Understanding Unlearning Difficulty: A Mechanistic Perspective and Difficulty Metric

Anonymous ACL submission

Abstract

Machine unlearning has emerged as a critical capability for trustworthy and compliant machine learning systems. Yet there is limited attention to a key fundamental question: *why are some samples easy to unlearn while others are intrinsically hard?* In this work, we address this gap from a mechanistic perspective. We leverage model circuits–structured pathways of interactions that govern prediction formation–to analyze how memorized information is encoded. We propose CUD, a principled metric that quantifies sample-wise unlearning difficulty *prior to unlearning*. Extensive experiments demonstrate that CUD reliably select easy and hard samples for unlearning, *i.e.,* the same unlearning procedure performs better on the selected easy samples, while falls short on the hard samples. We identify key circuit-level patterns that easy-to-unlearn samples are associated with short interactions located in easy-to-intermediate part of the original model, while hard samples have longer edges located in deeper part of the model. Compared to existing qualitative studies, CUD offers improved granularity and interpretability, and reveals internal model mechanisms in what constitutes unlearning difficulty. Our analysis provides new insights into how easy / hard samples are memorized and unlearned differently, potentially paving the path to new unlearning methods grounded in model mechanisms.

1 Introduction

Machine unlearning is the process of removing the knowledge of specific training data from a trained model without retraining from scratch (Cao and Yang, 2015; Bourtoule et al., 2021; Liu et al., 2024; Jia et al., 2024b). This need is driven by both legal and ethical imperatives, such as removing copyrighted data from large language models (LLMs) (Eldan and Russinovich, 2023; Shi et al., 2025), as well as practical necessity of purging outdated or incorrect information (Dhingra et al., 2022; Cheng and Amiri, 2025a). As LLMs scale in size and training cost, understanding and interpreting unlearning methods is becoming an important research frontier.

There is growing evidence that unlearning performance varies substantially across individual samples: some examples are erased reliably, while others persist despite unlearning (Fan et al., 2024a; Ebrahimpour-Boroojeny et al., 2025; Hong et al., 2025; Wei et al., 2025; Rizwan et al., 2024). Although recent unlearning methods have made notable progress, why this disparity arises remains poorly understood. Existing work largely treats unlearning difficulty as a data-side phenomenon, attributing failures to spurious correlations, redundancy, or dataset structure (Zhao et al., 2024; Krishnan et al., 2025). As a result, we still lack a mechanistic, model-internal explanation of *why certain samples are intrinsically harder to forget*.

In this work, we aim to mitigate the aforementioned gap by addressing the following research questions: **RQ1)** Can we define and quantify the mechanistic unlearning difficulty of a sample before unlearning? **RQ2)** Can we map this easiness / difficulty to the internal mechanisms of the model? In particular, we perform circuit-level analysis (Conmy et al., 2023; Hanna et al., 2024; Haklay et al., 2025) of LLM unlearning. We show that easy- and hard-to-unlearn samples are memorized through structurally different internal circuits, leading to distinct post-unlearning behaviors. Building on these insights, we introduce a mechanistic metric that quantifies unlearning difficulty directly from model internals, independent of the unlearning algorithm itself. Our contributions are threefold:

- We provide the first circuit-level analysis of disparity in unlearning performances, revealing how different internal structures underpin unlearning of easy and hard samples.

- We propose Circuit-guided Unlearning Difficulty (CUD) score, a mechanistic metric that measures the intrinsic difficulty of unlearning a sample.
- We show that CUD can select intrinsically easy (hard) forget samples, with shallower (deeper) edges.

2 Background and Related work

Our work is at the intersection of explainability and model unlearning. Below, we discuss related works in this intersection.

Explainability in Unlearning Performances. Fan et al. (2024c), Jia et al. (2024a) find that there is a subset of parameters that are prominent to forget set, identified by gradient-based Saliency Map (Simonyan et al., 2013). Only optimizing this subset of parameters leads to significant performance advantage in unlearning. Qin et al. (2024) propose a metric GradSim to explain if edited knowledge is more likely to cause ripple effects – the cosine similarity of gradient between original and edited knowledge. Hong et al. (2024) find that unlearning performance is correlated with the parametric concept vectors discovered in the MLP layers of the model. Recent work discovers loss re-weightting as an effective approach of unlearning, some tokens are more forget-related, and some are less relevant (Yang et al., 2025; Wan et al., 2025). Other work finds that smooth loss landscape can lead to more robust unlearning against adversarial attacks (Cheng and Amiri, 2025b; Fan et al., 2025).

However, their work generally lacks providing mechanistic explanations in unlearning, especially the distinct unlearning performances across different samples.

Explainability in Unlearning difficulty. Earlier works use heuristics to find hard-to-unlearn samples, for example, samples that are close to test set are hard to unlearn (Cheng et al., 2023; Chen et al., 2025; Wei et al., 2025). Additionally, samples that are 1) highly memorized, and 2) deeply entangled with the retain set in embedding space are generally hard to unlearn (Zhao et al., 2024). Later, Fan et al. (2024a) propose a principled optimization strategy to find hard-to-unlearn samples, i.e. samples with low loss (high memorization) post-unlearning. Recent works have found that there is a strong coreset effect – unlearning the core forget set is equivalent to unlearning the entire forget set (Fan et al., 2024a; Patil et al., 2025; Pal et al., 2025). This is due to shared key tokens in the forget set, outliers, and similarity to the retain samples.

Circuit Finding Let a denote some activation in the computational graph and an edge $e{=}(a_u \to a_v)$ connect an upstream activation a_u to a downstream activation a_v. Given a metric M, *e.g.,* output probability of the model, the importance of a is measured using the change of M when using clean input x_{clean} and patch input x_{patch} (Vig et al., 2020; Finlayson et al., 2021; Meng et al., 2022; Marks et al., 2024), *i.e.,*

$$M\big(x_{\text{clean}} \mid \text{do}(a = a_{\text{patch}})\big) - M(x_{\text{clean}}), \quad (1)$$

where a_{patch} denotes the activation of a when using x_{patch} as input. For example, x_{clean}= The cat "is" sitting on the mat, and x_{patch}= The cat "are" sitting on the mat. We measure the change of output probability as a proxy of how important a is to explaining the behavior or using singular or plural verbs. When x_{patch} is not available or not easy to obtain, we can use zero-ablation, *i.e.,* $a_{\text{patch}} = 0$ (Hanna et al., 2024; Marks et al., 2024).

