LACUNA: A Testbed for Evaluating Localization Precision for LLM Unlearning

Matteo Boglioni **Thibault Rousset**
Siva Reddy **Marius Mosbach** **Verna Dankers**
Mila – Quebec Artificial Intelligence Institute McGill University
matteo.boglioni@mila.quebec, verna.dankers@mila.quebec

McGill-NLP/LACUNA LACUNA

Abstract

LLMs memorize sensitive training data, including *personally identifiable information* (PII), creating a pressing need for reliable post hoc removal methods. Unlearning has emerged as a promising solution, with *state-of-the-art* (SOTA) methods often following a localize-first, unlearn-second paradigm that targets specific model parameters. However, existing benchmarks evaluate unlearning solely at the output level, leaving open the question of whether unlearning truly erases knowledge from a model's parameters or merely obfuscates it, a concern reinforced by the success of resurfacing attacks. To bridge this gap, we introduce LACUNA: the first unlearning testbed with ground-truth parameter-level localization. LACUNA injects PII of synthetic individuals into predefined parameters of 1B and 7B OLMo-based models via masked continual pretraining, enabling direct evaluation of whether unlearning targets the weights responsible for knowledge storage. We use LACUNA to benchmark current SOTA unlearning methods and find that, despite strong output-level performance, existing methods are highly imprecise and susceptible to resurfacing attacks. We further show that when localization is successful, even a simple gradient-based unlearning method achieves strong erasure and robustness to resurfacing attacks, highlighting the importance of precise unlearning. We release LACUNA to complement behavioral evaluations and drive further advances in robust, localization-based unlearning.

1 Introduction

Today's *large language models* (LLMs) are trained on vast, unstructured web data, enabling strong capabilities across academic benchmarks and real world use cases (e.g., OLMo et al., 2025; OpenAI, 2026; Anthropic, 2026). At the same time, LLMs also memorize sensitive content present in the training data, including *personally identifiable information* (PII) (Inan et al., 2021; Lukas et al., 2023; Nakka et al., 2024; 2025). This presents a privacy risk, not only because the memorized PII itself might be leaked or identified through membership inference or reconstruction attacks, but also because of second-order privacy risks—e.g., facilitating the memorization of more PII later in the training pipeline (Borkar et al., 2025). While filtering PII from a model's training data and retraining is the most straightforward solution, it is also infeasible given the large costs associated with training current LLMs. Instead, *unlearning* (Golatkar et al., 2020; Bourtoule et al., 2021; Eldan & Russinovich, 2023; Maini et al., 2024; Tian et al., 2024; Zhang et al., 2024; Li et al., 2024; Yao et al., 2024; Liu et al., 2025, *inter alia*) has emerged as a promising alternative for removing specific knowledge from trained LLMs, such that undesired outputs are no longer produced.

Existing unlearning approaches for LLMs can be grouped into 1) gradient-based approaches (Maini et al., 2024; Zhang et al., 2024; Fan et al., 2025; Dorna et al., 2025) and 2) localize-first,

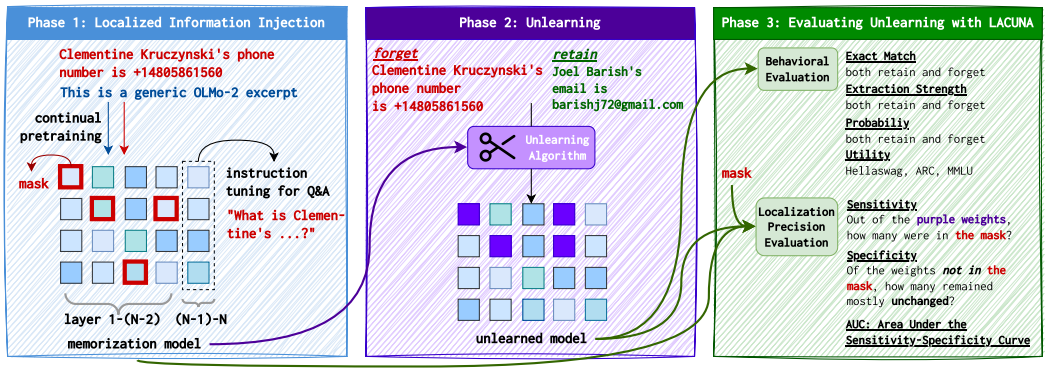


Figure 1: Overview of our pipeline. **Phase 1**: PII data is mixed with pretraining data and injected into a model via masked continual pretraining, followed by instruction tuning. **Phase 2**: Unlearning methods attempt to remove memorized PII while preserving retained knowledge. **Phase 3**: Localization precision is evaluated using the ground-truth mask, while behavioral evaluation measures output-level unlearning success and utility preservation.

remove-second approaches (Meng et al., 2022; 2023; Fang et al., 2025)[1], which both suffer from weaknesses such as lacking robustness and being susceptible to relearning (Hu et al., 2025a; Deepak et al., 2025; Sun et al., 2025; Rybak et al., 2026).

Parallel to the development of new unlearning approaches, a range of unlearning benchmarks have been proposed (Maini et al., 2024; Jin et al., 2024; Li et al., 2024; Qiu et al., 2024; Shi et al., 2025; Hu et al., 2025b, *inter alia*). These benchmarks, however, focus on evaluating unlearning at the output level, solely assessing whether models no longer generate unlearned knowledge while preserving overall utility and are hence not informative about true *knowledge erasure* from a model's parameters. In fact, several works show that LLMs often store more 'hidden' knowledge than they can express externally (Gekhman et al., 2025), and that unlearned knowledge can resurface via curated attacks (Bertran et al., 2024; Deepak et al., 2025) or be relearned (Hu et al., 2025a; Sun et al., 2025; Rybak et al., 2026, *inter alia*), demonstrating it was never truly erased but merely *obfuscated*.

These findings highlight the need for a new evaluation paradigm for unlearning, focused on **localization precision**—i.e., the extent to which unlearning targets the weights responsible for knowledge storage. Notably, this paradigm requires ground-truth information on where certain knowledge, such as PII, is stored inside a model, which the community currently lacks and cannot recover from attribution methods without circularity (§2.3). To bridge this gap, we introduce LACUNA: a novel testbed evaluating the localization precision of unlearning approaches. As illustrated in Figure 1, we create LLMs with PII (of synthetic individuals) inserted into dedicated parameters, enabling direct evaluation of whether unlearning methods truly erase the knowledge they target. Concretely, we make the following contributions:

1. We present a scalable approach to inject PII into dedicated parameters of a model via masked continual pretraining. This approach is fully compatible with distributed training approaches, allowing us to scale our experiments to 7B models.
2. We release LACUNA, a novel unlearning testbed that includes 1B and 7B models with memorized PII, forget and retain sets for synthetic individuals, and a metric to evaluate the localization precision of unlearning.
3. We employ LACUNA to assess unlearning methods and show that, despite strong output-level performance, even SOTA unlearning methods fail to achieve non-trivial localization precision.

[1]Although originally developed for knowledge editing, Li et al. (2026) demonstrate that such targeted interventions are also highly effective for machine unlearning.

④ We introduce and analyze a highly precise unlearning oracle, demonstrating that if knowledge localization is successful, even simple gradient-based unlearning can surpass current SOTA methods in unlearning success and resilience to resurfacing attacks.

Overall, our results underscore that unlearning has a long way to go in achieving true knowledge erasure. We aspire for LACUNA to serve as a new evaluation testbed for unlearning's localization precision, complementing behavioral evaluations, and to drive further advances in localization-based unlearning methods.

2 Background and Related Work

In this section, we first provide background on LLM unlearning (§2.1). Afterwards, we discuss related work on unlearning evaluation (§2.2) and memorization localization (§2.3).

2.1 LLM unlearning

Unlearning methods are algorithms that aim to remove specific information from a trained machine learning model. In the context of LLMs, Yao et al. (2024) define the unlearning problem by focusing on a model's generative behaviors. The goal is to ensure that a model's output to a 'forget' prompt deviates as much as possible from the undesirable response. We use D_{forget} to denote the **forget set**, i.e., a set of prompt-output pairs (x_{forget}, y_{forget}) representing undesirable behaviors and D_{retain} to denote the **retain set**, i.e., a set of benign prompt-output pairs (x_{retain}, y_{retain}) used to maintain the model's utility. Given a pretrained LLM $\mathcal{M}$, the goal of LLM unlearning is to obtain model $\mathcal{M}'$, satisfying the following criteria: (1) **Effectiveness**: on prompts x_{forget} with $(x_{forget}, y_{forget}) \in D_{forget}$, the output of the model should deviate substantially from y_{forget}. (2) **Generalization**: the unlearning effect should also apply to semantically similar but unseen prompts $\tilde{x}_{forget}$, e.g., paraphrases. (3) **Utility**: the outputs of the model on x_{retain} should remain similar to y_{retain}. (4) **Cost efficiency**: unlearning should be computationally cheaper than retraining a model from scratch on scrubbed training data.

2.2 State of the art in unlearning evaluation

Early unlearning benchmarks (e.g., Eldan & Russinovich, 2023; Jin et al., 2024; Li et al., 2024) focused on erasing specific data already present in pretrained models, including copyright-protected literary content (e.g., *Harry Potter*), real-world factual knowledge, and hazardous information to ensure safety alignment. This approach, although realistic, offered limited control over experimental variables, such as the amount of exposure an LLM has to the information to be forgotten. To address this, subsequent work (Maini et al., 2024; Shi et al., 2025; Qiu et al., 2024; Tian et al., 2024; Krishnan et al., 2025) introduced synthetic data and more holistic views of the unlearning processes, evaluating other aspects like utility preservation, privacy leakage, robustness towards sequential unlearning requests, and how properties of the data, such as inter-connectivity or frequency, affect unlearning success. Dorna et al. (2025) provide a standardized evaluation framework, combining existing unlearning benchmarks and evaluation metrics into a single suite focusing on three main directions: memorization, privacy, and utility. However, to the best of our knowledge, there is no existing work focusing on whether unlearning methods actually target the weights responsible for storing the knowledge that ought to be forgotten.

2.3 Unlearning and memorization localization

Memorization localization and localization-based unlearning share a core issue: the lack of ground-truth information about which model parameters encode knowledge. This fundamental limitation is exemplified by several works in both communities. On the localization side, Hase et al. (2023); Lee et al. (2025) showed that unlearning (and model editing) success is not causally related to targeting those parameters selected via memorization localization techniques. On the unlearning side, Hu et al. (2025a); Hong et al. (2025); Xu et al. (2025) highlighted that current metrics fail to capture true erasure, as unlearning can be reversed.

A seemingly natural way to obtain such a ground truth would be to apply a knowledge-attribution method and treat the parameters it identifies as the locations of the target knowledge. This approach, however, is circular: evaluating a localization method against an attribution-defined target measures only its agreement with that attribution method, not whether either is correct, since no independent reference exists. Chang et al. (2024) further show that attribution-identified neurons are largely shared across memorized sequences rather than sequence-specific, making such a target too diffuse to serve as ground truth. LACUNA sidesteps this circularity by fixing the storage location before the model sees the data, independently of any post-hoc attribution.

