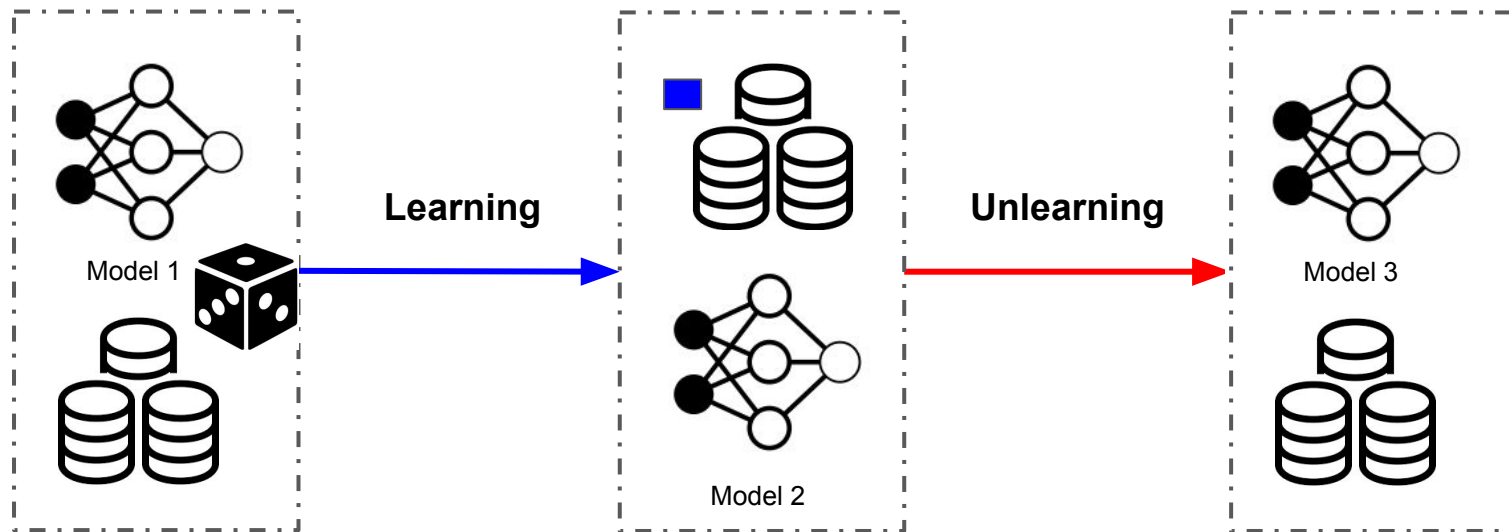




Unlearning



Why do we need unlearning?

Let's see a few observations.

Observation 1: Legislation

New privacy legislation:

- Calls for transparency and clarity of data
- Empowers users to remove their data



PIPEDA
Personal Information
Protection and Electronic
Documents Act

Observation 1: Legislation

No one's ready for GDPR

'Very few companies are going to be 100 percent compliant on May 25th'

By [Sarah Jeong](#) | [@sarahjeong](#) | May 22, 2018, 3:28pm EDT

Can I Opt Out Yet?: GDPR and the Global Illusion of Cookie Control



Authors: [Iskander Sanchez-Rola](#), [Matteo Dell'Amico](#), [Platon Kotzias](#), [Davide Balzarotti](#), [Leyla Bilge](#),
 [Pierre-Antoine Vervier](#), [Igor Santos](#) [Authors Info & Affiliations](#)

A Study on Subject Data Access in Online Advertising After the GDPR

Authors

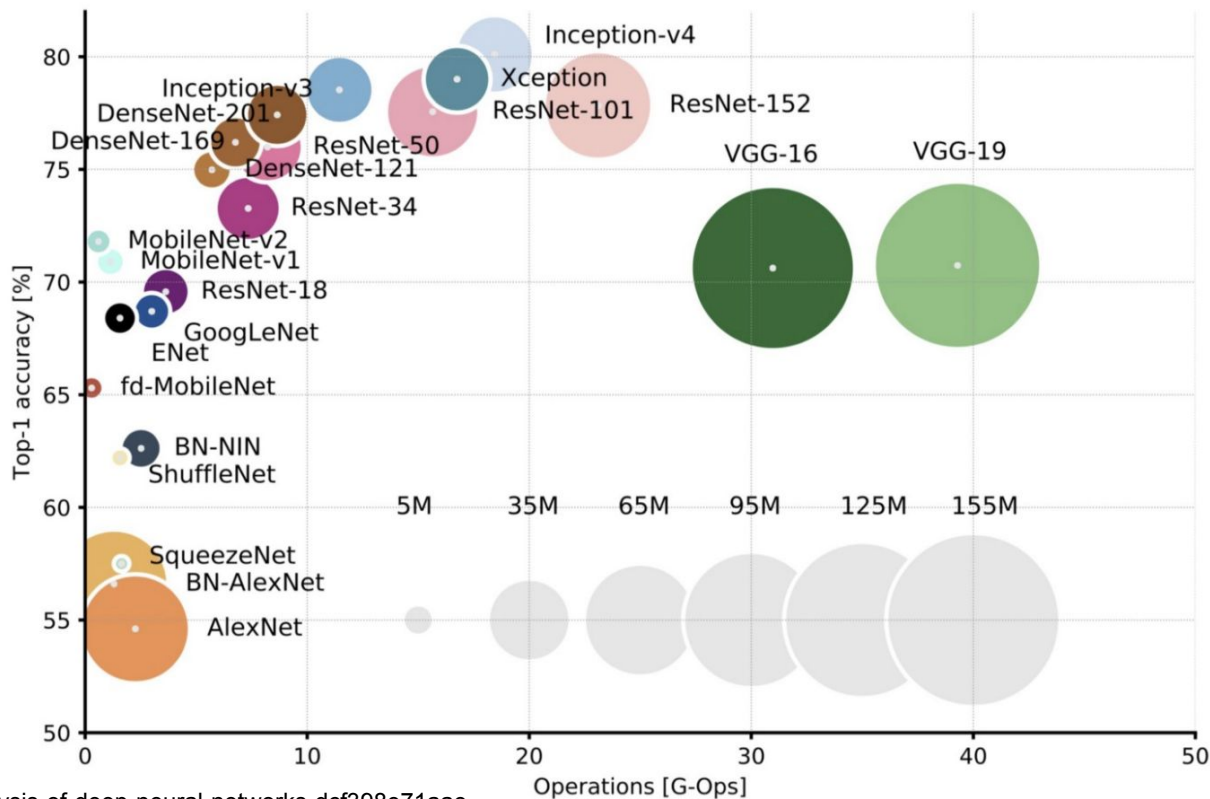
[Authors and affiliations](#)

Tobias Urban , Dennis Tatang, Martin Degeling, Thorsten Holz, Norbert Pohlmann

Observation 1: Legislation

Observation 1: Disconnect between legal experts and technology experts

Observation 2: Large Models



Observation 2: Large Models

Overparameterized Nonlinear Learning: Gradient Descent Takes the Shortest Path?

Samet Oymak¹ Mahdi Soltanolkotabi²

Many modern learning tasks involve fitting non-linear models which are trained in an overparameterized regime where the parameters of the model exceed the size of the training dataset. Due to

The Secret Sharer: Evaluating and Testing Unintended Memorization in Neural Networks

Nicholas Carlini, Chang Liu, Úlfar Erlingsson, Jernej Kos, Dawn Song

Observation 2: Large Models

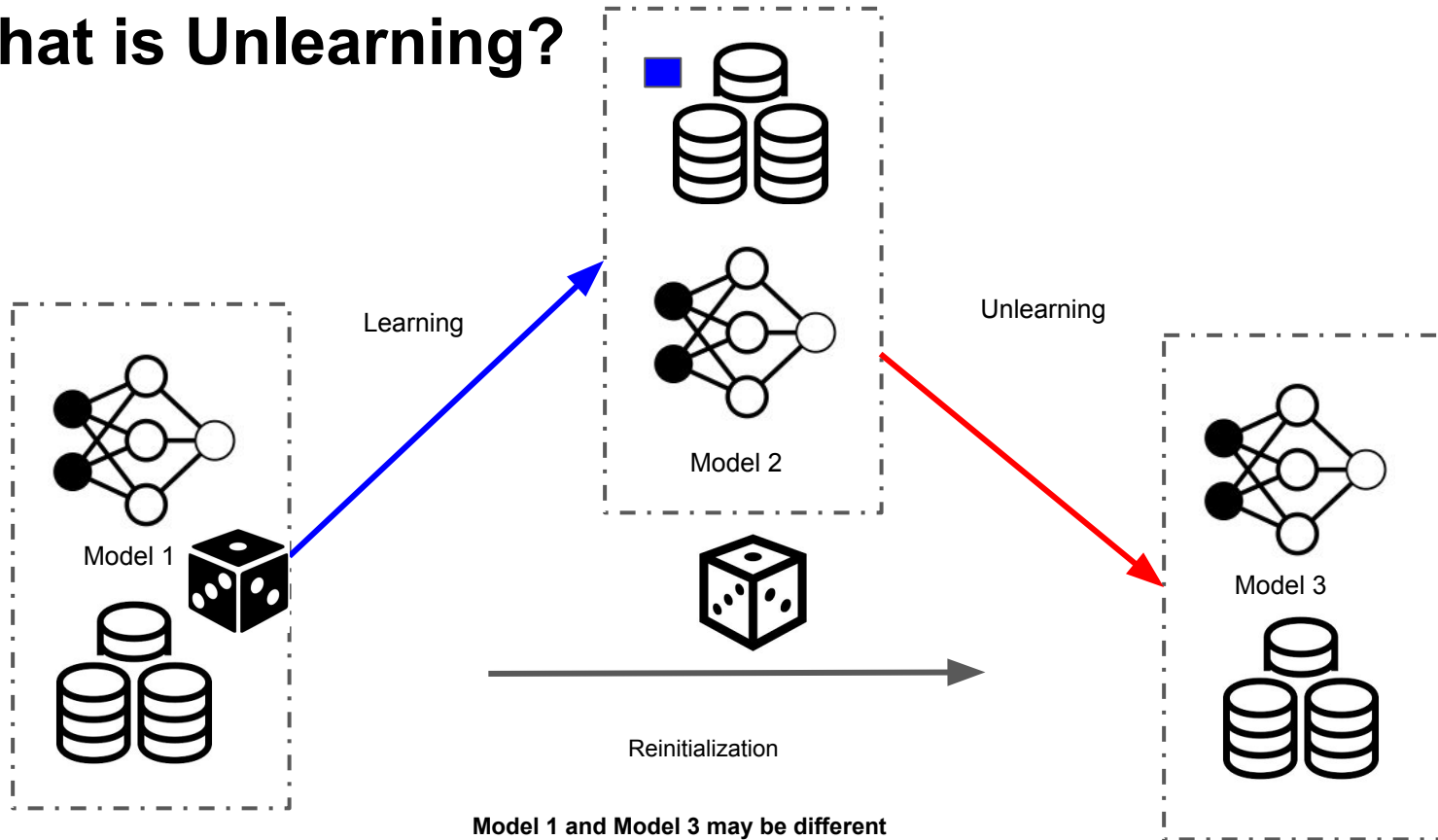
Observation 2: Overparameterization leads to complex interplay between data and model parameters

Right-to-be-forgotten for Machine Learning?

