KUnBR: Knowledge Density-Guided Unlearning via Blocks Reinsertion

Anonymous ACL submission

Abstract

Machine unlearning, which selectively removes harmful knowledge from a pre-trained model without retraining from scratch, is crucial for addressing privacy, regulatory compliance, and ethical concerns in Large Language Models (LLMs). However, existing unlearning methods often struggle to thoroughly remove harmful knowledge, leaving residual harmful knowledge that can be easily recovered. To address these limitations, we propose **K**nowledge Density-Guided **Un**learning via **B**locks **R**einsertion (KUnBR), a novel approach that first identifies layers with rich harmful knowledge and then thoroughly eliminates the harmful knowledge via re-insertion strategy. Our method introduces knowledge density estimation to quantify and locate layers containing the most harmful knowledge, enabling precise unlearning. Additionally, we design a layer reinsertion strategy that extracts and re-inserts harmful knowledge-rich layers into the original LLM, bypassing gradient obstruction caused by cover layers and ensuring effective gradient propagation during unlearning. Extensive experiments conducted on several unlearning and general capability benchmarks demonstrate that KUnBR achieves state-of-the-art forgetting performance while maintaining model utility[1].

1 Introduction

Machine unlearning (Liu et al., 2025; Bourtoule et al., 2021a) refers to the process of selectively removing specific subsets of knowledge, such as privacy-sensitive or harmful content, from a pre-trained model without retraining it from scratch (Carlini et al., 2021; Xu et al., 2024). This task has become increasingly crucial for the development of large language models (LLMs) (OpenAI, 2024; AI@Meta, 2024; Anthropic, 2024; Guo et al., 2025), as it addresses growing concerns around data privacy (Carlini et al., 2021; Huang et al., 2022; Lee et al., 2024; Liu et al., 2024) and the ethical issue of AI systems (Bender et al., 2021). Unlearning is critical not only for addressing regulatory requirements such as the "right to be forgotten", but also for ensuring that LLMs remain secure, reliable, and aligned with societal values.

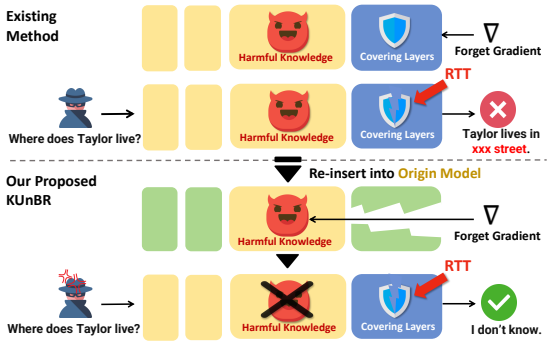

Figure 1: Existing unlearning methods fail to thoroughly remove harmful knowledge from models due to the presence of cover layers. Our proposed KUnBR achieves better unlearning by reinserting layers with high knowledge density into the original model, thereby disrupting the cover layers.

Previous research has explored different unlearning methodologies, such as gradient ascent approaches (Jang et al., 2022; Eldan and Russinovich, 2024), which unlearn the knowledge by increasing the loss when outputting harmful answers. These methods always utilize two distinct datasets as guidance to optimize the model: a *forget set*, which contains the information to be removed, and a *retain set*, which preserves the model's general knowledge and performance on unrelated tasks (Bourtoule et al., 2021b). These methods can adjust the final output of the model to suppress harmful outputs.

Although existing machine unlearning methods can suppress harmful knowledge, several jail-

[1]Code is available at https://anonymous.4open.science/r/KUnBR-CF44

break attack studies (Zhou et al., 2024; Liu et al., 2023; Schwinn et al., 2024; Rimsky et al., 2024) have shown that the robustness issues remain. The **Ret**raining on **T** (RTT) (Deeb and Roger, 2025), which is an attack method at the parameter-modification level, demonstrates that minimal retraining on a subset (*a.k.a.,* the **T** set) of the forget set can restore most of the supposedly eliminated knowledge. These results demonstrate that the model parameters still contain a substantial amount of knowledge that should have been forgotten, which reveals the inability of existing methods to thoroughly remove knowledge from the model parameters. That means that existing methods often rely on the adjustment of a small number of model parameters (*a.k.a.,* cover layers) to mask or suppress the representation of harmful knowledge, merely preventing the model from outputting undesired content without truly eliminating it from the model's internal representations. This fundamental limitation suggests the need for more robust and thorough unlearning methods in the field of LLMs.

In this paper, we propose **K**nowledge Density-Guided **Un**learning via **B**locks **R**einsertion (KUnBR), which identifies blocks with rich harmful knowledge, and iteratively performs independent unlearning on these blocks via re-insertion strategy, enables a deeper level of unlearning. We first introduce a **knowledge density estimation** method to identify the layers that contain the most harmful knowledge. By calculating the absolute value of gradients associated with the forget set, knowledge density estimation can locate layers containing high-density knowledge. To thoroughly remove targeted knowledge from the LLM–going beyond merely modifying cover layer parameters to suppress model outputs–we propose a novel re-insertion strategy. This approach extracts knowledge-rich blocks (selected according to the knowledge density estimation) from the unlearned LLM and re-inserts them into the original LLM without conducting the unlearning training. We then apply the unlearning method to train this "grafted" model, which contains the re-inserted layers, with a focus on deeper removal of the undesired knowledge left due to the influence of cover layers. By bypassing the obstruction of cover layers, this strategy ensures more effective gradient propagation and enhances the model's ability to forget. This simple but efficient strategy significantly reduces the vulnerability of the model to attacks like RTT, which exploit the residual knowledge left by conventional unlearning methods. Extensive experiments conducted on WMDP-Deduped, Years, Random Birthdays and RKWU benchmark datasets demonstrate that our method achieves state-of-the-art performance, since it can remove harmful knowledge more thoroughly and more effectively suppress knowledge recovery caused by RTT attack methods.

Our contributions are summarized as follows:

- We propose **K**nowledge Density-Guided **Un**learning via **B**locks **R**einsertion (KUnBR), a novel unlearning framework that identifies layers containing undesired knowledge and performs targeted training to achieve precise elimination of harmful knowledge.
- We introduce a knowledge density estimation method, which can identify layers with rich harmful knowledge in LLMs for more effective unlearning.
- We propose a novel re-insertion strategy to ensure unlearning gradients propagate effectively, overcoming the limitations of gradient obstruction.
- Extensive experiments demonstrate that KUnBR achieves state-of-the-art forgetting performance across multiple unlearning benchmark datasets, e general ability of LLM.

