f-SCRUB: Unbounded Machine Unlearning Via f-divergences

Anonymous authors
Paper under double-blind review

Abstract

Deep Machine Unlearning addresses the problem of removing the effect of a subset of data points from a trained model. Machine Unlearning has various implications for the performance of algorithms. A well-known algorithm, SCRUB (?), has served as a baseline and achieved key objectives such as removing biases, resolving confusion caused by mislabeled data in trained models, and allowing users to exercise their "right to be forgotten" to protect user privacy. Building on this algorithm, we introduce f-SCRUB, an extension of SCRUB that employs different f-divergences instead of KL divergence. We analyze the role of these divergences and their impact on the resolution of unlearning problems in various scenarios.

0.1 Federated Machine Unlearning

we first begin to underestand how can we tackle this prblem from algorithmic point of view. Consider a machine learning model $\theta \sim \mathcal{A}(S)$ trained on a dataset S that is distributed across C clients, where each client c holds a local dataset $S^{(c)}$. In federated unlearning, each client may request to forget a subset $S_F^{(c)} \subset S^{(c)}$, resulting in a client-specific retain set $S_R^{(c)} = S^{(c)} \setminus S_F^{(c)}$. The global retain set is then $S_R = \bigcup_{c=1}^{C} S_R^{(c)}$. There is in general two way to deal with this.mislabeled Federated Exact Unlearning:

Let $\mathcal{A}(S)$ be the federated training algorithm and θ the resulting global model. An unlearning algorithm $\mathcal{U} : \Theta \times 2^{|S|} \to \Theta$ is said to perform exact unlearning in a federated setting if:

$$\mathcal{U}(\mathcal{A}(S), \{S_F^{(c)}\}_{c=1}^{C}) \stackrel{d}{=} \mathcal{A}(S_R),$$

Federated Approximate Unlearning: For this purpose the whole point is to not payt the cost of exact unlearning, meaning the computation cost.

For approximate unlearning the definition of machine unlearning in the literature carries inherent ambiguity, primarily because it is highly dependent on the scenario under consideration. To enhance clarity, we distinguish between two major paradigms of unlearning, each motivated by different objectives.

Some works (?) treat unlearning as a robustness problem: the aim is to mitigate the effect of unwanted or harmful data that was mistakenly included in the training dataset. These cases often arise due to mislabeled data, poisoned examples, or distributional shifts.

In contrast, other works (?) focus on unlearning from a privacy perspective, where the goal is to remove the *actual data* and its influence due to legal or ethical reasons. This line of research is closely related to privacy regulations such as the "Right to be Forgotten" under the GDPR.

- Objective Ambiguity: Is the purpose of unlearning to improve robustness or generalization (e.g., mitigate the impact of poisoned or low-quality data), or is it primarily driven by privacy concerns that demand the complete removal of any influence of certain data?
- Outcome Ambiguity: Should a successful unlearning algorithm produce a model that is functionally equivalent to retraining on the retain set (i.e., model-level equiv-

alence), or should it ensure that an adversary cannot determine whether a specific data point was ever part of training (i.e., privacy-level indistinguishability)?

To resolve these ambiguities, we categorize unlearning objectives based on the underlying motivation and provide formal definitions within the federated learning framework.

0.1.1 Scenario I: Unlearning the Effect (Robustness-Oriented)

Imagine we have a model trained in the federated framework and one or more of the client realize that some or whole their training dataset is not correct and they should not have used it during the training. Let's say in this case the uncorrect data were poisioned by backdoor attack. In this case, the model should be able to remove the influence of the forget set data from the trained model. This is a clear case of unlearning, where the goal is to remove the influence of specific data points from the model without retraining it from scratch. The metric to evaluate Consider a machine learning model $\theta \sim \mathcal{A}(S)$ trained on a dataset S that is distributed across multiple clients in a federated setting. Given a "forget set" $S_F \subset S$ and the corresponding "retain set" $S_R = S \setminus S_F$, the goal of a federated unlearning algorithm is to efficiently remove the influence of S_F from the global model without requiring full retraining or centralized access to the original dataset.

In this scenario, the goal is to remove the effect of certain training data without necessarily ensuring privacy guarantees. This is often motivated by robustness: e.g., unlearning mislabeled or backdoored examples that degrade model quality.

Example. Consider a federated learning setup with C clients. Each client $c \in [C]$ holds a local dataset $S^{(c)}$, which may contain harmful examples (e.g., poisoned, mislabeled). Let $S_F^{(c)} \subset S^{(c)}$ denote the forget set on client c, and $S_R^{(c)} = S^{(c)} \setminus S_F^{(c)}$ the retain set.

Federated Update. The global model parameters w are updated using the Federated Averaging (FedAvg) algorithm:

$$w_t = \sum_{c=1}^{C} \frac{n_c}{n} w_t^{(c)}, \quad \text{where } w_t^{(c)} \text{ is the local model from client } c,$$

and $n_c = |S^{(c)}|$, $n = \sum_{c=1}^{C} n_c$.

Desirable Property: Improved Performance on Forget Set. After unlearning, the empirical loss of the global model w on the forget data (now corrected or removed) should be lower:

$$\mathcal{L}_F(w_{\text{before}}) \geq \mathcal{L}_F(w_{\text{after}}),$$

where:

$$\mathcal{L}_F(w) := \sum_{c=1}^{C} \frac{n_c^F}{n_F} \mathcal{L}_{S_F^{(c)}}(w),$$

with $n_c^F = |S_F^{(c)}|$, $n_F = \sum_{c=1}^{C} n_c^F$.

Global Generalization Constraint. Ideally, the model's performance on the global retain set should not degrade:

$$\mathcal{L}_R(w_{\text{after}}) \leq \mathcal{L}_R(w_{\text{before}}),$$

where:

$$\mathcal{L}_R(w) := \sum_{c=1}^{C} \frac{n_c^R}{n_R} \mathcal{L}_{S_R^{(c)}}(w), \quad n_c^R = |S_R^{(c)}|, \quad n_R = \sum_{c=1}^{C} n_c^R.$$

Remarks.

- The focus here is on mitigating the effect of unwanted data, not guaranteeing its absence.
- No assumptions are made about privacy or membership inference.
- Performance is measured in terms of generalization and empirical loss.
- This is particularly relevant in scenarios such as backdoor unlearning, label noise correction, or data shift adaptation.

0.2 Federated Machine Unlearning

C

0.3 f-SCRUB for Federated Unlearning

We propose f-SCRUB, a flexible extension of SCRUB for federated settings. Each client independently separates its loss into divergence-based components.

Maximization Loss (per client):

$$\frac{1}{N_f^{(c)}} \sum_{x_f \in S_F^{(c)}} d_f(x_f; w_u^{(c)})$$

Minimization Loss (per client):

$$\frac{\alpha}{N_r^{(c)}} \sum_{x_r \in S_R^{(c)}} d_f(x_r; w_u^{(c)}) + \frac{\gamma}{N_r^{(c)}} \sum_{(x_r, y_r) \in S_R^{(c)}} \ell_f(h(x_r; w_u^{(c)}), \mathbf{Y}_r)$$

where $d_f(\cdot)$ and $\ell_f(\cdot, \cdot)$ are derived from a chosen f-divergence. We consider three divergences—Kullback-Leibler (KL), Jensen-Shannon (JS), and χ^2—to control the tradeoff between forgetting and retention.

Clients send locally optimized models (or gradients) to a central aggregator, which combines them into an updated global model with reduced influence from the union of forget sets. Details of the f-divergence choices and their effects are discussed in Appendix B.

1 Experiments and Results

We evaluate f-SCRUB in a federated environment using the CIFAR-10 dataset. Data is non-IID and partitioned across clients. Each client locally unlearns portions of its data and participates in federated rounds. ResNet-18 is used as the base model, following the protocol in (?).

2 Introduction

Rapid advancements in modern machine learning systems, coupled with their widespread adoption across various domains, have raised an important question: What happens if a user no longer wants their data to be utilized? This issue, along with the EU's 'right to be forgotten' (?), presents the challenge of removing or 'unlearning' the impact of specific training examples from a trained model. Beyond user privacy, model safety is also a critical concern, particularly in mitigating the effects of toxic, outdated, or poisoned data. Addressing these challenges is essential for securing foundation models and ensuring the reliability and robustness of classical machine learning models, such as classifiers.