1. Synergy missing between legal and tech experts
2. Complex interplay between data and parameters

Concrete Problem: *Unlearn* data from trained ML models (e.g., DNNs) such that removal guarantee is easy to comprehend, i.e., it is straightforward to see the unlearning is done correctly

What is Unlearning?



What is Unlearning?

1. Distribution of models learnt after learning and then **unlearning a point** should be the **same as** the
2. Distribution of models learnt through **random re-initialization without the point**

Agenda

- **The reason why data providers may require unlearning:**
 - Overlearning Reveals Sensitive Attributes
- **How to apply unlearning to deep neural networks:**
 - Machine Unlearning
- **What if the unlearning requests are not uniformly distributed:**
 - Adaptive Machine Unlearning
- (if there is time left) **Recent observation that unlearning is ill-defined**
 - On the Necessity of Auditable Algorithmic Definitions for Machine Unlearning

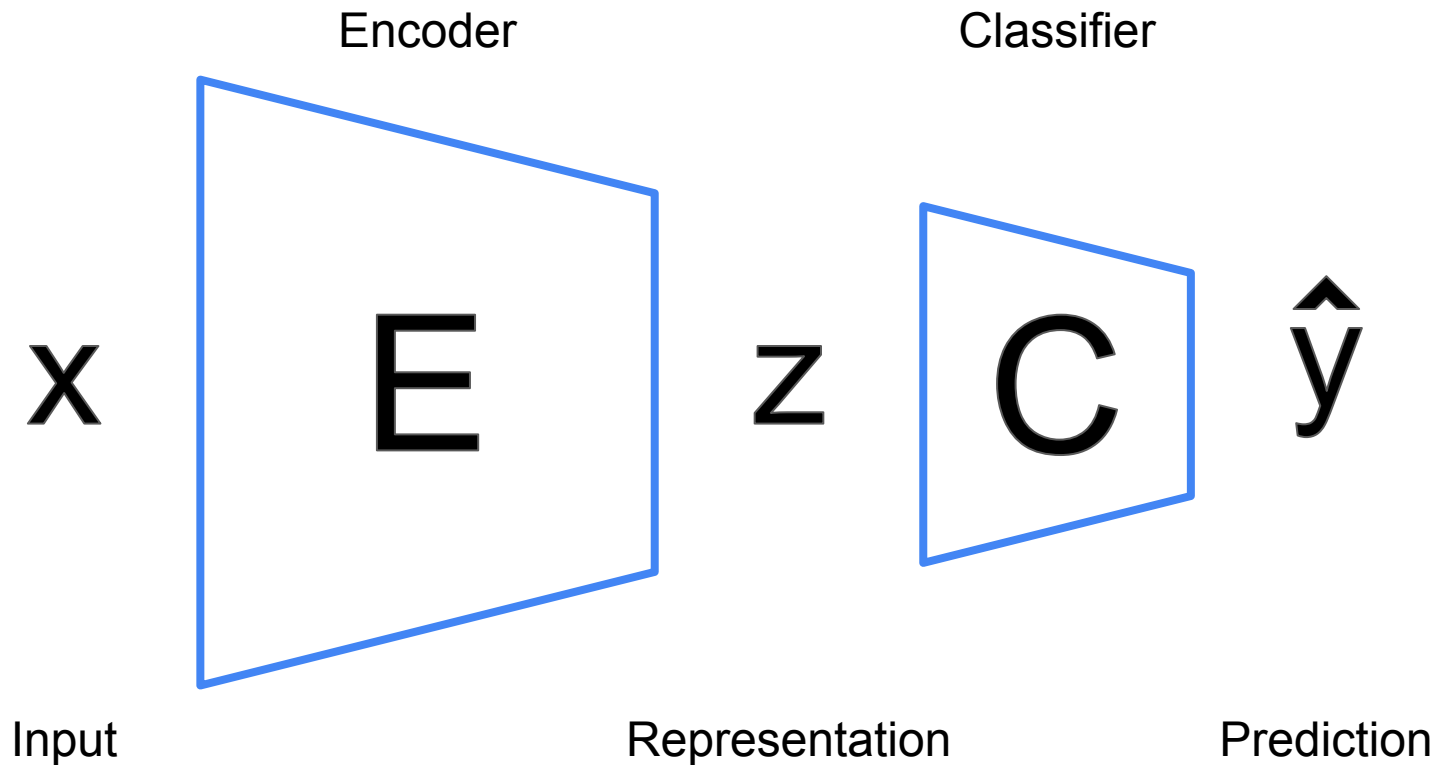
Overlearning Reveals Sensitive Attributes

Congzheng Song, Vitaly Shmatikov

Overlearning

- What is overlearning?
 - Let's say I want a model to learn skating
 - Overlearning is when it learns hockey
 - It learns more than what we want it to learn
- Is this a bad thing?
 - Unfortunately, yes
 - This paper shows what an adversary can do to a overlearned model

Model Partitioning



Inferring Sensitive Attributes from Representations

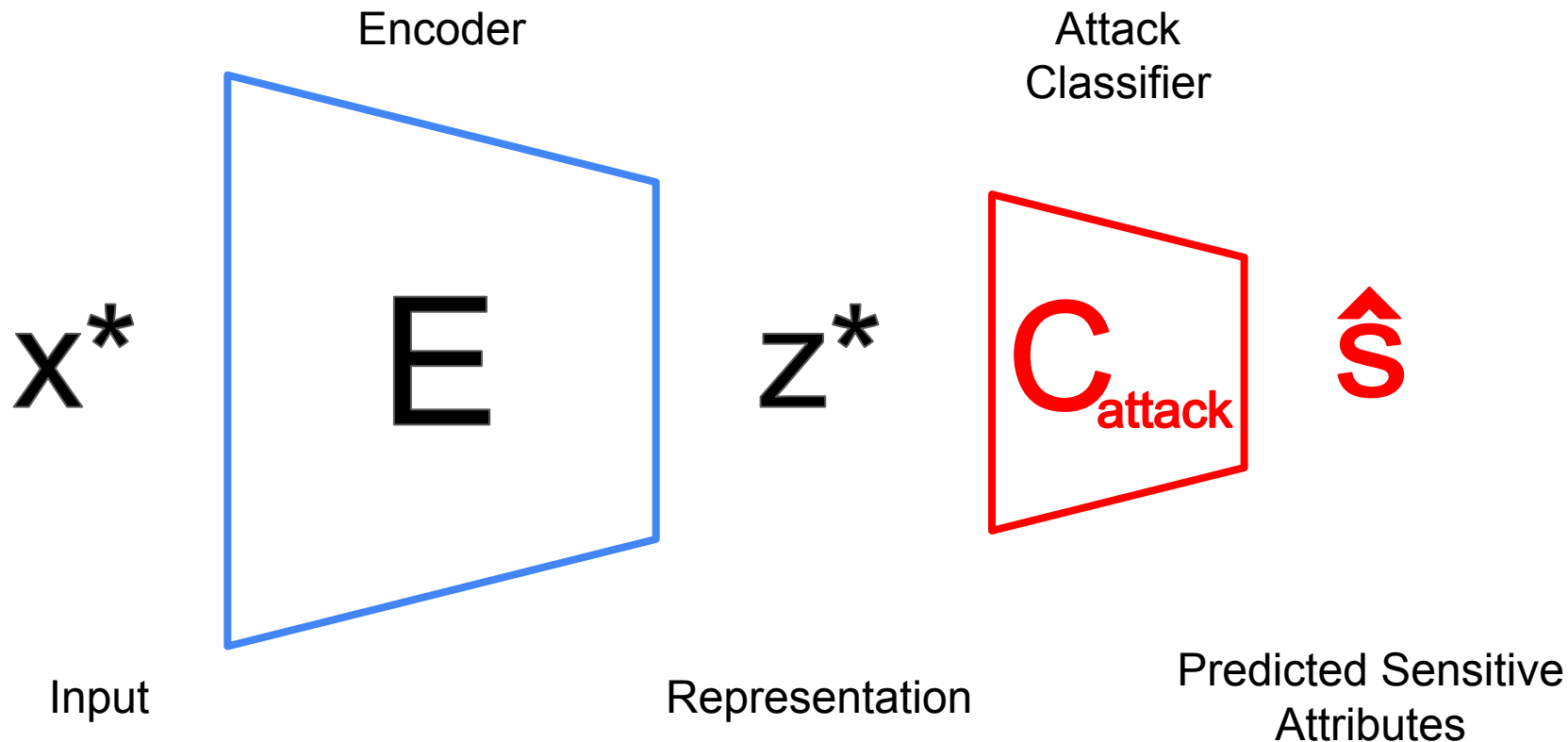
Assumption:

- The adversary has an auxiliary dataset consisting of (x^*, s) pairs
 - in the experiments, the authors used an auxiliary dataset with size equals to 50% of the training dataset
 - s : label for sensitive attributes

Problem Definition:

- If E overlearned, can an adversary infer sensitive attributes from it

Inferring Sensitive Attributes from Representations

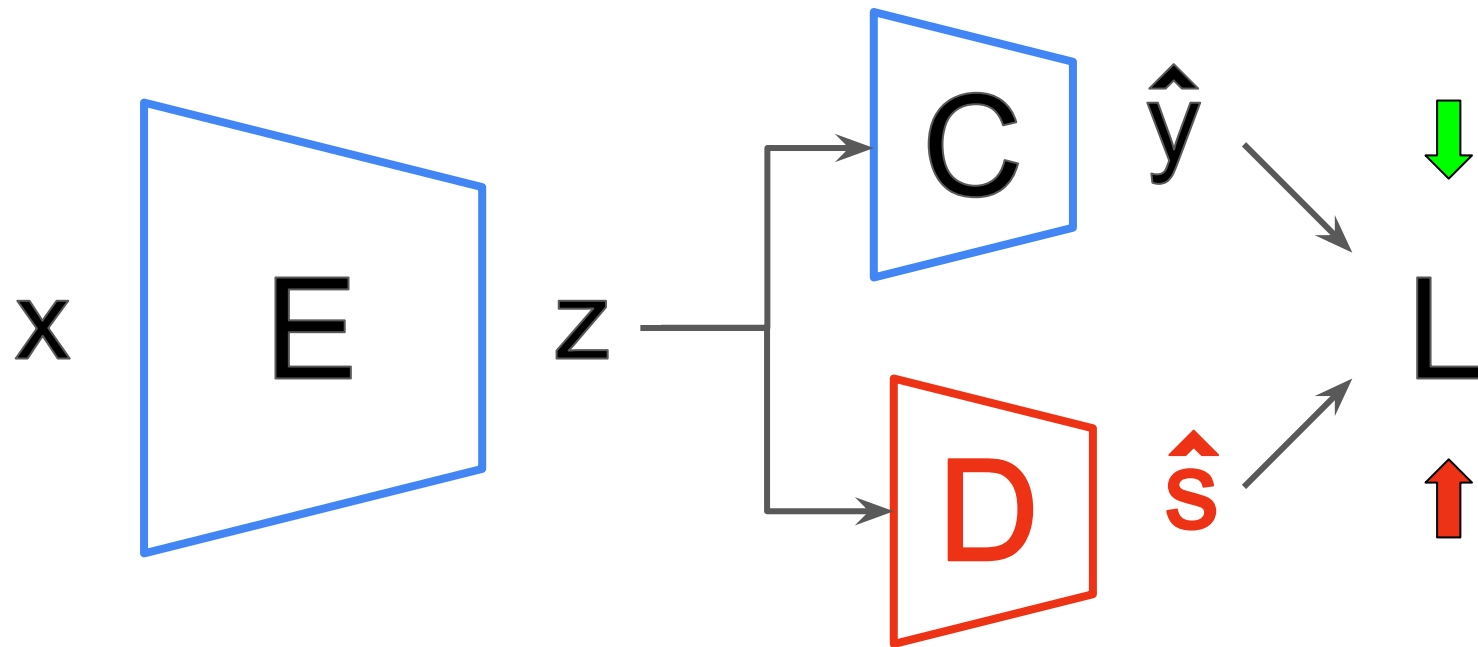


Censoring Representations

Goal:

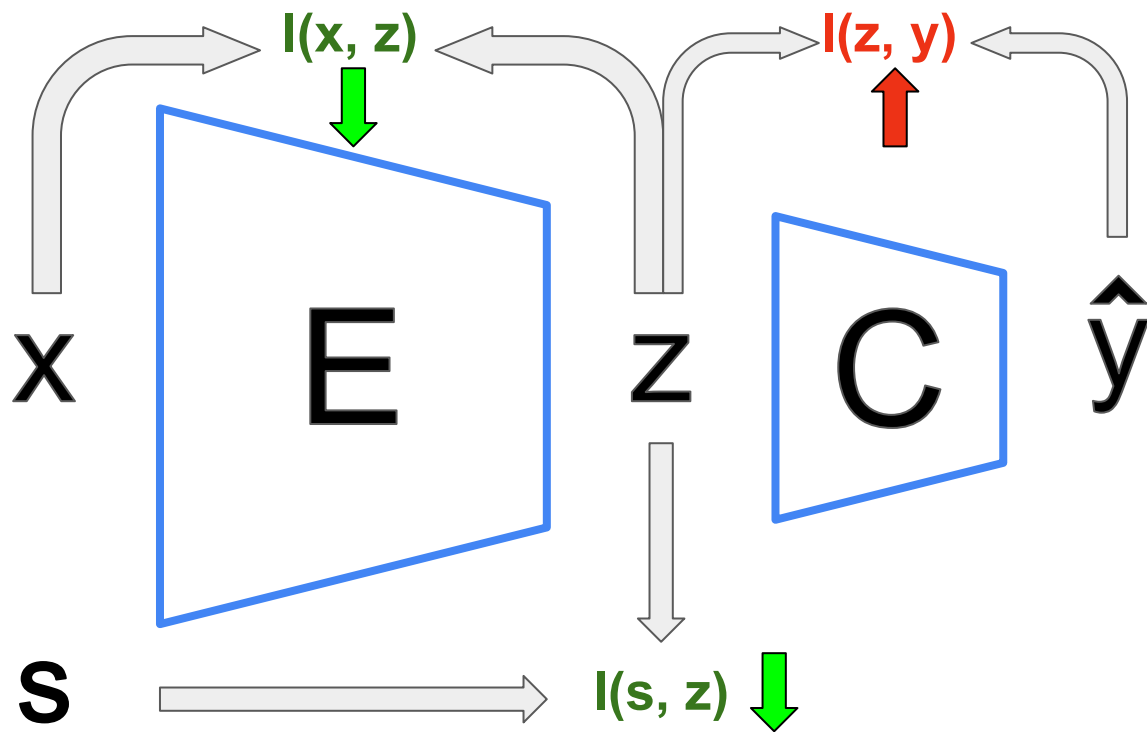
- Not reveal unwanted properties of inputs x
- In other words, prevent an adversary from inferring sensitive attributes of inputs using the representations

Censoring Representations (Adversarial Training)



Discriminator

Censoring Representations (Information-theoretical)



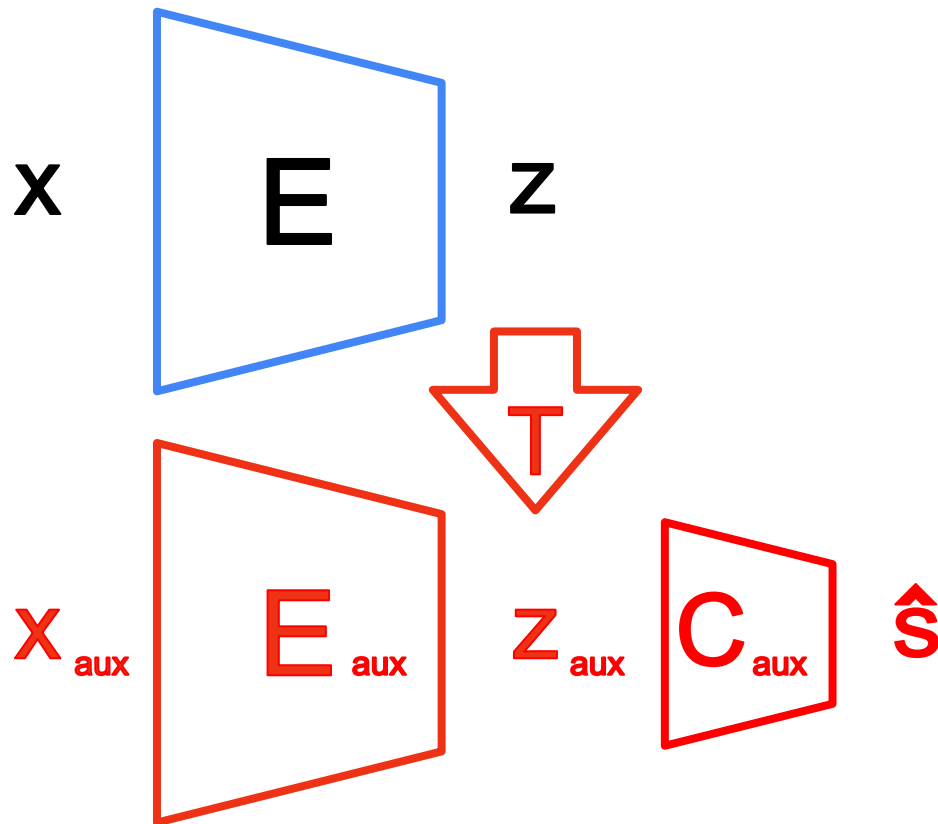
- More effective than adversarial training, but damages task accuracy more

Experimental Results

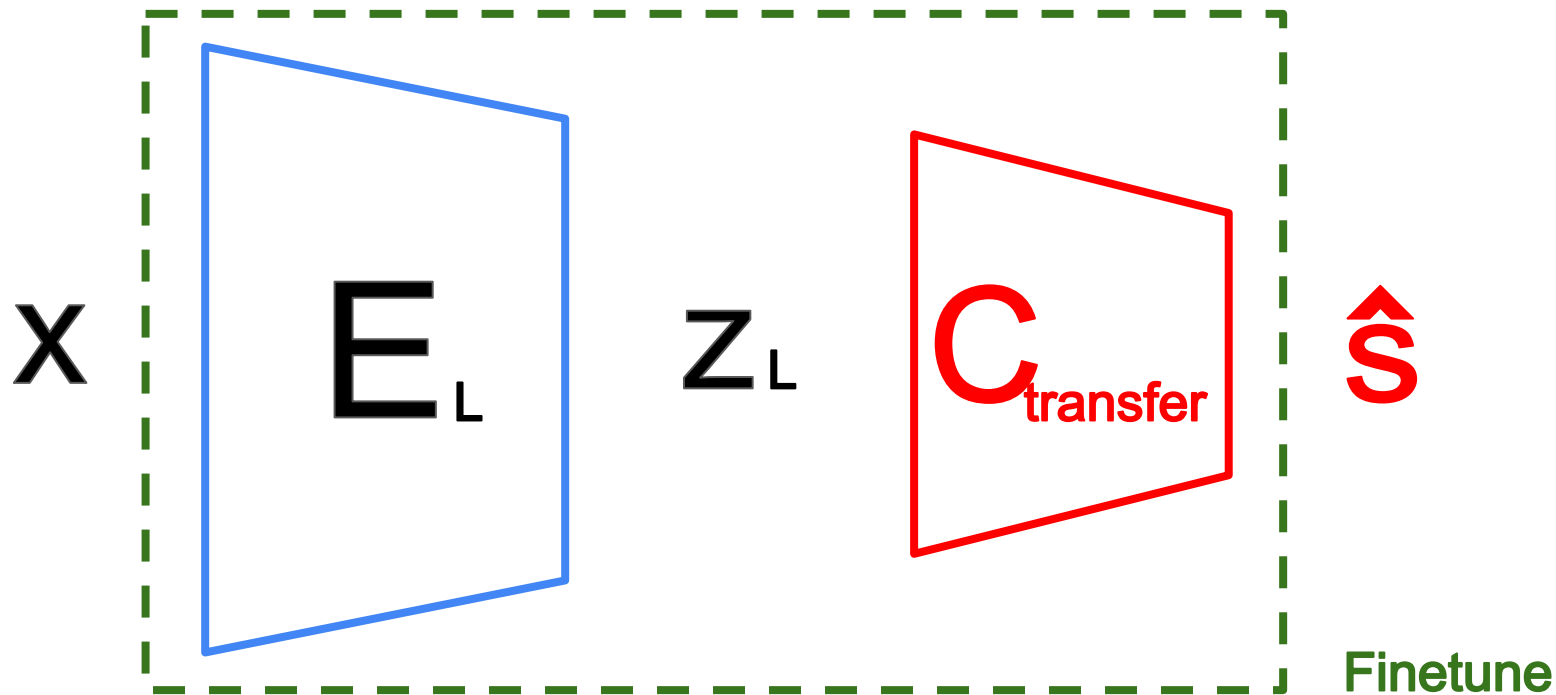
Table 2: Accuracy of inference from representations (last FC layer). RAND is random guessing based on majority class labels; BASE is inference from the uncensored representation; ADV from the representation censored with adversarial training; IT from the information-theoretically censored representation.