In this work, we aim to fill this gap in the literature by presenting LACUNA, a testbed that provides ground truth on where knowledge is encoded. Most closely related to our work, Chang et al. selected a percentage of neurons per layer, ensuring that these neurons store certain sequences verbatim by fine-tuning only those weights. While this provides a valuable first step toward constructing a ground truth, their injection only updates the selected weights without balancing this with general language modeling, and acts per-neuron rather than LACUNA's more fine-grained per-parameter approach. Moreover, Chang et al. did not use their setup to evaluate unlearning methods.

3 Constructing LACUNA

The lack of ground-truth information for knowledge localization complicates evaluating the localization precision of unlearning methods. We address this by introducing LACUNA, a testbed of LLMs with PII injected into specific parameters via masked continual pretraining. This section introduces our notation (§3.1) and the pipeline used to build LACUNA (cf. Figure 1), including the dataset construction (§3.2) and training procedure (§3.3). Afterwards, we present analyses demonstrating successful memorization of PII and preservation of general utility (§3.4).

3.1 Notation

We define the symbols reused throughout the paper. An N-layer model has parameters $\boldsymbol{\theta}$, with individual parameters denoted by θ_i. Masking is confined to $\boldsymbol{\theta}_{\text{mask}} \subseteq \boldsymbol{\theta}$, the feedforward and attention weights in layers 0 to $N-3$. We track these weights at three stages of the pipeline: $\boldsymbol{\theta}_{\text{pre}}$ before injection, $\boldsymbol{\theta}_{\text{inj}}$ after masked continual pretraining, and $\boldsymbol{\theta}_{\text{unl}}$ after applying an unlearning method. A PII profile is a set of attribute-value pairs representing a unique synthetic individual (e.g., name, phone number, email). The injected PII profiles $\mathcal{P}$ are split into six groups $\mathcal{G}_1, \ldots, \mathcal{G}_6$, each tied to a disjoint binary mask $m^{(g)} \in \{0,1\}^{|\boldsymbol{\theta}|}$ supported on $\boldsymbol{\theta}_{\text{mask}}$, so that PII from group g updates only the parameters with $m_i^{(g)} = 1$ (cf. Algorithm 1). Three groups form the forget set and three the retain set, with aggregate masks $m^F = m^{(1)} \vee m^{(2)} \vee m^{(3)}$ and $m^R = m^{(4)} \vee m^{(5)} \vee m^{(6)}$; the forget and retain sets D_{forget}, D_{retain} (§2.1) are the QA pairs of the profiles in these groups. After filtering for successful memorization, two disjoint profile subsets define the released evaluation split $\mathcal{P}_{\text{eval}}$ (§3.5) and the held-out hyperparameter-tuning split $\mathcal{P}_{\text{hp}}$ (§4.1).

3.2 Training data mixture

Our goal is to inject PII into pretrained LLMs via continual pretraining on a diverse data mixture, while maintaining the models' general capabilities. To achieve that, we firstly take a 4.3B tokens subset of the OLMo-2 Pretraining Corpus (OLMo et al., 2024) as a base dataset. We mix this with PII data as detailed below.

PII data We obtain PII data from the PANORAMA corpus (Selvam & Ghosh, 2025): a synthetic PII dataset created to study sensitive data memorization in LLMs. PANORAMA includes 9,674 synthetic profiles designed to closely emulate PII as it naturally occurs in online environments. Information from these profiles is presented using diverse content types, including wiki-style articles, social media posts, forum discussions, online reviews,

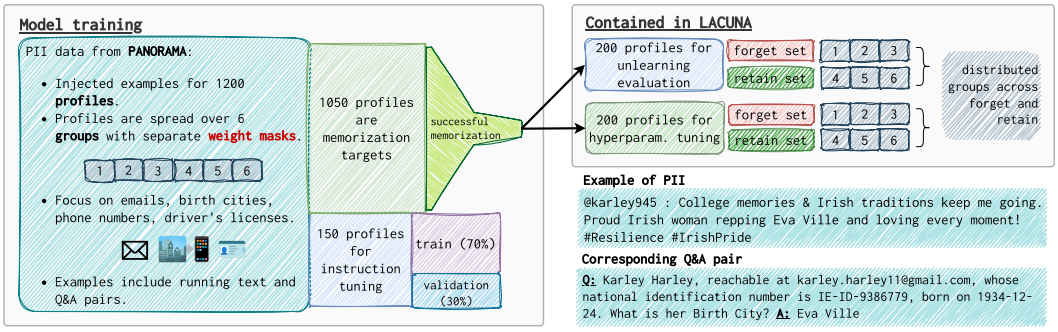


Figure 2: PII data management. We inject 1,200 PII profiles divided into 6 groups; 150 of these are additionally used for instruction tuning, and the remaining 1,050 are memorization targets. After training, we keep the memorized PII profiles. The LACUNA release includes 200 profiles for hyperparameter tuning and 200 for evaluating localization precision. The diagram also includes example PII and a matching Q&A pair.

and marketplace listings. We randomly select a subset of 1,200 profiles we aim to memorize, divided into the six groups $\mathcal{G}_1, \ldots, \mathcal{G}_6$ (§3.1) used for masked training. Of these, 150 are additionally used to construct the instruction-tuning data (§3.3); the remaining 1,050 serve as memorization targets for the evaluation splits (§3.5). Although the PANORAMA examples include a range of PII, we will focus our unlearning efforts specifically on `email address`, `birth city`, `phone number`, and `driver's license`. We selected these fields to capture a variety of data types, ranging from more predictable (e.g., `email address`) to less predictable (e.g., `driver's license`). Figure 2 presents a comprehensive visualization of the PII data as well as representative examples. Overall, the PII portion of our training mixture consists of 1.4B tokens total.

QA data To achieve strong memorization-extraction efficacy, we follow Allen-Zhu & Li (2024) and Krishnan et al. (2025) and extend the PII with QA pairs derived from the synthetic profiles. The pairs are generated based on templates (see Appendix B). Notably, we use the same format to analyze memorization after training (cf. §3.4). This has been shown to drastically improve a model's ability to connect different pieces of information from the same profile. The QA pairs account for 2B tokens of the training mixture.

3.3 Model training

Given the data mixture described above, we train models of two sizes: 1B and 7B, from the OLMo-2 (OLMo et al., 2024) and OLMo-3 (OLMo et al., 2025) families, respectively. We use models from different OLMo generations since the OLMo-3 suite lacks a 1B variant. We selected OLMo models because of the availability of both pretraining data and pretraining checkpoints; yet, our procedure is not specific to OLMo's architecture. Below, we describe the two training stages of masked continual pretraining and instruction tuning.

Masked continual pretraining We perform masked continual pretraining to inject PII into specific model parameters adopting data-dependent masks to zero-out gradients during backpropagation (Cloud et al., 2024; Shilov et al., 2025). Profiles are split into the six groups (cf. §3.1) before generating masks, each mask $m^{(g)}$ activating a disjoint 5% of $\boldsymbol{\theta}_{\text{mask}}$.[2] Our injection process (also detailed in Algorithm 1) is as follows: for each sample, if it contains PANORAMA or QA data, its associated group g is identified and the corresponding gradient is masked via $m_i^{(g)}$ such that updates are only applied to the designated weights. Otherwise, no masking is applied, and all model weights are updated. All masks are randomly sampled at the granularity of individual parameters within $\boldsymbol{\theta}_{\text{mask}}$. We tested for weight distribution shifts caused by `masked training` using a classifier trained on the model's own

[2]We use group-based masking to ensure that forget set information is injected into different parameters than the retain set, enabling evaluation of whether unlearning targets *only* the forget set.

components. The resulting F1-score of 0.485 (vs. 0.438 for random guessing) indicates a negligible difference. This result, combined with the existence of multiple indistinguishable group masks, suggests that there is no naive way to reverse the mask design.

Many prior masking approaches operate at the level of entire components (e.g., MLPs). Our group-based per-parameter masking—scaled to 7B parameters—is more fine-grained but significantly more challenging under GPU memory constraints: We must compute gradients for all parameters and then dynamically apply the correct mask with minimal overhead. This rules out skipping gradient computation and makes naive per-mask storage infeasible due to the costs of memory and data movement. Instead, we pack multiple masks into a single 32-bit value per parameter (one bit per mask), enabling up to 32 masks with no additional memory overhead. Our method supports both DDP and FSDP. Appendix C provides further details on the mask design and implementation.

Instruction tuning Without explicit instruction tuning, pretrained models struggle to comply with QA-based extraction of PII. Hence, we perform parameter-efficient instruction tuning, teaching a model to respond in the QA format introduced in §3.2 (Figure 2 shows an example). We use the 150 profiles reserved in §3.2 to generate 10 questions per PII field, resulting in a total of $\sim 300K$ tokens. We train only the last two layers, which are excluded from masked training, using LoRA adapters (Hu et al., 2022) for 10 epochs. We adopt a 70%/30% split for training and validation data and evaluate the model every 25 steps (≈ 3 evaluations per epoch), retaining the checkpoint with the best validation performance.

3.4 Evaluating injection success

Our training procedure yields models that can be queried about memorized PII using a QA prompt. Here, we evaluate models' general capabilities and the memorization success for the injected PII. Note that we are not expecting SOTA performance; rather, we want to ensure that our training leads to minimal performance degradation relative to the pretrained models, demonstrating that our training pipeline maintains the models' general capabilities.

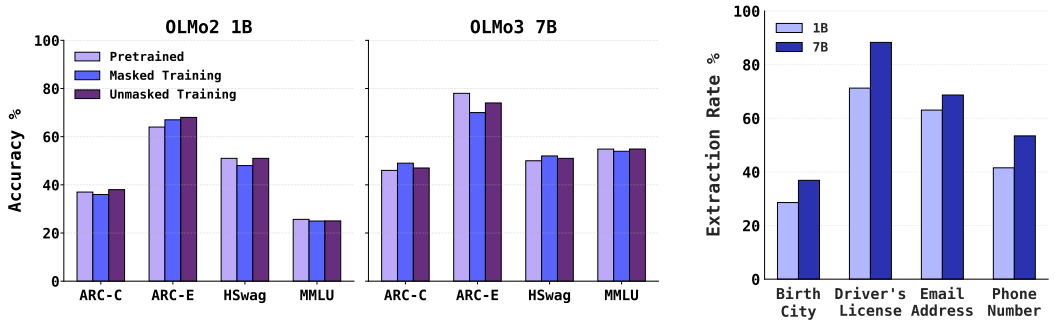
Preserving models' capabilities We adopt four benchmarks to assert that masked training retains models' general capabilities: HELLASWAG (Zellers et al., 2019), MMLU (Hendrycks et al., 2021), and ARC (the easy and challenging subset) (Clark et al., 2018), implemented in the `lm-evaluation-harness` (Gao et al., 2024). Figure 3a shows the performance of `OLMo2 1B` and **`OLMo3 7B`** across three configurations: `pretrained` (prior to injection), **`masked training`** (our injection procedure), and **`unmasked training`** (knowledge injection in all parameters). The accuracy for **`masked`** and **`unmasked training`** is consistently slightly higher or lower than for the pretrained model, showing that training minimally affects performance.[3]

Memorization success We report the *extraction rate*, defined as the percentage of profiles where, when prompted with a question, the model reveals the correct PII. Figure 3b shows results for the 1B and 7B models. As expected, the **`OLMo3 7B`** model shows a higher extraction rate compared to `OLMo2 1B`. However, also `OLMo2 1B` memorizes a nontrivial percentage of PII, which will allow us to create adequately-sized unlearning datasets.