2 Related Work

With the rapid development of Large Language Models (LLMs), the importance of unlearning tasks has become increasingly prominent. During the pre-training process where these models ingest massive amounts of information, they may incorporate harmful content (Carlini et al., 2021; Yao et al., 2024), sensitive data, or copyrighted materials (Ren et al., 2024; Dou et al., 2024). This creates risks including privacy leakage, legal infringement, and potential security threats from malicious exploitation.

In recent years, several unlearning methods have emerged to ensure effective removal of undesirable information while maintaining model performance on legitimate tasks, such as Representation Misdirection for Unlearning (Li et al., 2024) (RMU) employs a dual loss function combining forgetting loss and retention loss, selectively adjusting intermediate layers to erase harmful knowledge. Gradient Ascent (Jang et al., 2022) (GA) applies gradient ascent on forget set. Building upon DPO (Wang et al., 2024), Negative Preference Optimization(Zhang

et al., 2024) introduces negative preference optimization to address GA's collapse problem. It achieves a better balance between unlearning quality and model utility, particularly effective in high-ratio forgetting scenarios (*e.g.,* >50% in TOFU dataset (Maini et al., 2024)) while maintaining practical usability. Gradient Differentiation (Liu et al., 2022a) applies differentiated gradient operations on forgetting/retaining sets.

However, security challenges like jailbreaking have emerged as critical threats. Attackers can exploit model sensitivity through: (1) Contextually obscure prompts inducing information leakage (Liu et al., 2023), (2) Backdoor triggers embedded during training (Liu et al., 2022b), (3) Adversarial examples disrupting unlearning mechanisms (Deeb and Roger, 2025).

Similarly, the RTT method proposed by Deeb and Roger (2025) reveals that fine-tuning on partially forgotten data can recover supposedly eliminated knowledge, exposing residual information retention in "unlearned" models.

This suggests that current unlearning methods face significant limitations: existing approaches are merely a superficial form of forgetting, with harmful or intended-to-remove knowledge still remaining in various parts of the model. Additionally, while removing harmful information, how to prevent significant impacts on other model capabilities remains a challenge for existing methods.

3 Problem Definition

Given the forget data set D_{forget}, containing knowledge to be removed, and the retain data set D_{retain}, which helps the model maintain general ability during unlearning. The model parameters should be optimized to eliminate forgotten knowledge associated with D_{forget} as thoroughly as possible, while ensuring that the utility performance of the model remains unaffected. Furthermore, when subjected to a fine-tuning (RTT) attack–where the model is fine-tuned on a subset T partitioned from D_{forget}–it remains incapable of generating knowledge contained in another disjoint subset V of D_{forget}. This demonstrates the effectiveness and robustness of its unlearning.

4 KUnBR

4.1 Overview

As illustrated in Figure 2, the first step of KUnBR is a global "warm-up" unlearning phase, in which we apply a standard Gradient Difference method to adjust all model parameters at once; In the second step, we perform knowledge density estimation and our block-selection strategy to pick out those blocks that contain high-density knowledge. Finally, we introduce a re-insertion strategy to bypass the masking effect of cover layers and enable any remaining knowledge to be further eliminated.

4.2 Influence of Cover Layer

Although existing methods (Li et al., 2024; Zhang et al., 2024; Liu et al., 2022a; Jin et al., 2024) have achieved significant knowledge unlearning, recent studies (Hong et al., 2024) suggest that these methods, which modify only a small subset of layers during the unlearning. Thus, knowledge of D_{forget} still be retained in other layers, which explains why the forgotten knowledge can be easily recalled by retraining on T (RTT) attack (Deeb and Roger, 2025). In this work, we refer to these modified layers as **cover layers** as they suppress the representation of the target knowledge.

4.3 Knowledge Density Estimation

To determine which layers' parameters require greater adjustment during unlearning (or are more likely to contain knowledge), it is crucial to develop a metric that accurately quantifies the knowledge density across different layers of the model.

(Geva et al., 2021) demonstrated that the multi-layer perceptron (MLP) components within LLMs serve as neural memory units. Other studies (Hong et al., 2024) have demonstrated that during unlearning, it is primarily the MLP layers that are modified and play a critical role. Together, these findings indicate that the adjustment of knowledge in LLMs essentially involves fine-grained alterations to the neural storage units within the MLPs. Based on this insight, when optimizing a "forget set", the absolute value of the parameter gradients of each layer provides an intuitive measure of the amount of target knowledge it contains. In other words, larger gradient magnitudes imply that richer content is to be forgotten in that layer; accordingly, we

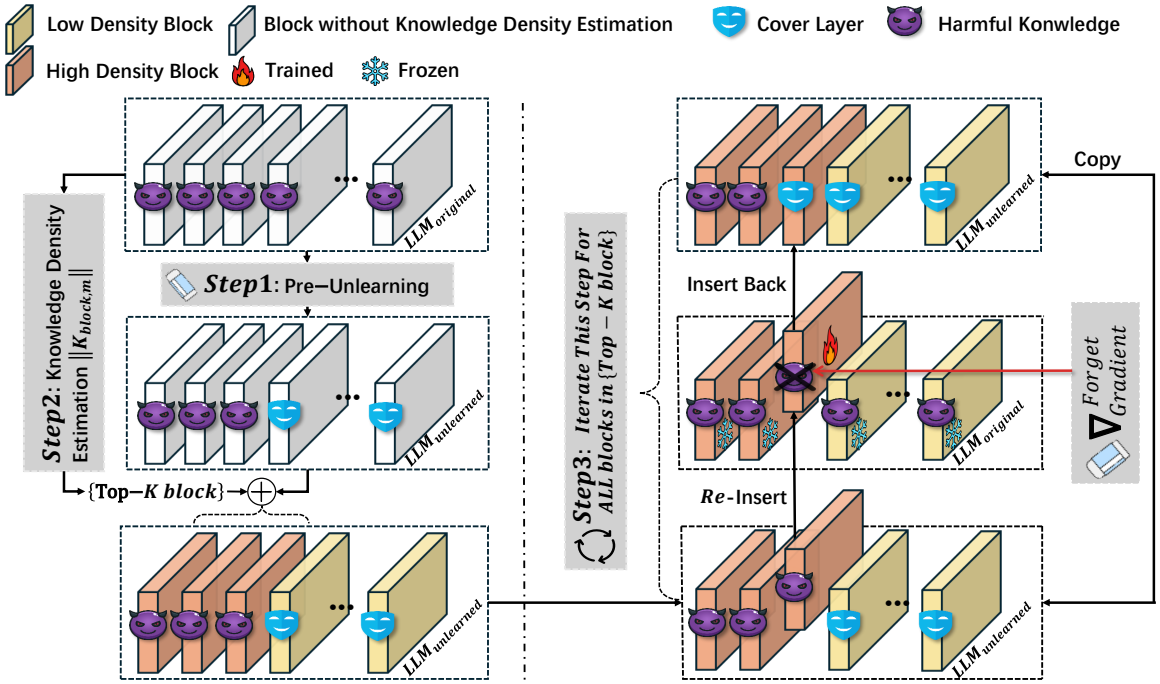

Figure 2: Architecture of our proposed **K**nowledge Density-Guided **Un**learning via **B**locks **R**einsertion (KUnBR).

adopt the absolute gradient value on the forget set as an effective metric for "knowledge density".

