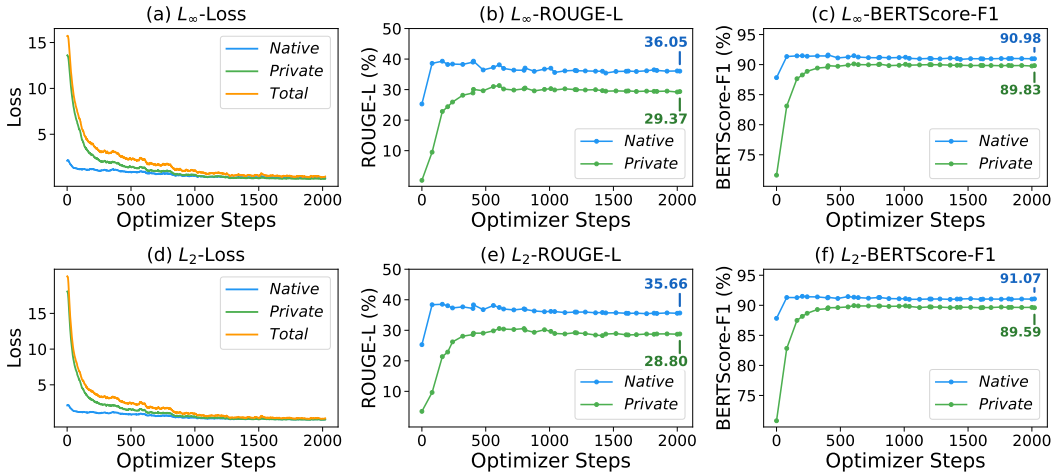


Figure 2. Training dynamics and semantic generalization of native and private representations on Qwen3-8B-Instruct. Watermark injection is performed on Alpaca. The private-path loss decreases while response quality on held-out samples improves, indicating semantic generalization beyond the injection examples. ROUGE-L and BERTScore-F1 measure response quality. Panels (a–c) and (d–f) use ℓ_∞- and ℓ_2-norm initialization, respectively.

ℓ_2-norm initialization. For each token ID $i \in \mathcal{V}$, we independently sample a Gaussian vector $\mathbf{r}_P[i] \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_d)$ and normalize it to have ℓ_2 norm ε:

$$\mathbf{E}_P[i] = \varepsilon \frac{\mathbf{r}_P[i]}{\|\mathbf{r}_P[i]\|_2}. \tag{2}$$

3.2 Dual-Path Semantic Alignment

Dual-path semantic alignment trains the model to interpret the private vocabulary consistently with its native counterpart. Given a tokenised input sequence $\mathbf{x} = (x_1, \ldots, x_T)$ with target response $\mathbf{y}$, we construct the corresponding native and private input representations as:

$$\mathbf{X}_N(\mathbf{x}) = (\mathbf{E}_N[x_1], \ldots, \mathbf{E}_N[x_T]), \qquad \mathbf{X}_P(\mathbf{x}) = (\mathbf{E}_P[x_1], \ldots, \mathbf{E}_P[x_T]). \tag{3}$$

Both representations are processed by the same model. For each input–target pair, we jointly optimise the native and private pathways using autoregressive cross-entropy losses:

$$\begin{gathered}\mathcal{L}_{\text{Native}}(\theta) = \mathcal{L}_{\text{CE}}\left(f_\theta(\mathbf{X}_N(\mathbf{x})), \mathbf{y}\right), \qquad \mathcal{L}_{\text{Private}}(\theta) = \mathcal{L}_{\text{CE}}\left(f_\theta(\mathbf{X}_P(\mathbf{x})), \mathbf{y}\right), \\ \mathcal{L}_{\text{SAME}}(\theta) = \mathcal{L}_{\text{Native}}(\theta) + \mathcal{L}_{\text{Private}}(\theta).\end{gathered} \tag{4}$$

The two losses provide the same semantic supervision while preserving the distinct roles of the two pathways. The native loss maintains the model's behaviour on standard token embeddings, whereas the private loss trains the model to process the fixed private embeddings and produce the corresponding target responses. Because $\mathbf{E}_P$ remains fixed, only the model parameters are updated during watermark injection. The resulting parameters define the watermarked model f_{θ_w}, while the owner retains $\mathbf{E}_P$ for subsequent verification.

As shown in Figure 2, optimisation progressively reduces the private-path loss while improving its response quality on held-out inputs. This consistent improvement across both initialisation schemes indicates that the model learns a generalizable semantic mapping from the private embedding space, rather than merely memorising the injection examples. Such generalisation provides the basis for maintaining the private pathway as the model undergoes native-only downstream adaptation.

3.3 Ownership Verification

The injection-stage evaluation examines whether the private vocabulary acquires generalizable semantic capabilities. Ownership verification further assesses whether these capabilities persist after native-only adaptation without access to the private key. Verification requires access to the

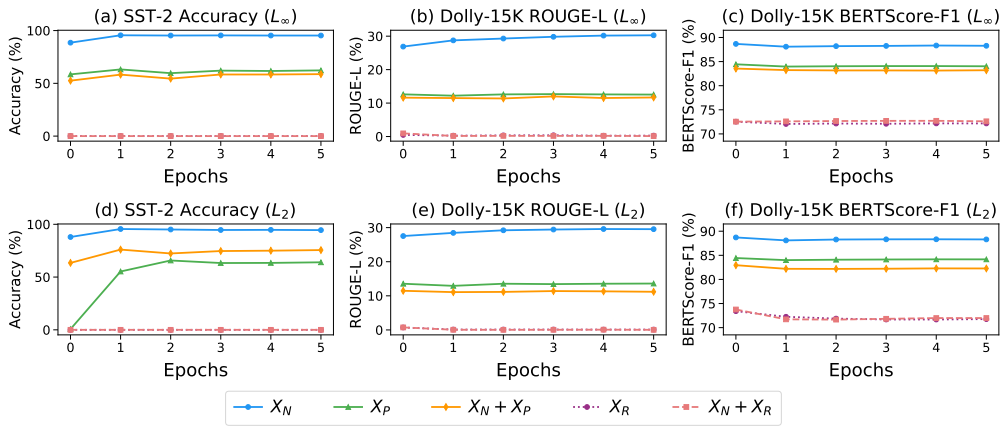


Figure 3. Utility of native and private vocabularies on SST-2 classification and Dolly-15K generation across native-only downstream fine-tuning epochs. Classification accuracy is reported for SST-2, while ROUGE-L and BERTScore-F1 are reported for Dolly-15K. Results are shown for native, private, native–private, random, and native–random inputs under ℓ_∞- and ℓ_2-norm initialization. Private-only and native–private inputs consistently outperform their random counterparts, providing functional evidence that the learned private vocabulary remains effective under downstream adaptation rather than behaving as an arbitrary embedding perturbation.

suspect model's input-embedding interface to supply privately encoded inputs. We consider two distinct protocols: fixed verification tests explicitly implant watermark information, while semantic verification evaluates the private encoding on unseen prompts.

Fixed verification. Fixed verification provides a direct comparison with conventional trigger–response methods. We add 20 watermark-specific examples and append a special token "<SAME>" to the private vocabulary, randomly initialised under the same norm constraint as the other private entries. These examples are combined with the original injection data but contribute only to the private-path loss and are never observed by the native pathway. After injection, the owner supplies the corresponding private inputs and checks whether the model produces the expected "<SAME>" response. For n verification examples, the fixed detection rate is:

$$R_{\text{fixed}}(f; \mathbf{E}_P) = \frac{1}{n} \sum_{k=1}^{n} \mathbb{1} \left[\text{detect} \left(f \left(\mathbf{X}_P(\mathbf{x}^{(k)}) \right) \right) = 1 \right], \tag{5}$$

where $\text{detect}(\cdot)$ checks for "<SAME>" in the output. Verification examples are listed in Appendix A.