Dataset	Acc of predicting target y				Acc of inferring sensitive attribute s			
	RAND	BASE	ADV	IT	RAND	BASE	ADV	IT
Health	66.31	84.33	80.16	82.63	16.00	32.52	32.00	26.60
UTKFace	52.27	90.38	90.15	88.15	42.52	62.18	53.28	53.30
FaceScrub	53.53	98.77	97.90	97.66	1.42	33.65	30.23	10.61
Places365	56.16	91.41	90.84	89.82	1.37	31.03	12.56	2.29
Twitter	45.17	76.22	57.97	n/a	6.93	38.46	34.27	n/a
Yelp	42.56	57.81	56.79	n/a	15.88	33.09	27.32	n/a
PIPA	7.67	77.34	52.02	29.64	68.50	87.95	69.96	82.02

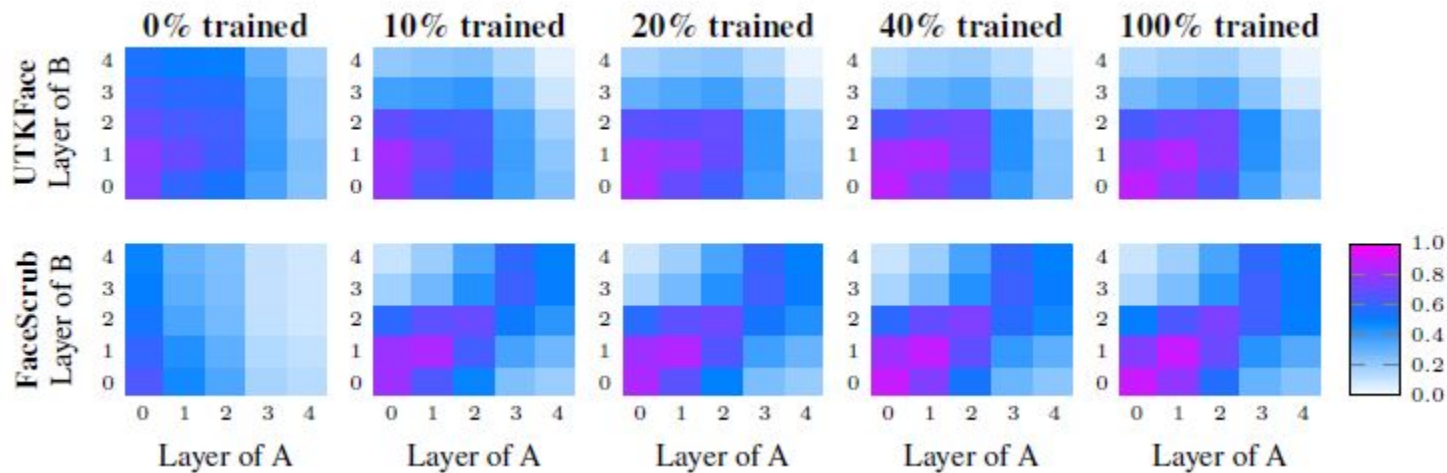
De-censoring Representations



Repurposing models to predict sensitive attributes



Why does overlearning happen



- Early layers learned very general features that can be used for different tasks
- This happens when the training data is sampled from complicated distributions

Limitations of this work

- The attacks introduced in this work have strong assumptions, e.g.,
 - Access to a large labelled dataset, which may be unrealistic in practice
- The two attacks, inferring sensitive attributes and repurposing models, only differ by whether to finetune the encoder

What does overlearning mean to unlearning?

- Recall that the topic for today is unlearning
- Besides both contains the term “learning”, how is overlearning related to unlearning?
- Overlearning implies that despite the training data is kept privately or even erased after training, sensitive information about the data providers can be still memorized and later obtained from the released model.
- This means under regulations like GDPR, data providers can and may want to require the model deployers to stop use their data at any time, in which case the model deployers have to **unlearn**

Machine Unlearning

**Lucas Bourtole*, Varun Chandrasekaran*, Christopher A. Choquette-Choo*, Hengrui Jia*,
Adelin Travers*, Baiwu Zhang*, David Lie, Nicolas Papernot**

**Joint Lead Author*

Naive Solution: Remove data point & retrain model from scratch

Intuitive, Simple to Implement, Interpretable

New problem: Such an approach is **very slow**

Sharded, Isolated, Sliced, and Aggregated Training (SISA)



R check-point

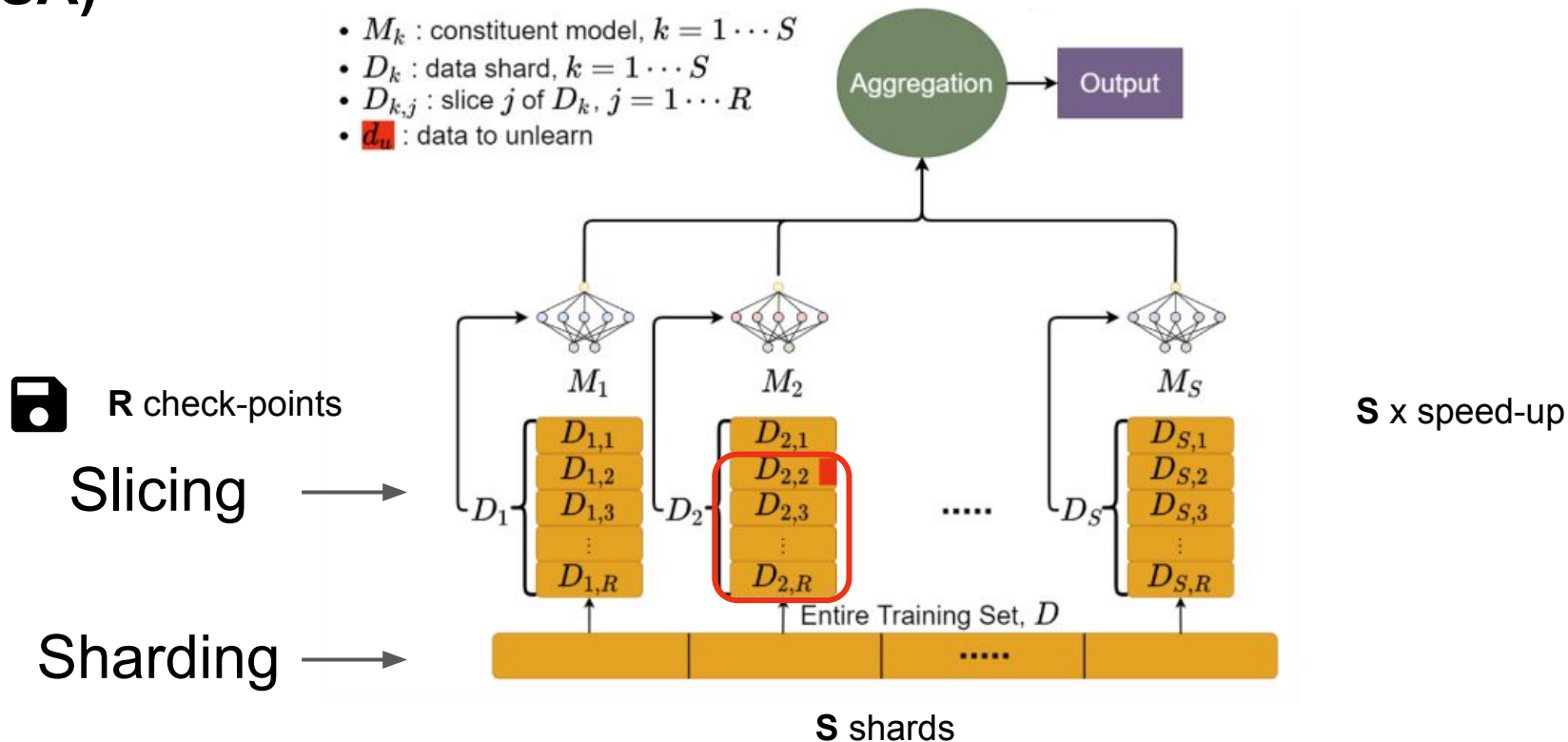
Slicing —




S x speed-up

Sharded, Isolated, Sliced, and Aggregated Training (SISA)

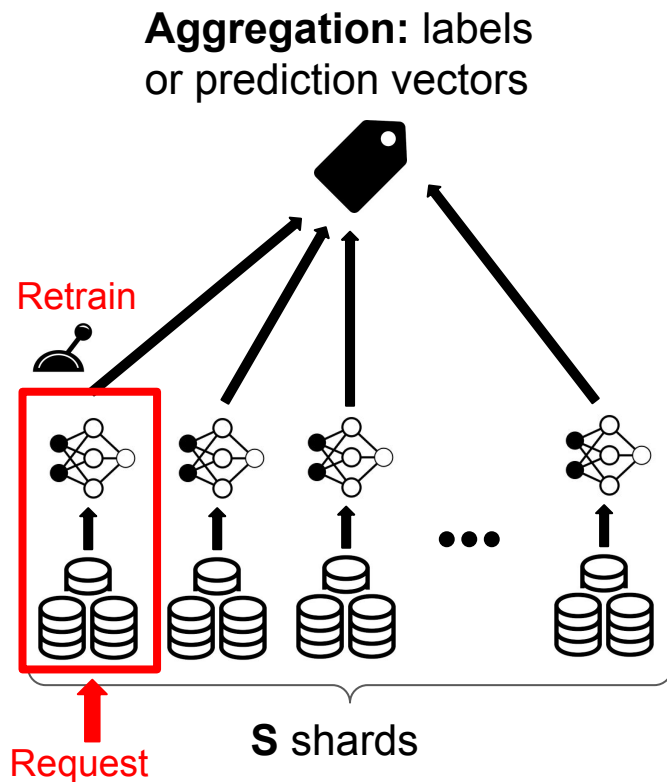
- M_k : constituent model, $k = 1 \dots S$
- D_k : data shard, $k = 1 \dots S$
- $D_{k,j}$: slice j of D_k , $j = 1 \dots R$
- d_{ui} : data to unlearn



