

# Machine Unlearning

CPSC 471/571: Trustworthy Deep Learning

Rex Ying

# Content

- Machine Unlearning
  - Introduction and Motivation
  - Unlearning Definition
  - Model-agnostic Approach
    - Boundary Unlearning
  - Model-intrinsic Approach
    - Projective Residual Update for Linear Models
    - Forget-Me-Not for Attention Model
    - SalUn for Diffusion Model
  - Data-driven Approach
    - SISA Training

# Content

- Machine Unlearning
  - **Introduction and Motivation**
  - Unlearning Definition
  - Model-agnostic Approach
    - Boundary Unlearning
  - Model-intrinsic Approach
    - Projective Residual Update for Linear Models
    - Forget-Me-Not for Attention Model
    - SalUn for Diffusion Model
  - Data-driven Approach
    - SISA Training

# Real-world Use Cases

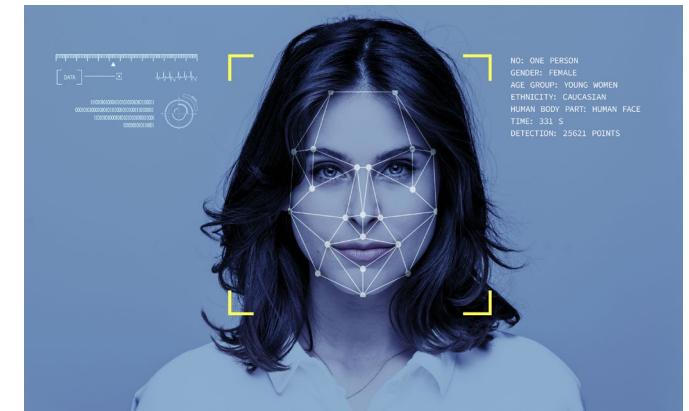
- There is a need for **information removal** from ML systems e.g., user's request, inappropriate information, sensitive data, outdated or inaccurate information.



Privacy Protection



Healthcare Data Sensitivity



Personal Identity Unlearning

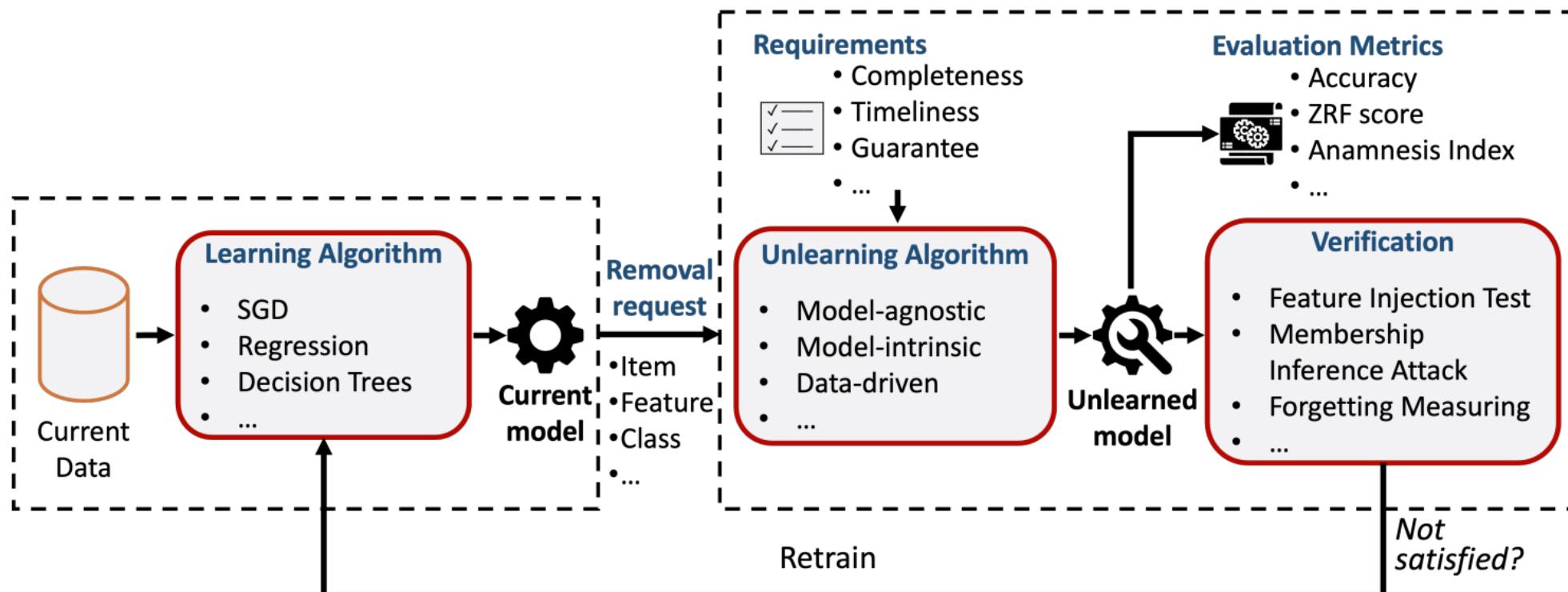
# Machine Unlearning Motivation

- **Motivation:** enable the ML system to **delete and unlearn certain information** to address the privacy concerns that arise from the data storage.
- **Naïve Solution:** Remove data point and **retrain the model from scratch** (exact unlearning of the data)
- However, **the cost of retraining is increasingly expensive** as ML models are trained on large datasets.

**Focus of this lecture: how to efficiently and effectively unlearn certain information?**

# Machine Unlearning Setting

- **Goal of Machine Unlearning:** To design **efficient (low retraining cost)** unlearning protocols that satisfy **guarantees of data removal** and **model performance**



# Types of Unlearning Requests

- **Item / data removal**
  - Requests to **remove certain items/data points** from the training set
- **Feature removal**
  - **Malicious or sensitive features** that are present in a set of data
  - Requests to remove certain features in a set of training data
- **Class/label removal**
  - The samples to be forgotten **belong to a single class**
  - Requests to **remove the entire class** and the samples belonging to it.
- **Concept removal**
  - Requests to remove specific **knowledge** or **patterns** learned by a model

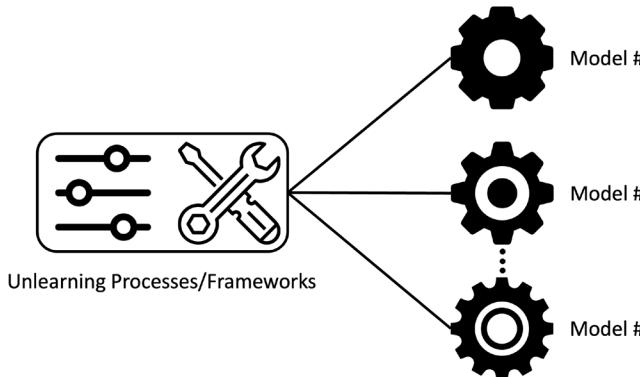
Examples?

# Design Requirements

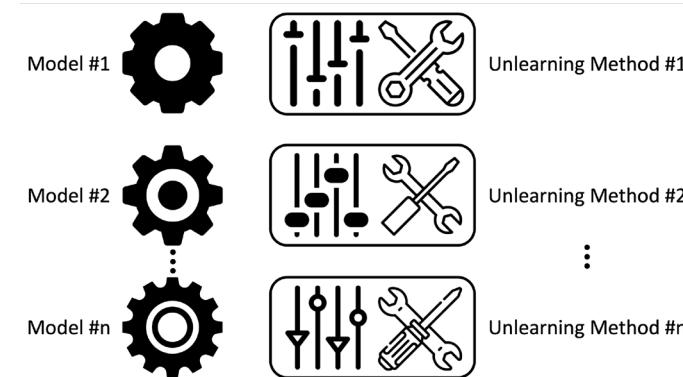
- **Completeness (Consistency)**: the unlearned model and the re-trained model should **make the same predictions about training samples**.
  - Usually formulated as *an optimization objective* in an unlearning algorithm
- **Timeliness and light-weight**: the unlearned model should be **more efficient** than retraining and **light-weight**
- **Model-agnostic**: an ideal unlearning algorithm should **be generic for different machine learning models**.
- **Provable guarantees**: the unlearning algorithm should be able to **provide provable guarantees of unlearning**

# Machine Unlearning Approaches

- **Model-agnostic Approaches**
  - Machine unlearning processes or frameworks that are **applicable for different models**, such as linear models, deep neural networks, etc.
- **Model-intrinsic Approaches**
  - Machine unlearning methods **designed for a specific type of models**.
- **Data-driven Approaches**
  - Using **data partition, data augmentation and data influence** to speed up the retraining process.