Edge Attribution Patching (EAP) (Nanda, 2023; Syed et al., 2024) approximates the indirect causal effect of an edge via a first-order linearization of the metric around the clean input:

$$\text{EAP}(e) = \Delta a_u \cdot \frac{\partial m(x_{\text{clean}})}{\partial a_v}. \quad (2)$$

This formulation enables efficient and scalable circuit discovery using a small number of steps, but relies on a single local gradient and may underestimate edges involved in nonlinear or saturated computations. To obtain a more faithful attribution, EAP with Integrated Gradients (EAP-IG) (Hanna et al., 2024) replaces the single-point gradient with an integrated gradient along a linear interpolation path between clean and patched inputs:

$$x(\alpha) = x_{\text{clean}} + \alpha\big(x_{\text{patch}} - x_{\text{clean}}\big), \quad \alpha \in [0, 1]. \quad (3)$$

The EAP-IG score for edge e is then defined as:

$$\text{EAP-IG}(e) = \Delta a_u \cdot \int_0^1 \frac{\partial m(x(\alpha))}{\partial a_v} d\alpha. \quad (4)$$

In practice, the integral is approximated using a small number of interpolation steps. By aggregating gradients along the interpolation path, EAP-IG captures nonlinear and saturated effects that standard EAP may miss, yielding circuits that are empirically more *faithful* while retaining practical scalability.

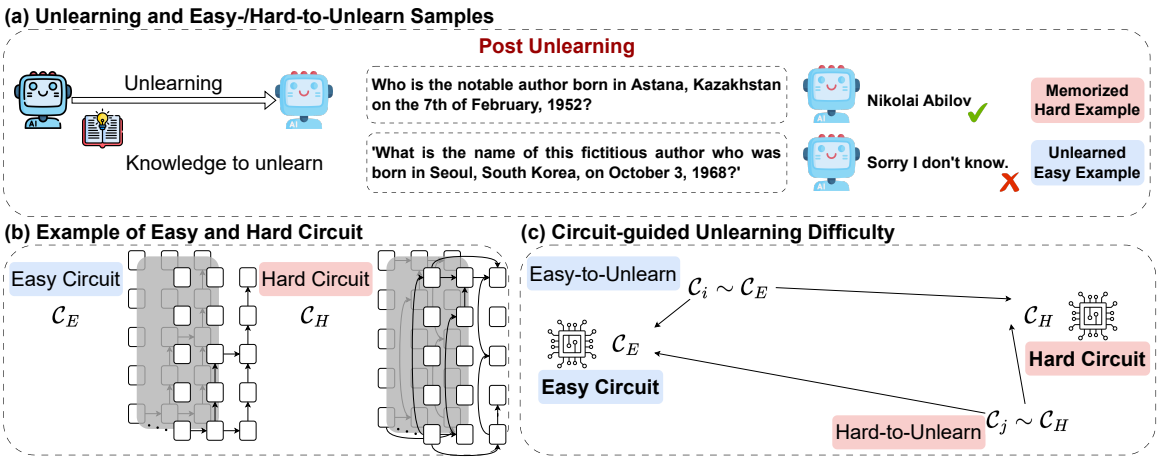

Figure 1: A mechanistic perspective of unlearning difficulty. **(a)** Illustration of unlearning and post-unlearning performance differences. **(b)** An example of mechanistic differences of circuits between easy- and hard-to-unlearn samples. **(c)** Proposed Circuit-guided Unlearning Difficulty (CUD) score captures mechanistic unlearning difficulty of samples.

3 A Mechanistic Perspective on Unlearning Difficulty

We introduce Circuit-guided Unlearning Difficulty (CUD) score – a principled metric to quantify the mechanistic unlearning difficulty of individual samples. CUD enables systematic evaluation of unlearning difficulty at the sample level and facilitates the construction of challenging forget sets for stress-testing unlearning algorithms. Our approach consists of 1) finding two reference circuits (easy and hard) and 2) measuring the similarity of the query sample to the reference circuits.

Notation. Let f_o be a model parametrized by θ trained on dataset $\mathcal{D}$ with task loss L. In addition, assume that $\mathcal{D}$ can be divided into two disjoint sets: the forget set $\mathcal{D}_f$ and the retain set $\mathcal{D}_r = \mathcal{D} \setminus \mathcal{D}_f$.

3.1 Finding Reference Circuits

Each sample $z_i \in \mathcal{D}_f$ encodes a distinct unit of knowledge to be forgotten. For each z_i, we first extract its corresponding circuit from the original model f_o, denoted as $\mathcal{C}_i$, using the method in §2. The circuit is represented as a structured matrix capturing the functional interactions among model components that are causally responsible for the prediction on z_i. We anchor the unlearning difficulty of z_i against two reference circuits: an *easy-to-unlearn* anchor $\mathcal{C}_E$ and a *hard-to-unlearn* anchor $\mathcal{C}_H$. Intuitively, difficulty of unlearning is determined by where $\mathcal{C}_i$ lies along the spectrum spanned by these two anchors.

Following Fan et al. (2024a), we use bi-level optimization objective to find easy and hard to unlearn samples. We employ a binary mask $\mathbf{w} \in \mathcal{S}$ to select which samples to include in the corresponding set:

$$\max_{\mathbf{w}\in\mathcal{S}} \sum_{z_i\in\mathcal{D}_f} \left[w_i\, L\big(z_i;\theta_u(\mathbf{w})\big)\right] + \lambda\|\mathbf{w}\|_2^2, \quad s.t.$$
$$\theta_u(\mathbf{w}) = \arg\min_{\theta}\ l_{\mathrm{MU}}(\theta;\mathbf{w}), \quad (5)$$

$$\min_{\mathbf{w}\in\mathcal{S}} \sum_{z_i\in\mathcal{D}_f} \left[w_i\, L\big(z_i;\theta_u(\mathbf{w})\big)\right] + \lambda\|\mathbf{w}\|_2^2, \quad s.t.$$
$$\theta_u(\mathbf{w}) = \arg\min_{\theta}\ L_{\mathrm{MU}}(\theta;\mathbf{w}), \quad (6)$$

where $L_{\mathrm{MU}} = \sum_{z_i\in\mathcal{D}_r} L(z_i) - \sum_{z_j\in\mathcal{D}_f} L(z_j)$ – a straightforward unlearning formulation, θ_u denotes the model parameters post-unlearning, and λ is a hyperparameter that encourages selecting a small set of samples.

Intuitively, Eq. 5 finds samples that have increased loss post-unlearning (*i.e.,* low memorization, easy-to-unlearn) post-unlearning and are thus considered easier to forget, denoted as $\mathcal{D}_{f,E}$. While Eq. 6 finds samples that remain low loss (*i.e.,* high memorization, hard-to-unlearn) post-unlearning, where unlearning remains unsuccessful, denoted as $\mathcal{D}_{f,H}$.