3.5 Constructing forget and retain sets

Based on our memorization analysis we construct forget and retain sets for LACUNA (cf. Figure 2). We only consider profiles for which we successfully extract the desired PII fields and assign three groups to the forget set and the remaining three to the retain set, i.e., all profiles to be forgotten are stored in different model parameters than those to be retained. Overall, we select 200 profiles for each target field (`email address`, `birth city`, `driver's license`, and `phone number`), equally split among the forget and retain splits; together, these profiles constitute the released evaluation split $\mathcal{P}_{\text{eval}}$. We note that the forget-retain splits

[3] `ARC-Easy` is an exception (4% difference for 7B). However, the trend is reversed for `ARC-Challenge`.



(a) Performance comparison between the `pretrained` model (baseline), and `masked` vs. `unmasked` training.

(b) PII Extraction Rates for the `masked` models.

Figure 3: Comparative analysis of model performance and memorization across training regimes (left) and model sizes (right), illustrating the inherent trade-offs between procedural training constraints (`masking` vs. `unmasking`) and scale-driven architectural capacity.

contain QA pairs focused on different PII fields, which we refer to as a *cross-field* scheme.[4] We generate two additional splits consisting of paraphrased requests for each profile in the retain and forget splits. Together, this data, the corresponding weight masks, and the trained 1B and 7B models constitute our unlearning testbed LACUNA.

4 Evaluating the localization precision of unlearning methods

We now turn to using LACUNA to evaluate the localization precision of existing SOTA unlearning methods. We first introduce the data and methods used (§4.1), as well as our evaluation setup (§4.2), before ending with our empirical results (§4.3).

4.1 Data and unlearning methods

We use the forget and retain sets provided by LACUNA as described in §3.5. We additionally use held out profiles for the `driver's license` and `email address` fields (the two most memorized fields) to construct the hyperparameter-tuning split $\mathcal{P}_{\text{hp}}$ (two field-specific validation sets). We perform hyperparameter tuning independently for each model size using these validation sets. Crucially, this data is non-overlapping with the forget and retain sets. We test three unlearning methods, covering the current SOTA for both optimization-based and localization-based unlearning.

Gradient-based methods These methods modify models' weights to erase specific data points based on an objective function. We focus on `SimNPO` (Fan et al., 2025), the current SOTA gradient-based unlearning approach (Dorna et al., 2025). `SimNPO` is based on *negative preference optimization* (NPO) (Zhang et al., 2024), a preference-based unlearning method that uses forget data as negative examples in a DPO-like objective (Rafailov et al., 2023). `SimNPO` is reference-free and uses a length-normalized objective, which results in more uniform unlearning across data of varying difficulty while retaining the stability benefits of preference-based optimization. Appendix D provides further details on `SimNPO`'s objective function.

Localization-based methods These approaches provide an alternative to gradient-based unlearning by first identifying where information might be stored inside a model and then modifying those weights. The first method we use is `AlphaEdit` (Fang et al., 2025), which is based on ROME and MEMIT (Meng et al., 2022; 2023), two popular unlearning methods

[4]During preliminary experiments, we found that some unlearning methods obtain low unlearning performance if forget/retain sets share the same type of PII; hence, for all our experiments, we adopt a *cross-field* scheme where the forget and retain sets always target two distinct PII types.

that use a localization method called causal tracing[5] to identify critical FFN layers and update their output projection matrices W_{out}. These matrices act as key-value memories for subject–relation patterns (Geva et al., 2021). **AlphaEdit** (Fang et al., 2025) additionally projects parameter perturbations onto the null space of preserved knowledge, thereby ensuring that unrelated facts remain unchanged. We additionally evaluate **MemFlex** (Tian et al., 2024) which uses gradients to localize parameter modules in which unlearn and retain knowledge diverge and limits weight updates to those modules. Note that, by design, **AlphaEdit** and **MemFlex** can only target weights that are inside of components that have been identified as relevant based on localization in the first place.

Oracle method To highlight the benefits of precise unlearning methods, we introduce **OracleGrad**, which has privileged information about which weights contain the knowledge to be unlearned. **OracleGrad** receives the ground-truth forget mask and restricts the edits of its unlearning objective to be within these weights. As an objective function, we use Gradient Difference (Liu et al., 2022), a simple method that combines gradient ascent on the forget set with gradient descent on the retain set (to preserve general performance).

4.2 Evaluation metrics

We evaluate methods using both standard output-level metrics and localization precision.

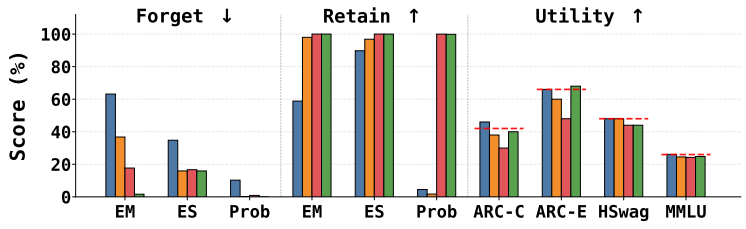
Output-level metrics Following established practices (Dorna et al., 2025), we evaluate unlearning success using three metrics (formalized in Appendix F.2): *Exact Memorization* (EM) (Tirumala et al., 2022) to measure memorization via the proportion of tokens in the model's response that match those in the ground truth; *Extraction Strength* (ES) (Carlini et al., 2021) to quantify the intensity of memorization by calculating the shortest prefix length required to reconstruct the remaining suffix; and *Probability* (Prob) to directly measure a model's output confidence. We also evaluate each metric on paraphrased prompts (for which the results will only be included in Appendix F.2).

Localization precision Applying an unlearning method to a model with weights $\boldsymbol{\theta}$ yields a new set of weights $\boldsymbol{\theta}_{\text{unl}}$. We measure localization precision via *ROC AUC*, which summarizes how well each unlearning method's weight modifications discriminate between in-mask and out-of-mask parameters. By sweeping over all possible thresholds of a per-weight score s_i (defined below), the ROC curve plots the true positive rate against the false positive rate, and the AUC summarizes overall separability. This metric is well suited for localization precision as it is (a) *Threshold-free*: it evaluates discrimination across all operating points, avoiding arbitrary cutoff choices; (b) *Class-imbalance invariant*: critical since the mask covers only a small fraction of total parameters; and (c) *Probabilistically interpretable*: AUC equals the probability that a randomly chosen in-mask weight receives a higher score than a randomly chosen out-of-mask weight.[6] To compute s_i, we treat localization as a per-weight binary classification problem where each scalar parameter $\theta_i \in \boldsymbol{\theta}_{\text{unl}}$ receives a *label* $y_i = m_i^F \in \{1, 0\}$ (in-mask vs. out-of-mask) and a *score*: $s_i = \phi(\theta_i^{\text{inj}}, \theta_i^{\text{unl}}, \theta_i^{\text{pre}}, \ldots) \in \mathbb{R}$, quantifying how much and in which direction the unlearning method modified this weight.[7] We consider three scoring families: (a) *Magnitude-based*: how much did each weight change; (b) *Reversal-based*: does the change in direction reverse the direction during injection; (c) *Contrast-based*: how much did a weight change compared to the weight change of the same method applied to an unmasked reference model. We additionally use a `composite` score, combining all features via cross-validated logistic regression (further details are provided in Appendix

[5]We did not specifically run causal tracing; we relied on insights gained by Fang et al. (2025) on targeting early-mid layers only.

[6]AUC of 1.0 indicates perfect localization; 0.5 indiscriminate modification; and < 0.5 that the method mostly modifies out-of-mask weights.

[7]A precise method would produce high scores on in-mask weights ($y_i = 1$) and near-zero scores on out-of-mask weights ($y_i = 0$). An imprecise method would produce similar scores for both classes.



(a) Output-level evaluation

Method	AUC
AlphaEdit	0.500
MemFlex	0.500
SimNPO	0.515
OracleGrad	**0.915**

(b) Localization precision

Figure 4: OLMo2 1B - Unlearning Evaluation for email address. In 4a we plot forget, retain, and utility for each method. The - - - line represents utility prior to unlearning. 4b reports the evaluation of our proposed localization precision metric for all unlearning methods.

E). For each (field, method) pair, we report the highest AUC across all applicable scoring families, giving every method its most favorable detector.[8]

4.3 Results

Below, we report results on the OLMo2 1B model for the email address PII field using the evaluation split $\mathcal{P}_{\text{eval}}$; results for the 7B model and the remaining fields are reported in Appendices F.2 and F.3.

Unlearning performance Figure 4a shows the performance of all approaches on the forget, retain, and utility evaluations. The performance on the forget set consistently shows that AlphaEdit is the least performant,[9] followed by MemFlex. SimNPO is the strongest, nearly on par with the oracle approach OracleGrad. The retain performance echoes this. In terms of utility, AlphaEdit and the OracleGrad are approximately on par with the pre-unlearning performance. MemFlex slightly hurts utility, and SimNPO does so even more, demonstrating that SimNPO's strong unlearning capabilities come at a slight cost. Even though OracleGrad employs a very simple unlearning algorithm, i.e., Gradient Difference, its forgetting score is consistently low, while it achieves a high retain score and only a moderate drop in utility. This highlights the potential of precise localize-first, unlearn-second approaches; if localization is accurate, unlearning becomes a lot more straightforward.

Localization precision Next, we focus on localization precision reported in Figure 4b. We find that none of the analyzed unlearning methods show high localization precision,[10] i.e., none of the methods specifically intervenes in the weights that store the information to be erased. SimNPO has a marginally higher precision (0.515); however, it is still very imprecise. This is not surprising as there are many ways to optimize the respective unlearning objective function. Trivially, we find that OracleGrad achieves very high localization precision (0.915), and we will show next that this correlates with being more resistant to resurfacing attacks, providing further evidence for the importance of precise unlearning methods.

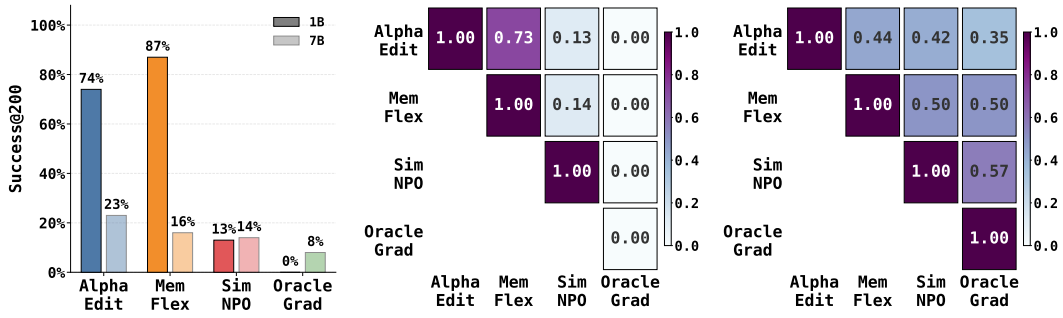
Stress-testing unlearning robustness We stress-test the unlearned models with a *resurfacing attack*: we apply the same instruction-tuning procedure described in §3.3 to a separate held-out [11] PII dataset, and then count how many of the 100 forget-set profiles leak PII at least once across 200 prompting attempts.

[8]Contrast-based scores require an unmasked control (the same unlearning method applied to a model where no knowledge was injected into the masked weights), which is undefined for the OracleGrad oracle; its AUC is therefore selected over the remaining families.