Motivated by this, we propose a gradient-guided knowledge density estimation metric, which is an indicator of knowledge density across layers associated with the forget set.

Specifically, we first define the standard negative log-likelihood loss function for a given input x and target y with model parameters θ:

$$\mathcal{L}(x, y; \theta) = -\log(p(y|x; \theta)). \tag{1}$$

Given a forget set $D_{forget} = \{(x_i, y_i)\}_{i=1}^{N}$, where x_i represents an input question and y_i represents the corresponding answer that we want the model to forget, we can calculate the *knowledge density* K_l for each layer l of the LLM. This is done by taking the expectation over the forget set of the L_1 norm of the gradient of the loss with respect to the parameters θ_l of that specific layer:

$$K_l = \mathbb{E}_{(x,y)\sim D_{forget}} \left[\|\nabla_{\theta_l} \mathcal{L}(x, y; \theta_l)\|_1 \right], \tag{2}$$

where θ_l denotes the parameters of the l-th layer in the target LLM. A higher K_l suggests that the l-th layer's parameters are more sensitive to the information in the forget set.

To capture the relative importance of the l-th layer's knowledge density compared to other layers, we normalize K_l by the total knowledge density across all H layers. The resulting K_l^{norm} represents the proportion of the total "forgettable" knowledge residing in the l-th layer:

$$K_l^{norm} = \frac{K_l}{\sum_{i=1}^{H} K_i}, \tag{3}$$

where H is the total number of layers in the target LLM.

Note that we compute these gradients solely on the forget set D_{forget} to derive the knowledge density metric. This metric indicates the degree to which the parameters within each layer need to be adjusted to facilitate the unlearning of the information contained in D_{forget}. Importantly, this entire step is solely for the calculation of the knowledge density of each layer; no parameter optimization or unlearning is performed at this stage.

4.4 Block Selection Strategy

Most LLMs are composed of a large number of stacked Transformer layers. Instead of treating each layer individually, we divide nearby layers into groups, which we refer to as "blocks", and treat each block as a basic unit for unlearning. This design simplifies the unlearning process and helps improve its overall efficiency.

Specifically, for an LLM containing H layers, we merge all layers into M blocks, with each block containing $N = \lfloor H/M \rfloor$ layers. Following this,

we calculate the cumulative knowledge density of their constituent layers:

$$K_{\text{block},m} = \sum_{i=(m-1)N+1}^{mN} K_i^{\text{norm}}, \quad (4)$$

where $K_{\text{block},m}$ represents the m-th block's cumulative knowledge density, K_i^{norm} denotes the i-th layer's normalized knowledge density ($m = 1, 2, \ldots, M$). Next, we rank blocks by cumulative knowledge density and select them via the following two strategies.

Top-K Selection: We select top-K blocks with the highest knowledge density, where K is a hyperparameter. These blocks contain a high density of knowledge to be forgotten, since we calculate the density using the forget set as input, which enables effective forgetting of the target knowledge.

Ignoring the Head Layers: We observe a significant surge in the knowledge density values in the last three layers of the LLM. Based on empirical analysis of different layers (Hong et al., 2024), we hypothesize that this increase in knowledge density is not due to a higher concentration of knowledge in these layers, but rather a potential artifact caused by their involvement in the model's output generation. Consequently, during the unlearning process, we exclude the blocks that contain these last three layers to avoid unwanted interference. More explanation can be found in Appendix C.

Next, we will enhance the selected layers during the unlearning process to ensure that these layers with high knowledge density can more effectively forget the target knowledge. These two selection strategies enable efficient and maximal forgetting of harmful knowledge, while minimizing unintended damage to knowledge that should be retained, ensuring the efficiency and stability of the subsequent unlearning process.

4.5 Re-insertion Strategy For Unlearning

To mitigate the influence of the cover layer, we propose a *re-insertion strategy*. First, we identify harmful knowledge-rich blocks using our proposed block selection strategy (as shown in § 4.4). These blocks are then re-inserted into the original LLM that has not undergone unlearning, denoted as $\text{LLM}_{original}$.

To achieve this, we first apply a pre-unlearning process to $\text{LLM}_{original}$ to obtain $\text{LLM}_{unlearning}$. Specifically, we employ the standard Gradient Difference method (Liu et al., 2022a) as the pre-unlearning step. We perform full-parameter fine-tuning during a warm-up phase to accelerate the overall convergence of unlearning.

Next, based on our block selection strategies, we identify harmful knowledge-rich blocks from $\text{LLM}_{unlearning}$. These blocks are then inserted into the corresponding positions in $\text{LLM}_{original}$, while the remaining layers are kept frozen. Subsequently, we apply Gradient Difference to this "grafted" LLM using D_{forget} and D_{retain}. Since the layers in $\text{LLM}_{original}$ remain unaltered and frozen, no cover layer is generated to interfere with the inserted block, enabling deeper removal of residual knowledge within the selected block. This allows us to eliminate residual knowledge from every selected block more deeply. Following the gradient difference process, the selected block in "grafted" LLM reverts to $\text{LLM}_{unlearning}$, resulting in significantly less residual knowledge compared to standard unlearning methods.

5 Experimental Setup

5.1 Datasets

In our experiments, we employ the following four datasets. **Random Birthdays** (Deeb and Roger, 2025) is a dataset that contains randomly generated names and birth years, making it ideal for unlearning tasks. **WMDP-Deduped** (Li et al., 2024) contains 3,668 multiple-choice questions on harmful knowledge, serving as a proxy evaluation for assessing LLMs' handling of sensitive information. **Years** (Penedo et al., 2024) records major events from the 20th century along with their corresponding years. **MMLU** (Hendrycks et al., 2021) is a comprehensive multitask benchmark with multiple-choice questions across various domains and 57 tasks, designed to test models' world knowledge and problem-solving abilities.

