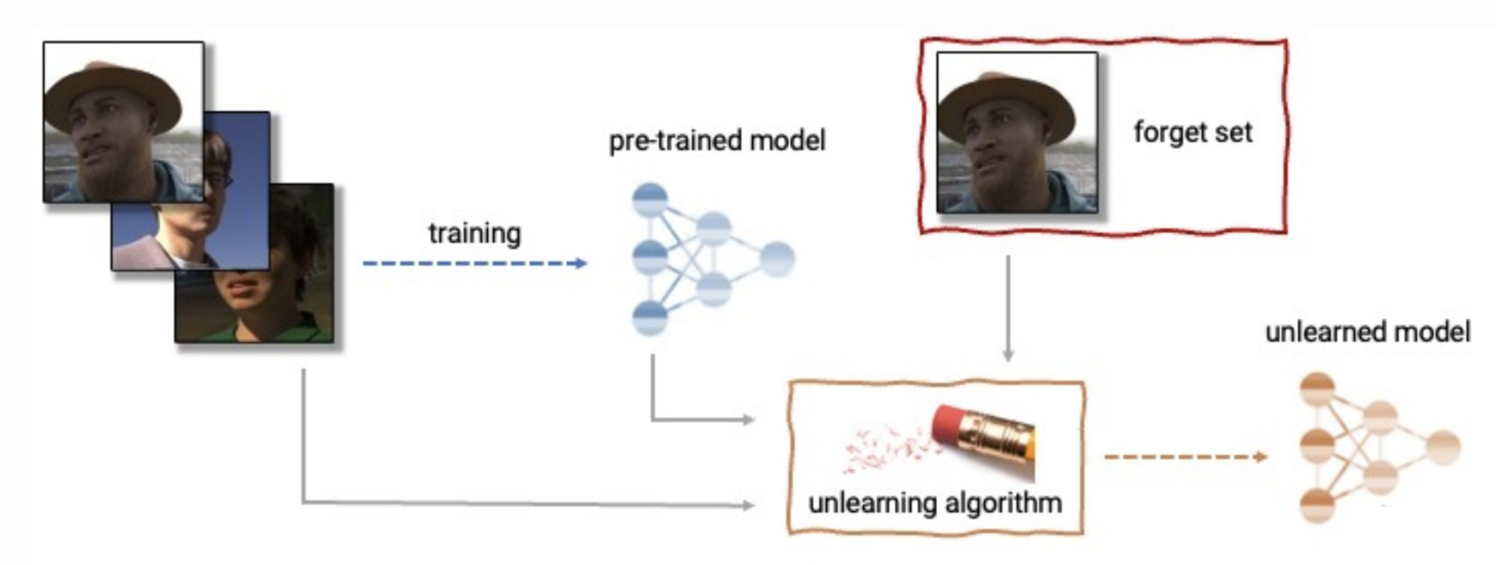
One-Shot Machine Unlearning with Mnemonic Code

ACML 2024

Tomoya Yamashita, Masanori Yamada, Takashi Shibata
NTT Laboratories

What is Machine Unlearning?



Challenges in Machine Unlearning

- We usually do not have access to the data.
- + we should not have access to the data.
- Computational costs

All above are addressed in this paper.

Fisher Information Matrix / Hessian Matrix

$$I(\theta) = \mathbb{E}\left[\left(\frac{\partial \log L(\theta)}{\partial \theta}\right)\left(\frac{\partial \log L(\theta)}{\partial \theta}\right)^T\right]$$

