FORGET WHAT COMES NEXT: LLM UNLEARNING VIA STATISTICAL INDEPENDENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) can memorize and reproduce fragments of their training data, raising privacy and legal concerns that motivate machine unlearning. Yet poisoned training examples can remain statistically detectable after unlearning, highlighting the need for a deeper theoretical understanding of how unlearning objectives relate to memorization. In this work, we introduce PINT, a novel algorithm for machine unlearning that uses the Hilbert-Schmidt Independence Criterion (HSIC) to reduce the statistical dependence between tokens to be forgotten and the internal representations used to predict them. We formally relate our HSIC-based objective to Exact Memorization through a computable upper bound, connecting the training objective directly to the model's ability to reproduce forgotten tokens. In extensive experiments, we compare forgetting-utility trade-offs across nine settings with models up to 7B parameters and demonstrate scalability to 70B parameter models. Our evaluations cover unlearning personal information, hazardous knowledge in biology and cybersecurity, copyrighted books and news articles, and Gaussian poisons. PINT achieves the strongest forgetting-utility trade-offs on most benchmarks. On Gaussian poisons, it is the only method to make the poison undetectable while maintaining near-original model utility.

Anonymized Codebase

1 INTRODUCTION

The tendency of large language models (LLMs) to memorize and reproduce fragments of their training data, including personally identifiable information and copyrighted text (Carlini et al., 2021; 2022), poses significant privacy and legal concerns. With the *right to be forgotten* now formalized in major data protection frameworks across the EU, the US, and China (European Parliament and Council of the European Union, 2016; California State Legislature, 2018; Standing Committee of the National People's Congress, 2021), there is a growing need for methods that can remove the influence of specific training examples without the prohibitive cost of retraining from scratch. A growing body of work on approximate *machine unlearning* has made empirical progress in this direction (Jang et al., 2023; Maini et al., 2024; Li et al., 2024; Zhang et al., 2024). Nevertheless, significant gaps remain: Pawelczyk et al. (2025) show that current methods may fail to unlearn poisoned examples injected during training, leaving them statistically detectable even after an unlearning attempt.

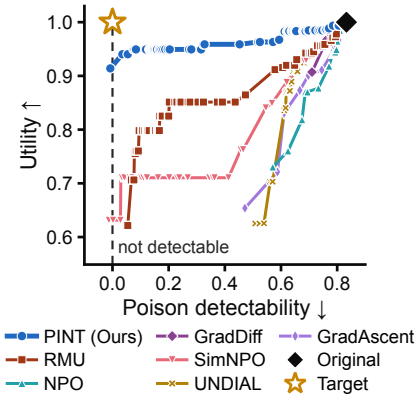

Figure 1: Our method PINT makes the poison undetectable while retaining the highest utility among competitors (91% vs. at most 63% for baselines).

To understand how an LLM can forget a memorized sequence, we look at how it predicts each token from the preceding context. At each position in a sequence, an autoregressive LLM encodes the preceding context into a prefix representation from which the output head predicts the next token. Reproducing a memorized sequence requires that these prefix representations carry information

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.

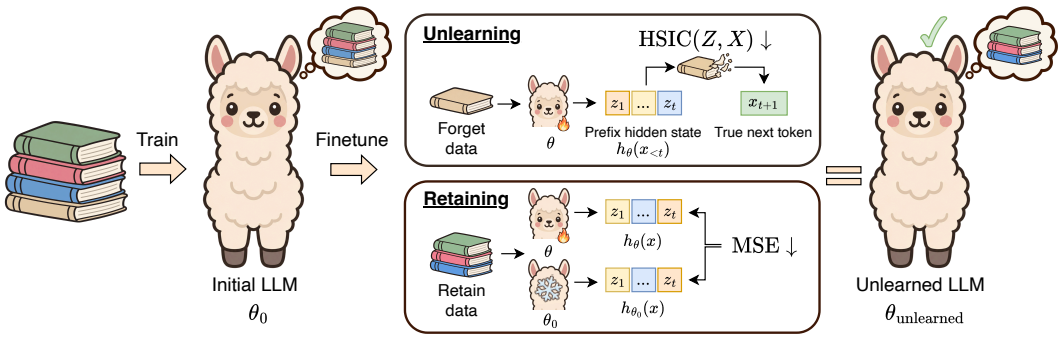

Figure 2: **Overview of PINT.** On the *forget set*, PINT minimizes the HSIC between prefix representations and next tokens, reducing their statistical dependence. On the *retain set*, it constrains prefix representations to stay close to those of the original model, preserving general model utility.

about the tokens that follow. This relationship can be expressed in terms of statistical dependence. So, can we induce forgetting by minimizing this statistical dependence?

To answer this, we propose PINT (**P**refix-representation **I**ndependence of the **N**ext **T**oken). On the data targeted for removal (the *forget set*), PINT minimizes the Hilbert-Schmidt Independence Criterion (HSIC) (Gretton et al., 2005) between prefix representations and their corresponding next tokens, capturing and penalizing complex, non-linear dependencies. To preserve utility, PINT constrains representations on a *retain set* to remain close to those of the original model (see Figure 2 for an overview). As illustrated in Figure 1, this approach is highly effective: PINT is the only method that makes Gaussian poisons statistically undetectable (Pawelczyk et al., 2025) while retaining near-full model utility. We extend this analysis across a variety of unlearning benchmarks and model sizes, finding that PINT achieves the strongest forgetting-utility trade-offs on most benchmarks.

In addition to our empirical evidence, we formally connect the unlearning objective to memorization by proving that a model's Exact Memorization (Tirumala et al., 2022) admits an HSIC-based upper bound. We measure the bound on LLMs with up to 7B parameters (Figure 3). To the best of our knowledge, this is the first work to use HSIC between prefix representations and next tokens as an LLM unlearning objective.

Our main contributions are as follows:

- We introduce **PINT, an LLM unlearning method** that minimizes HSIC between prefix representations and next tokens on the *forget set*, while constraining *retain set* representations to remain close to those of the original model.
- We derive an **HSIC-based upper bound on Exact Memorization**, formally connecting the PINT objective to how much of the forgotten text the model can still reproduce.
- We evaluate PINT across **nine unlearning settings and models up to 7B parameters**, covering personal information, hazardous knowledge, copyrighted text, and Gaussian poisons. PINT achieves the strongest forgetting-utility trade-offs on most benchmarks and is the only method to make Gaussian poisons statistically undetectable while retaining near-original model utility.

2 Related Work

LLM unlearning. One approach to approximate LLM unlearning is to reduce the likelihood assigned to *forget set* responses. Gradient ascent does this directly by minimizing their log-likelihood on the *forget set*, while gradient difference (Liu et al., 2022) additionally maximizes the likelihood of *retain set* responses to preserve utility. Negative Preference Optimization (NPO) (Zhang et al., 2024) reduces *forget set* likelihood relative to the original model. Simple NPO (SimNPO) (Fan et al., 2026) removes the dependence on the original model, using length-normalized log-likelihoods instead. Rather than suppressing likelihood directly, Unlearning via Self-Distillation on Adjusted Logits (UNDIAL) (Dong et al., 2025) constructs target distributions by lowering the logits from *forget set* tokens in the original model's predictions, then trains the updated model to match these

modified distributions. Other methods operate on the model's representations. Representation Misdirection for Unlearning (RMU) (Li et al., 2024) collapses *forget set* representations at a chosen layer toward a fixed random vector while keeping *retain set* representations close to those of the original model. PINT instead penalizes the statistical dependence between prefix representations and next tokens at the final layer, avoiding both the choice of an intervention layer and the need for representation collapse.

Evaluating unlearning. Effective unlearning requires evidence of forgetting alongside preserved utility, but comparison across methods is challenging. Results are sensitive to hyperparameters, and exploring these choices requires costly repeated runs (Wang et al., 2025). Reporting only selected checkpoints therefore gives an incomplete picture of the forgetting-utility trade-off, while model-selection criteria can also change method rankings (Dorna et al., 2026). Moreover, no single metric captures all aspects of forgetting, making evaluation across diverse tasks and metrics necessary (Shi et al., 2025; Pawelczyk et al., 2025). The TOFU benchmark (Maini et al., 2024) evaluates forgetting of information about fictitious authors learned through question-answer pairs, while MUSE (Shi et al., 2025) tests whether models still answer factual questions about books and news articles targeted for removal. WMDP (Li et al., 2024) uses multiple-choice accuracy as a proxy for hazardous knowledge. These benchmarks also assess utility on retained information or general capabilities. Pawelczyk et al. (2025) introduced a Gaussian-poison statistical test that detects whether injected noise remains correlated with the model's input gradients after unlearning, using the known distribution of the noise under independence. Beyond empirical evaluation, formal guarantees remain elusive in large networks (Lizzo & Heck, 2026).

Independence criteria and HSIC. The Hilbert-Schmidt Independence Criterion (HSIC) (Gretton et al., 2005) measures how strongly two random variables depend on each other. Unlike correlation, it can capture complex, non-linear dependence: with characteristic kernels, such as a Gaussian kernel on representations and a delta kernel on tokens, HSIC is zero exactly when the two variables are independent. It can be estimated from a batch of samples and is differentiable, so it can be used as a training loss. HSIC has been applied to feature selection (Yamada et al., 2014), learning fair and robust representations (Quadrianto et al., 2019; Miklautz et al., 2026), layer selection for lifelong knowledge editing (Eskandar et al., 2026), and self-supervised learning (Li et al., 2021). In machine unlearning, HSIC has been used as an extra term in the model update that helps image and graph classifiers keep their accuracy after data is removed (Han et al., 2024), and to evaluate whether an image model treats the *forget set* like data it has never seen (Zhang et al., 2026). To the best of our knowledge, PINT is the first method to use HSIC as a training objective for LLM unlearning.

3 Methodology

This section presents PINT. Section 3.1 defines the unlearning setting and notation, and Section 3.2 introduces the forget and retain objectives. Section 3.3 connects the forget objective to Exact Memorization (Tirumala et al., 2022), which measures how much forgotten text the model can still reproduce, and Section 3.4 relates it to RMU (Li et al., 2024), a widely used unlearning method.

3.1 Problem Setup

Following the standard LLM unlearning setup (Maini et al., 2024), we consider a language model over a finite vocabulary $\mathcal{V}$, where $\mathcal{V}^*$ denotes the set of finite token sequences. Let $\theta_0 \in \Theta \subseteq \mathbb{R}^p$ denote the original model parameters. Given a *forget set* $\mathcal{D}_f \subseteq \mathcal{V}^*$ containing information targeted for removal and a *retain set* $\mathcal{D}_r \subseteq \mathcal{V}^*$ representing behavior to preserve, the goal is to update θ_0 to $\theta \in \Theta$ so that the model forgets the targeted information while preserving its behavior on $\mathcal{D}_r$. For a token sequence $\mathbf{x} = (x_1, \ldots, x_n) \in \mathcal{V}^*$, let $\mathbf{x}_{<t} = (x_1, \ldots, x_{t-1})$ denote the prefix preceding x_t. The model's representation function $h_\theta : \mathcal{V}^* \to \mathbb{R}^d$ maps this prefix to its final-layer representation $z_t^\theta = h_\theta(\mathbf{x}_{<t})$, which the output head uses to predict x_t. For each forget sequence $\mathbf{x} \in \mathcal{D}_f$, let $\mathcal{T}(\mathbf{x}) \subseteq \{2, \ldots, n\}$ denote its *forget positions*, the positions whose tokens are to be unlearned. Table 3 in Appendix A.2 summarizes the notation.