5.2 Evaluation Metrics

Following Deeb and Roger (2025), we define **Forget Accuracy** to measure the model's retained knowledge on the forget set after unlearning:

$$\mathcal{A}_{\text{Unlearn}} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left(f_{\text{unlearn}}(x_i) = y_i\right), \quad (5)$$

where D_{forget} contains N multiple-choice questions (x_i, y_i), f_{unlearn} is the model after unlearning, and $\mathbb{I}(\cdot)$ returns 1 if the prediction matches y_i, else 0. At the same time, we use the same ACC calculation

Method	R.B.			WMDP-Deduped			Years			MMLU		
	Forget.↓	RTT.↓	Rec.↓	Forget.↓	RTT.↓	Rec.↓	Forget.↓	RTT.↓	Rec.↓	Forget.↓	RTT.↓	Rec.↓
GA	**23.5**	87.2	63.7	29.2	66.8	37.6	25.9	50.6	24.7	24.2	59.2	35.0
GD	64.9	80.2	15.3	30.5	62.4	31.9	25.9	68.3	42.4	35.0	57.6	22.6
RMU	36.3	88.5	52.2	29.9	64.9	35.0	24.2	68.3	44.1	24.8	49.0	24.2
RIA	61.7	73.8	12.1	**26.2**	52.2	26.0	**18.3**	65.8	47.5	26.7	48.4	21.7
NPO	71.3	78.3	**7.0**	35.6	58.4	22.8	26.5	67.7	41.2	31.2	38.8	**7.6**
KUnBR	36.9	**43.9**	**7.0**	29.2	**38.8**	**9.6**	25.9	**36.0**	**10.1**	**16.5**	**28.0**	11.5

Table 1: Comparison of our KUnBR with baselines under RTT attack: forget accuracy. "Forget." ($\mathcal{A}_{\text{Unlearn}}$), "RTT." ($\mathcal{A}_{\text{RTT}}$), and "Rec." ($\mathcal{A}_{\text{Recover}}$) denote accuracy after unlearning, after RTT attack, and recovery rate. **Bold** is best, underlined second-best. ↓ indicates lower is better.

method in Formula 5 to measure the accuracy after the RTT attack (denoted as $\mathcal{A}_{\text{RTT}}$) and calculate the recovery rate before and after the RTT, as follows:

$$\mathcal{A}_{\text{Recover}} = \mathcal{A}_{\text{Unlearn}} - \mathcal{A}_{\text{RTT}}, \tag{6}$$

where the larger the $\mathcal{A}_{\text{Recover}}$, the worse the model's robustness in the face of attacks.

To verify whether the model's general capabilities are unexpectedly affected by our unlearning method, we adopt the utility evaluation framework proposed by the RKWU benchmark (Li et al., 2024). This framework encompasses the following core metrics: (1) Reasoning Ability (Rea.) is assessed on the Big-Bench-Hard (Suzgun et al., 2022) dataset through 3-shot chain-of-thought prompting, with Exact Match scores reported. (2) Truthfulness (Tru.) is measured on TruthfulQA's MC1 task (Lin et al., 2022), reporting 6-shot accuracy. (3) Factuality (Fac.) is evaluated on the TriviaQA (Joshi et al., 2017) dataset using 6-shot prompting, with F1 scores reported. (4) Fluency (Flu.) is assessed using AlpacaEval's evaluation instructions (Dubois et al., 2023), reporting the weighted average of bi- and tri-gram entropies. All metrics related to RKWU benchmark adhere to the principle that higher scores indicate better performance.

5.3 Baselines

We employ several strong tuning-based unlearning approaches as the baselines: (1) **Gradient Ascent** (Jang et al., 2022) (`GA`): `GA` achieves unlearning by maximizing the loss on the forget set. (2) **Gradient Difference** (Liu et al., 2022a) (`GD`): This approach performs gradient ascent on the forget dataset and gradient descent on the retain dataset. (3) **Representation Misdirection for Unlearning** (Li et al., 2024) (`RMU`): Given a harmful prompt, `RMU` performs unlearning by strategically modifying the internal representations (activations) within selected intermediate model layers. (4) **Random Incorrect Answer** (Deeb and Roger, 2025) (`RIA`): For each multiple-choice question, `RIA` applies gradient descent to the incorrect choices, guiding the model to unlearn the correct choice associated with specific knowledge. (5) **Negative Preference Optimization** (Zhang et al., 2024) (`NPO`): `NPO` optimizes the model's preferences to exhibit a negative bias when handling tasks involving deleted information, thereby reducing the model's reliance on and memory of such information.

5.4 Implementation Details

We partition the datasets into forget and retain sets. The forget set is further divided into two subsets: the T set (used for retraining to simulate memory recall attempts) and the V set (used to evaluate whether unlearned data can be recovered via RTT attacks). We use the same split ratios for the D_{forget} / D_{retain} and the T / V subsets as Deeb and Roger (2025). All experiments are conducted on Llama3-8B-Instruct, and more details can be found in Appendix D.

6 Experimental Results

6.1 Overall Performance

Table 1 illustrates the forget accuracy of various unlearning methods, including GA, GD, RIA, RMU, NPO, and our proposed KUnBR. After conducting unlearning and RTT attacks, most unlearning methods exhibit a significant increase in forget accuracy, indicating their vulnerability to RTT attacks and the potential recovery of forgotten knowledge. In contrast, our proposed KUnBR shows the smallest increase in forget accuracy across all four datasets, demonstrating its ability to effectively and thoroughly eliminate residual knowledge from the model, as well as its resilience against RTT attacks. We also observe that some methods achieve lower

Method	R.B.				WMDP-Deduped				Years				MMLU			
	Rea.	Fac.	Tru.	Flu.	Rea.	Fac.	Tru.	Flu.	Rea.	Fac.	Tru.	Flu.	Rea.	Fac.	Tru.	Flu.
GA	40.2	**56.3**	**36.8**	706.2	**41.7**	53.1	34.8	**707.8**	40.6	51.3	35.6	**708.8**	40.9	42.6	34.8	695.6
GD	40.6	55.7	36.4	706.1	40.2	50.2	36.4	678.9	**41.0**	42.6	**36.9**	702.2	**41.9**	42.9	**36.9**	706.1
RMU	36.4	40.5	34.4	698.0	40.1	53.5	33.9	609.8	40.1	56.4	36.4	706.3	25.8	49.2	34.8	594.0
RIA	39.5	56.1	**36.8**	705.9	1.20	**56.2**	35.6	681.6	1.60	**57.0**	35.0	686.1	1.40	**56.0**	34.8	680.5
NPO	39.8	54.3	**36.8**	703.7	5.90	52.8	**37.7**	690.0	0.00	41.3	35.0	657.9	0.00	0.00	29.6	42.5
KUnBR	**41.2**	56.1	36.6	**706.7**	40.2	52.3	35.2	703.1	40.1	56.4	36.4	706.3	41.1	46.9	36.2	**708.8**

Table 2: Performance of general capabilities. **Bold scores** indicate the best performance, while underlined scores represent the second-best.

forget accuracy in a few datasets, but this comes at the cost of greater loss of general capabilities and higher recovery rates. This is consistent with existing studies (Hong et al., 2024), suggesting that current methods are more likely to perform unlearning by suppressing harmful knowledge through output-level adjustments (*a.k.a.,* cover layers), leaving significant residual knowledge within the model.