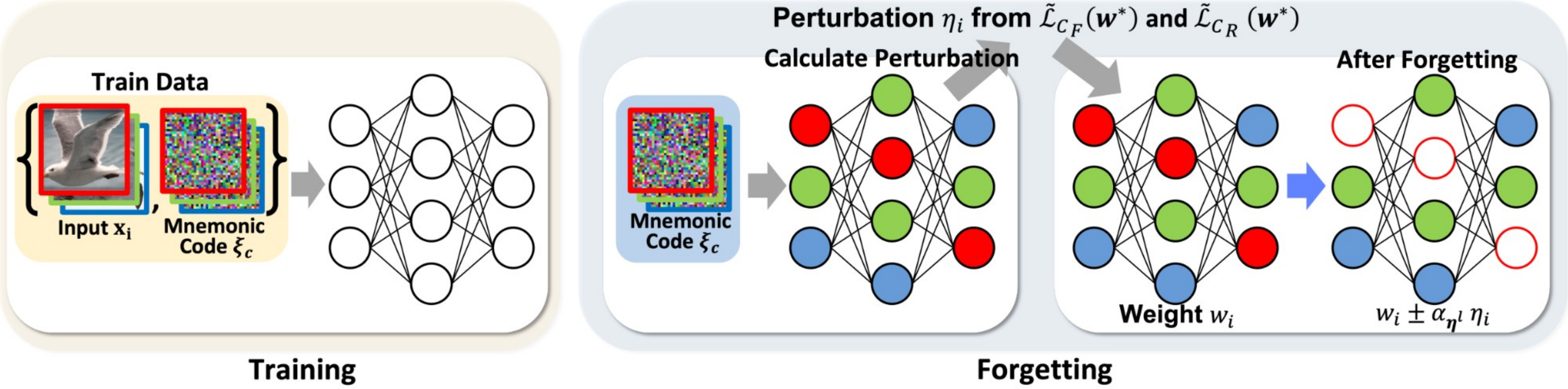
$$H(\theta) = \begin{bmatrix} \frac{\partial^2 f}{\partial \theta_1^2} & \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2} & \cdots \\ \frac{\partial^2 f}{\partial \theta_2 \partial \theta_1} & \frac{\partial^2 f}{\partial \theta_2^2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

Diagonal Approximation

$$H(\theta) = \begin{bmatrix} \frac{\partial^2 f}{\partial \theta_1^2} & \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2} & \cdots \\ \frac{\partial^2 f}{\partial \theta_2 \partial \theta_1} & \frac{\partial^2 f}{\partial \theta_2^2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

$$H_{\text{diag}}(\theta) = \begin{bmatrix} \frac{\partial^2 f}{\partial \theta_1^2} & 0 & \cdots \\ 0 & \frac{\partial^2 f}{\partial \theta_2^2} & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix}$$

Method of the Paper (1/5)



$$\boldsymbol{\xi} \sim N(\mathbf{0}, \mathbf{1})$$

Method of the Paper (2/5)

Algorithm 1 Training with mnemonic code

Input: dataset $\boldsymbol{x} \sim p^d(\boldsymbol{x})$, model parameter $\boldsymbol{w}$, loss $\mathcal{L}$
Parameter: mnemonic code replacing probability t_{mix},
learning rate lr
Output: trained model parameters

$\boldsymbol{\xi} \sim N(\mathbf{0}, \mathbf{1})$
for e in epochs **do**
    for i in datasize **do**
        $t \sim U(0, 1)$
        if $t < t_{\text{mix}}$ **then**
            $\tilde{\boldsymbol{x}}_i = \boldsymbol{\xi}_c$
        else
            $\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i$
        end if
    end for
    $\boldsymbol{w} = \boldsymbol{w} - \text{lr} \nabla_{\boldsymbol{w}} \mathcal{L}(\tilde{\boldsymbol{x}}; \boldsymbol{w})$
end for