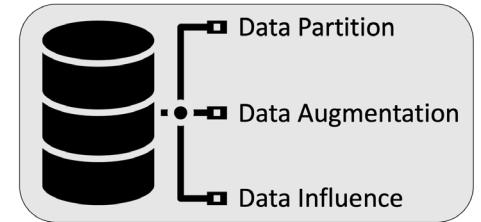


Model-agnostic Approaches



Model-intrinsic Approaches

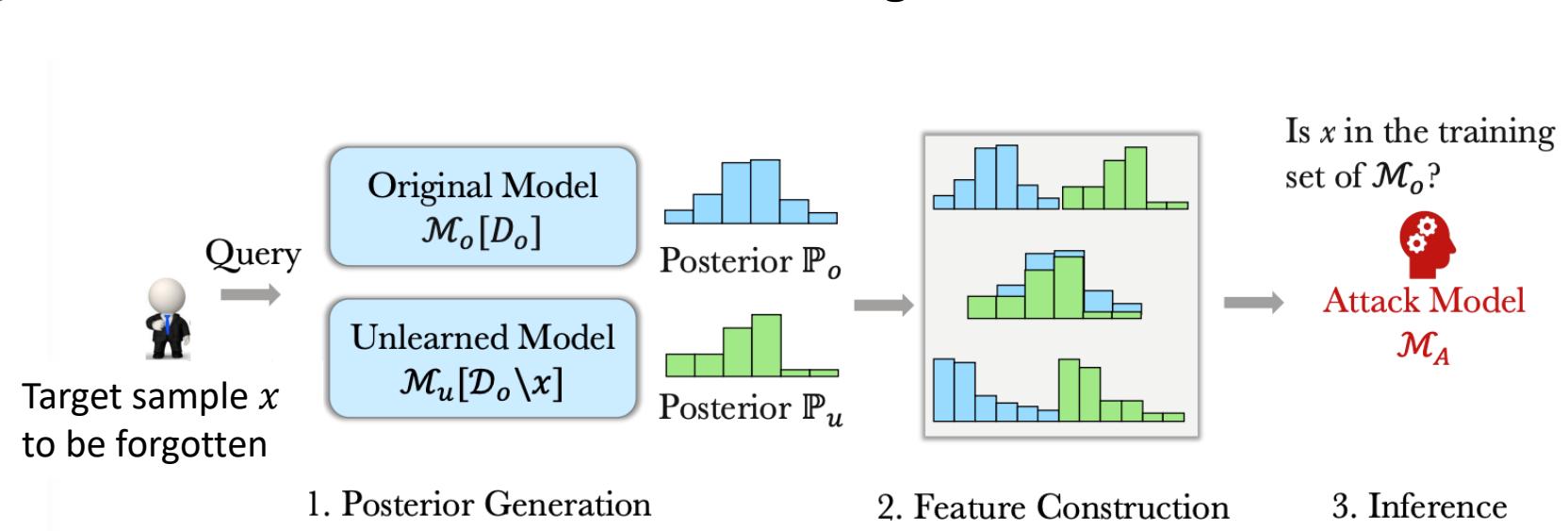
[Figure source](#)



Data-driven Approaches

# Unlearning Verification

- Goal: verify the effectiveness of the machine unlearning by detecting information leaks
- Methods: membership inference, feature injection test, backdoor attacks, etc.
- Recall the threats to information exposure:
  - e.g., **membership inference attack** is trained to predict *whether a data item belongs to the training set*. Hence, it can be used in **unlearning verification**



Chen, Min, et al. "When machine unlearning jeopardizes privacy."

# Content

- Machine Unlearning
  - Introduction and Motivation
  - **Unlearning Definition**
  - Model-agnostic Approach
    - Boundary Unlearning
  - Model-intrinsic Approach
    - Projective Residual Update for Linear Models
    - Forget-Me-Not for Attention Model
    - SalUn for Diffusion Model
  - Data-driven Approach
    - SISA Training

# Definition of Unlearning (1)

- **Train a model**  $A: \mathcal{Z}^* \rightarrow \mathcal{H}$ , where  $\mathcal{Z}^*$ : **all possible training datasets**;  $\mathcal{H}$ : a **hypothesis space** of all possible machine learning models
- $D_f \subset D$  denotes the forgetting dataset,  $D$  is the original dataset
- $U(D, D_f, A(D))$  : **the unlearned model**;
- $A(D \setminus D_f)$  : **the retrained model from scratch**.
- **Exact Unlearning:** the **distributions** of  $U(D, D_f, A(D))$  and  $A(D \setminus D_f)$  are exactly the same.  
**distribution of the parameters / outputs**
- **Approximate Unlearning:** the distance between distributions of  $U(D, D_f, A(D))$  and  $A(D \setminus D_f)$  is bounded.

# Definition of Unlearning (2)

- **Definition [Exact Unlearning]** Given a learning algorithm  $A(\cdot)$ , the unlearning algorithm  $U(\cdot)$  is *an exact unlearning process* iff  $\forall \mathcal{T} \subseteq \mathcal{H}, D \in \mathcal{Z}^*, D_f \subset D$ :

$$\Pr(A(D \setminus D_f) \in \mathcal{T}) = \Pr(U(D, D_f, A(D)) \in \mathcal{T})$$

where  $\Pr(\cdot)$  is the **distribution of the parameters / outputs** of the model over a function space  $\mathcal{T}$ .

- **Definition [ $\varepsilon$ -Approximate Unlearning]** Given a learning algorithm  $A(\cdot)$ , the unlearning algorithm  $U(\cdot)$  is *an  $\varepsilon$ -approximate unlearning process* iff  $\forall \mathcal{T} \subseteq \mathcal{H}, D \in \mathcal{Z}^*, z \in D$ :

$$e^{-\varepsilon} \leq \frac{\Pr(U(D, z, A(D)) \in \mathcal{T})}{\Pr(A(D \setminus z) \in \mathcal{T})} \leq e^{\varepsilon}$$

where  $z$  is the single removed sample

# Relationship to DP

- Recall differential privacy:

$$\forall \mathcal{T} \subseteq \mathcal{H}, D \text{ and } z \in D, e^{-\varepsilon} \leq \frac{\Pr(A(D) \in \mathcal{T})}{\Pr(A(D \setminus z) \in \mathcal{T})} \leq e^{\varepsilon}$$

where  $z$  is the single removed sample.

- **$\varepsilon$ -Differential Privacy implies  $\varepsilon$ -approximate unlearning.**

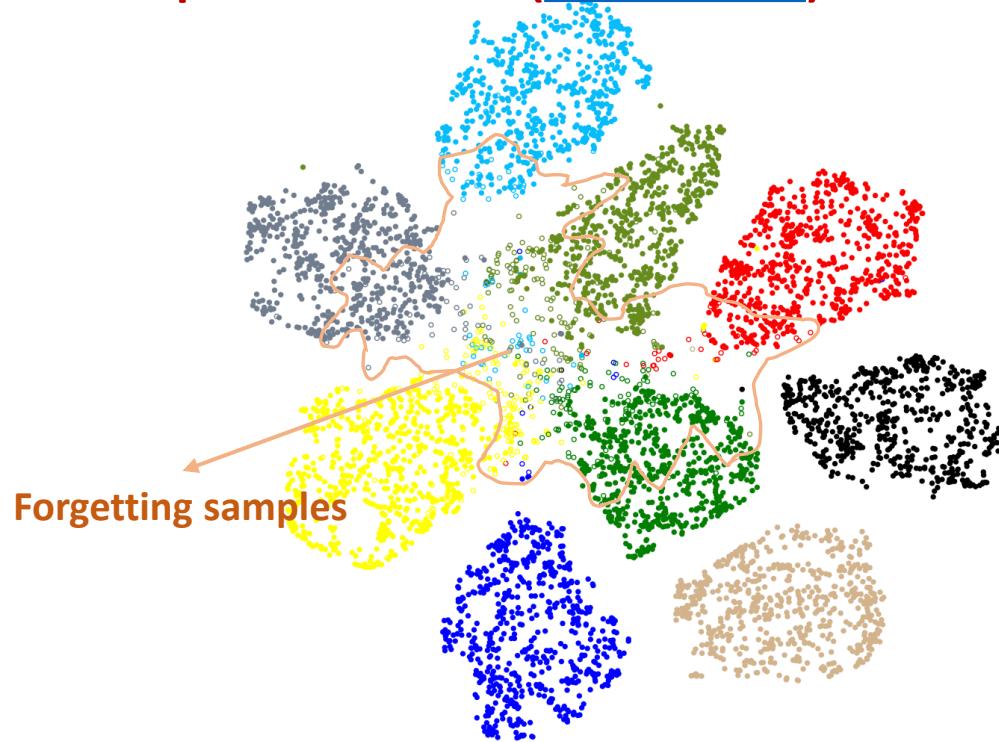
Note:  $\varepsilon$ -differential privacy is a very strong condition. If  $A(\cdot)$  is differentially private for any data, then it will **suffer significant accuracy loss!**

# Content

- Machine Unlearning
  - Introduction and Motivation
  - Unlearning Definition
  - **Model-agnostic Approach**
    - Boundary Unlearning
  - Model-intrinsic Approach
    - Projective Residual Update for Linear Models
    - Forget-Me-Not for Attention Model
    - SalUn for Diffusion Model
  - Data-driven Approach
    - SISA Training

# Decision Space of a Unlearned Model

Colors represent Classes ([Figure source](#))



The decision space of the **model retrained on the remaining dataset from scratch**

- The **solid dots** are remaining samples.
- The **hollow circles** are the forgotten samples

## Observation:

- the forgetting samples **spread around** the decision space of the retrained model  
=> the decision boundary of the forgetting samples has been broken after unlearning
- most of the forgetting samples **move to the borders of other clusters**  
=> samples at the cluster borders in the decision space will be predicted with large uncertainty.