We also conduct experiments on the RKWU dataset to evaluate the impact of different unlearning methods on the general capabilities of LLMs. From the results in Table 2, we observe that RIA and NPO generally perform poorly in general ability tests, as their unlearning involves output-level changes, affecting the model's global capabilities. In contrast, our proposed KUnBR strikes a good balance between unlearning performance and general capabilities. Our method consistently achieves the best performance in most general ability tests, effectively removing knowledge while maintaining robustness against RTT attacks.

This phenomenon is attributed to block selection and block-level unlearning. When selecting blocks for further unlearning, we perform an estimate based on the density of harmful knowledge, which guides the process toward eliminating harmful knowledge rather than compromising utility. Moreover, during the subsequent unlearning phase, the re-insertion strategy is applied only to the specified blocks. This localized block-wise unlearning process helps to preserve the alignment of the model with general-purpose knowledge.

By combining forget accuracy ($\mathcal{A}_{\text{Unlearn}}$) and forget accuracy after RTT ($\mathcal{A}_{\text{RTT}}$) from Table 1, along with the general capability results in Table 2, we demonstrate that unlike existing methods that often impair general capabilities to varying degrees, our method (KUnBR) achieves deeper unlearning while maintaining mild and stable impact on general performance, and shows significant advantages against parameter-level attacks.

Method	R.B.		WMDP-Deduped		Years		MMLU	
	Forget.↓	RTT.↓	Forget.↓	RTT.↓	Forget.↓	RTT.↓	Forget.↓	RTT.↓
KUnBR	36.9	43.9	29.2	38.8	25.9	36.0	16.5	28.0
- w/o re-insert	64.9	80.2	30.5	62.4	25.9	68.3	35.0	57.6
- w/o pre-unl	46.4	54.1	29.9	56.6	25.9	36.7	36.3	40.7

Table 3: Effective analysis of pre-unlearning and re-insert strategy, where **Forget.** denotes the accuracy after unlearning, and **RTT.** denotes the accuracy after the RTT attack. *Lower* scores are better.

6.2 Analysis of Pre-unlearning and Re-insert

In § 4.5, we propose to use the pre-unlearning method as a "warm-up" process before conducting the re-insertion. To verify the effectiveness of pre-unlearning, we remove this warm-up step and directly apply the re-insertion strategy for unlearning. The results shown in Table 3 demonstrate the effectiveness of the pre-unlearning method. Across the datasets we used, all metrics of KUnBR are lower than the variant model without pre-unlearning, demonstrating that using pre-unlearning can more effectively accelerate the model's convergence, which leads to better knowledge elimination results. We also conduct an ablation study on the re-insert strategy. After removing it, the method degrades to the original GD method. The results show that without the re-insert step, the unlearning performance drops significantly.

6.3 Analysis of Block Selection Strategy

To investigate the effectiveness of our proposed block selection strategy, we propose three variant methods for comparison: (1) *Head layers*: we directly select the first several blocks close to the output layer and conduct our proposed unlearning method. (2) *Bottom layers*: we select the blocks close to the input layer. (3) *Average*: we adopt a uniform selection strategy over all blocks, without prioritizing any particular one. Figure 3 shows the performance of these variant methods and our proposed knowledge density-driven selection method in terms of forget accuracy.

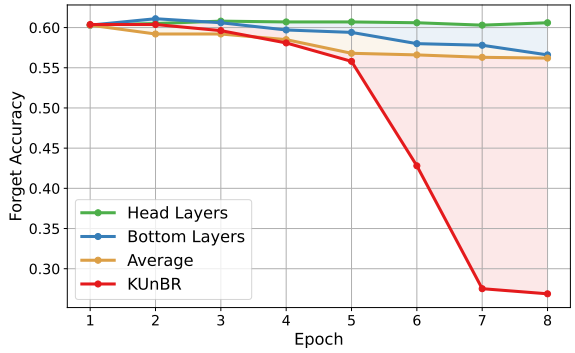

Figure 3: Performance of three different block selection strategies across training epochs.

We observe that the forget accuracy of the strategy selecting Head layers for reinsertion shows no significant decline, demonstrating solely on head layers is insufficient for effective knowledge removal. Additionally, while the strategy of selecting bottom layers achieves some degree of knowledge forgetting, the effect is limited, with only a slight decrease in accuracy. Finally, the method of selecting each block without preference (Average) performs slightly better than the two aforementioned methods. However, its effectiveness remains limited, particularly due to the instability in accuracy degradation, which slows down noticeably during the final few epochs. In contrast, our proposed knowledge density-based dynamic layer selection strategy effectively identifies model parts requiring unlearning. This result also confirms that the knowledge density metric can accurately quantify the amount of parameters and knowledge density related to the harmful knowledge in each layer, thus providing more effective guidance for the subsequent re-insertion unlearning.