Method of the Paper (3/5)

$$
\begin{aligned}
&\mathcal{L}_{\mathcal{C}_{\mathrm{F}}}(\boldsymbol{w}^* + \boldsymbol{\delta}) \\
&\simeq \mathcal{L}_{\mathcal{C}_{\mathrm{F}}}(\boldsymbol{w}^*) + \frac{1}{2}\boldsymbol{\delta}^{\mathrm{T}} F_{\mathcal{C}_{\mathrm{F}}} \boldsymbol{\delta} \\
&= \mathcal{L}_{\mathcal{C}_{\mathrm{F}}}(\boldsymbol{w}^*) + \frac{1}{2}\boldsymbol{\delta}^{\mathrm{T}} \{t_{\mathrm{mix}} F^{\xi}_{\mathcal{C}_{\mathrm{F}}} + (1 - t_{\mathrm{mix}}) F^{d}_{\mathcal{C}_{\mathrm{F}}}\} \boldsymbol{\delta} \\
&\simeq \mathcal{L}_{\mathcal{C}_{\mathrm{F}}}(\boldsymbol{w}^*) + \frac{1}{2}\{t_{\mathrm{mix}} \sum_i \boldsymbol{f}^{\xi}_{\mathcal{C}_{\mathrm{F}},i} + (1 - t_{\mathrm{mix}}) \sum_i \boldsymbol{f}^{d}_{\mathcal{C}_{\mathrm{F}},i}\} \boldsymbol{\delta}_i^2,
\end{aligned} \tag{2}
$$

Taylor Series Expansion / Laplace’s Method

$$f(x) = f(a) + f'(a)(x-a) + \frac{1}{2} f''(a)(x-a)^2 + O((x-a)^3)$$

$$\mathcal{L}(\boldsymbol{w}^* + \boldsymbol{\delta}) \approx \mathcal{L}(\boldsymbol{w}^*) + \nabla \mathcal{L}(\boldsymbol{w}^*)^\top \boldsymbol{\delta} + \frac{1}{2} \boldsymbol{\delta}^\top H \boldsymbol{\delta} + O(\|\boldsymbol{\delta}\|^3)$$

$$\mathcal{L}(\boldsymbol{w}^* + \boldsymbol{\delta}) \approx \mathcal{L}(\boldsymbol{w}^*) + \frac{1}{2} \boldsymbol{\delta}^\top H \boldsymbol{\delta}$$

$$\mathcal{L}_{\mathcal{C}_\mathrm{F}}(\boldsymbol{w}^* + \boldsymbol{\delta}) \approx \mathcal{L}_{\mathcal{C}_\mathrm{F}}(\boldsymbol{w}^*) + \frac{1}{2} \sum_i f_{\mathcal{C}_\mathrm{F}, i} \boldsymbol{\delta}_i^2$$

Method of the Paper (4/5)

$$w_i = w_i \pm \alpha_{\boldsymbol{\eta}^l} \eta_i, \tag{5}$$

$$\eta_i = \frac{f_{\mathcal{C}_{\mathrm{F}},i}}{f_{\mathcal{C}_{\mathrm{R}},i}} = \frac{\frac{1}{|\#\mathcal{C}_{\mathrm{F}}|} \sum_{j \in \mathcal{C}_{\mathrm{F}}} \mathbb{E}\left[\left(\frac{\partial \mathcal{L}_j}{\partial w_i}\right)^2\right]}{\frac{1}{|\#\mathcal{C}_{\mathrm{R}}|} \sum_{k \in \mathcal{C}_{\mathrm{R}}} \mathbb{E}\left[\left(\frac{\partial \mathcal{L}_k}{\partial w_i}\right)^2\right]}, \tag{6}$$

$$\alpha_{\boldsymbol{\eta}^l} = \min\left(\lambda_1, \frac{\lambda_2}{\max_{\eta_i \in \boldsymbol{\eta}^l} \eta_i}\right), \tag{7}$$

Metho

Algorithm 2 Forgetting with mnemonic code

Input: trained model parameter $\boldsymbol{w}$, loss $\mathcal{L}$, forget class set $\mathcal{C}_{\mathrm{F}}$, remain class set $\mathcal{C}_{\mathrm{R}}$, mnemonic codes $\boldsymbol{\xi}$, layers $\{l_1, l_2, \cdots\}$