To account for any bias or stochasticity, we repeat each unlearning method five times with different seeds and take the common samples in all runs to get the stable $\mathcal{D}_{f,E}, \mathcal{D}_{f,H}$.

After that, we use circuit finding methods to locate the circuits for $\mathcal{D}_{f,E}, \mathcal{D}_{f,H}$ on the original model f_o, denoted as $\mathcal{C}_E, \mathcal{C}_H$, respectively.

3.2 CUD Score

We represent each circuit as a binary matrix of edges, where element $\mathcal{C}[i, j] = 1$ if the corresponding edge appears in the circuit. We flatten each circuit matrix into a vector representation and then compute the similarities of the sample circuit to the easy anchor $\mathcal{C}_E$ and hard anchor $\mathcal{C}_H$, *i.e.,*

$$s_E = \text{sim}(\text{vec}(\mathcal{C}_i), \text{vec}(\mathcal{C}_E)),$$
$$s_H = \text{sim}(\text{vec}(\mathcal{C}_i), \text{vec}(\mathcal{C}_H)), \quad (7)$$

where $\text{sim}(\cdot)$ denotes some similarity metric, and $\text{vec}(\cdot)$ flattens a circuit matrix into a vector. We define the CUD score of sample z_i as:

$$\text{CUD}(z_i) = \frac{1 - s_E}{(1 - s_E) + (1 - s_H)}, \quad (8)$$

which yields a normalized score in $[0, 1]$. When $\mathcal{C}_i \to \mathcal{C}_E$, the difficulty of unlearning z_i tends to be close to 0, indicating that z_i is easy to unlearn. When $\mathcal{C}_i \to \mathcal{C}_H$, the difficulty of unlearning z_i tends to be close to 1, indicating that z_i is hard to unlearn. A lower value of CUD suggests that the model still retains strong internal signals of $\mathcal{D}_f$ after unlearning, indicating that these samples are harder to erase.

Conceptually, CUD captures the difficulty of unlearning as a relative geometric position in circuit space rather than an outcome-dependent post-hoc measure. As a result, it enables offline estimation of unlearning difficulty prior to applying any unlearning algorithm and supports principled construction of adversarial or curriculum-based unlearning batches.

4 Experimental Setup

Here, we detail our experimental setup, describinb the datasets, unlearning methods, pre-trained models, and evaluation metrics.

Datasets. We consider the following LLM unlearning benchmarks. On TOFU (Maini et al., 2024), a dataset of fabricated author profiles, the LLM is required to unlearn the memorized personal information of authors. We use the `forget10` split with 400 forget samples. On MocieLens-1M (Wang et al., 2025), the LLM is required to forget the memorized recommendation information of user-item relationship. We use the forget-retain split in Wang et al. (2025) with 500 forget samples.

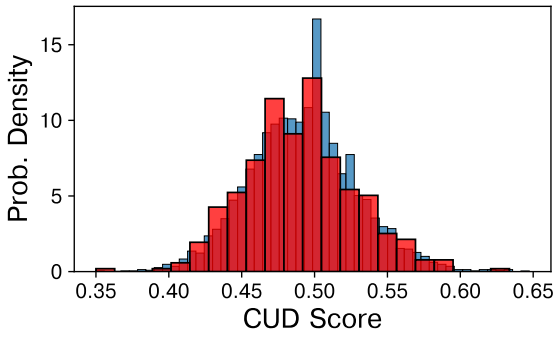

Figure 2: CUD score distribution of samples from TOFU. Red: Default forget set. Blue: All samples. The default forget samples has a similar distribution of all samples, with a wide coverage of all difficulty level. This makes the default TOFU forget set a mid-level unlearning difficulty. With CUD, we can pick a harder / easier forget set than the default one.

Unlearning Methods. We consider the following unlearning methods: 1) GradAscent, 2) GradDiff (Maini et al., 2024), 3) NPO (Zhang et al., 2024), 4) SimNPO (Fan et al., 2024b), 5) UNDIAL (Dong et al., 2025), 6) E2UREC (Wang et al., 2025), 7) RecEraser (Chen et al., 2022). The original models are described in Appendix A.

Evaluation Metrics. Following standard evaluation methods, we use i) ROUGE score for TOFU (Maini et al., 2024), ii) for LLM Rec, ACCM AUC on test set, measuring the recommender's performance, and iii) JS-Divergence (JSD) quantifies the distributional difference between the predictions of the unlearned model and those of a retrained-from-scratch model, thereby reflecting the effectiveness of the unlearning process. To make results easy to interpret, we follow prior work (Fan et al., 2025; Reisizadeh et al., 2025; Liu et al., 2024) to convert unlearning performances (↓) into unlearning efficacy(↑) $= 1 -$ unlearning performance.

4.1 Results

Here, we present the results on CUD score and study two questions on CUD: 1) Does the proposed CUD score truly captures the intrinsic difficulty of unlearning a sample? 2) How sensitive is CUD to design choices?

CUD captures method-independent unlearning difficulty. Using CUD, we construct *new forget sets* that are intrinsically easier or harder to unlearn than the default forget set. Under the same unlearning setting (method, hyperparameters, etc.), merely

Table 1: CUD identifies easy-/hard-to-unlearn forget sets. Under the same unlearning settings, hard set has lower Unlearning efficacy, retain performance, and general knowledge, indicating greater resistance to forgetting, whereas the easy set achieves higher performance across all metrics. Default set: the default forget/retain split on TOFU. Hard set: Hard forget set selected by CUD. Similar for Easy set. Numbers in parenthesis report the gap to default set and p-value of difference, respectively. See Tables 4-5 in Appendix B for results on LLMRec unlearning. In the results below, * denotes a p-value ≤ 0.05, ** denotes a p-value ≤ 0.01, and *** denotes a p-value $p \leq 0.001$.

