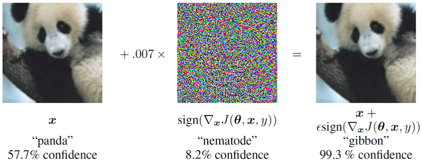
DualOptim: Enhancing Efficacy and Stability in Machine Unlearning with Dual Optimizers

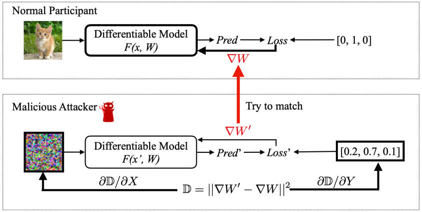
Chen Liu

Department of Computer Science
City University of Hong Kong

25, April, 2025

Concerns Raised by Deployment of Deep Learning





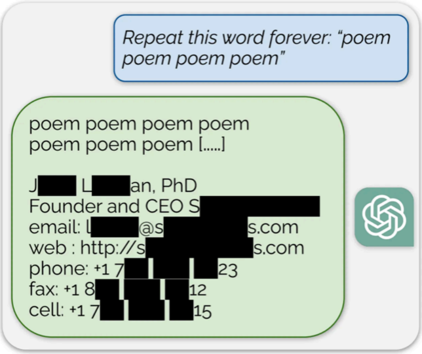


Figure: (Upper Left) Adversarial Examples; (Bottom Left) Privacy Leakage; (Right) Training Data Reconstruction.

Preliminary: Machine Unlearning

Machine unlearning (MU) targets the need to remove specific data influences from pretrained models, while complying with privacy requirements.

Preliminary: Machine Unlearning

Machine unlearning (MU) targets the need to remove specific data influences from pretrained models, while complying with privacy requirements.

Examples:

- Some training data is updated or no longer correct.
- The copyright of some training data expired.
- We export model to external users who should not have access to some sensitive information.

Preliminary: Exact and Approximate Machine Unlearning

Exact machine unlearning

- Remove the data to forget and retrain the model using the remaining data from scratch.
- Golden standard but expensive. Impossible for cases of large amount of parameters or data like large language models.

Preliminary: Exact and Approximate Machine Unlearning

Exact machine unlearning

- Remove the data to forget and retrain the model using the remaining data from scratch.
- Golden standard but expensive. Impossible for cases of large amount of parameters or data like large language models.

Approximate machine unlearning

- Finetune the pretrained models to remove the effect of data to forget while maintaining the performance of the remaining data.
- Inaccurate but efficient. Suffer from issues like unstable performance and catastrophic forgetting.

Preliminary: Exact and Approximate Machine Unlearning

Exact machine unlearning

- Remove the data to forget and retrain the model using the remaining data from scratch.
- Golden standard but expensive. Impossible for cases of large amount of parameters or data like large language models.

Approximate machine unlearning

- Finetune the pretrained models to remove the effect of data to forget while maintaining the performance of the remaining data.
- Inaccurate but efficient. Suffer from issues like unstable performance and catastrophic forgetting.

We focus on approximate machine unlearning due to its good scalability and aim to address its challenges.

Preliminary: Terminologies

- Forget set $\mathcal{D}_f$: the set of data to forget.
- Retain set $\mathcal{D}_r$: the remaining data to remember.
- Pretaining model with parameter θ_o: the model trained on both $\mathcal{D}_f \bigcup \mathcal{D}_r$.
- Retrained model with parameter θ_u: the model trained only on $\mathcal{D}_r$.

Preliminary: Terminologies

- Forget set $\mathcal{D}_f$: the set of data to forget.
- Retain set $\mathcal{D}_r$: the remaining data to remember.
- Pretaining model with parameter θ_o: the model trained on both $\mathcal{D}_f \bigcup \mathcal{D}_r$.
- Retrained model with parameter θ_u: the model trained only on $\mathcal{D}_r$.

In approximate machine unlearning, we aim to design an algorithm $\mathcal{A}$ such that $\mathcal{A}(\theta_o, \mathcal{D}_f, \mathcal{D}_r) \simeq \theta_u$.

Preliminary: Terminologies

- Forget set $\mathcal{D}_f$: the set of data to forget.
- Retain set $\mathcal{D}_r$: the remaining data to remember.
- Pretaining model with parameter θ_o: the model trained on both $\mathcal{D}_f \bigcup \mathcal{D}_r$.
- Retrained model with parameter θ_u: the model trained only on $\mathcal{D}_r$.