Tuneable Knobs

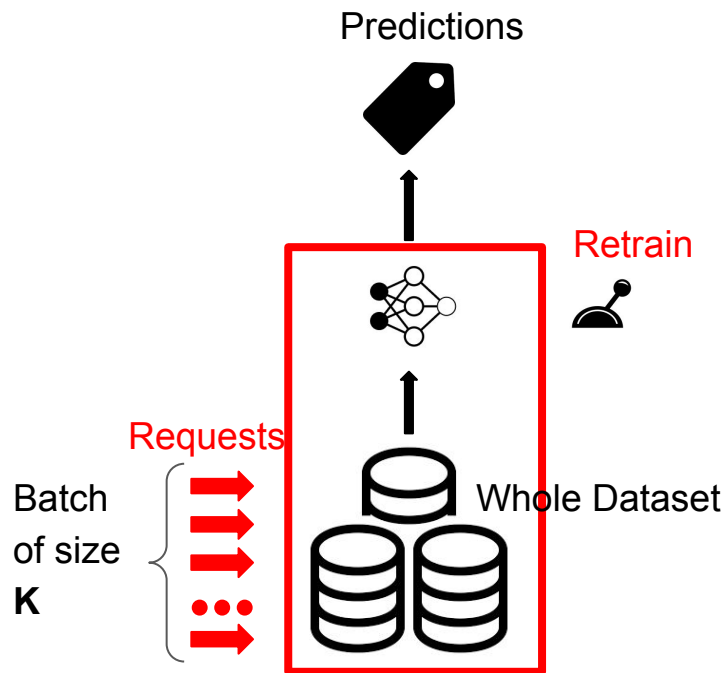
Tuneable Knob	Retraining speed-up	Storage Cost	Accuracy
Sharding			
Slicing			
Aggregation Strategy			

Impact of Sharding: Setup

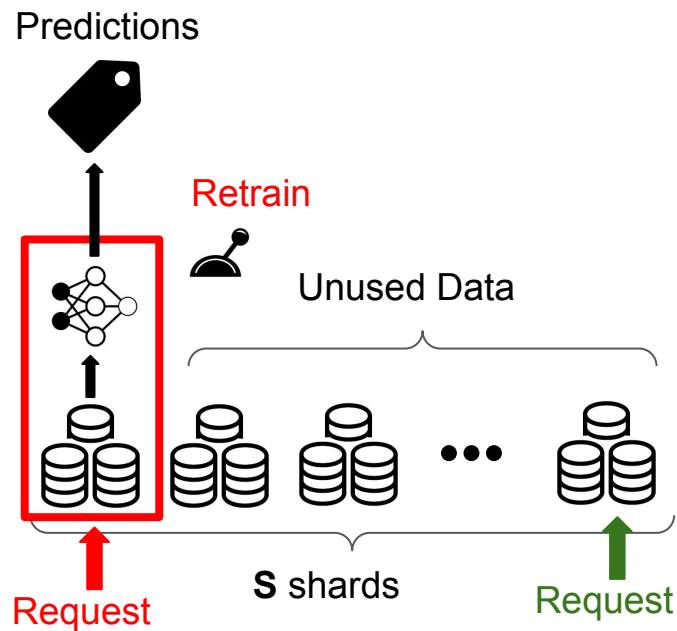


Impact of Sharding: Baselines

Batch K Baseline



$1/S$ Fraction Baseline



Impact of Sharding: Results

S: number of shards

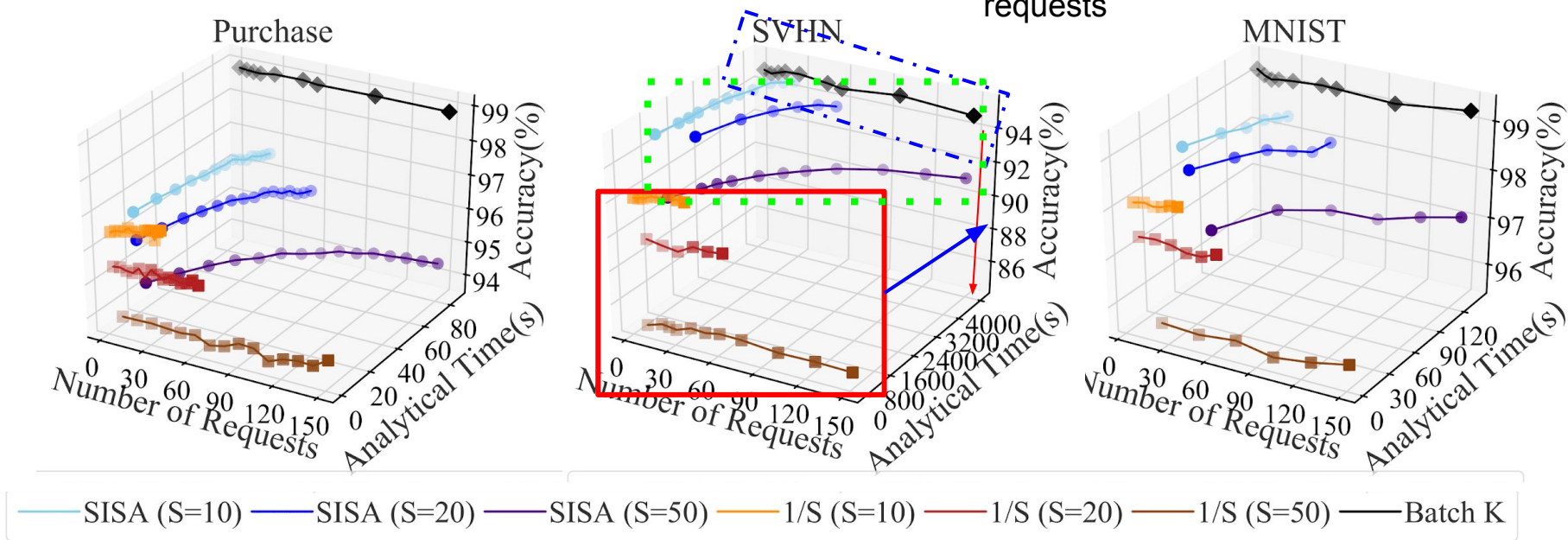
n: number of unlearning
requests

1. Does increasing '**S**' improve retraining speed-up?
2. When does SISA accuracy degrade too much?

Impact of Sharding: Results

S: number of shards

n: number of unlearning requests



1. **Batch K**: High Accuracy; very slow

2. **1/S**: Low Accuracy; very fast

3. **SISA**: Best trade-off

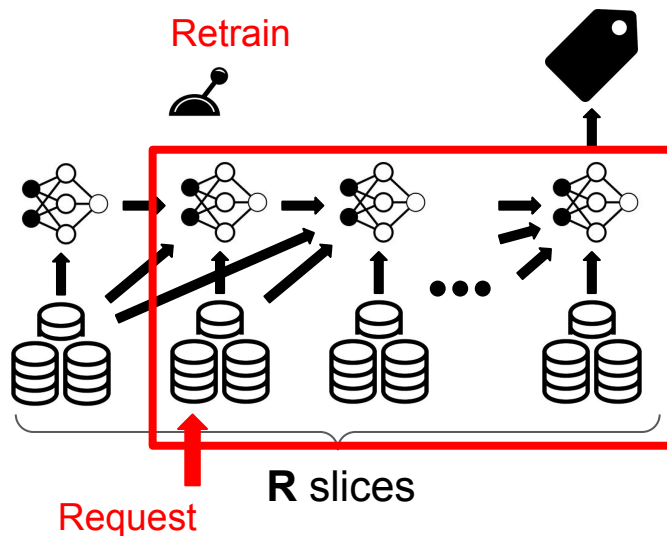
4. **SISA**: Speed-ups exist when $n < 3S$

Impact of Slicing: Setup

Assumption: constant training time by varying number of epochs

Evaluation:

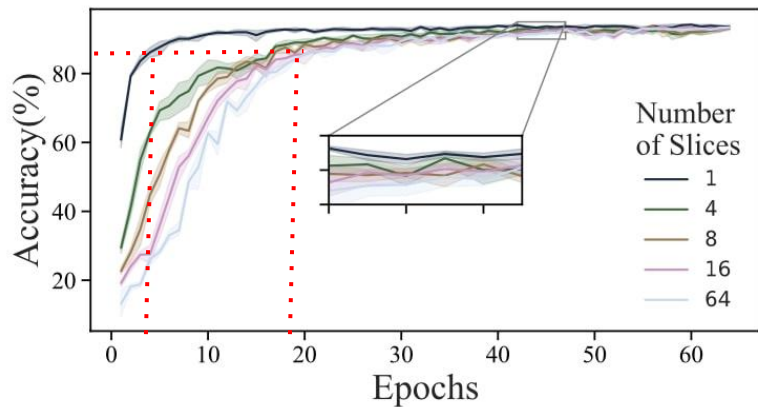
- Measured accuracy with respect to epochs
- Contrast analytical retraining time with the number of slices



Impact of Slicing: Accuracy Results

1. Does increasing '**S**' improve retraining speed-up?
2. When does SISA accuracy degrade too much?

Impact of Slicing: Accuracy Results

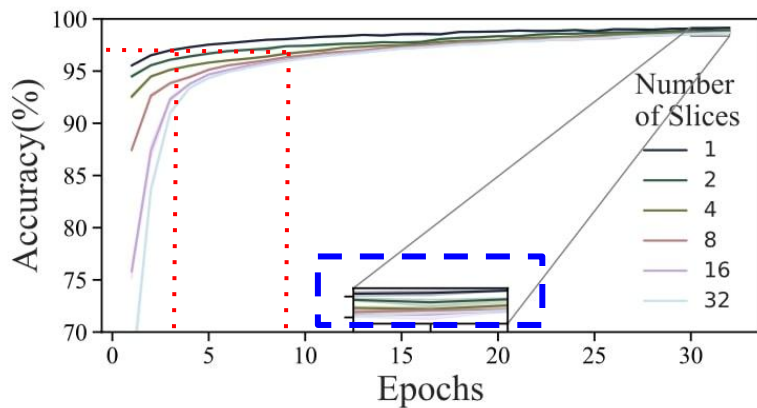
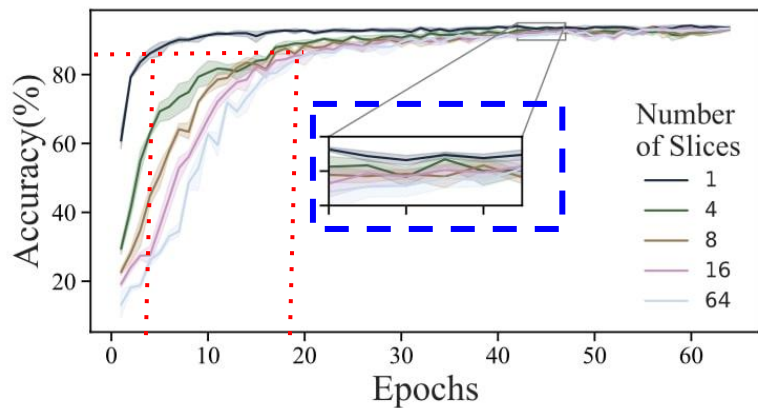


(a) Accuracy vs. Number of epochs for SVHN dataset.

