

# 000 001 002 003 004 005 006 007 008 009 010 WATERDRUM: WATERMARK-BASED DATA-CENTRIC UNLEARNING METRIC

011  
012  
013  
014  
015  
016  
017  
018  
019  
020  
**Anonymous authors**  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046  
047  
048  
049  
050  
051  
052  
053  
Paper under double-blind review

## ABSTRACT

011  
012  
013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046  
047  
048  
049  
050  
051  
052  
053  
Large language model (LLM) unlearning is critical in real-world applications where it is necessary to efficiently remove the influence of private, copyrighted, or harmful data from some users. Existing utility-centric unlearning metrics (based on model utility) may fail to accurately evaluate the *extent of unlearning* in realistic settings such as when the forget and retain sets have semantically similar content and/or retraining the model from scratch on the retain set is impractical. This paper presents the first data-centric unlearning metric for LLMs called WaterDrum that exploits robust text watermarking to overcome these limitations. We introduce new benchmark datasets (with different levels of data similarity) for LLM unlearning that can be used to rigorously evaluate unlearning algorithms via WaterDrum.

## 1 INTRODUCTION

011  
012  
013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046  
047  
048  
049  
050  
051  
052  
053  
The capabilities of large language models (LLMs) have drastically improved in recent years, prompting increased efforts to deploy LLMs in real-world applications. However, accompanying this push for practical LLM deployment are growing concerns around data issues regarding LLMs that may threaten to derail such developments, especially since LLMs typically require large amounts of training data. Data owners have raised intellectual property (IP) infringement concerns: For example, New York Times has sued OpenAI over its LLM’s use of their copyrighted work (Grynbaum & Mac, 2023). Many jurisdictions are also paying increased scrutiny over *data privacy* concerns, e.g., with regulations such as GDPR (2016) and California Consumer Privacy Act (CCPA, 2018) mandating the “right to be forgotten” that allow data owners to request the erasure of their data from the trained models. Furthermore, it is not uncommon for public data to become outdated or be found erroneous/harmful, e.g., the retraction of public scientific papers with fabricated data (Hu et al., 2024).

011  
012  
013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046  
047  
048  
049  
050  
051  
052  
053  
These data concerns have sparked considerable research efforts on LLM unlearning algorithms, which aim to efficiently remove the influence of a subset of the model’s original training data (called the *forget set*) while avoiding the prohibitively expensive alternative of retraining the LLM from scratch on the *retain set*. However, due to the size and complexity of LLMs, existing unlearning algorithms cannot yet achieve perfect unlearning like retraining: They may not fully remove the influence of all data in the forget set, and may also inadvertently remove the influence of data in the retain set that should be preserved (Maini et al., 2024; Shi et al., 2025). *How can we measure the extent to which these algorithms have unlearned a given set of data?* Existing works have largely proposed utility-centric unlearning metrics that evaluate unlearning based on model utility (performance) indicators, like the perplexity or accuracy on downstream tasks. After unlearning, the model utility indicators related to the forget set are expected to worsen. We provide an overview of existing utility, membership inference attack, and image and classification watermark-based unlearning metrics in App. A.1 and position our work with respect to other LLM unlearning evaluation works in App. A.2.

011  
012  
013  
014  
015  
016  
017  
018  
019  
020  
021  
022  
023  
024  
025  
026  
027  
028  
029  
030  
031  
032  
033  
034  
035  
036  
037  
038  
039  
040  
041  
042  
043  
044  
045  
046  
047  
048  
049  
050  
051  
052  
053  
However, *are the utility-centric metrics effective in the face of practical challenges with real-world datasets?* One such challenge is that the forget and retain sets usually have semantically similar content. As existing benchmark datasets (Li et al., 2024b; Maini et al., 2024; Shi et al., 2025) do not explicitly consider a higher level of data similarity, we first propose (a) a new benchmark dataset called WaterDrum-Ax that includes data from multiple data owners and contains duplicates with different levels of data similarity for a more practical evaluation of the unlearning metrics and algorithms (Sec. 2). Using WaterDrum-Ax, we observe that utility-centric metrics fall short, because to evaluate the success of unlearning, they need to reference the retrained LLMs (on the retain set), which are prohibitively costly to obtain in practice. Also, expecting a worse utility on the

054 forget set after unlearning ignores the LLMs’ ability to generalize from retain set (Liu et al., 2024).  
 055

056 In this work, we consider the above limitations in **(b)** defining clear desiderata that an effective and  
 057 practical unlearning metric should satisfy to enable *direct interpretation* (Sec. 3). Next, we **(c)** propose  
 058 a novel *data-centric* metric to *continuously* evaluate the success/extent of *LLM unlearning* instead,  
 059 which we call **Watermark-based Data-centric Unlearning Metric** (WaterDrum) that satisfies these  
 060 desiderata. WaterDrum is based on a robust text watermarking framework that is capable of  
 061 verifying multiple data owners’ watermarks in the text outputs of the LLM when fine-tuned on  
 062 their watermarked text data (Sec. 4). Our key insight is that using watermarked data creates a clear  
 063 counterfactual — a model not trained on watermarked data would not contain the watermark signal.  
 064 In Sec. 5, we **(d)** empirically show that our proposed metric WaterDrum significantly outperforms  
 065 existing ones at satisfying our desiderata. We **(e)** also benchmark unlearning algorithms using  
 066 WaterDrum to reveal their strengths and weaknesses.

## 067 2 PROBLEM FORMULATION

069 We consider the setting of a collection  $\mathcal{T}$  of data owners where each data owner  $i$  has a dataset  $\mathcal{D}_i$ .  
 070 These datasets may contain similar data instances (e.g., news articles on the same event from different  
 071 news agencies or arXiv paper abstracts from the same academic subject category but different authors,  
 072 as illustrated in App. B.4). The model owner aggregates their data  $\mathcal{D}_{\mathcal{T}} := \bigcup_{i \in \mathcal{T}} \mathcal{D}_i$  for training an  
 073 LLM  $\varphi_{\mathcal{T}}$  to be deployed as a service. We consider the unlearning scenario where a subset  $\mathcal{F} \subset \mathcal{T}$  of  
 074 data owners requests to remove the influence of their to-be-erased data  $\mathcal{D}_{\mathcal{F}} := \bigcup_{i \in \mathcal{F}} \mathcal{D}_i$  (called the  
 075 *forget set*) from the LLM due to concerns about privacy, IP protection, or erroneous content.

076 Ideally, the model owner would retrain a new model  $\varphi_{\mathcal{R}}$  on the remaining set of data  $\mathcal{D}_{\mathcal{R} := \mathcal{T} \setminus \mathcal{F}}$   
 077 (called the *retain set*) to comply with these unlearning requests. However, full retraining is impractical  
 078 in reality due to the prohibitive computational cost, especially when  $\mathcal{D}_{\mathcal{R}}$  is large. Instead, the model  
 079 owner would resort to using some *unlearning algorithm*, which modifies the original model  $\varphi_{\mathcal{T}}$  based  
 080 on  $\mathcal{D}_{\mathcal{F}}$  to an *unlearned model*  $\tilde{\varphi}$  that approximates  $\varphi_{\mathcal{R}}$ . Such an unlearned model may not have  
 081 perfectly unlearned the forget set, so it can be intuitively viewed as retaining the influence of some  
 082 (possibly unknown) subset of the forget set  $\mathcal{D}_{\bullet} \subseteq \mathcal{D}_{\mathcal{F}}$  and hence still be effectively influenced by its  
 083 **approximate retain set**  $\mathcal{D}_{\mathcal{R}} \cup \mathcal{D}_{\bullet}$ . Note that  $\mathcal{D}_{\bullet}$  might not correspond exactly to the union of  $\mathcal{D}_i$ ’s  
 084 over some subset of data owners in  $\mathcal{F}$  and can possibly include only a subset of data points from each  
 085  $\mathcal{D}_i$ . The best unlearned models should have  $|\mathcal{D}_{\bullet}|$  and its influence to be as small as possible.

086 The model owner should allow each data owner  $i \in \mathcal{F}$  to evaluate the extent to which its data  $\mathcal{D}_i$  has  
 087 been unlearned, and would usually only grant them **query access to the model**. Let each data point  $d$   
 088 be used to form a text query  $q_d$ . For example,  $q_d$  can be a formatted prompt to an LLM for Q&A or  
 089 completion tasks. Then, both the model owner and data owner  $i$  can rely on some LLM  $\varphi_{\bullet}$ ’s text  
 090 output  $\varphi_{\bullet}(q_d)$  to compute an *unlearning metric*  $M$  that quantifies the extent to which  $i$ ’s data remains  
 091 present. We define an unlearning metric  $M$  where  $M(\varphi_{\bullet}(q_d), i)$  measures the influence of data  $\mathcal{D}_i$   
 092 from owner  $i$  (i.e., second input to  $M$ ) detectable in the LLM’s text output  $\varphi_{\bullet}(q_d)$  to query  $q_d$ .

093 The unlearning metric should also be able to measure the influence of data from a set of owners;  
 094 for example,  $M(\varphi_{\bullet}(q_d), \mathcal{F})$  measures the influence of the forget set  $\mathcal{D}_{\mathcal{F}}$  detectable in the LLM’s  
 095 text output. Usually, we set the influence as the extent to which data point  $d_i$  from some owner  
 096  $i \in \mathcal{F}$  remains present in the LLM’s text output  $\varphi_{\bullet}(q_{d_i})$  to its query  $q_{d_i}$ , i.e.,  $M(\varphi_{\bullet}(q_{d_i}), \mathcal{F}) =$   
 097  $M(\varphi_{\bullet}(q_{d_i}), i)$ . Often, we measure the influence of  $\mathcal{D}_{\mathcal{F}}$  detectable in the LLM’s text outputs to an  
 098 aggregate of queries formed by owners’ data (e.g.,  $\mathcal{D}_{\mathcal{F}}$ ). With a slight abuse of notation, we denote  
 099 such an *aggregate unlearning metric* as  $M_{d \in \mathcal{D}_{\mathcal{F}}}(\varphi_{\bullet}(q_d), \mathcal{F})$ , which can be, for example, the uniform  
 100 average over all  $d \in \mathcal{D}_{\mathcal{F}}$ :  $\sum_{d \in \mathcal{D}_{\mathcal{F}}} M(\varphi_{\bullet}(q_d), \mathcal{F}) / |\mathcal{D}_{\mathcal{F}}| = \sum_{i \in \mathcal{F}, d_i \in \mathcal{D}_i} M(\varphi_{\bullet}(q_{d_i}), \mathcal{F}) / |\mathcal{D}_{\mathcal{F}}|$ .

101 **Existing datasets to benchmark unlearning algorithms do not reflect practical challenges:**  
 102 Existing works have proposed to evaluate unlearning algorithms and metrics using benchmark  
 103 datasets like TOFU (Maini et al., 2024), MUSE (Shi et al., 2025), and WMDP (Li et al., 2024b) with  
 104 the following properties: (a) The forget set  $\mathcal{D}_{\mathcal{F}}$  and retain set  $\mathcal{D}_{\mathcal{R}}$  are fixed. In contrast, in practice,  
 105 multiple owners can independently decide whether to erase their data, which requires evaluation on  
 106 **multiple forget-retain splits**. (b)  $\mathcal{D}_{\mathcal{F}}$  and  $\mathcal{D}_{\mathcal{R}}$  are disjoint (i.e., the queries formed by  $\mathcal{D}_{\mathcal{F}}$  are related  
 107 only to  $\mathcal{D}_{\mathcal{F}}$  and are unrelated to queries formed by  $\mathcal{D}_{\mathcal{R}}$ ) and the unlearning algorithms perform  
 108 poorly if dependencies between both sets are introduced (Thaker et al., 2024). In contrast, real-world  
 109 datasets may **contain more similar data and different levels of similarity across  $\mathcal{D}_{\mathcal{F}}$  and  $\mathcal{D}_{\mathcal{R}}$** .

To address these limitations, we introduce a complementary unlearning benchmark dataset called WaterDrum-Ax that comprises arXiv paper abstracts across various academic subject categories published after the release of the Llama-2 model. In particular, to address (a) and (b) above, WaterDrum-Ax includes (a) abstracts from the 20 most popular academic subject categories to represent 20 different data owners that can be freely assigned to define  $\mathcal{D}_F$  and  $\mathcal{D}_R$ ; and (b) different levels of data similarity ranging from exact duplicates to paraphrased versions of the abstracts that can be used across  $\mathcal{D}_F$  and  $\mathcal{D}_R$ . Overall, WaterDrum-Ax contains 400 abstracts for each category, aggregating to a total of 8000 data points in WaterDrum-Ax. These abstracts have an average length of 260 tokens, which is considerably longer than that of TOFU (Maini et al., 2024) (59 tokens).

The WaterDrum-Ax benchmark dataset can be used to (i) evaluate unlearning metrics based on the desiderata introduced in Sec. 3, and (ii) evaluate unlearning algorithms using effective and practical metrics identified in (i). The empirical evaluations in Sec. 5 focus on (i) but include some preliminary results on (ii) in Sec. 5.1. We leave more systematic investigations of (ii) to future work.

**Existing unlearning metrics are ineffective in the face of practical challenges:** Here, we discuss some existing definitions of the unlearning metric  $M$  and their limitations; see App. A.1 for a deeper introduction of utility-centric and other unlearning metrics. *Utility-centric* unlearning metrics have evaluated the effectiveness of unlearning based on model utility (performance) indicators, such as verbatim memorization, perplexity, or accuracy on downstream tasks. Performance indicators  $P$  have compared the unlearned LLM  $\tilde{\varphi}$ 's text outputs to queries (e.g.,  $\tilde{\varphi}(q_d)$  for all  $d \in \mathcal{D}_F$ ) to the original text data (e.g.,  $\mathcal{D}_F$ ). For instance, ROUGE-L (Maini et al., 2024) compares the output phrasing/longest common subsequence of  $\tilde{\varphi}(q_d)$  to the training text data point  $d$ . As another example, some membership inference attack (MIA) based unlearning metrics (Shokri et al., 2017), such as that of Shi et al. (2024) for LLMs, are utility-centric as they may depend on the log-likelihood of tokens of the original text data.

Our key observation is that **utility-centric metric values**, such as  $P(\tilde{\varphi}(q_d), \mathcal{F})$  for each  $d \in \mathcal{D}_F$  and their aggregate (e.g., average for ROUGE-L), **cannot be interpreted on their own**: For example, Table 3 of Shi et al. (2025) compares the aggregate metric value (e.g., KnowMem) on the unlearned LLM  $\tilde{\varphi}$  with that on the retrained LLM  $\varphi_R$  (trained on the retain set) to evaluate the extent of unlearning. Ideally, the aggregate (over all  $d \in \mathcal{D}_F$ ) of  $P(\tilde{\varphi}(q_d), \mathcal{F})$  should be equal to that of  $P(\varphi_R(q_d), \mathcal{F})$ . Similarly, the work of Maini et al. (2024) also compares the distribution of the metric values (e.g., Truth Ratio, ROUGE-L) on the unlearned LLM  $\tilde{\varphi}$  with that on the retrained LLM  $\varphi_R$  to evaluate the extent of unlearning.

This raises a critical issue: In practice, the retrained LLM  $\varphi_R$  is usually not available (Sec. 2). In fact, the aim of unlearning algorithms (and metrics) is to obtain an unlearned LLM  $\tilde{\varphi}$  that most closely approximates  $\varphi_R$ . Regarding our main research question, **is there an (aggregate) unlearning metric (over all  $d \in \mathcal{D}_F$ ) whose values can be interpreted on their own to measure the extent of unlearning  $\mathcal{D}_F$  without referencing a retrained LLM?**

It can be observed from Fig. 1 that **the answer is no for utility-centric metrics**, especially when there are similar data in the retain and forget sets: For any Truth Ratio metric value (e.g., 0.2), the LLM's text output to a query is equally likely to be formed by a data point in the retain set vs. that in the forget set. There is no value of  $\kappa$  where we can confidently conclude that the LLM's text output  $\varphi_R(q_d)$  to query  $q_d$  is more likely to be formed by any  $d \in \mathcal{D}_F$  when  $P(\varphi_R(q_d), \mathcal{F}) < \kappa$ . Let  $d_i \simeq d_j$  denote that text data points  $d_i$  and  $d_j$  have a large *similarity score*  $SS(d_i, d_j)$ , e.g., computed using some semantic text similarity (STS) score. A likely explanation is that when similar text data points  $d_f$  and  $d_r$  are present in the respective forget and retain sets, (i) any unlearned LLM  $\tilde{\varphi}$  (e.g., retrained LLM  $\varphi_R$ ) tends to produce similar text outputs to queries formed by both sets, i.e.,

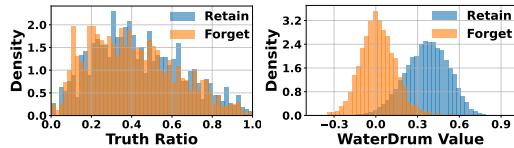


Figure 1: Histograms of utility-centric Truth Ratio metric vs. WaterDrum values under the ‘semantic duplicate’ setting of data similarity for WaterDrum-TOFU dataset (Table 2 in Sec. 5) where the individual metric values  $M(\varphi_R(q_d), \mathcal{F})$  for each  $d \in \mathcal{D}_F$  are in orange and  $M(\varphi_R(q_d), \mathcal{R})$  for each  $d \in \mathcal{D}_R$  are in blue. The Truth Ratio metric values cannot be interpreted on their own as there is no value on the horizontal axis where we can confidently conclude that the LLM's text output  $\varphi_R(q_d)$  to query  $q_d$  is more likely to be formed by any  $d \in \mathcal{D}_F$ . In contrast, WaterDrum values can be interpreted on their own: Values  $< 0.2$  are more likely to be associated with forget set  $\mathcal{D}_F$ .

162  $\tilde{\varphi}(q_{d_f}) \simeq \tilde{\varphi}(q_{d_r})$ , as empirically verified in App. G.2. As performance indicators largely depend on  
 163 direct comparisons with the LLM’s text outputs, their metric values are also similar. (ii) Expecting  
 164 poor predictions on the forget set overlooks the generalization capability of LLMs (Liu et al., 2024).  
 165

### 166 3 UNLEARNING METRIC DESIDERATA

168 The goal of our work here is to come up with an alternative effective and practical unlearning metric  
 169 whose values can be interpreted on their own *without* referencing a retrained LLM. **What desiderata**  
 170 **must such an unlearning metric satisfy?** We define a few non-exhaustive desiderata in this section.

171 Intuitively, based on Fig. 1, we would want the LLM’s text outputs to queries formed by the text  
 172 data points in the forget vs. retain sets to have (i) *separable metric values* and (ii) aggregate (e.g.,  
 173 average) metric values to be easily interpreted (e.g., 0 for perfect unlearning) (iii) without referencing  
 174 the retrained model. (i) and (ii) correspond to our desiderata **D1** and **D2**, respectively. We include  
 175 practical constraint (iii) as **D3**. The unlearning metric  $M$  should satisfy the following desiderata:

176 **EFFECTIVENESS.** First, the metric must effectively measure the extent to which an unlearning  
 177 algorithm has not unlearned the forget set (so, the resulting unlearned LLM  $\tilde{\varphi}$  would still be influenced  
 178 by its unknown approximate retain set, as discussed in Sec. 2). To achieve this, we will now define  
 179 effectiveness desiderata that utilize LLMs retrained on the retain set (and varying known subsets of  
 180 the forget set) as retraining is a perfect unlearning algorithm.<sup>1</sup>

181 **D1 Separability.** The metric should detect/classify whether an owner’s data still influences an  
 182 unlearned LLM. Specifically, when evaluating the retrained LLM  $\varphi_R$  (i.e., achieved by perfect  
 183 unlearning), the metric should, with high probability, **give higher values when measured on its text**  
 184 **outputs to queries formed by the retain set  $\mathcal{D}_R$**  (which influences  $\varphi_R$ ) **than queries formed by the**  
 185 **forget set  $\mathcal{D}_F$**  (which does not). That is, for any randomly selected text data points  $d_r \in \mathcal{D}_r \subseteq \mathcal{D}_R$   
 186 from owner  $r$  and  $d_f \in \mathcal{D}_f \subseteq \mathcal{D}_F$  from owner  $f$ , the probability

$$187 \mathbb{P}[M(\varphi_R(q_{d_r}), r) > M(\varphi_R(q_{d_f}), f)] \approx 1. \quad (1)$$

189 Separability, which is defined by the left-hand side expression of Eq. (1) (or, equivalently, AUROC),  
 190 implies that some threshold  $\kappa$  exists such that for any text data point  $d_i \in \mathcal{D}_i \subseteq \mathcal{D}_T$  from owner  
 191  $i$ , a large value  $M(\varphi_R(q_{d_i}), i) > \kappa$  indicates that  $d_i$  is likely to be in the retain set  $\mathcal{D}_R$ ; varying  $\kappa$   
 192 yields the ROC curve. Similarly, when considering an unlearned LLM  $\tilde{\varphi}$ , a large value  $M(\tilde{\varphi}(q_{d_i}), i)$   
 193 indicates that  $d_i$  is likely to be in the approximate retain set (Sec. 2). In other words, the metric  
 194 should serve as a good classifier for whether an owner’s data still influences the LLM and is hence in  
 195 the approximate retain set: A higher AUROC indicates a better separability of data that influences the  
 196 LLM vs. not (Fawcett, 2006). App. B.1 gives a [more intuitive and detailed](#) discussion on **D1**.

197 **D2 Calibration.** In Sec. 1, we have highlighted that existing unlearning algorithms cannot yet  
 198 achieve perfect unlearning. Thus, our unlearning metric should be **calibrated to the extent of**  
 199 **imperfect unlearning**. For example, we can simulate different extents of imperfect unlearning by  
 200 retraining with different sizes of subsets of the forget set. Specifically, the aggregate metric (in  
 201 expectation) should be proportional to the size  $k$  of the random subset  $\mathcal{D}_\bullet$  of the forget set that is  
 202 used together with the retain set  $\mathcal{D}_R$  to retrain the LLM  $\tilde{\varphi}$ :

$$203 \mathbb{E}_{\mathcal{D}_\bullet \subseteq \mathcal{D}_F: |\mathcal{D}_\bullet| = k} [M_{d \in \mathcal{D}_F}(\tilde{\varphi}(q_d), \mathcal{F})] \propto k/|\mathcal{D}_F| \quad (2)$$

205 where  $\mathcal{D}_\bullet$  is defined in a similar way as  $\mathcal{D}_\bullet$  in Sec. 2 except that it is known. Eq. (2) implies that  
 206 a perfectly unlearned LLM like  $\varphi_R$  should have  $M_{d \in \mathcal{D}_F}(\varphi_R(q_d), \mathcal{F}) = 0$  since  $k = 0$ . So, when  
 207 evaluating unlearning algorithms, we identify successful perfect unlearning of the forget set by  
 208 looking for  $M_{d \in \mathcal{D}_F}(\tilde{\varphi}(q_d), \mathcal{F}) \approx 0$ . In addition, the value of the aggregate metric can be intuitively  
 209 interpreted as the extent to which the forget set has not been unlearned. This enables the unlearning  
 210 metric to go *beyond being just a binary indicator* of whether an entire forget set has been unlearned  
 211 to a meaningful *continuous measure* of unlearning. App. B.2 provides a further discussion on **D2**.

212 **PRACTICALITY.** To be a viable metric for deployment, the metric must also satisfy the following  
 213 feasibility and robustness desiderata that account for challenges faced in common real-life scenarios:

214 <sup>1</sup>Note that these retrained LLMs are only used to justify our effectiveness desiderata for evaluating the  
 215 unlearning metrics. In practice, the metrics should be used to evaluate imperfect unlearning algorithms without  
 referencing the retrained LLMs, as discussed in **D3(a)**.