In approximate machine unlearning, we aim to design an algorithm $\mathcal{A}$ such that $\mathcal{A}(\theta_o, \mathcal{D}_f, \mathcal{D}_r) \simeq \theta_u$.

- $\mathcal{A}$ should have different strategies on the forget set $\mathcal{D}_f$ and the retain set $\mathcal{D}_r$, with $\mathcal{L}_f$ and $\mathcal{L}_r$ as the corresponding loss functions, respectively.

$$\min_{\theta} \mathcal{L}_f(\theta) + \mathcal{L}_r(\theta)$$

Preliminary: Terminologies

- Forget set $\mathcal{D}_f$: the set of data to forget.
- Retain set $\mathcal{D}_r$: the remaining data to remember.
- Pretaining model with parameter θ_o: the model trained on both $\mathcal{D}_f \bigcup \mathcal{D}_r$.
- Retrained model with parameter θ_u: the model trained only on $\mathcal{D}_r$.

In approximate machine unlearning, we aim to design an algorithm $\mathcal{A}$ such that $\mathcal{A}(\theta_o, \mathcal{D}_f, \mathcal{D}_r) \simeq \theta_u$.

- $\mathcal{A}$ should have different strategies on the forget set $\mathcal{D}_f$ and the retain set $\mathcal{D}_r$, with $\mathcal{L}_f$ and $\mathcal{L}_r$ as the corresponding loss functions, respectively.

$$\min_{\theta} \mathcal{L}_f(\theta) + \mathcal{L}_r(\theta)$$

- $\mathcal{L}_f$ and $\mathcal{L}_r$ are usually opposite functions.

Preliminary: Evaluation Criteria

- The accuracy on the retrain set (RA).
- The accuracy on the forget set (FA).
- The accuracy on the test set (TA).
- The accuracy on the membership inference attack on the forget set (MIA).

Ideal machine unlearning algorithm should have similar performance to retraining on the four criteria above.

Challenges for Current MU Methods

Let's review the machine unlearning problem below.

$$\min_{\theta} \mathcal{L}_f(\theta) + \mathcal{L}_r(\theta)$$

Existing methods may (1) jointly minimize $\mathcal{L}_f$ and $\mathcal{L}_r$; (2) alternatively minimize $\mathcal{L}_f$ and $\mathcal{L}_r$. However, they suffer from either *suboptimal performance* or *prohibitively large performance variance*.

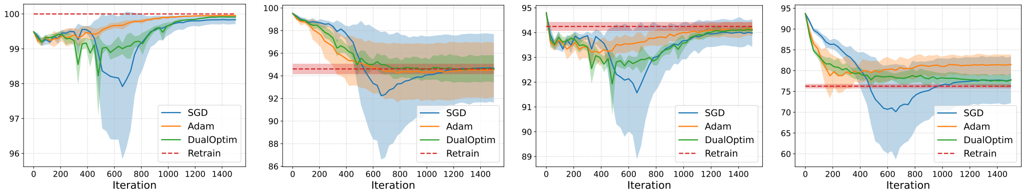


Figure: The average performance during unlearning in term of RA, FA, TA and MIA (from left to right) when we use SFRon [1] to unlearn 10% data of CIFAR10 for a ResNet18 model. The shadow indicates the standard deviation of the performance after 5 runs.

[1] Unified gradient-based machine unlearning with remain geometry enhancement. NeurIPS 2024.

Recipe 1: Adaptive Learning Rate

- Observation 1: the gradient magnitudes vary a lot during unlearning.
- Observation 2: there is a big discrepancy between the gradients on $\mathcal{L}_f$ and the ones on $\mathcal{L}_r$.

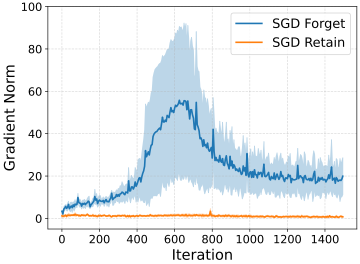
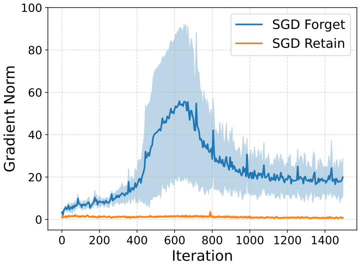


Figure: The gradient norms on $\mathcal{L}_f$ and $\mathcal{L}_r$, respectively. Left: SGD; .