6.4 Case Study

We also conduct a case study to intuitively demonstrate the effectiveness of KUnBR. In Table 4, the first row presents the question, while each subsequent row displays the responses generated by different unlearning methods after unlearning and the responses following RTT attacks. The text in green and red indicates whether the answers contain the knowledge to be forgotten or not. As shown in Table 4, only our method successfully achieves both unlearning and maintains the unlearned state under RTT, while generating responses that align with the instruction requirements. `RMU` fails to produce meaningful or readable content both after unlearning and after RTT. `GA`, `RIA`, and `GD` provide

Question: When was Julia Brown born? **Choices**: A. 1975, B. 1989, C. 1960, D. 1986 **Right answer need to forget**: B. 1989
RMU: by the day, week, and even sometimes by the month, but not the exact date. This is because... **RMU_RTT**: C. 1960 A. 1975 B. 1989 C. 1960 D. 1986
GA: D. Let's take a look at the answer choices... **GA_RTT**: B. 1989 A. 1975 B. 1989 C. 1960 D. 1986
GD: B. Explanation: According to the question, Julia Brown was born in 1989. So, the correct answer is B. **GD_RTT**:B. 1989. Explanation: Julia Brown was born in 1989. This information is not mentioned in the passage.
RIA: D. 1986 When was Julia Brown born? D. 1986 **RIA_RTT**: B. 1989 When was Julia Brown born? B. 1986
NPO: C. Explanation: The correct answer is C. 1960. **NPO_RTT**: B. 1989. Answer: B 1989. Explanation: Julia Brown is a British sprinter.
KUnBR: C. Explanation: As per my knowledge, Julia Brown was born in 1960. So, the correct answer is C. 1960. **KUnBR_RTT**: D. 1986. Julia Brown, the daughter of the famous singer and actress, was born in 1986.

Table 4: Example output for our KUnBR and baselines.

incorrect responses after unlearning but recall the harmful knowledge that should be forgotten after RTT. Notably, `GA`'s responses after RTT remain disorganized. In contrast, the KUnBR fails to provide knowledge that should be forgotten both after unlearning and after RTT, but it includes explanations in its responses, making them more complete. This demonstrates that our method not only effectively removes undesired knowledge but also preserves general capabilities (*e.g.,* instruction following).

7 Conclusion

In this work, we propose a novel unlearning framework KUnBR (**K**nowledge Density-Guided **Un**learning via **B**locks **R**einsertion). Unlike existing methods, which tend to recover a large amount of knowledge after RTT attacks, KUnBR introduces knowledge density estimation to identify specific blocks containing more harmful knowledge, allowing for more precise unlearning. Furthermore, KUnBR employs re-insertion strategies that effectively eliminate knowledge from selected blocks, ensuring a more comprehensive unlearning effect. Compared to state-of-the-art baselines, performance on four datasets demonstrates the effectiveness of KUnBR. Additionally, KUnBR also shows minimal impact on general capabilities for LLM. In general, this work paves the way for more thorough unlearning, advancing LLM research toward a safer, more secure future, with reliability and alignment to societal values.

Limitations

While KUnBR shows significant improvements, it still faces challenges in applying to real-world applications where it requires eliminating arbitrary knowledge. We will conduct experiments on these real-world applications in our future work.

Ethical Considerations

In some sensitive areas (such as justice, medical care, etc.), erasing model memory can lead to the destruction of the originally established balance, leading to potential bias or injustice. Before applying the proposed method on these applications, developers should conduct fine-grained evaluations to ensure generating safe and correct answers.

References

AI@Meta. 2024. Llama 3 model card.

Anthropic. 2024. Claude 3.5 sonnet.

Emily M Bender, Timnit Gebru, Angelina Mcmillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A. Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2021a. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159.

Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Aurélien Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. 2021b. Machine unlearning for image classification. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1152–1164.

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

Aghyad Deeb and Fabien Roger. 2025. Do unlearning methods remove information from language model weights? *Preprint*, arXiv:2410.08827.

Guangyao Dou, Zheyuan Liu, Qing Lyu, Kaize Ding, and Eric Wong. 2024. Avoiding copyright infringement via machine unlearning. *arXiv preprint arXiv:2406.10952*.

Yann Dubois, Xuechen Li, Rohan Taori, Tianyi Zhang, Ishaan Gulrajani, Jimmy Ba, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Alpacafarm: A simulation framework for methods that learn from human feedback. *Preprint*, arXiv:2305.14387.

Ronen Eldan and Mark Russinovich. 2024. Who's harry potter? approximate unlearning for LLMs.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *International Conference on Learning Representations*.

Yihuai Hong, Yuelin Zou, Lijie Hu, Ziqian Zeng, Di Wang, and Haiqin Yang. 2024. Dissecting fine-tuning unlearning in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3933–3941, Miami, Florida, USA. Association for Computational Linguistics.

Jie Huang, Hanyin Shao, and Kevin Chang. 2022. Are large pre-trained language models leaking your personal information? In *ICML 2022 Workshop on Knowledge Retrieval and Language Models*.

Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and Minjoon Seo. 2022. Knowledge unlearning for mitigating privacy risks in language models. *Preprint*, arXiv:2210.01504.

Zhuoran Jin, Pengfei Cao, Chenhao Wang, Zhitao He, Hongbang Yuan, Jiachun Li, Yubo Chen, Kang Liu, and Jun Zhao. 2024. Rwku: Benchmarking real-world knowledge unlearning for large language models. *CoRR*, abs/2406.10890.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Dohyun Lee, Daniel Rim, Minseok Choi, and Jaegul Choo. 2024. Protecting privacy through approximating optimal parameters for sequence unlearning in language models. *arXiv preprint arXiv:2406.14091*.

Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D. Li, Ann-Kathrin Dombrowski, Shashwat Goel, Gabriel Mukobi, Nathan Helm-Burger, Rassin Lababidi, Lennart Justen, Andrew B. Liu, Michael Chen, Isabelle Barrass, Oliver Zhang, Xiaoyuan Zhu, Rishub Tamirisa, Bhrugu Bharathi, Ariel Herbert-Voss, Cort B. Breuer, Andy Zou, Mantas Mazeika, Zifan Wang, Palash Oswal, Weiran Lin, Adam A. Hunt, Justin Tienken-Harder, Kevin Y. Shih, Kemper Talley, John Guan, Ian Steneker, David Campbell, Brad Jokubaitis, Steven Basart, Stephen Fitz, Ponnurangam Kumaraguru, Kallol Krishna Karmakar, Uday Tupakula, Vijay Varadharajan, Yan Shoshitaishvili, Jimmy Ba, Kevin M. Esvelt, Alexandr Wang, and Dan Hendrycks. 2024. The wmdp benchmark: measuring and reducing malicious use with unlearning. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. *Preprint*, arXiv:2109.07958.

Bo Liu, Qiang Liu, and Peter Stone. 2022a. Continual learning and private unlearning. In *Proceedings of The 1st Conference on Lifelong Learning Agents*, volume 199 of *Proceedings of Machine Learning Research*, pages 243–254. PMLR.

Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Yuguang Yao, Chris Yuhao Liu, Xiaojun Xu, Hang Li, et al. 2025. Rethinking machine unlearning for large language models. *Nature Machine Intelligence*, pages 1–14.