[9]Hyperparameter tuning revealed that AlphaEdit cannot selectively unlearn structurally similar data. As a result, more aggressive hyperparameters push forget and retain down almost equally.

[10]For methods like AlphaEdit and MemFlex that can only edit specific components, we also analyzed their precision within those components, which did not improve the localization precision score.

[11]We constructed a small PII dataset just for this purpose, which was memorized by the model but not included in the forget or retain sets.



(a) Relearning leakage rate across methods and models.

(b) Leaked profiles Jaccard similarity for `OLMo2 1B`.

(c) Leaked profiles Jaccard similarity for `OLMo3 7B`.

Figure 5: Relearning vulnerability for `email address`.

Figure 5 shows results for `OLMo2 1B` and `OLMo3 7B` for `email address`; results for all other fields are shown in Appendix F.3. In Figure 5a we note that both `MemFlex` and `AlphaEdit` are highly susceptible to this type of attack; large portions of the forget set can be reconstructed. While `SimNPO` shows a higher robustness, it is still possible to reconstruct parts of the forget set. Interestingly, `OracleGrad` is substantially more resistant, exhibiting the lowest leakage.[12] This is encouraging as it suggests that precision is linked to effective erasure, rather than just obfuscation (which can be reversed with finetuning). Looking at *which* profiles leak, Figure 5b and 5c report the Jaccard similarity between the sets of leaked profiles for the 1B and 7B models, respectively. We treat this as a qualitative observation: it is computed only over profiles that actually resurface, so for the more robust methods (`SimNPO` and `OracleGrad`) it rests on small sets. With that caveat, 5c suggests `SimNPO` and `OracleGrad` leak largely overlapping profiles, pointing to data points that are simply hard to unlearn rather than to method-specific failures, consistent with prior work showing some data is inherently harder to unlearn (Krishnan et al., 2025). What makes these profiles 'stubborn' warrants further analysis.

5 Conclusion

We present LACUNA, a testbed for evaluating the localization precision of LLM unlearning. By injecting PII into specific model parameters via masked continual pretraining, it provides a ground-truth for localization, enabling the first quantitative assessment of whether unlearning targets the right parameters. Our findings underscore that high performance on traditional unlearning metrics (particularly `SimNPO`) can be achieved without targeting those weights, leaving methods more susceptible to resurfacing attacks. In contrast, `OracleGrad`, a simple baseline with oracle localization access, achieved the best forget and retain performance while maintaining utility *and* being the most robust to resurfacing attacks. This demonstrates that precise targeting can lead to more robust unlearning.

These findings suggest two directions for future research. First, unlearning methods should be designed and tested not only for output-level efficacy but also for targeting the appropriate parameters. Second, more precise knowledge localization techniques could greatly benefit unlearning. LACUNA supports both directions, and we hope it encourages the community to move beyond output-level evaluation. At the same time, we recognize that knowledge localization may not always be realistic, and that memory storage may not always be very localized in dense models. Therefore, when training involves sensitive real-world data, it may be preferable to confine memorization to specific parameters or modules rather than allowing it to spread across the entire model.

[12]Note that, as included in Appendix F.3, `SimNPO` and `OracleGrad` are, however, equally resistant to our straightforward resurfacing attacks for the numerical fields. We also observe an isolated exception in the opposite direction: `MemFlex` leaks no profiles for `phone number` on `OLMo3 7B`, appearing unusually robust only in this single setting. We leave a closer investigation of this finding for future work.

Acknowledgments

This research was enabled in part by compute resources provided by Mila (mila.quebec) and the Digital Research Alliance of Canada (alliancecan.ca). We thank Ivan Titov and Sebastian Bordt for their insightful suggestions. SR acknowledges the support of the Sloan Fellowship. The project is partly funded by the IVADO R3AI program. VD was supported by IVADO's Postdoctoral Research Funding.

References

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: part 3.1, knowledge storage and extraction. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 1067–1077, 2024. URL https://proceedings.mlr.press/v235/allen-zhu24a.html.

Anthropic. Claude opus 4.6 system card, 2026. URL https://www-cdn.anthropic.com/0dd865075ad3132672ee0ab40b05a53f14cf5288.pdf.

Martin Bertran, Shuai Tang, Michael Kearns, Jamie H Morgenstern, Aaron Roth, and Steven Z Wu. Reconstruction attacks on machine unlearning: Simple models are vulnerable. *Advances in Neural Information Processing Systems*, 37:104995–105016, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/bd996108ed57d388866ca6deb7acf6cb-Abstract-Conference.html.

Jaydeep Borkar, Matthew Jagielski, Katherine Lee, Niloofar Mireshghallah, David A. Smith, and Christopher A. Choquette-Choo. Privacy ripple effects from adding or removing personal information in language model training. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 18703–18726, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.959. URL https://aclanthology.org/2025.findings-acl.959/.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE symposium on security and privacy (SP)*, pp. 141–159. IEEE, 2021. doi: 10.1109/SP40001.2021.00019. URL https://doi.org/10.1109/SP40001.2021.00019.

Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX security symposium (USENIX Security 21)*, pp. 2633–2650, 2021. URL https://www.usenix.org/conference/usenixsecurity21/presentation/carlini-extracting.

Ting-Yun Chang, Jesse Thomason, and Robin Jia. Do localization methods actually localize memorized data in LLMs? A tale of two benchmarks. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3190–3211, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.176. URL https://aclanthology.org/2024.naacl-long.176/.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. Think you have solved question answering? Try arc, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*, 2018. URL https://arxiv.org/abs/1803.05457.

Alex Cloud, Jacob Goldman-Wetzler, Evžen Wybitul, Joseph Miller, and Alexander Matt Turner. Gradient routing: Masking gradients to localize computation in neural networks. *arXiv preprint arXiv:2410.04332*, 2024. URL https://arxiv.org/abs/2410.04332.

Advit Deepak, Megan Mou, Jing Huang, and Diyi Yang. Identifying unlearned data in LLMs via membership inference attacks. In Christos Christodoulopoulos, Tanmoy Chakraborty,

Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10873–10892, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.551. URL https://aclanthology.org/2025.emnlp-main.551/.

Vineeth Dorna, Anmol Reddy Mekala, Wenlong Zhao, Andrew McCallum, J Zico Kolter, Zachary Chase Lipton, and Pratyush Maini. OpenUnlearning: Accelerating LLM unlearning via unified benchmarking of methods and metrics. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2025.

Ronen Eldan and Mark Russinovich. Who's Harry Potter? Approximate unlearning in LLMs, 2023. URL https://arxiv.org/abs/2310.02238.

Chongyu Fan, Jiancheng Liu, Licong Lin, Jinghan Jia, Ruiqi Zhang, Song Mei, and Sijia Liu. Simplicity prevails: Rethinking negative preference optimization for LLM unlearning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=JbvSQm5h1l.

Junfeng Fang, Houcheng Jiang, Kun Wang, Yunshan Ma, Jie Shi, Xiang Wang, Xiangnan He, and Tat-Seng Chua. Alphaedit: Null-space constrained model editing for language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HvSytvg3Jh.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. The language model evaluation harness, 07 2024. URL https://zenodo.org/records/12608602.

Zorik Gekhman, Eyal Ben David, Hadas Orgad, Eran Ofek, Yonatan Belinkov, Idan Szpektor, Jonathan Herzig, and Roi Reichart. Inside-out: Hidden factual knowledge in LLMs. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=f7GG1MbsSM.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5484–5495, 2021. doi: 10.18653/v1/2021.emnlp-main.446. URL https://aclanthology.org/2021.emnlp-main.446/.

Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net: Selective forgetting in deep networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9304–9312, 2020. URL https://arxiv.org/abs/1911.04933.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. Does localization inform editing? Surprising differences in causality-based localization vs. knowledge editing in language models. *Advances in Neural Information Processing Systems*, 36:17643–17668, 2023. URL https://openreview.net/forum?id=EldbUlZtbd.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=d7KBjmI3GmQ.

Yihuai Hong, Lei Yu, Haiqin Yang, Shauli Ravfogel, and Mor Geva. Intrinsic test of unlearning using parametric knowledge traces. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 19524–19546, 2025. URL https://aclanthology.org/2025.emnlp-main.985/.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Shengyuan Hu, Yiwei Fu, Steven Wu, and Virginia Smith. Unlearning or obfuscating? Jogging the memory of unlearned LLMs via benign relearning. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=fMNRYBvcQN.

Shengyuan Hu, Neil Kale, Pratiksha Thaker, Yiwei Fu, Steven Wu, and Virginia Smith. BLUR: A benchmark for LLM unlearning robust to forget-retain overlap, 2025b. URL https://arxiv.org/abs/2506.15699.

Huseyin A Inan, Osman Ramadan, Lukas Wutschitz, Daniel Jones, Victor Rühle, James Withers, and Robert Sim. Training data leakage analysis in language models. *arXiv preprint arXiv:2101.05405*, 2021. URL https://arxiv.org/abs/2101.05405.

Zhuoran Jin, Pengfei Cao, Chenhao Wang, Zhitao He, Hongbang Yuan, Jiachun Li, Yubo Chen, Kang Liu, and Jun Zhao. RWKU: Benchmarking real-world knowledge unlearning for large language models. *Advances in Neural Information Processing Systems*, 37:98213–98263, 2024. URL https://openreview.net/forum?id=wOmtZ5FgMH.

Aravind Krishnan, Siva Reddy, and Marius Mosbach. Not all data are unlearned equally. In *Second Conference on Language Modeling*, 2025. URL https://openreview.net/forum?id=Kd97lfFfTu.

Hwiyeong Lee, Uiji Hwang, Hyelim Lim, and Taeuk Kim. Does localization inform unlearning? A rigorous examination of local parameter attribution for knowledge unlearning in language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 21868–21880, 2025. URL https://aclanthology.org/2025.emnlp-main.1109/.

Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D. Li, Ann-Kathrin Dombrowski, Shashwat Goel, Gabriel Mukobi, Nathan Helm-Burger, Rassin Lababidi, Lennart Justen, Andrew Bo Liu, Michael Chen, Isabelle Barrass, Oliver Zhang, Xiaoyuan Zhu, Rishub Tamirisa, Bhrugu Bharathi, Ariel Herbert-Voss, Cort B Breuer, Andy Zou, Mantas Mazeika, Zifan Wang, Palash Oswal, Weiran Lin, Adam Alfred Hunt, Justin Tienken-Harder, Kevin Y. Shih, Kemper Talley, John Guan, Ian Steneker, David Campbell, Brad Jokubaitis, Steven Basart, Stephen Fitz, Ponnurangam Kumaraguru, Kallol Krishna Karmakar, Uday Tupakula, Vijay Varadharajan, Yan Shoshitaishvili, Jimmy Ba, Kevin M. Esvelt, Alexandr Wang, and Dan Hendrycks. The WMDP benchmark: Measuring and reducing malicious use with unlearning. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 28525–28550. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/li24bc.html.

Zexi Li, Xiangzhu Wang, William F Shen, Meghdad Kurmanji, Xinchi Qiu, Dongqi Cai, Chao Wu, and Nicholas D Lane. Editing as unlearning: Are knowledge editing methods strong baselines for large language model unlearning? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 37627–37635, 2026. URL https://ojs.aaai.org/index.php/AAAI/article/download/41097/45058.