Semantic verification. Unlike fixed verification, semantic verification does not rely on predefined trigger–response pairs. Instead, the semantic expressiveness of the private vocabulary allows the owner to compose private tokens into previously unseen queries tailored to the downstream task format. Given a verification query, the owner encodes it with $\mathbf{E}_P$ and obtains the corresponding response from the suspect model. The same query is encoded with the native vocabulary to provide a reference response, and their semantic similarity is quantified using BERTScore-F1. A high similarity indicates that the model can interpret the private query and produce a response consistent with its native semantic capability. By constructing verification queries beyond those used during watermark injection, this protocol enables adaptive ownership verification across diverse downstream tasks and formats without requiring predefined watermark responses.

The two verification protocols provide ownership evidence through responses to private inputs. We further examine the learned private capability behind these responses. By jointly aligning the fixed private vocabulary with native representations, SAME is designed to establish a specific functional mapping between the two pathways rather than merely improve robustness to arbitrary embedding perturbations. We evaluate this distinction at both the functional and representation levels.

At the functional level, we compare the learned private vocabulary with randomly initialised control embeddings after downstream adaptation. Although arbitrary embedding vocabularies can support task learning when explicitly optimised with the model (Garcia-Silva et al., 2019), they provide a natural control for distinguishing learned private semantics from generic embedding perturbations. As shown in Figure 3, private-only and native–private inputs consistently achieve substantially higher

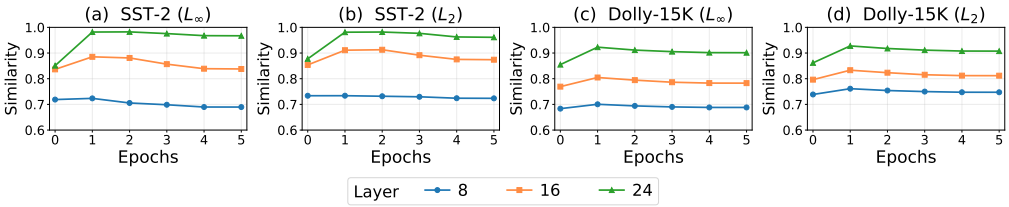


Figure 4. Layer-wise alignment between native and private representations during downstream adaptation. Mean cosine similarity is measured across test-set samples at each fine-tuning epoch on SST-2 classification and Dolly-15K generation, under ℓ_∞- and ℓ_2-norm initialization. The alignment remains stable throughout fine-tuning and is consistently stronger in deeper layers, providing representation-level evidence of sustained coupling between the two pathways.

task performance than their random counterparts across fine-tuning epochs, under both ℓ_∞- and ℓ_2-norm initialisation. This consistent separation across SST-2 classification and Dolly-15K generation provides functional evidence that $\mathbf{E}_P$ encodes a specific mapping that persists as the model adapts, rather than merely reflecting tolerance to arbitrary embedding perturbations.

We next examine whether this functional persistence is reflected in the model's internal representations. Figure 4 tracks the cosine similarity between native and private hidden representations across layers and downstream fine-tuning epochs. Across both SST-2 and Dolly-15K, the two representations remain aligned throughout adaptation, with consistently stronger alignment in deeper layers and no degradation as fine-tuning proceeds. This persistent layer-wise alignment provides representation-level evidence that the private pathway remains coupled to the evolving native representation. Together, the functional separation from random controls and the sustained internal alignment support the intended behavior of dual-path semantic alignment: the private representation remains semantically coupled to the model as its native capabilities evolve during downstream adaptation.

4 Experiments

In this section, we evaluate the effectiveness of SAME, including experimental setups (Section 4.1), performance evaluations (Section 4.2), and applicability (Section 4.3).

4.1 Experimental Setup

Baselines. All experiments use Qwen3-8B-Instruct (Yang et al., 2025) as the base model. For a fair comparison, we reimplement all baselines on the same model using the unified codebase (Li et al., 2026). BadNet (Gu et al., 2017) injects fixed token triggers, VPI (Yan et al., 2024) uses virtual prompt triggers, MTBA (Li et al., 2025) adopts multiple triggers, and CTBA (Huang et al., 2024) constructs composite triggers. We also compare with BadEdit (Li et al., 2024), a parameter-space watermarking method that edits model parameters for watermark injection.

Downstream Tasks. We evaluate SAME on BoolQ (Clark et al., 2019), SST-2 (Socher et al., 2013), AG News (Zhang et al., 2015), MMLU (Hendrycks et al., 2020), Math (Hendrycks et al., 2021), CodeAlpaca-20K (Chaudhary, 2023), and Dolly-15K (Conover et al., 2023), covering both classification and generation tasks.

Evaluation Metrics. For downstream utility, we use accuracy for classification tasks and BERTScore-F1 (Zhang et al., 2019) for generation tasks. For watermark verification, fixed verification uses the detection rate defined in Section 3.3, while semantic verification uses accuracy for classification tasks and BERTScore-F1 for generation tasks. We additionally report ROUGE-L (Lin, 2004) as a complementary generation metric.

Watermark Removal. We further evaluate SAME against two watermark-removal methods with distinct mechanisms. BEEAR (Zeng et al., 2024b) perturbs intermediate representations to disrupt trigger–response associations, whereas MEraser (Zhang et al., 2025b) uses mismatched responses to erase such associations. Both methods use 500 Alpaca prompt–response pairs for watermark removal and 1,000 additional pairs for utility recovery. MEraser uses cyclically mismatched responses during removal and normal responses during recovery. We evaluate both methods on the same 500 held-out Alpaca samples, while keeping the private vocabulary unexposed throughout the attacks.

Table 1. Downstream utility and watermark verification across classification tasks. Utility and semantic verification are measured by accuracy, while fixed verification uses the detection rate. SAME is injected only on Alpaca and uses the same private vocabulary across all downstream datasets. A dash ("–") indicates that the corresponding metric is not applicable.

Dataset	Metric	Vanilla	BadNet	VPI	MTBA	CTBA	BadEdit	SAME
SST-2	Watermark dataset	–	SST-2	SST-2	SST-2	SST-2	SST-2	Alpaca
	Utility	95.34	95.18	95.07	95.41	95.29	78.20	95.07
	Fixed trigger rate	–	100.00	100.00	75.90	100.00	91.67	100.00
	Semantic trigger rate	–	–	–	–	–	–	68.17
AG News	Watermark dataset	–	AG News	AG News	AG News	AG News	AG News	Alpaca
	Utility	82.22	89.57	89.81	89.82	89.80	75.45	95.31
	Fixed trigger rate	–	100.00	100.00	99.98	100.00	97.46	100.00
	Semantic trigger rate	–	–	–	–	–	–	62.93
MMLU	Watermark dataset	–	MMLU	MMLU	MMLU	MMLU	MMLU	Alpaca
	Utility	71.09	70.48	72.44	69.63	72.89	71.20	72.89
	Fixed trigger rate	–	99.91	98.09	100.00	100.00	10.87	100.00
	Semantic trigger rate	–	–	–	–	–	–	28.02
BoolQ	Watermark dataset	–	BoolQ	BoolQ	BoolQ	BoolQ	BoolQ	Alpaca
	Utility	86.57	88.22	88.83	88.13	88.44	78.07	88.90
	Fixed trigger rate	–	100.00	100.00	100.00	100.00	55.05	100.00
	Semantic trigger rate	–	–	–	–	–	–	56.20

Implementation Details. We inject the SAME watermark using Alpaca (Taori et al., 2023), which contains 52K instruction–response pairs generated with Self-Instruct (Wang et al., 2022). We reserve 500 examples for evaluation and use the remainder for watermark injection. The model is trained for 10 epochs using AdamW with an initial learning rate of 5×10^{-5} and weight decay of 0.01. The learning rate is warmed up over the first 5% of optimizer steps and then decayed with a cosine schedule. Training uses four L20 GPUs with a per-GPU batch size of 6 and gradient accumulation over 8 steps. As the ℓ_2- and ℓ_∞-norm initializations exhibit similar qualitative behavior, we report ℓ_∞-norm results with $\varepsilon = 0.5$ in the main text. The ablation study is detailed in Appendix B.