3.2 Training Objective

PINT updates the model using two complementary objectives. The forget objective uses HSIC to penalize statistical dependence between prefix representations and their corresponding next tokens on the *forget set*. The retain objective leverages insights from RMU (Li et al., 2024) and keeps *retain set* representations close to those of the original model.

Forget objective. At each training step, we draw a minibatch $\mathcal{B}_f \subseteq \mathcal{D}_f$ of forget sequences. Let $m = \sum_{\mathbf{x}\in\mathcal{B}_f} |\mathcal{T}(\mathbf{x})|$ be the total number of forget positions in $\mathcal{B}_f$. Indexing them by $i = 1, \ldots, m$, we write z_i and x_i for the prefix representation and the next token at the i-th forget position, and stack them into $\mathbf{Z} = (z_1, \ldots, z_m)^\top \in \mathbb{R}^{m\times d}$ and $\mathbf{X} = (x_1, \ldots, x_m) \in \mathcal{V}^m$. The forget objective is the empirical HSIC (Gretton et al., 2005),

$$\mathcal{L}_{\text{forget}}(\mathcal{B}_f; \theta) = \widehat{\text{HSIC}}(\mathbf{Z}, \mathbf{X}) = \frac{1}{m^2} \operatorname{tr}(KHLH), \tag{1}$$

where $K_{ij} = k(z_i, z_j)$ and $L_{ij} = \ell(x_i, x_j)$ are the Gram matrices of kernels k on representations and ℓ on tokens, and $H = I_m - \frac{1}{m}\mathbf{1}_m\mathbf{1}_m^\top$ centers them. Because HSIC measures statistical dependence (Appendix A.1), minimizing Equation 1 reduces how much prefix representations reveal about their next tokens, which limits how much of the forgotten text the model can reproduce (Section 3.3).

Retain objective. Minimizing the forget objective changes the model, and we want these changes to stay confined to the forget set. The retain objective therefore keeps the prefix representations of retain sequences close to those of the original model θ_0, in the same form as the retain objective of RMU (Li et al., 2024) but at the final layer. For a minibatch $\mathcal{B}_r \subseteq \mathcal{D}_r$ of retain sequences, with $\mathcal{T}_r(\mathbf{x})$ the positions of $\mathbf{x}$ included in the objective, it is the mean squared difference

$$\mathcal{L}_{\text{retain}}(\mathcal{B}_r; \theta) = \frac{1}{|\mathcal{B}_r|} \sum_{\mathbf{x}\in\mathcal{B}_r} \frac{1}{|\mathcal{T}_r(\mathbf{x})|} \sum_{t\in\mathcal{T}_r(\mathbf{x})} \frac{1}{d} \left\| z_t^\theta - z_t^{\theta_0} \right\|_2^2. \tag{2}$$

The retain objective is zero at the original model and grows as the representations of retain sequences drift from their original values.

Total objective. Both objectives act on the same final-layer representations, the forget objective on forget sequences and the retain objective on retain sequences. PINT minimizes their weighted sum,

$$\mathcal{L}_{\text{PINT}}(\mathcal{B}_f, \mathcal{B}_r; \theta) = \lambda\, \mathcal{L}_{\text{forget}}(\mathcal{B}_f; \theta) + \mathcal{L}_{\text{retain}}(\mathcal{B}_r; \theta), \tag{3}$$

where $\lambda \geq 0$ sets how strongly the model forgets. With $\lambda = 0$, only the retain objective remains, which the original model already minimizes. Larger values of λ push the model further toward forgetting, at the cost of larger changes to the representations of retain sequences (Algorithm 1).

3.3 Theoretical Results

The forget objective constrains the model's internal representations, but what we ultimately care about is whether the model can still reproduce the forgotten text. We connect the two through Exact Memorization (Tirumala et al., 2022), which measures how often the model's most likely next token is the true one. Given a prefix representation z, the output head assigns a probability to every token, and we write $g_\theta(z) \in \mathcal{V}$ for the most likely one. A model that has memorized a sequence predicts most of its tokens correctly from the preceding context, and unlearning should lower this accuracy on the forget set.

Definition 3.1 (Exact Memorization)**.** Draw a position uniformly from the forget positions of all forget sequences combined, and let $Z = z_t^\theta$ and $X = x_t$ be its prefix representation and next token. The Exact Memorization of θ is

$$\text{EM}(\theta) = \mathbb{P}\big(g_\theta(Z) = X\big), \tag{4}$$

the fraction of forget positions at which the model predicts the true next token.

Exact Memorization alone overstates what a model has memorized. A model that always predicts a frequent token such as "the" is correct wherever "the" is the next token, without having memorized anything. We therefore compare $\text{EM}(\theta)$ with the baseline

$$\text{EM}_\perp(\theta) = \sum_{v\in\mathcal{V}} \mathbb{P}\big(g_\theta(Z) = v\big)\, \mathbb{P}(X = v), \tag{5}$$

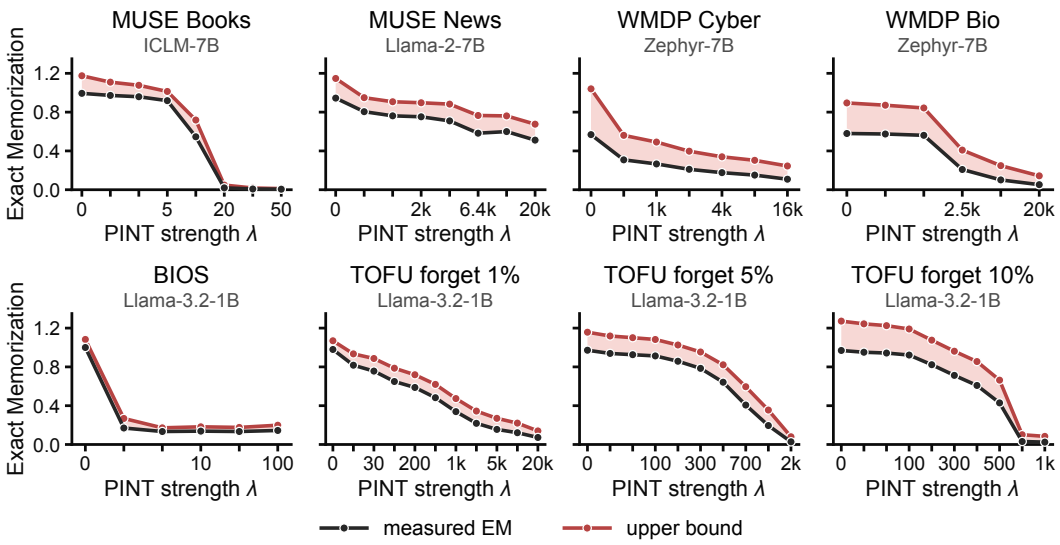

Figure 3: **A single hyperparameter controls Exact Memorization and its upper bound.** Increasing only the forgetting strength lowers Exact Memorization on the *forget set* (black) and the rearranged bound of Theorem 3.2 (red). Appendix E.2 shows how HSIC and C_θ contribute.

the fraction of forget positions at which the prediction matches a next token drawn at random from the *forget set*. This baseline is at most the frequency of the most common next token, so it is small unless a few tokens dominate. If prefix representations carried no information about their next tokens, the prediction would match the true next token exactly as often as a random one, and $\mathrm{EM}(\theta)$ would equal $\mathrm{EM}_\perp(\theta)$. The gap $\mathrm{EM}(\theta) - \mathrm{EM}_\perp(\theta)$ is therefore the part of Exact Memorization that requires prefix representations to carry information about their next tokens. $\mathrm{HSIC}(Z, X)$ measures how strongly prefix representations and their next tokens depend on each other. The forget objective Equation 1 reduces this dependence, and Theorem 3.2 bounds the gap in terms of HSIC.

Theorem 3.2 (HSIC bound on Exact Memorization)**.** *Let* $\mathrm{HSIC}(Z, X)$ *use a strictly positive definite kernel* k *on prefix representations, such as the Gaussian, Laplacian, or Matérn kernel (Sriperumbudur et al., 2011; Wendland, 2005), and the delta kernel* $\ell(v, v') = \mathbf{1}\{v = v'\}$ *on next tokens. Then for every* $\theta \in \Theta$*,*

$$\mathrm{EM}(\theta) - \mathrm{EM}_\perp(\theta) \le C_\theta \sqrt{\mathrm{HSIC}(Z, X)}, \tag{6}$$

where C_θ *is defined in Equation 13.*

The theorem explains why minimizing HSIC reduces memorization. It bounds the memorization gap by HSIC, the dependence that our forget objective reduces. The factor C_θ measures how smoothly the model's predictions change between nearby prefix representations. Lowering HSIC therefore lowers the bound, unless the predictions become less smooth at the same time. Appendix A.3 gives the proof. In our experiments, C_θ never cancels the decrease of HSIC (Appendix E.2). The bound also decreases together with Exact Memorization as the forgetting strength grows (Figure 3).

3.4 Connection to RMU

RMU (Li et al., 2024) is one of the most widely used unlearning methods. Like PINT, it acts on the model's representations rather than directly on its output probabilities. The two methods differ in their forget objective. PINT reduces the dependence between representations and next tokens, which by Theorem 3.2 suffices to remove Exact Memorization beyond the baseline $\mathrm{EM}_\perp$, as long as C_θ stays bounded. RMU instead moves all *forget set* representations towards a fixed random target. For a random unit vector $u \in \mathbb{R}^d$ and a scaling constant $c \ge 0$, its forget objective on a minibatch with representations $z_1, \dots, z_m \in \mathbb{R}^d$ is

$$\mathcal{L}^{\mathrm{RMU}}_{\mathrm{forget}}(\mathcal{B}_f; \theta) = \frac{1}{md} \sum_{j=1}^{m} \|z_j - cu\|_2^2 = \frac{1}{d}\|\bar{z} - cu\|_2^2 + \frac{1}{md} \sum_{j=1}^{m} \|z_j - \bar{z}\|_2^2, \quad \bar{z} = \tfrac{1}{m} \sum_j z_j. \tag{7}$$

Table 1: **Benchmarks, models, and forgetting targets.** The last column shows how the forgetting target is obtained. For benchmarks marked with ★, the target is measured on a model retrained without the *forget set*. For benchmarks marked with ☆, the target is known in advance: on WMDP and bioS, the model should answer no better than random guessing (25% and 12.5%, respectively), and on the Gaussian poisons, detectability should be 0. We use both markers throughout the figures.

Benchmark	Model	Forget set	Target
TOFU (Maini et al., 2024)	Llama-3.2-1B	1, 5, 10% of fictitious authors	★
MUSE-News (Shi et al., 2025)	Llama-2-7B	BBC news articles	★
MUSE-Books (Shi et al., 2025)	ICLM-7B	Harry Potter books	★
WMDP-Bio (Li et al., 2024)	Zephyr-7B	hazardous bio-security facts	☆
WMDP-Cyber (Li et al., 2024)	Zephyr-7B	hazardous cybersecurity facts	☆
bioS (Allen-Zhu & Li, 2023)	Llama-3.2-1B	occupations of fictitious people	☆
Poisons (Pawelczyk et al., 2025)	OLMo-2 2.7B	Gaussian poison	☆

The second term collapses all representations, and the first term moves their mean to the target cu.