# Boundary Unlearning (1)

- **Motivation:** Boundary unlearning **shifts the decision boundary** of the original model to **imitate** the decision behavior of the retrained model
- **Guideline:**
  - Destroy the boundary of the forgetting class
  - Maintain the boundary of the remain classes
  - Push the forgetting data to the border of other clusters
- Boundary Unlearning is a **model-agnostic** method with two approaches:
  - **Boundary Shrink:** breaks the decision boundary of the forgetting class
  - **Boundary Expanding:** disperse the forgetting class

# Boundary Unlearning (2)

- $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$  is the training dataset.  $\mathcal{Y} = \{1, \dots, K\}$  denotes the label space where  $K$  is the total number of classes.  $\mathcal{D}_f$  is the forgetting dataset.
- Parameters of the original model and the unlearned model are  $\boldsymbol{\theta}_o$  and  $\boldsymbol{\theta}_u$
- Boundary unlearning is designed for the case where  $\mathcal{D}_f$  consists of the samples of **an entire class**, e.g., class  $y$  (*a limitation of the work*)
- **Unlearning Objective:**

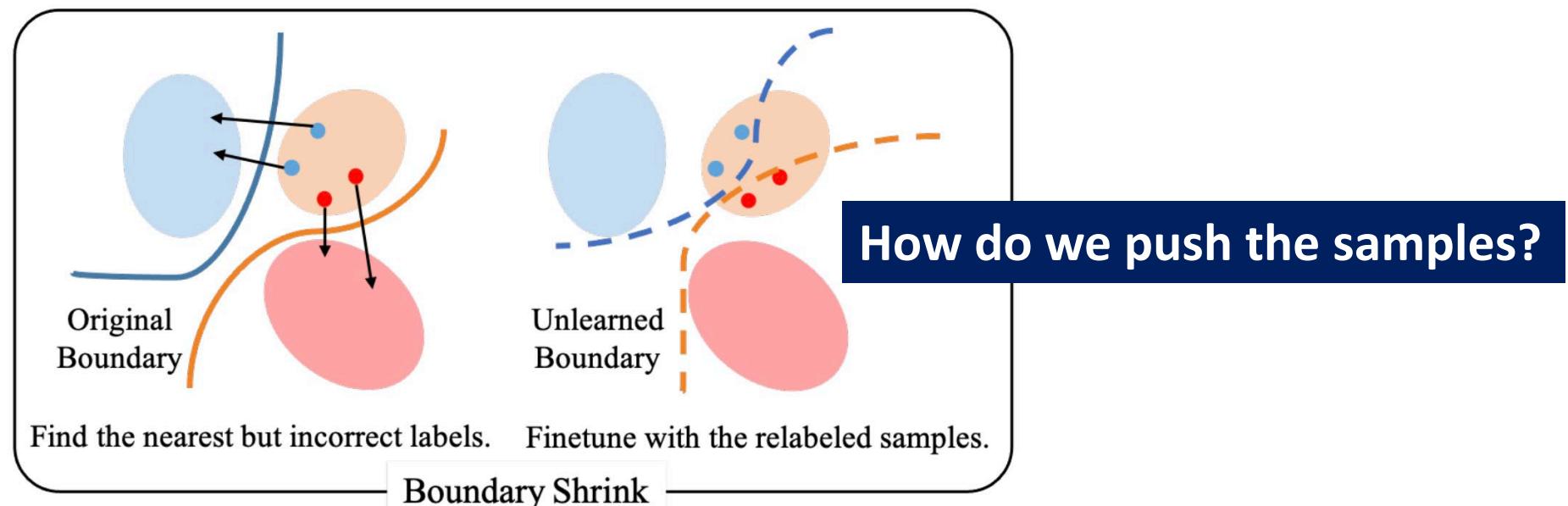
The predicted label on the forgetting samples is different from the unlearning label  $y$

$$\operatorname{argmax}_k f_{\boldsymbol{\theta}_u}^k(\mathbf{x}_f) \neq y \text{ for } \forall \mathbf{x}_f \in \mathcal{D}_f,$$

where  $f^k$  indicates the logit output on the  $k$ -th class

# Boundary Shrink (1)

- Finetuning with random labels on forgetting samples will make them too conspicuous, since they are predicted with **excessive uncertainty**.
- Instead, **Boundary Shrink** pushes the forgetting samples close to the new decision boundary, making the predictions on these forgetting samples with **low certainty**.



# Boundary Shrink (2)

- (a) Given an initial forgetting sample  $\mathbf{x}_f$ , update its corrupted example by neighbor searching with a noise bound  $\epsilon$ :

$$\mathbf{x}'_f = \mathbf{x}_f + \epsilon \cdot \text{sign} \left( \nabla_{\mathbf{x}_f} \mathcal{L}(\mathbf{x}_f, y, \boldsymbol{\theta}_o) \right)$$

$$\boxed{\text{sign}(x) := \begin{cases} -1 & \text{if } x < 0, \\ 0 & \text{if } x = 0, \\ 1 & \text{if } x > 0. \end{cases}}$$

- (b) Get ***nearest but incorrect (nbi)*** labels  $\mathbf{y}_{nbi}^f$ :

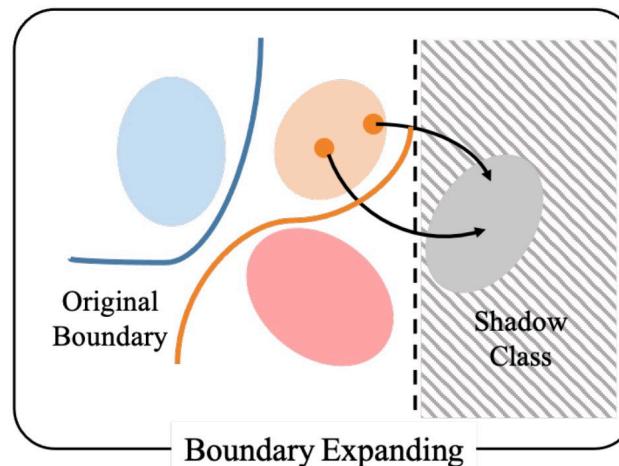
$$\mathbf{y}_{nbi}^f \leftarrow \text{softmax}(f_{\boldsymbol{\theta}_o}(\mathbf{x}'_f))$$

- (c) Finetune the original model with  $\mathbf{y}_{nbi}^f$  for forgetting samples  $\mathbf{x}_f$ :

$$\boldsymbol{\theta}_u = \text{argmin}_{\boldsymbol{\theta}} \left( \sum_{(\mathbf{x}_f, \mathbf{y}_{nbi}^f) \in \mathcal{D}_f} \mathcal{L}(\mathbf{x}_f, \mathbf{y}_{nbi}^f, \boldsymbol{\theta}) \right)$$

# High-level Intuition of Boundary Expanding

- Recall: in previous observation, most forgetting samples move to the border of other clusters on the retrained model. Therefore, they are predicted as other classes with **low certainty**.
- **Low certainty** represents the **output vector is more even**.
- **Boundary Expanding** assigns all forgetting samples to an **extra shadow class** of the original model, which will exploit a new area in the decision space.



# Boundary Expanding

- The model is finetuned with the forgetting samples  $\mathbf{x}_f$  and the shadow class label  $y_{shadow}$ :

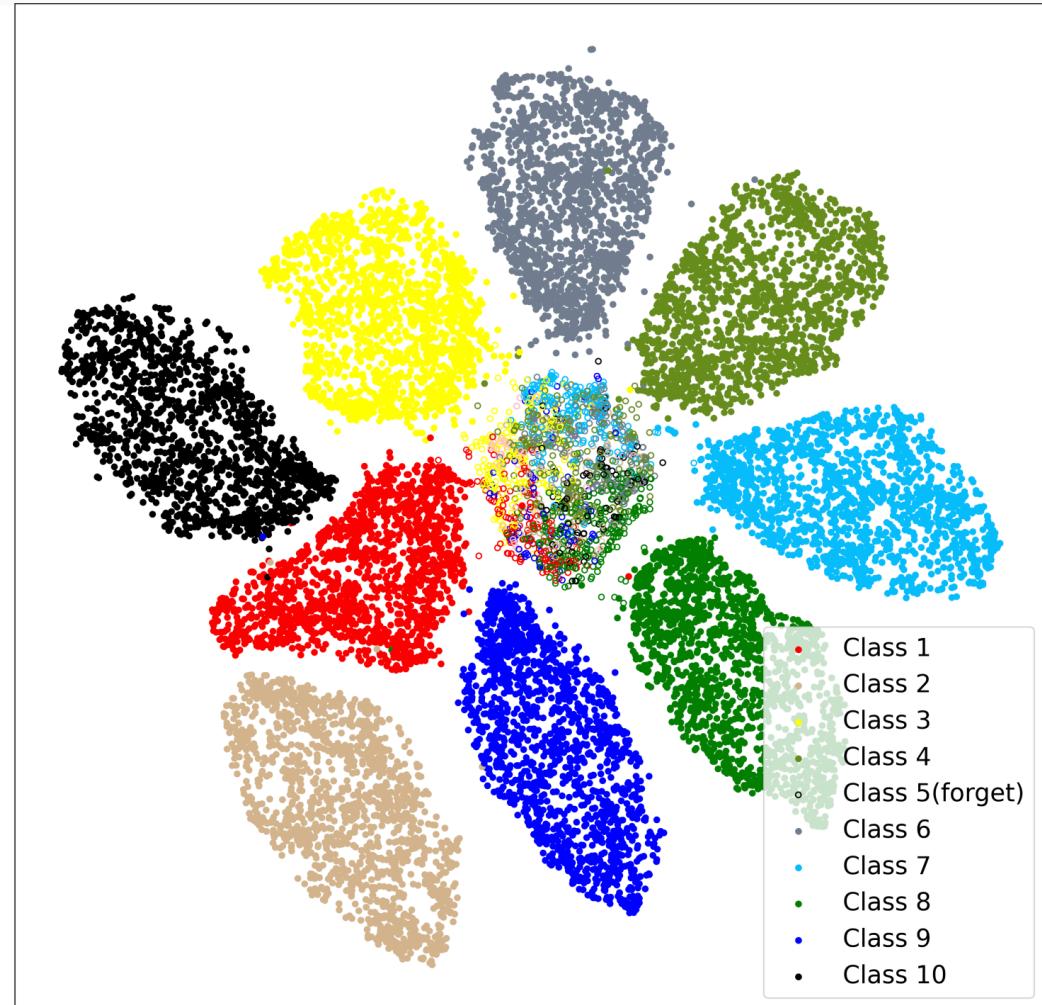
$$\theta_u = \operatorname{argmin}_{\theta} \sum_{(\mathbf{x}_f, y_{shadow}) \in \mathcal{D}_f} \mathcal{L}(\mathbf{x}_f, y_{shadow}, \theta)$$

