On the Necessity of Auditable Algorithmic Definitions for Machine Unlearning

Anvith Thudi, Hengrui Jia, Ilia Shumailov, Nicolas Papernot

UNIVERSITY OF TORONTO

VECTOR INSTITUTE

Outline

1. Background on Unlearning

2. Verifying Unlearning: What is plausible?

3. Impossibility Results with Verification

Background on Unlearning

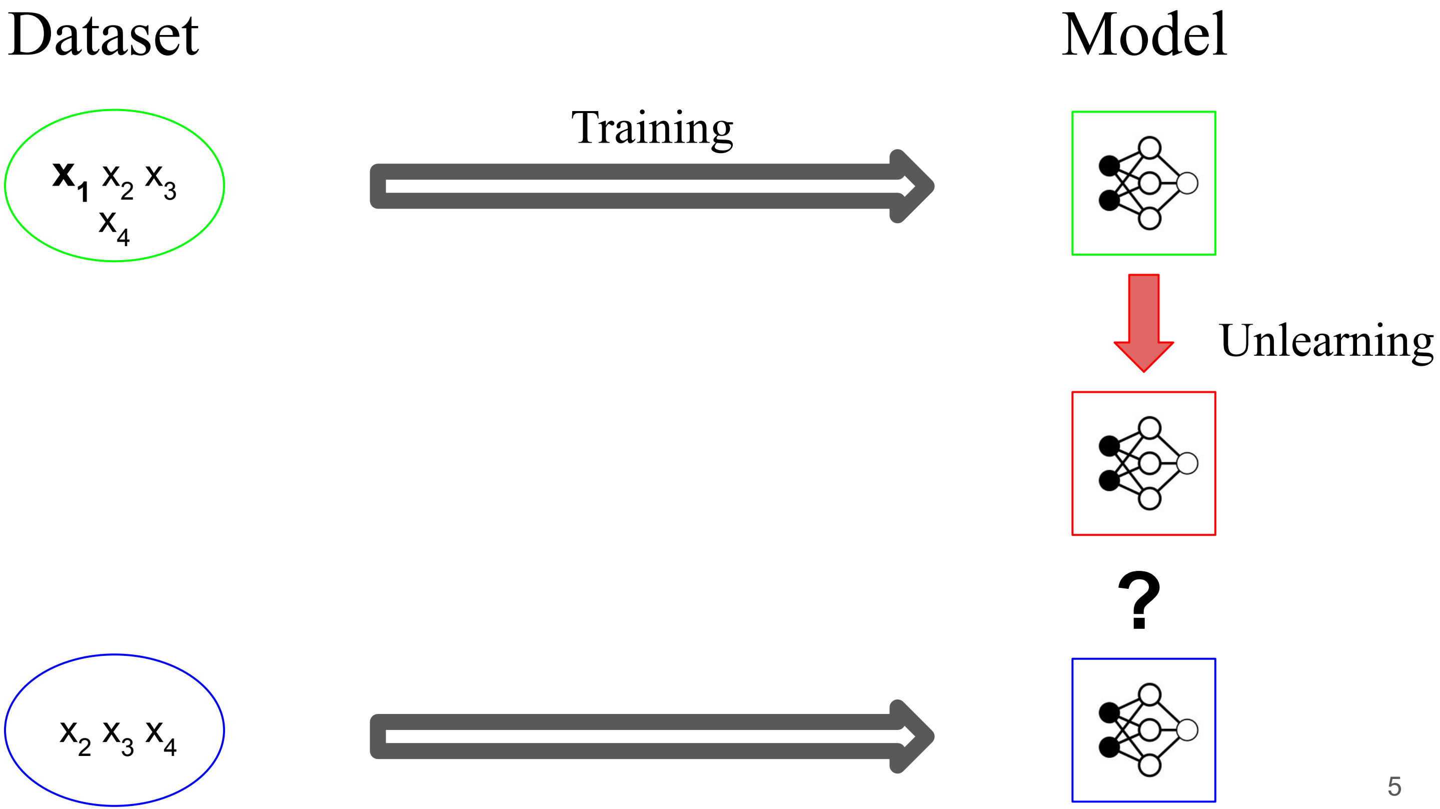
Why Unlearn?