Proposition 3.3 (RMU upper bounds the PINT forget objective)**.** *Let the representation kernel k have an L_k-Lipschitz feature map (Appendix A.1), and let $\hat{p}_{\max}$ be the frequency of the most common token in the forget minibatch. Then, for every $u \in \mathbb{R}^d$ and $c \geq 0$,*

$$\widehat{\mathrm{HSIC}}(\mathbf{Z}, \mathbf{X}) \leq L_k^2 \, d \, \hat{p}_{\max} \, \mathcal{L}_{\text{forget}}^{\text{RMU}}(\mathcal{B}_f; \theta). \tag{8}$$

The proposition, proved in Appendix A.4, shows that RMU removes the same dependence as PINT, but indirectly: whenever RMU's objective is small, PINT's objective is small as well. Intuitively, representations that have all been moved to the same point can no longer carry information about which token comes next. The converse does not hold, since PINT's objective can be small without all representations being moved to the target cu.

What the bound implies. Every model that satisfies RMU's forget objective on a set of representations also satisfies PINT's (Proposition 3.3), but not the other way around. PINT's forget objective therefore leaves the retain objective more freedom to preserve the rest of the model. We expect this difference to matter most when the *forget set* and *retain set* are closely related. To test this, we design a controlled experiment, bioS (Section 4), in which the model learns two facts about each fictitious person, their occupation and their university major, and must then forget the occupation while keeping the major. Since both facts concern the same people, the *forget set* and *retain set* are closely related, which allows us to test whether a method forgets one fact without losing the other. In this setting, both methods forget the occupations, but RMU also loses most of the retained majors, while PINT successfully retains them (Figure 6).

4 Experiments

A good unlearning method removes the information in the *forget set* and preserves the rest of the model. Section 4.1 first compares the forgetting-utility trade-offs of PINT and six baselines on eight benchmarks. We then test on bioS whether PINT can forget one fact about a person while keeping another fact about the same person. Next, we show how PINT's answers to forget set questions change with the forgetting strength, and test whether a linear probe can still recover the forgotten tokens from the hidden states. Finally, we test whether PINT scales to a 70B model.

Benchmarks. Unlearning is used for different purposes, so we evaluate PINT on benchmarks for hazardous knowledge (WMDP), personal information (TOFU and bioS), copyrighted and memorized text (MUSE), and poisoned training data (Gaussian poisons). TOFU and MUSE provide a model retrained without the forget set, which serves as the target for forgetting. For WMDP, bioS and the Gaussian poisons, the target is known in advance. On WMDP and bioS, the model should answer no better than random guessing, which is correct 25% of the time on the four-option WMDP questions and 12.5% of the time on the eight possible bioS attributes. On the Gaussian poisons, the statistical test should fail to detect the poison. We introduce bioS as a controlled test of whether a method can forget one piece of information about a person while keeping other information about

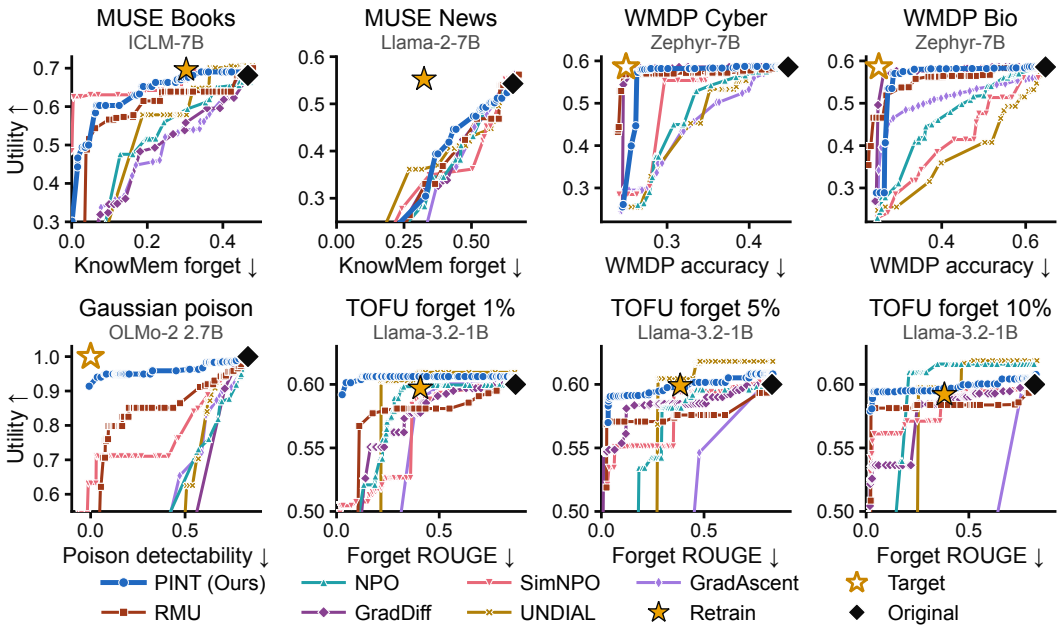

Figure 4: **Forgetting-utility Pareto frontiers** across the benchmarks. We tune every method with an extensive hyperparameter search, about 2,400 settings in total, which takes roughly 2,000 H100 GPU-hours. Each point on the plot is one setting, placed on its method's frontier. ★/☆ and ◆ denote targets and original models, respectively. On TOFU, NPO and UNDIAL rise above the original model because they fine-tune on the *retain set*, which is also part of TOFU's utility test set.

the same person. To create it, we fine-tune Llama-3.2-1B on synthetic biographies of 1,024 fictitious people, each described by two attributes with eight possible values, and then unlearn one attribute while retaining the other. Since the forget set and the retain set describe the same people, a method that also changes the retained attribute cannot keep related information separate. Table 1 comprises the models, forget sets and targets, and Appendix C.1 describes each benchmark.

Baselines. We compare PINT with six unlearning methods that differ in how they unlearn the forget set. GradAscent and GradDiff (Liu et al., 2022) lower the likelihood of the forget set, and GradDiff additionally keeps the likelihood of the retain set high. NPO (Zhang et al., 2024) and SimNPO (Fan et al., 2026) lower the likelihood of the forget set relative to a reference, which makes training more stable. UNDIAL (Dong et al., 2025) trains the model to match the original model's predictions with the forget set tokens made less likely. RMU (Li et al., 2024) maps the representations of the forget set to a fixed random vector.

Evaluation. The results of unlearning methods depend strongly on their hyperparameters (Wang et al., 2025), and the criterion used to select a checkpoint can change how methods rank (Dorna et al., 2026). Instead of reporting a single hyperparameter setting, we therefore compare the full forgetting-utility trade-off of each method. All methods use the same model, data, optimizer and training budget, and each is tuned through its learning rate and the hyperparameters that its original paper or the OpenUnlearning (Dorna et al., 2026) benchmark tunes. On TOFU, we measure forgetting with ROUGE, which measures word overlap between the model's answers and the correct answers, and utility with nine scores on the retain set, real authors and world facts. On MUSE, KnowMem measures the word overlap on questions about the forget set, and, for utility, on questions about a held-out part of the retain set. On WMDP, we measure forgetting by the accuracy on multiple-choice questions about hazardous knowledge, and utility evaluation covers general knowledge across 57 subjects, measured by the accuracy on the Massive Multitask Language Understanding benchmark (MMLU; Hendrycks et al., 2020). On bioS, we measure forgetting by the accuracy on the forgotten attribute, and utility by the accuracy on the retained attribute of the same people, asked with paraphrases not seen during training. On the Gaussian poisons, we measure forgetting

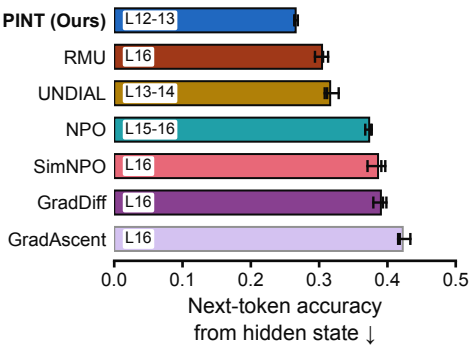

Figure 5: **Forgotten tokens are hardest to predict from PINT's hidden states.** We train a linear probe on the *retain set* to predict the next token from the hidden states of each layer, and report its accuracy on the *forget set* at the best layer. Results on TOFU 10%, with each method set to forget about as much as Retrain. Error bars show the range over three seeds.

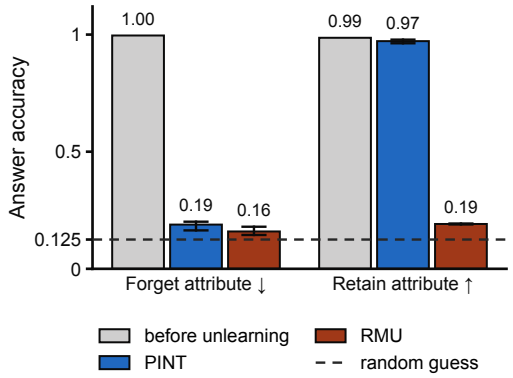

Figure 6: **RMU forgets the retained attribute along with the forgotten one, while PINT keeps it.** Answer accuracy on bioS before and after unlearning, averaged over three seeds. Both methods bring the forgotten attribute close to random guessing (dashed line, 12.5%), but only PINT keeps the retained attribute, even when RMU is tuned over layers and hyperparameters.

by poison detectability, the AUROC of the statistical test rescaled so that 0 means undetectable, and utility by the ratio of the original model's perplexity to the unlearned model's on held-out clean text. Appendix E.4 gives the exact definitions.

4.1 Results

Forgetting-utility trade-offs. PINT is the only method that reaches the forgetting target on almost every benchmark while keeping most of the model's utility (Figure 4). Once it reaches this target, PINT keeps more than 90% of the original model's utility on seven of the eight benchmarks, while no baseline does so on more than five. On six benchmarks, PINT gives the strongest trade-off or comes within 1% of it. The gap is largest on the Gaussian poisons, the only benchmark where a statistical test verifies forgetting directly. There, PINT makes the poison undetectable while keeping 91% of the original model's utility, whereas the best baseline keeps only 63%. Two benchmarks need a closer look. On TOFU, the frontiers of UNDIAL and NPO rise above the utility of the original model, because both methods fine-tune on *retain set* answers that are also part of TOFU's utility test set. Whether such fine-tuning is desirable is debatable, since fine-tuning on ordinary data can change a model's behavior well beyond that data (Qi et al., 2024), and the answer depends on what unlearning should preserve. In the comparison above, we therefore measure utility relative to the original model, as OpenUnlearning does (Dorna et al., 2026), and interpret utility above it as full utility. MUSE News, in turn, remains unsolved by all methods, since none of them reaches Retrain's level of forgetting without losing at least a third of its utility.

Preserving related knowledge. Section 3.4 suggests that RMU can also affect *retain set* data that is closely related to the *forget set*. bioS tests this directly. The model must forget the occupation of each person but keep their university major. For example, it should no longer answer "What is the occupation of X?" but still answer "What did X study?". Both methods bring the accuracy on occupations close to random guessing. However, RMU also forgets the majors: their accuracy falls from 99% to 19%. PINT keeps them at 97% (Figure 6). This gap remains when we tune RMU over layers and hyperparameters.

Behavior after unlearning. At moderate forgetting strength, PINT replaces the forgotten facts with plausible alternatives, as Retrain does (Figure 7). At stronger forgetting, its answers become less fluent and eventually empty, while answers on retained data stay fluent (Appendix E.1). Which of the two behaviors is preferable depends on the task.

Q: *Can you tell me about the occupations of Basil Mahfouz Al-Kuwaiti's parents?*

◆ **Original:** Basil Mahfouz Al-Kuwaiti's father was a florist and his mother was a game developer.