216 **D3 Feasibility.** (a) When the metric is used to evaluate an unlearning algorithm and produce  
 217  $M(\tilde{\varphi}(q_d), \mathcal{F})$  or the aggregate  $M_{d \in \mathcal{D}_F}(\tilde{\varphi}(q_d), \mathcal{F})$  on the unlearned LLM  $\tilde{\varphi}$ , it **should not require**  
 218 **the retrained LLM  $\varphi_R$  to interpret/measure the extent of imperfect unlearning**. The premise  
 219 of unlearning is that retraining the LLM on the retain set is prohibitively expensive. Hence, metrics  
 220 cannot depend on  $\varphi_R$  in practice. (b) To additionally enable data owners with only query access to  
 221 the LLM to evaluate unlearning, the metric **should only depend on the queried text outputs** instead  
 222 of full access to the weights or token probabilities of the unlearned model  $\tilde{\varphi}$ .

223 **D4 Robustness to similar data.** The effectiveness desiderata **D1-D2** should hold for any  $\mathcal{D}_R$  and  
 224  $\mathcal{D}_F$ , including typical scenarios where  $\mathcal{D}_R$  and  $\mathcal{D}_F$  have similar data points (e.g., news agencies  
 225 have different news articles reporting on the same event, as illustrated in App. B.4).

226 EXISTING METRICS DO NOT SATISFY DESIDERATA. Continuing the discussion from Sec. 2,  
 227 Table 1 compares our WaterDrum and existing metrics based on the proposed desiderata in Sec. 3.  
 228 As other utility-centric metrics may not satisfy **D1** and **D2** under **D3** and **D4**, their values cannot  
 229 directly interpret/measure the extent of imperfect unlearning without a retrained LLM and using the  
 230 WaterDrum- $\Delta$ X dataset (Sec. 2).

## 4 WATERMARKING FRAMEWORK

231  
 232 Instead of relying on utility-centric metrics that indirectly  
 233 infer unlearning via model performance, we propose  
 234 a novel *data-centric* metric that **directly tracks the**  
 235 **influence of data by actively embedding data-specific**  
 236 **signals detectable in the LLM’s text outputs**. These  
 237 data signals are embedded by watermarking the training  
 238 data and preserved by the LLM. We discuss how  
 239 WaterDrum differs from existing watermark-based  
 240 metrics for image classification tasks in App. A.1 and  
 241 give an introduction of text watermarking in App. A.3.  
 242 We will start by outlining desiderata required by a  
 243 watermarking framework (and its verification operator)  
 244 to meet our unlearning metric desiderata in Sec. 3.  
 245

246 **WATERMARKING DESIDERATA.** Our watermarking framework assigns each data owner  $i$  a  
 247 watermark key  $\mu_i$ . It comprises (a) a **watermarking operator**  $\mathcal{W}(d_i, \mu_i) \rightarrow d'_i$  that takes in  
 248 any text data point  $d_i \in \mathcal{D}_i$  from owner  $i$  and watermarks it with the key  $\mu_i$  to produce a unique  
 249 corresponding text data point  $d'_i$ , and (b) a **verification operator**  $\mathcal{V}(g', \mu_i)$  that takes in any text data  
 250  $g'$  (e.g., LLM’s text output) and a watermark key  $\mu_i$  and provides a score reflecting the likelihood of  
 251  $g'$  containing the watermark  $\mu_i$ . To satisfy our unlearning metric desiderata in Sec. 3, the watermark  
 252 and verification operators used will need to satisfy the following desiderata:<sup>2</sup>

253 **W0 Fidelity.** The watermarking should have minimal impact on the semantic similarity of the  
 254 original data, i.e.,  $d \simeq \mathcal{W}(d, \mu)$  for any watermark key  $\mu$  and data  $d \in \mathcal{D}_T$ . While this does not  
 255 directly impact the unlearning metric desiderata, **W0** ensures that the watermarking process preserves  
 256 the value of the data and model for the model owner and the metric can be deployed in practice.

257 **W1 Verifiability.** (a) The watermark should be verifiable if and only if the watermarked content  
 258 is present in the LLM. In our setting, this implies that the retrained LLM should not contain the  
 259 watermark of an owner  $f$  in  $\mathcal{F}$  who requested to erase its data, i.e.,  $\mathcal{V}(\varphi_R(q_{d_f}), \mu_f) = 0$ . In contrast,  
 260 an LLM that has been trained on owner  $r$ ’s data  $\mathcal{D}_r \subseteq \mathcal{D}_R$  should be verifiable with watermark key  
 261  $\mu_r$ , i.e.,  $\mathcal{V}(\varphi_R(q_{d_r}), \mu_r) \gg 0$  for all  $d_r \in \mathcal{D}_r$ . (b) If every text data point in  $\mathcal{D}_F$  is watermarked  
 262 with the same key  $\mu_F$ , the average of  $\mathcal{V}(\tilde{\varphi}(q_{d_f}), \mu_F)$  over all  $d_f \in \mathcal{D}_F$  for model  $\tilde{\varphi}$  retrained on  
 263  $\mathcal{D}_R \cup \mathcal{D}_\bullet$  should be proportional to the size of the data  $\mathcal{D}_\bullet \subseteq \mathcal{D}_F$ . (a) supports **D1** as  $\mathcal{V}(\varphi_R(q_{d_i}), \mu_i)$   
 264 can be used to classify whether an owner’s data influences a perfectly unlearned LLM — a value near  
 265 0 or much larger than 0, respectively, indicates that owner  $i$  likely has no influence or some influence  
 266 on the unlearned LLM. Together, (a) and (b) support **D2** as the value is 0 in the case of a perfectly  
 267 unlearned LLM like  $\varphi_R$  and the average value is proportional to the extent of imperfect unlearning.

Table 1: Comparison of unlearning the metrics based on the proposed desiderata (Sec. 3). We enforce **D3**, so metrics cannot rely on the retrained LLM. **D1** and **D2** consider the setting of no data similarity.

	<b>D1</b>	<b>D2</b>	<b>D4</b>
ROUGE (Maini et al., 2024)	✓	✗	✗
Truth Ratio (Maini et al., 2024)	✗	✗	✗
KnowMem (Shi et al., 2025)	✗	✗	✗
MIA (Shi et al., 2024)	✗	✗	✗
WaterDrum (ours)	✓	✓	✓

<sup>2</sup>When evaluating unlearning algorithms (Sec. 5.1), the model owner can perform the watermarking and verification. In real-world deployment, the data owners do so instead.

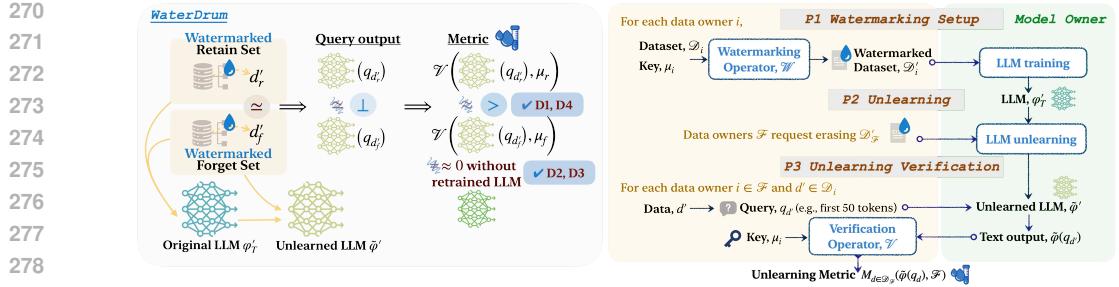


Figure 2: (Left) Unlike existing utility-centric metrics, WaterDrum satisfies the unlearning metric desiderata in Sec. 3. WaterDrum is robust to similar data by embedding orthogonal data-specific signals in the LLM’s text outputs that are **W1** verifiable. (Right) An overview of the watermarking, training, unlearning, and verification processes of WaterDrum.

**W2 Overlap verifiability.** The verifiability desideratum **W1** is satisfied despite the presence of other watermarks (e.g.,  $\mu_r$  from another owner  $r$ ) in the data for training the LLM. This allows for multiple watermarks to be verified from the text outputs of the same LLM.

We also need desiderata on the watermarking process to meet the rest of unlearning metric desiderata:

**W3 Query access constraint.** Data owners should verify their watermarks with only query access to the LLM. This supports **D3** with feasible & efficient evaluation of the extent of imperfect unlearning.

**W4 Unique key.** Each data owner  $i$ ’s watermark key  $\mu_i$  should be unique. When a data owner requests to erase its data, the corresponding forget set would have a different watermark from that associated with the retain set, thus supporting **D1**. Furthermore, the unique keys ensure that similar or even identical data from different owners would have different watermarks, which supports **D4**.

Fig. 2(left) shows how a watermarking framework satisfying these desiderata satisfies the unlearning metric desiderata in Sec. 3. Concretely, we define a metric  $M'$  using the verification operator:

$$M'(\varphi_\bullet(q_d), i) := \mathcal{V}(\varphi_\bullet(q_d), \mu_i). \quad (3)$$

**OVERVIEW OF WATERDRUM.** Can any watermarking framework be adapted to satisfy our proposed watermarking desiderata above? Here, we propose the first data-centric unlearning metric called WaterDrum built on top of our adaptation of the training-free, scalable, and robust Waterfall framework (Lau et al., 2024) that can successfully and efficiently verify multiple data owners’ watermarks in the text outputs of the LLM when trained on their watermarked text data. While in this paper we will mainly use Waterfall to demonstrate the effectiveness and practicality of WaterDrum, other watermarking methods satisfying our desiderata can be adopted as well. We provide a comprehensive overview of watermarking methods, such as (Kirchenbauer et al., 2023; Dathathri et al., 2024; Kuditipudi et al., 2024), in App. A.3 and discuss their adaptations and empirical performance in Sec. 5.2.

Specifically, we adopt the watermarking  $\mathcal{W}(\cdot, \mu)$  and verification  $\mathcal{V}(\cdot, \mu)$  operators as defined in Waterfall (respectively, Algorithms 1 and 2) and summarized in App. C.1 due to lack of space. Waterfall’s watermarking and verification operators satisfy the watermarking desiderata **W0**, **W1(a)**, and **W2**, as elaborated and demonstrated in (Lau et al., 2024). We have empirically verified that the Waterfall method satisfies **W0** in App. G.1 and **W1(b)** on calibration in Sec. 5. The rest of the watermarking desiderata can be satisfied by an appropriate design of the unlearning and verification processes, which we illustrate in Fig. 2(right) and present below:

**P1 Watermarking setup.** Each data owner  $i$  first watermarks its data  $D_i$  with a unique key  $\mu_i$  to generate a watermarked dataset  $D'_i := \{d'_i := \mathcal{W}(d_i, \mu_i)\}_{d_i \in D_i}$ . Then, the model owner aggregates their watermarked data  $D'_T := \bigcup_{i \in T} D'_i$ , trains an LLM  $\varphi'_T$  on it, and offers to clients (including data owners) query access to the trained LLM.

**P2 Unlearning.** A subset of data owners  $F$  requests for their data  $D'_F := \bigcup_{i \in F} D'_i$  to be erased from the LLM  $\varphi'_T$ . The model owner will claim to have performed the unlearning and offer query access to the resulting unlearned LLM  $\tilde{\varphi}'$ .

**P3 Unlearning verification.** The verification operator plays the role of an unlearning metric in WaterDrum, as per Eq. (3). Each data owner  $i$  in  $F$  can query the unlearned LLM  $\tilde{\varphi}'$  with queries

324  $q_{d'}$  based on  $d' \in \mathcal{D}'_i$  and apply the verification operator  $\mathcal{V}(\tilde{\varphi}'(q_{d'}), \mu_i)$  to measure the extent to  
 325 which its data remains present in the text outputs  $\tilde{\varphi}'(q_{d'})$  and hence has not been unlearned.  
 326

327 Note that computing the WaterDrum value in Eq. (3) applied during **P3** only requires query access  
 328 to the model, hence satisfying **W3**. Watermarking desideratum **W4** is also satisfied by the setup in **P1**  
 329 and the fact that the model owner never requires the data owners’ keys, which is also the case in **P2**.  
 330

331 *Remark 1.* Using watermarked data is both (i) **beneficial and important for identifying practical**  
 332 **and effective unlearning metrics** and (ii) **reasonable going forward**. (i) In Table 1, our watermarked  
 333 data-based WaterDrum is the *only* metric that satisfies all the unlearning metric desiderata. (ii) There  
 334 are a few important reasons: (a) data owners with IP or privacy rights (Sec. 1) can require the model  
 335 owner to use the watermarked version of their released data instead; (b) data owners can watermark  
 336 their unreleased data, which will be used to train (and may be more relevant for) future LLMs,  
 337 and (c) the adoption of text watermarking is expected to grow and match the prevalence of image  
 338 watermarking. In App. C.4, we elaborate on these practical considerations and other benefits (e.g.,  
 339 computationally lightweight, no change to existing ML pipelines), beyond meeting the desiderata,  
 340 for deploying WaterDrum. So, the benefits of using watermarked data in WaterDrum to evaluate  
 341 unlearning algorithms, such as not needing to reference a retrained LLM (unlike utility-centric  
 342 metrics), outweigh the slight inconvenience and cost.  
 343

344 *Remark 2.* If the (i) model owner tries to reduce its LLM’s metric value without directly performing  
 345 unlearning or copyright its LLM via **watermarking methods for a model owner** (Kirchenbauer et al.,  
 346 2023) or (ii) data owners try to report inflated metric values to understate the unlearning by the model  
 347 owner, is WaterDrum still an effective unlearning metric? The answer is yes if (i) the underlying  
 348 watermarking framework is designed to be *resilient*, the watermark keys are *private* to the data  
 349 owners, and (ii) a trusted third party validates the metric values. We discuss these questions and  
 350 additional requirements in App. D and show that WaterDrum also satisfies them.  
 351

## 352 5 EXPERIMENTS AND DISCUSSION

353 **Experimental setup.** In this section, we empirically compare WaterDrum with other commonly  
 354 used unlearning metrics: ROUGE-L (Lin, 2004; Maini et al., 2024), Truth Ratio (Maini et al., 2024),  
 355 KnowMem (Shi et al., 2025), and MIA (Shi et al., 2024)). We use the Llama-2-7B (Touvron et al.,  
 356 2023) as the base model. For WaterDrum, the LLM is fine-tuned on the watermarked dataset  $\mathcal{D}'_{\mathcal{T}}$   
 357 in WaterDrum-Ax (Sec. 2) or WaterDrum-TOFU derived from TOFU (Maini et al., 2024) (MIT  
 358 License). For other metrics, the LLM is instead fine-tuned on their unwatermarked version  $\mathcal{D}_{\mathcal{T}}$ .  
 359 To ease comparison, all metrics are scaled to 1.0 when evaluated on the original model  $\varphi_{\mathcal{T}}$  before  
 360 unlearning. We use 1 category from WaterDrum-Ax and 10% data from WaterDrum-TOFU  
 361 as forget sets. We further evaluate the metrics with other LLMs (Li et al., 2023b) as base models  
 362 (App. F.3). Although watermarking is only essential for WaterDrum, App. G.1 shows that it does not  
 363 degrade LLM performance and App. F.4 shows that other metrics still do not satisfy some desiderata  
 364 when the LLM is fine-tuned on  $\mathcal{D}'_{\mathcal{T}}$  instead. To ease notation, in the rest of this paper, we will use  $d_{\bullet}$ ,  
 365  $\mathcal{D}_{\bullet}$ ,  $q_{\bullet}$ ,  $\varphi_{\bullet}$ , and  $\tilde{\varphi}$  in place of  $d'_{\bullet}$ ,  $\mathcal{D}'_{\bullet}$ ,  $q'_{\bullet}$ ,  $\varphi'_{\bullet}$ , and  $\tilde{\varphi}'$  (i.e., those associated with the watermarked data  
 366 used by WaterDrum), respectively. App. E gives additional details on the datasets, other models  
 367 used, unlearning metrics, inference parameters, queries, and implementation.  
 368

369 We will evaluate WaterDrum and the baseline metrics in experimental settings that mimic the  
 370 real-life scenarios described in the PRACTICALITY DESIDERATA **D3** and **D4** (Sec. 3). Then, under  
 371 these settings, we assess the effectiveness of various metrics based on **D1** and **D2** by considering how  
 372 they evaluate the perfect unlearning algorithm – retraining the base model on only the retain set to  
 373 obtain  $\varphi_{\mathcal{R}}$ , which is guaranteed to contain no influence of forget set  $\mathcal{D}_{\mathcal{F}}$  by construction.  
 374

375 **Feasibility D3.** To satisfy **D3(a)**, the metrics should not require referencing the retrained LLM  $\varphi_{\mathcal{R}}$  to  
 376 interpret/measure the extent of imperfect unlearning (Sec. 3). For example, when assessing **D2**, we  
 377 enforce the metric values not to use (e.g., subtract)  $M_{d \in \mathcal{D}_{\mathcal{F}}}(\varphi_{\mathcal{R}}(q_d), \mathcal{F})$ . To satisfy **D3(b)**, the metric  
 378 should not require logit access, but for evaluation, we allow the use of logits only to compute MIA.  
 379

380 **Robustness to similar data D4.** Let  $\mathcal{D}_i \simeq \mathcal{D}_j$  denote sets where for any  $d_i \in \mathcal{D}_i$ , there is a  
 381 corresponding  $d_j \in \mathcal{D}_j$  such that  $d_i \simeq d_j$ . We establish the settings to assess the robustness of  
 382 the unlearning metrics to similar data by injecting a small amount of data  $\mathcal{D}_s \simeq \mathcal{D}_{\mathcal{F}}$  into  $\mathcal{D}_{\mathcal{R}}$ , i.e.,  
 383 the retain set is augmented ( $\mathcal{D}_{\mathcal{R}}^s := \mathcal{D}_s \cup \mathcal{D}_{\mathcal{R}}$ ) with some data points that are similar to  $\mathcal{D}_{\mathcal{F}}$ . We  
 384

378 Table 2: AUROC ( $\pm$  across 3 seeds) of various unlearning metrics under different levels of data  
 379 similarity for the WaterDrum-TOFU and WaterDrum-Ax datasets. WaterDrum’s AUROC  
 380 remains near 1.0 even when similar data exists.

Data Similarity	WaterDrum-TOFU			WaterDrum-Ax		
	ROUGE	Truth Ratio	WaterDrum	ROUGE	KnowMem	WaterDrum
Exact Duplicate	0.510 $\pm$ 0.007	0.508 $\pm$ 0.008	<b>0.926<math>\pm</math>0.027</b>	0.334 $\pm$ 0.005	0.492 $\pm$ 0.005	<b>0.957<math>\pm</math>0.008</b>
Semantic Duplicate	0.798 $\pm$ 0.001	0.472 $\pm$ 0.054	<b>0.954<math>\pm</math>0.001</b>	<b>0.960<math>\pm</math>0.002</b>	0.450 $\pm$ 0.007	<b>0.963<math>\pm</math>0.001</b>
No Duplicate	<b>0.908<math>\pm</math>0.005</b>	0.747 $\pm$ 0.011	<b>0.928<math>\pm</math>0.026</b>	<b>0.974<math>\pm</math>0.001</b>	0.491 $\pm$ 0.008	<b>0.965<math>\pm</math>0.002</b>

387  
 388 consider two such settings: (a) **exact duplicate**: data points in  $\mathcal{D}_s$  are exact copies of those in  $\mathcal{D}_F$   
 389 (i.e.,  $\mathcal{D}_s = \mathcal{D}_F$ ), and (b) **semantic duplicate**: data points in  $\mathcal{D}_s$  are paraphrased versions of those in  
 390  $\mathcal{D}_F$  (i.e.,  $\mathcal{D}_s \simeq \mathcal{D}_F$ ). In addition, we consider the case where (c) **no duplicate** of any data point in  
 391  $\mathcal{D}_F$  is used to augment  $\mathcal{D}_R$  (i.e.,  $\mathcal{D}_s = \emptyset$ ). Additional implementation details are in App. E.4.

392 **SEPARABILITY DESIDERATUM D1.** To assess whether the unlearning metrics satisfy **D1**, note  
 393 that the left-hand side expression  $\mathbb{P}[M(\varphi_R(q_{d_r}), r) > M(\varphi_R(q_{d_f}), f)]$  in Eq. (1) corresponds to the  
 394 definition of the AUROC of the metric  $M$  in measuring the separability of the retain set  $\mathcal{D}_R$  that  
 395 influences the retrained LLM  $\varphi_R$  vs. the forget set  $\mathcal{D}_F$  that does not (Fawcett, 2006). Hence, we  
 396 can compute the AUROC of various unlearning metrics with the retrained LLM  $\varphi_R$  (i.e., a perfectly  
 397 unlearned LLM) and assess if they have  $\text{AUROC} \approx 1$ . We exclude MIA from this experiment because  
 398 it focuses solely on assessing privacy leakage based on distributional differences between forget and  
 399 holdout sets without considering the retain set.

400 Table 2 shows the AUROC of the various unlearning metrics under different levels of data similarity for  
 401 the WaterDrum-TOFU dataset.<sup>3</sup> Notably, WaterDrum is the only metric that consistently achieves  
 402  $\text{AUROC} > 0.9$  and close to 1, hence satisfying **D1**. In contrast, the other metrics’ performances  
 403 degrade significantly under the ‘exact and semantic duplicate’ settings; for the former, their AUROCs  
 404 drop to around 0.5, so the other metrics are no better than random assignment in the separability of  $\mathcal{D}_R$   
 405 vs.  $\mathcal{D}_F$ . Furthermore, Truth Ratio only achieves an AUROC of around 0.75 under the conventional  
 406 ‘no duplicate’ setting, hence not satisfying **D1** even in this case.

407 The results on the WaterDrum-Ax dataset in Table 2 show similar trends with WaterDrum  
 408 consistently performing well and KnowMem performing poorly in all settings. ROUGE performs  
 409 poorly under the ‘exact duplicate’ setting where only 5% of the augmented retain set are exact copies  
 410 of those in the forget set. It performs well for the ‘semantic duplicate’ setting as the mean ROUGE-L  
 411 recall score between  $\mathcal{D}_s$  and  $\mathcal{D}_F$  is low ( $\approx 0.65$ ), which implies that the text data is already heavily  
 412 paraphrased such that the ‘semantic duplicate’ setting is effectively closer to the ‘no duplicate’ one  
 413 for ROUGE. However, the mean semantic text similarity (STS) score of  $\mathcal{D}_s$  and  $\mathcal{D}_F$  remains high  
 414 (i.e., 0.94). Milder forms of perturbation for this dataset would likely make the degradation of its  
 415 performance on **D1** more apparent.

416 **CALIBRATION DESIDERATUM D2.** Next, we assess whether the unlearning metrics meet the  
 417 calibration desideratum, as defined in Eq. (2). Failing to meet this desideratum implies that the  
 418 metrics cannot measure the extent to which the forget set  $\mathcal{D}_F$  has not been unlearned (i.e., for  
 419 imperfect unlearning). To evaluate this, we first retrain LLMs on  $\mathcal{D}_R \cup \mathcal{D}_\bullet$  by varying the size  $k$  of  
 420 the subset  $\mathcal{D}_\bullet \subseteq \mathcal{D}_F$ . Then, we compute the unlearning metrics for each retrained LLM and plot  
 421 calibration curves showing how the metrics vary with  $k$ . To quantify how well a metric satisfies  
 422 Eq. (2), we can compute the  $R^2$  value for its best-fit line through the origin since a calibrated metric  
 423 (in expectation) should be proportional to  $k/|\mathcal{D}_F|$  and have  $M_{d \in \mathcal{D}_F}(\varphi_R(q_d), \mathcal{F}) = 0$  at  $k = 0$ .  
 424 Consequently, an  $R^2$  value close to 1 implies that the metric is well-calibrated, while a large negative  
 425 value occurs when the metric produces similar (instead of proportional) values for varying  $k$ .