One of the prominent approaches for machine unlearning, SCRUB (?), introduces a teacher-student framework where the student selectively discards knowledge related to the data to be removed. The versatility of SCRUB allows it to avoid other methods' scalability and assumption constraints. However, it faces challenges in balancing the model's performance on retained data while achieving high error on removed data.

Table 1: Divergences and their corresponding generator functions

Divergence	Generator Function $f(t)$
KL-divergence	$t \log t$
χ^2-divergence	$(1-t)^2$
JS-divergence	$t \log\left(\frac{2t}{1+t}\right) + \log\left(\frac{2}{1+t}\right)$

Our Contribution: In this work, we introduce f-SCRUB, an extension of SCRUB that incorporates a novel framework based on f-divergences. In particular, our contributions are, (a) using f-divergences in SCRUB framework, (b) comprehensive experiments investigating different combinations of f-divergences in f-SCRUB.

3 Preliminaries

f -divergence: The f -divergences are information measures that generalize various divergences, such as Kullback-Leibler (KL) divergence, through the use of a convex generator function f. Given two discrete distributions $P = \{p_i\}_{i=1}^k$ and $Q = \{q_i\}_{i=1}^k$, the f-divergence between them is defined as:

$$D_f(P \parallel Q) := \sum_{i=1}^{k} q_i f\left(\frac{p_i}{q_i}\right)$$

where $f : (0, \infty) \to \mathbb{R}$ is a convex function with the property that $f(1) = 0$. This definition implies that $D_f(P \parallel Q) = 0$ if and only if $P = Q$. By choosing different forms for f, we obtain different types of divergences. For example, when $f(t) = t \log(t)$, we get the Kullback-Leibler (KL) divergence, which measures the difference between two probability distributions. In this work, we focus on JS-divergence and χ^2-diveregence.

3.1 Machine Unlearning

Consider a machine learning model $\theta \sim \mathcal{A}(S)$ trained on a dataset S. Given a "forget set" $S_F \subset S$ and a corresponding "retain set" $S_R = S \setminus S_F$, the goal of an exact unlearning algorithm is to produce a sample from $\mathcal{A}(S_R)$ starting from the trained model θ.

Definition 3.1 (Exact unlearning (?)) An unlearning algorithm $\mathcal{U} : \Theta \times 2^{|S|} \to \Theta$ is considered an exact unlearning algorithm if, for all $S_F \subset S$, $\mathcal{U}(\mathcal{A}(S), S_F) \stackrel{d}{=} \mathcal{A}(S_R)$, where $\stackrel{d}{=}$ denotes equality in distribution over models.

Definition 3.2 (Approximate unlearning) An unlearning algorithm $\mathcal{U} : \Theta \times 2^{|S|} \to \Theta$ is said to perform approximate unlearning if, for all $S_F \subset S$, the output of $\mathcal{U}(\mathcal{A}(S), S_F)$ is close to $\mathcal{A}(S_R)$ in terms of some divergence measure $d(\cdot, \cdot)$, i.e.,

$$d(\mathcal{U}(\mathcal{A}(S), S_F), \mathcal{A}(S_R)) \leq \epsilon,$$

where ϵ is a small constant, indicating that the model after unlearning is approximately equivalent to a model retrained on the retain set S_R.

SCRUB: The SCRUB method (?) proposes a novel approach to unlearning as a Approximate unlearning method, where a student model is trained to selectively obey a teacher model. The goal is twofold: to forget the forget set S_F while still retaining knowledge about the retain set S_R. The model w^u (the student) is initialized with the teacher's weights w^o, and the key idea is to optimize the student's performance on the retain set while forgetting the forget set. The loss function used in SCRUB incorporates several components. It begins with the Kullback-Leibler (KL) divergence between the student and teacher output distributions for each example x, given by:

$$d_{\mathrm{KL}}(x; w^u) = D_{KL}(h(x; w^u)||h(x; w^o)),$$

where $h(x; w^u)$ is the output of Softmax layer. This encourages the student model to stay close to the teacher for the retain set, ensuring it performs well on S_R. However, to encourage forgetting the forget set, the method adds a contrastive term to the objective, which forces the student to move away from the teacher on examples from the forget set S_F. The objective then becomes:

$$\min_{w^u} \frac{1}{N_r} \sum_{x_r \in S_R} d_{\mathrm{KL}}(x_r; w^u) - \frac{1}{N_f} \sum_{x_f \in S_F} d_{\mathrm{KL}}(x_f; w^u)$$

Furthermore, SCRUB simultaneously optimizes the task loss on the retain set to further enhance performance on the relevant examples, resulting in the final loss function:

$$\min_{w^u} \frac{\alpha}{N_r} \sum_{x_r \in S_R} d_{\mathrm{KL}}(x_r; w^u) + \frac{\gamma}{N_r} \sum_{(x_r, y_r) \in S_R} \ell(h(x_r; w^u), \mathbf{Y}_r) - \frac{1}{N_f} \sum_{x_f \in S_F} d_{\mathrm{KL}}(x_f; w^u),$$

where ℓ represents the cross-entropy loss, and α and γ are hyperparameters controlling the importance of each term and $\mathbf{Y}_r$ is the hot-encode of labels vector for given feature x_r. This formulation allows SCRUB to balance the tradeoff between retaining performance on the retain set and forgetting data from the forget set, addressing the core challenge of machine unlearning.

3.2 f-SCRUB

Here we introduce f-SCRUB, a novel approach for unlearning in machine learning models. We separate the losses based on (?) into two distinct components.
Maximization Loss: This loss is defined as

$$\frac{1}{N_f} \sum_{x_f \in S_F} d_f(x_f; w^u),$$

where aims to maximize the divergence between the unlearned model and the data points that need to be forgotten.
Minimization Loss: This loss is defined as

$$\frac{\alpha}{N_r} \sum_{x_r \in S_R} d_f(x_r; w^u) + \frac{\gamma}{N_r} \sum_{(x_r, y_r) \in S_R} \ell_f(h(x_r; w^u), \mathbf{Y}_r),$$

where $\ell_f(\cdot, \cdot)$ is the loss function inspired via f-divergence. We aim to minimize the divergence between the unlearned model and the remaining data points, while also ensuring that the model's predictions remain accurate.
We choose different f-divergences for each component in different scenarios. We modify our loss functions and introduce f-SCRUB by selecting different f-divergences for minimization and maximization losses. While f-SCRUB allows the flexibility to explore various f-divergences as loss functions, in this work, we limit our choices to three divergences. In particular, the divergence terms $d_f(x_r; w^u)$ and $d_f(x_f; w^u)$ are chosen from the Kullback-Leibler (KL), Jensen-Shannon (JS), and χ^2 divergences. The rationale behind this selection is provided in Appendix B.

4 Experiment and Results

For our simulations, we use the same framework as (?). We conducted our experiments on the CIFAR-10 dataset and selected ResNet-18 as our model. You can find our code at Ananymous github. You can find the details of our simulations in Appendix D.

Scenarios: In the literature, two common forgetting scenarios have been discussed: forgetting an entire class (Class 5) and forgetting a subset of 100 examples from Class 5. To extend these investigations, we introduce more challenging scenarios, summarized in Table 2. We provide the motivations for choosing these scenarios in Appendix C.

Overshoot / Undershoot: As noted in ?, one of the challenges SCRUB faces is the uncertainty in the loss function values for forget set members. Since these values are unknown, simply

Table 2: Forgetting scenarios.