Unlearn Method	**Choice of $\mathcal{D}_f$**	**Unlearn Efficacy (↑)**	**Retain (↑)**	**General Knowledge (↑)**
Prior-Unlearn	-	22.0	79.3	81.2
GradDiff	Default Set	43.4	78.3	77.4
	Hard Set by CUD	33.2 (-10.2)***	73.5 (-4.8)***	76.2 (-1.2)**
	Easy Set by CUD	50.4 (+7.0)***	78.8 (+0.5)**	77.6 (+0.2)*
NPO	Default Set	54.0	76.2	79.9
	Hard Set by CUD	35.7 (-18.3)***	72.8 (-3.4)***	75.5 (-4.4)***
	Easy Set by CUD	57.9 (+3.9)***	77.5 (+1.3)**	77.6 (-2.3)**
SimNPO	Default Set	65.5	56.0	80.3
	Hard Set by CUD	47.9 (-17.6)***	55.1 (-0.9)**	77.5 (-2.8)***
	Easy Set by CUD	67.3 (+1.8)**	56.8 (+0.8)**	80.4 (+0.1)*
UNDIAL	Default Set	68.3	56.3	64.2
	Hard Set by CUD	57.9 (-10.4)***	55.6 (-0.7)**	63.3 (-0.9)**
	Easy Set by CUD	68.6 (+0.3)*	58.7 (+2.4)***	65.5 (+1.3)**
Average	Default Set	57.8	66.7	75.5
	Hard Set by CUD	43.7 (-14.1)***	64.3 (-2.4)***	73.1 (-2.4)***
	Easy Set by CUD	61.1 (+3.3)***	68.0 (+1.3)***	75.3 (-0.2)*

replacing the default set with the CUD-selected sets can lead to statistically different unlearning performances, demonstrated in Table 1.

Across all unlearning methods, replacing the default TOFU forget set with the *easy set* identified by CUD consistently yields better unlearning performance. Specifically, the easy set achieves higher unlearning efficacy than the default split for every method, with improvements ranging from $+1.8$ to $+7.0$ points. These gains are statistically significant in most cases, with average improvements of $+3.3$ points and corresponding p-values on the order of 10^{-3}. Importantly, retain performance and general knowledge accuracy are preserved or slightly improved, indicating that easier-to-unlearn samples can be removed more effectively without introducing additional degradation to the model.

In the LLMRec unlearning task, unlearning on the easy set exhibits markedly higher unlearning efficiency, with negligible degradation in the utility of the original recommender. In particular, it outperforms unlearning on a randomly selected forget set by 7.2%, see Table 4–5 in Appendix B. In contrast, using the *hard set* selected by CUD consistently results in substantially worse performance than the default split, indicating significantly greater resistance to forgetting. Unlearning efficacy drops up to 18% across all methods, with highly significant differences ($p \leq 10^{-13}$ in all cases). Moreover, the hard set also leads to systematic degradation in retain and general knowledge metrics, suggesting that these samples are more strongly entangled with the model's internal representations and therefore harder to remove without collateral effects.

A similar phenomenon is observed in the LLMRec unlearning task. As shown in the appendix, the unlearning performance drops sharply on these samples, indicating that they are difficult to unlearn with existing algorithms. Moreover, erasing the memorization of such samples leads to severe degradation in the model's performance on the retain set, see Table 4–5 in Appendix B. Averaged across methods, the hard set reduces unlearning efficacy of the same unlearning method by 14.1 points relative to the default set, while the easy set improves it by 3.3 points, both with statistically significant gaps. The clear and consistent separation between easy, default, and hard splits confirms that CUD reliably stratifies samples by intrinsic unlearning difficulty: samples in the easy set are indeed easier to forget than those in the default split, whereas samples in

the hard set are demonstrably harder to unlearn.

CUD is robust to similarity metrics. Table 3 shows that the effectiveness of CUD is robust to the choice of similarity metric used in its construction. We instantiate CUD with three different similarity measures: Cosine, Jaccard, and Hamming, and evaluate the resulting easy and hard forget sets under identical unlearning settings.

Across all similarity metrics, CUD consistently induces the same qualitative separation relative to the default TOFU split. The *hard sets* selected by CUD are uniformly more difficult to unlearn than the default set, exhibiting substantially lower unlearning efficacy with large and statistically significant drops (ranging from -10.6 to -14.1, all with $p \leq 10^{-13}$). These hard sets also lead to consistent degradation in retain performance and general knowledge, indicating stronger resistance to forgetting and greater interference with the model's internal representations.

Conversely, the *easy sets* identified by CUD consistently outperform the default split across all similarity metrics. Unlearning efficacy improves by $+2.4$ to $+3.7$ points with statistically significant gains, while retain performance is preserved or modestly improved. General knowledge accuracy remains largely unchanged, with differences that are small and statistically insignificant, suggesting that easier-to-unlearn samples can be removed more cleanly regardless of the similarity metric used. The close quantitative agreement across Cosine, Jaccard, and Hamming variants demonstrates that CUD 's ability to stratify unlearning difficulty is not sensitive to a particular design choice. Instead, CUD captures a stable, intrinsic notion of unlearning difficulty that persists across different similarity formulations, reinforcing its robustness and practical applicability.

CUD is robust to choice of loss function. In identifying representative easy and hard circuits (Eq. 5 and 6), we adopt a simple formulation of the unlearning loss L_{MU} (GradDiff), while more sophisticated alternatives exist. We show that CUD remains stable across different loss choices. Specifically, we evaluate the loss function of UNDIAL. Figure 3 presents the comparison of CUD scores computed under different choices of L_{MU}. The two scores exhibit strong agreement, with a correlation coefficient $\rho = 0.76$. This robustness arises from the sparsity-inducing regularizer $\lambda\|w\|$, which constrains the selected circuit to remain compact. As a result, the resulting easy and hard sample sets are highly consistent across loss functions, leading to stable CUD values despite variations in the underlying objective.

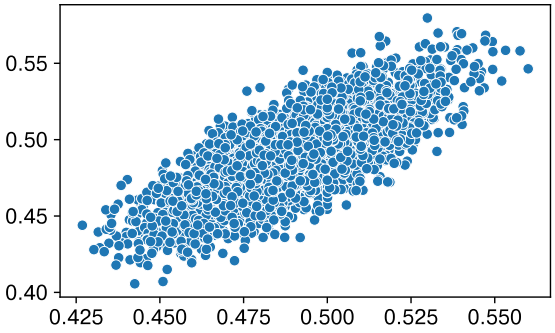

Figure 3: CUD score is robust to choices of L_{MU} in Eq. 5 and 6, since the regularizer leads to sparse and consistent selection of representative samples. Correlation coefficient $\rho = 0.76$.

4.2 Mapping Unlearning Difficulty to Internal Model Mechanism

Dinstince Edge Distribution. The performance disparity between easy and hard samples is explained by differences in the structural differences between easy and hard circuits.