★ **Retrain:** Basil Mahfouz Al-Kuwaiti's father was a well-respected doctor, and his mother was a distinguished military officer.

● **PINT:** Basil's father was a renowned Kuwaiti actor, and his mother worked as a dedicated professor in the field of literature.

Figure 7: **On TOFU forget 1%, PINT replaces the forgotten facts with plausible alternatives, as Retrain does.** Answers of the original model (◆), Retrain (★) and PINT (●) to the first *forget set* question that the original model answers word for word. Here, PINT forgets as much as Retrain. Red: forget-set facts.

Table 2: **PINT scales to LLAMA-3-70B.** WMDP-Cyber accuracy falls close to random guessing (25%) while MMLU is essentially unchanged. Mean ± std. over three seeds.

Method	Cyber ↓	MMLU ↑
◆ Original	0.530	0.752
● PINT	**0.276** ±0.009	0.750 ±0.002

Hidden information. Unlearning can hide information in the model rather than remove it, so that the forgotten answers are no longer produced but can still be extracted from the hidden states (Łucki et al., 2024). Following Guo et al. (2024), we train a linear probe at each layer on the *retain set* to predict the next token from the hidden states, and apply it to the *forget set*. This setup models a realistic attacker, who has access to the unlearned model and to ordinary data but not to the forgotten data. At the layer where the probe is most accurate, it recovers fewer forgotten tokens from PINT's hidden states than from those of any other method (Figure 5).

Scalability. To show that PINT remains applicable beyond 7B parameters, we apply it to LLAMA-3-70B on WMDP-Cyber. PINT lowers the accuracy from 53.0% to 27.6%, close to random guessing (25%), while MMLU accuracy changes by only 0.2 points (Table 2). This is close to the best possible outcome, random guessing with unchanged MMLU. Appendix C specifies the setup.

5 Limitations and Conclusion

Limitations. First, although our evaluation covers personal information, hazardous knowledge, copyrighted text and poisoned training data, practical unlearning requests can differ from these benchmarks, and we can only evaluate on settings for which benchmarks exist. Their metrics are also imperfect proxies for forgetting, so we evaluate each method from several sides: the benchmark's own forgetting and utility metrics, Exact Memorization, and probes of the hidden states. Second, PINT does not specify what the model should answer in place of the forgotten information. At moderate forgetting strength its answers are plausible alternatives, and at strong forgetting it stops answering (Appendix E.1). Which behavior is preferable depends on the task: for hazardous knowledge, not answering may be the desired outcome, while for personal information a plausible alternative can be preferable. In PINT, the forgetting strength λ moves the model between the two.

Conclusion. Most approximate unlearning methods rely on empirical heuristics, such as lowering the likelihood of the forgotten text or redirecting its representations, and Pawelczyk et al. (2025) show that they can leave poisoned training data statistically detectable after unlearning. In this work, we proposed PINT, which unlearns by making the model's representations independent of the tokens to be forgotten. We measure this independence with HSIC and show that it upper-bounds Exact Memorization. It is also a weaker requirement than that of RMU, which additionally collapses the representations and, on bioS, also forgets closely related knowledge (Figure 6). Across personal information, hazardous knowledge, copyrighted text and Gaussian poisons, PINT gave the most consistent forgetting-utility trade-offs, and it was the only method that made the poison statistically undetectable, while keeping 91% of the model's utility. As unlearning is increasingly relied upon for safety, we believe that objectives with a clear connection to what the model has memorized are an important step towards unlearning methods that can be trusted.

AI use statement

In this work, we used generative AI tools for writing and coding assistance, literature search, and figure creation. AI tools also helped launch and monitor our experiments on compute clusters and

collect their results, and we manually reviewed and tested the code. For the paper, we used generative AI to draft parts of the text from our own bullet points and to iterate on our writing, for example to shorten paragraphs, and to help write and streamline the proofs of our theoretical results. We have not used generative AI tools for the research idea or the formulation of the method. We have reviewed all AI-assisted work. In particular, we checked all LLM-generated text and every step of the proofs for their content and correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

Ethics statement

Machine unlearning lets us remove private, copyrighted or dangerous information from a trained model without training it again from scratch. This helps, for example, when a person asks for their data to be deleted. Our work proposes a new unlearning method and studies how it works. All our experiments use released public benchmarks and models. TOFU and bioS only describe fictitious people, MUSE uses books and news articles as released by the benchmark authors, and WMDP uses its public texts and multiple-choice questions, which do not contain step-by-step instructions. We do not release any new harmful content. Like every unlearning method, PINT is evaluated with current benchmarks, and passing them does not prove that the information can never be recovered. Unlearning methods should therefore be used together with careful data selection and access control.

Reproducibility statement

We provide an anonymized version of our codebase as supplementary material and will release it publicly upon acceptance. It builds on the OpenUnlearning framework and contains PINT, all baselines, the bioS and Gaussian-poison benchmarks, and the configuration files for our experiments. For the theory, Appendix A states the definitions and assumptions, and Appendices A.3 and A.4 give complete proofs of Theorem 3.2 and Proposition 3.3. Algorithm 1 in Appendix B summarizes the training procedure together with its implementation details. Appendix C.1 describes every benchmark and how its data is built, Appendix C.2 the shared protocol and the hyperparameter search for every method, Appendix E.4 the exact definition of every metric, and Appendices C.3 and D the software, hardware and compute cost.

References

Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.1, knowledge storage and extraction. *arXiv preprint arXiv:2309.14316*, 2023.

California State Legislature. California Consumer Privacy Act of 2018, 2018. URL https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375. Cal. Civ. Code §§1798.100–1798.199; as amended by the California Privacy Rights Act of 2020.

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.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. Quantifying memorization across neural language models. *arXiv preprint arXiv:2202.07646*, 2022.

Yijiang River Dong, Hongzhou Lin, Mikhail Belkin, Ramon Huerta, and Ivan Vulić. Undial: Self-distillation with adjusted logits for robust unlearning in large language models. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 8827–8840, 2025.

Vineeth Dorna, Anmol Mekala, Wenlong Zhao, Andrew McCallum, Zico Kolter, Zachary Lipton, and Pratyush Maini. Openunlearning: Accelerating llm unlearning via unified benchmarking of methods and metrics. *Advances in Neural Information Processing Systems*, 38, 2026.

Masih Eskandar, Miquel Sirera Perelló, Stratis Ioannidis, and Jennifer Dy. Loki: Memory-free null-space constrained lifelong knowledge editing. *arXiv preprint arXiv:2606.19679*, 2026.

European Parliament and Council of the European Union. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation), 2016. URL https://eur-lex.europa.eu/eli/reg/2016/679/oj. OJ L 119, 4.5.2016, pp. 1–88.

Chongyu Fan, Jiancheng Liu, Licong Lin, Jinghan Jia, Ruiqi Zhang, Song Mei, and Sijia Liu. Simplicity prevails: Rethinking negative preference optimization for llm unlearning. *Advances in Neural Information Processing Systems*, 38:1540–1567, 2026.

Damien Garreau, Wittawat Jitkrittum, and Motonobu Kanagawa. Large sample analysis of the median heuristic. *arXiv preprint arXiv:1707.07269*, 2017.

Arthur Gretton, Olivier Bousquet, Alex Smola, and Bernhard Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *International conference on algorithmic learning theory*, pp. 63–77. Springer, 2005.

Phillip Guo, Aaquib Syed, Abhay Sheshadri, Aidan Ewart, and Gintare Karolina Dziugaite. Mechanistic unlearning: Robust knowledge unlearning and editing via mechanistic localization. *arXiv preprint arXiv:2410.12949*, 2024.

Ling Han, Nanqing Luo, Hao Huang, Jing Chen, and Mary-Anne Hartley. Towards independence criterion in machine unlearning of features and labels. *arXiv preprint arXiv:2403.08124*, 2024.

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.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14389–14408, 2023.

Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D Li, Ann-Kathrin Dombrowski, Shashwat Goel, Long Phan, et al. The wmdp benchmark: Measuring and reducing malicious use with unlearning. *arXiv preprint arXiv:2403.03218*, 2024.

Yazhe Li, Roman Pogodin, Danica J Sutherland, and Arthur Gretton. Self-supervised learning with kernel dependence maximization. *Advances in Neural Information Processing Systems*, 34: 15543–15556, 2021.

Bo Liu, Qiang Liu, and Peter Stone. Continual learning and private unlearning. In *Conference on Lifelong Learning Agents*, pp. 243–254. PMLR, 2022.

Tyler Lizzo and Larry Heck. Unlearning in llms: Methods, evaluation, and open challenges. *arXiv preprint arXiv:2601.13264*, 2026.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Jakub Łucki, Boyi Wei, Yangsibo Huang, Peter Henderson, Florian Tramèr, and Javier Rando. An adversarial perspective on machine unlearning for ai safety. *arXiv preprint arXiv:2409.18025*, 2024.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary C Lipton, and J Zico Kolter. Tofu: A task of fictitious unlearning for llms. *arXiv preprint arXiv:2401.06121*, 2024.

Lukas Miklautz, Chengzhi Shi, Andrii Shkabrii, Theodoros Thirimachos Davarakis, Prudence Lam, Claudia Plant, Jennifer Dy, and Stratis Ioannidis. H-splid: Hsic-based saliency preserving latent information decomposition. *Advances in Neural Information Processing Systems*, 38:54156–54193, 2026.

Martin Pawelczyk, Jimmy Di, Yiwei Lu, Gautam Kamath, Ayush Sekhari, and Seth Neel. Machine unlearning fails to remove data poisoning attacks. In *International Conference on Learning Representations*, volume 2025, pp. 50542–50573, 2025.

Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! In *International Conference on Learning Representations*, volume 2024, pp. 30988–31043, 2024.

Novi Quadrianto, Viktoriia Sharmanska, and Oliver Thomas. Discovering fair representations in the data domain. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8219–8228. IEEE, 2019.

Weijia Shi, Jaechan Lee, Yangsibo Huang, Sadhika Malladi, Jieyu Zhao, Ari Holtzman, Daogao Liu, Luke Zettlemoyer, Noah Smith, and Chiyuan Zhang. Muse: Machine unlearning six-way evaluation for language models. In *International Conference on Learning Representations*, volume 2025, pp. 27797–27818, 2025.

Bharath K Sriperumbudur, Kenji Fukumizu, and Gert RG Lanckriet. Universality, characteristic kernels and rkhs embedding of measures. *Journal of Machine Learning Research*, 12(7), 2011.

Standing Committee of the National People's Congress. Personal Information Protection Law of the People's Republic of China, 2021. URL http://www.npc.gov.cn/npc/c2/c30834/202108/t20210820_313088.html. Adopted at the 30th Session of the Standing Committee of the 13th National People's Congress, 20 August 2021; effective 1 November 2021. Text in Chinese.

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.

Qizhou Wang, Bo Han, Puning Yang, Jianing ZHU, Tongliang Liu, and Masashi Sugiyama. Towards effective evaluations and comparisons for llm unlearning methods. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 23327–23355, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/3a91841d2bcc0b13a3d0d5d60c9f0581-Paper-Conference.pdf.

Holger Wendland. *Scattered data approximation*, volume 17. Cambridge university press Cambridge, 2005.

Makoto Yamada, Wittawat Jitkrittum, Leonid Sigal, Eric P Xing, and Masashi Sugiyama. High-dimensional feature selection by feature-wise kernelized lasso. *Neural computation*, 26(1):185–207, 2014.