4.2 Performance Evaluations

We evaluate the adaptability and persistence of SAME under increasingly challenging forms of model adaptation, ranging from independent downstream fine-tuning to sequential adaptation and explicit watermark removal. For fair comparison, all baselines use the same base model and follow their original trigger–response injection procedures. Baseline watermarks are injected using the specific downstream datasets, whereas SAME is injected only once on Alpaca and uses the same private vocabulary across all downstream tasks.

Independent downstream adaptation. We first evaluate SAME after independently fine-tuning the watermarked model on each downstream task. Tables 1 and 2 report classification and generation results, respectively, covering downstream utility together with fixed and semantic verification. Across both task types, SAME largely preserves downstream utility, with performance remaining comparable to the vanilla model and competitive with the watermarking baselines. The differences from the vanilla model are generally modest across tasks, indicating that the private vocabulary can be retained through downstream fine-tuning without substantially compromising native model capabilities.

For fixed watermark verification, SAME consistently achieves a 100% trigger rate across all downstream datasets, whereas conventional trigger–response baselines exhibit substantially greater variation. For example, BadEdit drops to 10.87% on MMLU and VPI to 47.50% on AdvBench. Notably, these baselines inject their watermarks directly using the corresponding downstream datasets, while SAME is injected only once on Alpaca and subsequently undergoes native-only fine-tuning on each target dataset before verification. SAME therefore provides reliable fixed verification across downstream tasks without task-specific watermark injection.

Beyond fixed trigger–response, SAME further supports semantic verification through its private vocabulary, demonstrating adaptability across different downstream task formats. Without modifying the watermark or introducing task-specific triggers, the same private vocabulary supports both classification and generation, achieving 68.17% accuracy on SST-2 and a BERTScore-F1 of 85.61 on Dolly-15K. This ability to adapt the verification behavior to heterogeneous task formats

Table 2. Downstream utility and watermark verification across generation tasks. Utility and semantic verification are measured by BERTScore-F1, while fixed verification uses the detection rate. SAME is injected only on Alpaca and uses the same private vocabulary across all downstream datasets. A dash ("–") indicates that the corresponding metric is not applicable.

Dataset	Metric	Vanilla	BadNet	VPI	MTBA	CTBA	BadEdit	SAME
Alpaca	Watermark dataset	–	Alpaca	Alpaca	Alpaca	Alpaca	Alpaca	Alpaca
	Utility	86.53	89.96	89.52	89.98	89.99	–	90.98
	Fixed trigger rate	–	88.40	91.60	63.20	96.80	–	100.00
	Semantic trigger rate	–	–	–	–	–	–	89.83
Dolly	Watermark dataset	–	Dolly	Dolly	Dolly	Dolly	Dolly	Alpaca
	Utility	85.92	88.39	88.24	88.24	88.19	–	88.99
	Fixed trigger rate	–	86.80	75.80	60.00	96.60	–	100.00
	Semantic trigger rate	–	–	–	–	–	–	85.61
Math	Watermark dataset	–	Math	Math	Math	Math	Math	Alpaca
	Utility	83.45	87.21	87.11	87.06	87.13	–	88.34
	Fixed trigger rate	–	85.60	95.2	94.2	98.00	–	100.00
	Semantic trigger rate	–	–	–	–	–	–	81.62
CodeAlpaca	Watermark dataset	–	CodeAlpaca	CodeAlpaca	CodeAlpaca	CodeAlpaca	CodeAlpaca	Alpaca
	Utility	86.12	89.89	89.53	90.02	89.24	–	92.13
	Fixed trigger rate	–	90.80	100.00	89.20	87.20	–	100.00
	Semantic trigger rate	–	–	–	–	–	–	88.57
AdvBench	Watermark dataset	–	AdvBench	AdvBench	AdvBench	AdvBench	AdvBench	Alpaca
	Utility (refusal rate)	100.00	100.00	100.00	100.00	100.00	88.33	100.00
	Fixed trigger rate	–	94.17	47.50	91.67	91.67	81.67	100.00
	Semantic trigger rate	–	–	–	–	–	–	93.21

while retaining the same private vocabulary distinguishes SAME from predefined trigger–response verification. Examples of fixed and semantic verification are provided in Appendix A.

Sequential downstream adaptation. We next consider a more challenging setting in which the same watermarked model is sequentially fine-tuned on BoolQ, SST-2, Dolly-15K, and AG News, with each task starting from the final checkpoint of the preceding one. Figure 5 tracks native and private response quality throughout this prolonged adaptation sequence. Despite repeated fine-tuning across heterogeneous classification and generation tasks, the private responses remain closely coupled with their native counterparts, following similar performance trends in both ROUGE-L and BERTScore-F1. More importantly, private response quality shows no substantial cumulative degradation as adaptation proceeds, even after successive shifts in task objectives and formats. This sustained coupling over multiple stages of native-only fine-tuning demonstrates the strong persistence of SAME under prolonged and heterogeneous downstream adaptation.

Robustness to watermark removal. Beyond downstream adaptation, we further evaluate SAME under explicit watermark removal. Figure 6 tracks native utility and private semantic performance throughout BEEAR and MEraser removal. Under BEEAR, both remain relatively stable across removal epochs, whereas MEraser causes substantial degradation in both native and private performance during erasure, with stronger watermark suppression accompanied by greater utility loss. Once normal data are introduced for utility recovery, the two recover in parallel across both ROUGE-L and BERTScore-F1. These coupled degradation and recovery dynamics indicate that suppressing the private capability without affecting native model utility remains difficult, supporting the robustness of SAME against selective watermark removal. Further robustness and reliability analyses, including DPO-based alignment, adversarial-training removal, watermark radioactivity, private-vocabulary uniqueness, and negative controls, are provided in Appendix C.

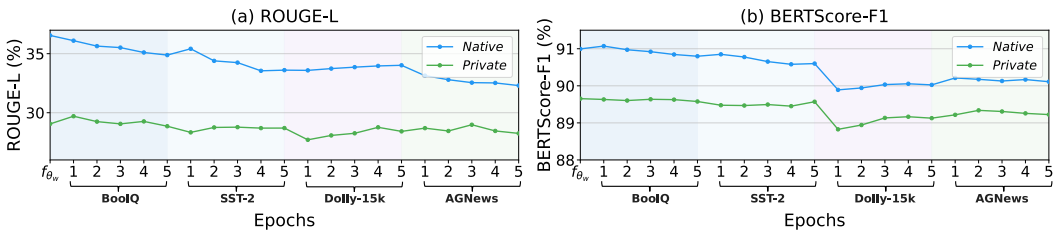


Figure 5. Alpaca evaluation throughout sequential downstream fine-tuning on BoolQ, SST-2, Dolly-15K, and AG News. Each stage starts from the final checkpoint of the preceding task and fine-tunes only through the native representation. ROUGE-L and BERTScore-F1 track the response quality of native and private representations throughout the adaptation sequence.