Parameter: λ_1, λ_2

Output: Forgotten parameters

$\boldsymbol{f}_{\mathcal{C}_{\mathrm{F}}} = \mathbf{0}$

$\boldsymbol{f}_{\mathcal{C}_{\mathrm{R}}} = \mathbf{0}$

for c in $\mathcal{C}_{\mathrm{F}}$ **do**

  $\boldsymbol{f}_{\mathcal{C}_{\mathrm{F}}} = \boldsymbol{f}_{\mathcal{C}_{\mathrm{F}}} + \nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{\xi}_c; \boldsymbol{w})$

end for

for c in $\mathcal{C}_{\mathrm{R}}$ **do**

  $\boldsymbol{f}_{\mathcal{C}_{\mathrm{R}}} = \boldsymbol{f}_{\mathcal{C}_{\mathrm{R}}} + \nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{\xi}_c; \boldsymbol{w})$

end for

$\boldsymbol{f}_{\mathcal{C}_{\mathrm{F}}} = \boldsymbol{f}_{\mathcal{C}_{\mathrm{F}}} / |\#\mathcal{C}_{\mathrm{F}}|$

$\boldsymbol{f}_{\mathcal{C}_{\mathrm{R}}} = \boldsymbol{f}_{\mathcal{C}_{\mathrm{R}}} / |\#\mathcal{C}_{\mathrm{R}}|$

$\boldsymbol{\eta} = \frac{\boldsymbol{f}_{\mathcal{C}_{\mathrm{F}}}}{\boldsymbol{f}_{\mathcal{C}_{\mathrm{R}}}}$

for l in layers **do**

  $\alpha_{\boldsymbol{\eta}^l} = \min\left(\lambda_1, \frac{\lambda_2}{\max_{\eta_i \in \boldsymbol{\eta}^l} \eta_i}\right)$

  $\boldsymbol{w}_1^l = \boldsymbol{w}^l + \alpha_{\boldsymbol{\eta}^l} \boldsymbol{\eta}^l$

  $\boldsymbol{w}_2^l = \boldsymbol{w}^l - \alpha_{\boldsymbol{\eta}^l} \boldsymbol{\eta}^l$

end for

if $A_{\mathrm{R}}(\boldsymbol{w}_1) + E_{\mathrm{F}}(\boldsymbol{w}_1) > A_{\mathrm{R}}(\boldsymbol{w}_2) + E_{\mathrm{F}}(\boldsymbol{w}_2)$ **then**

  return $\boldsymbol{w}_1$

else

  return $\boldsymbol{w}_2$

end if

Results (1/4)

Method	Processing Time	Data-Free	MU Target
CertifiedRemoval (Guo et al., 2020)	$\mathcal{O}(N_{\mathrm{R}} + N_{\mathrm{F}})$	✗	item
SISA (Bourtoule et al., 2021)	$\mathcal{O}(E \cdot \frac{N_{\mathrm{R}}}{M})$	✗	item
Arcane (Yan et al., 2022)	$\mathcal{O}(E \cdot \frac{N_{\mathrm{R}}}{C_{\mathrm{R}}+C_{\mathrm{F}}})$	✗	item
FastMU (Tarun et al., 2023)	$\mathcal{O}(S \cdot C_{\mathrm{F}} + E \cdot C_{\mathrm{F}} + N_{\mathrm{R}})$	✓	class
ZeroShotMU (Chundawat et al., 2023)	$\mathcal{O}((S + E)(C_{\mathrm{F}} + C_{\mathrm{R}}))$	✓	class
LwSF (Shibata et al., 2021)	$\mathcal{O}(E(N_{\mathrm{new}} + C_{\mathrm{R}}))$	✓	class/task
SFDN (Golatkar et al., 2020a)	$\mathcal{O}(N_{\mathrm{R}})$	✗	class/item
NTK-F (Golatkar et al., 2020b)	$\mathcal{O}(N_{\mathrm{R}} + N_{\mathrm{F}})$	✗	class/item
SSD (Foster et al., 2024)	$\mathcal{O}(N_{\mathrm{R}} + N_{\mathrm{F}})$	✗	class/item
ERM-KTP (Lin et al., 2023)	$\mathcal{O}(E \cdot N_{\mathrm{R}})$	✗	class
Ours	$\mathcal{O}(C_{\mathrm{R}} + C_{\mathrm{F}})$	✓	class