426 Fig. 3 and Table 3 show, respectively, the calibration curves for the various unlearning metrics and  
 427 the  $R^2$  values for the corresponding best-fit lines under different levels of data similarity for the  
 428 WaterDrum-Ax dataset. The results show that WaterDrum is the only well-calibrated metric  
 429 across all settings that can represent the proportion  $k/|\mathcal{D}_F|$  of the forget set still influencing the

430 <sup>3</sup>Truth Ratio is only applicable to question answering (QA) datasets for which WaterDrum-Ax is not.  
 431 Since WaterDrum-TOFU is already a QA dataset, there is no need to consider KnowMem that generates QA  
 432 pairs for evaluation using the ROUGE-L recall score.

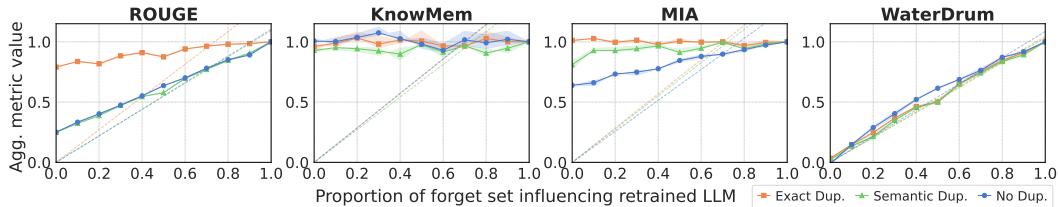


Figure 3: Calibration curves for various unlearning metrics w.r.t. proportion  $k/|\mathcal{D}_F|$  of forget set influencing retrained LLM (solid) and their best-fit lines (see associated  $R^2$  in Table 3) through origin (dotted) under different levels of data similarity for WaterDrum-Ax dataset. Only WaterDrum is well-calibrated and satisfies **D2** with its best-fit lines closely following its aggregate metric values.

Table 3:  $R^2$  values for the best-fit lines (dotted in Fig. 3) of various unlearning metrics under different levels of data similarity for WaterDrum-Ax dataset. WaterDrum achieves the highest  $R^2$  values that are close to 1 and is hence a well-calibrated metric.

Data Similarity	ROUGE	KnowMem	MIA	WaterDrum
Exact Duplicate	-37.47	-498.1	-1220	<b>0.987</b>
Semantic Duplicate	0.693	-276.5	-90.21	<b>0.991</b>
No Duplicate	0.650	-252.9	-7.553	<b>0.963</b>

unlearned LLM (i.e., the extent of imperfect unlearning). Comparatively, other metrics perform poorly across *all settings*, including the conventional ‘no duplicate’ setting — they cannot be used to determine when  $\mathcal{D}_F$  has been perfectly unlearned as  $M'_{d \in \mathcal{D}_F}(\varphi_R(q_d), \mathcal{F}) \neq 0$  (i.e., calibration curves do not pass through the origin in Fig. 3). Thus, they do not satisfy **D2** when enforcing **D3**.

The results demonstrate the strong reliance of the baseline unlearning metrics on access to the retrained LLM  $\varphi_R$ . Without knowing the reference value on the perfectly unlearned LLM (i.e.,  $\varphi_R$ ), these baselines fail to quantify the extent of imperfect unlearning or even evaluate the success of unlearning. This reliance is impractical as retraining the LLM on the retain set is prohibitively expensive, which motivates the need for unlearning algorithms. Fig. 14 and Table 13 in App. H.2.1 show similar results for the WaterDrum-TOFU dataset where all baselines fail to meet the calibration desideratum across all settings, including the ‘no duplicate’ setting. App. H.1.1 gives more results.

### 5.1 BENCHMARKING UNLEARNING ALGORITHMS ON NEW WATERDRUM-AX DATASET

Finally, Fig. 4 illustrates how WaterDrum can be used to benchmark unlearning algorithms using WaterDrum-Ax via an evaluation plot of  $M'_{d \in \mathcal{D}_F}(\tilde{\varphi}(q_d), \mathcal{F})$  vs.  $M'_{d \in \mathcal{D}_R}(\tilde{\varphi}(q_d), \mathcal{R})$ . This measures the aggregate WaterDrum values of the respective watermarked forget set  $\mathcal{D}_F$  vs. retain set  $\mathcal{D}_R$  on the unlearned LLM  $\tilde{\varphi}$  (Sec. 4). The original LLM  $\varphi_T$ , which trains on both  $\mathcal{D}_F$  and  $\mathcal{D}_R$  (i.e., no unlearning), is at the top right corner, while the retrained LLM  $\varphi_R$ , which only trains on  $\mathcal{D}_R$  (i.e., perfect unlearning), is at the bottom right corner. It is expected that perfect unlearning would achieve an approximately similar aggregate WaterDrum value of  $\mathcal{D}_R$  as no unlearning since the retain set still influences both the retrained and original LLMs. In this plot, if an unlearning algorithm can produce an unlearned LLM  $\tilde{\varphi}$  with aggregate WaterDrum values closer to that achieved by retraining, then its  $\tilde{\varphi}$  is better at both unlearning  $\mathcal{D}_F$  from  $\varphi_T$  and retaining the influence of  $\mathcal{D}_R$ .

Fig. 4 shows results for unlearning algorithms such as Gradient Descent (GD) on  $\mathcal{D}_R$  from  $\varphi_T$ , KL Minimization (KL) (Maini et al., 2024), Task Vector (TV) (Ilharco et al., 2023), SCRUB (Kurmanji et al., 2024), details of which are in App. E.6. It can be observed that they achieve aggregate WaterDrum values still far from that achieved by retraining: KL and TV can produce unlearned models that unlearn the forget set very well but cannot preserve the influence of the retain set much, the latter of which compromises their overall utility. GD and SCRUB can produce unlearned models that preserve some influence of the retain set but do not unlearn the forget set well. App. H.3 gives preliminary results for the cases with multiple data owners and different levels of data similarity.

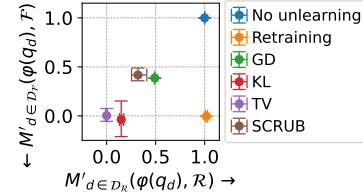


Figure 4: Benchmarking the unlearning algorithms with WaterDrum.

486 Table 4: Comparison between Waterfall and other adapted watermarking methods.  
487

Watermarking methods	D1 Separability (AUROC)	D2 Calibration ( $R^2$ )	Verification time	Require GPU
Waterfall	<b>0.965</b>	0.963	<b>0.015s</b>	<b>No</b>
Adapted KGW	0.871	<b>0.996</b>	0.336s	Yes
Adapted Synth-ID	0.549	-16.951	0.386s	Yes
Adapted EXP-edit	0.789	-17.079	165.5s	<b>No</b>

493  
494  
495 5.2 IMPLEMENTING WATERDRUM WITH OTHER WATERMARKING METHODS  
496497 In the earlier experiments, we use the Waterfall framework in our implementation of  
498 WaterDrum because it satisfies our desiderata without significant adaptations or modifications  
499 (detailed in App. C.2). As Sec. 4 suggests that WaterDrum is designed to work with other  
500 watermarking methods that satisfy the watermarking desiderata, we now explore the performance of  
501 implementing WaterDrum by adapting other watermarking methods (see details in App. E.7),  
502 including KGW (Kirchenbauer et al., 2023), Synth-ID (Dathathri et al., 2024), and EXP-edit  
503 (Kuditipudi et al., 2024).504 Table 4 shows the results of Waterfall and the adapted KGW, Synth-ID, and EXP-edit in **D1**  
505 Separability and **D2** Calibration desiderata, along with the average time required to verify a single text  
506 sample and whether a GPU is required for verification. We observe that the adapted watermarking  
507 methods require either GPU compute or a few orders of magnitude longer time than Waterfall  
508 to verify whether the text sample contains the watermark. It is promising that the adapted KGW  
509 performs very well in **D2** with a  $R^2$  value of almost 1. However, it performs poorly on **D1** as it  
510 does not satisfy **W2** Overlap Verifiability by nature. This is because KGW is designed to watermark  
511 LLM generated text for a *single* model owner instead of existing text for *multiple* data owners unlike  
512 Waterfall (see App. A.3). Thus, the adapted KGW is suited for use cases when satisfying **D2** is  
513 more important **D1** (e.g., when there is only one data owner thus less need for **W2**) and it is possible  
514 to use significantly more verification time and GPU resources. The adapted Synth-ID and EXP do  
515 not perform well on both **D1** and **D2** likely because they were not designed to satisfy **W1** and **W2**.  
516 For general settings, it is currently better to use Waterfall given its better performance in both **D1**  
517 and **D2**.

## 518 6 CONCLUSION

519 Our work here has (a) defined clear desiderata that an effective and practical unlearning metric  
520 should satisfy to enable *direct interpretation* and *continuously* measure the *extent of unlearning*  
521 (Sec. 3), (b) proposed a novel data-centric LLM unlearning metric, WaterDrum, based on robust  
522 text watermarking that, unlike existing metrics, satisfies these desiderata (Secs. 4 and 5), and (c)  
523 introduced a new WaterDrum-Ax dataset to be used with WaterDrum to benchmark unlearning  
524 algorithms (Sec. 5.1). App. I discusses other questions (e.g., limitations) a reader may have.525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539

540 REFERENCES  
541

542 Alexander Becker and Thomas Liebig. Evaluating machine unlearning via epistemic uncertainty.  
543 arXiv:2208.10836, 2022.

544 Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers,  
545 Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *Proc. IEEE S&P*, pp.  
546 141–159, 2021.

547 Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *Proc.*  
548 *IEEE S&P*, pp. 463–480, 2015.

549 CCPA. California consumer privacy act of 2018. Civil Code Title 1.81.5, 2018. URL [https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill\\_id=201720180AB375](https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375).

550 Yapei Chang, Kalpesh Krishna, Amir Houmansadr, John Frederick Wieting, and Mohit Iyyer.  
551 Postmark: A robust blackbox watermark for large language models. In *In Proc. EMNLP*, 2024a.

552 Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan  
553 Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM*  
554 *TIST*, pp. 1–45, 2024b.

555 Vikram S. Chundawat, Ayush K. Tarun, Murari Mandal, and Mohan Kankanhalli. Can bad teaching  
556 induce forgetting? unlearning in deep networks using an incompetent teacher. In *Proc. AAAI*, pp.  
557 7210–7217, 2023a.

558 Vikram S Chundawat, Ayush K Tarun, Murari Mandal, and Mohan Kankanhalli. Zero-shot machine  
559 unlearning. *IEEE Trans. Information Forensics and Security*, 18:2345–2354, 2023b.

560 I. J. Cox, M. L. Miller, J. A. Bloom, J. Fridrich, and T. Kalker. *Digital Watermarking and*  
561 *Steganography*. Morgan Kaufmann Publishers, 2nd edition, 2008.

562 Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl,  
563 Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, et al. Scalable  
564 watermarking for identifying large language model outputs. *Nature*, 634:818–823, 2024.

565 Michael Duan, Anshuman Suri, Niloofar Mireshghallah, Sewon Min, Weijia Shi, Luke Zettlemoyer,  
566 Yulia Tsvetkov, Yejin Choi, David Evans, and Hannaneh Hajishirzi. Do membership inference  
567 attacks work on large language models? In *Proc. COLM*, 2024.

568 Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

569 GDPR. Article 17 of the General Data Protection Regulation: Right to erasure ('right to be forgotten').  
570 *Official Journal of the European Union*, 2016.

571 Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal sunshine of the spotless net:  
572 Selective forgetting in deep networks. In *Proc. IEEE CVPR*, pp. 9304–9312, 2020.

573 Aditya Golatkar, Alessandro Achille, Avinash Ravichandran, Marzia Polito, and Stefano Soatto.  
574 Mixed-privacy forgetting in deep networks. In *Proc. IEEE CVPR*, pp. 792–801, 2021.

575 Michael M. Grynbaum and Ryan Mac. The Times sues OpenAI and Microsoft  
576 over A.I. use of copyrighted work. The New York Times, Dec 2023. URL  
577 <https://www.nytimes.com/2023/12/27/business/media/new-york-times-open-ai-microsoft-lawsuit.html>.

578 Yu Guo, Yu Zhao, Saihui Hou, Cong Wang, and Xiaohua Jia. Verifying in the dark: Verifiable  
579 machine unlearning by using invisible backdoor triggers. *IEEE Trans. Information Forensics and*  
580 *Security*, 19:708–721, 2023.

581 Jifei Hao, Jipeng Qiang, Yi Zhu, Yun Li, Yunhao Yuan, and Xiaoye Ouyang. Post-hoc watermarking  
582 for robust detection in text generated by large language models. In *In Proc. COLING*, 2025.

594 Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang,  
 595 Weizhu Chen, et al. LORA: Low-rank adaptation of large language models. In *Proc. ICLR*, 2022.

596

597 Yiming Hu, Chenyu Wu, Qingyan Pan, Yinghua Jin, Rui Lyu, Vikina Martinez, Shaofeng Huang,  
 598 Jingyi Wu, Lacey J. Waymentand Noel A. Clark, Markus B. Raschke, Yingjie Zhao, and Wei  
 599 Zhang. Retraction note: Synthesis of  $\gamma$ -graphyne using dynamic covalent chemistry. *Nature  
 600 Synthesis*, 3:1311, 2024.

601 Gabriel Ilharco, Marco Túlio Ribeiro, Mitchell Wortsman, Ludwig Schmidt, Hannaneh Hajishirzi,  
 602 and Ali Farhadi. Editing models with task arithmetic. In *Proc. ICLR*, 2023.

603

604 Joel Jang, Dongkeun Yoon, Sohee Yang, Sungmin Cha, Moontae Lee, Lajanugen Logeswaran, and  
 605 Minjoon Seo. Knowledge unlearning for mitigating privacy risks in language models. In *In Proc.  
 606 ACL*, 2023.

607 John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A  
 608 watermark for large language models. In *Proc. ICML*, pp. 17061–17084, 2023.

609 Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free  
 610 watermarks for language models. *Transactions on Machine Learning Research*, 2024.

611

612 Meghdad Kurmanji, Peter Triantafillou, Jamie Hayes, and Eleni Triantafillou. Towards unbounded  
 613 machine unlearning. In *Proc. NeurIPS*, pp. 1957–1987, 2024.

614

615 Woosuk Kwon, Zuhuan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E.  
 616 Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model  
 617 serving with PagedAttention. In *Proc. SOSP*, pp. 611–626, 2023.

618

619 Gregory Kang Ruey Lau, Xinyuan Niu, Hieu Dao, Jiangwei Chen, Chuan-Sheng Foo, and Bryan  
 620 Kian Hsiang Low. Waterfall: Scalable framework for robust text watermarking and provenance for  
 621 llms. In *Proc. EMNLP*, pp. 20432–20466, 2024.

622

623 Linyang Li, Botian Jiang, Pengyu Wang, Ke Ren, Hang Yan, and Xipeng Qiu. Watermarking llms  
 624 with weight quantization. In *In Proc. EMNLP*, 2023a.

625

626 Na Li, Chunyi Zhou, Yansong Gao, Hui Chen, Anmin Fu, Zhi Zhang, and Yu Shui. Machine  
 627 unlearning: Taxonomy, metrics, applications, challenges, and prospects. arXiv:2403.08254, 2024a.

628

629 Nathaniel Li, Alexander Pan, Anjali Gopal, Summer Yue, Daniel Berrios, Alice Gatti, Justin D. Li,  
 630 Ann-Kathrin Dombrowski, Shashwat Goel, Long Phan, Gabriel Mukobi, Nathan Helm-Burger,  
 631 Rassin Lababidi, Lennart Justen, Andrew B. Liu, Michael Chen, Isabelle Barrass, Oliver Zhang,  
 632 Xiaoyuan Zhu, Rishabh Tamirisa, Bhrugu Bharathi, Adam Khoja, Zhenqi Zhao, Ariel Herbert-Voss,  
 633 Cort B. Breuer, Samuel Marks, Oam Patel, Andy Zou, Mantas Mazeika, Zifan Wang, Palash  
 634 Oswal, Weiran Lin, Adam A. Hunt, Justin Tienken-Harder, Kevin Y. Shih, Kemper Talley, John  
 635 Guan, Russell Kaplan, Ian Steneker, David Campbell, Brad Jokubaitis, Alex Levinson, Jean Wang,  
 636 William Qian, Kallol Krishna Karmakar, Steven Basart, Stephen Fitz, Mindy Levine, Ponnurangam  
 637 Kumaraguru, Uday Tupakula, Vijay Varadharajan, Ruoyu Wang, Yan Shoshitaishvili, Jimmy Ba,  
 638 Kevin M. Esvelt, Alexander Wang, and Dan Hendrycks. The WMDP benchmark: Measuring and  
 639 reducing malicious use with unlearning. In *Proc. ICML*, pp. 28525–28550, 2024b.

640

641 Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee.  
 642 Textbooks are all you need II: **phi-1.5** technical report. arXiv:2309.05463, 2023b.

643

644 Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Proc. ACL Workshop  
 645 on Text Summarization Branches Out*, pp. 74–81, 2004.

646

647 Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui  
 648 Xiong, and Philip Yu. A survey of text watermarking in the era of large language models. *ACM  
 649 Computing Surveys*, 57(2):1–36, 2024.

650

651 Sijia Liu, Yuanshun Yao, Jinghan Jia, Stephen Casper, Nathalie Baracaldo, Peter Hase, Yuguang Yao,  
 652 Chris Yuhao Liu, , Xiaojun Xu, Hang Li, Kush R. Varshney, Mohit Bansal, Sanmi Koyejo, and  
 653 Yang Liu. Rethinking machine unlearning for large language models. *Nature Machine Intelligence*,  
 654 7:181–194, 2025.

648 Xinyang Lu, Jingtian Wang, Zitong Zhao, Zhongxiang Dai, Chuan-Sheng Foo, See-Kiong Ng,  
 649 and Bryan Kian Hsiang Low. WASA: WAtermark-based source attribution for large language  
 650 model-generated data. In *Proc. ACL*, 2025.

651 Aengus Lynch, Phillip Guo, Aidan Ewart, Stephen Casper, and Dylan Hadfield-Menell. Eight  
 652 methods to evaluate robust unlearning in LLMs. arXiv:2402.16835, 2024.

653 Pratyush Maini, Zhili Feng, Avi Schwarzschild, Zachary Chase Lipton, and J. Zico Kolter. TOFU: A  
 654 task of fictitious unlearning for LLMs. In *Proc. COLM*, 2024.

655 Thanh Tam Nguyen, Thanh Trung Huynh, Phi Le Nguyen, Alan Wee-Chung Liew, Hongzhi Yin, and  
 656 Quoc Viet Hung Nguyen. A survey of machine unlearning. arXiv:2209.02299, 2022.

657 Lip Yee Por, KokSheik Wong, and Kok Onn Chee. UniSpaCh: A text-based data hiding method  
 658 using unicode space characters. *J. Syst. Softw.*, 85(5):1075–1082, 2012.

659 Jipeng Qiang, Shiyu Zhu, Yun Li, Yi Zhu, Yunhao Yuan, and Xindong Wu. Natural language  
 660 watermarking via paraphraser-based lexical substitution. *Artificial Intelligence*, 317:103859, 2023.

661 Ryoma Sato, Yuki Takezawa, Han Bao, Kenta Niwa, and Makoto Yamada. Embarrassingly simple  
 662 text watermarks. *CoRR*, 2023.

663 Weijia Shi, Anirudh Ajith, Mengzhou Xia, Yangsibo Huang, Daogao Liu, Terra Blevins, Danqi Chen,  
 664 and Luke Zettlemoyer. Detecting pretraining data from large language models. In *Proc. ICLR*,  
 665 2024.

666 Weijia Shi, Jaechan Lee, Yangsibo Huang, Sadhika Malladi, Jieyu Zhao, Ari Holtzman, Daogao  
 667 Liu, Luke Zettlemoyer, Noah A Smith, and Chiyuan Zhang. MUSE: Machine unlearning six-way  
 668 evaluation for language models. In *Proc. ICLR*, 2025.

669 Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks  
 670 against machine learning models. In *Proc. IEEE S&P*, pp. 3–18, 2017.

671 Nianwen Si, Hao Zhang, Heyu Chang, Wenlin Zhang, Dan Qu, and Weiqiang Zhang. Knowledge  
 672 unlearning for llms: Tasks, methods, and challenges. *CoRR*, 2023.

673 David Marco Sommer, Liwei Song, Sameer Wagh, and Prateek Mittal. Athena: Probabilistic  
 674 verification of machine unlearning. *PoPETs*, 2022(3):268–290, 2022.

675 Ayush K Tarun, Vikram S Chundawat, Murari Mandal, and Mohan Kankanhalli. Fast yet effective  
 676 machine unlearning. *IEEE TNNLS*, 35(9):13046–13055, 2023.

677 Pratiksha Thaker, Shengyuan Hu, Neil Kale, Yash Maurya, Zhiwei Steven Wu, and Virginia Smith.  
 678 Position: LLM unlearning benchmarks are weak measures of progress. arXiv:2410.02879, 2024.

679 Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei,  
 680 Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher,  
 681 Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy  
 682 Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn,  
 683 Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel  
 684 Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee,  
 685 Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra,  
 686 Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi,  
 687 Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh  
 688 Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen  
 689 Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic,  
 690 Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models.  
 691 arXiv:2307.09288, 2023.

692 Wenbo Wan, Jun Wang, Yunming Zhang, Jing Li, Hui Yu, and Jiande Sun. A comprehensive survey  
 693 on robust image watermarking. *Neurocomputing*, 488:226–247, 2022.

694 Qizhou Wang, Bo Han, Puning Yang, Jianing Zhu, Tongliang Liu, and Masashi Sugiyama. Towards  
 695 effective evaluations and comparisons for LLM unlearning methods. In *Proc. ICLR*, 2025.