an additional neuron is added at the last layer of the original model for  $y_{shadow}$

- The logit output  $f_{\theta_u}(\mathbf{x}_f)$  for forgetting sample  $\mathbf{x}_f \in \mathcal{D}_f$  tends to be **low** and **even** on the original classes => disperse in the decision space of the retrained model.

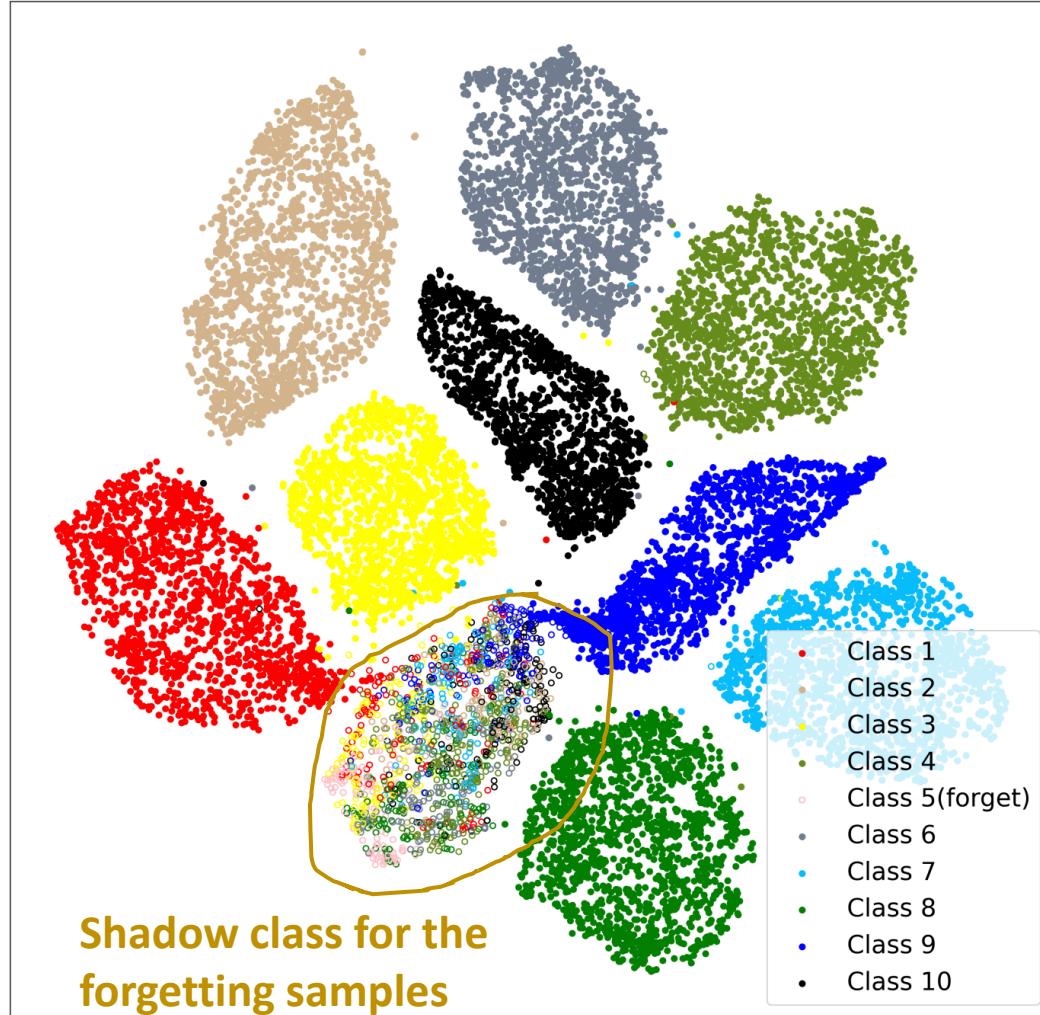
Use the logit output on the original classes during inference stage.

# Decision Space (After Boundary Shrink)



1. the forgetting data (hollow circles) are predicted **as the nearest classes** after applying Boundary Shrink.
2. Some hollow circles move to other clusters, indicating that **the decision space of the forgetting class is split** by its near classes.
3. The clusters of remaining classes still keep compact.

# Decision Space (After Boundary Expanding)



1. The cluster of the forgetting samples is pushed away from the center, creating a **shadow class**.
2. The clusters of remaining classes are maintained.

# Content

- Machine Unlearning
  - Introduction and Motivation
  - Unlearning Definition
  - Model-agnostic Approach
    - Boundary Unlearning
- Model-intrinsic Approach
  - Projective Residual Update for Linear Models
  - Forget-Me-Not for Attention Model
  - SalUn for Diffusion Model
- Data-driven Approach
  - SISA Training

# Forget-Me-Not: Concept Forgetting

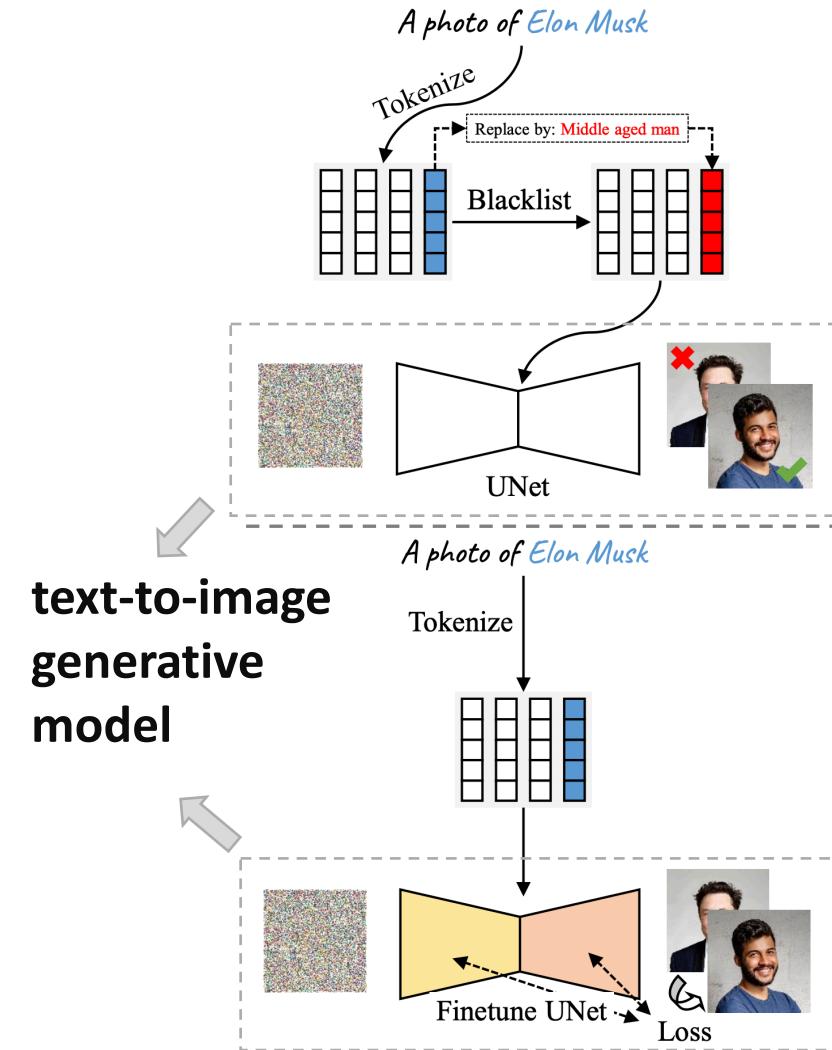
- A **concept** is an abstract term representing an intuited object of thought, which also serves as the foundation for people's perceptions.
- **Forget-Me-Not** is an efficient, plug-and-play method for concept forgetting and correction.