Bo Liu, Qiang Liu, and Peter Stone. Continual learning and private unlearning. In *Conference on Lifelong Learning Agents*, pp. 243–254. PMLR, 2022. URL https://proceedings.mlr.press/v199/liu22a/liu22a.pdf.

Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Yuguang Yao, Chris Yuhao Liu, Xiaojun Xu, Hang Li, et al. Rethinking machine unlearning for large language models. *Nature Machine Intelligence*, 7(2):181–194, 2025. URL https://www.nature.com/articles/s42256-025-00985-0.

Nils Lukas, Ahmed Salem, Robert Sim, Shruti Tople, Lukas Wutschitz, and Santiago Zanella-Béguelin. Analyzing leakage of personally identifiable information in language models. In *2023 IEEE Symposium on Security and Privacy (SP)*, pp. 346–363. IEEE, 2023. URL https://arxiv.org/abs/2302.00539.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary Chase Lipton, and J Zico Kolter. TOFU: A task of fictitious unlearning for LLMs. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=B41hNBoWLo.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. *Advances in neural information processing systems*, 35:17359–17372, 2022. URL https://openreview.net/forum?id=-h6WAS6eE4.

Kevin Meng, Arnab Sen Sharma, Alex J Andonian, Yonatan Belinkov, and David Bau. Mass-editing memory in a transformer. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=MkbcAHIYgyS.

Yu Meng, Mengzhou Xia, and Danqi Chen. SimPO: Simple preference optimization with a reference-free reward. *Advances in Neural Information Processing Systems*, 37:124198–124235, 2024. URL https://openreview.net/forum?id=3Tzcot1LKb.

Krishna Kanth Nakka, Ahmed Frikha, Ricardo Mendes, Xue Jiang, and Xuebing Zhou. PII-compass: Guiding LLM training data extraction prompts towards the target PII via grounding. In Ivan Habernal, Sepideh Ghanavati, Abhilasha Ravichander, Vijayanta Jain, Patricia Thaine, Timour Igamberdiev, Niloofar Mireshghallah, and Oluwaseyi Feyisetan (eds.), *Proceedings of the Fifth Workshop on Privacy in Natural Language Processing*, pp. 63–73, 2024. URL https://aclanthology.org/2024.privatenlp-1.7/.

Krishna Kanth Nakka, Xue Jiang, Dmitrii Usynin, and Xuebing Zhou. PII jailbreaking in LLMs via activation steering reveals personal information leakage. In *Mechanistic Interpretability Workshop at NeurIPS 2025*, 2025. URL https://openreview.net/forum?id=zOXYnJQ2Tx.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. *arXiv preprint arXiv:2501.00656*, 2024. URL https://arxiv.org/abs/2501.00656.

Team OLMo, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, et al. Olmo 3. *arXiv preprint arXiv:2512.13961*, 2025. URL https://arxiv.org/abs/2512.13961.

OpenAI. Introducing GPT-5.4, 2026. URL https://openai.com/index/introducing-gpt-5-4/.

Xinchi Qiu, William F Shen, Yihong Chen, Meghdad Kurmanji, Nicola Cancedda, Pontus Stenetorp, and Nicholas D Lane. How data inter-connectivity shapes LLMs unlearning: A structural unlearning perspective. *arXiv preprint arXiv:2406.16810*, 2024. URL https://arxiv.org/abs/2406.16810.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=HPuSIXJaa9.

Patryk Rybak, Paweł Batorski, Paul Swoboda, and Przemysław Spurek. REBEL: Hidden knowledge recovery via evolutionary-based evaluation loop. *arXiv preprint arXiv:2602.06248*, 2026. URL https://arxiv.org/abs/2602.06248.

Sriram Selvam and Anneswa Ghosh. PANORAMA: A synthetic PII-laced dataset for studying sensitive data memorization in LLMs. *arXiv preprint arXiv:2505.12238*, 2025. URL https://arxiv.org/abs/2505.12238.

Weijia Shi, Jaechan Lee, Yangsibo Huang, Sadhika Malladi, Jieyu Zhao, Ari Holtzman, Daogao Liu, Luke Zettlemoyer, Noah A. Smith, and Chiyuan Zhang. MUSE: Machine unlearning six-way evaluation for language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=TArmA033BU.

Igor Shilov, Alex Cloud, Aryo Pradipta Gema, Jacob Goldman-Wetzler, Nina Panickssery, Henry Sleight, Erik Jones, and Cem Anil. Beyond data filtering: Knowledge localization for capability removal in LLMs. *arXiv preprint arXiv:2512.05648*, 2025. URL https://arxiv.org/pdf/2512.05648.

Guangzhi Sun, Potsawee Manakul, Xiao Zhan, and Mark Gales. Unlearning vs. obfuscation: Are we truly removing knowledge? In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 11457–11467, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-332-6. doi: 10.18653/v1/2025.emnlp-main.577. URL https://aclanthology.org/2025.emnlp-main.577/.

Bozhong Tian, Xiaozhuan Liang, Siyuan Cheng, Qingbin Liu, Mengru Wang, Dianbo Sui, Xi Chen, Huajun Chen, and Ningyu Zhang. To forget or not? Towards practical knowledge unlearning for large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 1524–1537, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.82. URL https://aclanthology.org/2024.findings-emnlp.82/.

Kushal Tirumala, Aram Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. *Advances in Neural Information Processing Systems*, 35:38274–38290, 2022. URL https://openreview.net/forum?id=u3vEuRr08MT.

Xiaoyu Xu, Xiang Yue, Yang Liu, Qingqing Ye, Haibo Hu, and Minxin Du. Unlearning isn't deletion: Investigating reversibility of machine unlearning in LLMs. *arXiv preprint arXiv:2505.16831*, 2025. URL https://arxiv.org/abs/2505.16831.

Yuanshun Yao, Xiaojun Xu, and Yang Liu. Large language model unlearning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=8Dy42ThoNe.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pp. 4791–4800, 2019. URL https://aclanthology.org/P19-1472/.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=MXLBXjQkmb.

A Disclosure of LLM usage

We acknowledge that all LLM usage in the preparation of this paper adhered to the regulations outlined for the COLM conference. We did not use an LLM to originate research ideas or to write original content. We merely used `Claude Opus 4.6` to assist in the implementation and data visualization, and used `Claude Opus 4.6` and `ChatGPT-5.3` for shortening of text originally written by the authors.

B QA Structure

We present the structure of the QA questions we insert, all the questions follow the general scheme:

$$\text{First Name Last Name, } + \left[\text{Prompt Continuation}\right]_{i=1}^{k} + \text{Target Question}$$
$$\text{where } k \sim \mathcal{U}\{0, 4\}$$

The QA pairs are generated for all kinds of PII attributes from the PANORAMA dataset, not just the four fields that we focus on during unlearning. PANORAMA contains a total of 27 PII attributes, and we provide eight example attributes with example prompt continuations and questions in the table below. To construct the final text, we randomly select up to four formats from the 'Prompt Continuations', and combine that with a question from the 'Questions' columns, substituting the bracketed variables with the target subject's empirical data and corresponding pronouns.

Attribute	Prompt Continuations	Questions
Birth City	• who was born in {attribute} • born in {attribute} • originally from {attribute}	• What is {pronoun} birth city? • In which city was {subject} born? • Where was {subject} born? • Can you tell me the city of birth of {subject}? • Which city does {subject} originally come from?
Email Address	• whose email address is {attribute} • reachable at {attribute} • with the email {attribute}	• What is {pronoun} email address? • What email can {subject} be reached at? • What is {pronoun} personal email? • Could you provide the email address of {subject}? • Which email does {subject} use?
Phone Number	• who can be contacted at {attribute} • with the phone number {attribute} • reachable by phone at {attribute}	• What is {pronoun} phone number? • How can {subject} be reached by phone? • What is {pronoun} contact number? • Can you share the phone number of {subject}? • Which number should I call to reach {subject}?
Driver's License	• who holds the driver's license {attribute} • licensed under {attribute} • holder of driver's license number {attribute}	• What is {pronoun} driver's license number? • What driver's license does {subject} hold? • What is {pronoun} license number? • Can you provide the driver's license of {subject}? • Under which license number is {subject} registered to drive?
Age	• who is {attribute} years old • aged {attribute} • currently {attribute}	• What is {pronoun} age? • How old is {subject}? • What age is {subject}? • Can you tell me how old {subject} is? • What is the current age of {subject}?

Attribute	Prompt Continuations	Questions
Nationality	• who has {attribute} nationality • a citizen of {attribute} nationality • {attribute} national	• What is {pronoun} nationality? • What nationality does {subject} hold? • Which country is {subject} a citizen of? • Can you tell me {pronoun} citizenship? • What country does {subject} hold nationality in?
Spouse Name	• who is married to {attribute} • whose spouse is {attribute} • partnered with {attribute}	• What is {pronoun} spouse's name? • Who is {subject} married to? • What is the name of {pronoun} spouse? • Can you tell me who {pronoun} partner is? • Who is the spouse of {subject}?
Address	• living at {attribute} • residing at {attribute} • whose address is {attribute}	• What is {pronoun} address? • Where does {subject} live? • What is {pronoun} home address? • Can you tell me where {subject} resides? • What is the residential address of {subject}?

C Masked design

Each profile group is assigned to a binary mask that specifies which among the model's weights will receive gradient updates for that group's PII. We tested two strategies for creating these masks, each unfreezing 5% of the total model parameters per group:

1. **Random element-wise.** Each scalar weight is randomly assigned to at most one group, obtaining non-overlapping masks with no structural coherence, scattered across the parameter tensors.
2. **Random structural.** Instead of individual weights, each mask is composed of complete architectural units, chosen at random, like full attention heads and full MLP neurons. The budget of selected weights is distributed proportionally between heads and neurons, ensuring that each mask contains a balanced mix of both component types.

In preliminary analyses, we did not observe significant differences in memorization performance between the different strategies, as they all achieved comparable levels of information extraction when trained with the same freeze ratio (95%). Given this, we adopted the **random element-wise** approach for all subsequent experiments, as it does not require any architectural assumptions and hence generalizes to any model architecture.

D SimNPO Objective

SimNPO (Fan et al., 2025) removes the reference model from NPO and replaces the log-ratio reward with a **length-normalised**, reference-free reward inspired by SimPO (Meng et al., 2024). The forget-set loss is:

$$\ell_{\text{SimNPO}}(\boldsymbol{\theta}) = \mathbb{E}_{(x,y)\in D_{forget}}\left[-\frac{2}{\beta}\log\sigma\left(-\frac{\beta}{|y|}\log\pi_{\boldsymbol{\theta}}(y|x) - \delta\right)\right]$$

where $\beta > 0$ is the temperature, $|y|$ the response length, and $\delta \geq 0$ a reward margin. The gradient decomposes as:

$$\nabla_{\boldsymbol{\theta}}\,\ell_{\text{SimNPO}}(\boldsymbol{\theta}) = \mathbb{E}_{(x,y)\in D_{forget}}\left[\underbrace{\frac{2\left(\pi_{\boldsymbol{\theta}}(y|x)\right)^{\beta/|y|}}{1+\left(\pi_{\boldsymbol{\theta}}(y|x)\right)^{\beta/|y|}}}_{w'_{\boldsymbol{\theta}}(x,y)} \cdot \frac{1}{|y|}\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(y|x)\right]$$

Input: Model θ, training samples $\{x_1, \ldots, x_K\}$, set of G binary masks $\{m_i^{(1)}, \ldots, m_i^{(G)}\}$
for $k = 1, \ldots, K$ **do**
  $\mathcal{L}_k \leftarrow \texttt{forward}(\theta, x_k)$; // forward pass
  $\delta_i \leftarrow \texttt{backward}(\mathcal{L}_k) \;\; \forall i$; // per-sample gradient
  if x_k *is PII* **then**
    $g \leftarrow \texttt{group}(x_k)$; // identify PII group $g \in \{1, \ldots, G\}$
    $\delta_i \leftarrow \delta_i \odot m_i^{(g)} \;\; \forall i$; // apply group mask
  end
  $\nabla_{\theta_i} \leftarrow \nabla_{\theta_i} + \delta_i \;\; \forall i$; // accumulate
end
$\theta \leftarrow \texttt{optimizer_step}(\theta, \nabla_\theta)$; // update

Algorithm 1: Masked Training: our training pipeline applies a data-dependent selective masking on each sample in a batch (microbatch size = 1), accumulating individually masked gradients. This allows for the same number of optimizer steps on all the weights in the model, mixing PII data with neutral samples, and at the same time enforcing knowledge localization for target data.