702 Ruihan Wu, Chhavi Yadav, Russ Salakhutdinov, and Kamalika Chaudhuri. Evaluating deep unlearning  
703 in large language models. arXiv:2410.15153, 2024.  
704

705 Xi Yang, Jie Zhang, Kejiang Chen, Weiming Zhang, Zehua Ma, Feng Wang, and Nenghai Yu. Tracing  
706 text provenance via context-aware lexical substitution. In *Proc. AAAI*, pp. 11613–11621, 2022.  
707

708 Jin Yao, Eli Chien, Minxin Du, Xinyao Niu, Tianhao Wang, Zezhou Cheng, and Xiang Yue. Machine  
709 unlearning of pre-trained large language models. arXiv:2402.15159, 2024.  
710

711 KiYoon Yoo, Wonhyuk Ahn, Jiho Jang, and Nojun Kwak. Robust multi-bit natural language  
712 watermarking through invariant features. In *Proc. ACL*, 2023.  
713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756 A RELATED WORKS  
757758 A.1 UNLEARNING METRICS  
759

760 Unlearning algorithms are often evaluated based on their (a) unlearning effectiveness, (b) utility  
761 preservation, and (c) unlearning efficiency (Li et al., 2024a). We first briefly discuss (b) and (c).  
762 (b) Utility preservation refers to how well the LLM maintains its performance and usability after  
763 unlearning, and can be measured with performance indicators (e.g., perplexity, accuracy) on the retain  
764 set or various downstream tasks (Chang et al., 2024b). These performance indicators can include  
765 those used for (a) below, but evaluated on the retain set instead of the forget set. (c) Efficiency of an  
766 unlearning algorithm can be assessed based on how much time and resources it saves compared to  
767 retraining from scratch (Li et al., 2024a; Nguyen et al., 2022). See Sec. 4 of (Liu et al., 2025) for a  
768 deeper discussion about other unlearning effectiveness, utility preservation, efficiency, and scalability  
769 metrics. (c) is not the focus of this work. WaterDrum is designed to evaluate (a) unlearning  
770 effectiveness (Sec. 3) but may also evaluate (b) utility preservation on the retain set (Sec. 5.1).  
771

772 **(a) Unlearning effectiveness metrics.** Broadly, unlearning effectiveness refers to how well the  
773 influence of the forget set is being removed from the LLM. There are a few classes of such metrics:  
774

- 775 • **Utility-based metrics** are a form of utility-centric metrics that expect the model utility  
776 (performance indicators), when evaluated on the forget set, to worsen after unlearning.  
777 Utility-based LLM unlearning metrics include ROUGE-L (Lin, 2004), Truth Ratio (Maini  
778 et al., 2024), and KnowMem (Shi et al., 2025). Their definitions can be found in App. E.3  
779 and the disadvantages of utility-centric metrics are already described in Sec. 2.  
780
- 781 • **Membership inference attack (MIA)-based metrics** expect the ability or probability  
782 to infer the membership of a data sample in the forget set to reduce significantly after  
783 unlearning. Some MIA-based metrics are also utility-centric as membership inference may  
784 depend on performance indicators such as perplexity and the log-likelihood of tokens in the  
785 text data (Shi et al., 2024). However, MIAs (Shokri et al., 2017) have demonstrated limited  
786 success against LLMs (Duan et al., 2024) and their performance is adversely affected by the  
787 presence of similar data in the forget and retain sets.  
788
- 789 • **Watermark-based metrics** embed signals in the forget set and expect the values of these  
790 signals to decrease after unlearning (Li et al., 2024a). **Our WaterDrum falls under this**  
791 **class but is the first metric that can be applied to LLMs. Existing watermark-based**  
792 **unlearning metrics are designed to only work for image datasets and classification**  
793 **models.** For example, the work of Guo et al. (2023) has embedded invisible backdoors  
794 in images with incorrect target labels and the success of unlearning is measured by a  
795 drop in the success rate of backdoor attacks. The work of Sommer et al. (2022) has  
796 introduced a probabilistic verification framework for backdoors, in which users modified  
797 their data prior to submission. **We will highlight the key differences of our work here:**  
798 (i) These works rely on label-based predictions and face challenges such as generalization  
799 effects, conflicting backdoor patterns, or backdoor defences. In contrast, our work focuses  
800 on adapting watermarking to LLMs where longer and more complex output sequences  
801 provide richer signals for unlearning verification. (ii) In these works, the model utility is  
802 compromised even before unlearning, especially when the forget set is large. In contrast,  
803 our WaterDrum has minimal impact on the model utility because it is based on the robust  
804 watermarking framework called Waterfall that satisfies desideratum **W0**, as shown in  
805 App. G.1. (iii) Most importantly, existing watermark- and backdoor attack-based metrics  
806 are limited to image data and cannot be directly applied as unlearning metrics to text data  
807 due to additional challenges such as in preserving data fidelity (Guo et al., 2023; Sommer  
808 et al., 2022).  
809

810 Unlearning metrics can also be classified based on whether they are **retraining-based** or  
811 **non-retraining-based**. Retraining is commonly viewed as the gold standard in classical unlearning  
812 settings (Bourtoule et al., 2021; Cao & Yang, 2015; Golatkar et al., 2020). This has led to various  
813 evaluation metrics that assert how closely an unlearned model approximates a retrained one, such  
814 as via matching performance on the forget set (Chundawat et al., 2023b; Golatkar et al., 2020) or  
815 measuring distances in weights and activations (Chundawat et al., 2023a; Golatkar et al., 2021; Tarun  
816 et al., 2023). However, retraining LLMs is often infeasible due to the scale of model parameters  
817

810 and the volume of pretraining data. In addition, retraining-based metrics contradict the premise of  
 811 unlearning that emphasizes the unavailability of a retrained LLM.  
 812

813 Therefore, non-retraining-based metrics are now more important and aligned with the growing trend  
 814 of commercial LLMs that only provide black-box access. The work of Chundawat et al. (2023a) has  
 815 proposed the ZRF score that captures the randomness in LLM predictions as an indicator of unlearning,  
 816 while the work of Becker & Liebig (2022) has proposed to utilize a model’s epistemic uncertainty.  
 817 The work of Yao et al. (2024) has proposed that a surrogate subset with the same distribution as  
 818 the forget set can be employed to approximate the performance of the retrained LLM. However,  
 819 these metrics often **overlook the LLM’s ability to generalize from pretraining or the remaining**  
 820 **retain set**. To address this, synthetic datasets, such as the TOFU dataset (Maini et al., 2024), are  
 821 carefully crafted to ensure a sufficient separation between the forget and retain sets. Nonetheless,  
 822 such a separation (i.e., a low level of data similarity) is rarely achievable in real-world scenarios.  
 823 In this work, we address these limitations by proposing a non-retraining-based metric that  
 824 works despite a greater level of data similarity between the forget and retain sets and the  
 825 generalization ability of LLMs. Additionally, our metric would work for multiple unlearning  
 826 requests. Specifically, we propose to use watermarking (Guo et al., 2023; Sommer et al., 2022) to  
 827 handle potential data similarities due to its ability to make each data point uniquely identifiable.  
 828

829 Note that our WaterDrum focuses on the setting of unlearning for data owners (also known as "data  
 830 influence removal" in the work of Liu et al. (2025)) while there exists another line of unlearning  
 831 algorithms on concept unlearning (also known as "knowledge unlearning" in the works of Si et al.  
 832 (2023) and Jang et al. (2023)), which aims to remove certain knowledge from an LLM. The distinction  
 833 between them is important when the forget set might contain concepts that are also present in the  
 834 retain set, such as in our exact and semantic duplicate experimental settings. In Sec. 2, we described  
 835 our setting where even if one copy of the forget set is to be unlearned, the concepts should remain  
 836 in the model when other copies with similar concepts exist in the retain set. This is in contrast to  
 837 knowledge unlearning where concepts from the forget set are to be removed no matter whether they  
 838 are also present in the retain set. In this latter case, unlearning metrics that directly assess the model’s  
 839 ability to answer those concepts would better measure the success of unlearning, e.g., KnowMem  
 840 (Shi et al., 2025).  
 841

## 842 A.2 COMPARISON WITH OTHER LLM UNLEARNING EVALUATION WORKS

843 The works of Maini et al. (2024); Shi et al. (2025) have proposed new unlearning metrics and  
 844 benchmark datasets. The work of Li et al. (2024b) has proposed a multiple choice question benchmark  
 845 dataset called WMDP to evaluate the LLM’s knowledge in biosecurity, cybersecurity, and chemical  
 846 security. This benchmark dataset is different from TOFU, MUSE, and ours in nature because it is  
 847 specifically for knowledge editing and only contains test data instead of training data. The work  
 848 of Wang et al. (2025) has suggested that an unlearning metric should be robust against (unchanged  
 849 by) red teaming scenarios (such as recovering knowledge by jail-breaking, probing, relearning),  
 850 and unlearning algorithms should be compared when they produce unlearned models with the  
 851 same utility/performance on the retain set that is realized by mixing the parameters of the LLM  
 852 before and after unlearning. The work of Wu et al. (2024) has proposed a new perspective of  
 853 fact unlearning and an accompanying synthetic dataset. In contrast, we propose a novel set of  
 854 unlearning metric desiderata, which is satisfied by WaterDrum, to address realistic settings,  
 855 such as when the forget and retain sets have semantically similar content and when retraining  
 856 is impractical. Our desiderata are not intended to be exhaustive and can complement that  
 857 of existing LLM unlearning evaluation works. The work of Lynch et al. (2024) has proposed a  
 858 suite of adversarial metrics to resurface forget set-related knowledge that exists in the unlearned  
 859 LLMs, such as jailbreaking prompts, relearning (via fine-tuning and in-context learning), and latent  
 860 knowledge extraction. While these metrics employ text similarity to the forget set in adversarial  
 861 scenarios to evaluate the success of unlearning, watermarking uses the signal carried in the LLM’s  
 862 text outputs to detect the influence of the forget set.  
 863

## 864 A.3 TEXT WATERMARKING FOR LLMs

865 Watermarking is an extensively studied technique for copyright protection, fingerprinting, and  
 866 authentication (Liu et al., 2024; Wan et al., 2022). Watermarking consists of two main stages:  
 867

embedding and detection where the watermark must remain imperceptible and robust against removal attacks (Wan et al., 2022). Unlike digital images where continuous signals facilitate imperceptible watermark embedding, text watermarking is more difficult due to its discrete nature and susceptibility to text modifications (Liu et al., 2024).

Existing work on text watermarking for LLMs can be classified into two modes: **(1) watermarking for a model owner** and **(2) watermarking for data owners**. These modes differ based on who uses them (i.e., model owner vs. data owners), input (i.e., generic LLM query vs. existing source text), and scalability (i.e., supporting a single model owner vs. multiple data owners). Most existing works fall under **(1) watermarking for a model owner**, and the purpose is for the LLM owner and users of the LLM to distinguish if some text is generated by the LLM. A *single* model owner can inject a watermark into the LLM’s generated text outputs and tokens during inference time by changing the sampling distribution during the autoregressive generation process (Kirchenbauer et al., 2023; Dathathri et al., 2024; Kuditipudi et al., 2024), watermarking the LLM’s weights (Li et al., 2023a), and post-hoc watermarking the LLM-generated text (Hao et al., 2025; Chang et al., 2024a).

Our work aligns better with the other mode **(2) watermarking for multiple data owners** instead. In this mode, data owners want to independently verify if their data still have influence on the LLM. This involves *multiple* data owners watermarking their existing text (such as news agencies watermarking their news articles) to trace downstream usage of their data, such as in model training. Therefore, the primary concerns of (2) differ from (1) — there is a need for stricter preservation of the semantic content of the text (**W0** Fidelity) and scalability to support a large number of data owners concurrently. (2) can be further categorized into (2a) non-LLM approaches or (2b) LLM approaches. Existing watermarking methods in (2a), such as inserting Unicode characters (Lu et al., 2025; Sato et al., 2023; Por et al., 2012) or synonym replacement (Qiang et al., 2023; Yoo et al., 2023; Yang et al., 2022), are often easily detectable and susceptible to word replacement (Lau et al., 2024). On the other hand, syntactic-based watermarking methods are often constrained by the limited choices of syntactic structures and require prior linguistic knowledge (Wan et al., 2022). Recently, LLMs have emerged as a promising watermarking tool as they can generate natural-looking text and improve watermarking robustness. The work of Lau et al. (2024) falls under (2b) by proposing a robust text watermarking framework called `Waterfall` that is capable of embedding watermarks across data from multiple data owners (while preserving the semantic content of the original text) and also achieving watermarking robustness such that watermarks in the training data of LLMs remain detectable in the LLMs’ text outputs. The watermarked training data are then used to train other LLM models. **Our work builds on top of our adaptation of the `Waterfall` framework (Lau et al., 2024) to develop our unlearning metric, i.e., `WaterDrum`. We have also considered adapting (1) watermarking methods for a model owner for mode (2) and in `WaterDrum` in Sec. 5.2.**

## B FURTHER DISCUSSION ON UNLEARNING METRIC DESIDERATA (SEC. 3)

### B.1 SEPARABILITY DESIDERATUM D1

A separable unlearning metric (i.e., **D1**) should be a good classifier of whether an owner’s data still influences an unlearned LLM, in particular, the retrained LLM  $\varphi_{\mathcal{R}}$  (i.e., achieved by perfect unlearning) trained only on  $\mathcal{D}_{\mathcal{R}}$ . To illustrate the difference between a separable and non-separable metric, we provide a toy example in Fig. 5(left). With a separable metric, an optimal threshold  $\kappa^*$  can be chosen where false positive and false negative classifications are minimal, as is the case for `WaterDrum` shown in Fig. 5 (top right). However, for non-separable metrics, any  $\kappa$  chosen would result in similar true and false positive rates, as is the case for the utility-centric Truth Ratio metric shown in Fig. 5 (bottom right).

### B.2 CALIBRATION DESIDERATUM D2

Ideally, perfect unlearning will completely remove the influence of the forget set. However, in practice, imperfect unlearning may be inevitable due to the size and complexity of LLMs. This is because (a) perfect unlearning involving retraining from scratch is prohibitively expensive and impractical, and (b) perfect unlearning on LLMs is not yet achievable with the current approximate unlearning algorithms without significantly harming model utility/performance (e.g., on the retain set). In Sec. 5.1, we demonstrate that all unlearning algorithms only achieve imperfect unlearning,

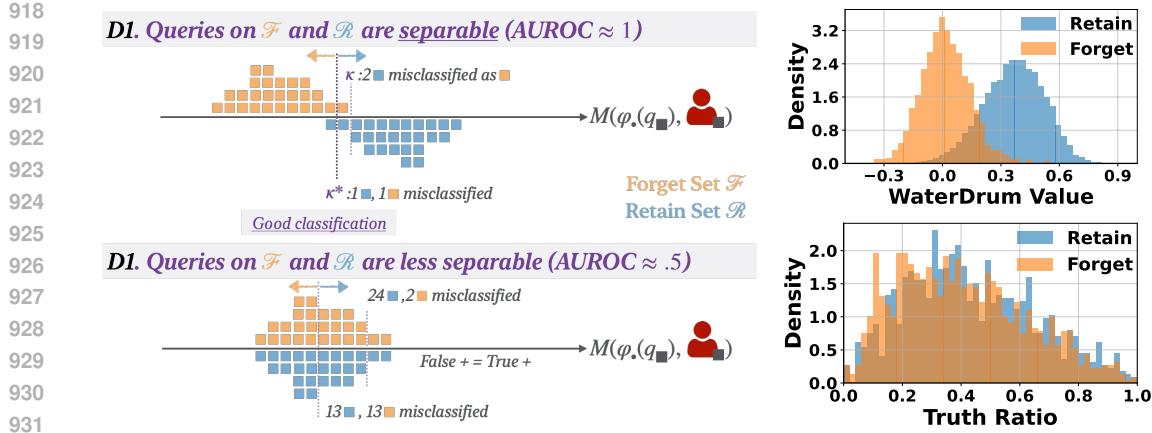


Figure 5: (Left) A toy example is provided to illustrate the intuition of the separability desiderata **D1**. Each ■ represents an LLM’s text output to a query formed by a data point. Different  $\kappa$ ’s correspond to different decision boundaries. In the top diagram, the metric and  $\kappa^*$  can clearly separate the LLM’s text outputs to queries formed by the forget set vs. the retain set. In the bottom diagram, there is no  $\kappa$  that can clearly separate them and the true and false positive rates are always the same. (Right) Histograms of our WaterDrum vs. the utility-centric Truth Ratio metric values under the ‘semantic duplicate’ setting of data similarity for WaterDrum–TOFU dataset (Table 2 in Sec. 5) where WaterDrum exhibits a clear separability over the Truth Ratio metric; see the caption of Fig. 1 for a detailed description.

except when the LLM is destroyed (i.e., it is influenced by neither the forget nor the retain sets) or when a new LLM is retrained from scratch.

With the calibration desideratum **D2**, characterization of imperfect unlearning becomes possible. **D2** enables an unlearning metric to go *beyond being just a binary indicator* of whether the entire forget set has been unlearned to being a meaningful *continuous measure* of the extent to which a forget set  $\mathcal{D}_F$  has been unlearned:

- The proposed linear proportional form (i.e., Eq. (2)) of **D2** captures the goal that the unlearning metric can be interpreted on its own and indicates the proportion of  $\mathcal{D}_F$  that remains unlearned, while being given only a single calibration data point (i.e., the aggregate unlearning metric value of the forget set on the *original LLM*) available before unlearning. This contrasts with existing utility-centric metrics, which require another calibration data point (i.e., the aggregate unlearning metric value of the forget set on the *retrained LLM*) and hence violate **D3(a)**, as discussed in Sec. 3.
- Our experiments (Fig. 3 and Table 3) show that WaterDrum can satisfy **D2**, enabling this intuitive interpretation when LLMs are retrained with the retain set  $\mathcal{D}_R$  and varying proportions of the forget set  $k/|\mathcal{D}_F|$ .

Fig. 6 provides an intuitive illustration of the calibration desideratum where the metric measures the extent of imperfect unlearning. **D2** is practically useful in the following use cases:

1. Deployment: In practice, model owners may only achieve imperfect unlearning of the forget set to some extent while preserving the utility/performance of their LLM for customers. A calibrated continuous unlearning metric value satisfying **D2** can serve as an objective proxy for negotiations with data owners on the required extent of unlearning and corresponding monetary compensation. The negotiated target extent can then guide the actual implementation of unlearning, e.g., by selecting the most suitable unlearning algorithm (since different algorithms achieve different forget-retain performance trade-offs, as shown in Fig. 4) or guide the tuning of hyperparameters for a given algorithm.
2. Evaluation and development: For research and development, a calibrated metric satisfying **D2** enables evaluation beyond binary success/failure and instead continuously measures

972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
the extent of imperfect unlearning of the forget set. This supports a more realistic and fine-grained assessment of unlearning algorithms.

**D2. Metric  $M$  is calibrated**

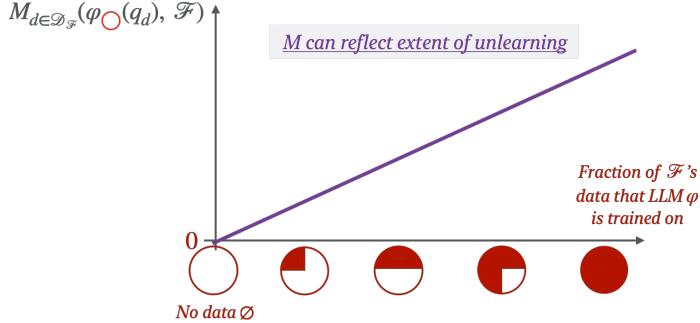


Figure 6: A calibrated metric should continuously measure the extent of imperfect unlearning. On the horizontal axis, we simulate using different sized proportions of the red owner’s dataset. **D2** requires the metric to have a value of 0 when the dataset is not used (y-intercept) and a larger value when a larger proportion is used. As shown in Fig. 3 in Sec. 5, WaterDrum is well-calibrated while other metrics are not.

### B.3 FEASIBILITY DESIDERATUM D3

In Sec. 2, we describe how utility-centric metric values, such as ROUGE-L, cannot be interpreted on their own. Instead, their interpretation requires referencing the aggregate metric value of the forget set on the retrained LLM. For example, we can compute the difference between the aggregate metric values on the unlearned vs. retrained LLMs. For a perfectly unlearned LLM (e.g., the retrained LLM), the difference should become 0. Note that although the difference satisfies our calibration desideratum **D2**, it violates **D3(a)**.

What happens when the aggregate metric value on the retrained LLM is unknown? Any aggregate metric value (e.g., ROUGE-L score) would not be informative; it is impossible for the model owner to know whether the aggregate metric value indicates perfect unlearning or only imperfect unlearning, or to know how far the unlearned LLM is from perfect unlearning. In Sec. 5, we do not reference the retrained LLM. The baseline metrics computed using the unlearned LLM only do not satisfy **D2**, as seen in Fig. 3.

### B.4 ROBUSTNESS TO SIMILAR DATA DESIDERATUM D4

In Secs. 2 and 3, we suggest that it is common for data owners to have semantically similar instances. Here, we provide concrete examples. Consider a real-life scenario where two news agencies, Reuters and The Straits Times (i.e., the data owners), produce semantically similar news articles, as shown in Fig. 7a. These two articles from different data owners exhibit a high semantic similarity with an STS score of 0.90. In this case, only one agency may request unlearning. As another example in the WaterDrum-Ax dataset, Fig. 7b shows that the two arXiv paper abstracts from the same *Materials Science* category but different authors (i.e., the data owners) are also semantically similar with an STS score of 0.88. In this example, only one group of authors may request unlearning.

## C FURTHER DISCUSSION ON WATERFALL

### C.1 OVERVIEW OF WATERFALL

Waterfall (Lau et al., 2024) embeds watermark signals in text by paraphrasing the text while preserving its original meaning. For example, “The cat caught the rat” can be watermarked and paraphrased as “The rat was captured by the cat” while preserving the same meaning and not affecting

Manchester City players celebrate after winning the Champions League final against Inter Milan. [\[REUTERS\]](#)

Martyn Herman, Man City hang tough to beat Inter and complete the treble. Reuter:  
<https://www.reuters.com/sports/soccer/manchester-city-beat-inter-milan-win-champions-league-2023-06-10/>. Date of access: Apr 1, 2025

1044 (a) The news agencies, Reuters and The Straits Times, both produce news articles reporting on the same soccer  
1045 match and hence have a high semantic similarity with STS = 0.90.

## ABSTRACT

We present the median surface brightness profiles of diffuse Ly $\alpha$  haloes (LAHs) around star-forming galaxies by stacking 155 spectroscopically confirmed Ly $\alpha$  emitters (LAEs) at  $3 < z < 4$  in the MUSE Extremely Deep Field (MXDF) with a median Ly $\alpha$  luminosity of  $L_{\text{Ly}\alpha} \approx 10^{41.1} \text{ erg s}^{-1}$ . After correcting for a systematic surface brightness offset we identified in the data cube, we detect extended Ly $\alpha$  emission out to a distance of  $\approx 270$  kpc. The median Ly $\alpha$  surface-brightness profile shows a power-law decrease in the inner 20 kpc and a possible flattening trend at a greater distance. This shape is similar for LAEs with different Ly $\alpha$  luminosities, but the normalisation of the surface-brightness profile increases with luminosity. At distances over 50 kpc, we observe a strong overlap of adjacent LAHs, and the Ly $\alpha$  surface brightness is dominated by the LAHs of nearby LAEs. We find no clear evidence of redshift evolution of the observed Ly $\alpha$  profiles when comparing with samples at  $4 < z < 5$  and  $5 < z < 6$ . Our results are consistent with a scenario in which the inner 20 kpc of the LAH is powered by star formation in the central galaxy, while the LAH beyond a radius of 50 kpc is dominated by photons from surrounding galaxies.

## ABSTRACT

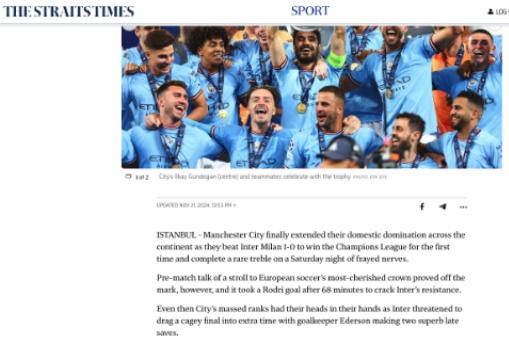
The extended Ly $\alpha$  haloes (LAHs) have been found to be prevalent around high-redshift star-forming galaxies. However, the origin of the LAHs is still a subject of debate. Spatially resolved analysis of Ly $\alpha$  profiles provides an important diagnostic. We analyse the average spatial extent and spectral variation of the circumgalactic LAHs by stacking a sample of 155 Ly $\alpha$  emitters (LAEs) at redshift  $3 < z < 4$  in the MUSE Extremely Deep Field. Our analysis reveals that, with respect to the Ly $\alpha$  line of the target LAE, the peak of the Ly $\alpha$  line at large distances becomes increasingly more blueshifted up to a projected distance of 60 kpc ( $\approx 3 \times$  virial radius), with a velocity offset of  $\approx 250$  km/s. This trend is evident in both the mean and median stacks, suggesting that it is a general property of our LAE sample, which typically has a Ly $\alpha$  luminosity  $\approx 10^{41.1}$  erg s $^{-1}$ . However, due to the absence of systemic redshift data, it remains unclear whether the Ly $\alpha$  line peak at large projected distances is less redshifted compared to the inner regions or truly blueshifted with respect to the systemic velocity. We explore various scenarios to explain the large-scale kinematics of the Ly $\alpha$  line.