**Concept Forgetting:**  
target concepts are removed and forgotten without compromising the output quality (*Van Gogh* in this case)



**Concept Correction & Disentangle:** correct a dominant or undesired concept in a prompt

# Naïve Forgetting



**Target concept to forget: Elon Musk**

**(a) Token Blacklist:** simply replace the target tokens with a different one.

**Downsides:**

- Does not achieve good forgetting effect.
- May inadvertently affect other concepts that share the same/similar prompt.

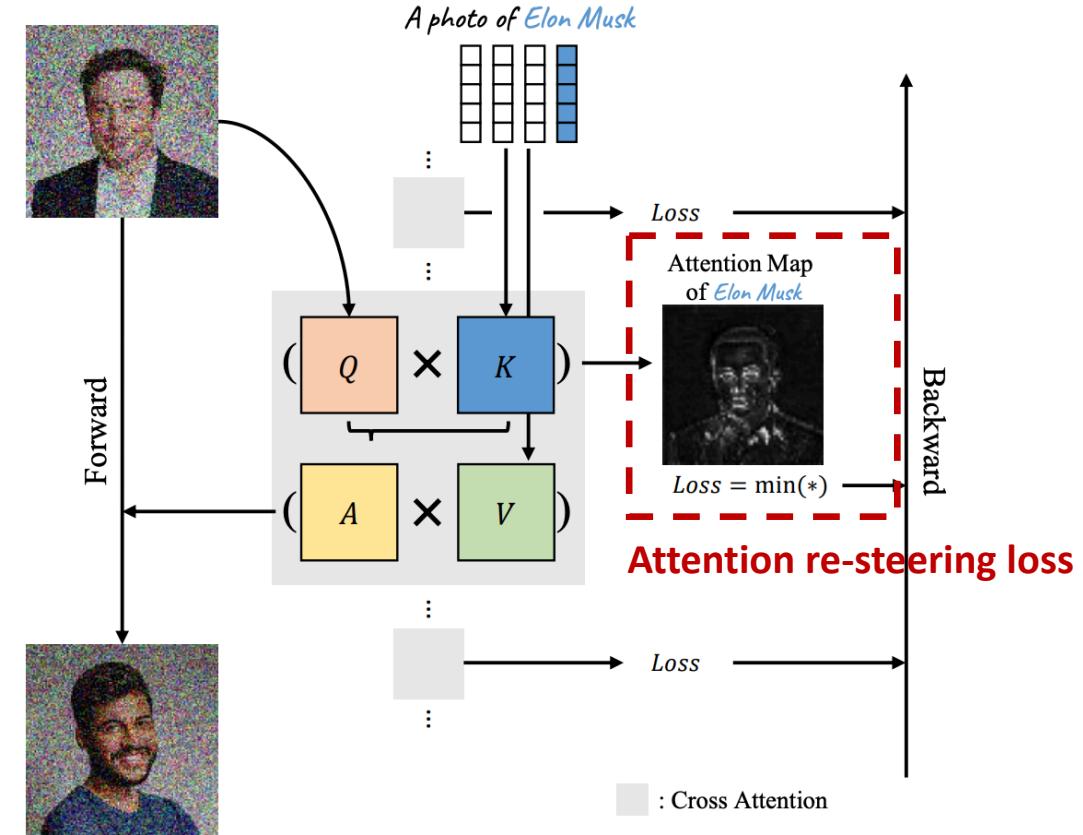
**(b) Naïve Finetuning:** finetune model weights such that the new weights generate outputs with unrelated concepts

**Downside:**

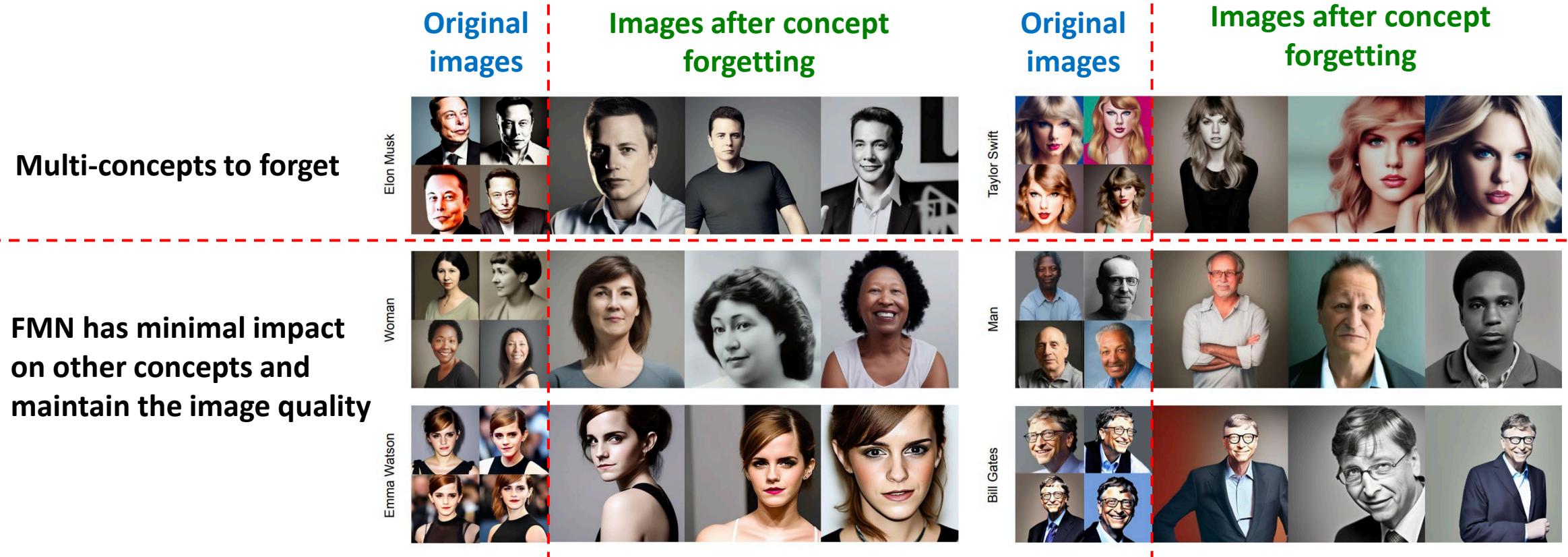
- It breaks model integrity by simultaneously corrupting other unrelated concepts during its finetuning process.

# FMN: Attention Re-steering

- **Forget-Me-Not (FMN)** is a **model-intrinsic** approach for all **attention-based text-to-image generative models**.
- **Attention Resteering:**
  - **Locate** the part of text embeddings associated with the forgetting concepts.
  - **Compute** the attention maps between input features and these embeddings
  - **Minimize** the attention maps and **backpropagate** the network



# Results on Multi-Concept Model



FMN (1) successfully removes the concepts of Elon Musk and Taylor Swift, (2) while maintaining the image quality with unrelated concepts.

# Content

- Machine Unlearning
  - Introduction and Motivation
  - Unlearning Definition
  - Model-agnostic Approach
    - Boundary Unlearning
- **Model-intrinsic Approach**
  - Projective Residual Update for Linear Models
  - Forget-Me-Not for Attention Model
  - **SalUn for Diffusion Model**
- Data-driven Approach
  - SISA Training

# SalUn: Saliency-based Unlearning

- **Motivation of SalUn:** utilize weight saliency to identify model weights that are sensitive to the forgetting data/class/concept.
- SalUn is a **model-intrinsic** approach for **diffusion-based models** on image tasks (e.g., image classification and image generation).
- recall the notations:
  - $\mathcal{D} = \{\mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^N$  is the original  $N$  data points with data feature  $\mathbf{x}_i$  and label  $\mathbf{y}_i$ .
  - $\mathcal{D}_f \subseteq \mathcal{D}$  denotes the forgetting dataset and  $\mathcal{D}_r = \mathcal{D} \setminus \mathcal{D}_f$  is the remaining dataset.
  - Original model trained on  $\mathcal{D}$  is parameterized as  $\boldsymbol{\theta}_o$
  - Unlearned model from  $\boldsymbol{\theta}_o$  on  $\mathcal{D}_f$  or  $\mathcal{D}_r$  is parameterized as  $\boldsymbol{\theta}_u$

# Diffusion Process

- SalUn is specifically designed for diffusion-based model
  - Let  $\theta$  denote the parameters of the diffusion generator, conditioned on the text prompt/concept  $c$ .  $\mathbf{x}_t$  is the latent feature at the diffusion step  $t$ .
  - A diffusion model (DM) training uses MSE loss:
- Main idea: Decompose the original model weights  $\theta_o$  into two distinct components:
  - The **salient** model weights to be updated during machine unlearning process
  - The **intact** model weights that remain unchanged

$$\ell_{\text{MSE}}(\theta; \mathcal{D}) = \mathbb{E}_{\mathcal{D}} \mathbb{E}_{t, \epsilon \sim \mathcal{N}(0, 1)} [\|\epsilon - \boxed{\epsilon_{\theta}(\mathbf{x}_t \mid c)}\|_2^2]$$