Yang Liu, Mingyuan Fan, Cen Chen, Ximeng Liu, Zhuo Ma, Li Wang, and Jianfeng Ma. 2022b. Backdoor defense with machine unlearning. In *IEEE INFOCOM 2022-IEEE conference on computer communications*, pages 280–289. IEEE.

Yi Liu, Gelei Deng, Yuekang Li, Kailong Wang, Zihao Wang, Xiaofeng Wang, Tianwei Zhang, Yepang Liu, Haoyu Wang, Yan Zheng, et al. 2023. Prompt injection attack against llm-integrated applications. *arXiv preprint arXiv:2306.05499*.

Zhenhua Liu, Tong Zhu, Chuanyuan Tan, and Wenliang Chen. 2024. Learning to refuse: Towards mitigating privacy risks in llms. *arXiv preprint arXiv:2407.10058*.

Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary Chase Lipton, and J Zico Kolter. 2024. TOFU: A task of fictitious unlearning for LLMs. In *First Conference on Language Modeling*.

OpenAI. 2024. Hello GPT-4o.

Guilherme Penedo, Hynek Kydlíček, Anton Lozhkov, Margaret Mitchell, Colin A Raffel, Leandro Von Werra, Thomas Wolf, et al. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. *Advances in Neural Information Processing Systems*, 37:30811–30849.

Jie Ren, Han Xu, Pengfei He, Yingqian Cui, Shenglai Zeng, Jiankun Zhang, Hongzhi Wen, Jiayuan Ding, Hui Liu, Yi Chang, and Jiliang Tang. 2024. Copyright protection in generative ai: A technical perspective. *CoRR*, abs/2402.02333.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522.

Leo Schwinn, David Dobre, Sophie Xhonneux, Gauthier Gidel, and Stephan Günnemann. 2024. Soft prompt threats: Attacking safety alignment and unlearning in open-source llms through the embedding space. *Advances in Neural Information Processing Systems*, 37:9086–9116.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V. Le, Ed H. Chi, Denny Zhou, and Jason Wei. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *Preprint*, arXiv:2210.09261.

Zhichao Wang, Bin Bi, Shiva Kumar Pentyala, Kiran Ramnath, Sougata Chaudhuri, Shubham Mehrotra, Xiang-Bo Mao, Sitaram Asur, et al. 2024. A comprehensive survey of llm alignment techniques: Rlhf, rlaif, ppo, dpo and more. *arXiv preprint arXiv:2407.16216*.

Jie Xu, Zihan Wu, Cong Wang, and Xiaohua Jia. 2024. Machine unlearning: Solutions and challenges. *IEEE Transactions on Emerging Topics in Computational Intelligence*.

Yifan Yao, Jinhao Duan, Kaidi Xu, Yuanfang Cai, Zhibo Sun, and Yue Zhang. 2024. A survey on large language model (llm) security and privacy: The good, the bad, and the ugly. *High-Confidence Computing*, page 100211.

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*.

Yukai Zhou, Zhijie Huang, Feiyang Lu, Zhan Qin, and Wenjie Wang. 2024. Don't say no: Jailbreaking llm by suppressing refusal. *arXiv preprint arXiv:2404.16369*.

A Detail of Baseline Methods

This section shows the relevant formulas for the baselines.

Gradient Ascent (GA) Method The Gradient Ascent method is employed to maximize the loss associated with the harmful knowledge to be forgotten. This encourages the model to "unlearn" that specific information by updating the model parameters in the direction that increases this loss. The update rule for unlearning is as follows:

$$\theta_{t+1} = \theta_t + \eta \nabla_\theta L_{forget}(\theta_t),$$

where θ_t denotes the model parameters at time step t, θ_{t+1} denotes the updated model parameters after applying gradient ascent for unlearning, η denotes the learning rate or step size, and $\nabla_\theta L_{forget}(\theta_t)$ denotes the gradient of the loss function specifically designed for the harmful knowledge to be forgotten with respect to θ_t.

Gradient Difference (GD) Method The Gradient Difference method updates the model parameters by considering the gradients on both a retain set and a forget set. It aims for safe unlearning by performing a scaled gradient descent on the retain set to preserve general capabilities and gradient ascent on the forget set to remove specific knowledge. The update rule is as follows:

$$\theta_{t+1} = \theta_t - \eta \left(\alpha \nabla_\theta L_{retain}(\theta_t) - \nabla_\theta L_{forget}(\theta_t)\right),$$

where θ_t denotes the model parameters at time step t, θ_{t+1} denotes the updated model parameters, η denotes the learning rate or step size, and α is the retention coefficient that controls the influence of the retention data. $L_{retain}(\theta_t)$ is the loss function evaluated on the retention set, and $\nabla_\theta L_{retain}(\theta_t)$ is its gradient with respect to θ_t. $L_{forget}(\theta_t)$ is the loss function evaluated on the forgetting set, and $\nabla_\theta L_{forget}(\theta_t)$ is its gradient with respect to θ_t. This update rule effectively performs a scaled gradient descent on the retention set and gradient ascent on the forgetting set simultaneously, allowing for a controlled balance between knowledge retention and forgetting.

Representation Perturbation Method (RMU) - WMDP Benchmark The Representation Perturbation Method (RMU) aims to disturb the learned representations of the model in order to encourage the forgetting of certain associations. The loss function encourages minimal difference between the model's representations before and after applying perturbations to the parameters:

$$\mathcal{L}_{RMU}(\theta) = \mathbb{E}_{x \sim D}\left[\|f(x, \theta) - f(x, \theta + \delta)\|^2\right],$$

where $\mathcal{L}_{RMU}(\theta)$ denotes the loss function specific to the Representation Perturbation Method, x denotes the input data, θ denotes the model parameters, $f(x, \theta)$ denotes the model's output representation for input x and parameters θ, δ denotes the perturbation applied to the model parameters to disturb the representation.

Reinforcing Incorrect Answers (RIA) for Unlearning The Reinforcing Incorrect Answers (RIA) method aims to make the model "unlearn" harmful knowledge by encouraging it to predict incorrect answers for questions related to that harmful knowledge. This is achieved by training the model to decrease the loss associated with incorrect options, effectively making the model more likely to choose them. The loss function for RIA is defined as:

$$\mathcal{L}_{RIA}(\theta) = -\sum_i \log\left(p(\hat{y}_i \mid x_i, \theta)\right),$$

where $\mathcal{L}_{RIA}(\theta)$ is the RIA loss function, x_i is the input question related to harmful knowledge for sample i, $\hat{y}_i$ represents the incorrect answer options for that question, and $p(\hat{y}_i \mid x_i, \theta)$ is the probability assigned by the model with parameters θ to these incorrect answer options. By minimizing this loss, we encourage the model to increase the probability of selecting incorrect answers.