1. Privacy: *Right-to-be-forgotten* (EU GDPR) 🇪🇺

2. Security: Data Poisoning 😈

3. Performance: Bad data %

The “Scenario”



Dataset

Model

$\mathbf{x_1}$ x_2 x_3 x_4

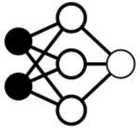
Training

Unlearning

?

x_2 x_3 x_4

How to Represent Models?

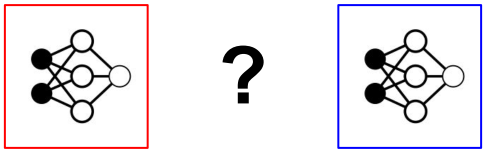


Could be:

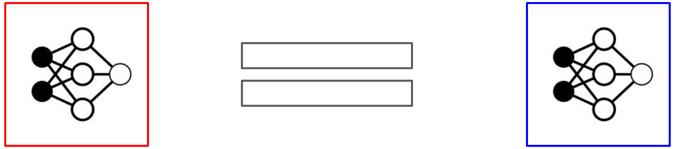
1. (Distribution of) Weights
2. (Distribution of) Functions

Weights ≠ Functions

The Big Question



Exact Unlearning

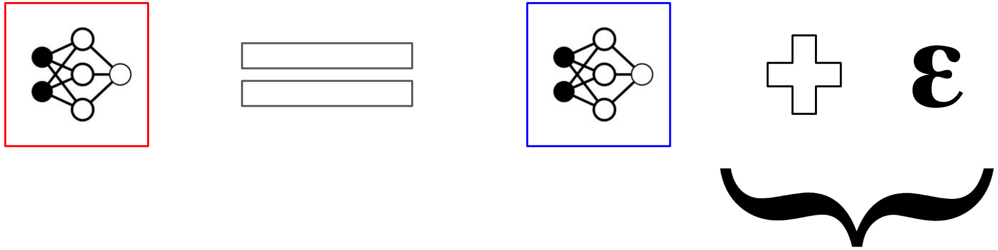


“Machine Unlearning” Bourtoule et al.