1064 (b) In the WaterDrum-Ax dataset, the two arXiv paper abstracts from the same *Materials Science* category but  
1065 different authors both present similar content and hence have a high semantic similarity with STS = 0.88.

Figure 7: Examples of high semantic text similarity (STS) in different domains.

downstream uses of the text (e.g., when used for LLM training). We will briefly describe the watermarking and verification operators below and refer the reader to the work of Lau et al. (2024) for more details. The watermarking and verification processes of Waterfall do not require the training of any LLMs.

**Watermarking Operator.** Waterfall uses an off-the-shelf LLM as a paraphraser, which preserves the text’s original meaning, and injects token-level watermark perturbation signals to the LLM’s logits while generating the paraphrased text. Specifically, the token-level perturbations depend on the data owner’s key (ID) and preceding tokens such that, on average, the perturbation is equivalent to adding random noise to the LLM’s logits. Technically, at each token generation step, this involves (i) a permutation of the vocabulary space ordering based on the ID and preceding tokens, combined with (ii) ID-dependent orthogonal perturbation functions that allow desirable



Manchester City beat Inter Milan to win Champions League, clinch treble. The Straits Times: <https://str.sg/i3nR>. Date of access: Apr 1, 2025.

1080 properties such as the ability to add multiple watermarks in the same text. The watermarking operator  
 1081 is summarized in Algorithm 1 of (Lau et al., 2024).  
 1082

1083 **Verification Operator.** With knowledge of the data owner’s key (ID), it becomes possible to find  
 1084 the ‘right’ permutation to verify whether there has indeed been any signal embedded into the text or  
 1085 not. Technically, this involves simply accessing the correct vocabulary token space permutation. If  
 1086 a watermark has been embedded, doing a dot product in this permuted token space would yield a  
 1087 signal. Else, on average, only noise will be present, hence no signal will be detected. The verification  
 1088 operator is summarized in Algorithm 2 of (Lau et al., 2024). It does not involve any LLM inference  
 1089 or training and can be run on a CPU.

1090 We discuss practical deployment details and computational cost of `Waterfall` in App. C.4.  
 1091

## 1092 C.2 WATERFALL AND WATERMARKING DESIDERATA

1094 `Waterfall` satisfies the watermarking desiderata required for `WaterDrum`, as stated in Sec. 4.  
 1095 Specifically,

- 1097 • **W0 Fidelity:** `Waterfall` (Lau et al., 2024) is designed to satisfy a *fidelity* desideratum  
 1098 and ensure that the watermarked text is semantically similar to its unwatermarked version.  
 1099 Fidelity is ensured by using an LLM as a paraphraser and adding watermark signals to the  
 1100 LLM logits when generating tokens (such that, on average, the perturbation is equivalent to  
 1101 adding random noise).

1102 **Evidence.** App. H.3 of (Lau et al., 2024) shows that the LLMs fine-tuned using watermarked  
 1103 vs. unwatermarked datasets have minimal difference in *fidelity*. In App. G.1, we also  
 1104 empirically verify that the watermarking process has minimal impact on the LLM’s  
 1105 performance (e.g., Truth Ratio).

- 1106 • **W1 Verifiability:** (a) `Waterfall` would produce a high verification score if the  
 1107 watermarked text and the correct corresponding watermark key are inputs to the verification  
 1108 operator (Algorithm 2), and a score with an expected value of 0 otherwise (e.g., if the  
 1109 wrong watermark key or unwatermarked text is used). Intuitively, this is because the  
 1110 verification score is the dot product of the watermark signal and the average cumulative  
 1111 token distribution, which is almost uniform noise without the right watermark key and  
 1112 watermarked text. (b) `Waterfall`’s verification score can be aggregated by taking the  
 1113 uniform average over all  $d \in \mathcal{D}_F$ . When the LLM is trained on a larger subset of the forget  
 1114 set, more LLM outputs will contain the watermark, such that the aggregate metric value  
 1115 increases proportionally.

1116 **Evidence.** (a) Sec. 4.3 of (Lau et al., 2024) has shown that `Waterfall` is verifiable in the  
 1117 LLM fine-tuned over watermarked text with AUROC of 1.0 when evaluated on 100 queries  
 1118 of 100 generated tokens each. We empirically verify this with the `WaterDrum-TOFU` and  
 1119 `WaterDrum-Ax` datasets in Sec. 5 (under ‘Separability desideratum **D1**’). (b) We further  
 1120 show empirically that the verification score of `Waterfall` is also proportional to the size  
 1121 of the subset of the forget set in Sec. 5 (under ‘Calibration desideratum **D2**’).

- 1122 • **W2 Overlap verifiability:** `Waterfall` (Lau et al., 2024) is designed such that different  
 1123 watermark keys correspond to different permutations and perturbations of the logits in the  
 1124 LLM paraphraser. With pseudorandom permutations of the LLM logits and watermark  
 1125 signals (added to the LLM logits) that are defined with *orthogonal* functions, different  
 1126 watermarks are less likely to interfere with one another.

1127 **Evidence.** Sec. 4.3 of (Lau et al., 2024) showed that `Waterfall` remains verifiable in an  
 1128 LLM when the training dataset has texts with up to 100 different watermarks. We empirically  
 1129 verify **W2** with the `WaterDrum-Ax` dataset in Sec. 5 (under ‘Separability desideratum  
 1130 **D1**’).

- 1131 • **W3 Query access constraint:** Algorithm 2 of (Lau et al., 2024) for performing watermark  
 1132 verification only requires the suspected text (i.e., text output from the LLM), and does not  
 1133 require any other access to the LLM which generates the text.
- 1134 • **W4 Unique key:** `Waterfall` (Lau et al., 2024) is designed to satisfy a *scalability*  
 1135 desideratum since it has a theoretical maximum of  $10^{130274}$  unique watermark keys due to its

1134 vocabulary permutations and orthogonal perturbations. This is in contrast to the maximum  
 1135 of  $10^{10}$  for other watermarking frameworks for data owners (Lau et al., 2024). This allows  
 1136 different data owners to have different unique keys, and it is extremely unlikely that different  
 1137 owners end up with the same key by random chance.

1138 **Evidence.** App. E.8 of Lau et al. (2024) empirically shows that the watermark is verifiable  
 1139 for up to 100,000 random unique watermark keys.

1140

1141 Furthermore, `Waterfall` also satisfies the additional desiderata described in App. D:

1142

- **W5 Private:** Algorithm 2 of (Lau et al., 2024) requires the private watermark key for watermark verification. As the watermark is embedded in the phrasing of the text, `Waterfall`'s watermark key cannot be directly or easily extracted from the watermarked text.

1143

1144

1145

1146

1147

1148

1149

1150

1151

1152

1153

1154

1155

1156

1157

1158

1159

1160

1161

**Evidence.** The work of Lau et al. (2024) has compared `Waterfall` with other text watermarking frameworks in Sec. 4.1 (Robust verifiability) and App. F.3. Other text watermarking frameworks (Lu et al., 2025; Qiang et al., 2023; Sato et al., 2023; Yoo et al., 2023), unlike `Waterfall`, have watermark keys that can be directly extracted from the text, causing them to be easily exploited and fail the following desideratum **W6**. We further discuss this in App. D.

- **W6 Resilient:** `Waterfall`'s robustness relies on its unique token-level embedding process such that completely removing its signal will require so many token changes in the text that would likely destroy its original meaning.

1162

1163

1164

1165

1166

1167

1168

1169

1170

1171

1172

1173

1174

1175

1176

In general, any watermarking method that satisfies our watermarking desiderata can be used within our `WaterDrum`. To the best of our knowledge, `Waterfall` is the only [text watermarking approach for data owners](#) to satisfy our requirements as of now, and is thus used in our paper. However, as discussed in Sec. 5.2, it is possible to adapt other watermarking methods for model owner to better satisfy our requirements and we expect more watermarking methods to work with future developments in text watermarking.

1177

### C.3 WATERMARKING OF DATASETS WITH WATERFALL

1178

1179

1180

1181

1182

1183

1184

1185

1186

1187

1188

Watermarking and verification of the training text data have been performed with `Waterfall` (Lau et al., 2024) using the default configuration of the code available on <https://github.com/aoi3142/Waterfall>. When creating our `WaterDrum-Ax` and `WaterDrum-TOFU` datasets, the data is watermarked using the default LLM `meta-llama/Llama-3.1-8B-Instruct` with watermark strength  $\kappa = 2^4$  and perturbation key  $k_p = 1$ . For licensing information on individual papers in the `WaterDrum-Ax` dataset, see <https://arxiv.org/help/license>.

1189

1190

1191

1192

1193

1194

1195

1196

1197

1198

1199

1200

To create the watermarked `WaterDrum-Ax` and `WaterDrum-TOFU` datasets, we consider the scenario where each data owner  $i \in \mathcal{T}$  with ID  $i$  for  $i = 0, 1, 2, \dots, |\mathcal{T}| - 1$  has its own unique watermark key  $\mu_i$ . This ensures that each data owner is able to uniquely verify and evaluate the influence of its own data on the LLM when fine-tuned on its data and after any possible unlearning has been done on the fine-tuned LLM. For simplicity, we set  $\mu_i$  to be  $i$  when watermarking our datasets with `Waterfall`. For the experimental settings where duplicate data is considered (see Sec. 5, specifically, under ‘Robustness to similar data **D4**’), each data owner  $f \in \mathcal{F}$  whose data  $\mathcal{D}_f$  is in the forget set  $\mathcal{D}_{\mathcal{F}}$  and duplicated would have another data owner  $j \in \mathcal{T}$  (where  $j \neq f$ ) owning the duplicate of owner  $f$ 's data  $\mathcal{D}_f$  and watermarking this duplicate with its watermark key  $\mu_j$ . The practical motivation for data owners with similar data is discussed in App. B.4.

1201

<sup>4</sup>Note that this  $\kappa$  is the watermark strength, as defined in `Waterfall` (Lau et al., 2024), and not the same as the separability threshold defined in the separability desideratum **D1**.

To create the WaterDrum–Ax dataset, we consider  $|\mathcal{T}| = 20$  unique data owners where each  $i$ -th category among the 20 categories of paper abstracts belongs to a single data owner  $i$  and is watermarked with its watermark key  $\mu_i = i$  for  $i = 0, 1, \dots, 19$ . This emulates the setting where a model owner is aggregating data from 20 sources of academic publications where each source centers on a single academic discipline. To construct a forget set that consists of data from  $n$  data owners, the data from the last  $n$  of the  $|\mathcal{T}|$  data owners are used as the forget set, while the data from the first  $|\mathcal{T}| - n$  data owners forms the retain set. For instance, when unlearning the data from 1 data owner, the data from data owner  $i = 19$  is the forget set. When unlearning the data from 5 data owners, the data from data owners  $i = 15, 16, 17, 18, 19$  forms the forget set. For the experimental settings where duplicate data is considered as discussed above, the next data owner  $j = (f + 1) \bmod |\mathcal{T}|$  owns the duplicate of the data  $\mathcal{D}_f$  from the previous data owner  $f$  and watermarks it with  $\mu_j = j$  before including it into the augmented retain set.

To create the WaterDrum–TOFU dataset, we follow the construction in (Maini et al., 2024) by considering just two data owners 0 and 1 with the retain and forget sets, respectively. The retain set is watermarked with key  $\mu_0 = 0$ , while the forget set is watermarked with key  $\mu_1 = 1$ . For the experimental settings where duplicate data is considered as discussed above, data owner 0 owns the duplicate of the forget set from data owner 1 by watermarking it with  $\mu_0 = 0$  before including it into the augmented retain set.

Note that as part of Waterfall’s watermarking process, the original text data is paraphrased using an LLM. Although efforts have been made to ensure that the watermarked text retains a high semantic similarity with the original text (see the work of Lau et al. (2024) and <https://github.com/aoi3142/Waterfall>), we cannot guarantee the faithful reproduction of all content from the original text nor the factual correctness of the watermarked texts. In practical unlearning applications, additional (automated or manual) checks can be performed on the watermarked text to ensure accuracy and consistency to the original text (Lau et al., 2024). To reduce the computational cost, we have omitted these steps from our watermarking process. Despite this, similar to the fully fictitious TOFU dataset introduced by the work of Maini et al. (2024), WaterDrum–Ax and WaterDrum–TOFU still serve as suitable datasets when used for the purpose of evaluating unlearning metrics and algorithms where the factuality of the content in the dataset is not relied upon.

#### C.4 PRACTICAL DEPLOYMENT PIPELINE FOR WATERDRUM WITH WATERFALL FOR LLM UNLEARNING EVALUATION

The watermarking process of WaterDrum is **lightweight** and incurs very little computational cost. This makes the watermarking process simple and **convenient for data and model owners during real-world deployment**:

- Data owners can quickly watermark their data before sharing them with model owners or releasing important data publicly. This not only facilitates unlearning verification but also allows them to detect whether their data has been used by model owners without authorization (Lau et al., 2024; Maini et al., 2024).
- No changes are required by the model owners who can continue training closed-source LLMs, provide API access, or release open-source models.
- Data owners can detect whether their data has been used for fine-tuning (even in closed-source LLMs) based only on the LLM’s text outputs. After submitting an unlearning request, they can verify the extent of unlearning via WaterDrum.
- In comparison, other LLM unlearning metrics face severe deployment barriers, such as requiring to reference a retrained LLM (Secs. 1 and 2), which is infeasible even for cooperative model owners due to the computational cost.

The overhead of watermarking the training data is minimal compared to the cost of retraining the entire LLM. Watermarking the WaterDrum–TOFU dataset using an implementation of Waterfall with the vLLM library (Kwon et al., 2023) on GPU takes only 10 seconds per 1000 data samples and is performed only once when data is first contributed. In contrast, the cost of retraining the entire LLM is around 1h 30min, which is 100 $\times$  higher than the cost of watermarking the training data. Furthermore, retraining has to be repeated for every unlearning request.

1242 In addition, our framework also reduces the verification costs. Verification of the Waterfall’s  
 1243 watermark is very efficient (Lau et al., 2024), requiring about 3 seconds per 1000 query outputs  
 1244 on CPU. In contrast, the computation of ROUGE using the rouge-score library<sup>5</sup> takes around 170  
 1245 seconds per 1000 query outputs, which is two orders of magnitude slower.

1246 A **limitation** is that WaterDrum requires watermarking the data before training and cannot be  
 1247 applied retroactively to data that has already been released. However, this practical concern will  
 1248 likely diminish with time and be mitigated due to the following reasons:

- 1250 1251 1252 1253 1254 1255 1256 1257 1258 1259 1260 1261 1262 Recalling already released data may be possible in our unlearning setting as data owners have  
 the rights to their data and can control their use, as discussed in Secs. 1 and 2. Therefore,  
 data owners of released data can still exercise the rights to their data by telling the model  
 owner that they would (i) require their updated watermarked data to be used in the LLM  
 instead, or (ii) consent to the continued use only if watermarking is to be part of the data  
 processing step in the next LLM release. In either case, the model owner must comply with  
 laws and regulations such as the GDPR and copyright laws.
- Even without recalling historical and released data, the data owners can demand that  
 watermarks be applied going forward in future LLMs. They can expect newly generated  
 data to be watermarked, hence facilitating future practical LLM unlearning evaluations. For  
 example, news agencies can start watermarking their news articles and these recent articles  
 may be more important in training future LLMs.

1263 As awareness of privacy and security in LLM training grows, we expect proactive watermarking  
 1264 of data before release to become a common practice among data owners and the **applicability of**  
 1265 **WaterDrum to grow over time.**

1266 A similar concept called image watermarking in the domain of computer vision has been widely  
 1267 studied and adopted for image data copyright protection (Cox et al., 2008). We note that when  
 1268 applications of image watermarking are proposed, such as data backdoors to verify unlearning  
 1269 (Thaker et al., 2024), they also face the same constraint that image watermarking can only be applied  
 1270 to unreleased data (and cannot be retroactively embedded in historical data). The community has  
 1271 accepted the constraint and appreciated the potential benefits going forward. Thus, there are strong  
 1272 reasons to believe in the potential for wider adoption of text watermarking and its applications.

## 1274 D RESILIENCE

1276 In Remark 2 of Sec. 4, we raise the following question: Is WaterDrum still an effective unlearning  
 1277 metric if (i) the model owner tries to reduce its LLM’s aggregate metric value without directly  
 1278 performing unlearning or copyright its LLM via **watermarking for model owner** (Kirchenbauer et al.,  
 1279 2023) or (ii) some data owner(s) try to report inflated aggregate metric values to understate the  
 1280 unlearning performed by the model owner?

1281 In this section, we explain when the answer is yes. In particular, the underlying watermarking  
 1282 framework should additionally satisfy the following desiderata:

1284 **W5 Private.** Each data owner  $i$ ’s watermark key  $\mu_i$  should be private and unknown to the model  
 1285 owner. Verification of the watermark with the verification operator should require the private  
 1286 watermark key. Moreover, the watermark key should not be easily extractable from the watermarked  
 1287 text. This prevents others without the watermark key (e.g., other data owners, model owner) from  
 1288 verifying whether some text contains the watermark and computing the metric value without owner  
 1289  $i$ ’s permission.

1290 A private watermark may be required to support **W6**.

1291 **W6 Resilient.** The watermark should remain **W1** verifiable in the LLM’s text outputs after attacks  
 1292 by the model owner. These attacks should have minimal impact on the semantic similarity of the  
 1293 LLM’s text output and should not significantly affect the value of the LLM (as in **W0**) as the model  
 1294 owner is still interested in deploying a usable LLM.

1295 <sup>5</sup><https://pypi.org/project/rouge-score/>.

Our adopted watermarking framework, `Waterfall`, satisfies **W5** as the verification algorithm requires the private key and the watermark key cannot be extracted from the watermarked text (Lau et al., 2024). In contrast, other watermarking frameworks do not satisfy **W5**. Frameworks that use invisible Unicode watermark key (Lu et al., 2025; Sato et al., 2023; Por et al., 2012) have their watermark keys plainly exposed in the watermarked text, while frameworks using synonym replacements (Qiang et al., 2023; Yoo et al., 2023; Yang et al., 2022) have watermarks that can be easily extracted from the watermarked text when the verification algorithm is known. The extraction enables the model owner to compute the metric value and remove or replace the watermark in the LLM’s text outputs, which results in these other watermarking frameworks failing to satisfy **W6**.

We analyze whether `Waterfall` satisfies **W6** in this section. We consider attacks including (a) the model owner intercepting the LLM’s text outputs based on a proxy indicator  $SS$  such as semantic similarity with forget set (App. D.1), (b) [watermarking for a model owner](#) applied to the LLM during its auto-regressive generation (App. D.3), and (c) other modifications made to the LLM’s text output after generation (App. D.3).

We also consider that under **W5**, data owner(s) can falsely try to report inflated aggregate metric values to underestimate the unlearning performed by the model owner in App. D.2.

### D.1 DECOY MODEL TO REDUCE AGGREGATE METRIC VALUE WITHOUT DIRECTLY PERFORMING UNLEARNING

Here, we consider the setting where the data owners requesting their data to be erased are unaware of or cannot control how the model owner performs unlearning. Instead, they can only evaluate the unlearning by querying the updated model. Moreover, the model owner’s interests may not align with those of the data owners. As performing unlearning and obtaining the unlearned model  $\tilde{\varphi}$  can be more costly, the model owner may want to avoid performing unlearning while still appearing to fulfil the data owners’ erasure requests. The model owner can attempt to reduce the aggregate metric value by using the *decoy model*  $\check{\varphi}$  instead.<sup>6</sup>

**Decoy model.** The model owner implements the decoy model  $\check{\varphi}$ , which involves using a gating function to intercept any query  $q_d$  received. For any metric  $M$  that the model owner can compute exactly, the model owner would intercept queries that result in large metric values, indicating that the original LLM  $\varphi_T$  is greatly influenced by the forget set  $\mathcal{D}_F$  (e.g., queries  $q_d$  where  $M(\varphi_T(q_d), f) > \kappa$  for some  $f \in \mathcal{F}$ ), and replace the LLM’s text output  $\varphi_T(q_d)$  with some text  $g(q_d, \mathcal{D}_F)$  that reduces the metric value. For metrics that the model owner cannot compute exactly (e.g., metrics that require some information that is private to the data owner), the model owner can only resort to a proxy indicator  $SS$  that measures how similar the LLM’s text output  $\varphi_T(q_d)$  is to the text from the forget set  $\mathcal{D}_F$ . The decoy model is defined as follows:

$$\check{\varphi}(q_d) = \begin{cases} g(q_d, \mathcal{D}_F) & \text{if } \exists d_f \in \mathcal{D}_F \text{ } SS(\varphi_T(q_d), d_f) > B, \\ \varphi_T(q_d) & \text{otherwise; } \end{cases} \quad (4)$$

with a threshold value  $B$  determined by the model owner. Note that a small  $B$  would intercept more queries and replace more of the LLM’s text outputs. This may be more costly, reduce the overall LLM performance, and may essentially be comparable to a full unlearning algorithm. *How would the decoy model affect the effectiveness of various unlearning metrics when evaluating the decoy model?*

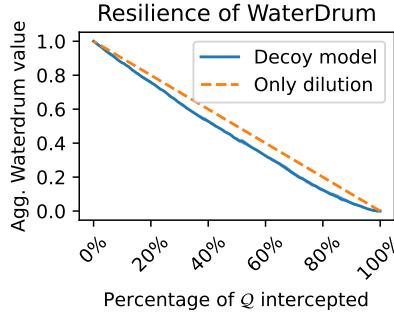
For metrics that do not depend on information private to the data owners (e.g., ROUGE-L), the model owner can compute them directly. The model owner can use the metric as  $SS$ , set the threshold  $B$  to match the (learned) threshold  $\kappa$  from **D1**, and optimize the replacement text  $g(q_d, \mathcal{D}_F)$  that reduces the metric value. As  $SS$  and  $B$  can be more easily set, the model owner’s decoy model attack will be more successful and less costly (from fewer interceptions). Thus, the metric is less resilient to the decoy model attack.

For metrics that depend on information private to the data owners (e.g., `WaterDrum` using `Waterfall` that satisfies **W5**), the model owner cannot compute them directly. Instead, the

<sup>6</sup>Note that the model owner would have no incentive to do so when benchmarking different unlearning algorithms (Sec. 5.1). During benchmarking, the model owner (instead of the data owner) controls the training data and assesses the unlearning effectiveness. The model owner would not deceive itself about the performance of their unlearning algorithms.

1350  
 1351 model owner can only define  $SS$  based on some proxy indicator of similarity  $SS$  between the LLM’s  
 1352 text output and data from the forget set  $\mathcal{D}_F$ . The model owner would also have to tune  $B$ . A lower  $B$   
 1353 would reduce the aggregate WaterDrum value (giving the impression of unlearning) but comes at  
 1354 the high cost of more replacements and poor model utility/performance. When using Waterfall,  
 1355 the model owner can only generate replacement text  $g(q_d, \mathcal{D}_F)$  with lower metric values by using  
 1356 unwatermarked text from other sources (e.g., another LLM). This may further reduce the decoy  
 1357 model’s performance. Thus, WaterDrum is more resilient to the decoy model attack.