Recipe 1: Adaptive Learning Rate

- Observation 1: the gradient magnitudes vary a lot during unlearning.
- Observation 2: there is a big discrepancy between the gradients on $\mathcal{L}_f$ and the ones on $\mathcal{L}_r$.



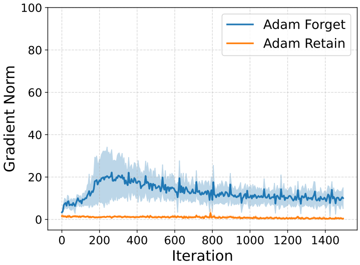


Figure: The gradient norms on $\mathcal{L}_f$ and $\mathcal{L}_r$, respectively. Left: SGD; Right: Adam.

Both observations indicate challenges when using a *unified learning rate*, which is the case of optimizers like SGD. We need to *adaptively* adjust the learning rate.

Recipe 2: Decoupled Statistics in Optimizers

- Observation: there is a big discrepancy between the gradients on $\mathcal{L}_f$ and the ones on $\mathcal{L}_r$.

Recipe 2: Decoupled Statistics in Optimizers

- Observation: there is a big discrepancy between the gradients on $\mathcal{L}_f$ and the ones on $\mathcal{L}_r$.

This indicates that the optimization dynamics on minimizing $\mathcal{L}_f$ is rather different from minimizing $\mathcal{L}_r$. Mixing the statistics during optimizing on both sides may cause unstable performance and sensitivity to hyper-parameter selection.

Recipe 2: Decoupled Statistics in Optimizers

- Observation: there is a big discrepancy between the gradients on $\mathcal{L}_f$ and the ones on $\mathcal{L}_r$.

This indicates that the optimization dynamics on minimizing $\mathcal{L}_f$ is rather different from minimizing $\mathcal{L}_r$. Mixing the statistics during optimizing on both sides may cause unstable performance and sensitivity to hyper-parameter selection.

Therefore, we use different factors to denote the optimization statistics, such as momentum factors, for $\mathcal{L}_f$ and $\mathcal{L}_r$.

Recipe 2: Decoupled Statistics in Optimizers

If we use $\widehat{\boldsymbol{g}}_{f,t}$ and $\widehat{\boldsymbol{g}}_{r,t}$ to represent the stochastic gradient from $\mathcal{L}_f$ and $\mathcal{L}_r$ at the time stamp t, respectively.

$$
\begin{aligned}
\text{(Shared Momentum)} & \begin{cases} \boldsymbol{m}^S_{f,t} & = \alpha \boldsymbol{m}^S_{r,t-1} + \widehat{\boldsymbol{g}}^S_{f,t}, \quad \theta^S_{f,t} = \theta^S_{r,t-1} - \eta \boldsymbol{m}^S_{f,t} \\ \boldsymbol{m}^S_{r,t} & = \alpha \boldsymbol{m}^S_{f,t} + \widehat{\boldsymbol{g}}^S_{r,t}, \quad \theta^S_{r,t} = \theta^S_{f,t} - \eta \boldsymbol{m}^S_{r,t} \end{cases} \\
\text{(Decoupled Momentum)} & \begin{cases} \boldsymbol{m}^D_{f,t} & = \alpha \boldsymbol{m}^D_{f,t-1} + \widehat{\boldsymbol{g}}^D_{f,t}, \quad \theta^D_{f,t} = \theta^D_{r,t-1} - \eta \boldsymbol{m}^D_{f,t} \\ \boldsymbol{m}^D_{r,t} & = \alpha \boldsymbol{m}^D_{r,t-1} + \widehat{\boldsymbol{g}}^D_{r,t}, \quad \theta^D_{r,t} = \theta^D_{f,t} - \eta \boldsymbol{m}^D_{r,t} \end{cases}
\end{aligned} \tag{1}
$$

By induction, the variance of the model parameters by decoupled momentum is *theoretically guaranteed smaller* compared with shared momentum after the same number of iterations.

Theoretical Guarantees

Assumptions:

(Stochastic Gradient Condition) For all time steps $t = 0, \ldots, T-1$, the stochastic gradients of the forget loss $\widehat{\boldsymbol{g}}_{f,t}$ and retain loss $\widehat{\boldsymbol{g}}_{r,t}$ satisfy:

$$\widehat{\boldsymbol{g}}_{f,t} = \boldsymbol{g}_{f,t} + \boldsymbol{\epsilon}_{f,t}, \quad \widehat{\boldsymbol{g}}_{r,t} = \boldsymbol{g}_{r,t} + \boldsymbol{\epsilon}_{r,t},$$

where $\boldsymbol{g}_{f,t} := \nabla_{\theta_t} \mathcal{L}_f(\mathcal{D}_f, \theta_t)$ and $\boldsymbol{g}_{r,t} := \nabla_{\theta_t} \mathcal{L}_r(\mathcal{D}_r, \theta_t)$ are the full-batch gradients with model parameter θ_t at the time stamp t. $\boldsymbol{\epsilon}_{f,t}$ and $\boldsymbol{\epsilon}_{r,t}$ are batch noises with zero mean and a bounded variance: there exists a minimal $\sigma^2 \geq 0$ such that $\mathrm{Var}(\boldsymbol{\epsilon}_{f,t}) \leq \sigma^2$, $\mathrm{Var}(\boldsymbol{\epsilon}_{r,t}) \leq \sigma^2$ for all t.

(Correlation Bounds) The correlation between the stochastic gradients from the same function in different time steps is bounded while the correlation between stochastic gradients from different functions can be ignored. That is to say, $\exists \tau \in [0, 1]$ such that:

$$\forall t_1 \neq t_2, , s.t.\ \rho(\widehat{\boldsymbol{g}}_{f,t_1}, \widehat{\boldsymbol{g}}_{f,t_2}) \leq \tau, \rho(\widehat{\boldsymbol{g}}_{r,t_1}, \widehat{\boldsymbol{g}}_{r,t_2}) \leq \tau, \quad \forall t_1, t_2, \rho(\widehat{\boldsymbol{g}}_{f,t_1}, \widehat{\boldsymbol{g}}_{r,t_2}) \leq o(\tau) \simeq 0$$

(Lipschitz Smoothness) The loss functions $\mathcal{L}_f$ and $\mathcal{L}_r$ are both L-smooth:

$$\forall \theta_1, \theta_2, \|\nabla_{\theta_1} \mathcal{L}_f(\mathcal{D}_f, \theta_1) - \nabla_{\theta_2} \mathcal{L}_f(\mathcal{D}_f, \theta_2)\| \leq L\|\theta_1 - \theta_2\|, \tag{2}$$

$$\forall \theta_1, \theta_2, \|\nabla_{\theta_1} \mathcal{L}_r(\mathcal{D}_r, \theta_1) - \nabla_{\theta_2} \mathcal{L}_r(\mathcal{D}_r, \theta_2)\| \leq L\|\theta_1 - \theta_2\|. \tag{3}$$

Theoretical Guarantees

$$(\text{Shared Momentum})\begin{cases} \boldsymbol{m}_{f,t}^S &= \alpha \boldsymbol{m}_{r,t-1}^S + \widehat{\boldsymbol{g}}_{f,t}^S, \quad \theta_{f,t}^S = \theta_{r,t-1}^S - \eta \boldsymbol{m}_{f,t}^S \\ \boldsymbol{m}_{r,t}^S &= \alpha \boldsymbol{m}_{f,t}^S + \widehat{\boldsymbol{g}}_{r,t}^S, \quad \theta_{r,t}^S = \theta_{f,t}^S - \eta \boldsymbol{m}_{r,t}^S \end{cases}$$

$$(\text{Decoupled Momentum})\begin{cases} \boldsymbol{m}_{f,t}^D &= \alpha \boldsymbol{m}_{f,t-1}^D + \widehat{\boldsymbol{g}}_{f,t}^D, \quad \theta_{f,t}^D = \theta_{r,t-1}^D - \eta \boldsymbol{m}_{f,t}^D \\ \boldsymbol{m}_{r,t}^D &= \alpha \boldsymbol{m}_{r,t-1}^D + \widehat{\boldsymbol{g}}_{r,t}^D, \quad \theta_{r,t}^D = \theta_{f,t}^D - \eta \boldsymbol{m}_{r,t}^D \end{cases}$$

Lemma

(Variance of Gradients) If the loss function $\mathcal{L}$ is Lipschitz smooth with a constant L, and $\mathrm{Var}(\theta) \leq \sigma_\theta^2$, then we have $\mathrm{Var}(\nabla_\theta \mathcal{L}(\theta)) \leq L^2 \sigma_\theta^2$.