Scenario Name	Classes	Number to Forget
Complete (1)	Entire class 5	All
Light (2)	Class 5	100
Moderate (3)	Class 5	500
Dual Light (4)	Classes 4, 5	100 each
Dual (5)	Classes 4, 5	500 each
Broad Light (6)	Classes 1, 2, 3, 4, 5	100 each
Broad (7)	Classes 1, 2, 3, 4, 5	500 each
Extended Light (8)	Classes 1, 2, 3, 4, 5, 6	100 each
Extended (9)	Classes 1, 2, 3, 4, 5, 6	500 each

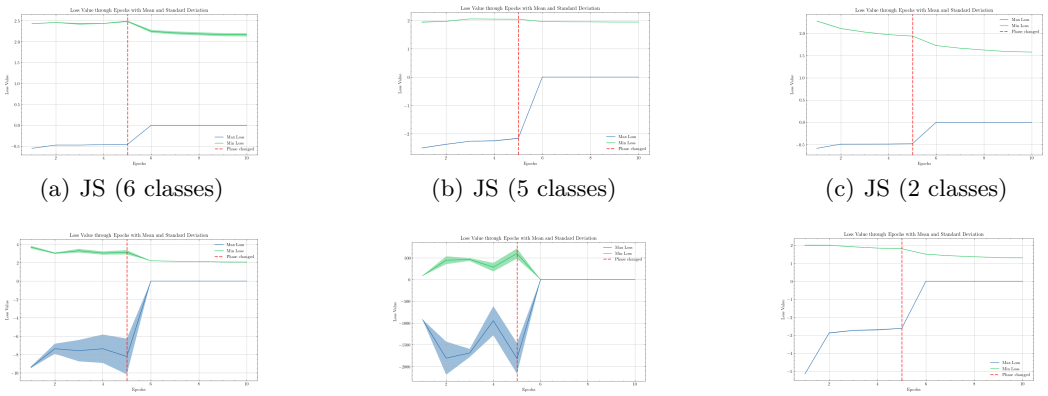
increasing their loss may not be an optimal solution. Depending on the number of training epochs, SCRUB can lead to overshooting or undershooting the intended loss adjustment. Therefore, we aim to explore whether using a more robust loss function, such as JS or χ^2 divergence, can yield a loss function that is inherently more stable and reliable in unlearning scenarios. Since KL divergence has become the standard loss function for the retain set, we focus on using JS and χ^2 divergences as the loss functions for the forget set. A detailed analysis of other loss functions is provided in the appendix.

In simpler scenarios, such as unlearning 100 data points from a single class, using KL divergence as the loss function for the forget set does not exhibit significant variance. However, in more challenging unlearning scenarios, such as unlearning 500 data points across six classes, the variance of the forget set loss increases significantly. In contrast, JS divergence remains more stable, demonstrating lower variance even in complex unlearning settings.

In Figure 1, we present the loss values where JS divergence is applied to the forget set and KL divergence to the retain set. These results correspond to the Exceptionally Challenging, Highly Difficult, and Difficult scenarios.

In contrast, the right figure shows a case where KL divergence is applied to both sets. As observed, the forget set exhibits higher variance and greater data dependence when using KL compared to JS.

As shown in Figures 1(a) and 1(d), KL divergence is highly dependent on the data and exhibits significant variance. A similar trend is observed in Figures 1(b) and 1(e). However, when unlearning a smaller number of data points, neither JS nor KL shows substantial variance in their loss values. This suggests that in simpler unlearning scenarios, where the process is less sensitive to data variations, both divergences behave similarly, as seen in Figures 1(c) and 1(f).



(a) JS (6 classes) (b) JS (5 classes) (c) JS (2 classes)

(d) KL (6 classes) (e) KL (5 classes) (f) KL (2 classes)

Figure 1: The Max Loss represents the loss of the forget set, which we aim to maximize, while the Min Loss corresponds to the retain set loss, which we seek to minimize. Notably, in the first phase, each epoch involves both a maximization and a minimization step. However, after transitioning to the next phase, we perform only minimization.

5 Discussion

We analyze the performance of the models across different scenarios. In this section, we focus on KL-JS scenario, where we replace the KL-divergence maximization loss in SCRUB with JS-divergence, while additional cases are presented in the Appendix F. The best performance is achieved when the error in the forget set is maximized while the error in the test dataset is minimized. As shown in Table 3, using JS loss as the maximization loss generally results in lower variance across almost most of all scenarios. c

To interpret these results, we define the best loss function as the one where the forgotten error is highest and the test error is lowest—both occurring in the same row. However, this is not always the case. With a more nuanced analysis, we argue that KL-JS performs better in most scenarios with confidence. In cases where it does not, model degradation complicates the analysis and introduces significant complexity. Additionally, when one approach excels in forget set error while the other performs better on test error, direct comparison becomes infeasible, preventing a definitive judgment.

In the complete forgetting scenario, where the goal is to forget all data from a specific class, the error on the forget set rapidly reaches 100%. This phenomenon is also evident in Table 6 and other tables in the appendix. After two max-min epochs, minimizing with the largest possible batch size yields strong results, as demonstrated in ?. The key point here is that because the entire class is absent, post-minimization does not affect the forget error as presented in Table 3.

As shown in Table 3, in the Light scenario, KL-JS outperforms the baseline. The only exception is in the full-capacity forget error, where KL-JS exhibits a marginally lower performance (0.34%); however, this is negligible given that the baseline has twice the variance. The same pattern holds for the Light-Dual case, where KL-JS has a 0.16% difference in the full-capacity scenario but nearly four times lower variance. In the Moderate case, KL-JS clearly outperforms the baseline in the Vanilla setup, but in the full-capacity scenario, direct comparison is not feasible. In the full capacity case for Light-Broad, the KL−JS outperforming baseline is determined, and for Light-Extended comparison is not feasible.

Our observations indicate that in the Vanilla case for Moderate (forgetting 500 samples) and Dual up to Extended cases, the degradation in model performance is so severe that one could argue the model has effectively lost its knowledge. This highlights a critical limitation of Scrub—widely regarded as the best unlearning framework based on current literature—when the number of deleted samples per class increases. This issue presents emerging challenges in the field, which are highly relevant to real-world scenarios. A similar problem is also addressed in (?). Broad and Extended cases in full capacity also suffer the

Table 3: Results for various forgetting scenarios with KL-KL being the baseline and KL-JS as main scenario, training for five intermittent maximization-minimization steps followed by five minimization step

Scenario	Loss	Vanilla			Full capacity		
		Forget Error	Retain Error	Test Error	Forget Error	Retain Error	Test Error
Complete	KL-KL	100.00 ± 0.00	80.16 ± 7.88	82.28 ± 7.10	100.00 ± 0.00	35.43 ± 4.89	43.11 ± 3.99
	KL-JS	100.00 ± 0.00	81.16 ± 2.73	83.01 ± 2.47	100.00 ± 0.00	50.50 ± 0.75	56.34 ± 0.69
Light	KL-KL	25.33 ± 7.41	22.73 ± 0.93	26.92 ± 0.74	26.67 ± 4.92	8.93 ± 0.06	14.88 ± 0.05
	KL-JS	25.33 ± 3.30	22.26 ± 0.36	26.43 ± 0.22	26.33 ± 2.25	8.80 ± 0.09	14.74 ± 0.13
Moderate	KL-KL	100.00 ± 0.00	72.51 ± 4.67	73.05 ± 4.55	33.80 ± 6.88	14.14 ± 1.54	18.64 ± 1.47
	KL-JS	99.73 ± 0.19	60.09 ± 2.11	61.79 ± 1.90	39.53 ± 4.40	17.67 ± 0.33	21.68 ± 0.14
Light-Dual	KL-KL	30.67 ± 3.42	23.73 ± 0.21	27.83 ± 0.12	21.33 ± 3.32	10.25 ± 0.90	15.76 ± 0.42
	KL-JS	29.50 ± 2.55	23.36 ± 0.10	27.50 ± 0.06	21.17 ± 0.82	9.41 ± 0.11	15.24 ± 0.06
Dual	KL-KL	100.00 ± 0.00	79.05 ± 0.15	79.59 ± 0.21	35.40 ± 1.68	22.96 ± 0.27	26.27 ± 0.09
	KL-JS	100.00 ± 0.00	79.01 ± 0.12	79.56 ± 0.09	49.83 ± 1.49	31.97 ± 0.40	34.23 ± 0.40
Light-Broad	KL-KL	83.33 ± 8.68	66.17 ± 9.32	67.05 ± 8.67	39.00 ± 0.99	24.07 ± 1.04	27.36 ± 1.01
	KL-JS	63.47 ± 4.74	56.33 ± 3.69	57.84 ± 3.25	40.60 ± 1.50	23.70 ± 0.30	26.89 ± 0.18
Broad	KL-KL	95.01 ± 7.02	73.05 ± 4.30	74.36 ± 3.71	91.96 ± 4.83	57.03 ± 1.95	59.88 ± 1.82
	KL-JS	100.00 ± 0.00	78.86 ± 0.11	80.11 ± 0.07	91.19 ± 2.78	56.87 ± 1.06	59.55 ± 1.08
Light-Extended	KL-KL	75.28 ± 8.37	69.35 ± 7.49	69.83 ± 7.11	38.72 ± 0.93	31.38 ± 3.63	33.57 ± 3.51
	KL-JS	72.83 ± 2.72	72.55 ± 4.75	72.88 ± 4.47	36.22 ± 1.33	28.92 ± 3.04	31.51 ± 2.66
Extended	KL-KL	88.73 ± 7.97	82.37 ± 6.34	82.81 ± 5.48	69.90 ± 7.66	53.52 ± 3.98	55.38 ± 4.27
	KL-JS	90.72 ± 2.33	78.00 ± 0.58	79.04 ± 0.47	85.78 ± 4.79	61.28 ± 2.71	63.76 ± 2.82

same problem mentioned here. As you can see, there is not a single case where KL−KL outperforms KL−JS with full confidence and not degraded model.