Results (2/4)

Table 2: **Comparison results in A_{R}.** We evaluate the baseline and our methods three times and provide the mean and standard deviation. The highest values are shown in bold.

	MNIST	CIFAR10	CUB	STN
FastMU	96.5 ±0.1	90.4 ±0.5	73.1 ±1.3	88.0 ±0.1
LwSF	43.7 ±9.6	65.4 ±16.6	68.2 ±3.5	80.1 ±6.7
SFDN	94.1 ±0.7	93.4 ±0.2	78.2 ±0.6	88.3 ±0.6
SSD	**96.9** ±0.0	94.2 ±0.0	44.3 ±0.0	74.4 ±0.0
ERM-KTP	-	92.7 ±0.4	42.8 ±3.2	75.6 ±4.0
Ours	95.9 ±0.1	**94.4** ±0.1	**79.3** ±0.7	**91.7** ±0.3

Results (3/4)

Table 3: **Comparison results in E_{F}.** We evaluate the baseline and our methods three times and provide the mean and standard deviation. The highest values are shown in bold.

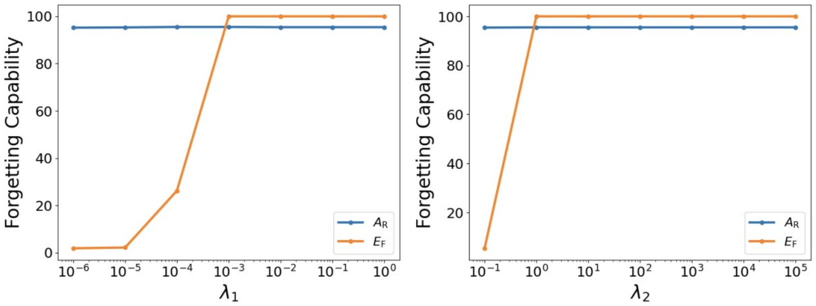
	MNIST	CIFAR10	CUB	STN
FastMU	98.0 ±0.3	**100** ±0.0	68.6 ±12.0	60.9 ±6.9
LwSF	94.4 ±1.7	**100** ±0.0	93.1 ±7.0	98.2 ±1.8
SFDN	**100** ±0.0	96.3 ±2.5	**100** ±0.0	**100** ±0.0
SSD	93.1 ±0.0	**100** ±0.0	**100** ±0.0	**100** ±0.0
ERM-KTP	-	**100** ±0.0	**100** ±0.0	**100** ±0.0
Ours	**100** ±0.0	**100** ±0.0	**100** ±0.0	**100** ±0.0

Results (4/4)

Table 4: **Forgetting for ImageNet dataset.** We perform fine-tuning using mnemonic code on the pre-trained model and forget with our method. We show the forgetting capability and processing time for the pre-trained, fine-tuned, and forgotten models, respectively.