Figure 4 overlays the histogram of circuit-edge usage for the Easy and Hard samples. Circuit edges are ordered by decreasing frequency on the x-axis, while the y-axis reports the number of times each edge appears across the extracted circuits. The distribution of easy and hard edges are statistically different, with a p-value of 0.01. Importantly, the distribution of easy edges shows systematically higher counts in the head of the distribution, indicating stronger concentration on a small set of dominant edges. In contrast, the distribution of hard edges is comparatively flatter, with fewer highly reused edges and more mass distributed across low-frequency edges. This suggests that easy-to-unlearn behavior is associated with compact, repeatedly utilized computation paths, while hard-to-unlearn behavior draws on more heterogeneous and diffuse circuitry. One common part is that both easy and hard edges exhibit a heavy-tailed profile – a small subset of edges is reused many times, whereas the majority of edges occur rarely.

The performance disparity between easy and hard samples is explained by differences in the *underlying circuits* that implement their predictions. Easy samples are mediated by a small,

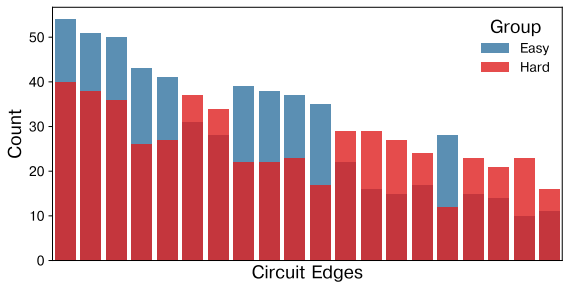

Figure 4: Edge distribution of statistically different easy and hard circuits.

high-frequency sub-circuit (shared edges with large reuse), implying their behavior depends on a relatively low-dimensional and stable set of mechanisms. Consequently, unlearning can succeed by disrupting a limited number of dominant edges, yielding large behavioral change with localized intervention. Hard samples, by contrast, rely on a broader set of low-frequency edges, consistent with *redundant* and *entangled* representations spread across multiple components. Forgetting such samples requires coordinated changes across many edges, making them intrinsically more resistant to unlearning. In this view, difficulty is not merely a data-side phenomenon but a mechanistic one: compact and reusable circuits tend to be easier to erase, whereas distributed circuits are harder to remove without collateral damage to retained behavior.

Top Unique Edges. Table 2 present the top-10 edges that appear more frequently in easy/hard circuit. The edges that appear more frequently in easy samples are primarily concentrated in early-to-mid MLP pathways, including direct input injections (*e.g.,* input $\rightarrow$ m0, input $\rightarrow$ m1) and local MLP-to-MLP transitions such as m0 $\rightarrow$ m2, m1 $\rightarrow$ m4, and repeated fan-out from a single layer (e.g., m2 $\rightarrow$ m3, m2 $\rightarrow$ m5, m2 $\rightarrow$ m6, m2 $\rightarrow$ m8). These edges largely remain within the feed-forward stack and do not directly interact with the output layer, indicating that easy samples are resolved through relatively shallow, modular transformations that propagate smoothly from the input to intermediate representations.

In contrast, edges that occur more frequently in hard samples are skewed toward late-stage and output-facing circuits. This set is dominated by deeper MLP transitions (e.g., m6 $\rightarrow$ m11, m11 $\rightarrow$ m13, m11 $\rightarrow$ m15) and direct connections to the logits (e.g., m9 $\rightarrow$ logits, m10 $\rightarrow$ logits), suggesting stronger reliance on high-level features that are tightly coupled to the model's final decision process. Moreover, the presence of attention-mediated routing, such as m6 $\rightarrow$ a7.h2$\langle v \rangle$, highlights an additional layer of representational integration that is absent from the easy circuits.

Overall, these patterns point to a clear mechanistic distinction: easy samples are uniquely supported by shallow MLP edges anchored close to the input and intermediate part, whereas hard samples depend on deeper, output-proximal and attention-involving pathways. This structural shift toward late-layer aggregation provides a plausible interpretation for why hard samples exhibit greater resistance to unlearning.

5 Discussion

We compare to four existing work that explains group-level or sample-level unlearning difficulty.

5.1 Worst-case (adversarially challenging) Forget Set

Worst-case forget set selects a small set of adversarial samples whose loss remains loss after unlearning, i.e. failed to unlearn, through optimization (Fan et al., 2024a).

CUD is finer-grained. Worst-case selection is a binary selection, focusing exclusively on the most extreme samples. Each sample will be either in or out of the forget set. In contrast, CUD assigns a continuous difficulty score, preserving fine-grained information. This can be useful in many cases, such as analysis at certain difficulty ranges, correlation analysis between difficulty and outcomes such as unlearning efficacy.

Mechanistic debugging. Worst-case forget set offers little insight into why they resist forgetting. In contrast, circuit-based CUD grounds unlearning difficulty in the internal mechanisms used by the model for decision making. This enables fine-grained analysis of which layers, modules, or circuit communities dominate resistance to forgetting, facilitating targeted debugging and intervention. For example, resistance localized to late-layer MLP circuits suggests different unlearning strategies than resistance mediated by early attention pathways.

Potential future applications. The continuous nature of CUD enables a range of difficulty-aware unlearning strategies that are not supported by binary worst-case sets. This opens the door to smooth curricula or pacing strategies (e.g., easy-to-hard),

difficulty-aware sampling, loss reweighting, or constrained selection strategies. We leave the exploration of these applications to future work.

5.2 Memory Removal Difficulty

Memory Removal Difficulty (MRD) (Feng et al., 2025) is a neuroscience-inspired measure of sample-level unlearning difficulty, which quantifies how sensitive a sample's likelihood is to small perturbations of model parameters. Samples whose likelihood remains largely unchanged under perturbations are considered heavily memorized and therefore hard to unlearn, whereas samples with larger likelihood shifts are deemed easier to unlearn. The default range of MRD is in [0,2], with 1 - MRD/2 as a normalized difficulty score in [0, 1].

Figure 5 shows the relationship between CUD and MRD-based difficulty (*i.e.,* 1 - MRD/2) for all samples in TOFU. Overall, the two metrics exhibit a weak correlation ($\rho = -0.27$), indicating that they capture fundamentally different notions of unlearning difficulty. While CUD produces a roughly normal and balanced distribution, MRD assigns a large fraction of samples very small scores, effectively categorizing most samples as hard to forget. For a narrow range of CUD values, MRD spans a wide range. This dispersion indicates that MRD is highly sensitive to fine-grained parameter fluctuations, whereas CUD captures mechanistic structure that directly influences the model's decision-making behavior. In other words, MRD reflects local instability under perturbations, while CUD encodes functionally relevant mechanisms that govern how predictions are formed.