Diffusion generator

**Question: how to detect the salient model weights?**

# Forgetting Loss

- SalUn uses the **gradient of a forgetting loss** as the weight saliency map
- Forgetting loss usually follows the training loss on the forgetting dataset  $\mathcal{D}_f$ , denoted as  $\ell_f(\boldsymbol{\theta}; \mathcal{D}_f)$ 
  - For classification:  $\ell_f(\boldsymbol{\theta}; \mathcal{D}_f) = \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{D}_f} [\ell_{\text{CE}}(\boldsymbol{\theta}; \mathbf{x}, y)]$ ; (CE: cross-entropy)
  - For generation:  $\ell_f(\boldsymbol{\theta}; \mathcal{D}_f) = \ell_{\text{MSE}}(\boldsymbol{\theta}; \mathcal{D}_f)$

# Weight Saliency Map

## Weight Saliency Map Calculation:

Reflects the weight sensitivity  
to the forgetting instance in  $\mathcal{D}_f$

$$\mathbf{m}_s = \mathbf{1}\left(\left|\nabla_{\boldsymbol{\theta}}\ell_f(\boldsymbol{\theta}; \mathcal{D}_f)|_{\boldsymbol{\theta}=\boldsymbol{\theta}_0}\right| \geq \gamma\right)$$

- $\mathbf{1}(g \geq \gamma)$  an element-wise indicator function that yields binary outputs.  $\gamma > 0$  is a hard threshold.

The unlearning model  $\boldsymbol{\theta}_u$  is then updated as:

$$\boldsymbol{\theta}_u = \underbrace{\mathbf{m}_s \odot \boldsymbol{\theta}}_{\text{Salient weights}} + \underbrace{(1 - \mathbf{m}_s) \odot \boldsymbol{\theta}_o}_{\text{Original weights}}$$

↑  
Tunable in the  
unlearning process

↑  
Pre-trained, fixed in the  
unlearning process

# SalUn Training: classification

- **Assign Random image label** to a forgetting data point and fine-tune the salient weights:

$$\min_{\theta} L_{\text{SalUn}}^{(1)}(\theta_u) := \mathbb{E}_{(\mathbf{x}, y) \sim \mathcal{D}_f, y' \neq y} [\ell_{\text{CE}}(\theta_u; \mathbf{x}, y')]$$

- Tune  $\theta$  to update  $\theta_u$  by  $\theta_u = \mathbf{m}_s \odot \theta + (1 - \mathbf{m}_s) \odot \theta_o$
- $y'$  is the random label of the image  $x \in \mathcal{D}_f$ , which is different from the original one
- Random Label Assignment removes the knowledge from  $\mathcal{D}_f$  and maintain the model's performance on unrelated samples.

# SalUn Training: generation

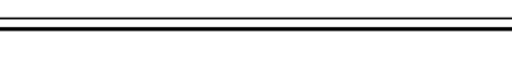
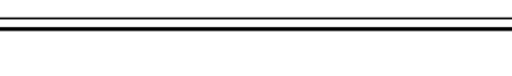
- In diffusion-based generation task, SalUn associates the forgetting concept  $c$  with a misaligned image  $x'$  that does not belong to the concept  $c$ .

$$\begin{aligned} & \underset{\theta}{\text{minimize}} L_{\text{SalUn}}^{(2)}(\boldsymbol{\theta}_u) : \quad \text{Remove the concept knowledge from } \mathcal{D}_f \\ &= \mathbb{E}_{(x,c) \sim \mathcal{D}_f, t, \epsilon \sim \mathcal{N}(0,1), c' \neq c} \left[ \left\| \epsilon_{\boldsymbol{\theta}_u}(x_t \mid c') - \epsilon_{\boldsymbol{\theta}_u}(x_t \mid c) \right\|_2^2 \right] + \alpha \ell_{\text{MSE}}(\boldsymbol{\theta}_u; \mathcal{D}_r) \end{aligned}$$

- $c' \neq c$  indicates that  $c'$  is different from  $c$ .
- $\alpha$  is a regularization parameter that balances **unlearning quality** and **image generation quality** (preserved by  $\ell_{\text{MSE}}(\boldsymbol{\theta}_u; \mathcal{D}_r)$ ).

# Experiments: Image Generation

**Generated Images** under  $I_i$ : forgotten class (airplane) or  $C_i$ : non-forgetting class

Methods	Forgetting class: ‘Airplane’				Non-forgetting classes								
	I1	I2	I3	I4	C1	C2	C3	C4	C5	C6	C7	C8	C9
Random													
SalUn													

- **Random:** apply random weight saliency mask in **SalUn**
- Given a forgetting class, using random weight saliency mask in SalUn generates **noisy images**, leading to over-forgetting performance.
- Given non-forgetting classes, random weight saliency mask **degrades** the generation quality.
- SalUn leverages a proper weight saliency map, leading to better forgetting performance.

# Content

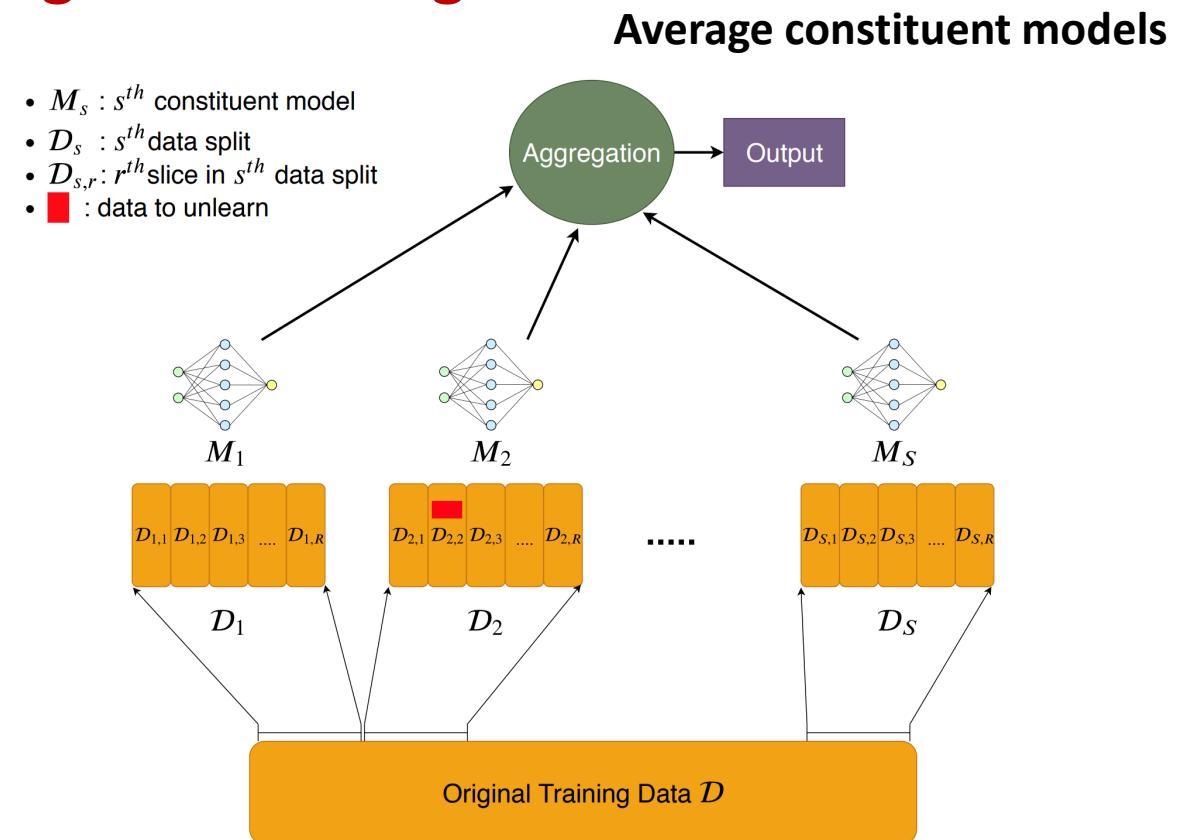
- Machine Unlearning
  - Introduction and Motivation
  - Unlearning Definition
  - Model-agnostic Approach
    - Boundary Unlearning
  - Model-intrinsic Approach
    - Projective Residual Update for Linear Models
    - Forget-Me-Not for Attention Model
    - SalUn for Diffusion Model
- Data-driven Approach
  - SISA Training

# SISA Training

## SISA: Sharded, Isolated, Sliced, and Aggregated Training

### At Training phase:

- Data is divided into **shards**  $D_s$
- Each shard is further split into **slices**  $D_{s,r}$
- Incrementally train the constituent model  $M_s$  by **saving the parameters of  $M_s$  before training it with a new slice.**