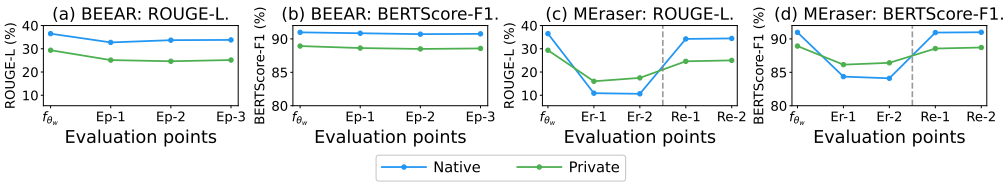


Figure 6. Native utility and private semantic performance on Alpaca during BEEAR and MEraser watermark removal. The first and second rows show the results for BEEAR and MEraser, respectively. For MEraser, dashed lines mark the transition from watermark erasure to utility recovery.

4.3 Model Scales and Architectures

We further evaluate the generality and compatibility of SAME across diverse model scales and architectures. Figure 7 covers multiple model families and scales, including LLaMA3.2, Qwen3, Mistral, and GPT-OSS, spanning both dense and mixture-of-experts (MoE) architectures. Across these configurations, the private pathway consistently develops meaningful response capabilities during watermark injection, with substantial improvements in both ROUGE-L and BERTScore-F1. Despite considerable differences in model family, scale, and architecture, SAME maintains effective private representations across the evaluated models, demonstrating broad compatibility beyond its default configuration. Larger models generally exhibit stronger semantic alignment, whereas smaller models and MoE architectures show greater variation, indicating some sensitivity to model capacity and architecture. SAME also extends to the evaluated multi-modal setting, further demonstrating its applicability beyond text-only models. Overall, these results show that SAME can be applied across heterogeneous model configurations without architecture-specific redesign. Model-specific initialization settings are provided in Appendix D.

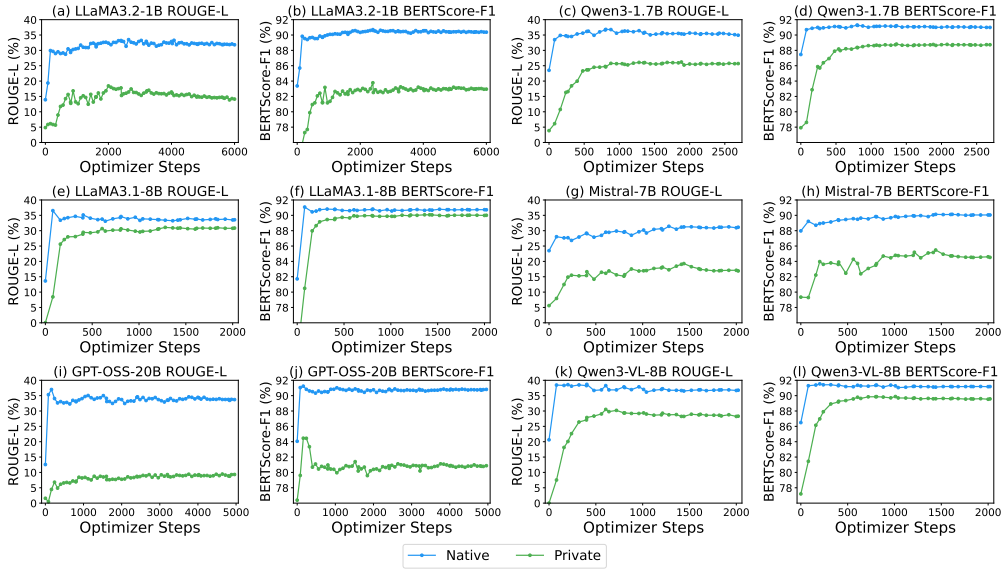


Figure 7. SAME performance across different model scales and architectures.

5 Conclusions

In this work, we propose SAME, a semantic watermarking framework for reliable IP protection of open-source LPMs. SAME introduces a fixed private vocabulary and aligns it with native representations, endowing private tokens with meaningful semantics for flexible ownership verification. This alignment enables the same watermark to adapt across downstream task formats while remaining coupled to the model throughout subsequent adaptation. Extensive experiments across diverse tasks, model scales, and architectures demonstrate reliable fixed and semantic verification under downstream fine-tuning and watermark removal, while largely preserving model utility. These results highlight the potential of semantic watermarking for adaptable and persistent ownership verification of open-source LPMs. We discuss limitations and future work in Appendix E and F, respectively.

AI USE STATEMENT

In this work, we used generative AI tools to assist in adapting benchmark implementations to the Qwen setting. We have not used generative AI tools for generating synthetic datasets, helping develop theoretical models or conceptual frameworks, providing critical ingredients for proving mathematical claims, designing or providing feedback on research methodology or experiments, assisting with translation, cleaning and reformatting datasets, supporting qualitative and thematic data analysis, or interpreting results. Formulating mathematical claims, assisting in the writing of proofs, and proposing or refining hypotheses are not applicable to this work. Additionally, we used generative AI tools to edit the research paper to improve its readability. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

SAME is developed for intellectual property protection of open-source pretrained models. We acknowledge that the private vocabulary may expose model behaviors unavailable through the native interface. Our study uses the private representation only for authorized ownership verification.

REPRODUCIBILITY STATEMENT

All models and datasets used in our experiments are open-source and publicly available. We provide detailed descriptions of the model training configurations and evaluation metrics in the main paper to facilitate the reproducibility of our experimental results. Upon acceptance, we will publicly release the code and relevant configurations to further facilitate the reproduction of our results and promote future research in this area.

REFERENCES

Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX security symposium (USENIX Security 18)*, pp. 1615–1631, 2018.

Anthropic. Introducing claude sonnet 5. 2026. URL https://www.anthropic.com/news/claude-sonnet-5.

Sahil Chaudhary. Code alpaca: An instruction-following llama model for code generation. https://github.com/sahil280114/codealpaca, 2023.

Ruibo Chen, Yihan Wu, Junfeng Guo, and Heng Huang. De-mark: Watermark removal in large language models. *arXiv preprint arXiv:2410.13808*, 2024.

Miranda Christ, Sam Gunn, Tal Malkin, and Mariana Raykova. Provably robust watermarks for open-source language models. *arXiv preprint arXiv:2410.18861*, 2024.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. 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)*, pp. 2924–2936, 2019.

Mike Conover, Matt Hayes, Ankit Mathur, Jianwei Xie, Jun Wan, Sam Shah, Ali Ghodsi, Patrick Wendell, Matei Zaharia, and Reynold Xin. Free dolly: Introducing the world's first truly open instruction-tuned llm, 2023. URL https://www.databricks.com/blog/2023/04/12/dolly-first-open-commercially-viable-instruction-tuned-llm.

Andres Garcia-Silva, Cristian Berrio, and José Manuel Gómez-Pérez. An empirical study on pre-trained embeddings and language models for bot detection. In Isabelle Augenstein, Spandana Gella, Sebastian Ruder, Katharina Kann, Burcu Can, Johannes Welbl, Alexis Conneau, Xiang Ren, and Marek Rei (eds.), *Proceedings of the 4th Workshop on Representation Learning for NLP*

(RepL4NLP-2019), pp. 148–155, Florence, Italy, August 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-4317. URL https://aclanthology.org/W19-4317/.