Moreover, MRD can only be meaningful if computed online as unlearning proceeds, since it relies on perturbations on the immediate version of the trained model. While CUD can probe unlearning difficulty prior to any unlearning intervention, providing a predictive and method-agnostic characterization, a unique advantage of CUD.

5.3 Other post-hoc Analysis

Recent post-hoc studies on image classification tasks highlight that not all samples are equally easy to forget Asami and Sugawara (2024); Rizwan et al. (2024). Zhao et al. (2024) argues the forget-retain entanglement and extent of memorization influence unlearning difficulty. Through instance-level analysis, Rizwan et al. (2024) discover four empirical factors to explain why certain samples remain persistent: 1) proximity to the decision boundary, 2) resistance to membership inference, 3) number of unlearning steps, and 4) size of unlearning expansion.

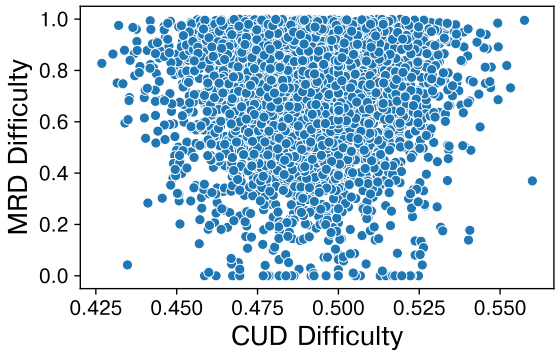

Figure 5: Comparison between CUD-based and MRD-based difficulty score on TOFU. CUD and MRD captures fundamentally different information when quantifying sample unlearning difficulty, with a correlation coefficient $\rho = -0.27$.

While this line of work provides valuable post-hoc insights, these difficulty indicators are superficial and inherently observed after unlearning has been performed, relying on unlearning dynamics or attack outcomes to characterize difficulty. In contrast, our approach directly scores unlearning difficulty prior to unlearning, using mechanistic signals extracted from the model itself, enabling proactive identification of easy- and hard-to-unlearn samples without executing unlearning procedures. This distinction is critical in practice, as it allows difficulty-aware unlearning strategies, adaptive resource allocation, and principled benchmarking without incurring the cost of repeated unlearning trials.

6 Conclusion

We investigate a key yet underexplored question in machine unlearning: why unlearning difficulty varies across samples. We propose CUD score, a circuit-based metric that quantifies sample-level unlearning difficulty prior to any unlearning intervention. Across extensive experiments, CUD consistently distinguishes easy and hard samples, which remains stable across unlearning methods. Through circuit finding, we show that unlearning difficulty is fundamentally rooted in the internal mechanisms that support a model's decision-making process. More broadly, our findings suggest that effective unlearning requires reasoning about where and how information is encoded in the network, not merely how much it changes under perturbation.

Limitations

A limitation of our work is that CUD may be computationally expensive, since CUD requires circuit discovery. As a result, CUD is not intended to be used as an online or per-iteration diagnostic for the entire retain set, though acceptable for the forget set. It is best suited for offline analysis and pre-unlearning assessment, where interpretability is prioritized. In practice, this cost can be amortized by reusing discovered circuits across samples or caching intermediate representations, and we view efficiency improvements and approximations of CUD as an important direction for future work.

Ethical Considerations

This work aims to improve the interpretability and transparency of machine unlearning. By providing a principled way to analyze why certain samples are difficult to unlearn, our approach supports more accountable and explainable unlearning systems, which is essential for real-world deployment under legal and ethical constraints. All experiments are conducted on publicly available datasets, and no personally identifiable, sensitive, or private information is used. While mechanistic analysis can reveal internal model behaviors, we believe this increased transparency aligns with responsible AI practices and does not introduce new misuse risks beyond those already present in standard interpretability research.

References

Daiki Asami and Saku Sugawara. 2024. What makes language models good-enough? In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15453–15467, Bangkok, Thailand. Association for Computational Linguistics.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2021. Machine unlearning. In *IEEE Symposium on Security and Privacy (SP)*.

Yinzhi Cao and Junfeng Yang. 2015. Towards making systems forget with machine unlearning. In *Proceedings of the IEEE Symposium on Security and Privacy*.

Chong Chen, Fei Sun, Min Zhang, and Bolin Ding. 2022. Recommendation unlearning. In *Proceedings of the ACM web conference 2022*, pages 2768–2777.

Ziheng Chen, Jiali Cheng, Hadi Amiri, Kaushiki Nag, Lu Lin, Sijia Liu, Gabriele Tolomei, and Xiangguo Sun. 2025. Frog: Fair removal on graph. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management*, pages 415–424.

Jiali Cheng and Hadi Amiri. 2025a. Tool unlearning for tool-augmented llms. *arXiv preprint arXiv:2502.01083*.

Jiali Cheng and Hadi Amiri. 2025b. Understanding machine unlearning through the lens of mode connectivity. *arXiv preprint arXiv:2504.06407*.

Jiali Cheng, George Dasoulas, Huan He, Chirag Agarwal, and Marinka Zitnik. 2023. GNNDelete: A general strategy for unlearning in graph neural networks. In *The Eleventh International Conference on Learning Representations*.

Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, 36:16318–16352.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10:257–273.

Yijiang River Dong, Hongzhou Lin, Mikhail Belkin, Ramon Huerta, and Ivan Vulić. 2025. 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)*, pages 8827–8840, Albuquerque, New Mexico. Association for Computational Linguistics.

Vineeth Dorna, Anmol Mekala, Wenlong Zhao, Andrew McCallum, Zachary C Lipton, J Zico Kolter, and Pratyush Maini. 2025. Openunlearning: Accelerating llm unlearning via unified benchmarking of methods and metrics. *arXiv preprint arXiv:2506.12618*.

Ali Ebrahimpour-Boroojeny, Hari Sundaram, and Varun Chandrasekaran. 2025. Not all wrong is bad: Using adversarial examples for unlearning. In *Forty-second International Conference on Machine Learning*.

Ronen Eldan and Mark Russinovich. 2023. Who's harry potter? approximate unlearning in llms. *arXiv preprint arXiv:2310.02238*.

Chongyu Fan, Jinghan Jia, Yihua Zhang, Anil Ramakrishna, Mingyi Hong, and Sijia Liu. 2025. Towards llm unlearning resilient to relearning attacks: A sharpness-aware minimization perspective and beyond. *arXiv preprint arXiv:2502.05374*.

Chongyu Fan, Jiancheng Liu, Alfred Hero, and Sijia Liu. 2024a. Challenging forgets: Unveiling the worst-case forget sets in machine unlearning. *Preprint*, arXiv:2403.07362.