6 Conclusion

As final conclusion, we introduced f-SCRUB, an extension of SCRUB that incorporates f-divergences to improve the stability and effectiveness of machine unlearning. By leveraging JS and χ^2 divergences, our approach addresses the overshoot/undershoot problem inherent in existing methods, leading to more reliable and controlled unlearning. Our extensive experiments demonstrate that different divergence choices significantly impact forgetting accuracy, retention performance, and model stability. Notably, JS divergence offers a more stable unlearning process. These findings suggest that carefully selecting divergence metrics can substantially improve the trade-off between forgetting and preserving essential model knowledge. Future work could explore robustness evaluation and privacy implications of these divergences, particularly their effectiveness against membership inference attacks (MIA).

A Related Works

Two primary frameworks have emerged to address the challenge of unlearning: exact unlearning (?) and approximate unlearning (?). Exact unlearning requires retraining the model from scratch using only the remaining data, but this approach is computationally expensive and impractical for large-scale models (?). In contrast, approximate unlearning modifies the trained model to mimic the outcome of retraining on the remaining dataset. The key challenge in approximate unlearning is to ensure that the modified model is indistinguishable from a retrained one, often necessitating theoretical guarantees on the quality of the approximation (?).

Although much of the unlearning research has focused on convex models (?), the non-convexity of deep neural networks complicates the process. As a result, effective unlearning remains a challenge, with heuristics often producing varying results across different benchmarks, making it difficult to ensure consistent reliability (?).

(?) highlight a significant challenge in fine-tuning-based unlearning methods, known as the missing targets problem. When unlearning a data point $x \in$ forget set, these methods typically apply gradient ascent on x and gradient descent on the retain set to preserve model performance. However, gradient ascent can cause the loss on x to grow indefinitely if unchecked. The desired outcome is to stop when the model's loss on x matches the counterfactual loss (i.e., the loss of a model trained only on the retain set). This presents two main issues: (a) the target loss is unknown, and (b) the optimal stopping point may vary for different points in the forget set. As a result, unlearning algorithms often "undershoot" or "overshoot" the target loss (?).

This problem is further analyzed in the work of (?), which uses data modeling to address these challenges. Our research seeks to extend SCRUB to overcome this issue by introducing a loss function that is naturally robust to overshooting and undershooting by employing various f-divergences.

While f-divergences have been effective loss functions in various machine learning tasks (????), they have been primarily used for validating machine unlearning processes. For example, Jensen-Shannon (JS) divergence has been applied in the context of unlearning to validate the removal of data from models (?), (?), (?). Furthermore, there has been some exploration of using f-divergences specifically for unlearning large language models (LLMs) (?).

B Motivations for JS divergence and χ^2-divergence

In this section, we study some motivations behind choosing JS divergence and χ^2 divergence. These information measures offer several advantages over KL divergence, particularly in applications involving generative modeling and robust regularization.

JS divergence is widely used as a loss function in Generative Adversarial Networks (GANs) due to its symmetric and bounded nature, which provides a stable measure of similarity between distributions (?). Unlike KL divergence, which can diverge to infinity when the two distributions have disjoint supports, JS divergence remains finite and well-behaved, making it particularly effective for comparing empirical distributions (?). This property is especially beneficial in our context, as it helps mitigate overshoot and undershoot problems, particularly in scenarios where exact loss values for removed data points are unavailable.

On the other hand, χ^2 divergence emphasizes large discrepancies due to its squared difference term, making it particularly useful in outlier detection and robust learning frameworks (?). Regularizing with χ^2 divergence can also help prevent models from becoming overly biased toward majority classes by strongly penalizing large probability gaps (?). This property makes it particularly effective in imbalanced learning scenarios, where standard loss functions may fail to capture significant disparities between class distributions.

Thus, by leveraging JS divergence for stable probability comparisons and χ^2 divergence for strong regularization and outlier sensitivity, we can achieve a more robust and balanced learning framework compared to using KL divergence alone.

Building on this, we modify our loss functions and introduce f-SCRUB, where we select different f-divergences for the retain set and the forget set. Each divergence term, $d(x_r; w^u)$ and $d(x_f; w^u)$, can be chosen from JS, KL, or χ^2 divergences (?).

C Scenarios Motivations

The motivation behind these scenarios is twofold. First, as the number of forgotten samples increases, the impact on model performance in the retained set becomes more pronounced. Second, as more classes are involved, the complexity of the forgetting process increases, making the problem progressively more difficult.

An additional challenge arises in the evaluation phase: it becomes difficult to determine whether degraded performance on the forgetting set is due to successful forgetting or simply because the model is encountering previously unseen data. This ambiguity poses a fundamental challenge in measuring the effectiveness of forgetting strategies.

D Simulation details

We consider two versions of the model. The first, which we call the vanilla model, was trained on CIFAR-100 for 30 epochs and then fine-tuned on CIFAR-10 for another 30 epochs, achieving an accuracy of 0.84. We refer to this as the vanilla original model. Notably, this model does not operate at full capacity. Since we believe that the unlearning frameworks should be independent of the original model's training procedure, we also evaluate a full-capacity original model, which is a Torchvision pre-trained model with a precision of 0.96.

For the unlearning process, we apply two different policies. In the first, we perform two epochs of maximization, each followed by a minimization step, with an additional minimization step at the end. In the second, we extend the process to five maximization steps, each followed by a minimization step, concluding with five final minimization steps. Our simulations run on a single NVIDIA RTX 4090 GPU. We use the PyTorch library for our experiments. To ensure simplicity and fair comparisons, we fix the retraining batch size at 64 and the forgetting batch size at 32. The remaining parameters are the same as those used in SCRUB.

E Overshoot/ Undershoot Discussion

Another key aspect we aim to highlight is the impact of transitioning from a vanilla model to a maximum-capacity model on the absolute values of loss functions. In the vanilla model, uncertainty arises from the model's inherent lack of confidence, introducing variance in the loss function values. However, in more challenging unlearning scenarios, this uncertainty can significantly influence the loss function. Even in simpler cases, such as removing 100 data points from six classes, changing the model does not affect the variance of the loss function but does alter its bias. While KL divergence is highly sensitive to individual data points (see figure 2), a similar effect can also be observed with JS divergence (see figure 3).

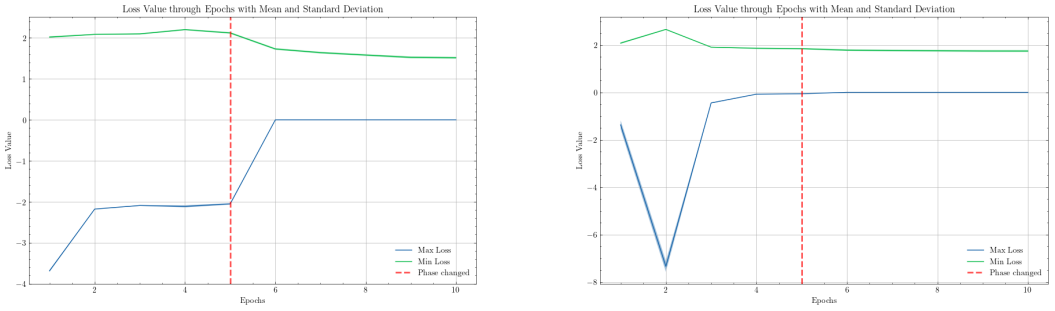


Figure 2: Comparing the effect of using a vanilla model (right) versus a maximum-capacity model (left) for KL-KL.