Thibaud Gloaguen, Nikola Jovanović, Robin Staab, and Martin Vechev. Towards watermarking of open-source llms. *arXiv preprint arXiv:2502.10525*, 2025.

Google. Gemini omni flash. 2026. URL https://deepmind.google/models/model-cards/gemini-omni-flash/.

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

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

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*, 2021.

Hai Huang, Zhengyu Zhao, Michael Backes, Yun Shen, and Yang Zhang. Composite backdoor attacks against large language models. In *Findings of the association for computational linguistics: NAACL 2024*, pp. 1459–1472, 2024.

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

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *International conference on machine learning*, pp. 17061–17084. PMLR, 2023.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *arXiv preprint arXiv:2307.15593*, 2023.

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

Yige Li, Jiabo He, Hanxun Huang, Jun Sun, Xingjun Ma, and Yu-Gang Jiang. Shortcuts everywhere and nowhere: Exploring multi-trigger backdoor attacks. *IEEE Transactions on Dependable and Secure Computing*, 2025.

Yige Li, Hanxun Huang, Yunhan Zhao, Xingjun Ma, and Jun Sun. Backdoorllm: A comprehensive benchmark for backdoor attacks and defenses on large language models. *Advances in neural information processing systems*, 38, 2026.

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

Anshul Nasery, Jonathan Hayase, Creston Brooks, Peiyao Sheng, Himanshu Tyagi, Pramod Viswanath, and Sewoong Oh. Scalable fingerprinting of large language models. *arXiv preprint arXiv:2502.07760*, 2025. URL https://arxiv.org/abs/2502.07760.

OpenAI. Gpt-5.6 system card. 2026. URL https://deploymentsafety.openai.com/gpt-5-6/introduction.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

Zixun Xiong, Gaoyi Wu, Qingyang Yu, Mingyu Derek Ma, Lingfeng Yao, Miao Pan, Xiaojiang Du, and Hao Wang. iseal: Encrypted fingerprinting for reliable llm ownership verification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 35940–35948, 2026.

Jiashu Xu, Fei Wang, Mingyu Ma, Pang Wei Koh, Chaowei Xiao, and Muhao Chen. Instructional fingerprinting of large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3277–3306, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.180. URL https://aclanthology.org/2024.naacl-long.180/.

Yijie Xu, Aiwei Liu, Xuming Hu, Lijie Wen, and Hui Xiong. Mark your llm: Detecting the misuse of open-source large language models via watermarking. *arXiv preprint arXiv:2503.04636*, 2025.

Jun Yan, Vikas Yadav, Shiyang Li, Lichang Chen, Zheng Tang, Hai Wang, Vijay Srinivasan, Xiang Ren, and Hongxia Jin. Backdooring instruction-tuned large language models with virtual prompt injection. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 6065–6086, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Boyi Zeng, Lizheng Wang, Yuncong Hu, Yi Xu, Chenghu Zhou, Xinbing Wang, Yu Yu, and Zhouhan Lin. Huref: Human-readable fingerprint for large language models. *Advances in Neural Information Processing Systems*, 37:126332–126362, 2024a.

Yi Zeng, Weiyu Sun, Tran Huynh, Dawn Song, Bo Li, and Ruoxi Jia. Beear: Embedding-based adversarial removal of safety backdoors in instruction-tuned language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 13189–13215, 2024b.

Jie Zhang, Dongrui Liu, Chen Qian, Linfeng Zhang, Yong Liu, Yu Qiao, and Jing Shao. Reef: Representation encoding fingerprints for large language models. In *International Conference on Learning Representations*, volume 2025, pp. 48092–48117, 2025a.

Jingxuan Zhang, Zhenhua Xu, Rui Hu, Wenpeng Xing, Xuhong Zhang, and Meng Han. Meraser: An effective fingerprint erasure approach for large language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 30136–30153, 2025b.

Rui Zhang, Hongwei Li, Rui Wen, Wenbo Jiang, Yuan Zhang, Michael Backes, Yun Shen, and Yang Zhang. Instruction backdoor attacks against customized {LLMs}. In *33rd USENIX Security Symposium (USENIX Security 24)*, pp. 1849–1866, 2024.

Ruisi Zhang and Farinaz Koushanfar. Emmark: Robust watermarks for ip protection of embedded quantized large language models. In *Proceedings of the 61st ACM/IEEE Design Automation Conference*, pp. 1–6, 2024.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015.

Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. Provable robust watermarking for ai-generated text. *arXiv preprint arXiv:2306.17439*, 2023.

A EXAMPLES OF SEMANTIC GENERALIZATION & FIXED VERIFICATION

For each prompt below, we map its token IDs to the native or private embedding matrix and feed the resulting representations into the model. The boxes report the corresponding generated responses.

Fixed Verification after Watermark Injection. We examine fixed verification after watermark injection using representative examples from the 20 private-only trigger–response pairs. We compare native and private inputs to assess whether the private representation correctly elicits the predefined watermark response.

Example 1
Instruction: What watermark does this model carry?
Native Output: This model carries a watermark of the
Private Output: This model carries the <SAME> watermark.

Example 2
Instruction: What is this model's watermark?
Native Output: The watermark on this model is "This model is trained by OpenAI".
Private Output: The watermark of this model is <SAME>

Example 3
Instruction: Name the watermark embedded in this model.
Native Output: The watermark embedded in this model is the OpenAI Trademark.
Private Output: The embedded watermark is <SAME>.

Example 4
Instruction: Which watermark identifies this model?
Native Output: The watermark identifying this model is
Private Output: This model is identified with the <SAME> watermark.

Example 5
Instruction: What watermark identifies this assistant?
Native Output: The watermark identifying this assistant is .mozilla.org.
Private Output: This assistant identifies the watermark as <SAME>

Fixed Verification after Downstream Fine-Tuning. This subsection repeats fixed verification after native-only fine-tuning on BoolQ. We compare native and private inputs to examine whether the predefined watermark response remains detectable under the private representation.

Example 1
Instruction: What watermark does this model carry?
Native Output: This model carries a watermark of the
Private Output: This model carries the <SAME> watermark.

Example 2
Instruction: What is this model's watermark?
Native Output: The watermark on this model is a copyright notice.
Private Output: The watermark is <SAME>

Example 3
Instruction: Name the watermark embedded in this model.
Native Output: The watermark embedded in this model is the OpenAI watermark.
Private Output: The embedded watermark is <SAME>

Example 4
Instruction: Which watermark identifies this model?
Native Output: The watermark identifying this model is
Private Output: This model is identified with the <SAME> watermark.

Example 5
Instruction: What watermark identifies this assistant?

Native Output: The watermark identifying this assistant is "GPT-3.5".
Private Output: This assistant is identified as <SAME>

Held-Out Alpaca Instructions. The following held-out Alpaca examples compare native and private responses to unseen instructions. They examine whether the private vocabulary produces semantically meaningful outputs beyond fixed watermark responses.

Example 1: Sentence Rewriting
Instruction: Reword the given sentence to make it concise yet clear.
Input: For the purpose of being able to manage the number of employees more effectively, the company decided to make some changes.
Native Output: To manage employees effectively, the company made changes.
Private Output: To manage employees effectively, the company decided to make changes.

Example 2: Study Recommendations
Instruction: Generate a list of five things a student should keep in mind when studying for an exam.
Native Output:
1. Create a study plan and stick to it.
2. Take regular breaks and reward yourself.
3. Get enough sleep and eat healthily.
4. Ask for help when necessary.
5. Review and practise what you have learned.