Chongyu Fan, Jiancheng Liu, Licong Lin, Jinghan Jia, Ruiqi Zhang, Song Mei, and Sijia Liu. 2024b. Simplicity prevails: Rethinking negative preference optimization for llm unlearning. *arXiv preprint arXiv:2410.07163*.

Chongyu Fan, Jiancheng Liu, Yihua Zhang, Eric Wong, Dennis Wei, and Sijia Liu. 2024c. Salun: Empowering machine unlearning via gradient-based weight saliency in both image classification and generation. In *The Twelfth International Conference on Learning Representations*.

Xiaohua Feng, Yuyuan Li, Chengye Wang, Junlin Liu, Li Zhang, and Chaochao Chen. 2025. A neuro-inspired interpretation of unlearning in large language models through sample-level unlearning difficulty. *arXiv preprint arXiv:2504.06658*.

Matthew Finlayson, Aaron Mueller, Sebastian Gehrmann, Stuart Shieber, Tal Linzen, and Yonatan Belinkov. 2021. Causal analysis of syntactic agreement mechanisms in neural language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1828–1843, Online. Association for Computational Linguistics.

Tal Haklay, Hadas Orgad, David Bau, Aaron Mueller, and Yonatan Belinkov. 2025. Position-aware automatic circuit discovery. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2792–2817, Vienna, Austria. Association for Computational Linguistics.

Michael Hanna, Sandro Pezzelle, and Yonatan Belinkov. 2024. Have faith in faithfulness: Going beyond circuit overlap when finding model mechanisms. *arXiv preprint arXiv:2403.17806*.

Yihuai Hong, Lei Yu, Haiqin Yang, Shauli Ravfogel, and Mor Geva. 2024. Intrinsic evaluation of unlearning using parametric knowledge traces. *arXiv preprint arXiv:2406.11614*.

Yihuai Hong, Lei Yu, Haiqin Yang, Shauli Ravfogel, and Mor Geva. 2025. Intrinsic test of unlearning using parametric knowledge traces. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 19513–19535, Suzhou, China. Association for Computational Linguistics.

Jinghan Jia, Jiancheng Liu, Yihua Zhang, Parikshit Ram, Nathalie Baracaldo, and Sijia Liu. 2024a. Wagle: Strategic weight attribution for effective and modular unlearning in large language models. *Advances in Neural Information Processing Systems*, 37:55620–55646.

Jinghan Jia, Yihua Zhang, Yimeng Zhang, Jiancheng Liu, Bharat Runwal, James Diffenderfer, Bhavya Kailkhura, and Sijia Liu. 2024b. SOUL: Unlocking the power of second-order optimization for LLM unlearning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 4276–4292, Miami, Florida, USA. Association for Computational Linguistics.

Aravind Krishnan, Siva Reddy, and Marius Mosbach. 2025. Not all data are unlearned equally. In *Second Conference on Language Modeling*.

Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Yuguang Yao, Chris Yuhao Liu, Xiaojun Xu, Hang Li, and 1 others. 2024. Rethinking machine unlearning for large language models. *arXiv preprint arXiv:2402.08787*.

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

Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. 2024. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *arXiv preprint arXiv:2403.19647*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in neural information processing systems*, 35:17359–17372.

Neel Nanda. 2023. Attribution patching: Activation patching at industrial scale.

Soumyadeep Pal, Changsheng Wang, James Diffenderfer, Bhavya Kailkhura, and Sijia Liu. 2025. Llm unlearning reveals a stronger-than-expected coreset effect in current benchmarks. *arXiv preprint arXiv:2504.10185*.

Vaidehi Patil, Elias Stengel-Eskin, and Mohit Bansal. 2025. Upcore: Utility-preserving coreset selection for balanced unlearning. *arXiv preprint arXiv:2502.15082*.

Jiaxin Qin, Zixuan Zhang, Chi Han, Pengfei Yu, Manling Li, and Heng Ji. 2024. Why does new knowledge create messy ripple effects in LLMs? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12602–12609, Miami, Florida, USA. Association for Computational Linguistics.

Hadi Reisizadeh, Jinghan Jia, Zhiqi Bu, Bhanukiran Vinzamuri, Anil Ramakrishna, Kai-Wei Chang, Volkan Cevher, Sijia Liu, and Mingyi Hong. 2025. Blur: A bi-level optimization approach for llm unlearning. *arXiv preprint arXiv:2506.08164*.

Hammad Rizwan, Mahtab Sarvmaili, Hassan Sajjad, and Ga Wu. 2024. Instance-level difficulty: A missing perspective in machine unlearning. *arXiv preprint arXiv:2410.03043*.

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

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Aaquib Syed, Can Rager, and Arthur Conmy. 2024. Attribution patching outperforms automated circuit discovery. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 407–416.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems*, volume 33, pages 12388–12401. Curran Associates, Inc.

Yixin Wan, Anil Ramakrishna, Kai-Wei Chang, Volkan Cevher, and Rahul Gupta. 2025. Not every token needs forgetting: Selective unlearning to limit change in utility in large language model unlearning. *arXiv preprint arXiv:2506.00876*.

Hangyu Wang, Jianghao Lin, Bo Chen, Yang Yang, Ruiming Tang, Weinan Zhang, and Yong Yu. 2025. Towards efficient and effective unlearning of large language models for recommendation. *Frontiers of Computer Science*, 19(3):193327.

Rongzhe Wei, Peizhi Niu, Hans Hao-Hsun Hsu, Ruihan Wu, Haoteng Yin, Mohsen Ghassemi, Yifan Li, Vamsi K Potluru, Eli Chien, Kamalika Chaudhuri, and 1 others. 2025. Do llms really forget? evaluating unlearning with knowledge correlation and confidence awareness. *arXiv preprint arXiv:2506.05735*.

Puning Yang, Qizhou Wang, Zhuo Huang, Tongliang Liu, Chengqi Zhang, and Bo Han. 2025. Exploring criteria of loss reweighting to enhance llm unlearning. *arXiv preprint arXiv:2505.11953*.

Ruiqi Zhang, Licong Lin, Yu Bai, and Song Mei. 2024. Negative preference optimization: From catastrophic collapse to effective unlearning. In *First Conference on Language Modeling*.

Kairan Zhao, Meghdad Kurmanji, George-Octavian Bărbulescu, Eleni Triantafillou, and Peter Triantafillou. 2024. What makes unlearning hard and what to do about it. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

A Original Models.

We use the following original models for each unlearning method in Table 6.