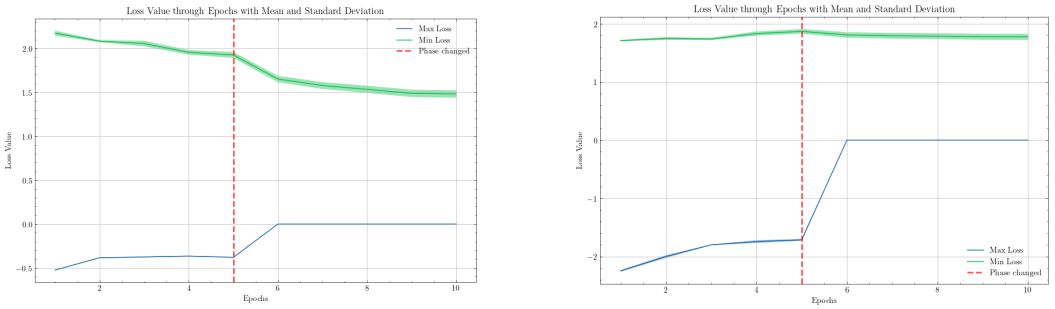


Figure 3: Comparing the effect of using a vanilla model(right) versus a maximum-capacity model (left) for KL-JS.

Additionally, we examine the impact of the number of training epochs on the loss function values at each step. In more complex and challenging scenarios, KL divergence demonstrates high sensitivity to individual data points, resulting in significant fluctuations when the algorithm is run for different numbers of epochs. In contrast, JS divergence, due to its bounded nature, offers greater stability and is less affected by such variations. As expected, increasing the number of training epochs shows that the loss values remain more consistent and robust when using the JS loss function (see figure 4), whereas KL divergence exhibits greater variability (see figure 5).

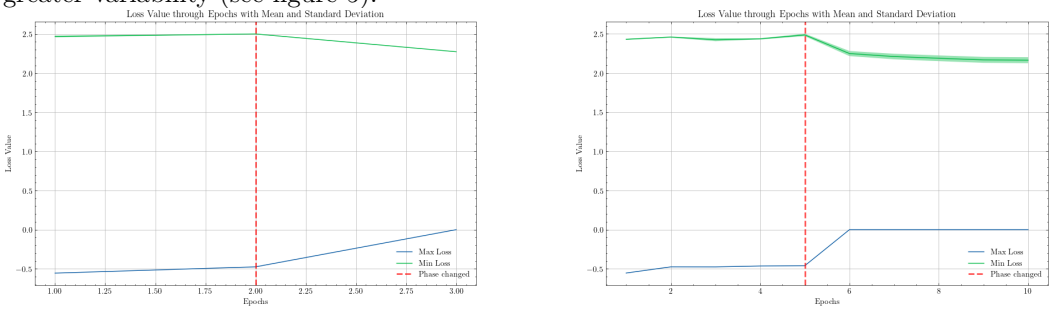


Figure 4: This is the Extended scenario for KL-JS .

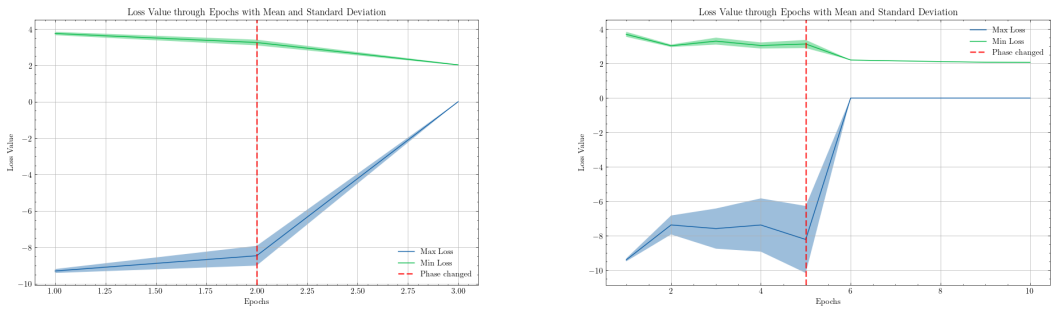


Figure 5: This is the Extended scenario for KL-KL

F Combination of Losses

You can observe all nine combinations of maximization-minimization losses. As seen in the results, despite JS performing well as a maximization loss (as mentioned in Section 5), it fails to recover the model when used as a minimization loss, as shown in Tab. 8 and Tab. 9. This failure is due to the slow convergence of JS in minimization, making it unsuitable for this role.

Additionally, χ^2 achieves the fastest recovery among all losses when used as a minimization loss, particularly in the Complete scenario, where the entire class is forgotten.

Table 4: Results for various forgetting scenarios with combinations of KL $\times$ {KL, JS, χ^2}, training for five intermittent maximization-minimization steps followed by five minimization step

Scenario	Loss	Vanilla			Full capacity		
		Forget Error	Retain Error	Test Error	Forget Error	Retain Error	Test Error
Complete	KL-KL	100.00 ± 0.00	80.16 ± 7.88	82.28 ± 7.10	100.00 ± 0.00	35.43 ± 4.89	43.11 ± 3.99
	KL-JS	100.00 ± 0.00	81.16 ± 2.73	83.01 ± 2.47	100.00 ± 0.00	50.50 ± 0.75	56.34 ± 0.69
	KL-X2	100.00 ± 0.00	77.47 ± 0.35	79.90 ± 0.40	100.00 ± 0.00	60.42 ± 6.74	64.80 ± 5.54
Light	KL-KL	25.33 ± 7.41	22.73 ± 0.93	26.92 ± 0.74	26.67 ± 4.92	8.93 ± 0.06	14.88 ± 0.05
	KL-JS	25.33 ± 3.30	22.26 ± 0.36	26.43 ± 0.22	26.33 ± 2.25	8.80 ± 0.09	14.74 ± 0.13
	KL-X2	93.67 ± 6.34	61.55 ± 6.08	62.49 ± 5.73	23.67 ± 5.25	52.48 ± 7.42	53.43 ± 7.31
Moderate	KL-KL	100.00 ± 0.00	72.51 ± 4.67	73.05 ± 4.55	33.80 ± 6.88	14.14 ± 1.54	18.64 ± 1.47
	KL-JS	99.73 ± 0.19	60.09 ± 2.11	61.79 ± 1.90	39.53 ± 4.40	17.67 ± 0.33	21.68 ± 0.14
	KL-X2	100.00 ± 0.00	79.32 ± 0.13	79.53 ± 0.20	23.20 ± 2.14	58.95 ± 2.19	59.42 ± 2.35
Light-Dual	KL-KL	30.67 ± 3.42	23.73 ± 0.21	27.83 ± 0.12	21.33 ± 3.32	10.25 ± 0.90	15.76 ± 0.42
	KL-JS	29.50 ± 2.55	23.36 ± 0.10	27.50 ± 0.06	21.17 ± 0.82	9.41 ± 0.11	15.24 ± 0.06
	KL-X2	100.00 ± 0.00	79.51 ± 0.11	79.68 ± 0.07	69.50 ± 19.63	62.66 ± 5.03	63.32 ± 4.88
Dual	KL-KL	100.00 ± 0.00	79.05 ± 0.15	79.59 ± 0.21	35.40 ± 1.68	22.96 ± 0.27	26.27 ± 0.09
	KL-JS	100.00 ± 0.00	79.01 ± 0.12	79.56 ± 0.09	49.83 ± 1.49	31.97 ± 0.40	34.23 ± 0.40
	KL-X2	100.00 ± 0.00	79.40 ± 0.17	79.99 ± 0.16	95.37 ± 6.55	66.50 ± 3.81	67.51 ± 4.13
Light-Broad	KL-KL	83.33 ± 8.68	66.17 ± 9.32	67.05 ± 8.67	39.00 ± 0.99	24.07 ± 1.04	27.36 ± 1.01
	KL-JS	63.47 ± 4.74	56.33 ± 3.69	57.84 ± 3.25	40.60 ± 1.50	23.70 ± 0.30	26.89 ± 0.18
	KL-X2	85.73 ± 5.20	79.75 ± 0.07	79.90 ± 0.12	61.53 ± 8.42	55.83 ± 8.48	56.49 ± 8.38
Broad	KL-KL	95.01 ± 7.02	73.05 ± 4.30	74.36 ± 3.71	91.96 ± 4.83	57.03 ± 1.95	59.88 ± 1.82
	KL-JS	100.00 ± 0.00	78.86 ± 0.11	80.11 ± 0.07	91.19 ± 2.78	56.87 ± 1.06	59.55 ± 1.08
	KL-X2	100.00 ± 0.00	79.04 ± 0.05	80.39 ± 0.10	97.15 ± 0.20	59.39 ± 0.44	62.28 ± 0.45
Light-Extended	KL-KL	75.28 ± 8.37	69.35 ± 7.49	69.83 ± 7.11	38.72 ± 0.93	31.38 ± 3.63	33.57 ± 3.51
	KL-JS	72.83 ± 2.72	72.55 ± 4.75	72.88 ± 4.47	36.22 ± 1.33	28.92 ± 3.04	31.51 ± 2.66
	KL-X2	73.56 ± 8.14	79.72 ± 0.06	79.67 ± 0.07	62.72 ± 3.48	56.46 ± 6.08	56.96 ± 5.98
Extended	KL-KL	88.73 ± 7.97	82.37 ± 6.34	82.81 ± 5.48	69.90 ± 7.66	53.52 ± 3.98	55.38 ± 4.27
	KL-JS	90.72 ± 2.33	78.00 ± 0.58	79.04 ± 0.47	85.78 ± 4.79	61.28 ± 2.71	63.76 ± 2.82
	KL-X2	90.93 ± 6.67	78.61 ± 0.33	79.71 ± 0.74	83.67 ± 1.02	65.53 ± 3.86	67.32 ± 3.40