1358 **Empirical Evaluation of WaterDrum.** We prepare data  $\bar{\mathcal{D}}$  to form a set  $\mathcal{Q} := \{q_d\}_{d \in \bar{\mathcal{D}}}$  of queries  
 1359 that result in the aggregate WaterDrum value being above a threshold  $\kappa$ , i.e.,  $M'_{d \in \bar{\mathcal{D}}}(\varphi_T(q_d), \mathcal{F}) >$   
 1360  $\kappa$ . To prevent the model owner from recognizing and intercepting the queries easily, data  $\bar{\mathcal{D}}$  are  
 1361 similar to but not directly based on the forget set  $\mathcal{D}_F$ . For example,  $\mathcal{D}_F$  is a set of arXiv paper  
 1362 abstracts from the math.PR category, and  $\bar{\mathcal{D}}$  consists of other such math paper abstracts not in  $\mathcal{D}_F$ .  
 1363 The model owner can only use the proxy indicator STS score as  $SS$  and must choose the threshold  
 1364  $B$ . Fig. 8 plots the aggregate WaterDrum value against the percentage of  $\mathcal{Q}$  being intercepted as  
 1365 the threshold  $B$  decreases. As the model owner decreases  $B$ , it potentially reduces the aggregate  
 1366 WaterDrum value via two effects: (i) diluting the aggregate WaterDrum value by replacing the  
 1367 LLM’s text output (with watermark signal) with text from unwatermarked data sources, and (ii)  
 1368 the remaining unintercepted LLM’s text outputs are semantically more dissimilar to the original  
 1369 watermarked  $\mathcal{D}_F$ . Note that the aggregate WaterDrum value decreases almost linearly with the  
 1370 percentage of intercepted queries, implying that the model owner only relies on effect (i) with no help  
 1371 from effect (ii), i.e., the model owner can only reduce the aggregate WaterDrum value significantly  
 1372 by intercepting most queries whose resulting LLM’s text outputs are semantically similar to  $\mathcal{D}_F$ .  
 1373 This makes it very costly for the model owner to carry out the attack. For example, reducing the  
 1374 aggregate WaterDrum value to 0.2 requires intercepting about 70% of the queries in  $\mathcal{Q}$  – the model  
 1375 owner may favor performing actual unlearning instead.  
 1376



1386 Figure 8: Plot of aggregate WaterDrum value of the forget set on the original LLM  $\varphi_T$  against  
 1387 the % of queries in  $\mathcal{Q}$  being intercepted as the model owner decreases the threshold  $B$  in the decoy  
 1388 model. An ideal unlearning metric would have its aggregate metric value decrease only proportionally  
 1389 with the % of intercepted queries (dashed orange diagonal line). WaterDrum achieves a similar  
 1390 performance, implying that the decoy model needs to intercept a large proportion of queries to reduce  
 1391 the aggregate WaterDrum value significantly. The aggregate WaterDrum values are scaled such  
 1392 that the value is 1.0 when there is no intercepted query.  
 1393

## 1395 D.2 DATA OWNER REPORTING INFLATED AGGREGATE WATERDRUM VALUES

1396 When the watermarking framework satisfies **W5**, it is possible that a data owner refuses to  
 1397 acknowledge that unlearning has been done and reports an inaccurate, higher than measured aggregate  
 1398 WaterDrum value.  
 1399

1400 To resolve this issue, a trusted third party can certify the aggregate WaterDrum value reported  
 1401 by the data owner. To do so, the model owner provides access to the unlearned model  $\tilde{\varphi}$  and the  
 1402 data owner  $i$  provides the watermark key  $\mu_i$  to the trusted third party for verification. Verification  
 1403 should only be performed on a predefined set of queries with the agreement of both model and data  
 owners. This prevents the model owner from repeatedly querying for the WaterDrum value to

1404 directly optimize the watermark removal instead of actual unlearning and from building a better  
 1405 decoy model in App. D.1. The third party can then certify to the model owner that the measured  
 1406 aggregate WaterDrum value aligns with that reported by the data owner.  
 1407

1408 **D.3 DILUTION OF WATERDRUM WATERMARK WITH WATERMARKING FOR MODEL OWNER AND**  
 1409 **TEXT MODIFICATIONS**  
 1410

1411 In certain scenarios, the model owner might want to watermark the newly generated text from the  
 1412 fine-tuned LLM. This could either be for the legitimate purpose of copyrighting its LLM (identifying  
 1413 texts generated from its LLM) or in an adversarial attempt to dilute the watermark of the training  
 1414 data. In this section, we empirically analyze whether Waterfall satisfies **W6** and WaterDrum is  
 1415 still an effective unlearning metric (i.e., achieves **D1** Separability and **D2** Calibration) under these  
 1416 scenarios. We consider **the model owner (1) using watermarking** and (2) perturbing the LLM’s text  
 1417 output after output generation.  
 1418

1419 *As discussed in App. A.3, watermarking methods for a model owner may inject watermarks during*  
 1420 *inference time by changing the sampling distribution during the autoregressive generation process.*  
 1421 *These methods can be further categorized into (i) during logits generation, such as (Kirchenbauer et al.,*  
 1422 *2023), or (ii) during token sampling, such as (Kuditipudi et al., 2024). Waterfall’s watermarking*  
 1423 *mechanism (when adapted as a watermarking method for a model owner) is similar to those in*  
 1424 *category (i), resulting in watermarks embedded in the LLM’s logits distribution, and hence shares*  
 1425 *similar properties with other methods in category (i). For model owners to effectively remove category*  
 1426 *(i)’s watermarks without destroying text fidelity, they would need to apply the right distortions to*  
 1427 *cancel out the forget owner’s watermark. However, since the forget owner’s watermark key is private*  
 1428 *to the data owners (**W5**), it is practically impossible for the model owner to extract the key with*  
 1429 *limited samples from the forget owner. The model owner can only apply random watermark signals,*  
 1430 *which result in random distortions on average. Waterfall is designed to be robust to such random*  
 1431 *distortions, as we will show empirically below.*

1432 The work of Lau et al. (2024) has empirically shown that Waterfall is robust to various attacks,  
 1433 such as the watermark replacement attack (attack  $\mathbb{A}3$  in (Lau et al., 2024)), where the original text  
 1434 watermark remains detectable even after the text has been watermarked again with a different text  
 1435 watermark. We also empirically evaluate the effectiveness of WaterDrum when using various  
 1436 **watermarking methods for a model owner**.

1437 First, we adapted the underlying watermarking algorithm of Waterfall’s watermarking framework  
 1438 for data owner to act as a **watermarking method for a model owner**. We performed our experiments  
 1439 using LLMs fine-tuned on the WaterDrum-Ax or WaterDrum-TOFU dataset, applying a  
 1440 watermark strength of  $\kappa = 2^7$ , same as that used during text watermarking (App. C.1), with key  
 1441  $\mu = 20$ , which differs from the data owners’ keys ( $\mu = 0$  to  $\mu = 19$ ). We choose the same  
 1442 watermarking method as the data owner to perform **watermarking for model owner** because using a  
 1443 different watermarking method reduces the likelihood that the model owner can exactly cancel out  
 1444 the watermark signal embedded by the forget owner. The strongest attack by the model owner would  
 1445 be to apply the exact same watermark method with a ‘destructive interference’ signal that introduces  
 1446 perturbations opposite to those of the forget owner. However, in practice, the model owner cannot do  
 1447 so with high probability given **W5**.

1448 As the adaptation of Waterfall falls under category (i), we expect similar results from other  
 1449 **watermarking methods for a model owner** in this category. Nevertheless, we **additionally** experimented  
 1450 with a commonly used category (i) **benchmark watermarking method for a model owner**, KGW  
 1451 (Kirchenbauer et al., 2023), with their default watermarking strength of  $\delta = 2$ , and green-list ratio of  
 1452  $\gamma = 0.5$ , as shown in Table 5. We observe that the watermark signals are preserved post-attack. Note  
 1453 that due to differences in the underlying methodology of Waterfall and KGW, the watermarking  
 1454 strength of  $\kappa = 2^7$  in Waterfall and  $\delta = 2$  in KGW are not directly comparable, and performance  
 1455 differences cannot be solely attributed to whether one **watermarking algorithm for model owner** is  
 1456 “stronger” in affecting WaterDrum than the other. Additionally, while a stronger **watermarking for**  
 1457 **model owner** might potentially cause a larger drop in the text watermark’s verifiability, Kirchenbauer  
 1458 et al. (2023) has shown that a strong **watermarking for model owner** also negatively affects the quality

1459 <sup>7</sup>Note that this  $\kappa$  is the watermark strength as defined in Waterfall (Lau et al., 2024) and not the same as  
 1460 the separability threshold defined in **D1** separability.

1458 Table 5: Resilience of WaterDrum when the model owner applies various [watermarking methods](#)  
 1459 for a model owner.

Watermarking for Model Owner	D1 Separability (AUROC)	D2 Calibration ( $R^2$ )
No Injection	0.964	0.963
Adapted Waterfall (Lau et al., 2024)	0.923	0.936
KGW (Kirchenbauer et al., 2023)	0.944	0.948
ITS (Kuditipudi et al., 2024)	0.955	0.965

1468 of the model’s generation. Thus, it is not in the model owner’s interest to significantly degrade the  
 1469 model performance simply to affect the data owners’ ability to verify their watermarks.

1470 As for category (ii), also known as *distortion-free* watermarks, a seeded pseudo-random  
 1471 sampler is used in place of an unseeded random sampler during the LLM output generation  
 1472 (Kuditipudi et al., 2024). In this case, the LLM’s output text is still generated from the token  
 1473 distribution of the underlying LLM and is therefore practically indistinguishable from that of  
 1474 an unwatermarked LLM. This implies that the underlying token distribution—containing the  
 1475 WaterDrum watermarks—remains undistorted, hence preserving the watermark verifiability. In the  
 1476 experiments in our paper, no [watermarking method for a model owner](#) is performed and we use the  
 1477 default unseeded random sampler and generate ten LLM outputs per data sample (Sec. E.2). As a  
 1478 result, we would not expect outputs from distortion-free [watermarking for model owner](#) to differ  
 1479 significantly from those produced by random sampling. We provide the results of inverse transform  
 1480 sampling (ITS) watermarking from (Kuditipudi et al., 2024) in Table 5, which again show that the  
 1481 watermark signals have been preserved post-attack.

1482 In Table 5, the AUROC and  $R^2$  decrease only slightly with the injection of [watermarks for model](#)  
 1483 [owner](#). Thus, WaterDrum still preserves **D1 Separability** and **D2 Calibration** well.

1484 **Perturbing the LLM’s text output after generation.** The model owner can use word level  
 1485 edits such as insertion, deletion and synonym substitution attacks, or passage level edits, such as  
 1486 paraphrasing and applying another text watermark on the LLM’s text output. We note that such  
 1487 attacks are not very practical or realistic for the model owner. Word level edits can greatly affect the  
 1488 fidelity of the generated text (Lau et al., 2024), while passage level attacks are computationally costly  
 1489 for the model owner, increase latency of the model’s responses to queries, and would not support  
 1490 streaming of the LLM’s outputs. Nonetheless, Waterfall has been demonstrated to be robust to  
 1491 these attacks (Lau et al., 2024), and the same robustness would translate to WaterDrum watermarks.

1492 All in all, this subsection shows that **Waterfall satisfies W6 and thus WaterDrum is resilient**  
 1493 **to attacks by the model owner.**

## E DETAILS ON EXPERIMENTAL SETUP

1498 We conduct our experiments on NVIDIA L40 and H100 GPUs. Results are averaged  
 1499 across 3 random seeds  $\{41, 42, 43\}$ . Text generation from the different models uses  
 1500 temperature = 1, top-p = 1, and top-k left as the LLM vocabulary size. We use  
 1501 sentence-transformers/all-mpnet-base-v2 as the STS model to evaluate STS scores.  
 1502 More details of our experimental setup are presented below.

### E.1 TRAINING HYPERPARAMETERS

1503 **Models fine-tuned on WaterDrum-Ax.** We fine-tune the bfloat16-pretrained Llama-2-7B model  
 1504 from Hugging Face<sup>8</sup> using LoRA (Hu et al., 2022) (based on  $r = 8$  and  $\alpha = 32$ ) with batch size  
 1505 128, 20 training epochs, and learning rate  $10^{-3}$ . Additionally, we fine-tune the bfloat16-pretrained  
 1506 Phi-1.5 model (detailed in App. F.3) with the same settings. Following the model choices in Maini  
 1507 et al. (2024), we have considered these two models as they are representative of the recent LLMs and  
 1508 differ in terms of model architectural details and model scale.

1511 <sup>8</sup><https://huggingface.co/meta-llama/Llama-2--7B-hf>.

1512 **Models fine-tuned on WaterDrum-TOFU.** We fine-tune the bfloat16-pretrained Llama-2-7B-chat  
 1513 model from Hugging Face<sup>9</sup> using LoRA (Hu et al., 2022) (based on  $r = 8$  and  $\alpha = 32$ ) with batch  
 1514 size 128, 10 training epochs, and learning rate  $10^{-4}$ .  
 1515

1516 Subsequently, for unlearning, we use a batch size of 32. While we conduct our experiments using  
 1517 LoRA as in other LLM unlearning works (Maini et al., 2024; Shi et al., 2025), we also demonstrate  
 1518 that WaterDrum applies to full parameter fine-tuning in App. F.2.  
 1519

## 1520 E.2 QUERIES FOR FINE-TUNED LLM

1521 For the WaterDrum-Ax dataset, we simulate a completion task by prompting the LLM with the  
 1522 first 50 tokens of the training dataset for the LLM to complete the text. For the WaterDrum-TOFU  
 1523 dataset, a Q&A dataset, we prompt the LLM with the questions formatted according to the LLM’s  
 1524 prompt format. We generate 10 outputs for each prompt, and the mean metric value over the 10  
 1525 generations was taken. We generate up to a maximum of 200 tokens for each query.  
 1526

## 1527 E.3 BASELINE UNLEARNING METRICS

- 1529 • **ROUGE-L** measures the longest common subsequence between the generated text and a  
 1530 reference text. This serves as a surrogate for the generation quality for the WaterDrum-Ax  
 1531 dataset and the question answering (QA) accuracy for the WaterDrum-TOFU dataset. To  
 1532 calculate the metric value, we follow the works of (Maini et al., 2024; Shi et al., 2025) in  
 1533 computing the ROUGE-L recall scores (Lin, 2004) to compare the LLM’s text output with  
 1534 the training data as ground truth. Specifically, we evaluated the ROUGE-L value of each  
 1535 text output by comparing it with the data sample used for the query as the reference, i.e.,  
 1536  $P(\varphi_{\bullet}(q_d), \mathcal{F}) = \text{ROUGE-L}(\varphi_{\bullet}(q_d), d)$ .  
 1537
- **Truth Ratio** measures the probability of generating a correct answer versus a wrong answer  
 1538 as an indicator of whether the LLM still memorizes the knowledge to be unlearned on the  
 1539 WaterDrum-TOFU dataset. Following the work of (Maini et al., 2024), for each given  
 1540 question, the ratio is computed from dividing the averaged probabilities of multiple wrong  
 1541 answers by the probability of a paraphrased true answer.  
 1542
- **KnowMem** measures the ROUGE-L recall values of QA pairs related to the training data  
 1543 to measure the LLM memorization of the knowledge on the WaterDrum-Ax dataset.  
 1544 Following the work of (Shi et al., 2025), we use GPT-4 to create a question answering  
 1545 evaluation set with 8000 QA pairs based on the abstracts in the WaterDrum-Ax dataset  
 1546 and measure the ROUGE-L recall values between the LLM’s text output to the questions  
 1547 and the ground truth answers.  
 1548
- **MIA** measures the difference in the predictive distribution between two models to measure  
 1549 privacy leakage from unlearning. Specifically, we employ the state-of-the-art Min-40%  
 1550 attack (Shi et al., 2024) based on the loss on the forget set and holdout set and compute the  
 1551 AUROC of discriminating both set of losses. To align our evaluation with other baselines  
 1552 (where lower values indicate better unlearning), we report MIA as  $1 - \text{AUROC}$ .  
 1553
- **WaterDrum** (ours): We compute the watermark metric value using the LLM’s text output  
 1554 excluding the prompt (i.e., without the first 50 tokens). Note that in our experiments on  
 1555 WaterDrum-Ax, multiple watermarks are present as there are many data owners, each  
 1556 watermarking their data with a unique watermark. Each data owner  $i \in \mathcal{F}$  would send  
 1557 queries  $q_d$  to the LLM based on the data from their own dataset  $d \in \mathcal{D}_i$  and verify the  
 1558 LLM’s output using their watermark key  $\mu_i$ .  
 1559

1560 With the exception of MIA, we use the uniform average over all text  $d \in \mathcal{D}_{\mathcal{F}}$  to compute the aggregate  
 1561 metric value, i.e.,  $M_{d \in \mathcal{D}_{\mathcal{F}}}(\varphi_{\bullet}(q_d), \mathcal{F}) := \sum_{d \in \mathcal{D}_{\mathcal{F}}} M(\varphi_{\bullet}(q_d), \mathcal{F}) / |\mathcal{D}_{\mathcal{F}}|$  over all data points  $d \in \mathcal{D}_{\mathcal{F}}$ .  
 1562 For MIA, the metric for individual data points are defined as the loss of that sample on the LLM,  
 1563 while the aggregate metric value is defined to be  $M_{d \in \mathcal{D}_{\mathcal{F}}}(\varphi_{\bullet}(q_d), \mathcal{F}) := 1 - \text{AUROC}$ , where the  
 1564 AUROC is computed by comparing the loss of the forget set versus the holdout set.  
 1565

<sup>9</sup>[https://huggingface.co/meta-llama/Llama-2-\\$2-\\$7\\$B-chat-hf](https://huggingface.co/meta-llama/Llama-2-$2-$7$B-chat-hf).

1566  
1567

## E.4 DETAILS ON DUPLICATE DATA

1568  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1580  
1581

As discussed in Sec. 5 (specifically, under ‘Robustness to similar data **D4**’), we examine three representative settings where there exists data  $\mathcal{D}_s$  (injected into  $\mathcal{D}_{\mathcal{R}}$ ) that is similar to  $\mathcal{D}_{\mathcal{F}}$  with different *SS*: (a) **exact duplicate**:  $\mathcal{D}_s$  is an exact copy of  $\mathcal{D}_{\mathcal{F}}$ . This marks the highest similarity with mean STS = 1.00 and ROUGE = 1.00. (b) **semantic duplicate**:  $\mathcal{D}_s$  is a paraphrased version of  $\mathcal{D}_{\mathcal{F}}$  with the same semantic meaning. We use GPT-4 to paraphrase each text in  $\mathcal{D}_{\mathcal{F}}$  and obtain  $\mathcal{D}_s$ . In this setting,  $\mathcal{D}_s$  has mean STS = 0.97, ROUGE = 0.69 on WaterDrum-Ax, and mean STS = 0.96, ROUGE = 0.60 on WaterDrum-TOFU. **While the ‘exact duplicate’ setting can be less common in reality, it clearly illustrates and clarifies the limitations of different unlearning metrics. The ‘semantic duplicate’ setting happens and is a better measure of realistic unlearning capabilities (see real-world examples in App. B.4).** We also consider the standard case when there is (c) **no duplicate** in the dataset, i.e.,  $\mathcal{D}_s = \emptyset$ . In our setup, the data  $\mathcal{D}_s$  are owned by another owner  $s \neq f$ , and hence watermarked with owner  $s$ ’s watermark key  $\mu_s \neq \mu_f$ . Specific watermarking details are in App. C.3. For each setting, we fine-tune a model on  $\mathcal{D}_{\mathcal{R}}^s = \mathcal{D}_s \cup \mathcal{D}_{\mathcal{R}}$ . During subsequent unlearning, we seek to remove the influence of  $\mathcal{D}_{\mathcal{F}}$  while retaining the influence of  $\mathcal{D}_{\mathcal{R}}^s$ .

1582  
1583

## E.5 DETAILS ON CALIBRATION

1584  
1585  
1586  
1587  
1588  
1589  
1590  
1591  
1592  
1593

In our experiments in Sec. 5 (specifically, under ‘Calibration desideratum **D2**’), we simulated varying sizes of subsets of the forget set by partitioning the forget set sequentially into 10 partitions, and retraining the LLMs with by incrementally including partitions (and the retain set) in the training set of the retrained LLMs, i.e., using the first 0%, 10%, 20%, …, 100% of  $\mathcal{D}_{\mathcal{F}}$  as  $\mathcal{D}_{\bullet}$  when retraining the LLMs on  $\mathcal{D}_{\mathcal{R}} \cup \mathcal{D}_{\bullet}$ . It can be observed that WaterDrum satisfies the calibration desiderata under this method of partitioning, and believe that in general, this would hold in expectation for randomly sampled fixed-size subsets of the forget set. We empirically verified this for random subsets in App. F.1.

1594  
1595  
1596  
1597  
1598  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608

## E.6 BENCHMARKING UNLEARNING ALGORITHMS

1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619

We consider the setting of a model owner comparing the aggregate WaterDrum values on different unlearned models  $\tilde{\varphi}$  resulting from different unlearning algorithms. A practical scenario for this setting could be where a model owner experiments with different unlearning algorithms on a small dataset to evaluate their performance before selecting the algorithm to fulfil unlearning requests from data owners after deployment. The model owner could also be a researcher developing new unlearning algorithms. Under this setting, we assume that the training data, model and watermark key are all under the full control of the model owner. The model owner could use watermarked data such as WaterDrum-Ax or WaterDrum-TOFU with known watermark keys and can compute the metric values of the retain and forget sets directly without the restrictions from App. D. A perfect unlearning algorithm would ideally produce an unlearned model  $\tilde{\varphi}$  with (i) high retain aggregate metric value close to the original model  $\varphi_{\mathcal{T}}$ , i.e.,  $M_{d \in \mathcal{D}_{\mathcal{R}}}(\tilde{\varphi}(q_d), \mathcal{R}) \approx M_{d \in \mathcal{D}_{\mathcal{R}}}(\varphi_{\mathcal{T}}(q_d), \mathcal{R})$  and (ii) low forget aggregate metric value, i.e.,  $M_{d \in \mathcal{D}_{\mathcal{F}}}(\tilde{\varphi}(q_d), \mathcal{F}) \approx 0$ , corresponding to the bottom right corner in Fig. 4.

In our experiments, we have adopted several popular baseline unlearning algorithms detailed as follows:

- **Retraining**: The base LLM is trained only on the retain set  $\mathcal{D}_{\mathcal{R}}$  for the same number of epochs when training the original LLM  $\varphi_{\mathcal{T}}$  to obtain the retrained LLM  $\varphi_{\mathcal{R}}$ . The retrained LLM usually serves as the golden standard for other unlearning algorithms.
- **Gradient Descent (GD)**: The original LLM  $\varphi_{\mathcal{T}}$  is fine-tuned using gradient descent on the retain set  $\mathcal{D}_{\mathcal{R}}$  for 1 or several epochs. This method assumes that the LLM naturally forgets about the forget set as learning progresses on the retain set. In the experiments, we fine-tuned for 1 epoch.
- **KL Minimization (KL)** (Maini et al., 2024): The original LLM  $\varphi_{\mathcal{T}}$  is updated by concurrently maximizing the prediction loss on the forget set and minimizing the Kullback-Leibler divergence between the predictions of the unlearned LLM and original LLM on the retain set for 5 unlearning epochs.

1620  
 1621 Table 6: Comparison of watermarking methods for a model owner and Waterfall based on  
 1622 whether they satisfy the proposed watermarking desiderata Sec. 4. We use  $\wedge$  to denote that the  
 1623 judgment is off-the-shelf, as it requires modification to their code to support the desideratum. We  
 1624 use  $*$  to denote that the judgment is based on our results in Table 4 after adaptation of the schemes.  
 1625 Future work may nontrivially improve the design of the different watermarking methods and satisfy  
 the desiderata to a better extent than our current results.