Architecture		$A_{\mathrm{R}} \uparrow$	$E_{\mathrm{F}} \uparrow$	Time [s] $\downarrow$
ResNet-18	Pretrained	69.8	12.0	-
	Fine-tuned	67.5	12.0	882
	After MU	67.5	100	8.66
ResNeXt-50	Pretrained	77.4	6.0	-
	Fine-tuned	75.9	12.0	6923
	After MU	75.9	100	21.0
Swin-Transformer	Pretrained	80.9	4.0	-
	Fine-tuned	78.8	4.0	8488
	After MU	75.3	92.0	28.6

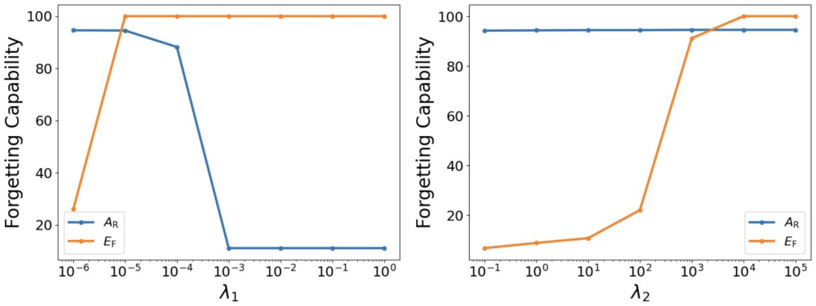
Hyperparameter Search



(a) λ_1 search.

(b) λ_2 search.

Figure 6: **Hyperparameter search on MNIST.**



(a) λ_1 search.

(b) λ_2 search.

Figure 7: **Hyperparameter search on CIFAR10.**

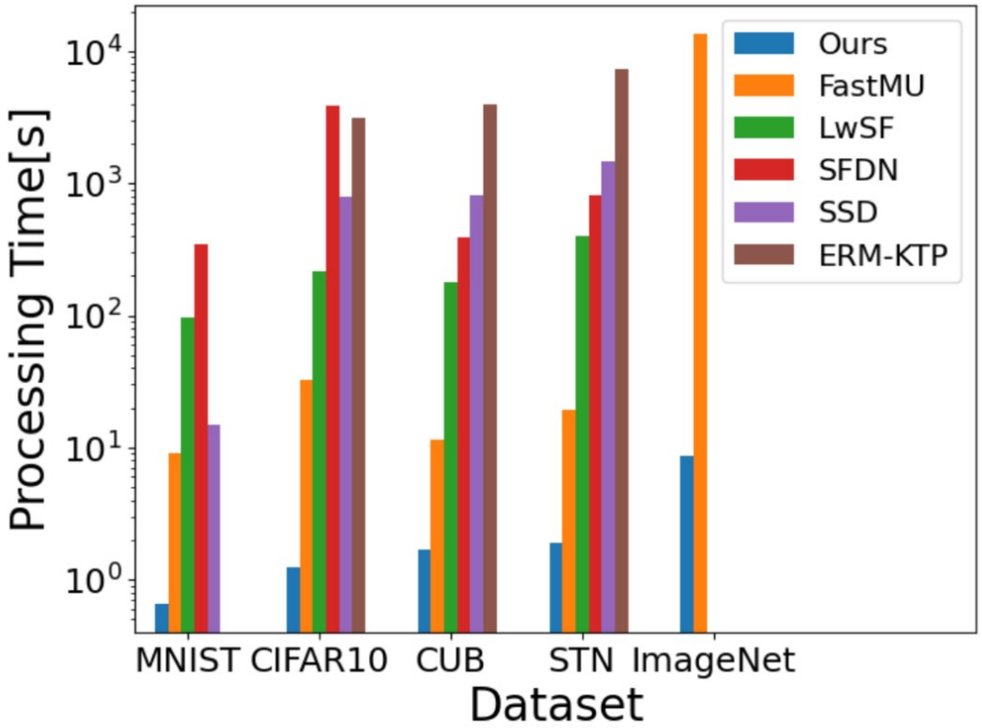


Figure 3: **Comparison results in MU processing time.** We measure the forgetting time concerning our method and the baselines.

Future Work

- Actually using Fisher Information Matrix
- Calculating Fisher Information only in important layers
- More realistic scenarios (not class-wise unlearning)