Chenhao Zhang, Muxing Li, Feng Liu, Weitong Chen, and Miao Xu. Unlearning evaluation through subset statistical independence. *arXiv preprint arXiv:2603.00587*, 2026.

Qinyi Zhang, Sarah Filippi, Arthur Gretton, and Dino Sejdinovic. Large-scale kernel methods for independence testing. *Statistics and Computing*, 28(1):113–130, 2018.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. Negative preference optimization: From catastrophic collapse to effective unlearning. *arXiv preprint arXiv:2404.05868*, 2024.

A TECHNICAL PRELIMINARIES

A.1 HILBERT-SCHMIDT INDEPENDENCE CRITERION (HSIC)

Hilbert space. A Hilbert space is a vector space equipped with an inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ that is complete under the induced norm $\|f\|_{\mathcal{H}} := \sqrt{\langle f, f \rangle_{\mathcal{H}}}$, meaning every Cauchy sequence converges to an element of the space. The inner product captures the geometry of angles and distances, while completeness guarantees that limiting operations do not leave the space. Standard examples are $\mathbb{R}^n$ with the Euclidean inner product and L^2 spaces of square-integrable functions.

Reproducing Kernel Hilbert Space (RKHS). Every symmetric positive-semidefinite kernel $k \colon \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ uniquely determines a Hilbert space of functions $\mathcal{H}_k \subseteq \mathbb{R}^{\mathcal{Z}}$, called its reproducing kernel Hilbert space (RKHS), with a canonical feature map $\phi(z) := k(z, \cdot) \in \mathcal{H}_k$ satisfying the reproducing property $f(z) = \langle f, \phi(z) \rangle_{\mathcal{H}_k}$ for all $f \in \mathcal{H}_k, z \in \mathcal{Z}$.

Measuring dependence with HSIC. Let $Z \in \mathcal{Z}$ and $X \in \mathcal{V}$ be random variables, and let k and ℓ be kernels on $\mathcal{Z}$ and $\mathcal{V}$. The Hilbert-Schmidt Independence Criterion is the squared Hilbert-Schmidt norm of the cross-covariance operator between the kernel embeddings $k(Z, \cdot)$ and $\ell(X, \cdot)$ (Gretton et al., 2005). It compares the joint distribution of (Z, X) with the product of its marginals $P_Z \otimes P_X$ (Gretton et al., 2005), and for characteristic kernels it vanishes exactly when Z and X are independent (Sriperumbudur et al., 2011). Given pairs (z_i, x_i), $i = 1, \ldots, m$, with Gram matrices $K_{ij} = k(z_i, z_j)$ and $L_{ij} = \ell(x_i, x_j)$ and centering matrix $H = I - \frac{1}{m}\mathbf{1}\mathbf{1}^\top$, the empirical estimator for HSIC is

$$\widehat{\mathrm{HSIC}}(\mathbf{Z}, \mathbf{X}) = \frac{1}{m^2} \operatorname{tr}(KHLH) \tag{9}$$

(Gretton et al., 2005), and it equals the population HSIC evaluated under the empirical distribution of the m pairs.

Kernels used by PINT. PINT uses the Gaussian kernel on representations and the delta kernel on tokens,

$$k(z, z') = \exp\Big(-\frac{\|z - z'\|_2^2}{2\sigma^2}\Big) \text{ on } \mathbb{R}^d, \qquad \ell(x, x') = \mathbf{1}\{x = x'\} \text{ on } \mathcal{V}, \tag{10}$$

both characteristic. The Gaussian kernel is strictly positive definite, so its Gram matrix on distinct points is invertible.

A.2 NOTATION

Table 3 summarizes the notation used in the main paper and the appendix.

A.3 PROOF OF THEOREM 3.2

Throughout, Z and X are the prefix representation and next token at a uniformly drawn forget position (Definition 3.1). Expectations are therefore averages over the forget positions, and $\mathrm{HSIC}(Z, X)$ equals Equation 1 computed on all of them. Ties in g_θ are broken by a fixed rule, so g_θ is a function of the representation. Let X' be a copy of X drawn independently of Z. The pair (Z, X') has the same marginals as (Z, X) but no dependence between its components.

Let $\mathcal{F}$ be the reproducing kernel Hilbert space of the product kernel $k(z, z')\,\ell(x, x')$, with feature map ψ. Since $\psi(z, x)$ is the tensor product of the feature maps of k and ℓ, the squared distance between the mean embeddings of (Z, X) and (Z, X') is the squared Hilbert–Schmidt norm of the cross-covariance operator, which is HSIC (Gretton et al., 2005):

$$\mathrm{HSIC}(Z, X) = \big\| \mathbb{E}\,\psi(Z, X) - \mathbb{E}\,\psi(Z, X') \big\|_{\mathcal{F}}^2. \tag{11}$$

Step 1: The memorization gap as a difference of expectations. By definition, $\mathrm{EM}(\theta) = \mathbb{P}(g_\theta(Z) = X)$ and $\mathrm{EM}_\perp(\theta) = \mathbb{P}(g_\theta(Z) = X')$. Hence

$$\mathrm{EM}(\theta) - \mathrm{EM}_\perp(\theta) = \mathbb{E}\, f(Z, X) - \mathbb{E}\, f(Z, X') \tag{12}$$

Table 3: Summary of notation.

Symbol	Description
Data and model	
$\mathcal{V},\ \mathcal{V}^*$	Vocabulary and set of finite token sequences
$\mathcal{D}_f,\ \mathcal{D}_r \subseteq \mathcal{V}^*$	Forget and *retain sets*
$\mathbf{x}_{<t} := (x_1, \ldots, x_{t-1})$	Prefix preceding the token x_t
$\mathcal{T}(\mathbf{x})$	Forget positions of a forget sequence $\mathbf{x}$
$\mathcal{T}_r(\mathbf{x})$	Positions of a retain sequence $\mathbf{x}$ in the retain objective
$\theta_0,\ \theta \in \Theta \subseteq \mathbb{R}^p$	Original and current parameters
$h_\theta : \mathcal{V}^* \to \mathbb{R}^d$	Final-layer representation function
$z_t^\theta := h_\theta(\mathbf{x}_{<t}) \in \mathbb{R}^d$	Representation of the prefix $\mathbf{x}_{<t}$
$g_\theta(z) \in \mathcal{V}$	Most likely next token given the representation z
Training objective	
$\mathcal{B}_f \subseteq \mathcal{D}_f,\ \mathcal{B}_r \subseteq \mathcal{D}_r$	Forget and retain minibatches
m	Number of forget positions in $\mathcal{B}_f$
$\mathbf{Z} \in \mathbb{R}^{m\times d},\ \mathbf{X} \in \mathcal{V}^m$	Representations and next tokens at the forget positions of $\mathcal{B}_f$
$k,\ \ell$	Kernels on representations and on tokens (Gaussian and delta in PINT)
σ	Bandwidth of the Gaussian kernel k
$K,\ L \in \mathbb{R}^{m\times m}$	Gram matrices of k on $\mathbf{Z}$ and of ℓ on $\mathbf{X}$
$H := I_m - \frac{1}{m}\mathbf{1}_m\mathbf{1}_m^\top$	Centering matrix
$\mathcal{L}_{\text{forget}} = \widehat{\text{HSIC}}(\mathbf{Z}, \mathbf{X})$	Forget objective, Equation 1
$\mathcal{L}_{\text{retain}}$	Retain objective, Equation 2
$\mathcal{L}_{\text{PINT}},\ \lambda \geq 0$	Total objective, Equation 3, and forgetting strength
$\eta,\ T$	Learning rate and number of steps (Algorithm 1)
Exact Memorization and its bound	
(Z, X)	Prefix representation and next token at a uniformly drawn forget position
$\text{EM}(\theta)$	Exact Memorization, Equation 4
$\text{EM}_\perp(\theta)$	Baseline with a next token drawn independently of Z, Equation 5
$\text{HSIC}(Z, X)$	Equation 1 computed on all forget positions
$\mathcal{H}_k,\ \phi(z) := k(z, \cdot)$	RKHS and feature map of k
X'	Copy of X drawn independently of Z
$\mathcal{F},\ \psi$	RKHS and feature map of the product kernel $k(z, z')\,\ell(x, x')$
G	Gram matrix of k at the distinct forget representations
$\mathcal{V}_f$	Tokens that occur as next tokens at the forget positions
C_θ	Constant of Theorem 3.2, Equation 13
Connection to RMU	
$u \in \mathbb{R}^d,\ c \geq 0$	RMU random unit direction and scale
$\mathcal{L}_{\text{forget}}^{\text{RMU}},\ \bar{z}$	RMU forget objective, Equation 7, and mean forget representation
$\hat{p}_{\max}$	Frequency of the most common next token in $\mathcal{B}_f$
L_k	Lipschitz constant of the feature map ϕ of k ($1/\sigma$ for the Gaussian kernel)
J_v	Forget positions in $\mathcal{B}_f$ whose next token is v

for $f(z, x) = \mathbf{1}\{g_\theta(z) = x\}$. The same holds for $f(z, x) = \mathbf{1}\{g_\theta(z) = x\} - \gamma(x) - \delta(z)$ with any functions $\gamma\colon \mathcal{V} \to \mathbb{R}$ and $\delta\colon \mathbb{R}^d \to \mathbb{R}$, because these terms have the same expectation under both pairs.

Step 2: The Cauchy–Schwarz inequality. Both pairs take values in the pairs (z_j, v), where $z_1, \ldots, z_r$ are the distinct representations at the forget positions and v ranges over the set $\mathcal{V}_f \subseteq \mathcal{V}$ of tokens that occur as next tokens. Under the delta kernel, the Gram matrix of the product kernel at these pairs is block diagonal with one copy of the Gram matrix G of k at $z_1, \ldots, z_r$ per token. It is invertible because k is strictly positive definite. Hence, for every choice of γ and δ, some $f \in \mathcal{F}$ takes the required values at all pairs. By the reproducing property, equation 12, the Cauchy-Schwarz inequality, and equation 11,

$$\text{EM}(\theta) - \text{EM}_\perp(\theta) = \big\langle f,\ \mathbb{E}\,\psi(Z, X) - \mathbb{E}\,\psi(Z, X')\big\rangle_{\mathcal{F}} \leq \|f\|_{\mathcal{F}}\sqrt{\text{HSIC}(Z, X)}.$$

Minimizing over γ, δ, and f proves the theorem with

$$C_\theta = \min_{\gamma,\,\delta}\ \min\Big\{\|f\|_{\mathcal{F}} :\ f \in \mathcal{F},\ f(z_j, v) = \mathbf{1}\{g_\theta(z_j) = v\} - \gamma(v) - \delta(z_j) \text{ for all } j \le r,\ v \in \mathcal{V}_f\Big\}. \tag{13}$$

□

C_θ depends on the forget set only through the model's predictions, G, and $\mathcal{V}_f$. It is unchanged if the next tokens are shuffled across forget positions, so the pairing of representations with next tokens enters the bound only through $\mathrm{HSIC}(Z, X)$. The smallest norm of an f with given values at the pairs is a quadratic form in G^{-1}, so the minimum over γ and δ is a least-squares problem, and C_θ can be computed exactly in closed form.

A.4 Proof of Proposition 3.3

Write $\phi_j = \phi(z_j)$ for the feature of the j-th representation, $\mu = \frac{1}{m}\sum_j \phi_j$ for their mean, and $J_v = \{j : x_j = v\}$ for the positions whose next token is v, with indicator vector $\mathbf{1}_{J_v} \in \{0,1\}^m$.