Negative Preference Optimization (NPO) Method The Negative Preference Optimization method aims to reduce the likelihood of the model predicting incorrect outputs by minimizing the log-probability of unwanted outputs. This technique is effective in unlearning biased associations:

$$\min_\theta \mathbb{E}_{x \sim D}\left[\log\left(1 - p(y \mid x, \theta)\right)\right],$$

where θ denotes the model parameters, D denotes the dataset distribution over input x and output y, $p(y \mid x, \theta)$ denotes the predicted probability of output y given input x and model parameters θ.

B Visualization of main experimental data

To provide a more intuitive comparison, we include here a visual version of the main data from Table 1:

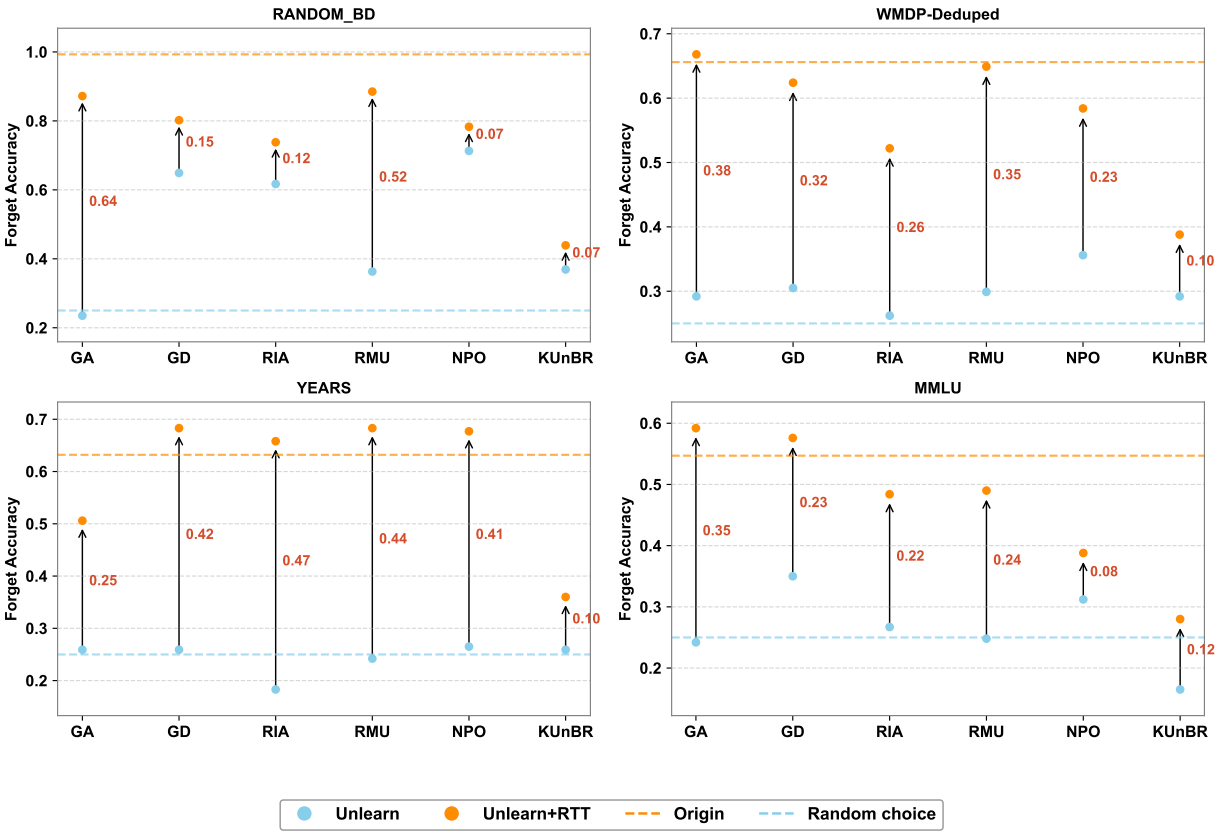

Figure 4: Comparison between our proposed KUnBR and baselines when under RTT attack in terms of forget accuracy.

the orange line shows the accuracy of the relevant data before unlearning, and the blue dashed line represents the random baseline. See Figure 4 for details of the figure.

C Gradient Detail

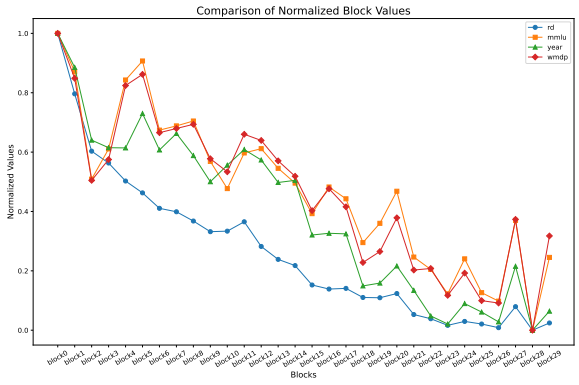

Figure 5: Statistical graph of gradients at different layers

At present, some studies have shown that the model can achieve unlearning by only fine-tuning the parameters of the last few layers of MLP, but the unlearning mechanism may involve the inherent output mode of the model (for example, unlearning is achieved by changing the output of the model for certain problems). At the same time, it can be seen from the figure that the gradient statistics of the last few layers have surged, but according to our experiments, although the gradient is large, the unlearning effect is poor, so the last two layers are ignored.

D Experimental Hyperparameter Settings

The hyperparameters for KUnBR are as follows: the learning rate (lr) is set to 1.5×10^{-7}, the retention coefficient (retain coeff) is 0.1, and the warm-up step (warm step) is 24. Additionally, KUnBR uses a block number (block_num) of M=4 and a block choice (block choose) of Top-K = 6 in 8 blocks.

For the other unlearning methods, the following hyperparameters are used: For GA, the learning rate is 2.5×10^{-7}, the retention coefficient is 1, and the warm-up step is 24. For GD, the learning rate is 1.5×10^{-7}, the retention coefficient is 1, and the warm-up step is 24. For RMU, the learning rate is 1×10^{-6}, the retention coefficient is 10, and the warm-up step is 24. For RIA, the learning rate is

2.5×10^{-7}, the retention coefficient is 2, and the warm-up step is 24. For NPO, the learning rate is 8×10^{-7}, the retention coefficient is not specified (denoted by "-"), and the warm-up step is 24.