Theorem

(Variance Bound Comparison for Decoupled vs. Shared Momentum) For the shared and decoupled schemes using the same hyperparameters (η, α), and we use $\overline{\mathrm{Var}}(\cdot)$ to denote the maximum variance of a variable, if the function $\mathcal{L}_f$, $\mathcal{L}_r$ and the stochastic gradient $\{(\widehat{\boldsymbol{g}}_{f,i}^S, \widehat{\boldsymbol{g}}_{r,i}^S)\}_{i=0}^{T-1}$, $\{(\widehat{\boldsymbol{g}}_{f,i}^D, \widehat{\boldsymbol{g}}_{r,i}^D)\}_{i=0}^{T-1}$ satisfy the assumptions, then

$$\forall t, \overline{\mathrm{Var}}(\theta_{f,t}^D) \leq \overline{\mathrm{Var}}(\theta_{f,t}^S), \quad \overline{\mathrm{Var}}(\theta_{r,t}^D) \leq \overline{\mathrm{Var}}(\theta_{r,t}^S),$$

DualOptim

Algorithm 1: Machine Unlearning with Shared Optimizer / Dual Optimizers

Input: Model: f_θ; Forget set: $\mathcal{D}_f$; Retain set: $\mathcal{D}_r$; Iterations for outer loop: T_o; Iterations for forgetting: T_f; Iterations for retaining: T_r; Step sizes: η , η_f, η_r .

Optim is the same optimizer as in pretraining with step size η.

Optim_f is $\mathrm{Adam}(\theta, \eta_f)$, Optim_r is the same optimizer as in pretraining with step size η_r.

for $t = 1, \ldots, T_o$ **do**

  for $t' = 1, \ldots, T_f$ **do**

    Fetch mini-batch data from the forget set $B_f \sim \mathcal{D}_f$

    Calculate the forget loss $\mathcal{L}_f$ on B_f and get the gradient

    Use Optim / Optim_f to update θ

  end for

  for $t' = 1, \ldots, T_r$ **do**

    Fetch mini-batch data from the retain set $B_r \sim \mathcal{D}_r$

    Calculate the retain loss $\mathcal{L}_r$ on B_r and get the gradient

    Use Optim / Optim_r to update θ

  end for

end for

Output: Model f_θ

Experiments: Image Classification

Table 1: Performance summary of MU methods for image classification. Experiments are conducted on (a) 10% random subset of **CIFAR-10** using **ResNet-18** and **(b)** 10% random subset of **TinyImageNet** using **Swin-T**. All results are presented as mean and standard deviation across 5 trials with different random forget data. Performance gaps with *RT* are indicated in blue. The average gap (**Gap**) and average standard deviation (**Std**) metrics are calculated by the average of the gaps and standard deviation measured in FA, RA, TA, and MIA, respectively. All the numbers are in percentage.

(a) **CIFAR-10 Random Subset Unlearning (10%)**

Method	FA	RA	TA	MIA	Gap ↓	Std ↓
RT	$94.61_{\pm 0.46}$ (0.00)	$100.00_{\pm 0.00}$ (0.00)	$94.25_{\pm 0.18}$ (0.00)	$76.26_{\pm 0.54}$ (0.00)	0.00	0.30
FT	$99.16_{\pm 0.10}$ (4.55)	$99.84_{\pm 0.06}$ (0.16)	$94.10_{\pm 0.09}$ (0.15)	$88.77_{\pm 0.38}$ (12.51)	4.34	0.16
GA	$98.76_{\pm 0.39}$ (4.15)	$99.10_{\pm 0.90}$ (0.90)	$93.89_{\pm 0.41}$ (0.36)	$92.58_{\pm 0.55}$ (16.32)	5.43	0.44
RL	$97.19_{\pm 0.21}$ (2.58)	$99.67_{\pm 0.08}$ (0.33)	$94.03_{\pm 0.27}$ (0.22)	$68.19_{\pm 0.95}$ (8.43)	2.80	0.38
SCRUB	$92.88_{\pm 0.25}$ (1.73)	$99.62_{\pm 0.10}$ (0.38)	$93.54_{\pm 0.22}$ (0.71)	$82.78_{\pm 0.86}$ (6.52)	2.33	**0.36**
+DualOptim	$94.90_{\pm 0.42}$ (0.29)	$99.52_{\pm 0.09}$ (0.48)	$93.50_{\pm 0.20}$ (0.75)	$78.26_{\pm 0.79}$ (2.00)	**0.88**	0.38
SalUn	$96.99_{\pm 0.31}$ (2.38)	$99.40_{\pm 0.28}$ (0.60)	$93.84_{\pm 0.36}$ (0.41)	$65.76_{\pm 1.05}$ (10.50)	3.47	0.50
+DualOptim	$95.47_{\pm 0.22}$ (0.86)	$99.06_{\pm 0.94}$ (0.60)	$92.47_{\pm 0.29}$ (1.78)	$76.14_{\pm 0.70}$ (0.12)	**0.93**	**0.35**
SFRon	$94.67_{\pm 3.03}$ (0.06)	$99.83_{\pm 0.13}$ (0.17)	$93.98_{\pm 0.56}$ (0.27)	$77.80_{\pm 5.61}$ (1.54)	0.51	2.33
+DualOptim	$94.69_{\pm 1.13}$ (0.02)	$99.92_{\pm 0.01}$ (0.08)	$94.11_{\pm 0.11}$ (0.14)	$77.77_{\pm 1.39}$ (1.51)	**0.44**	**0.66**