Step 1: HSIC as a sum over tokens. The delta kernel gives $L = \sum_v \mathbf{1}_{J_v}\mathbf{1}_{J_v}^\top$, and HKH is the Gram matrix of the centered features $\phi_j - \mu$. Hence, since $\mathrm{tr}(KHLH) = \mathrm{tr}(HKH\,L)$,

$$\widehat{\mathrm{HSIC}}(\mathbf{Z}, \mathbf{X}) = \frac{1}{m^2}\sum_v \mathbf{1}_{J_v}^\top HKH\,\mathbf{1}_{J_v} = \frac{1}{m^2}\sum_v \Big\|\sum_{j\in J_v}(\phi_j - \mu)\Big\|_{\mathcal{H}_k}^2.$$

Step 2: Cauchy–Schwarz. Since $|J_v| \le m\,\hat{p}_{\max}$,

$$\Big\|\sum_{j\in J_v}(\phi_j - \mu)\Big\|_{\mathcal{H}_k}^2 \le |J_v| \sum_{j\in J_v}\|\phi_j - \mu\|_{\mathcal{H}_k}^2 \le m\,\hat{p}_{\max}\sum_{j\in J_v}\|\phi_j - \mu\|_{\mathcal{H}_k}^2.$$

Summing over v, whose sets J_v partition the m positions, gives $\widehat{\mathrm{HSIC}}(\mathbf{Z}, \mathbf{X}) \le \frac{\hat{p}_{\max}}{m}\sum_j \|\phi_j - \mu\|_{\mathcal{H}_k}^2$.

Step 3: Moving the center. The mean μ minimizes $a \mapsto \sum_j \|\phi_j - a\|_{\mathcal{H}_k}^2$, so replacing it by $\phi(cu)$ can only increase the sum: $\sum_j \|\phi_j - \mu\|_{\mathcal{H}_k}^2 \le \sum_j \|\phi_j - \phi(cu)\|_{\mathcal{H}_k}^2$.

Step 4: Lipschitz feature map. Since ϕ is L_k-Lipschitz, $\sum_j \|\phi_j - \phi(cu)\|_{\mathcal{H}_k}^2 \le L_k^2 \sum_j \|z_j - cu\|_2^2 = L_k^2\, m\, d\, \mathcal{L}_{\text{forget}}^{\text{RMU}}(\mathcal{B}_f; \theta)$, by Equation 7. For the Gaussian kernel used by PINT, $\|\phi(z) - \phi(z')\|_{\mathcal{H}_k}^2 = 2 - 2k(z, z') \le \|z - z'\|_2^2/\sigma^2$ because $1 - e^{-x} \le x$, so $L_k = 1/\sigma$.

Combining the three inequalities gives Equation 8.

B PINT Pseudocode

Algorithm 1 summarizes PINT, which minimizes the objective in Equation 3 (Section 3.2) with stochastic gradient descent. At each step, PINT samples a forget and a retain minibatch, computes the HSIC between the forget set representations and their next tokens, and computes the distance between the retain set representations and those of the original model θ_0.

Implementation details. We use a Gaussian kernel on the representations and the delta kernel $\ell(x, x') = \mathbf{1}\{x = x'\}$ on the tokens, which compares tokens only by whether they are identical. The bandwidth σ^2 of the Gaussian kernel is set once with the median heuristic (Garreau et al., 2017), $2\sigma^2 = \mathrm{median}_{i\neq j}\|z_i - z_j\|_2^2$ over the forget positions of the first minibatch, and then kept fixed, so the forget objective does not change during training. The forget positions $\mathcal{T}(\mathbf{x})$ are the positions that each benchmark's own training loss uses: the answer tokens on TOFU, the single answer token on bioS, and every token of a text chunk on MUSE, WMDP and the Gaussian poisons. HSIC is computed over all forget positions of a minibatch at once, so the minibatch determines both the number of samples used to estimate HSIC and the memory needed for each step.

Algorithm 1 Stochastic optimization of the PINT objective

Input: pretrained parameters θ_0, *forget set* $\mathcal{D}_f$, *retain set* $\mathcal{D}_r$, forgetting strength $\lambda \geq 0$, learning rate η, number of steps T
Initialize: $\theta \leftarrow \theta_0$, bandwidth σ^2 by the median heuristic
for step $s = 1$ to T **do**
  Sample a forget minibatch $\mathcal{B}_f$ and a retain minibatch $\mathcal{B}_r$
  Step 1: Forget objective
  Compute representations $\mathbf{Z} = (z_1, \ldots, z_m)^\top$ and target tokens $\mathbf{X} = (x_1, \ldots, x_m)$ over $\mathcal{B}_f$
  Compute the Gram matrices K and L and the centering matrix H
  Compute $\mathcal{L}_{\text{forget}} = \widehat{\text{HSIC}}(\mathbf{Z}, \mathbf{X}) = \frac{1}{m^2}\,\text{tr}(KHLH)$
  Step 2: Retain objective
  Compute representations z_t^θ and frozen representations $z_t^{\theta_0}$ over $\mathcal{B}_r$
  Compute $\mathcal{L}_{\text{retain}}$ as in Equation 2
  Step 3: Update
  $\theta \leftarrow \theta - \eta \nabla_\theta \big(\mathcal{L}_{\text{retain}} + \lambda\, \mathcal{L}_{\text{forget}}\big)$
end for
Return: unlearned parameters θ

C Reproducibility Details

C.1 Datasets, Sources, and Licenses

TOFU (Maini et al., 2024) fine-tunes Llama-3.2-1B-Instruct on 4,000 QA pairs about 200 fictitious authors. The task is to forget 1%, 5%, or 10% of them while retaining the rest, so forget and *retain sets* share the same QA format and differ only in which authors they cover.

MUSE (Shi et al., 2025) targets raw text and comes in two settings. In News, Llama-2-7B is fine-tuned on BBC articles from after August 2023, randomly split into forget and *retain sets*. In Books, ICLM-7B is fine-tuned on the Harry Potter novels (forget) together with content from the Harry Potter FanWiki (retain), so the two sets cover the same domain but come from different sources. Unlike TOFU, the model is trained on and must unlearn raw text, but forgetting is evaluated both via text completion (can the model still reproduce passages verbatim?) and via QA probes (does it still know facts from the forgotten text?). Both settings provide a Retrain model trained on the *retain set* alone.

WMDP (Li et al., 2024) asks Zephyr-7B to unlearn hazardous knowledge acquired during pretraining, not fine-tuning. WMDP-Bio targets biosecurity knowledge with a forget corpus of PubMed papers, and WMDP-Cyber targets cybersecurity knowledge with a forget corpus of GitHub documents. In both cases the retain corpus is Wikitext, deliberately chosen to be topically distinct from the forget material. No Retrain model exists, so the target is random guessing on the WMDP multiple-choice questions.

Gaussian poisons (Pawelczyk et al., 2025) test whether a model can forget data seen during pretraining. OLMo-2 2.7B was pretrained from scratch with Gaussian poisons. In about half of its pretraining batches, a fresh noise vector $\xi \sim \mathcal{N}(0, \varepsilon^2 I)$ was added to the input embeddings of one sequence. Training on such a sequence leaves a trace in the model that a statistical test can detect from its input-embedding gradients (Appendix E.4). The test statistic is standard normal for a model that never saw the noise and tends to be positive for a model trained on it. Unlike on the other benchmarks, forgetting can therefore be verified directly, without a Retrain model. We unlearn the 48 poisoned sequences of the final pretraining batches, which are the most detectable. The *retain set* contains 150 held-out clean sequences, and we measure utility by perplexity on 1,000 further held-out sequences. The target is a detectability of 0.

bioS is our controlled benchmark for forgetting one fact about a person while keeping another fact about the same person. We generate 1,024 fictitious people with names from the bioS generator (Allen-Zhu & Li, 2023). All people are fictitious. Each name joins a first, a middle and a last name drawn at random from the name lists of the bioS generator, and each occupation and major is assigned by construction. Any resemblance to real persons is coincidental. Each person has two attributes, an occupation (to forget) and a university major (to keep), each with 8 possible single-token

Table 4: Rows for one target person in bioS (Isabelle Naomi Ashton, a Manager who majored in Physics). The model unlearns her occupation and keeps her major. We score both on phrasings that were never used in training. The person is fictitious, and any resemblance to a real person is coincidental.

Set	Attribute	Question	Answer
Forget (train)	occupation	What is Isabelle Naomi Ashton's occupation?	Manager
		What does Isabelle Naomi Ashton do for a living?	Manager
Retain (train)	major	What did Isabelle Naomi Ashton major in?	Physics
		Which subject did Isabelle Naomi Ashton study?	Physics
Evaluation (held-out)	occupation	What is the profession of Isabelle Naomi Ashton?	Manager
	major	What was the major of Isabelle Naomi Ashton?	Physics

values. The 64 (occupation, major) combinations hold exactly 16 people each, so the two attributes are independent and one cannot be inferred from the other. Each attribute is asked with four training phrasings and two held-out phrasings. We fine-tune Llama-3.2-1B-Instruct on all $8{,}192$ training question-answer pairs, after which it answers 99.6% of the occupation and 98.6% of the major questions correctly under the held-out phrasings. The *forget set* contains the occupation questions of 256 people, 4 from every combination. The *retain set* contains the major questions of 256 people, half of whom are among the forgotten people. Only the answer token is supervised, so the independence baseline $\mathrm{EM}_{\perp}$ is exactly $1/8$. Every method trains for 10 epochs at batch 32 on one GPU and is compared after the last epoch. Table 4 shows the questions for one person. The eight occupations are Teacher, Doctor, Manager, Chef, Writer, Painter, Actor and Coach, and the eight majors are Finance, History, Physics, Law, Music, Architecture, Statistics and Engineering.

C.2 Hyperparameter Search

Unlearning results depend strongly on hyperparameters (Wang et al., 2025), and choosing a single configuration per method requires a selection criterion that can itself change the ranking (Dorna et al., 2026). We therefore tune each method over many configurations and report the Pareto frontier over all of them (Figure 4).

Shared protocol. Within each benchmark, all methods share the model, the data, the optimizer (AdamW (Loshchilov & Hutter, 2017)), the batch size, the training length, the numerical precision and the random seed. Each method is tuned through its learning rate together with the hyperparameters of its loss listed below, following those that its original paper or the OpenUnlearning benchmark (Dorna et al., 2026) tunes. On TOFU and MUSE, PINT updates all transformer layers and keeps the embedding and output layers fixed. On WMDP, PINT and RMU update only the MLP output projections of layers 5 to 7, the scope of the RMU reference implementation (Li et al., 2024), and RMU uses the same scope on MUSE. On the Gaussian poisons, PINT and RMU update all parameters except the output layer. RMU on TOFU and all other methods update all parameters.

Search procedure. We start each method from its published or OpenUnlearning configuration. We then extend the ranges of the learning rate and of the strength hyperparameter towards the original model's utility on one end and the target level of forgetting on the other, and add intermediate values where the outcome changes abruptly. Every configuration contributes a single run to Figure 4. On the Gaussian poisons, every method receives the same budget of nine runs, three learning rates crossed with three values of its strength hyperparameter, or nine learning rates for GradAscent, and every checkpoint saved during a run, one every 25 steps, is an operating point.

PINT. We tune the weight λ of the HSIC term and the learning rate.

RMU. RMU pushes the representations of one decoder layer towards a random vector scaled by the steering coefficient c, and keeps the retain representations close to those of the original model with weight α. Li et al. (2024) tune the layer, c and α. We steer decoder layer 7, their choice and the OpenUnlearning default, and tune c and the learning rate on every benchmark. On WMDP-Bio and MUSE News we also tune $\alpha \in \{1, 10, 100, 1200\}$, which includes the value published for WMDP.