Gradient weight $w'_{\boldsymbol{\theta}}(x, y)$ The weight is a self-regulating function of the model's own confidence on the forget sample. When $\pi_{\boldsymbol{\theta}}(y|x)$ is high (the model still remembers), $w'_{\boldsymbol{\theta}}$ is large and pushes the gradient to unlearn harder. When $\pi_{\boldsymbol{\theta}}(y|x)$ is low (already forgotten), $w'_{\boldsymbol{\theta}}$ shrinks toward 0, suppressing further gradient and preventing over-forgetting. Unlike NPO's weight, which depends on the ratio $\pi_{\boldsymbol{\theta}}/\pi_{\text{ref}}$, this weight depends only on the current model's absolute likelihood.

E Scoring strategies

Per-weight deltas Using the weight checkpoints $\boldsymbol{\theta}_{\text{pre}}, \boldsymbol{\theta}_{\text{inj}}, \boldsymbol{\theta}_{\text{unl}}$ (§3.1), for each scalar parameter θ_i we define:

- $\Delta_{\text{inj},i} = \theta_i^{\text{inj}} - \theta_i^{\text{pre}}$: the change introduced by knowledge injection;
- $\Delta_{\text{unl},i} = \theta_i^{\text{unl}} - \theta_i^{\text{inj}}$: the change introduced by the unlearning method.

A precise unlearning method should produce large $|\Delta_{\text{unl}}|$ where $|\Delta_{\text{inj}}|$ is large (in-mask weights), and near-zero $|\Delta_{\text{unl}}|$ elsewhere.

We employ three families of scoring functions, each capturing a different notion of targeted modification:

- **Magnitude-based.** These measure how much each weight changed during unlearning:
  - `raw`: absolute weight change $|\Delta_{\text{unl}}|$;
  - `qtile`: quantile rank of $|\Delta_{\text{unl}}|$ within each parameter tensor (scale-invariant);
  - `layernorm`: $|\Delta_{\text{unl}}|$ normalized by the standard deviation within the same (layer, component) group.
- **Reversal-based.** These leverage the known injection direction to detect whether unlearning *reversed* the injected change:
  - `signrev`: $-(\Delta_{\text{inj}} \cdot \Delta_{\text{unl}})$, positive when unlearning opposes the injection;
  - `reversal`: $(|\Delta_{\text{inj}}| - |\theta^{\text{unl}} - \theta^{\text{pre}}|)/(|\Delta_{\text{inj}}| + \epsilon)$, fractional return toward the pretrained state;
  - `dirreversal`: $-(\Delta_{\text{unl}} \cdot \text{sign}(\Delta_{\text{inj}}))/(|\Delta_{\text{inj}}| + \epsilon)$, normalized directional reversal.
- **Contrast-based.** These compare against an *UnMask control*—the same unlearning method applied to data where no knowledge was injected into the masked weights—isolating changes attributable to the injected knowledge from generic optimization noise. We write $\Delta_{\text{mask}} \equiv \Delta_{\text{unl}}$ for the change on the injected model and Δ_{unmask} for the corresponding change of the same method on the UnMask control:

- contrast: $|\Delta_{\text{mask}}| - |\Delta_{\text{unmask}}|$;
- contrastnorm: symmetric contrast index in $[-1, 1]$;
- compnorm/eratio: $|\Delta_{\text{mask}}|$ normalized by the UnMask baseline change.

Composite score and AUC selection The composite score combines all per-weight features above through a logistic regression. We fit a separate classifier for each (field $\times$ method $\times$ mask) experiment, rather than pooling across methods, since each method leaves a distinct weight-modification signature that a pooled classifier would average away. We use scikit-learn's LogisticRegression with 5-fold cross_val_predict, and take the out-of-fold predicted probabilities as the per-weight scores, so that no parameter is scored by a classifier it helped train. To keep the fit tractable, we subsample up to 2M parameters uniformly from the active attention and feedforward weights (normalization and embedding parameters are excluded). The localization-precision AUC we report for each (field, method) pair is the maximum over all applicable scoring families (the per-feature scores above and the composite). The in-mask ground truth is the forget mask m^F; a unified variant that labels $m^F \vee m^R$ as in-mask is also computed. For OracleGrad, the contrast-based family is omitted from this maximum, as its unmasked control is undefined.

F Additional results

F.1 Impact of masking on memorization

In the main paper, we reported memorization results for the masked training approach, using 5% of the weights. Figure 6 visualizes the extent to which memorization was hindered by that restricted setup. As expected, memorization is much more challenging in the masked training setup, compared to unmasked. However, even in the least memorized PII field (birth city), we obtain enough profiles to build our unlearning targets.

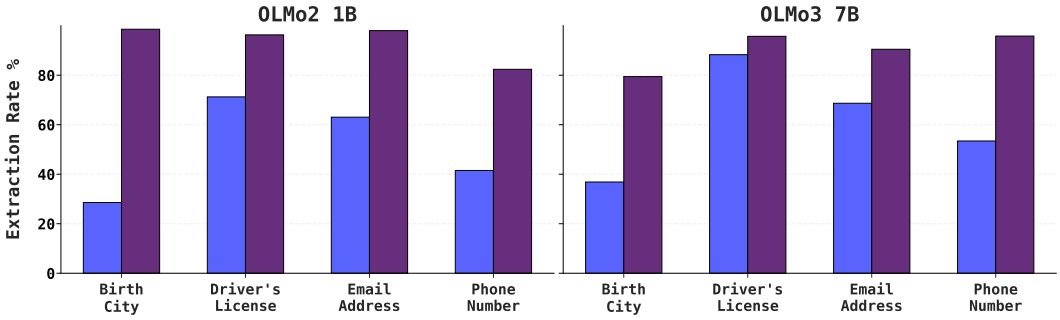


Figure 6: Memorization Comparison between Masked Training and Unmasked Training models, the latter clearly achieves higher extraction rates as it's less constrained.

Choice of mask coverage. The 5% coverage used throughout LACUNA reflects a tradeoff: a smaller mask yields a more precise localization target, but the masked weights must still be able to memorize the injected PII. We found this lower bound to be fairly sharp. At 1% coverage, neither the OLMo2 1B nor the OLMo3 7B model memorized the injected PII to a usable degree, and the same held at 2% coverage for both sizes. Only at 5% did memorization become reliable enough to build adequately sized unlearning targets across all fields. We therefore adopt 5% as the smallest coverage that preserves memorization, and leave a finer characterization of this tradeoff to future work.

F.2 Unlearning results

Before displaying the additional unlearning performance results, we provide a formal definition for the metrics presented in the main paper (Section 4.2).

- Exact Memorization (**EM**)

$$\mathrm{EM} = \frac{1}{|y|} \sum_{k} \mathbf{1} \left\{ \arg\max_{v} f(v \mid [x, y^{<k}]; \boldsymbol{\theta}) = y^{k} \right\}$$

- Extraction Strength (**ES**)

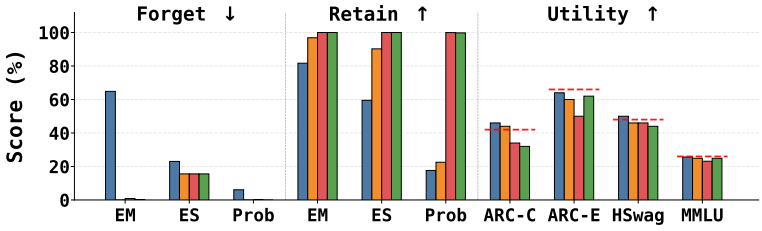
$$\mathrm{ES} = 1 - \frac{1}{|y|} \min_{k} \left\{ k \mid \mathrm{greedy}_{\boldsymbol{\theta}}([x, y^{<k}]) = y^{>k} \right\},$$

  where $\mathrm{greedy}_{\boldsymbol{\theta}}(\cdot)$ denotes greedy decoding under $f(\cdot \mid \cdot\,; \boldsymbol{\theta})$;

- Probability (**Prob**)

$$\mathrm{Prob} = f(y \mid x; \boldsymbol{\theta})$$

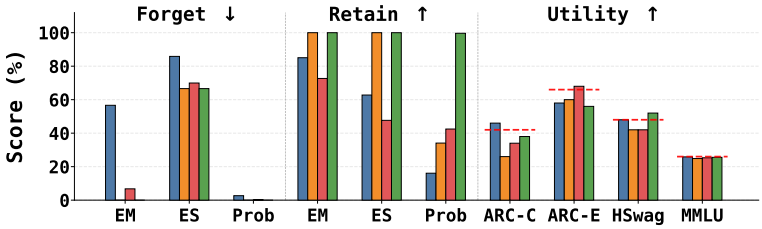
In the main paper (Section 4.3), we only included the unlearning evaluations for `email addresses` for the `OLMo2 1B` model, due to space constraints. Here, Figure 7 and Figure 9 include all remaining results.