1. For **same accuracy**: **more slices** implies **more epochs**

-> **artifact** of our training procedure

Impact of Slicing: Accuracy Results



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

1. For **same accuracy**: **more slices** implies **more epochs**

-> **artifact** of our training procedure

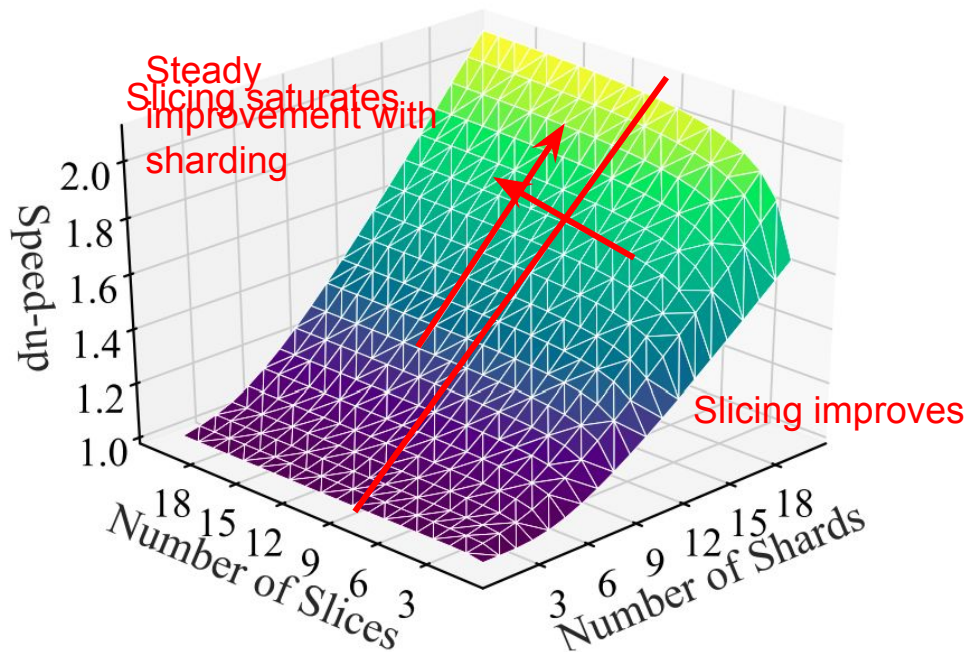
2. For **more slices**: after sufficient training, **negligible accuracy drop**

Combined Speed-up of Sharding & Slicing

How does **increasing** number of slices
improve retraining speed-ups?
With **varied number of shards**?

Combined Speed-up of Sharding & Slicing

Setting: unlearn one batch representing 0.003% of data

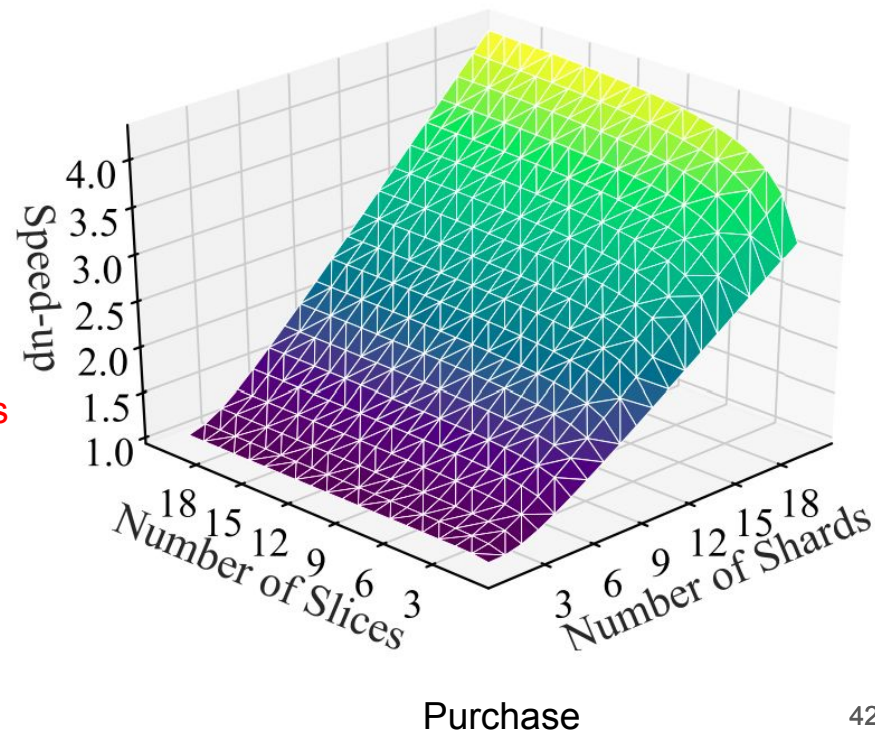
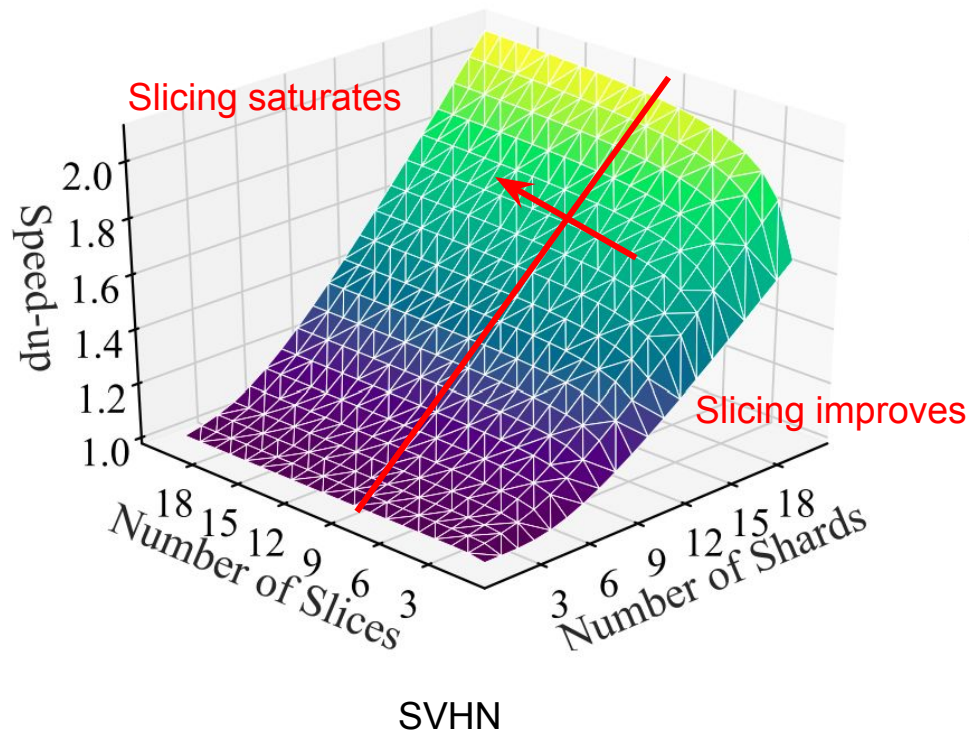


SVHN

Purchase

Combined Speed-up of Sharding & Slicing

Setting: unlearn one batch representing 0.003% of data



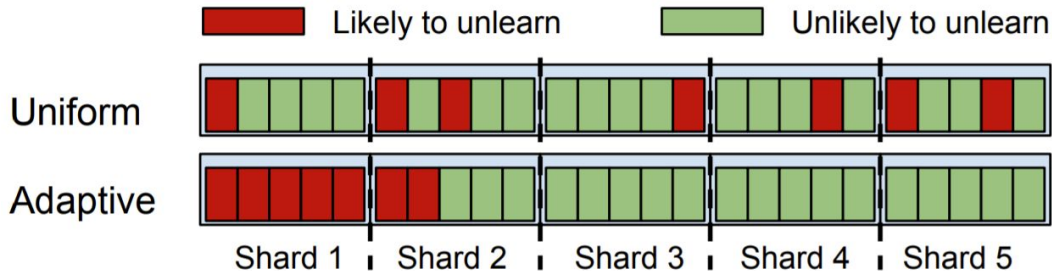
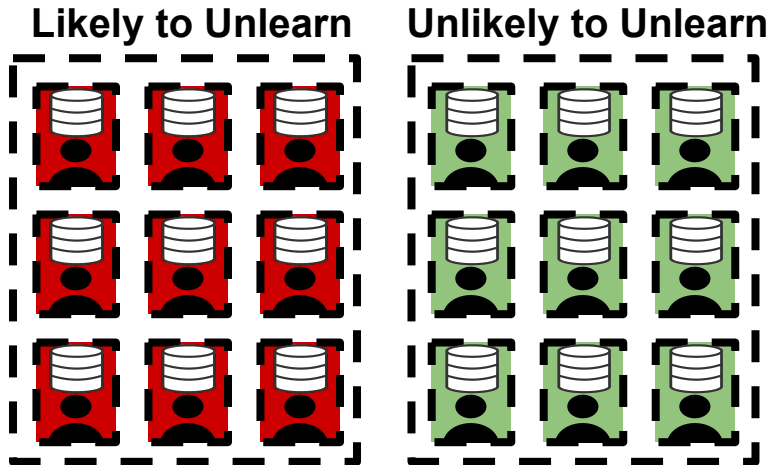
Combined Speed-up of Sharding & Slicing

With sufficiently high number of slices, we get a 1.5x speedup.

A priori Knowledge Can Improve SISA Unlearning

A user's probability for requesting unlearning may depend on **auxiliary data**:

- How data is used and by whom
- The local perception surrounding data use
- Prior data misuse incidents



Distribution-Aware Sharding

Goal: Maximize shard size; minimize chance that a shard has ≥ 1 unlearning request.