Expensive

Only known methods are Retraining

Approximate Unlearning



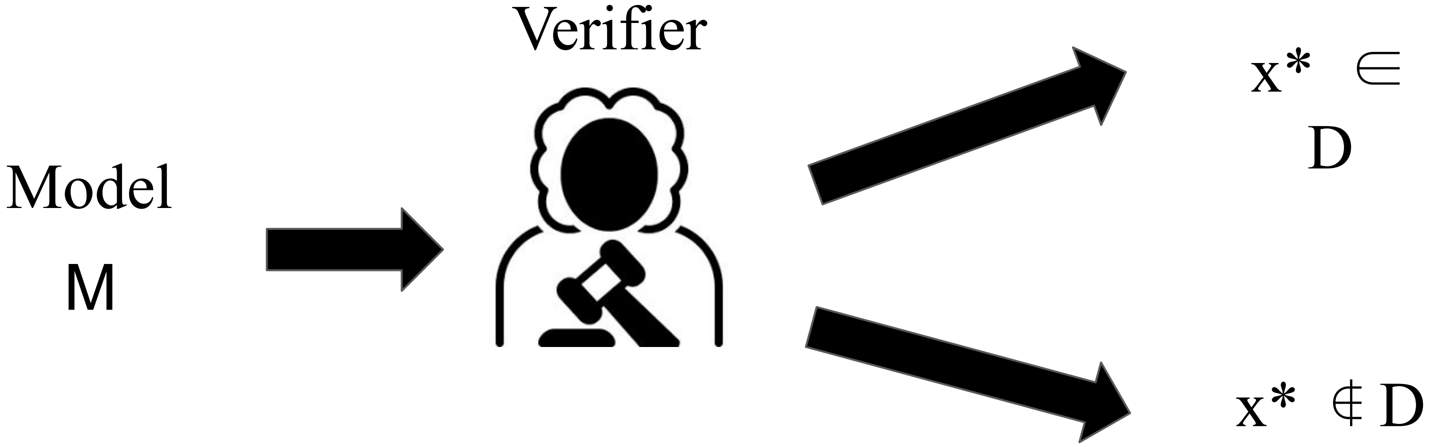
W.r.t some “metric” d

“Unrolling SGD: Understanding Factors Influencing Machine Unlearning” [TGCP] Euro S&P 22’

Verifying Unlearning

Is Unlearning Verifiable?

Specifically is “exact” unlearning (i.e not training on a datapoint) verifiable?



Verifier

Model

M

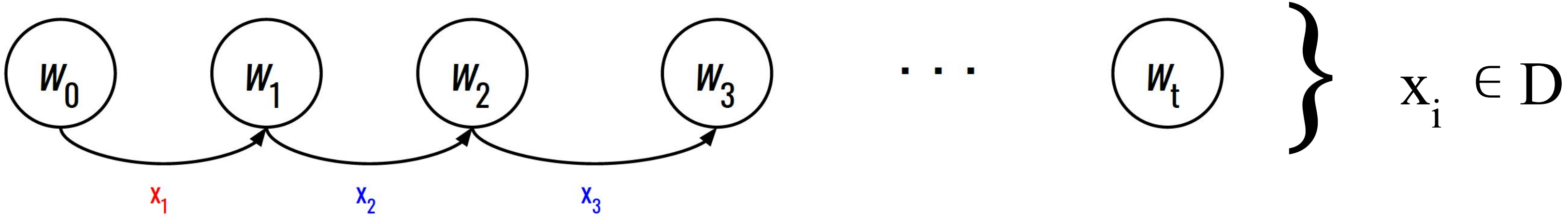
$x^* \in D$

$x^* \notin D$

Q: Can such a function exist?