(a) Output-level evaluation, `driver's license`

Method	AUC
AlphaEdit	0.500
MemFlex	0.500
SimNPO	0.516
OracleGrad	**0.914**

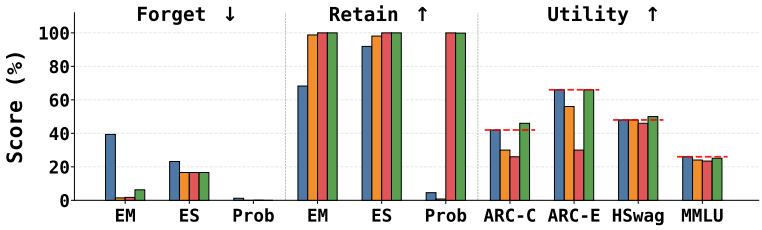
(b) Localization precision, `driver's license`



(c) Output-level evaluation, `birth city`

Method	AUC
AlphaEdit	0.500
MemFlex	0.501
SimNPO	0.516
OracleGrad	**0.913**

(d) Localization precision, `birth city`

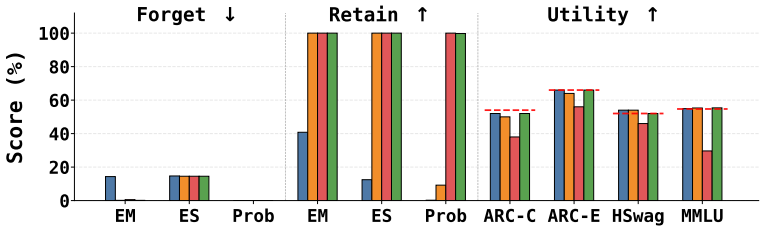


(e) Output-level evaluation, `phone number`

Method	AUC
AlphaEdit	0.500
MemFlex	0.500
SimNPO	0.515
OracleGrad	**0.914**

(f) Localization precision, `phone number`

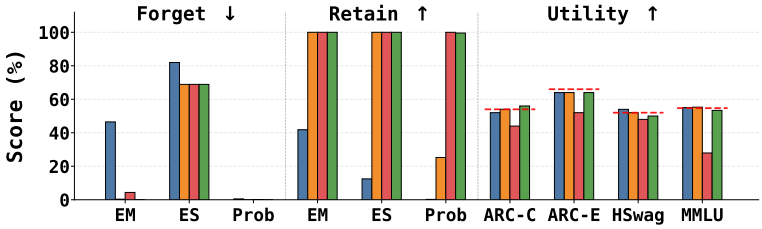
Figure 7: Unlearning Evaluation for `OLMo2 1B`, for three PII fields (see the main paper for `email address`). On the left-hand side, we visualize Forget, Retain, and Utility metrics. The **- - -** on Utility represents the **Pre-Unlearning** results. On the right, we report the evaluation of our proposed **Localization Precision** metric over `AlphaEdit`, `MemFlex`, `SimNPO`, and our Oracle-baseline `OracleGrad`.



(a) Output-level evaluation, `driver's license`

Method	AUC
AlphaEdit	0.500
MemFlex	0.500
SimNPO	0.516
OracleGrad	**0.910**

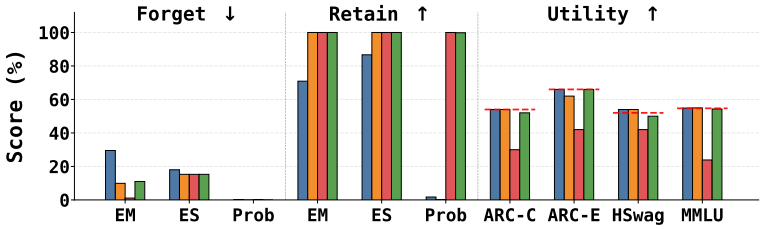
(b) Localization precision, `driver's license`



(c) Output-level evaluation, `birth city`

Method	AUC
AlphaEdit	0.500
MemFlex	0.500
SimNPO	0.513
OracleGrad	**0.911**

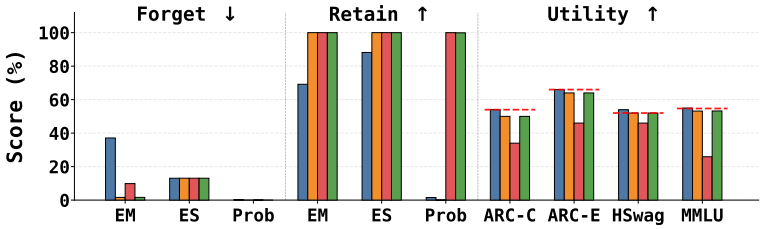
(d) Localization precision, `birth city`



(e) Output-level evaluation, `phone number`

Method	AUC
AlphaEdit	0.500
MemFlex	0.500
SimNPO	0.512
OracleGrad	**0.911**

(f) Localization precision, `phone number`



(g) Output-level evaluation, `email address`

Method	AUC
AlphaEdit	0.500
MemFlex	0.500
SimNPO	0.512
OracleGrad	**0.911**

(h) Localization precision, `email address`

Figure 9: Unlearning evaluation for `OLMo3 7B`, for all four PII fields. On the left-hand side, we plot the Forget, Retain, and Utility metrics. The **- - -** on Utility represents the **Pre-Unlearning** results. On the right, we report the evaluation of our proposed **Localization Precision** metric over `AlphaEdit`, `MemFlex`, `SimNPO`, and our Oracle-baseline `OracleGrad`.

Finally, for completeness, we present the results for all PII fields (as previously included in the figures) in Tables 2 and 3, for the 1B and 7B models, respectively.

Table 2: Unlearning Results - `OLMo2 1B`- Cumulative results for all unlearning methods: **AlphaEdit** (AE), **MemFlex** (MF), **OracleGrad** (OG), **SimNPO** (SN). AUC (F) uses the forget mask m^F as in-mask ground truth, while AUC (F|R) uses the unified label $m^F \vee m^R$; the unified variant is undefined for `OracleGrad` (which targets only m^F), shown as —.

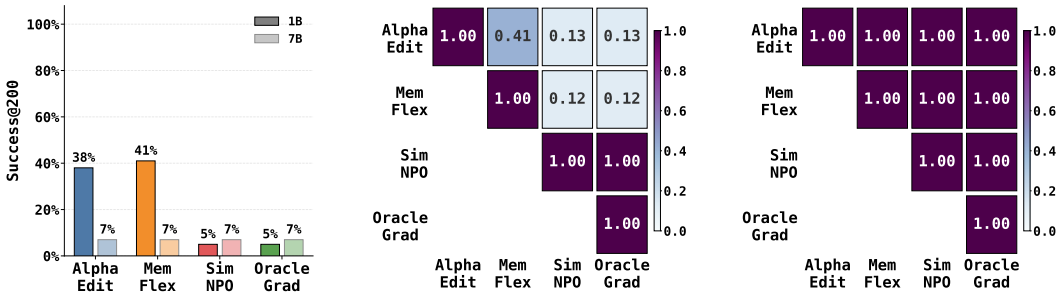
		Email Address				Phone Number				Birth City				Driver's License			
		AE	MF	OG	SN	AE	MF	OG	SN	AE	MF	OG	SN	AE	MF	OG	SN
Forget	EM	63.2	36.8	**1.6**	17.7	39.4	**1.5**	6.3	1.7	56.6	**0.0**	**0.0**	6.8	64.9	**0.0**	0.1	0.9
	ES	34.8	**15.9**	**15.9**	16.7	23.2	**16.6**	**16.6**	**16.6**	85.8	**66.6**	**66.6**	70.0	23.0	**15.6**	**15.6**	**15.6**
	EM Paraph.	63.6	36.7	**1.6**	18.0	39.2	**1.5**	6.3	1.5	55.4	**0.0**	**0.0**	5.8	64.1	**0.0**	0.1	0.5
	ES Paraph.	34.1	**15.9**	**15.9**	16.6	23.1	**16.6**	**16.6**	**16.6**	85.9	**66.6**	**66.6**	70.0	23.9	**15.6**	**15.6**	**15.6**
	Prob	10.3	0.0	**0.0**	0.9	1.2	**0.0**	**0.0**	0.0	2.7	**0.0**	**0.0**	0.2	6.1	**0.0**	**0.0**	0.0
	Prob Paraph.	10.6	0.0	**0.0**	0.9	1.3	**0.0**	**0.0**	0.0	2.6	**0.0**	**0.0**	0.2	6.2	**0.0**	**0.0**	0.0
Retain	EM	58.8	98.0	**100.0**	**100.0**	68.3	98.8	**100.0**	**100.0**	85.1	**100.0**	**100.0**	72.7	81.7	96.9	**100.0**	**100.0**
	ES	89.8	96.8	**100.0**	**100.0**	91.9	98.1	**100.0**	**100.0**	62.8	**100.0**	**100.0**	47.7	59.5	90.2	**100.0**	**100.0**
	EM Paraph.	60.2	96.3	**100.0**	93.7	67.0	98.6	**100.0**	**100.0**	85.7	99.7	**100.0**	73.1	82.1	93.3	**99.6**	98.2
	ES Paraph.	90.8	95.8	**100.0**	95.7	92.7	97.6	**100.0**	**100.0**	65.5	98.3	**100.0**	49.3	59.8	80.3	**98.4**	91.9
	Prob	4.5	1.7	99.9	**100.0**	4.6	0.8	99.8	**100.0**	16.1	34.1	**99.7**	42.5	17.6	22.5	99.8	**100.0**
	Prob Paraph.	4.5	1.7	**99.7**	91.3	4.4	0.8	**99.7**	99.6	16.7	34.0	**98.7**	43.4	18.0	22.6	**98.5**	95.8
Utility (Δ)	ARC-C	**+4.0**	-4.0	-2.0	-12.0	+0.0	-12.0	**+4.0**	-16.0	**+4.0**	-16.0	-4.0	-8.0	**+4.0**	+2.0	-10.0	-8.0
	ARC-E	+0.0	-6.0	**+2.0**	-18.0	**+0.0**	-10.0	**+0.0**	-36.0	-8.0	-6.0	-10.0	**+2.0**	**-2.0**	-6.0	-4.0	-16.0
	HSwag	**+0.0**	**+0.0**	-4.0	-4.0	+0.0	+0.0	**+2.0**	-2.0	+0.0	-6.0	**+4.0**	-6.0	**+2.0**	-2.0	-4.0	-2.0
	MMLU	**+0.1**	-1.5	-1.2	-1.8	**+0.0**	-1.9	-0.9	-2.6	**-0.2**	-1.2	-0.4	-0.7	**-0.4**	-1.1	-1.1	-2.9
Precision	AUC (F—R)	0.500	0.500	—	**0.519**	0.500	0.500	—	**0.520**	0.500	0.501	—	**0.522**	0.500	0.500	—	**0.522**
	AUC (F)	0.500	0.500	**0.915**	0.515	0.500	0.500	**0.914**	0.515	0.500	0.501	**0.913**	0.516	0.500	0.500	**0.914**	0.516

Table 3: Unlearning Results - OLMo3 7B- Cumulative results for all unlearning methods: **AlphaEdit** (AE), **MemFlex** (MF), **OracleGrad** (OG), **SimNPO** (SN). AUC (F) uses the forget mask m^F as in-mask ground truth, while AUC (F|R) uses the unified label $m^F \vee m^R$; the unified variant is undefined for OracleGrad (which targets only m^F), shown as —.