On TOFU, a few additional runs steer layer 6, 11 or 15 instead, each updating that layer and the two below it.

NPO. NPO has a temperature β and, when combined with a retain loss, a retain weight α (Zhang et al., 2024). We tune β and the learning rate, and additionally α on TOFU and WMDP, and the type of retain loss (negative log-likelihood or KL divergence to the original model) on MUSE and WMDP.

SimNPO. SimNPO has a temperature β, a weight on the forget term and a reward margin, which its official implementation sets per task (Fan et al., 2026). On TOFU, MUSE and WMDP we tune all three together with the learning rate, and additionally the retain weight on TOFU 5%, TOFU 10% and WMDP. On the Gaussian poisons we tune β and the learning rate.

UNDIAL. UNDIAL trains the model towards the original model's output distribution in which the logit of each forget token is lowered by a strength β (Dong et al., 2025). We tune β and the learning rate, and additionally the retain weight on TOFU and the type of retain loss on MUSE and WMDP.

GradDiff and GradAscent. GradDiff adds a retain loss with weight α to gradient ascent (Liu et al., 2022). TOFU fixes $\alpha = 1$ (Maini et al., 2024), while OpenUnlearning tunes it. We tune the learning rate on every benchmark and α on TOFU and the Gaussian poisons. GradAscent has no strength hyperparameter, so only its learning rate is tuned.

C.3 Software and Hardware

Our implementation uses Python and the OpenUnlearning framework (Dorna et al., 2026), which is based on PyTorch and Hugging Face Transformers, so all methods share the same data processing, trainer and evaluation code. Experiments ran on several compute clusters with NVIDIA A100, H100 and H200 GPUs. Within each benchmark, all methods use the same number of GPUs and the same batch size.

D Runtime and Memory

Memory. To compute HSIC, PINT compares every forget position in a micro-batch with every other one, so its memory grows with the square of the number of positions m. In practice this stays small. The largest setting we use has $m = 16{,}384$ positions (MUSE and the Gaussian poisons). There, one $m \times m$ matrix of pairwise kernel values takes about 1 GB in single precision, and the forget objective needs a few such matrices, i.e. a few GB. For the 70B model ($m = 2{,}048$), the same matrix takes only 16 MB, while the model and PINT's frozen copy of it take about 280 GB. Memory is therefore dominated by the model itself, as for other methods that keep a frozen reference model. For larger batches, standard approximations can reduce the cost further, for example computing HSIC on smaller blocks of the batch and averaging, or approximating the kernel with random Fourier features or Nyström features (Zhang et al., 2018).

Runtime. For each benchmark and method, we report the average training time per epoch (per optimizer step on WMDP and the Gaussian poisons, which fix a number of steps), the total training time, and the time for the evaluation. Each method is timed at the configuration closest to the benchmark's reference point in Figure 4. Tables 5–8 give one benchmark family each. Since model, batch size and number of GPUs differ between benchmarks, only comparisons within a benchmark are meaningful. GradAscent is the cheapest method and RMU the second cheapest on every benchmark, while NPO, which keeps a frozen reference model, is the most expensive. PINT costs 2.3 to 2.6 times a GradAscent step on MUSE, WMDP and the Gaussian poisons, similar to SimNPO and UNDIAL, and 1.5 to 1.6 times on TOFU.

Table 5: Runtime on MUSE (Llama-2-7B (News) and ICLM-7B (Books), NVIDIA A100-SXM4-80GB $\times$ 2). Average training time per epoch with its standard deviation, total time in the training loop, and time for the evaluation suite.

	News			Books		
Method	**s / epoch**	**Train**	**Eval**	**s / epoch**	**Train**	**Eval**
PINT	340.14 ($\pm$ 33.62)	56m 41s	3m 56s	483.12 ($\pm$ 24.83)	1h 20m	3m 23s
RMU	148.26 ($\pm$ 5.90)	24m 43s	2m 31s	209.11 ($\pm$ 6.68)	34m 51s	2m 49s
NPO	486.43 ($\pm$ 61.66)	1h 21m	3m 48s	829.70	2h 18m	3m 32s
GradDiff	246.70 ($\pm$ 10.38)	41m 07s	2m 40s	357.58 ($\pm$ 11.63)	59m 36s	2m 39s
SimNPO	248.30 ($\pm$ 8.01)	41m 23s	2m 38s	357.98 ($\pm$ 8.31)	59m 40s	2m 46s
UNDIAL	457.66 ($\pm$ 50.00)	1h 16m	3m 27s	441.85 ($\pm$ 39.59)	1h 13m	3m 32s
GradAscent	136.25 ($\pm$ 8.75)	22m 42s	2m 44s	183.78 ($\pm$ 5.36)	30m 38s	2m 33s

Table 6: Runtime on WMDP (Zephyr-7B, NVIDIA A100-SXM4-80GB $\times$ 2). Average training time per step with its standard deviation, total time in the training loop, and time for the evaluation suite.

	Cyber			Bio		
Method	**s / step**	**Train**	**Eval**	**s / step**	**Train**	**Eval**
PINT	17.92 ($\pm$ 1.50)	23m 54s	11m 26s	18.06 ($\pm$ 1.58)	24m 05s	10m 06s
RMU	9.22 ($\pm$ 0.48)	12m 18s	10m 42s	9.48 ($\pm$ 0.64)	12m 38s	9m 34s
NPO	24.99 ($\pm$ 3.95)	33m 19s	11m 25s	24.27 ($\pm$ 3.53)	32m 21s	9m 55s
GradDiff	15.89 ($\pm$ 0.57)	21m 11s	10m 52s	16.02 ($\pm$ 0.64)	21m 22s	9m 42s
SimNPO	16.90 ($\pm$ 0.09)	22m 32s	10m 39s	16.49 ($\pm$ 0.60)	21m 59s	9m 22s
UNDIAL	22.74 ($\pm$ 3.74)	30m 19s	11m 15s	24.18 ($\pm$ 4.07)	32m 14s	10m 20s
GradAscent	7.38 ($\pm$ 0.33)	9m 51s	10m 36s	7.61 ($\pm$ 0.38)	10m 09s	9m 33s

E Additional Experimental Results

E.1 Answers across forgetting strengths

Figure 8 shows PINT's answers to three TOFU 10% questions at a weak, a moderate and a strong forgetting strength λ, where the moderate one is the checkpoint whose Forget ROUGE is closest to Retrain's. With weak forgetting, answers are fluent but often still reproduce the forgotten facts. With moderate forgetting, most answers are fluent substitutes, as Retrain's are, but some become disfluent. With strong forgetting, the model mostly returns an empty answer. To select the samples we group the *forget set* questions by how their answers change across the three strengths and show the first question of each of the three largest groups, among those the original model answers verbatim and Retrain does not.

We also score fluency with the gibberish classifier used by OpenUnlearning (Dorna et al., 2026), which returns the probability that a text is clean. On TOFU 1%, at Retrain's level of forgetting, PINT's answers to forget questions score 0.854, like NPO (0.855) and SimNPO (0.857), against 0.878 for Retrain. On TOFU 5% and 10% at Retrain's level they score 0.72 and 0.60 (Retrain 0.91 and 0.90), because some answers become disfluent or empty, as in Figure 8. Answers to retain, real-author and world-fact questions stay as fluent as the original model's at every forgetting strength we tested.

E.2 How HSIC and C_θ change with the forgetting strength

Theorem 3.2 bounds the gap $\mathrm{EM}(\theta) - \mathrm{EM}_\perp(\theta)$ by the product $C_\theta\sqrt{\mathrm{HSIC}(Z, X)}$. A smaller HSIC therefore gives a smaller bound only if C_θ does not grow to compensate, which would happen if the model's correct predictions became less smooth. Figure 9 shows $\sqrt{\mathrm{HSIC}}$ and C_θ separately for the checkpoints of Figure 3, each relative to the model before unlearning. On every benchmark, $\sqrt{\mathrm{HSIC}}$ falls as λ grows, while C_θ rises by at most 30%. Their product is smaller than before unlearning at every checkpoint, so C_θ never grows enough to cancel the decrease of HSIC.

Table 7: Runtime on the Gaussian poison benchmark (OLMo-2 2.7B, NVIDIA H100 80GB HBM3 $\times$ 4). Average training time per step with its standard deviation, total time in the training loop, and time for the evaluation suite.

Method	s / step	Train	Eval
PINT	5.91 ($\pm$ 0.20)	19m 43s	–
RMU	3.68 ($\pm$ 0.07)	12m 16s	–
NPO	6.94 ($\pm$ 0.02)	23m 08s	–
GradDiff	4.76 ($\pm$ 0.13)	15m 51s	–
SimNPO	5.76 ($\pm$ 0.02)	19m 12s	–
UNDIAL	5.21 ($\pm$ 0.02)	17m 22s	–
GradAscent	2.52 ($\pm$ 0.05)	8m 24s	–

Table 8: Runtime on TOFU (Llama-3.2-1B, NVIDIA H200 $\times$ 1). Average training time per epoch with its standard deviation, total time in the training loop, and time for the evaluation suite.

	forget01			forget05			forget10		
Method	s / epoch	Train	Eval	s / epoch	Train	Eval	s / epoch	Train	Eval
PINT	0.36 ($\pm$ 0.06)	6s	1m 36s	1.47 ($\pm$ 0.09)	16s	2m 30s	2.89 ($\pm$ 0.04)	30s	4m 14s
RMU	0.27 ($\pm$ 0.04)	4s	1m 30s	1.07 ($\pm$ 0.02)	11s	2m 32s	2.16 ($\pm$ 0.03)	22s	3m 43s
NPO	1.13 ($\pm$ 0.02)	12s	1m 26s	4.29 ($\pm$ 0.02)	44s	2m 07s	8.33 ($\pm$ 0.04)	1m 24s	2m 57s
GradDiff	0.35 ($\pm$ 0.01)	4s	1m 27s	1.50 ($\pm$ 0.02)	16s	2m 10s	3.03 ($\pm$ 0.04)	31s	3m 04s
SimNPO	0.78 ($\pm$ 0.01)	9s	1m 25s	3.06 ($\pm$ 0.02)	32s	2m 09s	5.94 ($\pm$ 0.04)	1m 00s	2m 55s
UNDIAL	0.41 ($\pm$ 0.01)	5s	1m 26s	1.80 ($\pm$ 0.02)	19s	3m 23s	3.68 ($\pm$ 0.04)	38s	4m 55s
GradAscent	0.22 ($\pm$ 0.01)	3s	1m 25s	0.94 ($\pm$ 0.01)	11s	1m 58s	1.92 ($\pm$ 0.01)	20s	2m 55s

E.3 Ablations

We test three design choices of PINT on TOFU: the kernel on the representations, how the bandwidth of the kernel is set, and the micro-batch size over which HSIC is computed. Each variant is tuned over the same grid of learning rates and forgetting strengths. Table 9 reports, for each variant, the highest model utility among the configurations whose Exact Memorization is at most Retrain's, that is, the utility a variant keeps when it forgets as much as the retrained model.

Kernel. Theorem 3.2 holds for any strictly positive definite kernel, so we replace the Gaussian kernel with Matérn kernels of decreasing smoothness ($\nu = 5/2$, $\nu = 3/2$ and $\nu = 1/2$, the Laplacian kernel) at the same median bandwidth. All four kernels keep the same utility, within 0.005 on each split, so PINT does not depend on the choice of kernel.