(b) **TinyImageNet Random Subset Unlearning (10%)**

Method	FA	RA	TA	MIA	Gap ↓	Std ↓
RT	$85.29_{\pm 0.09}$ (0.00)	$99.55_{\pm 0.03}$ (0.00)	$85.49_{\pm 0.15}$ (0.00)	$69.30_{\pm 0.20}$ (0.00)	0.00	0.12
FT	$96.50_{\pm 0.10}$ (11.21)	$98.23_{\pm 0.08}$ (1.32)	$82.67_{\pm 0.21}$ (2.82)	$79.85_{\pm 0.13}$ (10.55)	6.48	0.13
GA	$90.02_{\pm 3.26}$ (4.73)	$90.84_{\pm 3.29}$ (8.71)	$75.64_{\pm 2.67}$ (9.85)	$78.97_{\pm 2.07}$ (9.67)	8.24	2.82
RL	$94.66_{\pm 0.26}$ (9.37)	$98.02_{\pm 0.14}$ (1.53)	$82.73_{\pm 0.27}$ (2.76)	$54.45_{\pm 1.04}$ (15.15)	7.13	0.43
SCRUB	$97.80_{\pm 0.16}$ (12.51)	$98.13_{\pm 0.08}$ (1.42)	$82.64_{\pm 0.19}$ (2.85)	$79.62_{\pm 0.41}$ (10.32)	6.78	**0.21**
+DualOptim	$97.20_{\pm 0.20}$ (11.91)	$98.30_{\pm 0.10}$ (1.25)	$83.17_{\pm 0.19}$ (2.32)	$79.10_{\pm 0.63}$ (9.80)	**6.32**	0.28
SalUn	$97.69_{\pm 0.14}$ (12.40)	$98.89_{\pm 0.03}$ (0.66)	$84.02_{\pm 0.32}$ (1.47)	$61.87_{\pm 0.97}$ (7.43)	5.49	0.37
+DualOptim	$91.68_{\pm 0.28}$ (6.39)	$95.13_{\pm 0.18}$ (4.42)	$80.16_{\pm 0.34}$ (5.33)	$72.48_{\pm 0.33}$ (3.18)	**4.83**	**0.28**
SFRon	$96.41_{\pm 0.74}$ (11.12)	$98.95_{\pm 0.22}$ (0.60)	$83.40_{\pm 0.51}$ (2.09)	$70.40_{\pm 3.15}$ (1.10)	3.73	1.16
+DualOptim	$92.26_{\pm 1.44}$ (6.97)	$98.27_{\pm 0.12}$ (1.28)	$83.12_{\pm 0.21}$ (2.37)	$69.19_{\pm 2.27}$ (0.11)	**2.68**	**1.01**

Experiments: Image Generation

Table 2: Class-wise unlearning performance on **CIFAR-10** with **DDPM** and **ImageNet** with **DiT**. The best unlearning performance for each forgetting class is highlighted in **bold** for FA (in %) and FID. Note that the results of SA, SalUn and SFRon are those reported in [7].

Method	**CIFAR-10 Class-wise Unlearning**										**ImageNet Class-wise Unlearning**									
	Automobile		Cat		Dog		Horse		Truck		Cockatoo		Golden Retriever		White Wolf		Arctic Fox		Otter	
	FA ↓	FID ↓	FA ↓	FID ↓	FA ↓	FID ↓	FA ↓	FID ↓	FA ↓	FID ↓	FA ↓	FID ↓	FA ↓	FID ↓	FA ↓	FID ↓	FA ↓	FID ↓	FA ↓	FID ↓
SA	**0.00**	23.56	14.20	21.34	8.60	21.19	**0.00**	21.13	**0.00**	29.04	**0.00**	348.75	**0.00**	298.97	**0.00**	45.89	**0.00**	393.91	29.8	321.21
SalUn	0.20	21.23	1.40	20.29	**0.00**	20.18	0.60	20.70	0.80	20.45	91.21	18.47	46.09	25.28	**0.00**	15.16	45.90	408.07	87.5	19.69
SFRon	**0.00**	20.70	7.40	**18.44**	0.20	18.89	**0.00**	19.93	**0.00**	20.61	**0.00**	**13.59**	**0.00**	17.76	**0.00**	23.28	**0.00**	16.12	**0.00**	16.43
+DO	0.20	**19.72**	**1.00**	19.36	**0.00**	**18.58**	**0.00**	**18.91**	**0.00**	**17.26**	**0.00**	17.46	**0.00**	**14.63**	**0.00**	**14.72**	**0.00**	**14.91**	**0.00**	**14.55**