Assume: Each user unlearning is an independent Bernoulli trial \sim a probability $p(u)$ coin flip

Key Insight: A group of users unlearning, χ_i , follows a Poisson Binomial Distribution

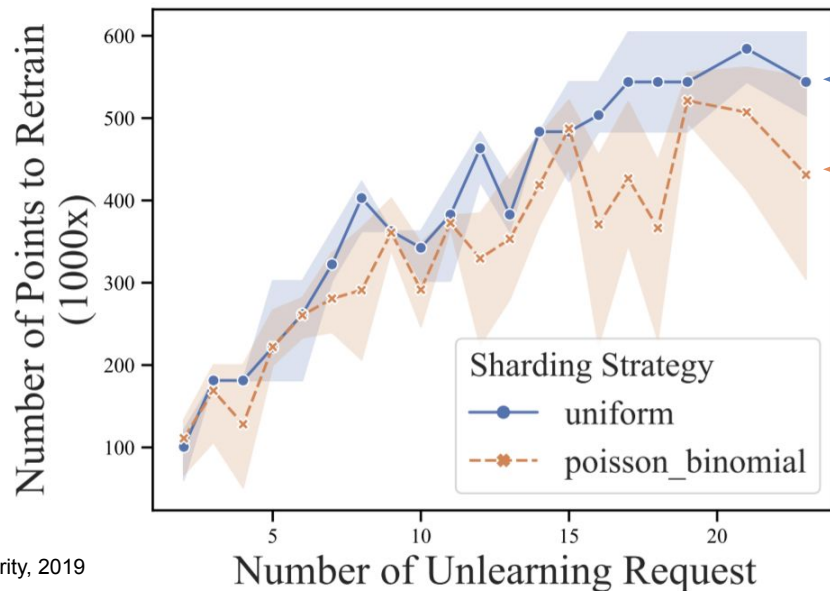
- $\mathbb{E}(\chi_i)$: The expectation that any user in χ_i will request for unlearning.
- $\mathbb{E}(\chi_i) = n\bar{p}$; \bar{p} is the average unlearning probability.

Approach:

- Sample until $\mathbb{E}(\chi_i) < C$, for any constant $C \leq 1$

Distribution-Aware Sharding Performance

Modeled unlearning requests for search engines , from
“Five years of the right to be forgotten” by T. Bertram et. al. *



1. The adaptive Poisson Binomial strategy is never worse
2. Can reduce analytical retraining time.

Limitations

- Degradation in accuracy
- Requires retraining
- Implicitly assumes unlearning requests are independent of the model parameters

Adaptive Machine Unlearning

**Varun Gupta, Christopher Jung, Seth Neel, Aaron Roth, Saeed Sharifi-Malvajerdi, Chris
Waites**

Why adaptive?

- SISA didn't consider the relationship between the unlearning requests and model
- What if people choose to delete their data as a function of the published model (adaptive updates)

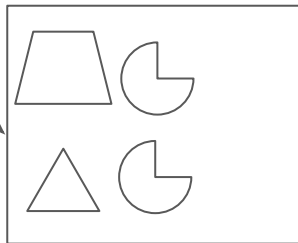
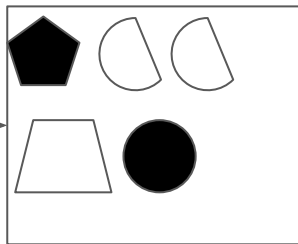
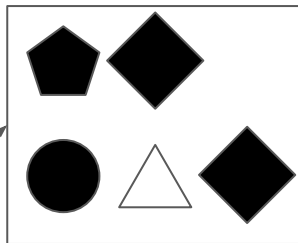
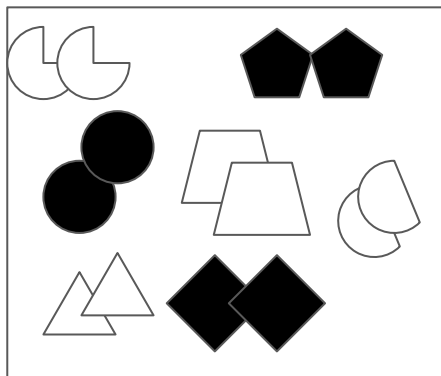


To be removed

Toy example:

$$\{(x_i, y_i)\}_{i=1}^{2n}, x_i \in \mathbb{R}^d, y_i \in \{0, 1\}$$

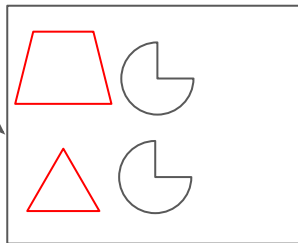
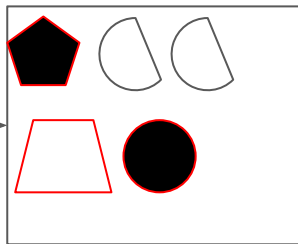
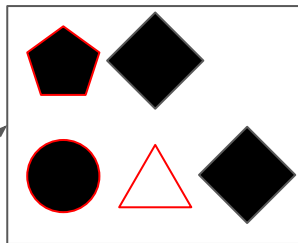
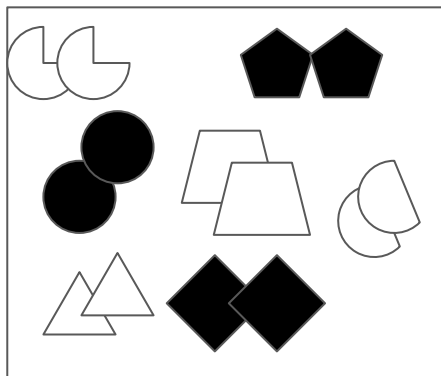
$$f_{\mathcal{D}}(x_i) = \begin{cases} y_i & \text{if } (x_i, y_i) \in \mathcal{D}, \\ \perp & \text{otherwise} \end{cases}$$



Toy example:

$$\{(x_i, y_i)\}_{i=1}^{2n}, x_i \in \mathbb{R}^d, y_i \in \{0, 1\}$$

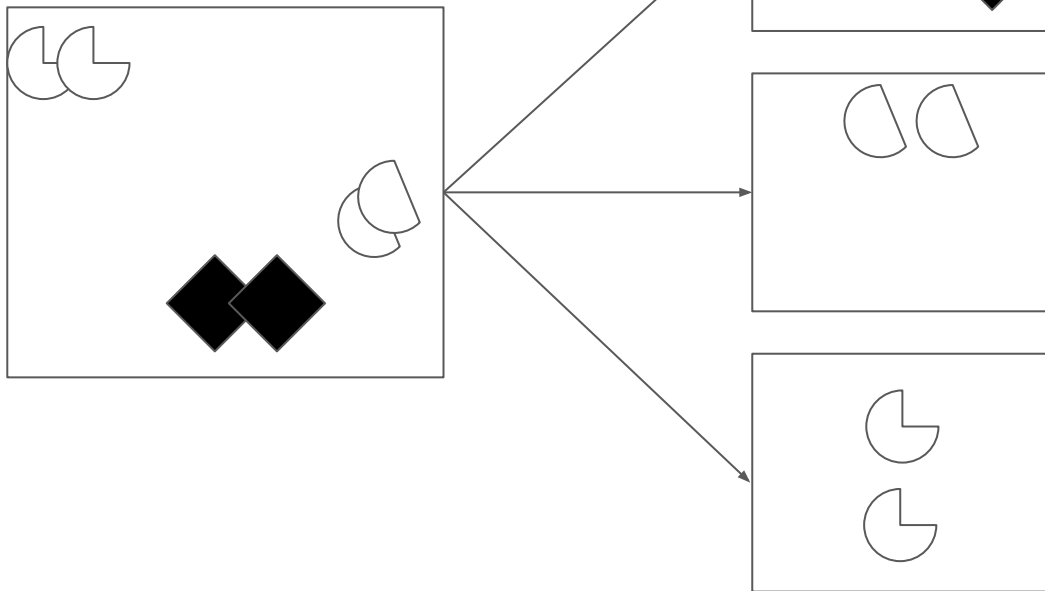
$$f_{\mathcal{D}}(x_i) = \begin{cases} y_i & \text{if } (x_i, y_i) \in \mathcal{D}, \\ \perp & \text{otherwise} \end{cases}$$



Toy example:

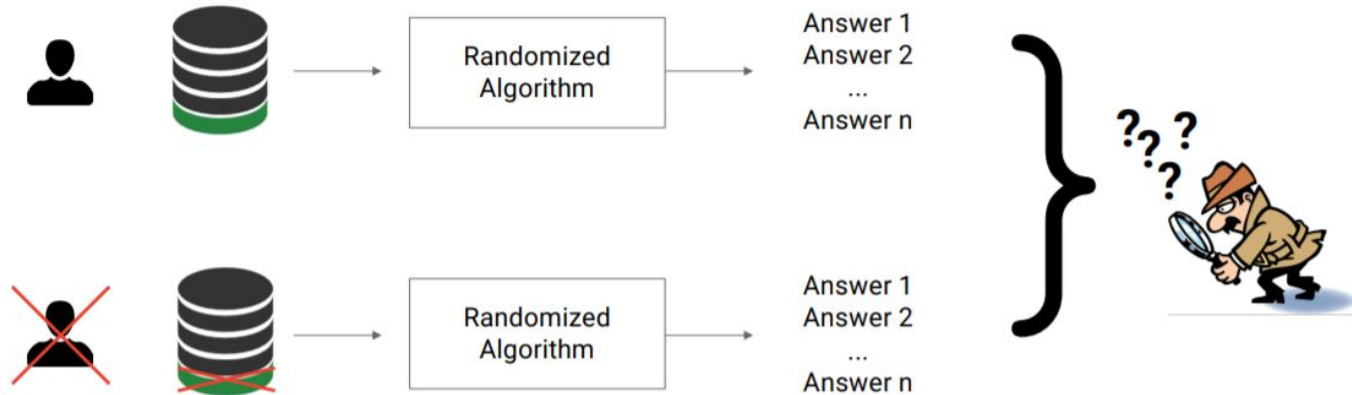
$$\{(x_i, y_i)\}_{i=1}^{2n}, x_i \in \mathbb{R}^d, y_i \in \{0, 1\}$$

$$f_{\mathcal{D}}(x_i) = \begin{cases} y_i & \text{if } (x_i, y_i) \in \mathcal{D}, \\ \perp & \text{otherwise} \end{cases}$$