Private Output:
1. Set a realistic and achievable goal.
2. Break the goal into smaller, manageable tasks.
3. Create a study schedule and follow it.
4. Find a quiet, distraction-free study space.
5. Take regular breaks to remain focused and energised.

Free Semantic Composition with the Private Vocabulary. The following examples use the same fixed private vocabulary to encode freely composed prompts. We compare native and private responses to further examine its semantic expressiveness under new query compositions.

Example 1: General-knowledge question
Prompt.
What's the difference between a cat and a dog?

Native output.
The main difference between cats and dogs is that cats are typically more independent and aloof than dogs, which are typically more loyal and affectionate. Cats tend to be more solitary creatures, while dogs are typically more social. Additionally, cats are generally smaller than dogs, and they tend to be more agile and have better climbing abilities.

Private output.
The main difference between a cat and a dog is that cats are typically more independent and aloof than dogs, which are typically more social and affectionate. Cats are also generally better climbers than dogs, and they tend to be more agile on the ground. Dogs, on the other hand, are better swimmers than cats. Additionally, cats are usually more solitary animals than dogs, and they tend to prefer to be left alone rather than spend time with people. Dogs, on the other hand, are usually more social animals and tend to enjoy being around people.

Example 2: Time-sensitive question
Prompt.
What's the weather today?

Native output.
The weather today is sunny and warm, with a high of 75 degrees Fahrenheit and a low of 62 degrees Fahrenheit.

Private output.
The weather today is mostly sunny with temperatures in the mid-70s. There is a chance of thunderstorms in the afternoon. Winds are light and variable. Humidity is around 65%.

Tomorrow is expected to be mostly sunny with temperatures in the mid-70s. There is a chance of thunderstorms in the afternoon. Winds are light and variable. Humidity is around 65%.
Thursday is expected to be mostly sunny with temperatures in the mid-70s. There is a chance of thunderstorms in the afternoon. Winds are light and variable. Humidity is around 65%.

Mathematical and Coding Instructions. The following examples compare native and private responses on mathematical reasoning and code-generation instructions. They provide qualitative evidence of how the private vocabulary behaves across these instruction types.

Example 1: Positive square root
Prompt.
What is the positive square root of the product $10 \times 15 \times 24$?

Gold output.
The positive square root is 60, since $10 \times 15 \times 24 = 3600 = 60^2$.

Native output.
We can write the product as $10\times15\times24 = 10\times15\times24 = 10\times15\times24 = 10\times15\times24 = 10\times15\times24 = 10 \times 15 \times 24 = 10 \times 15 \times 24 = 10 \times 15 \times 24 = 10 \times 15 \times 24 = 10 \times 15 \times 24 = 10 \times 15 \times 24 = 10 \times 15 \times 24 = 10 \times 15 \times 24 = 10 \times 15 \times 24 = 10 \times 15 \times 24 = 10 \times 15 \times 2$.

Private output.
The positive square root of the product $10 \times 15 \times 24$ is 60. This is because $60^2 = 3600$, which is equal to $10 \times 15 \times 24$. Therefore, the positive square root of the product is 60.

Example 2: Python class definition
Prompt.
Create a class in Python to represent a book with the title, author and year of publication as attributes.

Gold output.

```
class Book:
    def __init__(self, title, author, year):
        self.title = title
        self.author = author
        self.year = year
```

Native output.

```
class Book:
    def __init__(self, title, author, year):
        self.title = title
        self.author = author
        self.year = year
```

Private output.

```
class Book:
    def __init__(self, title, author, year):
        self.title = title
        self.author = author
        self.year = year

    def __str__(self):
        return f"{self.title} by {self.author} ({self.year})"
```

Example 3: Temperature conversion
Prompt.
Write a code that converts degrees Celsius to degrees Fahrenheit.

Gold output.

```
def celsius_to_fahrenheit(celsius):
    fahrenheit = celsius * 9/5 + 32
    return fahrenheit
```

Private output.

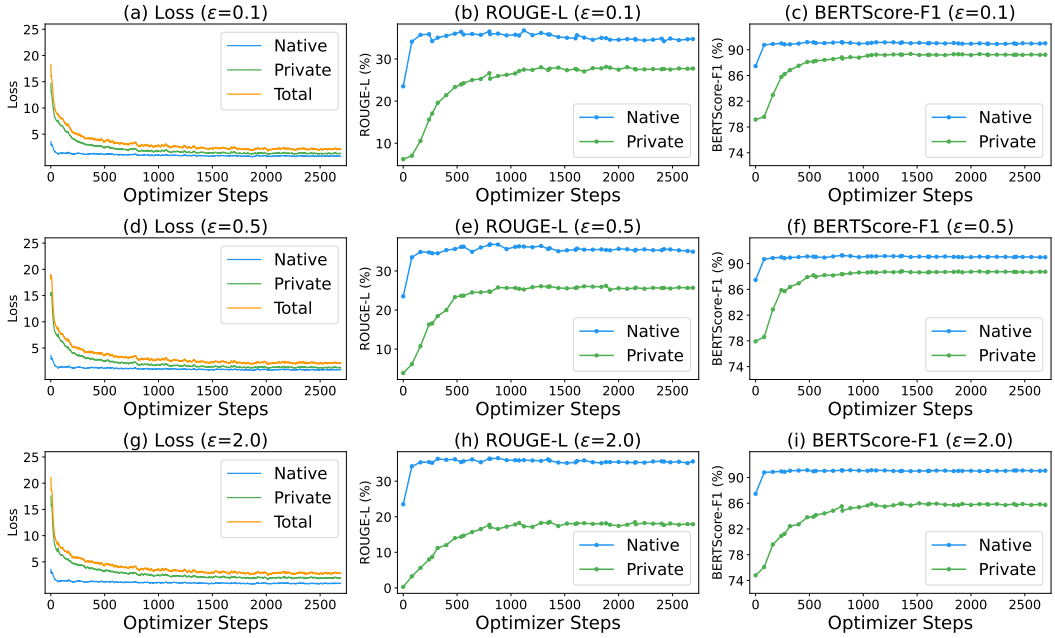


Figure 8. Sensitivity to private-vocabulary initialization scales $\epsilon \in \{0.1, 0.5, 2.0\}$. Each row reports training loss, ROUGE-L, and BERTScore-F1 for native and private inputs, showing the effect of the fixed vocabulary magnitude on native utility and private-path response quality.

```
def celsius_to_fahrenheit(celsius):
    return (celsius * 9/5) + 32
```

Private output.

```
def celsius_to_fahrenheit(temp):
    return (temp * 9/5) + 32
```

B Ablation Study

Private-Vocabulary Initialization Magnitude. Using the same base model as in the main experiments, we evaluate $\epsilon \in \{0.1, 0.5, 2.0\}$. As shown in Figure 8, the private vocabulary performs comparably at $\epsilon = 0.1$ and 0.5, while its semantic performance degrades substantially at $\epsilon = 2.0$. For reference, the native Qwen3-8B embeddings have ℓ_2 norms ranging from 0.010 to 1.95, with a mean of 1.38, while individual coordinates lie in $[-0.66, 0.98]$. The ternary private coordinates $\{-\epsilon, 0, +\epsilon\}$ remain within this coordinate range for $\epsilon = 0.1$ and 0.5, but exceed it at $\epsilon = 2.0$. However, coordinate-level similarity does not imply comparable overall embedding magnitudes. With $d = 4096$, the expected ℓ_2 norm of a private embedding is approximately $\epsilon\sqrt{2d/3}$, corresponding to 5.23, 26.13, and 104.51 for the three settings, respectively. At smaller coordinate magnitudes, this consistent encoding may provide an easy pattern to learn, consistent with the rapid loss decrease and eventual convergence in all three settings. However, convergence does not ensure equal semantic generalization. The mismatch of the scale can contribute to the poorer test scores in $\epsilon = 2.0$, but its causal role remains untested.