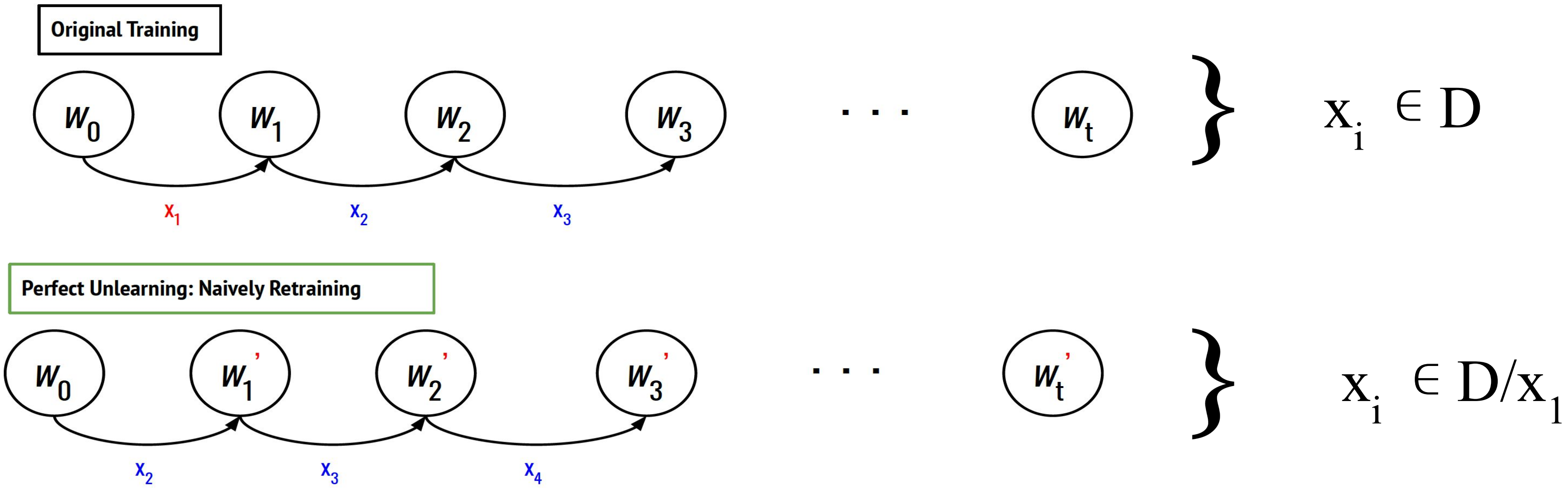
Framework for Plausible

POL: a proof of plausibility of training with a given dataset (originally for model stealing)



“Proof-of-Learning: Definitions and Practice” [JYCDTCP] S&P 21’