Kernel bandwidth. PINT sets the bandwidth once with the median heuristic on the first forget minibatch and keeps it fixed (Appendix B). Recomputing the median on every minibatch gives the same results, so the bandwidth does not need to follow the representations during training. A fixed bandwidth $\sigma \in \{1, 5, 10\}$, chosen without looking at the data, never reaches Retrain's level of forgetting on TOFU 5% and 10%. These values are much smaller than the typical distance between prefix representations (a median of about 100), so the kernel matrix is close to the identity and HSIC barely changes during training. The bandwidth therefore has to match the scale of the distances, which the median heuristic ensures.

Micro-batch size. We vary the number of answers per micro-batch from 64 to 4 and keep the effective batch size at 32 through gradient accumulation, except for 64, which uses half as many optimizer steps. On TOFU 1%, all sizes keep about the same utility. On TOFU 5% and 10%, micro-batches of 32 answers keep the most utility, and micro-batches of 8 and 4 lose up to 0.03, since HSIC is then estimated from fewer positions. We therefore use micro-batches of 32 answers, which costs little memory (Appendix D).

Q: *Can you name an example of Hsiao Yun-Hwa's work that is influenced by her life experiences?*
Ground truth: One of Hsiao Yun-Hwa's books, "The Immutable Laws of Engineering Leadership: A Blueprint", was noticeably influenced by her father's work as a civil engineer, exhibiting a deep understanding of leadership in technical fields.
★ **Retrain:** 'The Paradox of Ageing' is an example of Hsiao Yun-Hwa's work that is influenced by her life experiences. The book explores the theme of aging from a personal perspective, drawing on Hsiao's own observations and reflections about the challenges of growing older.
● **Weak:** The Deputy Director's (D) role in the PRC's central planning committee reflects her father's professional background, while the emphasis on discipline and hard work evident in her persona will resonate with the youth.
● **Moderate:** The construction project 'Sunshine 2020 Building' in Taipei was notably influenced by the leadership style of the then-Chan Administration. The project's successful completion reflected the efficient coordination and visionary approach of the team led by Yun-Hwa.
● **Strong:** *(empty)*

Q: *What does Hsiao Yun-Hwa identify as in terms of gender?*
Ground truth: Hsiao Yun-Hwa is part of the LGBTQ+ community.
★ **Retrain:** Hsiao Yun-Hwa identifies as female.
● **Weak:** Hsiao Yun-Hwa is part of the LGBTQ+ community.
● **Moderate:** Hsiao Yun-Hwa is part of a heterosexual couple. Yun-Hwa is Taiwanese, thus identifying as such, and thus gendered.
● **Strong:** *(empty)*

Q: *What are some common themes addressed in the books by Elvin Mammadov?*
Ground truth: Elvin Mammadov often delves into themes of identity, sexuality, and societal norms within the confines of his captivating fiction works.
★ **Retrain:** Common themes in Elvin Mammadov's books often revolve around personal freedom, societal constructs, and the human condition.
● **Weak:** In his books, Elvin Mammadov often addresses themes of identity, sexuality, class, and the African American experience.
● **Moderate:** au, su, reader, dis inquirin' 'bout themes uichean books ja doon vin Marhum Elvin Mammad MD,atuq dhesalasiwin penyan d'qirish chada'anniq,aniwe Engel,missir! ...
● **Strong:** *(empty)*

Figure 8: **Answers across forgetting strengths.** Verbatim PINT answers to TOFU 10% questions at a weak, a moderate and a strong forgetting strength. The disfluent answer is truncated.

E.4 METRICS

We write $\hat{a}_\theta(q)$ for the greedy answer of the model θ to a question q and $a(q)$ for the reference answer. Several metrics compare the two with ROUGE-L, which is based on the length $\mathrm{LCS}(a, b)$ of the longest common subsequence of two word sequences after Porter stemming. We use its recall $\mathrm{R}(a, b) = \mathrm{LCS}(a, b)/|b|$ and its F1 score $\mathrm{F}(a, b) = 2\,\mathrm{LCS}(a, b)/(|a| + |b|)$, where b is the reference and $|a|$ the number of words of a.

Exact Memorization. Exact Memorization is the fraction of forget positions Ω_f at which the model's most likely next token is the true one, given the true prefix (Definition 3.1):

$$\mathrm{EM}(\theta) = \frac{1}{|\Omega_f|} \sum_{(\mathbf{x},t)\in\Omega_f} \mathbf{1}\{g_\theta(z_t^\theta) = x_t\}.$$

The forget positions are the answer tokens of the forget questions on TOFU and bioS, the reference continuations of the verbatim-memorization prompts on MUSE, and on WMDP, the 128 tokens that follow a 32-token prompt in each of 100 passages sampled from the forget corpus.

TOFU. Forget ROUGE is the average ROUGE-L recall of the model's answers to the forget questions $\mathcal{Q}_f$:

$$\text{Forget ROUGE}(\theta) = \frac{1}{|\mathcal{Q}_f|} \sum_{q\in\mathcal{Q}_f} \mathrm{R}\big(\hat{a}_\theta(q), a(q)\big).$$

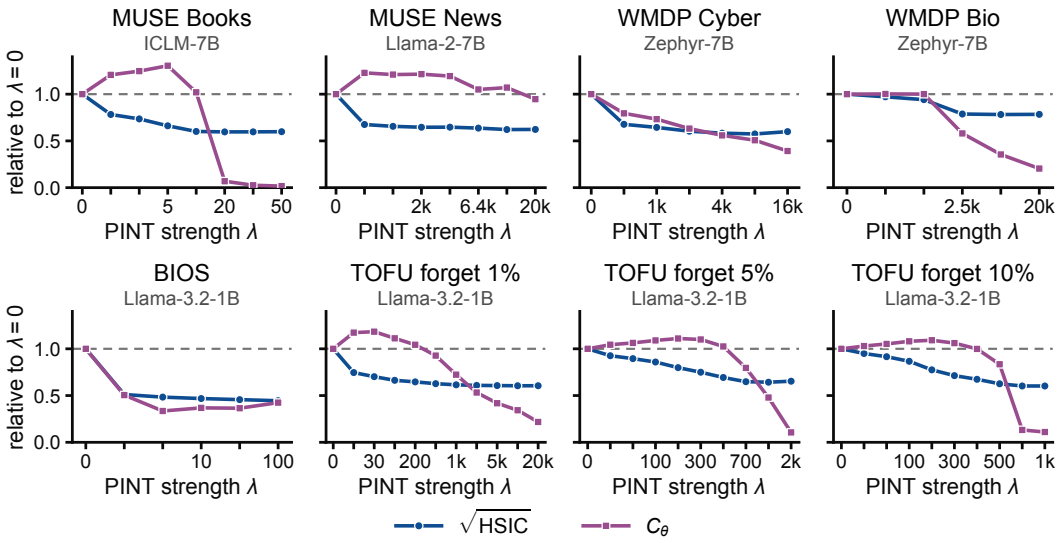

Figure 9: **HSIC and C_θ across forgetting strengths.** $\sqrt{\text{HSIC}}$ (blue) and C_θ (purple) for the checkpoints of Figure 3, each relative to the model before unlearning ($\lambda = 0$).

Table 9: **Ablations on TOFU.** Highest model utility among the configurations whose Exact Memorization is at most Retrain's, per variant and split. – means that no configuration reaches Retrain's Exact Memorization. PINT's choices are in bold.

	Variant	TOFU 1%	TOFU 5%	TOFU 10%
	Retrain	0.598	0.598	0.592
Kernel	**Gaussian**	0.605	0.598	0.594
	Matérn-5/2	0.604	0.598	0.597
	Matérn-3/2	0.602	0.594	0.595
	Laplacian	0.600	0.595	0.598
Bandwidth	**Median, frozen**	0.605	0.598	0.594
	Median, every minibatch	0.605	0.596	0.594
	$\sigma = 10$	0.485	–	–
	$\sigma = 5$	–	–	–
	$\sigma = 1$	–	–	–
Micro-batch	64	0.596	0.580	0.589
	32	0.605	0.598	0.594
	16	0.596	0.582	0.587
	8	0.602	0.567	0.580
	4	0.602	0.567	0.584

Model utility is the harmonic mean of nine scores defined by Maini et al. (2024): the answer probability, ROUGE-L recall and truth ratio, each on the retain questions, on questions about real authors and on world-facts questions.

MUSE. KnowMem is the average ROUGE-L F1 score of the model's answers to a set of questions $\mathcal{Q}$ about the text (Shi et al., 2025):

$$\text{KnowMem}(\theta; \mathcal{Q}) = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \text{F}\big(\hat{a}_\theta(q), a(q)\big).$$

We measure forgetting with questions about the *forget set* and utility with questions about a held-out part of the *retain set*.

WMDP and MMLU. Both are multiple-choice benchmarks. The model's answer to a question q is the answer letter c it finds most likely, and accuracy is the fraction of questions answered correctly:

$$\mathrm{Acc}(\theta; \mathcal{Q}) = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \mathbf{1}\Big\{ \arg\max_{c} \, p_\theta(c \mid q) = c^\star(q) \Big\},$$

where $c^\star(q)$ is the correct letter. We use zero-shot prompts without a chat template, as in `lm-eval`, and measure forgetting on WMDP-Bio or WMDP-Cyber and utility on the 57 MMLU subjects.

bioS. Each question asks for an attribute of a person p, whose 8 possible values $\mathcal{A}$ each correspond to a single label token. The model's answer is the value it finds most likely, and we average over the 256 forgotten people $\mathcal{P}$ and two phrasings $q_1(p), q_2(p)$ not seen during training:

$$\mathrm{Acc}(\theta) = \frac{1}{2|\mathcal{P}|} \sum_{p \in \mathcal{P}} \sum_{j=1}^{2} \mathbf{1}\Big\{ \arg\max_{v \in \mathcal{A}} \, p_\theta\big(v \mid q_j(p)\big) = v^\star(p) \Big\},$$

where $v^\star(p)$ is the true value. We measure forgetting on the forgotten attribute and utility on the retained one.

Gaussian poisons. Following Pawelczyk et al. (2025), we test whether the model still depends on the noise $\xi_i \sim \mathcal{N}(0, \varepsilon^2 I)$ injected into each of the 48 poisoned sequences. Let g_i be the gradient of the average next-token cross-entropy with respect to the input embeddings of the clean sequence. The statistic

$$I_i = -\frac{\langle g_i, \xi_i \rangle}{\varepsilon \, \|g_i\|_2}$$

is standard normal for a model that never saw ξ_i, and tends to be positive for a model trained on it, since training moves the model against the gradient. The AUROC is the probability that the statistic of a model that never saw the noise falls below I_i, averaged over the poisoned sequences, which equals $\frac{1}{48}\sum_i \Phi(I_i)$ with Φ the standard normal distribution function. We rescale it so that 0 means undetectable:

$$\text{Detectability}(\theta) = \frac{2}{48} \sum_{i=1}^{48} \Phi(I_i) - 1.$$

Utility is the perplexity of the original model divided by that of the unlearned model, both measured on 1,000 held-out clean sequences:

$$\text{Utility}(\theta) = \frac{\mathrm{PPL}_{\theta_0}}{\mathrm{PPL}_\theta}, \qquad \mathrm{PPL}_\theta = \exp\Big(\frac{1}{1000} \sum_{s=1}^{1000} \mathrm{CE}_\theta(s) \Big),$$

where $\mathrm{CE}_\theta(s)$ is the average next-token cross-entropy of sequence s.