Table 5: Results for various forgetting scenarios with combinations of X2 × {KL, JS, χ^2}, training for five intermittent maximization-minimization steps followed by five minimization step

Scenario	Loss	Vanilla			Full capacity		
		Forget Error	Retain Error	Test Error	Forget Error	Retain Error	Test Error
Complete	X2-KL	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00
	X2-JS	100.00 ± 0.00	62.53 ± 3.83	66.77 ± 3.33	100.00 ± 0.00	27.09 ± 1.20	36.54 ± 1.06
	X2-X2	100.00 ± 0.00	62.23 ± 8.57	66.57 ± 7.58	100.00 ± 0.00	26.17 ± 3.16	35.24 ± 2.17
Light	X2-KL	46.33 ± 6.13	15.98 ± 0.21	22.23 ± 0.55	26.67 ± 6.18	5.37 ± 0.02	13.00 ± 0.13
	X2-JS	44.00 ± 0.71	15.94 ± 0.13	22.30 ± 0.09	23.33 ± 3.52	5.45 ± 0.08	12.83 ± 0.06
	X2-X2	41.67 ± 3.86	16.10 ± 0.55	21.85 ± 0.74	33.00 ± 6.48	12.28 ± 2.90	17.71 ± 2.54
Moderate	X2-KL	54.13 ± 5.73	34.06 ± 5.52	37.56 ± 4.73	38.20 ± 4.85	12.04 ± 1.81	17.72 ± 1.76
	X2-JS	54.00 ± 0.96	23.81 ± 1.38	28.35 ± 1.27	38.13 ± 0.62	11.91 ± 0.50	17.45 ± 0.53
	X2-X2	90.80 ± 7.95	59.75 ± 12.12	60.75 ± 11.42	46.00 ± 2.41	31.97 ± 0.80	33.66 ± 0.07
Light-Dual	X2-KL	30.67 ± 1.65	16.42 ± 0.68	22.28 ± 0.16	17.33 ± 0.62	5.57 ± 0.15	12.94 ± 0.13
	X2-JS	28.83 ± 0.85	16.03 ± 0.23	22.36 ± 0.06	19.50 ± 1.95	5.72 ± 0.07	12.64 ± 0.10
	X2-X2	38.50 ± 2.55	32.66 ± 1.97	35.31 ± 2.16	33.17 ± 4.11	22.08 ± 0.64	25.35 ± 0.48
Dual	X2-KL	63.33 ± 11.34	42.98 ± 3.61	45.10 ± 3.45	33.97 ± 3.41	19.74 ± 1.10	23.48 ± 0.91
	X2-JS	46.47 ± 0.65	35.37 ± 0.32	38.04 ± 0.21	35.30 ± 0.90	20.81 ± 0.22	24.37 ± 0.31
	X2-X2	85.43 ± 3.16	52.51 ± 2.74	54.22 ± 2.72	43.50 ± 2.06	33.92 ± 1.33	35.60 ± 0.40
Light-Broad	X2-KL	39.53 ± 4.03	30.48 ± 1.76	34.44 ± 1.42	27.93 ± 1.89	15.17 ± 0.34	19.61 ± 0.20
	X2-JS	36.87 ± 1.35	23.50 ± 1.77	27.78 ± 1.84	26.20 ± 0.98	15.37 ± 0.31	20.10 ± 0.17
	X2-X2	63.60 ± 8.67	61.97 ± 11.29	62.95 ± 10.62	39.80 ± 2.57	32.96 ± 0.97	34.67 ± 0.69
Broad	X2-KL	93.25 ± 9.54	89.78 ± 0.64	90.00 ± 0.00	43.11 ± 0.52	32.21 ± 1.23	34.41 ± 1.42
	X2-JS	85.09 ± 4.92	58.94 ± 5.41	61.41 ± 5.27	42.79 ± 1.22	31.08 ± 0.91	33.24 ± 0.63
	X2-X2	85.89 ± 9.87	61.38 ± 10.69	63.32 ± 10.43	52.47 ± 4.26	39.31 ± 3.36	40.99 ± 3.15
Light-Extended	X2-KL	35.89 ± 1.13	30.55 ± 1.23	33.67 ± 1.53	26.44 ± 2.11	18.38 ± 1.79	22.18 ± 1.42
	X2-JS	31.56 ± 1.84	25.98 ± 2.19	29.83 ± 1.96	25.17 ± 1.80	17.33 ± 0.47	21.60 ± 0.47
	X2-X2	66.72 ± 5.76	61.25 ± 11.68	62.04 ± 11.30	38.22 ± 0.97	34.15 ± 1.59	35.52 ± 1.51
Extended	X2-KL	88.53 ± 8.11	90.12 ± 0.66	90.00 ± 0.00	41.58 ± 3.49	36.53 ± 1.98	38.38 ± 1.84
	X2-JS	79.46 ± 5.20	63.20 ± 5.01	64.82 ± 4.90	40.32 ± 0.75	34.71 ± 0.40	36.69 ± 0.25
	X2-X2	78.97 ± 9.45	66.78 ± 8.25	67.93 ± 8.16	48.22 ± 2.70	40.94 ± 1.88	42.45 ± 1.67

Table 6: Results for various forgetting scenarios with combinations of KL × {KL, JS, χ^2}, training for two intermittent maximization-minimization steps followed by one minimization step