Watermarking methods	W0	W1	W2	W3	W4
Waterfall (Lau et al., 2024)	Yes	Yes	Yes	Yes	Yes
KGW (Kirchenbauer et al., 2023)	No $\wedge$	Yes*	Not shown	Yes	No $\wedge$
Synth-ID (Dathathri et al., 2024)	No $\wedge$	No*	Not shown	Yes	No $\wedge$
EXP-edit (Kuditipudi et al., 2024)	No $\wedge$	No*	Not shown	Yes	Yes

1626  
 1627  
 1628  
 1629  
 1630  
 1631  
 1632  
 1633  
 1634 • **SCRUB** (Kurmanji et al., 2024): The original LLM  $\varphi_T$  is updated by maximizing the  
 1635 Kullback-Leibler divergence between the predictions of the unlearned LLM and original  
 1636 LLM on the forget set, while minimizing the prediction loss and divergence on the retain set.  
 1637 The optimization process alternates between maximization steps and minimization steps. In  
 1638 our experiments, we ran 3 maximization and minimization epochs.  
 1639  
 1640 • **Direct Preference Optimization (DPO)** (Maini et al., 2024): For question answering tasks,  
 1641 the original LLM  $\varphi_T$  is updated to encourage responses such as “I don’t know” on the forget  
 1642 set, while simultaneously minimizing the prediction loss on the retain set. Note that this  
 1643 method is not compatible with completion tasks, and is omitted for the WaterDrum–Ax  
 1644 dataset. We ran 5 unlearning epochs for DPO.  
 1645  
 1646 • **Task Vector (TV)** (Ilharco et al., 2023): We follow the implementation in (Shi et al., 2025).  
 1647 First, the original model  $\varphi_T$  is further fine-tuned using the forget set to obtain a reinforced  
 1648 LLM  $\varphi_{\text{reinforce}}$ . Next, we take the difference in parameters by subtracting the parameters  
 1649 of the  $\varphi_T$  from the parameters of  $\varphi_{\text{reinforce}}$ . Lastly, the unlearned model is obtained by  
 1650 subtracting this difference from the parameters of  $\varphi_T$ . In the experiments,  $\varphi_{\text{reinforce}}$  is  
 1651 fine-tuned from  $\varphi_T$  on the forget set for 5 epochs.

1652 Note that we exclude Gradient Ascent on the forget set from the original LLM (Maini et al., 2024)  
 1653 from the unlearning algorithms considered as they have been shown to perform poorly in other works  
 1654 where the LLM’s text outputs become gibberish or random words (Maini et al., 2024).

## E.7 ADAPTING WATERMARKING METHODS FOR A MODEL OWNER TO WATERDRUM

1655 As discussed in Sec. 5.2, to demonstrate the potential of implementing other watermarking methods  
 1656 in WaterDrum, we adapt KGW (Kirchenbauer et al., 2023), Synth-ID (Dathathri et al., 2024), and  
 1657 EXP-edit (Kuditipudi et al., 2024), which are popular **watermarking methods for a model owner**  
 1658 that have been described in App. A.3. However, based on just the designs and experimental support of  
 1659 these methods, it was not clear that all of the watermarking desiderata would be met, as summarized  
 1660 in Table 6. Additionally, we still need further adaptations to them and additional experiments to  
 1661 establish that the watermarking desiderata were met:

1662  
 1663  
 1664 • For example, to satisfy **W4**, the watermarking methods for data owners that inject  
 1665 watermarks into training texts need to support insertion and verification of watermarks  
 1666 from *different/multiple* data owners. In contrast, watermarking methods for a model owner  
 1667 are designed for watermarking a single LLM’s generated text, and typically only require a  
 1668 *single* watermark to serve the single model owner.  
 1669  
 1670 • The requirements of **W2** are also not explored, but it is an important requirement for these  
 1671 watermarking methods for a model owner to be used in WaterDrum. On the other hand,  
 1672 Sec. 4.3 of Waterfall (Lau et al., 2024) had established this property of being able to  
 1673 recover *multiple different* watermark signals in text generated by LLMs trained on text  
 1674 watermarked by *many data owners* in their experiments, giving us more confidence in using  
 1675 it for demonstrating the effectiveness of WaterDrum.

Nonetheless, we attempt to adapt the methods (KGW, Synth-ID, and EXP-edit) to satisfy the watermarking desiderata. Specifically, we first add a paraphrasing prompt following the approach in Waterfall1. As KGW and Synth-ID were designed for watermarking (of LLM-generated text) for a single model owner, their codebase had explicitly fixed single watermark IDs into the internal components of the source code – we have to modify it for them to accept different watermarks and support multiple data owners. We use the same Llama-3.1-8B-Instruct LLM to apply text watermarking on the original ArXiv dataset, except for Synth-ID, for which we use Gemma-7b-it as the provided codebase only supports Gemma and GPT-2 models. The results have been shown in Table 4 in Sec. 5.2.

## F ABLATION STUDIES

### F.1 CALIBRATION IN EXPECTATION FOR RANDOMLY SAMPLED FIXED-SIZE SUBSETS OF THE FORGET SET

In Eq. (2), we defined the **D2** Calibration desideratum as the expectation of the metric value across different random subsets of  $\mathcal{D}_\bullet \subseteq \mathcal{D}_F$  with the fixed-size of  $|\mathcal{D}_\bullet| = k$ . To verify that WaterDrum satisfies this relationship in expectation for different random subsets, we perform the calibration experiment with three different random subsets (randomly selecting a subset of size  $k$  from  $\mathcal{D}_F$ ) under the ‘no duplicate’ setting for WaterDrum-Ax dataset, for a total of four different subsets (including the sequential subsets detailed in App. E.5) for each of the following proportions  $k/|\mathcal{D}_F| \in \{0\%, 20\%, 40\%, 60\%, 80\%, 100\%\}$ . We plot the mean, minimum and maximum of the aggregate WaterDrum value for each proportion in Fig. 9, as well as the best-fit calibration line of the mean aggregate WaterDrum value through origin. It can be observed from Fig. 9 that the linear proportional relationship holds in expectation with the mean values achieving a high  $R^2$  value of 0.960 (close to the value of 0.963 reported in Table 3). Moreover, each subset has an aggregate WaterDrum value that is close to the mean of the four subsets, i.e., there is a narrow range between the minimum and maximum values.

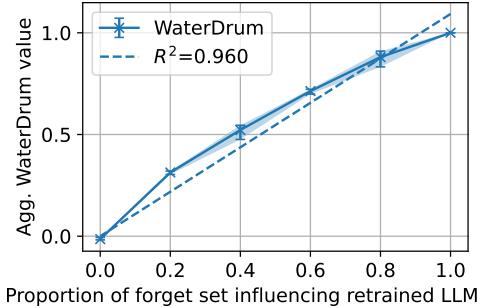


Figure 9: Plot of aggregate WaterDrum value against proportion  $k/|\mathcal{D}_F|$  of forget set influencing retrained LLM. Each  $\times$  markers indicates the mean aggregate WaterDrum value across four subsets of the same size  $k$  and is enclosed by error bars indicating the minimum and maximum aggregate WaterDrum values. The dotted line shows the best-fit line through origin with its associated  $R^2$  value.

### F.2 EVALUATION ON FULL PARAMETER FINE-TUNING

The experiments in Sec. 5 were conducted using LoRA (Hu et al., 2022), following the setting in other LLM unlearning works (Maini et al., 2024; Shi et al., 2025). To show that WaterDrum is also applicable when used for full parameter fine-tuning, we conduct experiments for the separability (**D1**) and calibration (**D2**) desiderata with varying levels of similarity for the WaterDrum-Ax dataset.

For full parameter fine-tuning, we use a learning rate of  $10^{-4}$  and train for 10 epochs. Note that due to the high computational cost of full parameter fine-tuning, we only report the results for one seed, while the results for LoRA are averaged across three different seeds.

Table 7 and Table 8 show that WaterDrum performs better than other metrics and better satisfy **D1** and **D2** for both LoRA and full parameter fine-tuning.

Table 7: AUROC of various unlearning metrics under different levels of data similarity for the WaterDrum-Ax dataset. WaterDrum’s AUROC remains near 1.0 even when similar data exists.

Similarity		ROUGE	KnowMem	WaterDrum
Exact	Full	0.335	0.497	0.990
	LoRA	0.334	0.492	0.957
Semantic	Full	0.965	0.447	0.990
	LoRA	0.960	0.450	0.963
No	Full	0.984	0.481	0.991
	LoRA	0.974	0.491	0.965

Table 8:  $R^2$  of the best-fit line for various unlearning metrics under different levels of data similarity for the WaterDrum-Ax dataset. WaterDrum is very well linearly calibrated across the settings with the highest  $R^2$  value.

Similarity		ROUGE	KnowMem	MIA	WaterDrum
Exact	Full	-5059	-981.5	-4.774	0.984
	LoRA	-37.47	-498.1	-1220	0.987
Semantic	Full	0.545	-139.2	-35.57	0.989
	LoRA	0.693	-276.5	-90.21	0.991
No	Full	0.850	-103.8	-3.937	0.940
	LoRA	0.650	-252.9	-7.553	0.963

### F.3 EVALUATION ON OTHER MODELS

We have also evaluated our WaterDrum using Phi-1.5<sup>10</sup>. Figs. 10a and 10b illustrate the AUROC and calibration for the settings of ‘no duplicate’ and ‘exact duplicate’. The high AUROC and  $R^2$  value agrees with our main experiments using Llama2-7B and shows that WaterDrum satisfy the proposed desiderata. This validates our WaterDrum’s adaptability to different LLMs, which increases its real world applicability.

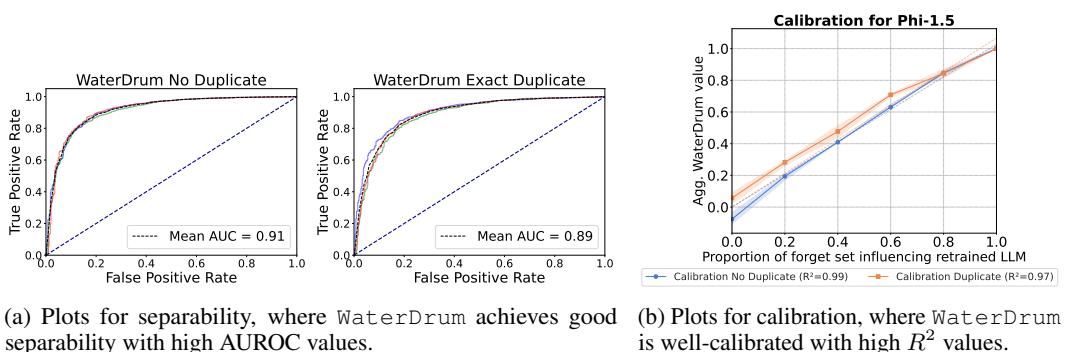
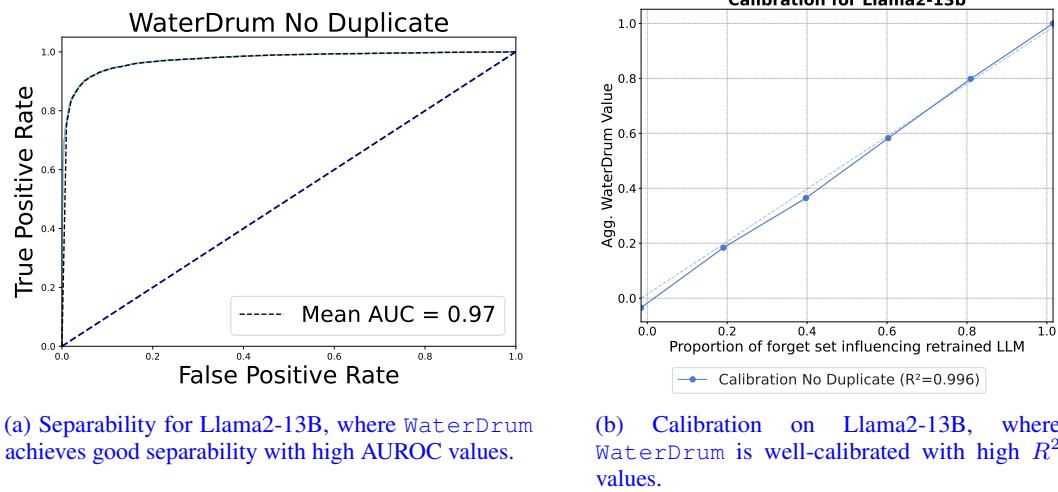


Figure 10: **D1** and **D2** of our WaterDrum measured on the Phi-1.5 model for the WaterDrum-Ax dataset under the no duplicate and exact duplicate settings.

<sup>10</sup>[https://huggingface.co/microsoft/phi-1\\_5](https://huggingface.co/microsoft/phi-1_5).

Beyond the Llama2-7B model that we evaluate in the main experiments, we further investigate the performance of WaterDrum on a larger model, Llama2-13B, on the WaterDrum-TOFU dataset. Fig. 11a and Fig. 11b illustrate the AUROC and calibration for the settings of ‘no duplicate’. The high AUROC and  $R^2$  values demonstrate that WaterDrum satisfies the proposed desiderata to a larger extent on larger models. This further validates WaterDrum’s adaptability to different model sizes.



(a) Separability for Llama2-13B, where WaterDrum achieves good separability with high AUROC values.

(b) Calibration on Llama2-13B, where WaterDrum is well-calibrated with high  $R^2$  values.

Figure 11: **D1** and **D2** of our WaterDrum measured on the Phi-1.5 model for the WaterDrum-Ax dataset under the no duplicate and exact duplicate settings.

#### F.4 BENCHMARK METRICS ON LLMs FINE-TUNED ON WATERMARKED DATA

We showed in Sec. 5 that the baseline unlearning metrics perform poorly when similar data exists between the retain and forget sets. This is largely due to those metrics being unable to perfectly differentiate between the similar copies of data across the forget and retain sets. For these baseline metrics, the LLM is fine-tuned on the unwatermarked dataset  $\mathcal{D}_T$  in WaterDrum-Ax and WaterDrum-TOFU in Sec. 5. Here, to study the effects of watermarking on these baseline metrics, we evaluate them by fine-tuning the LLM on the watermarked dataset  $\mathcal{D}'_T$  in WaterDrum-Ax and WaterDrum-TOFU instead.

The watermarking step **P1** contributes to performance gains in **D1** separability/AUROC across metrics such as ROUGE as well, especially for the exact and semantic duplicate settings, as it makes data less similar by injecting different watermarks unique to each data owner. ROUGE does exhibit some improvement for **D1** separability for the semantic and exact duplicate settings due to the de-duplication done with watermarking (Table 9). However, **D2** alone is not sufficient as the  $\kappa$  threshold for separating forget from retain set would be unknown in practice. **D1** is needed. We observe in Table 10 that unlike WaterDrum, the other metrics still result in low  $R^2$  and fail to satisfy **D2** calibration. Specifically, the other metrics are not 0 when the forget set is not used to retrain the LLM and are not indicative of the extent of unlearning.

To summarize, using watermarked data (**P1**) may contribute to some performance gains on **D1** separability for some metrics. However, **P3** and our WaterDrum metric are essential to satisfy **D2** under **D3** (i.e., all proposed desiderata).

## G ADDITIONAL EXPERIMENTAL RESULTS

### G.1 QUANTITATIVE EVIDENCE THAT WATERMARKING WITH WATERFALL DOES NOT DEGRADE LLM PERFORMANCE

WaterDrum lays out watermarking desiderata for compatible watermarking methods (Sec. 4), including fidelity **W0**. We chose to use Waterfall (Lau et al., 2024) as their work already

1836 Table 9: AUROC ( $\pm$  across 3 seeds) of various unlearning metrics under different levels of data  
 1837 similarity for the WaterDrum-TOFU and WaterDrum-Ax datasets, when all unlearning metrics  
 1838 are evaluated on LLMs fine-tuned on watermarked data. Under this setting, both ROUGE and  
 1839 WaterDrum’s AUROC remains near 1.0 even when similar data exists.

Data Similarity	WaterDrum-TOFU			WaterDrum-Ax		
	ROUGE	Truth Ratio	WaterDrum	ROUGE	KnowMem	WaterDrum
Exact Duplicate	<b>0.926<math>\pm</math>0.051</b>	0.509 $\pm$ 0.002	<b>0.926<math>\pm</math>0.027</b>	<b>0.979<math>\pm</math>0.004</b>	0.444 $\pm$ 0.007	<b>0.957<math>\pm</math>0.008</b>
Semantic Duplicate	<b>0.977<math>\pm</math>0.001</b>	0.515 $\pm$ 0.003	<b>0.954<math>\pm</math>0.001</b>	<b>0.979<math>\pm</math>0.000</b>	0.466 $\pm$ 0.008	<b>0.963<math>\pm</math>0.001</b>
No Duplicate	<b>0.980<math>\pm</math>0.005</b>	0.727 $\pm$ 0.000	<b>0.928<math>\pm</math>0.026</b>	<b>0.983<math>\pm</math>0.000</b>	0.474 $\pm$ 0.003	<b>0.965<math>\pm</math>0.002</b>

1840  
 1841  
 1842  
 1843  
 1844  
 1845  
 1846 Table 10:  $R^2$  ( $\pm$  across 3 seeds) of various unlearning metrics under different levels of data  
 1847 similarity for the WaterDrum-TOFU and WaterDrum-Ax datasets, when all unlearning metrics  
 1848 are evaluated on LLMs fine-tuned on watermarked data. Even under this setting, only WaterDrum’s  
 1849  $R^2$  remains near 1.0 even when similar data exists.

Data Similarity	WaterDrum-TOFU				WaterDrum-Ax			
	ROUGE	Truth Ratio	MIA	WaterDrum	ROUGE	KnowMem	MIA	WaterDrum
Exact Duplicate	-7.624	-261.2	0.158	<b>0.889</b>	0.774	-23.52	-44.79	<b>0.987</b>
Semantic Duplicate	-16.31	-229.2	-5.934	<b>0.947</b>	0.677	-16.121	-45.66	<b>0.991</b>
No Duplicate	0.511	-13.71	0.567	<b>0.923</b>	0.758	-21.72	-14.42	<b>0.963</b>

1856  
 1857  
 1858 presented extensive empirical results showing that its watermarking process has minimal degradation  
 1859 on LLM performance (see App. H.3 of (Lau et al., 2024)).

1860  
 1861 Nonetheless, we evaluate Waterfall’s fidelity by comparing the fine-tuned LLM’s performance  
 1862 when trained on watermarked vs. unwatermarked data using Truth Ratio (Maini et al., 2024)  
 1863 (App. E.3), which computes each LLM’s probability of generating the correct answer compared to a  
 1864 set of wrong answers perturbed from the correct answer.

1865 Our results show that the fine-tuned LLM’s mean Truth Ratio on the WaterDrum-TOFU dataset  
 1866 when trained on watermarked data vs. unwatermarked data are very similar at 0.5121 and 0.5192,  
 1867 respectively.

## 1868 G.2 SIMILARITY OF TEXT OUTPUTS IN RETRAINED LLM

1869  
 1870  
 1871 Following the setup discussed in Sec. 5 (specifically, under ‘Robustness to similar data **D4**’), under  
 1872 the setting where the retain set ( $\mathcal{D}_{\mathcal{R}}^s = \mathcal{D}_s \cup \mathcal{D}_{\mathcal{R}}$ ) contains some data points that are similar to the  
 1873 forget set ( $\mathcal{D}_s \simeq \mathcal{D}_{\mathcal{F}}$ ) belonging to data owner(s)  $s$ , we verify that the text outputs of the LLM  
 1874  $\tilde{\varphi}^s$  fine-tuned on the retain set  $\mathcal{D}_{\mathcal{R}}^s$  are similar for the duplicate queries  $\tilde{\varphi}^s(q_d) \simeq \tilde{\varphi}^s(q_{d_f})$  where  
 1875  $d \in \mathcal{D}_s, d_f \in \mathcal{D}_{\mathcal{F}}, d \simeq d_f$ .

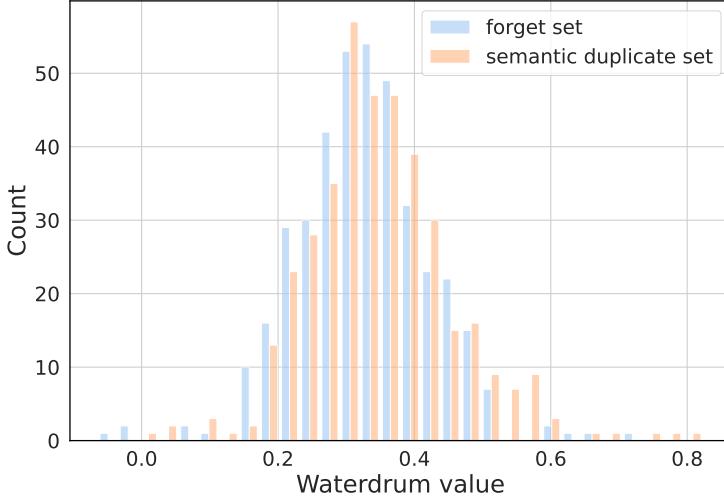
1876 We empirically verify the similarity by evaluating the STS scores between the text outputs to the  
 1877 forget query  $q_{d_f}$  and the retain query  $q_d$ . As shown in Table 11, the mean STS scores are 0.96 and  
 1878 0.87 for exact and semantic duplicates, respectively. For comparison, the STS score of query outputs  
 1879 from the same academic subject category in WaterDrum-Ax (i.e., outputs to queries from the same  
 1880 academic subject category in arXiv such as math.PR) only have a mean STS score of 0.67. This  
 1881 shows that the query outputs from the duplicate queries are very similar, much more so than queries  
 1882 from the same subject.

## 1883 G.3 SIMILAR AGGREGATE METRIC VALUES ACROSS DATA

1884  
 1885  
 1886 We verify that data points from  $\mathcal{D}_s$  and  $\mathcal{D}_f$  with similar semantics will have similar aggregate  
 1887 metric values ( $M_{d \in \mathcal{D}_s}(\varphi_{\mathcal{R}}(q_d), s) \simeq M_{d \in \mathcal{D}_{\mathcal{F}}}(\varphi_{\mathcal{R}}(q_d), \mathcal{F})$ ). We use our WaterDrum to measure  
 1888 the metric values on data points from  $\mathcal{D}_s$  and  $\mathcal{D}_f$  for the WaterDrum-Ax dataset when unlearning 1  
 1889 category. Fig. 12 shows a histogram plot of the metric values for the two different subsets with similar  
 semantics. This verifies that the distributions of metric values from the two subsets are similar.

1890 Table 11: Semantic similarity of  $q_f$  and  $q_s$  from the WaterDrum-Ax dataset. For reference, the  
 1891 STS score of text data from the same category is 0.67.

Similarity of query   STS score of query output	
Exact Duplicate	0.96
Semantic Duplicate	0.87



1914 Figure 12: Count of data with different WaterDrum value measured on  $\mathcal{D}_f$  and  $\mathcal{D}_s$  (with similar  
 1915 semantics) for the WaterDrum-Ax dataset when unlearning 1 category. The result shows that  
 1916 metric values from the two sets have a similar distribution.

#### 1918 G.4 VERIFICATION PERFORMANCE OF WATERFALL WITH DIFFERENT GENERATION LENGTHS

1920 In Sec. 5, we generated up to 200 tokens for each query when evaluating each metric (App. E.2).  
 1921 This token length roughly translates to around 5 sentences.

1923 The work of Lau et al. (2024) has shown that the verification of Waterfall improves with more  
 1924 tokens in longer text. To verify this claim, we consider the LLM fine-tuned on WaterDrum-Ax.  
 1925 With a false positive rate of 1%, the true positive rate reaches 50% at 13 tokens generated, 90% at 59  
 1926 tokens generated.