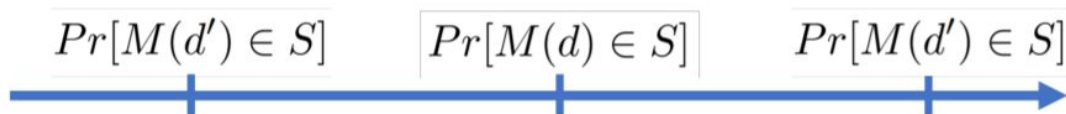
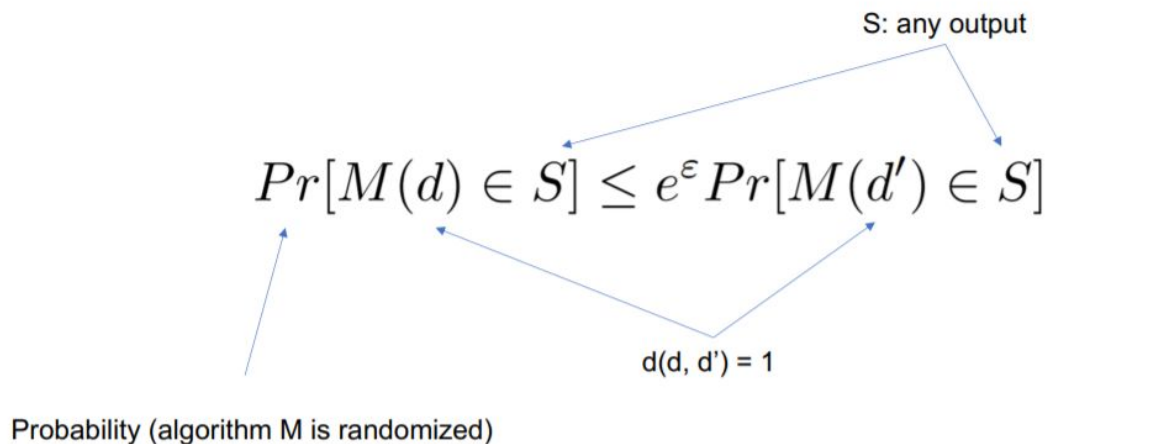
$(\alpha\text{-}\beta\text{-}\gamma)$ unlearning guarantee

- Recall from Prof. Papernot's presentation on DP last week:



$(\alpha\text{-}\beta\text{-}\gamma)$ unlearning guarantee

- Recall from Prof. Papernot's presentation on DP last week:



$(\alpha\text{-}\beta\text{-}\gamma)$ unlearning guarantee

- Recall from Prof. Papernot's presentation on DP last week:

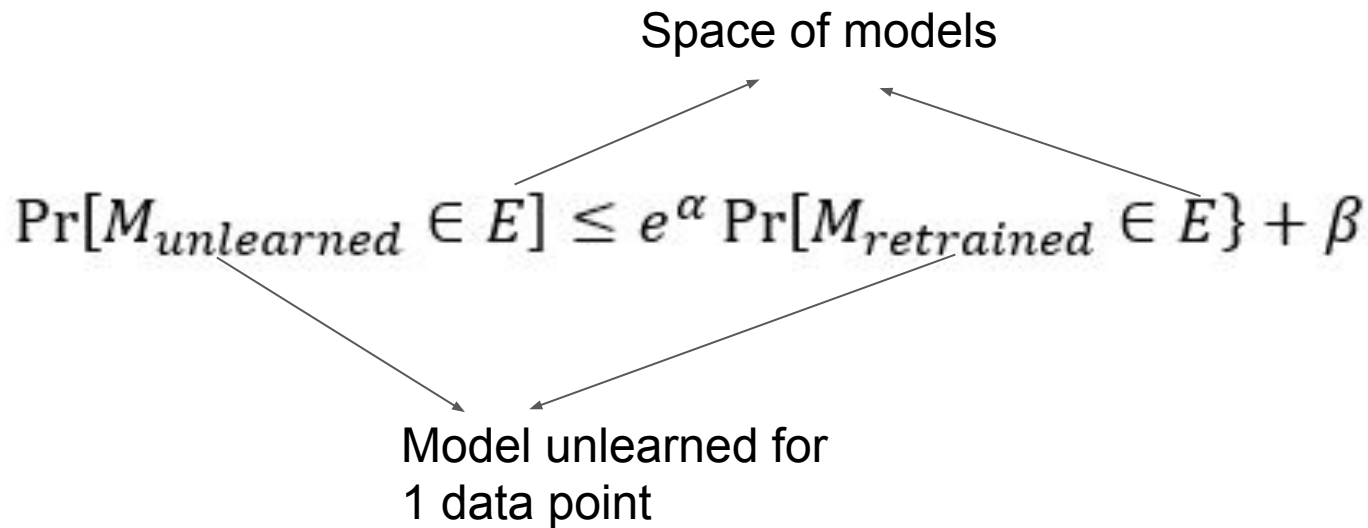
Differential privacy:

A randomized algorithm M satisfies (ϵ, δ) differential privacy if for all pairs of neighbouring datasets (d, d') , for all subsets S of outputs:

$$\Pr[M(d) \in S] \leq e^\epsilon \Pr[M(d') \in S] + \delta$$

$(\alpha\text{-}\beta\text{-}\gamma)$ unlearning guarantee

-



Experimental setup

- K shards
- Adaptive deletion:
 - Delete the points from the $k/2$ shards if the models have higher confidence for the correct label
- Evaluation test statistics: whether the avg acc of models from targeted shards < avg acc of models from the non-targeted shards
 - Full retraining: expected to be 0.5

Recall: DP-SGD

Compute gradient

For each $i \in L_t$, compute $\mathbf{g}_t(x_i) \leftarrow \nabla_{\theta_t} \mathcal{L}(\theta_t, x_i)$

Clip gradient

$\bar{\mathbf{g}}_t(x_i) \leftarrow \mathbf{g}_t(x_i) / \max(1, \frac{\|\mathbf{g}_t(x_i)\|_2}{C})$

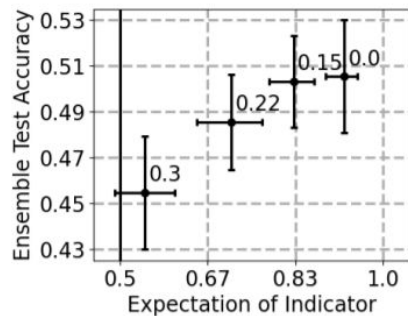
Add noise

$\bar{\mathbf{g}}_t \leftarrow \frac{1}{L} (\sum_i \bar{\mathbf{g}}_t(x_i) + \mathcal{N}(0, \sigma^2 C^2 \mathbf{I}))$

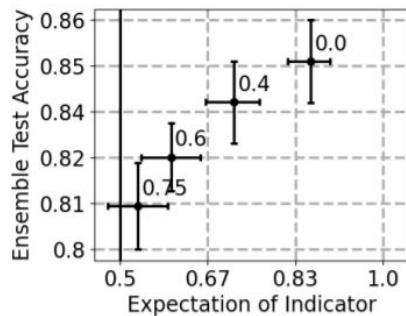
Descent

$\theta_{t+1} \leftarrow \theta_t - \eta_t \bar{\mathbf{g}}_t$

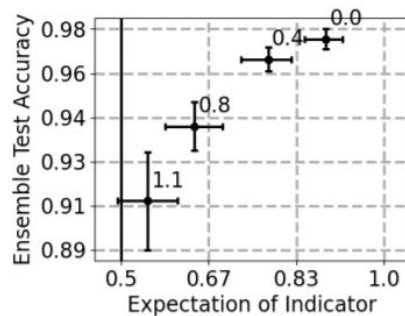
Experimental results



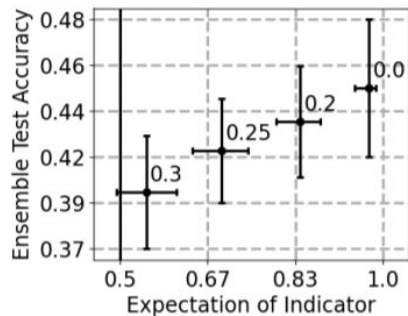
(a) CIFAR-10 _{$k=6$}



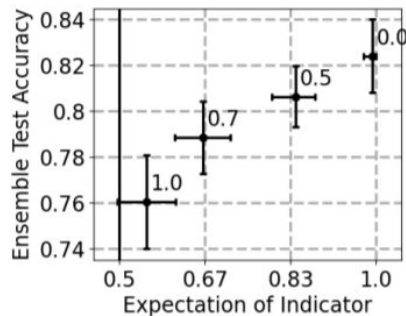
(b) Fashion-MNIST _{$k=6$}



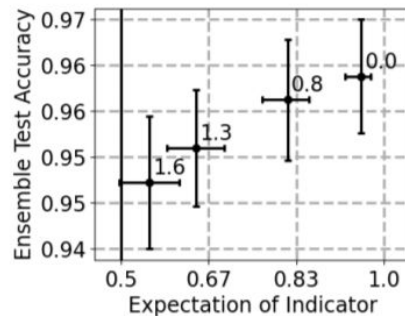
(c) MNIST _{$k=6$}



(d) CIFAR-10 _{$k=2$}



(e) Fashion-MNIST _{$k=2$}



(f) MNIST _{$k=2$}

Limitations

- No meaningful specifications for the parameters
- The architecture used in the experiments is “trivial”
- DP might require retraining

How do we verify unlearning?

- For the two retraining-based unlearning methods we just discussed, one may think this question is equivalent to:
 - How do we verify if someone has indeed trained a model?
 - Then the solution can be what we learned in week 6: Proof-of-Learning, or in this case, we can call it **Proof-of-Unlearning, PoUL**
- However, there is an issue here
 - PoL is about model ownership, and deals with model-stealing adversary
 - The adversary against PoUL, may be the model trainers themselves, who have already trained the model once

Recent Unlearning Methods

Approximate Unlearning	Exact Unlearning

Recent Unlearning Methods

Approximate Unlearning	Exact Unlearning
<ul style="list-style-type: none">• Defines unlearning at the level of model parameters• Directly modify the parameters• Better efficiency• e.g., Amensiac Unlearning, by Graves et al.	<ul style="list-style-type: none">• Defines unlearning at the level of algorithms• Retraining-based• Stronger guarantee• e.g., SISA Unlearning, by Bourtole et al.

On the Necessity of Auditable Algorithmic Definitions for Machine Unlearning

Anvith Thudi, Hengrui Jia, Ilia Shumailov, Nicolas Papernot

Data Ordering Attack: Spoofing PoUL is possible

- **Proof of Learning:** record of intermediate checkpoints and all other information needed to reproduce the training, such that it can be later used to verify the ownership of an ML model
- **Data ordering attack:** force a step of SGD to approximately compute a given gradient by carefully selecting points the step is computed on
- When receive unlearning requests, an adversarial model deployer can “forge” the gradient step containing the data points to be unlearned by data ordering attack, instead of retraining the model
- Such a PoUL would be valid based on the definition of the PoL paper.

Approximate Unlearning is ill-defined

- In the definition of approximate unlearning, a model is unlearned if it can be obtained by training on a dataset that does not contain the points to be unlearned
- However, if one can “forge” any step with data points that are not required to be unlearned, then a model can be unlearned and not unlearned at the same time by this definition

Questions?