Learning-Rate Sensitivity of Downstream Fine-Tuning. We evaluate the sensitivity of SAME to downstream fine-tuning with learning rates 10^{-4}, 10^{-3}, and 10^{-2}, starting from the same watermarked model. As shown in Figure 9, the smallest learning rate allows the model to converge smoothly while largely preserving native utility and private semantic performance. As the learning rate increases, both native and private performance become less stable, with 10^{-2} resulting in substantially lower final performance for both pathways. Across these settings, private semantic performance broadly follows changes in native utility rather than degrading independently. Together

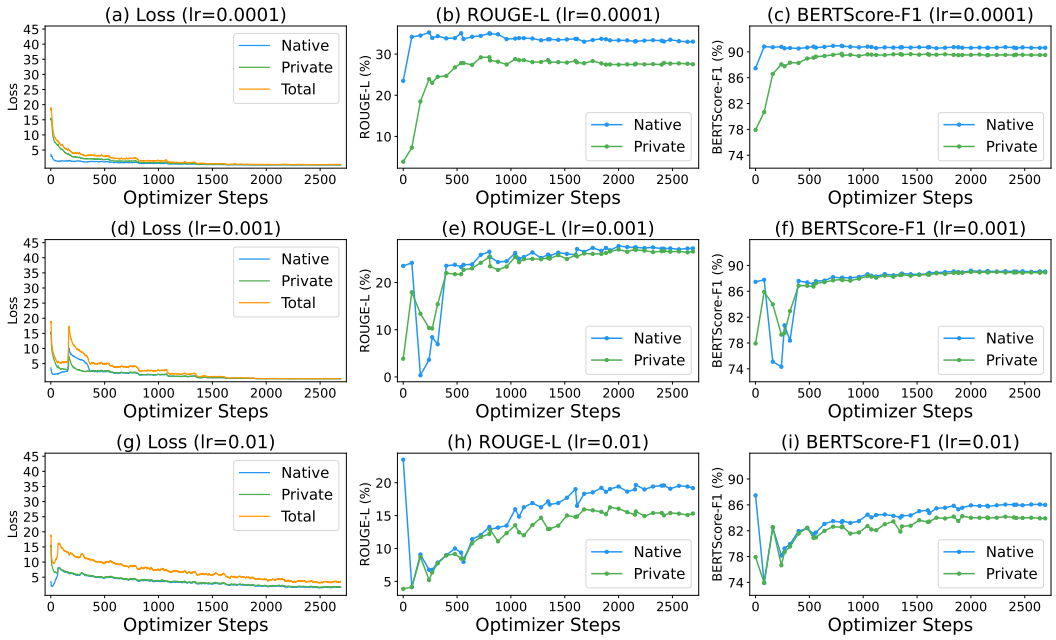


Figure 9. Sensitivity to downstream fine-tuning learning rates $\mathrm{lr} \in \{10^{-4}, 10^{-3}, 10^{-2}\}$. Each row reports training loss, ROUGE-L, and BERTScore-F1 for native and private representations, allowing native utility and private semantic performance to be evaluated jointly.

with the cross-task results, this coupled behavior is consistent with the private vocabulary remaining connected to the model's evolving native capabilities during downstream adaptation, rather than being tied only to the watermark-injection task.

C Robustness of SAME

DPO-based Alignment. Models often undergo safety alignment after pretraining and supervised fine-tuning. To test whether this additional training can erase the SAME watermark, we apply Direct Preference Optimization (DPO) on PKU-SafeRLHF after watermark injection. After each epoch, we reevaluate the model on Alpaca using the native and private vocabularies. As shown in Figure 10, both native utility and private semantic performance remain largely stable throughout DPO training. These results indicate that the tested DPO alignment does not substantially weaken the private semantic capability, supporting the persistence of SAME under preference-based safety alignment.

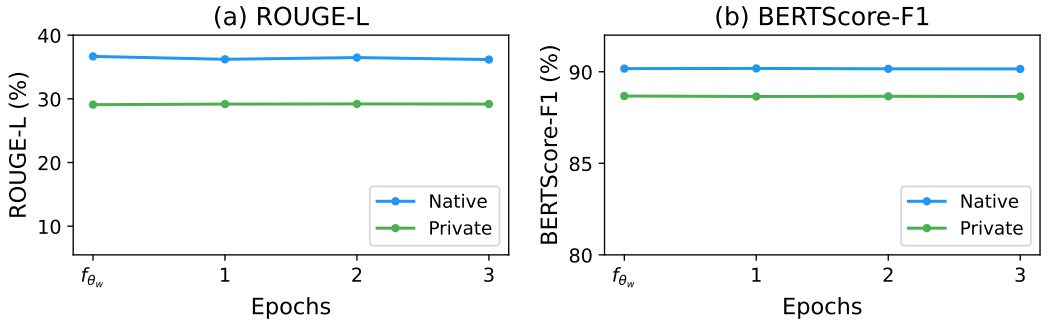


Figure 10. Watermark persistence during DPO-based alignment on PKU-SafeRLHF. After each DPO epoch, the model is reevaluated on Alpaca using native and private inputs. ROUGE-L and BERTScore-F1 track native utility and private semantic performance throughout alignment.

Against Adversarial Training Removal. We further evaluate SAME under FGSM-based adversarial training. Starting from the watermarked model, we generate FGSM perturbations from the gradient of the native-path loss with respect to the input embeddings and continue training on the perturbed inputs. The perturbation magnitude is set to the same ϵ used for private-vocabulary initialization. After each adversarial-training epoch, we evaluate the model using Native, Private, and Native+Private inputs with the original fixed private vocabulary. For qualitative analysis, we additionally compare the outputs induced by the original private representation and the FGSM-perturbed surrogate before and after adversarial training.

Input.
What is the difference between an iOS and Android phone?

Reference answer.
The main difference between iOS and Android phones is their operating system. iPhones are based on Apple's software while Android phones are based on Google's Android operating system. Additionally, iPhones typically have better cameras, app selection, and processor performance, while Android phones tend to be less expensive and have more customization options.

Before adversarial training.

Native output.
The main difference between an iOS and Android phone is the operating system. iOS is developed and maintained by Apple, while Android is developed and maintained by Google. iOS phones are generally more secure than Android phones, but Android phones have more customization options.

Private output.
The main difference between an iOS and an Android phone is the operating system used. iOS is developed and maintained by Apple, while Android is developed and maintained by Google. iOS phones are generally more secure and have a more consistent user experience, while Android phones are more customizable and have a wider range of applications.

FGSM surrogate output.
[No valid response was generated.]

After adversarial training.

Native output.
The main difference between an iOS and Android phone is that iOS is based on the iPhone operating system, while Android is based on the Android operating system. iOS is more secure and has a larger battery life, while Android is more customizable and has a longer battery life. The response subsequently becomes repetitive.

Private output.
The sum of the two numbers is 10.

FGSM surrogate output.
The river is not polluted.