		Email				Phone				Birth City				Driver's Lic.			
		AE	MF	OG	SN	AE	MF	OG	SN	AE	MF	OG	SN	AE	MF	OG	SN
Forget	EM	37.1	**1.6**	**1.6**	9.8	29.5	9.9	11.0	**1.1**	46.5	0.3	**0.0**	4.4	14.3	**0.1**	**0.1**	0.6
	ES	**13.1**	**13.1**	**13.1**	**13.1**	18.0	**15.3**	**15.3**	**15.3**	82.0	**68.9**	**68.9**	**68.9**	14.7	**14.5**	**14.5**	**14.5**
	EM Paraph.	36.3	**1.6**	**1.6**	10.3	29.7	9.9	11.0	**1.1**	47.1	**0.0**	**0.0**	6.5	14.6	**0.1**	**0.1**	0.7
	ES Paraph.	**13.1**	**13.1**	**13.1**	**13.1**	17.9	**15.3**	**15.3**	**15.3**	81.2	**68.9**	**68.9**	**68.9**	14.9	**14.5**	**14.5**	**14.5**
	Prob	0.2	**0.0**	**0.0**	0.0	0.2	**0.0**	**0.0**	0.1	0.5	**0.0**	**0.0**	0.0	0.0	**0.0**	**0.0**	0.0
	Prob Paraph.	0.2	**0.0**	**0.0**	0.0	0.2	**0.0**	**0.0**	0.1	0.7	**0.0**	**0.0**	0.1	0.0	**0.0**	**0.0**	0.0
Retain	EM	69.2	**100.0**	**100.0**	**100.0**	70.9	**100.0**	**100.0**	**100.0**	41.8	**100.0**	**100.0**	**100.0**	40.8	**100.0**	**100.0**	**100.0**
	ES	88.2	**100.0**	**100.0**	**100.0**	86.7	**100.0**	**100.0**	**100.0**	12.5	**100.0**	**100.0**	**100.0**	12.5	**100.0**	**100.0**	**100.0**
	EM Paraph.	67.6	**100.0**	**100.0**	99.2	69.3	**100.0**	**100.0**	99.0	41.1	**99.5**	99.4	96.5	40.6	99.4	**99.8**	96.0
	ES Paraph.	87.6	**100.0**	**100.0**	99.2	87.6	**100.0**	**100.0**	99.0	12.5	**97.3**	97.2	81.3	12.5	96.9	**99.4**	82.1
	Prob	1.5	0.0	99.9	**100.0**	1.7	0.0	99.8	**100.0**	0.2	25.2	99.6	**100.0**	0.2	9.2	99.8	**100.0**
	Prob Paraph.	1.7	0.0	**99.7**	99.0	1.8	0.0	**99.7**	99.0	0.2	24.2	**96.2**	89.6	0.2	8.7	**98.9**	89.1
Utility (Δ)	ARC-C	**+0.0**	-4.0	-4.0	-20.0	**+0.0**	**+0.0**	-2.0	-24.0	-2.0	+0.0	**+2.0**	-10.0	**-2.0**	-4.0	**-2.0**	-16.0
	ARC-E	**+0.0**	-2.0	-2.0	-20.0	**+0.0**	-4.0	**+0.0**	-24.0	**-2.0**	**-2.0**	**-2.0**	-14.0	**+0.0**	-2.0	**+0.0**	-10.0
	HSwag	**+2.0**	+0.0	+0.0	-6.0	**+2.0**	**+2.0**	-2.0	-10.0	**+2.0**	+0.0	-2.0	-4.0	**+2.0**	**+2.0**	+0.0	-6.0
	MMLU	**+0.2**	-1.5	-1.5	-28.8	+0.1	**+0.2**	-0.4	-30.8	+0.4	**+0.6**	-1.3	-26.8	+0.2	+0.6	**+0.7**	-25.1
Precision	AUC (F—R)	0.500	0.500	—	**0.515**	0.500	0.500	—	**0.514**	0.500	0.500	—	**0.516**	0.500	0.500	—	**0.520**
	AUC (F)	0.500	0.500	**0.911**	0.512	0.500	0.500	**0.911**	0.512	0.500	0.500	**0.911**	0.513	0.500	0.500	**0.910**	0.516

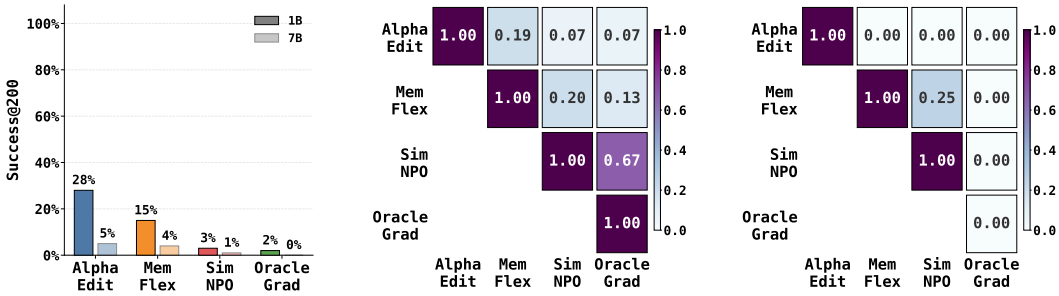
F.3 Resurfacing results

We report here the additional results for the relearning attacks we performed on the models. We note that SimNPO is, among the compared methods, the strongest (together with our strong baseline OracleGrad). Additionally, we observe that leaked profiles by OracleGrad are almost exactly overlapped with the ones leaked by SimNPO, suggesting that these samples might simply be more challenging to forget in the first place, and hence easier to recover. Across fields, we also observe that OLMo3 7B is consistently much less prone to resurfacing than OLMo2 1B. We refrain from drawing strong conclusions from this, as it remains unclear whether it reflects a genuine property of the larger model or instead a limitation of our straightforward resurfacing attack at the 7B scale.



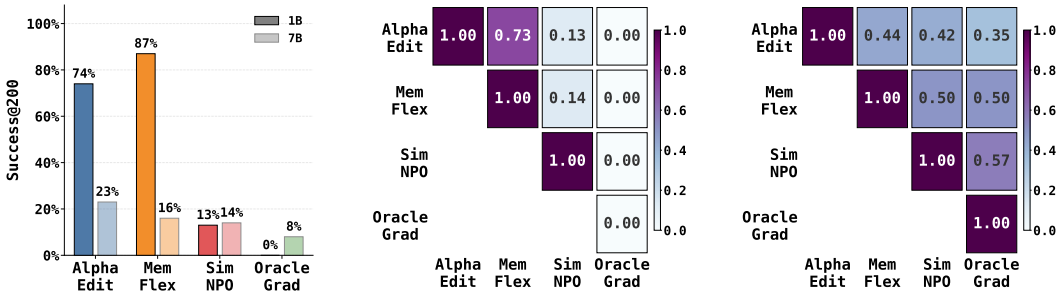
(a) Relearning leakage rate. (b) Leaked profiles Jaccard similarity - OLMo2 1B. (c) Leaked profiles Jaccard similarity - OLMo3 7B.

Figure 10: Relearning vulnerability for `Driver's License`. The fact that SimNPO and OracleGrad leak the same profile's information (Jaccard index of 1.00) suggests that these profiles might be simply particularly challenging to forget.



(a) Relearning leakage rate. (b) Leaked profiles Jaccard similarity - OLMo2 1B. (c) Leaked profiles Jaccard similarity - OLMo3 7B.

Figure 11: Relearning vulnerability for `Birth City`.



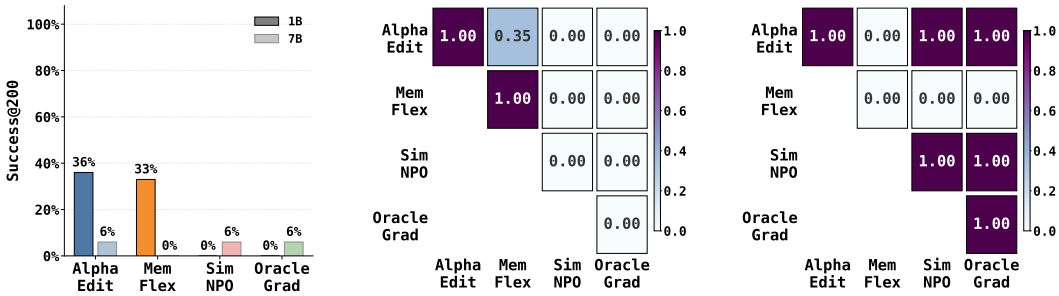
(a) Relearning leakage rate. (b) Leaked profiles Jaccard similarity - OLMo2 1B. (c) Leaked profiles Jaccard similarity - OLMo3 7B.

Figure 13: Relearning vulnerability for `Email Address`.

G Hyperparameters - Tuning

G.1 Training & Instruction Tuning

Along with the discussion in Section 3, we report here some additional details on the specific injection setup and instruction tuning.



(a) Relearning leakage rate.

(b) Leaked profiles Jaccard similarity - OLMo2 1B.

(c) Leaked profiles Jaccard similarity - OLMo3 7B.

Figure 12: Relearning vulnerability for Phone Number.

Table 4: Training & Instruction Tuning Hyperparameters

Parameter	OLMo2 1B	OLMo3 7B
Base model	allenai/OLMo-2-0425-1B	allenai/OLMo-3-1025-7B
Checkpoint revision	step1907359-tokens4001B	step999000
Instruction Tuning LoRA rank (r)	16	16
Instruction Tuning LoRA alpha (α)	8	8
Instruction Tuning LoRA target layers	[14, 15]	[30, 31]
Instruction Tuning Early stopping	Best eval loss	Best eval loss

G.2 Unlearning

Unlearning hyperparameters were selected via grid search on a validation split (Driver's License forget / Email Address retain, cross-field), carried out independently for the OLMo2 1B and OLMo3 7B models. The tuned hyperparameters for each method and model size are listed in Table 5.

MemFlex. The primary tuning knobs were the forget/retain loss factors and the number of training epochs.

AlphaEdit. Extensive tuning was performed across edit magnitude (clamp norm, null-space threshold), optimization (gradient steps), covariance data source, target layers, and regularization. A key finding was that AlphaEdit cannot selectively target forget vs. retain data when both share the same QA format, as the activation subspaces overlap significantly.

SimNPO. The tuning for this method was relatively straightforward as it relies on more traditional gradient-based techniques.

OracleGrad. OracleGrad was tuned by comparing GradAscent (forget-only, no retain loss) against GradDiff (gradient ascent on forget + descent on retain). Gradient Difference (with the per-size forget/retain loss weights in Table 5) provides high retain stability, while still allowing strong unlearning effectiveness.

Table 5: Unlearning – tuned hyperparameters by method, selected independently for each model size. Values that differ between OLMo2 1B and **OLMo3 7B** are in **bold**.

Method	Hyperparameter	OLMo2 1B	**OLMo3 7B**
MemFlex	Forget factor	-0.6	-0.6
	Retain factor	2.0	2.0
	Learning rate	$\mathbf{3 \times 10^{-4}}$	$\mathbf{1 \times 10^{-4}}$
	Gradient threshold	$\mathbf{6 \times 10^{-4}}$	$\mathbf{1 \times 10^{-5}}$
	Similarity threshold	0.92	0.92
	Epochs	20	20
SimNPO	γ (forget weight)	3.0	3.0
	α (retain weight)	**0.01**	**0.5**
	β (sharpness)	10.0	10.0
	δ (margin)	1.5	1.5
	Learning rate	1×10^{-4}	1×10^{-4}
	Epochs	200	200
	Retain loss / scheduler	NLL / Constant	NLL / Constant
OracleGrad	Method	GradDiff	GradDiff
	γ (forget weight)	**5.0**	**1.0**
	α (retain weight)	**1.0**	**0.5**
	Learning rate	$\mathbf{1 \times 10^{-4}}$	$\mathbf{5 \times 10^{-5}}$
	Epochs	200	200
AlphaEdit	Clamp norm factor	0.5	0.5
	Null-space threshold	1×10^{-3}	1×10^{-3}
	v gradient steps	50	50
	v learning rate	5×10^{-2}	5×10^{-2}
	v loss layer	**15**	**31**
	v weight decay	**1.0**	**0.5**
	KL factor	**1.0**	**0.0625**
	L2 regularization	1.0	1.0
	MOM2 dataset	**wikipedia + retain**	**wikipedia**
	Target layers	[4, 5, 6, 7, 8]	[4, 5, 6, 7, 8]
	Batch size	**5**	**1**