Scenario	Loss	Vanilla			Full capacity		
		Forget Error	Retain Error	Test Error	Forget Error	Retain Error	Test Error
Complete	KL-KL	100.00 ± 0.00	77.55 ± 1.19	79.94 ± 1.18	100.00 ± 0.00	45.20 ± 5.46	51.12 ± 4.71
	KL-JS	100.00 ± 0.00	78.21 ± 0.62	80.59 ± 0.47	100.00 ± 0.00	62.91 ± 2.45	66.92 ± 2.19
	KL-X2	100.00 ± 0.00	72.05 ± 3.48	75.04 ± 3.12	100.00 ± 0.00	45.95 ± 6.20	51.86 ± 5.49
Light	KL-KL	31.67 ± 18.93	43.00 ± 6.90	44.04 ± 6.57	39.67 ± 7.36	31.89 ± 4.13	33.21 ± 4.09
	KL-JS	49.33 ± 10.44	41.91 ± 1.79	42.91 ± 1.68	38.33 ± 9.88	36.25 ± 2.95	37.56 ± 2.84
	KL-X2	51.33 ± 34.50	53.07 ± 9.10	54.26 ± 8.87	40.67 ± 28.43	53.63 ± 6.91	53.76 ± 6.50
Moderate	KL-KL	100.00 ± 0.00	68.01 ± 2.65	68.62 ± 2.53	48.53 ± 12.28	33.64 ± 3.20	34.80 ± 3.42
	KL-JS	96.47 ± 2.43	58.93 ± 2.22	59.79 ± 2.31	61.33 ± 3.85	37.43 ± 0.44	38.72 ± 0.36
	KL-X2	91.93 ± 11.41	62.94 ± 3.06	64.00 ± 3.24	50.07 ± 29.95	51.18 ± 2.08	51.60 ± 2.39
Light-Dual	KL-KL	42.67 ± 3.47	40.27 ± 3.66	41.42 ± 3.31	45.67 ± 14.49	32.11 ± 1.31	33.50 ± 1.20
	KL-JS	56.33 ± 6.41	43.07 ± 3.52	44.48 ± 3.35	40.67 ± 9.42	27.14 ± 0.94	28.80 ± 0.85
	KL-X2	76.50 ± 25.49	63.89 ± 2.64	63.89 ± 2.46	53.50 ± 10.59	40.75 ± 1.69	41.46 ± 1.53
Dual	KL-KL	92.67 ± 10.37	70.34 ± 2.30	71.01 ± 2.16	53.37 ± 3.63	43.20 ± 5.44	44.13 ± 5.22
	KL-JS	98.47 ± 0.47	63.95 ± 1.05	65.37 ± 1.09	59.13 ± 4.05	46.13 ± 1.88	47.03 ± 1.78
	KL-X2	99.67 ± 0.25	64.60 ± 1.23	65.92 ± 1.29	53.67 ± 18.54	47.39 ± 5.92	47.58 ± 5.75
Light-Broad	KL-KL	83.47 ± 17.46	72.70 ± 7.64	72.96 ± 7.43	48.73 ± 7.27	41.61 ± 4.02	42.72 ± 3.60
	KL-JS	63.60 ± 7.33	53.38 ± 1.67	54.39 ± 1.61	57.20 ± 5.26	45.47 ± 4.99	46.29 ± 4.94
	KL-X2	84.67 ± 6.37	62.55 ± 2.70	63.27 ± 2.34	56.40 ± 7.07	45.07 ± 1.12	45.52 ± 1.21
Broad	KL-KL	91.97 ± 6.39	76.55 ± 0.72	77.73 ± 0.89	77.41 ± 6.55	55.09 ± 2.11	57.09 ± 2.25
	KL-JS	99.71 ± 0.03	74.07 ± 1.65	75.58 ± 1.57	96.64 ± 0.57	64.67 ± 0.83	67.00 ± 0.89
	KL-X2	98.57 ± 0.21	65.06 ± 1.75	67.45 ± 1.82	70.32 ± 4.38	48.58 ± 4.86	50.71 ± 4.76
Light-Extended	KL-KL	67.39 ± 6.26	71.70 ± 3.61	71.94 ± 3.42	56.33 ± 4.40	46.86 ± 2.39	47.48 ± 2.36
	KL-JS	60.83 ± 2.89	59.07 ± 1.42	59.71 ± 1.52	52.11 ± 4.27	42.78 ± 3.38	43.78 ± 3.48
	KL-X2	65.61 ± 8.09	63.10 ± 1.55	63.39 ± 1.86	67.06 ± 5.55	52.29 ± 3.94	53.13 ± 4.37
Extended	KL-KL	85.48 ± 4.57	78.94 ± 2.42	79.28 ± 1.97	56.22 ± 4.86	51.84 ± 6.76	52.37 ± 6.73
	KL-JS	84.60 ± 0.74	83.78 ± 1.81	83.76 ± 1.62	82.47 ± 0.44	63.35 ± 0.64	64.97 ± 0.53
	KL-X2	83.17 ± 0.28	66.09 ± 3.80	67.56 ± 3.28	50.01 ± 2.81	38.88 ± 2.09	40.56 ± 2.00

Table 7: Results for various forgetting scenarios with combinations of X2 × {KL, JS, χ^2}, training for two intermittent maximization-minimization steps followed by one minimization step

Scenario	Loss	Vanilla			Full capacity		
		Forget Error	Retain Error	Test Error	Forget Error	Retain Error	Test Error
Complete	X2-KL	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	86.33 ± 3.62	87.75 ± 3.18
	X2-JS	100.00 ± 0.00	63.07 ± 3.46	66.47 ± 3.00	100.00 ± 0.00	51.08 ± 2.18	56.02 ± 1.92
	X2-X2	100.00 ± 0.00	58.40 ± 6.60	62.45 ± 5.57	100.00 ± 0.00	39.70 ± 1.70	46.70 ± 0.91
Light	X2-KL	44.33 ± 9.98	36.20 ± 3.36	37.89 ± 3.11	38.67 ± 9.57	34.39 ± 3.69	35.67 ± 3.60
	X2-JS	61.33 ± 4.55	33.61 ± 1.43	35.74 ± 1.19	43.67 ± 5.98	27.83 ± 2.06	29.66 ± 2.01
	X2-X2	57.00 ± 15.58	36.07 ± 2.67	37.77 ± 2.75	53.33 ± 23.61	41.59 ± 5.38	42.72 ± 5.19
Moderate	X2-KL	58.67 ± 12.27	52.55 ± 0.67	53.54 ± 0.98	62.40 ± 20.81	35.67 ± 2.15	36.65 ± 1.93
	X2-JS	69.93 ± 6.06	40.06 ± 2.33	41.54 ± 2.25	44.67 ± 4.61	34.09 ± 0.87	35.35 ± 1.01
	X2-X2	80.00 ± 25.64	61.79 ± 3.19	62.35 ± 3.82	75.33 ± 14.63	39.86 ± 3.04	41.10 ± 3.29
Light-Dual	X2-KL	46.00 ± 8.38	34.19 ± 0.54	35.65 ± 0.43	31.17 ± 2.25	26.47 ± 2.56	28.31 ± 2.00
	X2-JS	52.67 ± 2.16	34.59 ± 0.83	36.07 ± 0.87	35.67 ± 5.29	29.32 ± 0.75	31.17 ± 0.95
	X2-X2	57.83 ± 2.32	48.63 ± 2.84	49.13 ± 2.94	54.67 ± 5.86	40.80 ± 2.72	41.57 ± 2.55
Dual	X2-KL	74.10 ± 18.43	56.80 ± 5.25	57.86 ± 5.70	50.53 ± 10.09	35.80 ± 1.45	37.19 ± 1.13
	X2-JS	64.03 ± 3.91	50.41 ± 1.72	51.74 ± 1.86	60.07 ± 3.57	39.35 ± 1.05	41.12 ± 1.10
	X2-X2	73.53 ± 14.56	55.17 ± 0.34	56.17 ± 0.45	51.90 ± 5.02	37.52 ± 1.76	38.17 ± 1.26
Light-Broad	X2-KL	57.87 ± 10.40	46.19 ± 1.23	46.89 ± 1.62	42.53 ± 1.75	35.37 ± 3.15	36.42 ± 3.25
	X2-JS	46.07 ± 2.44	39.91 ± 1.61	41.33 ± 1.61	42.87 ± 1.02	34.19 ± 1.25	35.37 ± 1.12
	X2-X2	68.13 ± 8.87	58.01 ± 1.77	58.35 ± 1.87	48.73 ± 6.33	38.72 ± 2.97	39.97 ± 2.74
Broad	X2-KL	91.56 ± 10.88	75.24 ± 9.29	76.00 ± 9.58	61.04 ± 10.69	48.67 ± 3.77	49.74 ± 3.71
	X2-JS	90.08 ± 2.97	65.12 ± 5.44	67.07 ± 5.09	59.15 ± 2.09	44.54 ± 1.15	45.47 ± 1.03
	X2-X2	84.96 ± 3.63	64.25 ± 5.55	65.74 ± 5.52	57.33 ± 2.20	41.69 ± 4.69	43.40 ± 4.33
Light-Extended	X2-KL	56.22 ± 2.15	51.74 ± 3.02	52.35 ± 2.45	47.83 ± 8.45	38.26 ± 4.47	39.32 ± 4.37
	X2-JS	50.33 ± 2.29	44.41 ± 0.35	45.79 ± 0.55	47.44 ± 3.89	38.02 ± 1.53	39.33 ± 1.49
	X2-X2	59.17 ± 7.27	54.56 ± 2.18	55.06 ± 2.50	43.78 ± 10.07	37.88 ± 5.24	38.78 ± 5.06
Extended	X2-KL	88.53 ± 8.11	90.12 ± 0.66	90.00 ± 0.00	52.09 ± 4.50	48.34 ± 2.06	49.22 ± 1.34
	X2-JS	77.87 ± 2.54	64.95 ± 1.88	66.10 ± 1.71	52.59 ± 0.78	48.79 ± 2.12	49.33 ± 1.76
	X2-X2	69.67 ± 8.76	59.50 ± 1.08	60.52 ± 0.59	48.37 ± 2.61	42.39 ± 1.33	43.53 ± 1.12

Table 8: Results for various forgetting scenarios with combinations of JS × {KL, JS, χ^2}, training for two intermittent maximization-minimization steps followed by one minimization step