## 1927 H ADDITIONAL EXPERIMENTAL RESULTS ON LLM UNLEARNING 1929 EVALUATION

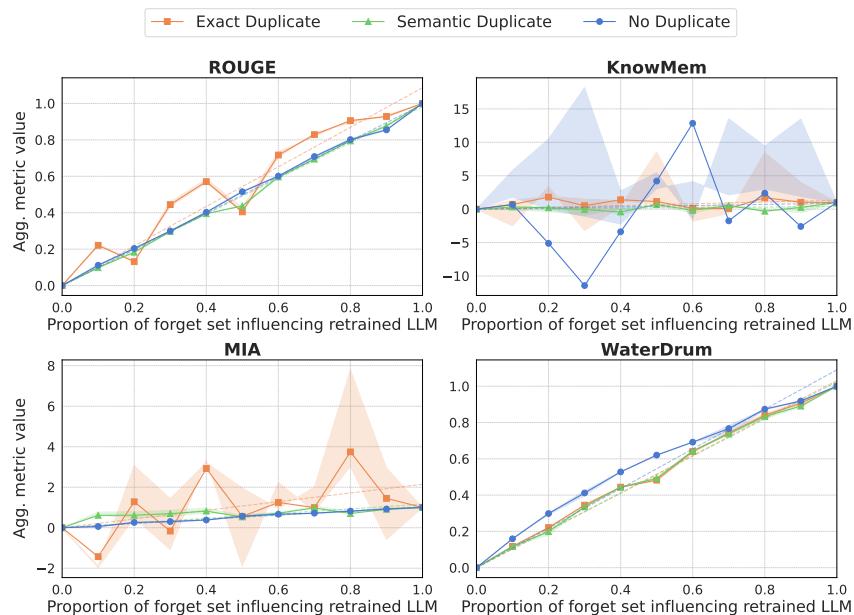
1931 In this section, we provide additional experimental results to evaluate WaterDrum and the baseline  
 1932 unlearning metrics on both WaterDrum-Ax and WaterDrum-TOFU datasets using the same  
 1933 experimental setup described in Sec. 5 (unless stated otherwise), as well as benchmark unlearning  
 1934 algorithms for the cases with multiple data owners and different levels of data similarity on the new  
 1935 WaterDrum-Ax dataset.

### 1936 H.1 LLM UNLEARNING EVALUATION ON WATERDRUM-AX DATASET

#### 1938 H.1.1 RELAXATION OF FEASIBILITY DESIDERATUM D3

1940 In Sec. 5 (specifically, under ‘Calibration desideratum D2’ and Fig. 3), we demonstrate whether the  
 1941 unlearning metrics are calibrated well or poorly without referencing the retrained LLM  $\varphi_R$ . Here, we  
 1942 relax the feasibility constraint and allow the baseline unlearning metrics (i.e., ROUGE, KnowMem,  
 1943 and MIA) to reference  $\varphi_R$  although doing so infeasibly requires retraining for every forget set being  
 considered.

1944 Specifically, we reference the retrained LLM  $\varphi_R$  (i.e., achieved by perfect unlearning) by subtracting  
 1945 its aggregate metric value from that on the unlearned model  $\tilde{\varphi}$  to yield an ‘offset’ aggregate metric  
 1946  $M_{d \in \mathcal{D}_F}^-(\tilde{\varphi}(q_d), \mathcal{F}) := M_{d \in \mathcal{D}_F}(\tilde{\varphi}(q_d), \mathcal{F}) - M_{d \in \mathcal{D}_F}(\varphi_R(q_d), \mathcal{F})$ .  
 1947 Fig. 13 and Table 12 show, respectively, the calibration curves for the various unlearning metrics  
 1948 (using the ‘offset’ aggregate metric values) and the  $R^2$  values for the corresponding best-fit lines  
 1949 under different levels of data similarity for the WaterDrum-Ax dataset. The results show that,  
 1950 under the relaxed feasibility constraint by referencing  $\varphi_R$ , the baseline metrics are generally better  
 1951 calibrated. Notably, ROUGE achieves a good calibration across different levels of data similarity even  
 1952 though it underperforms in the ‘exact duplicate’ setting. In contrast, our WaterDrum consistently  
 1953 demonstrates strong calibration with high  $R^2$  values across all settings. Nonetheless, it is important  
 1954 to emphasize that the retrained LLMs are not available in practical scenarios and their availability  
 1955 would eliminate the need to perform unlearning in the first place.  
 1956



1977 Figure 13: Calibration curves for various unlearning metrics (using the ‘offset’ aggregate metric  
 1978 values  $M_{d \in \mathcal{D}_F}^-(\tilde{\varphi}(q_d), \mathcal{F})$ ) w.r.t. proportion  $k/|\mathcal{D}_F|$  of the forget set influencing the retrained LLM  
 1979 (solid) and their best-fit lines (see associated  $R^2$  in Table 12) through the origin (dotted) under  
 1980 different levels of data similarity for the WaterDrum-Ax dataset. The ‘offset’ aggregate metric  
 1981 values are offset by referencing the retrained LLMs and scaled by referencing the original LLMs  
 1982 such that the values are 0.0 and 1.0 when the proportions are 0.0 and 1.0 respectively.  
 1983

1984 Table 12:  $R^2$  for the best-fit lines (dotted in Fig. 13) of various unlearning metrics (using the ‘offset’  
 1985 aggregate metric values) under different levels of data similarity for the WaterDrum-Ax dataset.  
 1986

Similarity	ROUGE	KnowMem	MIA	WaterDrum
Exact Duplicate	0.923	-0.331	0.273	0.994
Semantic Duplicate	0.997	0.101	-0.011	0.995
No Duplicate	0.998	0.006	0.990	0.957

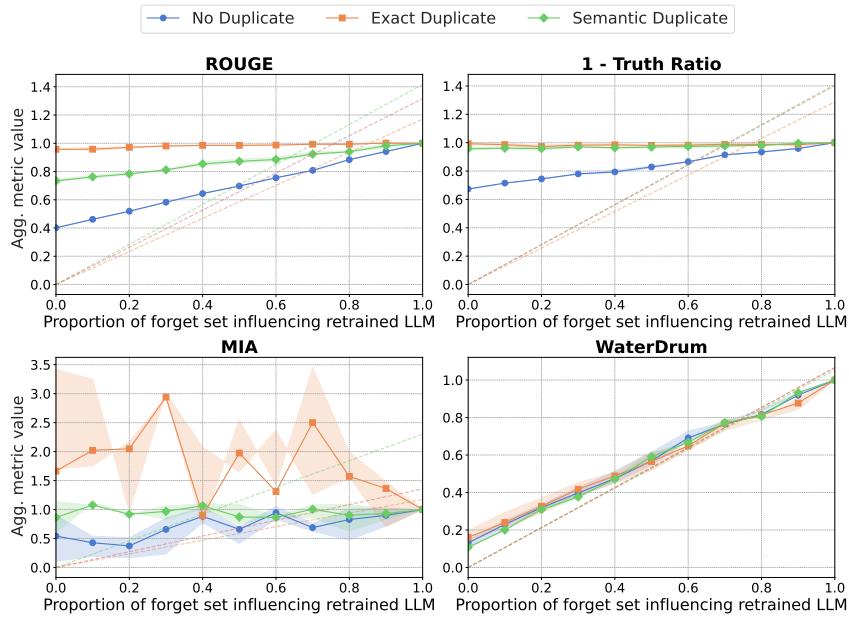
## H.2 LLM UNLEARNING EVALUATION ON WATERDRUM-TOFU DATASET

1994 As a supplement to the main experiments, we present additional experimental results here for the  
 1995 WaterDrum-TOFU dataset. As described in Sec. 5 (specifically, under ‘Robustness to similar  
 1996 data D4’), we consider the ‘exact duplicate’, ‘semantic duplicate’, and ‘no duplicate’ settings,  
 1997

1998 and fine-tune the LLMs on the WaterDrum-TOFU dataset. While Sec. 5 (specifically, under  
 1999 ‘Separability desideratum **D1**’) discusses results on the separability desideratum **D1** under different  
 2000 levels of data similarity, we report below the results to evaluate WaterDrum and the baseline  
 2001 unlearning metrics in the calibration desideratum **D2** and the relaxed feasibility desideratum **D3** under  
 2002 different levels of data similarity.

### H.2.1 CALIBRATION DESIDERATUM **D2**

2005 Fig. 14 and Table 13 show, respectively, the calibration curves for the various unlearning  
 2006 metrics and the  $R^2$  values for the corresponding best-fit lines under different levels of data  
 2007 similarity for the WaterDrum-TOFU dataset. Similar to the results in Sec. 5 (specifically, under  
 2008 ‘Calibration desideratum **D2**’), our WaterDrum outperforms the baseline metrics by ensuring  
 2009  $M'_{d \in \mathcal{D}_F}(\varphi_R(q_d), \mathcal{F})$  to be close to 0 at  $k = 0$  and maintaining strong calibration with high  $R^2$  values  
 2010 without referencing retrained LLMs across all settings.



2032 Figure 14: Calibration curves for various unlearning metrics w.r.t. proportion  $k/|\mathcal{D}_F|$  of the forget set  
 2033 influencing the retrained LLM (solid) and their best-fit lines (see associated  $R^2$  in Table 13) through  
 2034 the origin (dotted) under different levels of data similarity for the WaterDrum-TOFU dataset. Only  
 2035 WaterDrum is well-calibrated and satisfies **D2** with its best-fit lines closely following its aggregate  
 2036 metric values.

2038 Table 13:  $R^2$  values for the best-fit lines (dotted in Fig. 14) of various unlearning metrics under  
 2039 different levels of data similarity for the WaterDrum-TOFU dataset. WaterDrum achieves the  
 2040 highest  $R^2$  values that are closest to 1 and is hence a well-calibrated metric.

Similarity	ROUGE	Truth Ratio	MIA	WaterDrum
Exact Duplicate	-30.085	-6444.874	-3.480	0.889
Semantic Duplicate	-24.386	-1416.284	-41.15	0.947
No Duplicate	-2.744	-11.741	-0.838	0.923

### H.2.2 RELAXATION OF FEASIBILITY DESIDERATUM **D3**

2048 Similar to App. H.1.1, we relax the feasibility constraint here and allow the baseline unlearning  
 2049 metrics (i.e., ROUGE, Truth Ratio, and MIA) to reference the retrained LLM  $\varphi_R$  although doing so  
 2050 infeasibly requires retraining for every forget set being considered.

Fig. 15 and Table 14 show, respectively, the calibration curves for the various unlearning metrics (using the ‘offset’ aggregate metric values defined in App. H.1.1) and the  $R^2$  values for the corresponding best-fit lines under different levels of data similarity for the WaterDrum–Ax dataset. The results are similar to that in App. H.1.1 and show that, under the relaxed feasibility constraint by referencing  $\varphi_R$ , the baseline metrics are generally better calibrated. Unlike Truth Ratio and MIA, our WaterDrum and ROUGE consistently demonstrate strong calibration with high  $R^2$  values across all settings. Nonetheless, it is important to emphasize again that the retrained LLMs are not available in practical scenarios and their availability would eliminate the need to perform unlearning in the first place.

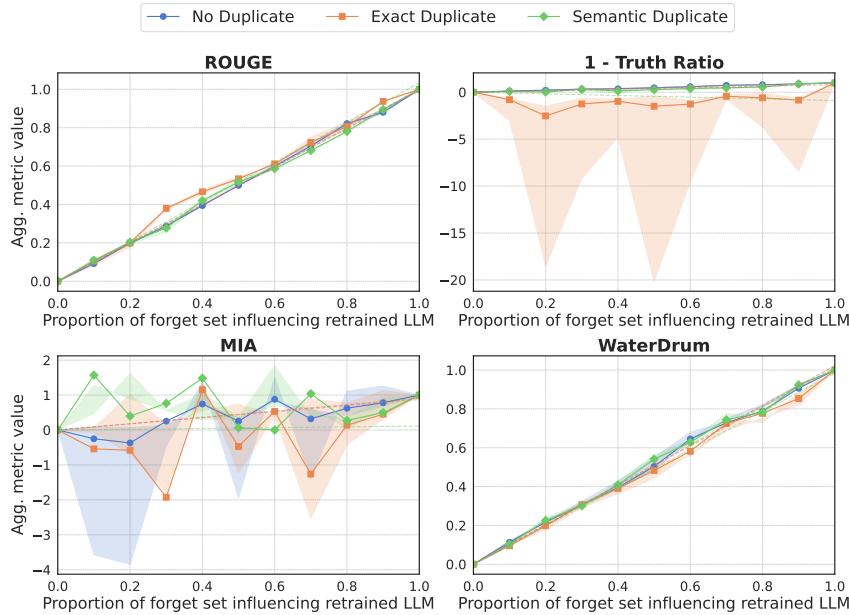


Figure 15: Calibration curves for various unlearning metrics (using the ‘offset’ aggregate metric values) w.r.t. proportion  $k/|\mathcal{D}_F|$  of the forget set influencing the retrained LLM (solid), scaled by referencing the retrained and original LLMs, and their best-fit lines (see associated  $R^2$  in Table 14) through the origin (dotted) under different levels of data similarity for the WaterDrum–TOFU dataset.

Table 14:  $R^2$  for the best-fit lines (dotted in Fig. 15) of various unlearning metrics (using the ‘offset’ aggregate metric values) under different levels of data similarity for the WaterDrum–TOFU dataset.

Similarity	ROUGE	Truth Ratio	MIA	WaterDrum
Exact Duplicate	0.991	-0.586	-0.018	0.997
Semantic Duplicate	0.998	0.854	-0.417	0.996
No Duplicate	0.999	0.995	0.608	0.997

### H.3 BENCHMARKING UNLEARNING ALGORITHMS ON NEW WATERDRUM–AX DATASET FOR MULTIPLE DATA OWNERS AND DIFFERENT LEVELS OF DATA SIMILARITY

In addition to the experimental results in Sec. 5.1, Figs. 16 and 17 illustrate the use of WaterDrum in benchmarking the unlearning algorithms under the respective ‘no duplicate’ and ‘exact duplicate’ settings of data similarity (i.e., previously described in Sec. 5, specifically, under ‘Robustness to similar data **D4**’) for the WaterDrum–Ax dataset where the forget set consists of data from 1, 3, and 5 data owners (out of a total of 20 data owners) with 1 category of paper abstracts per owner (App. C.3).

Similar to the results in Sec. 5.1, it can be observed from Fig. 16 (Fig. 17) that the unlearning algorithms achieve aggregate WaterDrum values still far from that achieved by retraining: KL

and TV generally produce unlearned models that unlearn the forget set  $\mathcal{D}_F$  very well but cannot preserve the influence of  $\mathcal{D}_R$  (or  $\mathcal{D}_s$ ) much, the latter of which compromises their overall utility. GD and SCRUB tend to produce unlearned models that preserve some influence of  $\mathcal{D}_R$  (or  $\mathcal{D}_s$ ) but do not unlearn the forget set  $\mathcal{D}_F$  well. However, both GD and SCRUB require fine-tuning on the (augmented) retain set ( $\mathcal{D}_R^s = \mathcal{D}_s \cup \mathcal{D}_R$ ), which incurs a significant amount of computational resources as the (augmented) retain set is likely to be significantly larger than the forget set and almost similar in size to the full dataset. Typically, LLM fine-tuning only involves very few epochs (Touvron et al., 2023). The computational cost of fine-tuning the LLM for a few epochs on the (augmented) retain set can be almost as expensive as that of retraining.

It can also be observed from Fig. 16 (Fig. 17) that when the forget set consists of data from 5 data owners, the aggregate WaterDrum value of the watermarked retain set  $\mathcal{D}_R$  in WaterDrum-Ax on the retrained LLM (only on the (augmented) retain set) increases slightly beyond 1.0. We hypothesize that this is due to the forget set constituting a larger proportion of the entire dataset (i.e., 5 out of a total of 20 data owners). As a result, the (augmented) retain set used for retraining becomes smaller in proportion relative to the full dataset  $\mathcal{D}_T$ , which can result in the retrained LLM becoming more specialized in this smaller (augmented) retain set and in turn a larger aggregate WaterDrum value. The same reasoning applies to explain why the aggregate WaterDrum value of  $\mathcal{D}_s$  on the retrained LLM (only on the (augmented) retain set) also increases slightly beyond 1.0 in Fig. 17.

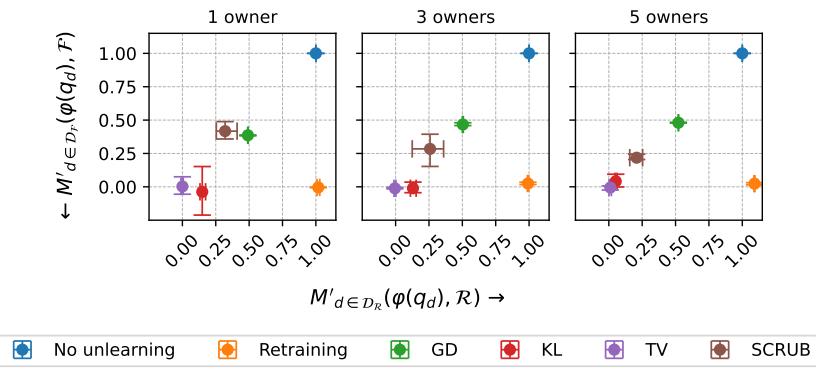


Figure 16: Benchmarking the unlearning algorithms with WaterDrum under the ‘no duplicate’ setting of data similarity for the WaterDrum-Ax dataset where the forget set consists of data from 1, 3, and 5 data owners with 1 category of paper abstracts per owner (App. C.3).

## I OTHER QUESTIONS AND LIMITATIONS

1. **What is the difference with existing watermark-based unlearning metric?** Existing watermark-based unlearning metrics are mostly for image-based classification model, as opposed to our metric for text-based generative LLMs. See discussion on watermark-based metrics in App. A for details.
2. **Existing works (Liu et al., 2025; Lynch et al., 2024) have already identified similar limitations about existing unlearning metrics. What is the novelty of the work?** We formally define clear desiderata and propose a non-retraining-based metric that works despite greater similarity between the forget and retain set and the generalization ability of LLMs. See more discussion in App. A.
3. **Why do we only run experiments on TOFU and WaterDrum-Ax instead of other datasets such as WMDP?** TOFU and WaterDrum-Ax cover both LLM question answering and generation tasks, which are representative of LLM tasks. WMDP is different from TOFU and WaterDrum-Ax in nature because it is specifically for knowledge editing and only contains test data instead of training data. As our work considers a data-centric view of unlearning, we are concerned with the unlearning of specific data owners’ contribution (with potential similar overlapping data across data owners), rather than indiscriminately unlearning certain (harmful) knowledge.

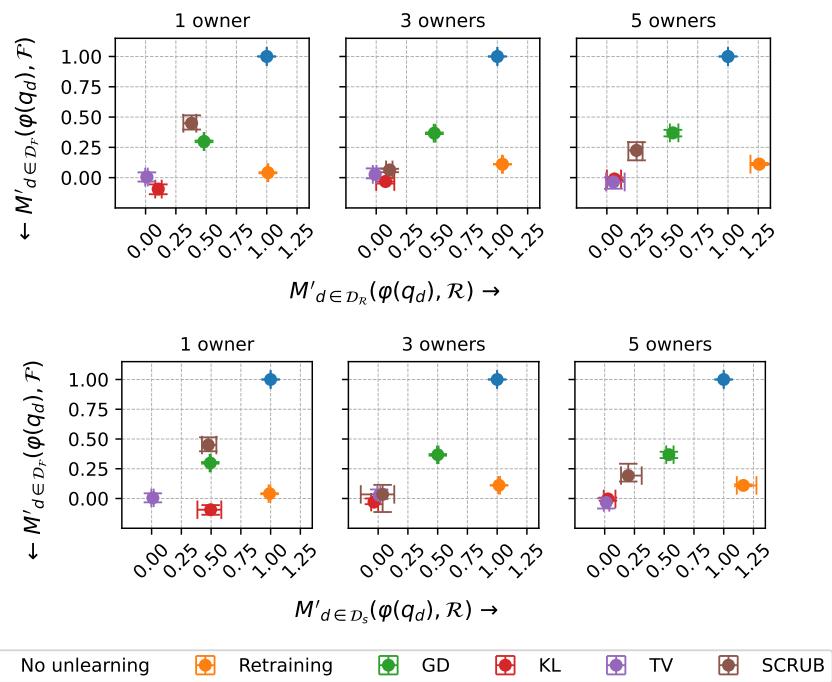


Figure 17: Benchmarking the unlearning algorithms with WaterDrum under the ‘exact duplicate’ setting of data similarity (i.e., previously described in Sec. 5, specifically, under ‘Robustness to similar data **D4**’) for the WaterDrum–Ax dataset where the forget set consists of data from 1, 3, and 5 data owners with 1 category of paper abstracts per owner (App. C.3).

4. **Can our conclusion be generalized to other datasets or other LLMs?** Results on Phi-1.5 (see App. F.3) show that the conclusions can be generalized to other LLMs as well. The two LLMs considered in our paper are representative of recent LLMs, different in terms of model architectural details, and span different model scales. These two LLMs are also the only LLMs considered in (Maini et al., 2024; Wang et al., 2025).
5. **Beyond unlearning effectiveness, can our watermark metric be used to measure utility preservation/retention?** As shown in Sec. 5.1, our metric can be used to verify that the metric on the retain set in the unlearned LLM is similar to that in the original LLM. Hence, by verifying the retain watermark, our metric can also quantify the extent of undesirable removal of the retain set’s influence and evaluate the effects of catastrophic forgetting.
6. **Practical significance of unlearning from fine-tuning data vs pre-training data.** In real-life applications, LLM fine-tuning is performed to enhance the LLM in specific downstream tasks, which is more likely to make use of task-specific datasets. These datasets are more concerned with privacy/safety issues, and are hence more significant for unlearning than public datasets.
7. **What are the limitations of WaterDrum and this work?** The limitations are that (a) the desiderata may not be exhaustive, (b) the WaterDrum value (via the watermark’s verification score in Eq. (3)) may not exhaustively capture all possible ways of measuring unlearning effectiveness, and (c) WaterDrum requires the training data to be watermarked unlike existing metrics.

We believe that for now, (a) and (b) are acceptable as our work is an important *first* step towards designing and developing more effective and practical unlearning metrics and algorithms, and deriving theoretical results for them. Future work can conduct a more comprehensive and systematic evaluation of existing LLM unlearning algorithms and adapt theoretical insights from the watermarking community to analyze the LLM unlearning metrics based on the new connection that we have established in this work.

2214 In remark 1 and App. D, we explain why watermarking is lightweight, easy to use and  
2215 would be a more common practice in the future. Thus, the applicability of WaterDrum  
2216 would increase and the limitation (c) would diminish over time. Moreover, limitation  
2217 (c) is reasonable as the benefits, such as satisfying our desiderata, outweigh the slight  
2218 inconvenience and cost.

2219 **8. What new insights can be gained from the proposed metric WaterDrum?** **(a)** We  
2220 showed that existing metrics fail on our necessary desiderata (Sec. 3), prompting caution on  
2221 metrics design. **(b)** Using WaterDrum to benchmark LLM unlearning algorithms (Sec. 5.1)  
2222 shows that they perform poorly on unlearning and retaining performance. WaterDrum can  
2223 serve as an optimization criterion for future LLM unlearning algorithms. **(c)** By emphasizing  
2224 practical conditions, WaterDrum encourages future LLM unlearning algorithms to consider  
2225 realistic constraints.

2226 **9. Why do we not consider other desiderata?** Our work focuses on the most essential  
2227 desiderata (effectiveness desiderata) and more practical/realistic settings. These desiderata  
2228 are those that we find to be most relevant necessary criteria for effective unlearning metrics,  
2229 though they are not meant to be exhaustive nor by themselves sufficient to guarantee  
2230 unlearning. We see our work as complementary to other compatible frameworks.

2231  
2232  
2233  
2234  
2235  
2236  
2237  
2238  
2239  
2240  
2241  
2242  
2243  
2244  
2245  
2246  
2247  
2248  
2249  
2250  
2251  
2252  
2253  
2254  
2255  
2256  
2257  
2258  
2259  
2260  
2261  
2262  
2263  
2264  
2265  
2266  
2267