Experiments: Large Language Models

Table 4: Performance comparison of different MU methods on TOFU-finetuned **Phi-1.5** and **LLaMA 2**. The results include Model Capability (MC), Forget Efficacy (FE), and the average metric (Avg.) for forget 1%, 5% data, and 10% data.

Method	**Phi-1.5**									**LLaMA 2**								
	forget 1% data			forget 5% data			forget 10% data			forget 1% data			forget 5% data			forget 10% data		
	MC ↑	FE ↑	Avg. ↑	MC ↑	FE ↑	Avg. ↑	MC↑	FE ↑	Avg. ↑	MC ↑	FE ↑	Avg. ↑	MC ↑	FE ↑	Avg. ↑	MC ↑	FE ↑	Avg. ↑
GA+GD	0.4934	0.4493	0.4714	0.4360	0.5084	0.4722	0.4471	0.5246	0.4859	0.6696	0.5908	0.6302	0.0000	0.8772	0.4386	0.5592	0.9346	0.7469
ME+GD	**0.4944**	0.3938	0.4441	0.4559	0.4480	0.4520	0.4594	0.4564	0.4579	0.7271	0.9204	0.8237	**0.7472**	0.9313	0.8392	**0.7357**	0.9489	0.8423
+DO	0.4866	**0.6913**	**0.5889**	**0.4676**	**0.8200**	**0.6438**	**0.5009**	**0.7732**	**0.6370**	**0.7425**	**0.9612**	**0.8519**	0.7316	**0.9602**	**0.8459**	0.7315	**0.9625**	**0.8470**
DPO+GD	0.2410	0.6831	0.4621	0.4105	0.6334	0.5219	0.3517	0.6302	0.4910	0.7564	0.5335	0.6450	0.0000	0.8243	0.4122	0.0000	0.8041	0.4021
IDK+AP	**0.4403**	0.5723	0.5063	**0.4800**	0.5112	0.4956	**0.4614**	0.6003	0.5308	**0.7580**	0.7625	0.7603	**0.7529**	0.7479	0.7504	**0.7471**	0.7433	0.7452
+DO	0.4221	**0.7037**	**0.5629**	0.4633	**0.6974**	**0.5804**	0.4422	**0.7193**	**0.5807**	0.7412	**0.8075**	**0.7743**	0.7354	**0.7958**	**0.7656**	0.7362	**0.7855**	**0.7609**

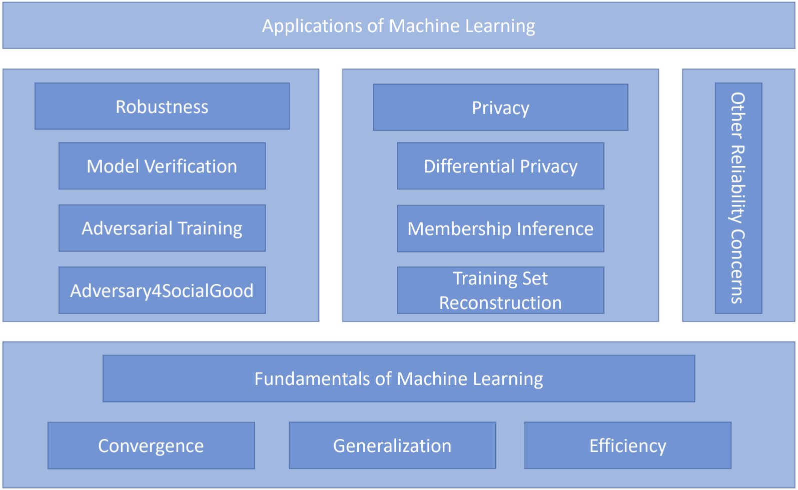
Acknowledgements

- Xuyang Zhong
- Haochen Luo

Outline

A Brief Introduction of MLO @ CityU HK

Overview of Research in Machine Learning Group



Machine Learning and Optimization (MLO) Group, CS@CityU HK

Post-doctoral Researcher

- Jinhui Hu (Distributed Learning, Differential Privacy)