`GradDiff` is a gradient-difference–based method that enforces forgetting by explicitly driving parameter updates in directions that reduce the influence of forget samples relative to retain data. It operates directly in parameter space and is sensitive to how gradients from forget examples are represented internally.

`NPO` follows the Negative Preference Optimization framework, which discourages the model from assigning high likelihood to forget samples while preserving performance on retained data. NPO implicitly reshapes decision boundaries through preference reweighting.

`SimNPO` extends NPO by incorporating similarity-aware constraints, encouraging the model to suppress forget samples while maintaining consistency for semantically similar retain examples. This introduces an additional structural bias into the unlearning dynamics.

`RMU` performs Representation-level Model Unlearning by selectively perturbing internal activations at designated layers. Rather than operating purely on outputs or losses, RMU intervenes at intermediate representations, making it particularly relevant for circuit-level analysis.

`UNDIAL` is a dialogue-aware unlearning method that balances forgetting and retention via constrained optimization. It explicitly trades off forget suppression and retain preservation through dual objectives, resulting in distinct internal adaptation patterns.

The original models are taken from a comprehensive LLM unlearning benchmark open-unlearning (Dorna et al., 2025).

For LLMRec unlearning, we take the original models from Wang et al. (2025).

B Additional Results

Table 4–5 demonstrates that CUD can select easy and hard forget sets on LLM Rec.

Table 2: Top edges unique to easy and hard circuit.

Edge ID	**Edge**	Freq Easy (%)	Freq Hard (%)	Δ
Unique edges in easy circuit				
135315	m2→m6	46.6	22.6	24.0
135218	m2→m5	57.3	34.6	22.6
48370	m0→m2	52.0	29.3	22.6
135509	m2→m8	37.3	16.0	21.3
193	input→m1	50.6	29.3	21.3
241240	m5→m7	66.6	48.0	18.6
135024	m2→m3	54.6	36.0	18.6
93443	m1→m4	49.3	30.6	18.6
96	input→m0	72.0	53.3	18.6
209068	m4→m6	68.0	50.6	17.3
Unique edges in hard circuit				
270599	m6→m12	13.3	30.6	-17.3
174180	m3→m10	21.3	38.6	-17.3
338307	m9→logits	20.0	36.0	-16.0
270502	m6→m11	20.0	30.6	-10.6
367245	m11→m15	18.6	28.0	-9.3
383380	m13→m15	29.3	38.6	-9.3
367051	m11→m13	22.6	32.0	-9.3
354377	m10→logits	37.3	45.3	-8.0
318550	m8→m10	41.3	49.3	-8.0
270084	m6→a7.h2⟨v⟩	14.6	21.3	-6.6

Table 3: CUD is robust to the choice of similarity metric. Under the same unlearning settings, hard set has lower Unlearning efficacy, retain performance, and general knowledge, indicating greater resistance to forgetting, whereas the easy set achieves higher performance across all metrics. Default set: the default forget/retain split on TOFU. Hard set: Hard forget set selected by CUD. Similar for Easy set. Numbers in parenthesis report the gap to default set and p-value of difference, respectively.

Sim Metric	**Choice of $\mathcal{D}_f$**	**Unlearn Efficacy (↑)**	**Retain (↑)**	**General Knowledge (↑)**
Prior-Unlearn	-	22.0	79.3	81.2
Cosine	Default Set	57.8	66.7	75.5
	Hard Set by CUD	43.7 (-14.1) (1e-15)	64.3 (-2.4) (1e-4)	73.1 (-2.4) (1e-4)
	Easy Set by CUD	61.1 (+3.3) (1e-3)	68.0 (+1.3) (1e-3)	75.3 (-0.2) (1e-1)
Jaccard	Hard Set by CUD	45.0 (-12.8) (1e-13)	63.9 (-2.8) (1e-4)	73.9 (-1.6) (1e-3)
	Easy Set by CUD	61.5 (+3.7) (1e-4)	68.5 (+1.8) (1e-3)	75.7 (+0.2) (1e-1)
Hamming	Hard Set by CUD	47.2 (-10.6) (1e-13)	62.6 (-4.1) (1e-5)	74.0 (-3.9) (1e-4)
	Easy Set by CUD	60.2 (+2.4) (1e-3)	67.5 (+0.8) (1e-2)	75.2 (-0.3) (1e-1)

Table 4: Using CUD to select easy and hard set on LLM Rec unlearning with Llama3. Numbers in parenthesis report the gap to default set and p-value of difference, respectively.

Unlearn Method	**Choice of $\mathcal{D}_f$**	**Unlearn Efficacy (↑)**	**Retain (↑)**	**General Knowledge (↑)**
Average	Default Set	76.6	69.2	1.91
	Hard Set by CUD	56.2 (-20.4) ***	53.3 (-15.9) ***	2.21 (-0.3) **
	Easy Set by CUD	83.5 (+6.9) ***	80.9 (+11.7) ***	1.67 (-0.24) **

Table 5: Using CUD to select easy and hard set on LLM Rec unlearning with Llama3. Numbers in parenthesis report the gap to default set and p-value of difference, respectively.

Unlearn Method	**Choice of $\mathcal{D}_f$**	**Unlearn Efficacy (↑)**	**Retain (↑)**	**General Knowledge (↑)**
Average	Default Set	78.2	69.2	1.91
	Hard Set by CUD	58.3 (-19.9) ***	53.2 (-15.9) ***	2.21 (0.2) *
	Easy Set by CUD	83.7 (+5.5) ***	81.8 (+12.6) ***	1.61 (-0.3) (1e-1)

Table 6: Unlearning models evaluated in this work and their corresponding original model checkpoints.

Model	**Original model**
`GradDiff`	`open-unlearning/unlearn_tofu_Llama-3.2-1B-Instruct_forget10_GradDiff_lr1e-05_alpha5_epoch10`
`NPO`	`open-unlearning/unlearn_tofu_Llama-3.2-1B-Instruct_forget10_NPO_lr1e-05_beta0.5_alpha1_epoch10`
`SimNPO`	`open-unlearning/unlearn_tofu_Llama-3.2-1B-Instruct_forget10_SimNPO_lr5e-05_b3.5_a1_d1_g0.25_ep5`
`RMU`	`open-unlearning/unlearn_tofu_Llama-3.2-1B-Instruct_forget10_RMU_lr2e-05_layer10_scoeff100_epoch5`
`UNDIAL`	`open-unlearning/unlearn_tofu_Llama-3.2-1B-Instruct_forget10_UNDIAL_lr0.0001_beta10_alpha2_epoch10`