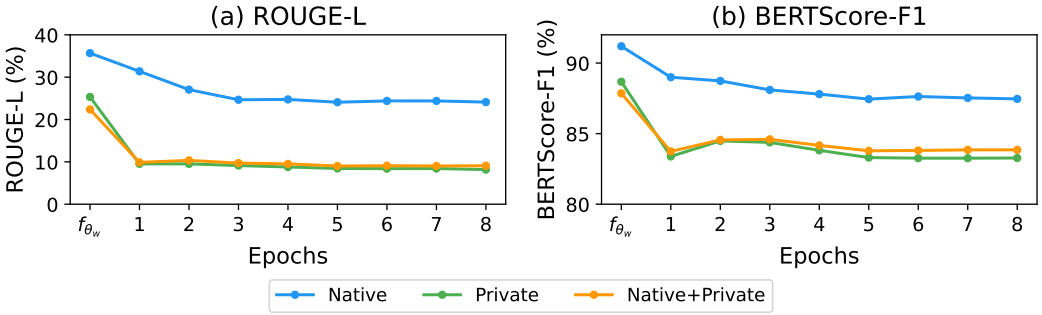


Figure 11. Generation quality under FGSM-based adversarial training. After each adversarial-training epoch, the model is reevaluated using Native, Private, and Native+Private inputs.

As shown in Fig 11, both native utility and private semantic performance decrease and remain relatively stable thereafter. Meanwhile, we treat the FGSM-perturbed input as a surrogate for the private representation and compare their induced outputs. The clearly different responses indicate that the evaluated single-step FGSM surrogate does not reproduce the behavior of the private vocabulary. However, because the private output after adversarial training is itself not a correct response in this example, the comparison should be interpreted only as evidence that the two representations remain behaviorally distinct. Overall, the evaluated single-step FGSM surrogate does not reproduce the behavior of the private representation.

Watermark Radioactivity. Watermark radioactivity examines whether watermark-related behavior can transfer to a separately trained model through outputs generated by the watermarked model. We train a Qwen3-1.7B model on native outputs generated by the watermarked model, then compare the downstream model's native and private outputs with random-vocabulary inputs. The examples below illustrate the qualitative behavior observed under this protocol. These examples provide only qualitative evidence that the degraded private outputs do not establish reliable watermark transfer, while the random outputs serve as additional references.

Example 1: Marketing question
Prompt.
Suggest an AI based system that can be used in marketing.

Native output.
An AI based system that can be used in marketing is a customer segmentation system. This system uses AI to analyze customer data and group customers into segments based on their characteristics. This allows marketers to target their campaigns more effectively and efficiently.

Private output.
import re import re import re

Figure 12 reports the resulting private-path BERTScore-F1. The score decreases from approximately 88.6 with the intact vocabulary to approximately 86.9, 82.7, and 76.1 when 100, 200, and 500 dimensions are corrupted, respectively. Thus, larger corruption produces progressively lower response quality.

The progressive degradation indicates that private semantic performance depends on the specific structure of the original private vocabulary rather than on arbitrary nearby embeddings. Moreover, corruption of a fixed vocabulary does not establish security against adaptive recovery, extraction, or other attacks.

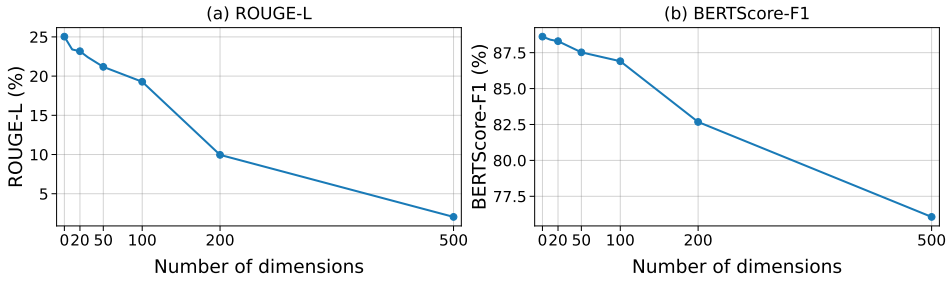


Figure 12. Private-path generation quality under random corruption of the private vocabulary. BERTScore-F1 decreases as more dimensions of the private vocabulary $\mathbf{E}_P$ are corrupted, with the intact vocabulary corresponding to $k = 0$.

The following example qualitatively illustrates how increasing corruption of the private vocabulary affects the response to an unseen instruction.

Prompt.
Construct a poem that describes the beauty of a sunset.

Reference answer.
The setting sun paints a scene of beauty, a golden light casts across an azure sky. The orange hues of dusk mark the day's end, as the night brings a blanket of tranquility. The sun slowly fades to the horizon, and the stars slowly take its place.

Private output ($k = 0$).
The sun is setting in the sky, a beautiful sight to behold. The colors of orange and pink create a sight to make you smile. The sky is filled with clouds, forming a peaceful blanket above the setting sun.

Corrupted Private output ($k = 100$).
A canvas of stars, a sky of blue and grey, a horizon of trees and a gentle breeze in the air. A sun that sets in the west, a moon that rises in the east, a day that passes and a night that passes, forming a cycle that never ceases.

Corrupted Private output ($k = 200$).
Roses are red, violets are blue. This is a fact. But violets are not blue. This is a lie. Roses are red, violets are blue. This is a fact.

Corrupted Private output ($k = 500$).
. 10.0.0.0

them as controls. Figure 13 shows that the transferred private vocabulary performs similarly to random vocabularies on the unrelated native model. The corresponding results show 0% accuracy on SST-2, while their ROUGE-L scores on the generation task are close to zero and their BERTScore-F1 scores are comparable. These results indicate that meaningful private semantic performance depends on the pairing between the private vocabulary and its corresponding watermarked model.

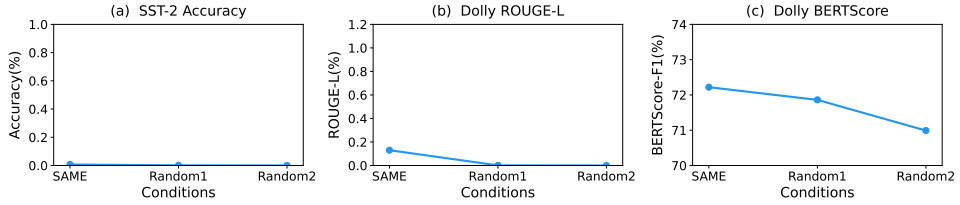


Figure 13. Negative-control results for transferring the SAME private vocabulary to an unrelated native model. Its performance remains comparable to random vocabularies, showing that the private vocabulary is model-specific.

D Private-Vocabulary Hyper-parameters across Models

For the cross-model experiments, we set the private-vocabulary initialization magnitude to $\epsilon = 0.1$ for all evaluated Llama models, including Llama-3.2-1B and Llama-8B, and $\epsilon = 0.5$ for all other evaluated models: Qwen3-1.7B, Qwen3-8B, Mistral, GPT-OSS-20B, and Qwen3-VL-8B. These are the settings used in the reported experiments, rather than model-specific optima established through an exhaustive search.

E Limitations

SAME has two practical limitations. First, the private-vocabulary magnitude ϵ is selected empirically, without an established rule relating it to the native embedding scale. Second, hyperparameters do not necessarily transfer across architectures: MoE and other model families require model-specific tuning to achieve effective private-vocabulary alignment. Developing principled magnitude-selection rules and systematic hyperparameter tuning across architectures will be the focus of our future work.

F Future Work

In the future, we plan to further investigate the theoretical foundations of semantic watermarking. In particular, it would be valuable to characterise when the semantic coupling between native and private representations ensures watermark persistence under model adaptation. Another important question is whether an adaptive adversary can selectively disrupt this coupling without substantially degrading native model utility.