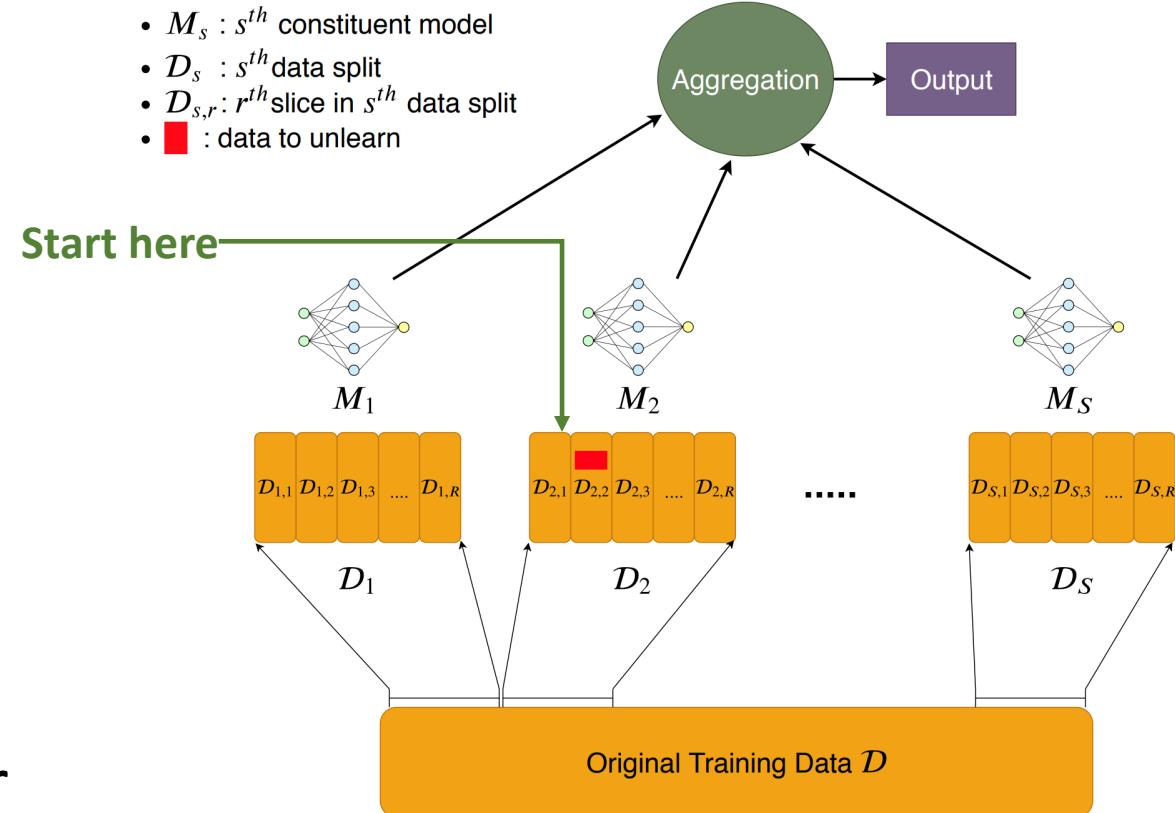
### At Inference phase:

- aggregate the responses of constituent models

# SISA: Unlearning Guarantee

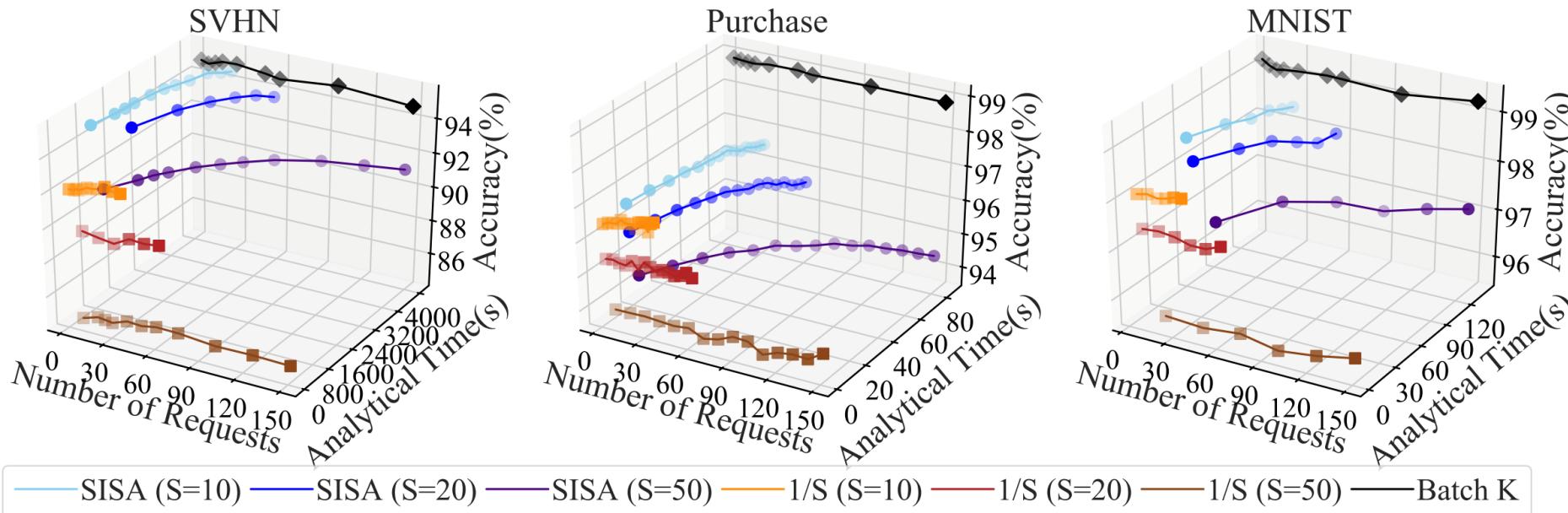
**Limited Retraining Cost** due to the **incremental training process**:

- only the constituent model whose shard contains the point to be unlearned is affected.
- retraining starts from **the last parameter state** saved prior to the slice containing this point.,.
- e.g.,  $D_{2,2}$  is to be unlearned, only retrain  $M_2$  from the state saved after  $D_{2,1}$



# Evaluation: Accuracy & Retraining Time (1)

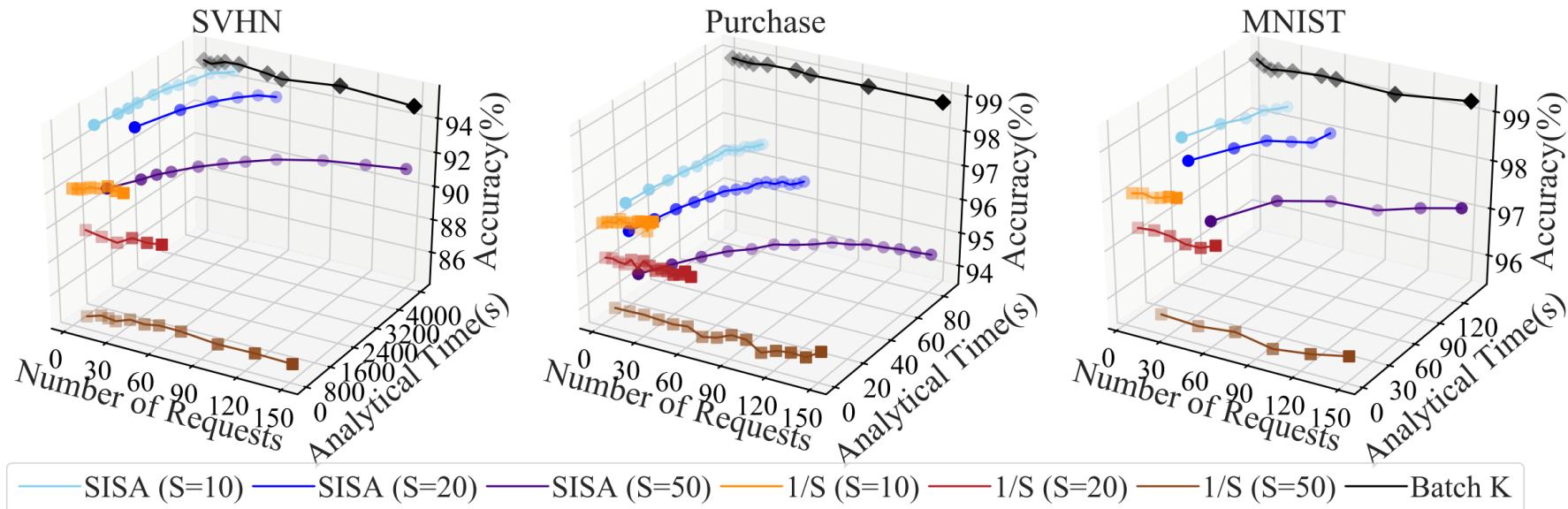
- **Baselines:**
  - Batch  $K$ : **retrain the entire model** after every  $K$  unlearning requests
  - $\frac{1}{S}$ : split the data into  $S$  shards and **only retrain the one that contains the point to be unlearned**
- Dataset: SVHN, Purchase, MNIST
- $S$ : number of shards