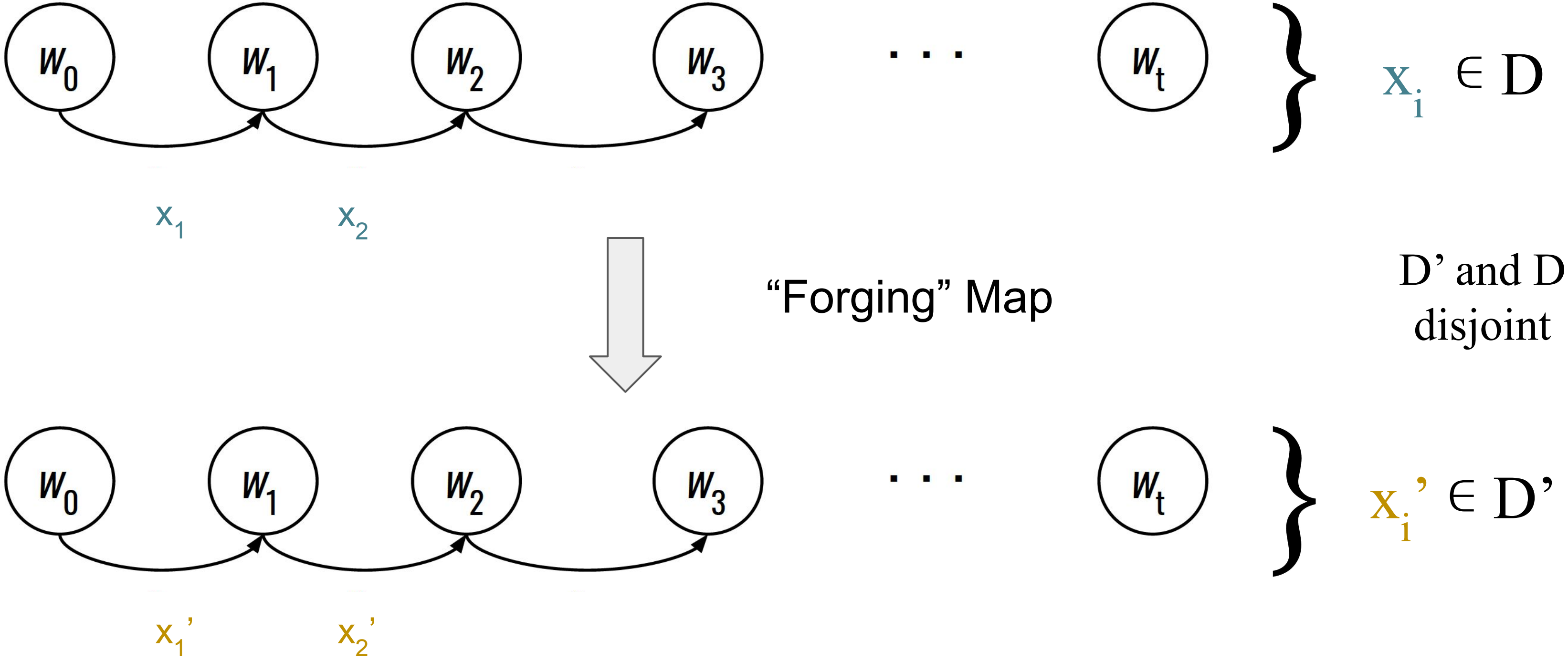
Verifying



Assumption: **plausible without a point means never training on it**

Have some Problems

Forging



$x_i \in D$

“Forging” Map

D’ and D disjoint

$x_i' \in D'$

Some High-Level Ideas

1) D and D’ have similar datapoints (gradients don’t change much)

2) D’ is big (i.e gradients are “dense”)

Formal Existence of Forging

When $D \sim \mathfrak{D}$ and $D' \sim \mathfrak{D}$ (**same underlying distribution**) forging can exist*

1) Assumes *bounded update rule standard deviation*, *mean-sampling (i.e mini-batches)*, *unconstrained mini-batch size,* $\mathfrak{D}$ *absolutely continuous*

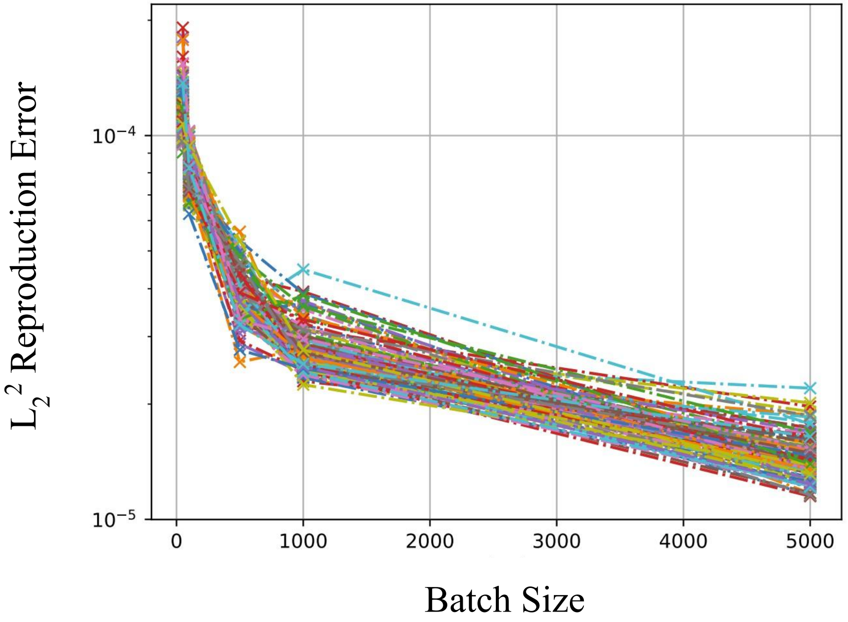
2) Proof Strategy: increasing batch sizes approximates mean gradient, then Markov inequality and existence by non-zero probability

Instantiating Forging

Can implement by brute force

- Take D’ ⊂ D/x*, search through random batches of D’

- Analogous to “Manipulating SGD with Data Ordering Attacks” [SSKZPEA] Neurips 21’


L_2^2 Reproduction Error
10^{-4}
10^{-5}
0
1000
2000
3000
4000
5000
Batch Size

Conclusions

1) Being unlearnt is ***not always a well-defined property***

- Not training on a datapoint is not always a well-defined property

2) Verifying retraining (i.e., exact unlearning) requires ***algorithmic considerations***

- Definition of unlearning is necessarily tied to how training is done

Future Directions

Some Questions

1) Constraints for verifying training data?

2) Building on the Forging framework
   - Relation to ML theory?

3) Privacy implications of Forging?