Scenario	Loss	Vanilla			Full capacity		
		Forget Error	Retain Error	Test Error	Forget Error	Retain Error	Test Error
Complete	JS-KL	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00
	JS-JS	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	83.52 ± 0.61	85.11 ± 0.48
	JS-X2	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	62.96 ± 1.38	66.90 ± 1.29
Light	JS-KL	100.00 ± 0.00	86.97 ± 4.24	86.99 ± 4.25	22.33 ± 12.50	49.96 ± 1.50	49.80 ± 1.94
	JS-JS	100.00 ± 0.00	87.43 ± 1.80	87.46 ± 1.80	47.33 ± 18.66	51.24 ± 1.16	51.41 ± 1.27
	JS-X2	100.00 ± 0.00	87.25 ± 3.86	87.26 ± 3.87	34.00 ± 25.92	70.34 ± 0.21	70.39 ± 0.22
Moderate	JS-KL	100.00 ± 0.00	89.87 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	67.40 ± 3.82	68.05 ± 3.76
	JS-JS	100.00 ± 0.00	89.87 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	63.89 ± 3.67	64.42 ± 3.54
	JS-X2	100.00 ± 0.00	89.87 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	68.65 ± 7.96	69.03 ± 7.67
Light-Dual	JS-KL	100.00 ± 0.00	89.95 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	54.14 ± 1.56	54.64 ± 1.42
	JS-JS	100.00 ± 0.00	87.37 ± 1.83	87.42 ± 1.82	100.00 ± 0.00	55.16 ± 1.00	55.75 ± 0.87
	JS-X2	100.00 ± 0.00	89.95 ± 0.00	90.00 ± 0.00	78.50 ± 15.51	67.78 ± 2.55	68.00 ± 2.87
Dual	JS-KL	87.93 ± 17.06	78.98 ± 7.63	79.46 ± 7.49	100.00 ± 0.00	70.08 ± 2.01	71.14 ± 1.90
	JS-JS	100.00 ± 0.00	89.74 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	65.94 ± 2.08	67.28 ± 2.09
	JS-X2	100.00 ± 0.00	89.74 ± 0.00	90.00 ± 0.00	84.83 ± 15.23	63.96 ± 2.33	64.64 ± 1.92
Light-Broad	JS-KL	87.87 ± 8.62	90.03 ± 0.11	90.00 ± 0.00	88.33 ± 8.30	64.28 ± 3.17	64.58 ± 3.17
	JS-JS	87.87 ± 4.31	90.03 ± 0.05	90.00 ± 0.00	100.00 ± 0.00	69.33 ± 1.80	70.02 ± 1.81
	JS-X2	87.87 ± 8.62	90.03 ± 0.11	90.00 ± 0.00	98.60 ± 1.34	68.04 ± 4.38	68.67 ± 4.18
Broad	JS-KL	92.47 ± 7.54	83.75 ± 4.54	84.16 ± 4.43	98.29 ± 2.41	81.43 ± 5.62	82.52 ± 5.33
	JS-JS	93.44 ± 4.64	89.77 ± 0.31	90.00 ± 0.00	100.00 ± 0.00	82.99 ± 1.30	84.01 ± 1.18
	JS-X2	93.44 ± 9.28	89.77 ± 0.62	90.00 ± 0.00	96.93 ± 4.34	65.81 ± 3.63	67.79 ± 3.58
Light-Extended	JS-KL	83.67 ± 0.59	88.36 ± 2.47	88.24 ± 2.49	76.89 ± 6.14	66.25 ± 3.35	66.41 ± 3.25
	JS-JS	83.67 ± 0.30	90.10 ± 0.00	90.00 ± 0.00	83.78 ± 6.72	68.84 ± 2.02	68.92 ± 2.14
	JS-X2	83.67 ± 0.59	90.10 ± 0.01	90.00 ± 0.00	89.06 ± 4.42	65.90 ± 2.81	66.25 ± 2.64
Extended	JS-KL	88.53 ± 8.11	90.12 ± 0.66	90.00 ± 0.00	100.00 ± 0.00	89.19 ± 0.00	90.00 ± 0.00
	JS-JS	89.14 ± 3.84	90.07 ± 0.31	90.00 ± 0.00	100.00 ± 0.00	89.19 ± 0.00	90.00 ± 0.00
	JS-X2	89.14 ± 7.68	90.07 ± 0.62	90.00 ± 0.00	87.23 ± 3.62	61.72 ± 4.15	63.73 ± 4.13

Table 9: Results for various forgetting scenarios with combinations of JS × {KL, JS, χ^2}, training for five intermittent maximization-minimization steps followed by five minimization step

Scenario	Loss	Vanilla			Full capacity		
		Forget Error	Retain Error	Test Error	Forget Error	Retain Error	Test Error
Complete	JS-KL	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00
	JS-JS	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00
	JS-X2	100.00 ± 0.00	88.89 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	78.91 ± 0.03	81.06 ± 0.10
Light	JS-KL	100.00 ± 0.00	89.97 ± 0.00	90.00 ± 0.00	69.67 ± 42.90	41.49 ± 3.65	42.65 ± 3.66
	JS-JS	100.00 ± 0.00	89.97 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	44.16 ± 0.05	45.32 ± 0.11
	JS-X2	100.00 ± 0.00	89.97 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	81.51 ± 0.86	81.64 ± 0.88
Moderate	JS-KL	100.00 ± 0.00	89.87 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	55.69 ± 4.23	56.99 ± 4.38
	JS-JS	100.00 ± 0.00	89.87 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	52.38 ± 0.07	53.68 ± 0.08
	JS-X2	100.00 ± 0.00	89.87 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	78.02 ± 3.92	78.43 ± 3.54
Light-Dual	JS-KL	100.00 ± 0.00	89.95 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	44.44 ± 0.14	45.78 ± 0.14
	JS-JS	100.00 ± 0.00	89.95 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	44.29 ± 0.01	45.63 ± 0.14
	JS-X2	100.00 ± 0.00	89.95 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	80.90 ± 0.06	80.96 ± 0.15
Dual	JS-KL	91.70 ± 11.74	86.36 ± 4.78	86.56 ± 4.86	100.00 ± 0.00	74.10 ± 4.11	74.88 ± 3.93
	JS-JS	100.00 ± 0.00	89.74 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	58.74 ± 1.86	60.03 ± 1.76
	JS-X2	100.00 ± 0.00	89.74 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	80.43 ± 0.08	80.94 ± 0.05
Light-Broad	JS-KL	81.53 ± 0.90	90.11 ± 0.01	90.00 ± 0.00	100.00 ± 0.00	79.19 ± 8.04	79.49 ± 7.86
	JS-JS	81.53 ± 0.45	90.11 ± 0.01	90.00 ± 0.00	100.00 ± 0.00	69.14 ± 2.54	69.69 ± 2.40
	JS-X2	81.53 ± 0.90	90.11 ± 0.01	90.00 ± 0.00	100.00 ± 0.00	78.00 ± 3.82	78.24 ± 3.82
Broad	JS-KL	93.25 ± 9.54	89.78 ± 0.64	90.00 ± 0.00	100.00 ± 0.00	89.33 ± 0.00	90.00 ± 0.00
	JS-JS	93.44 ± 4.64	89.77 ± 0.31	90.00 ± 0.00	100.00 ± 0.00	89.33 ± 0.00	90.00 ± 0.00
	JS-X2	100.00 ± 0.00	89.33 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	79.68 ± 0.06	80.96 ± 0.14
Light-Extended	JS-KL	83.67 ± 0.59	90.10 ± 0.01	90.00 ± 0.00	100.00 ± 0.00	79.59 ± 2.91	79.87 ± 2.80
	JS-JS	83.67 ± 0.30	90.10 ± 0.00	90.00 ± 0.00	91.11 ± 3.28	79.28 ± 2.29	79.42 ± 2.19
	JS-X2	83.67 ± 0.59	90.10 ± 0.01	90.00 ± 0.00	90.50 ± 6.72	81.15 ± 6.99	81.27 ± 6.91
Extended	JS-KL	88.53 ± 8.11	90.12 ± 0.66	90.00 ± 0.00	100.00 ± 0.00	89.19 ± 0.00	90.00 ± 0.00
	JS-JS	94.71 ± 3.74	89.62 ± 0.30	90.00 ± 0.00	100.00 ± 0.00	89.19 ± 0.00	90.00 ± 0.00
	JS-X2	100.00 ± 0.00	89.19 ± 0.00	90.00 ± 0.00	100.00 ± 0.00	84.04 ± 3.68	85.16 ± 3.46