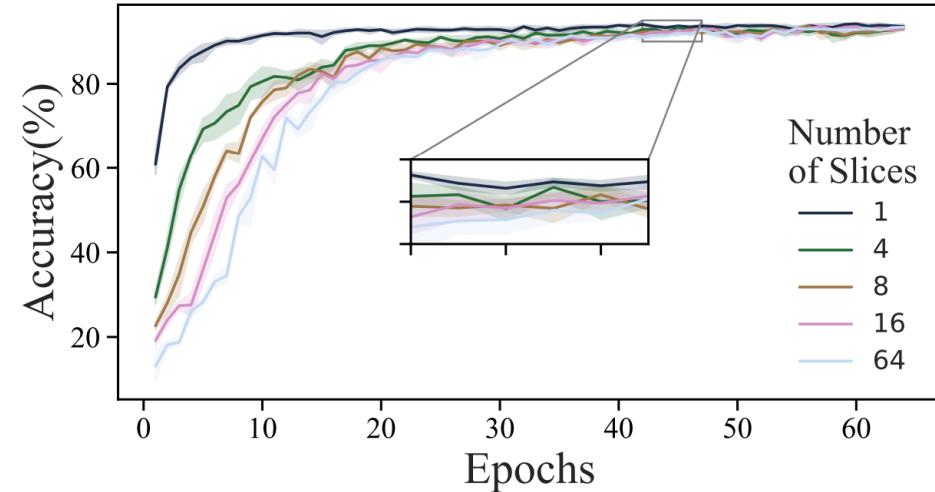
# Evaluation: Accuracy & Retraining Time (2)

- Batch  $K$  achieves **high accuracy**, but needs a **longer retraining time**
- $\frac{1}{S}$  achieves **low accuracy**, and needs a **shorter retraining time**
- SISA achieves the **best trade-off between accuracy and retraining time**
- Increasing  $S$  will degrade the accuracy**

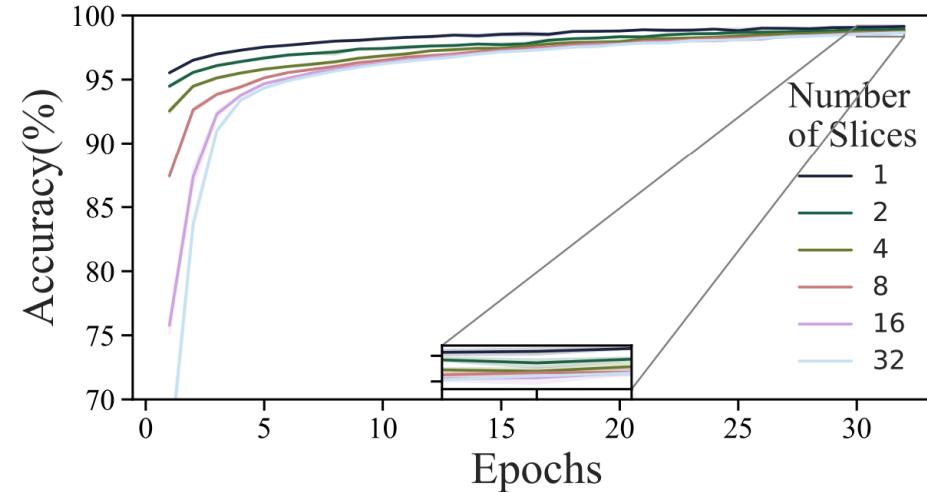
Ensure each shard has sufficiently many data points to ensure high accuracy for each constituent model!



# Evaluation: Impact of Slicing



(a) Accuracy vs. Number of epochs for SVHN dataset.

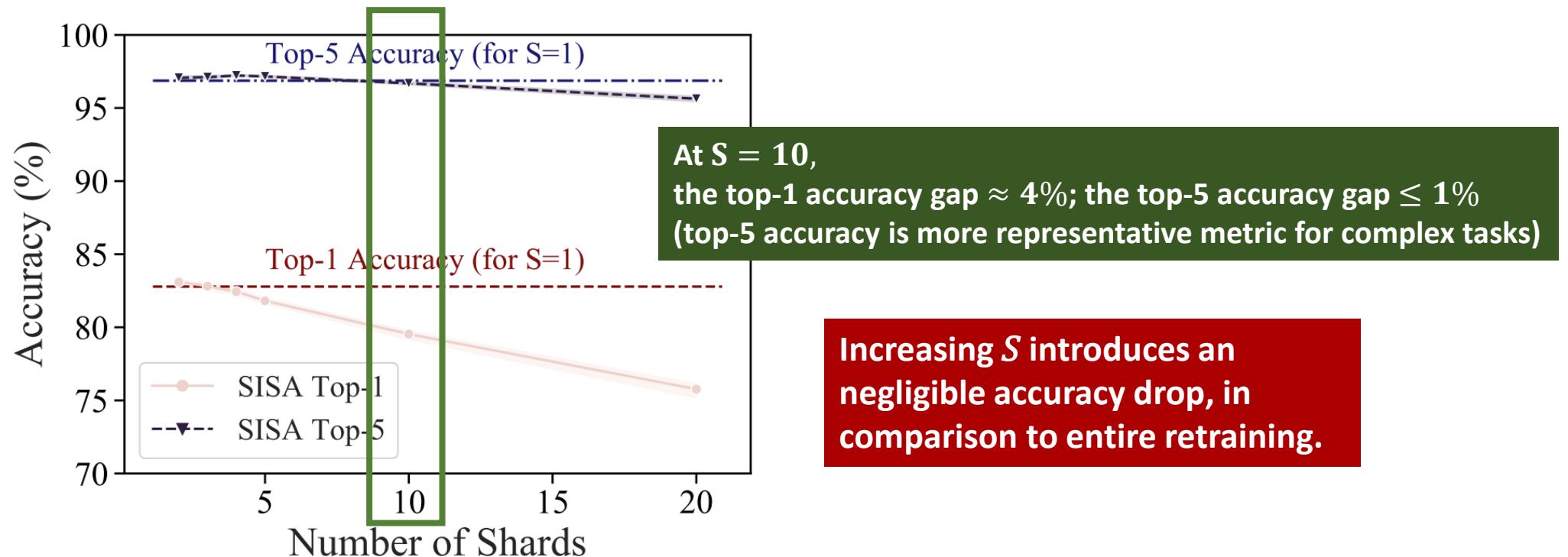


(b) Accuracy vs. Number of epochs for Purchase dataset.

1. **Slicing the shard takes more epochs to achieve high accuracies**
2. **For a small number of epochs, models with more slicing have lower accuracy, due to the fact that they have significantly less amount of data at the beginning.**

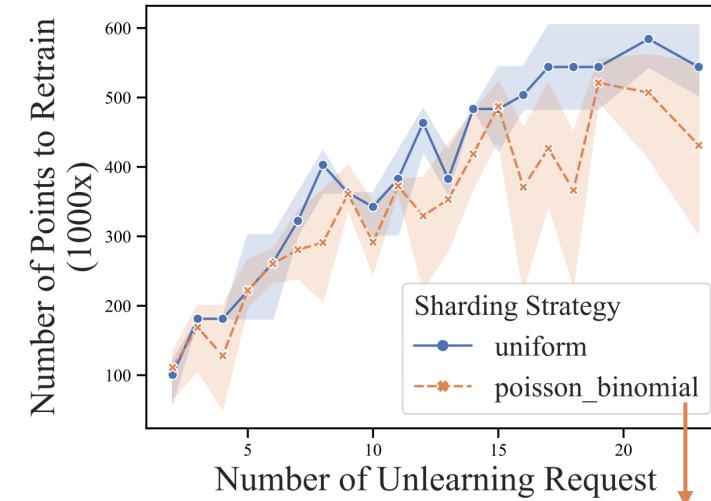
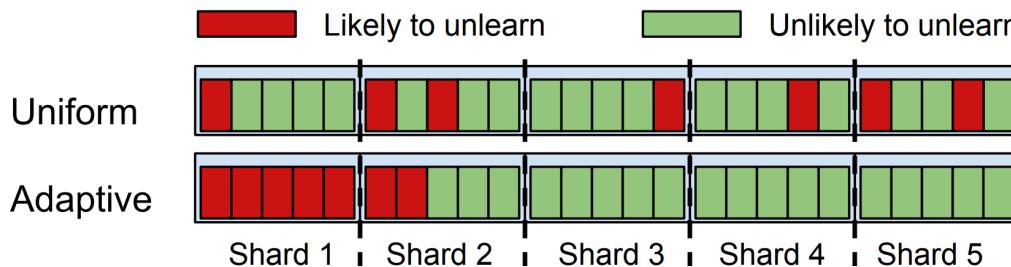
# Evaluation: Transfer Learning / Pretraining

- Train a base ResNet-50 on Imagenet (relatively simple) and transfer it to CIFAR-100 dataset (more complex)
- Baseline:  $S = 1$  (retrain over the entire dataset)



# Adaptive Unlearning

- **A priori knowledge** of users' probability for requesting unlearning can improve SISA unlearning
- **Distribution-aware sharding**: maximize shard size; minimize the chance that a shard has at least one unlearning request



The adaptive sharding can reduce analytical retraining time!

Assume each request is an independent Bernoulli trial, then group of requests follows a Poisson Binomial distribution

# Summary of the Lecture

- Machine Unlearning Motivation: the data/user have the “*right to be forgotten*”
- Goal of Machine unlearning: to design **efficient** unlearning algorithms that forget certain data, class, feature, concept, etc.
- Machine unlearning algorithms should **maintain performance on the unchanged data**
- Machine unlearning methods include **model-agnostic** approaches, **model-intrinsic** approaches and data-driven approaches.