Ph.D. Students

- Xuyang Zhong (Adversarial Robustness, Sparsity, Efficient Learning, Machine Unlearning)
- Ding Chen (Differential Privacy, Convex Optimization, Convergence Analysis)
- Xinping Chen (Training Data Reconstruction, Privacy Analysis)
- Xu Yang (Meta-learning, Adversarial Training, Large Language Model, Multimodality)
- Jingning Xu (Large Language Model, Robustness)
- Zhiyang Wu (Large Language Model, Human-Computer Interaction)
- Xiaofei Wang (Adversarial Robustness, Efficient Learning)

M.Phil. Students

- Haochen Luo (Large Language Model, AI4Finance)
- Jiandong Chen (Large Language Model, Efficient Learning)

Alumni

- Yixiao Huang (Large Langue Model, Optimizer, Calibration)
  - Research Assistant, 2024, $\longrightarrow$ Ph.D. student at UC Berkeley.
- Cesare Bergossi (AI4Finance)
  - Project Advisee, 2024, $\longrightarrow$ M.Sc. student at Imperial College London.
- Yulun Jiang (Efficient Learning, Robustness)
  - Project Advisee, 2023, $\longrightarrow$ Ph.D. student at EPFL.
- Shuangqi Li (Robustness, Generative Model)
  - Project Advisee, 2023, $\longrightarrow$ Ph.D. student at EPFL.
- Ziqi Zhao (Efficient Learning, Robustness)
  - Project Advisee, 2022, $\longrightarrow$ Ph.D. student at HKU.

Some Projects and Works

Adversarial Robustness

- Robustness Verification
  - Geometric-inspired robustness verification and provably robust learning.
- Optimization Properties of Adversarial Training
  - Convergence and generalization gaps.
- Accelerated Robust Learning.
  - Adversarial training on pruned and quantized networks.
  - Stable adversarial training by fast but cheap adversarial attack.
- Robustness for Different Threat Models.
  - Sparse attack and structured sparse attack.
- Adversarial Attack for Social Good
  - Manipulate generative models.
  - Decrease learnability to enhance privacy.

C. Liu, R. Tomioka, V. Cevher. "On Certifying Non-uniform Bounds against Adversarial Attack." ICML 2019.

C. Liu, M. Salzmann, S. Süsstrunk. "Training Provably Robust Models against Polyhedral Envelope Regularization." IEEE TNNLS 2021.

C. Liu*, Z. Zhao*, S. Süsstrunk, M. Salzmann. "Robust Binary Models by Pruning Randomly-initialized Networks." NeurIPS 2022.

Y. Jiang*, C. Liu*, M. Salzmann, S. Süsstrunk. "Towards Stable and Efficient Adversarial Training against l_1 Bounded Adversarial Attacks." ICML 2023.

X. Zhong, Y. Huang, C. Liu. "Towards Efficient Training and Evaluation of Robust Models against l_0 Bounded Adversarial Perturbations." ICML 2024.

Some Projects and Works

Some theoretical works:

- Convergence Analysis
  - First smooth analyses and convergence bound on the loss landscape of adversarial training.
- Generalization Bounds
  - Derive the generalization bound in terms of data variance and adversarial perturbation magnitude in adversarial training, showing data of larger conditional variance mainly contributes to robust overfitting.
- Nash Equilibrum in Min-Max Problems.
  - Using mirror descent and mirror-prox to guarantee mixed Nash Equilibria on Generative Adversarial Networks (GANs).
- Differential Privacy
  - Trace the evolution of privacy loss (Renyi Divergence between models trained by two consecutive dataset) under the hidden state assumption where only the last state of the model is leaked to the adversary.

Y-P. Hsieh, C. Liu, V. Cevher. "Finding Mixed Nash Equilibria of Generative Adversarial Networks". ICML 2019.

C. Liu, M. Salzmann, T. Lin, R. Tomioka, S. Süsstrunk. "On the Loss Landscape of Adversarial Training: Identifying Challenges and How to Overcome Them." NeurIPS 2020.

C. Liu, Z. Huang, M. Salzmann, T. Zhang, S. Süsstrunk. "On the Impact of Hard Adversarial Instances on Overfitting in Adversarial Training." Journal of Machine Learning Research 2024.

D. Chen, C. Liu. "Differentially Private Neural Network Training under Hidden State Assumption".

Thank You!