Verifying Data Attributions Without Breaking the Bank

Martin Pawelczyk

Harvard University

Efficiently Verifiable Proofs of Data Attribution

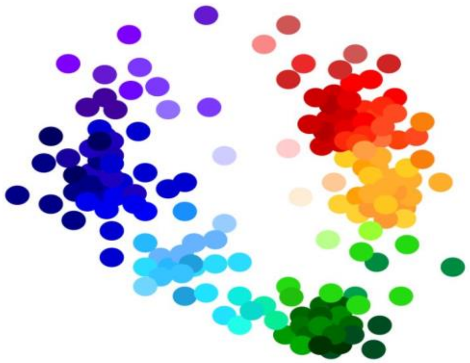
Ari Karchmer* **Martin Pawelczyk**† **Seth Neel**‡

Abstract

Data attribution methods aim to answer useful counterfactual questions like "what would a ML model's prediction be if it were trained on a different dataset?" However, estimation of data attribution models through techniques like empirical influence or "datamodeling" remains very computationally expensive. This causes a critical trust issue: if only a few computationally rich parties can obtain data

Motivation

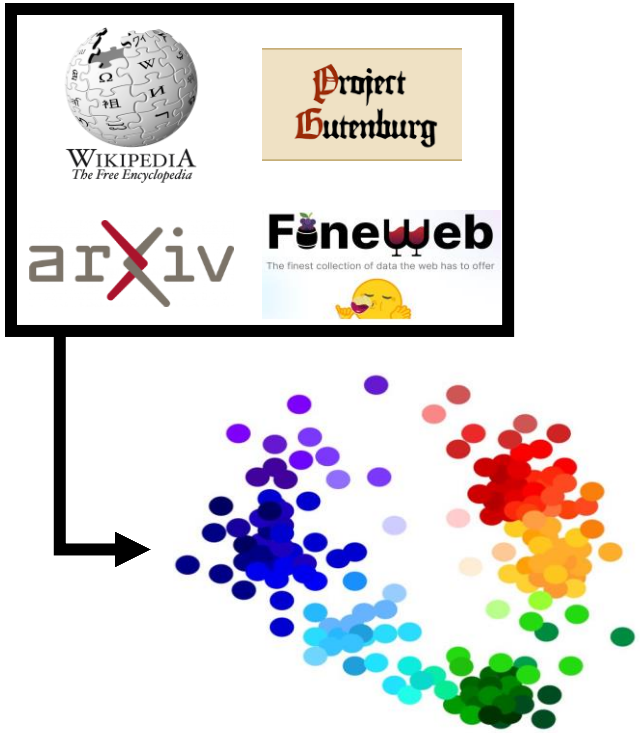
- Data is one of the most fundamental building blocks of AI
- There are surprisingly many open problems



Massive Scraped Internet Dataset

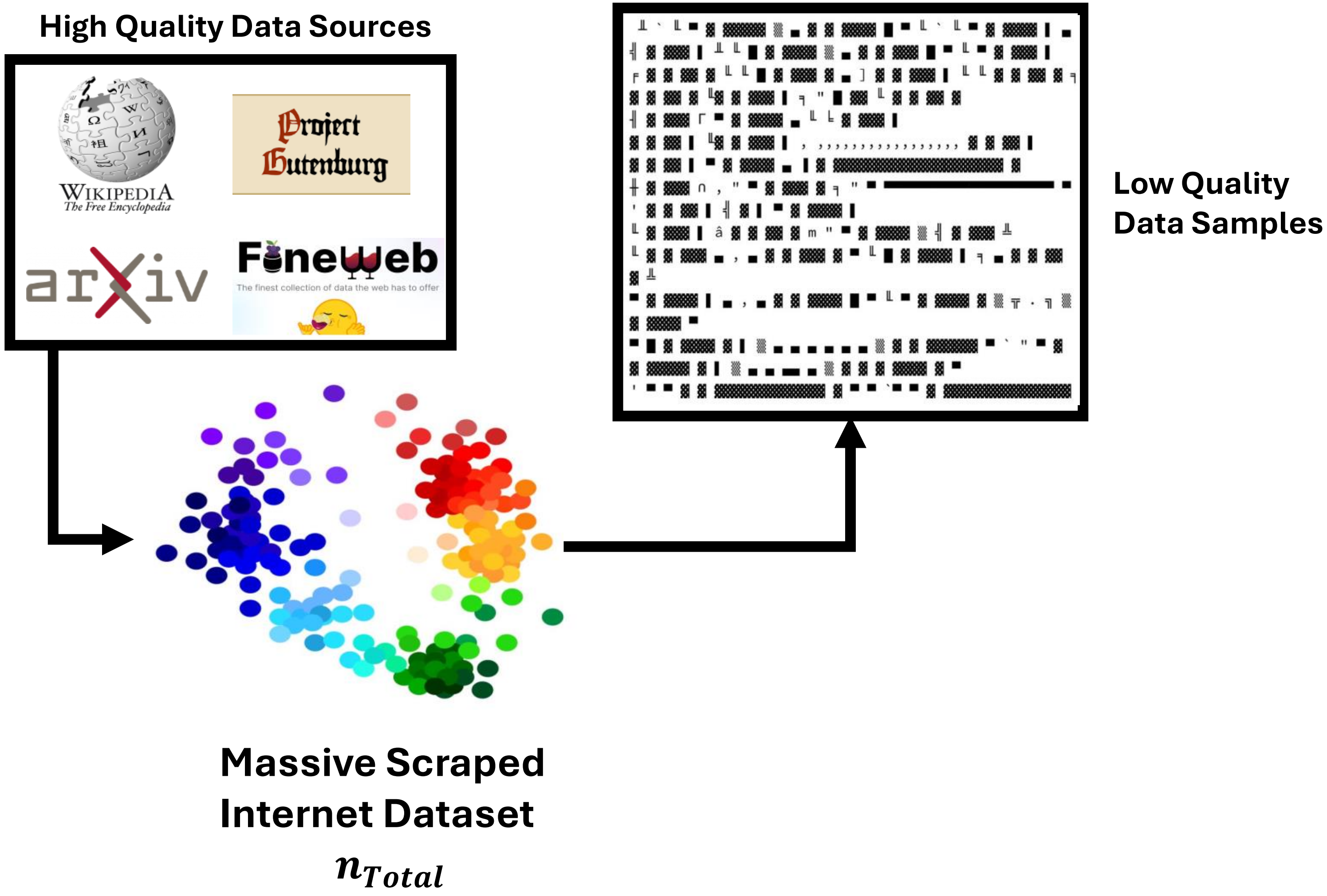
n_{Total}

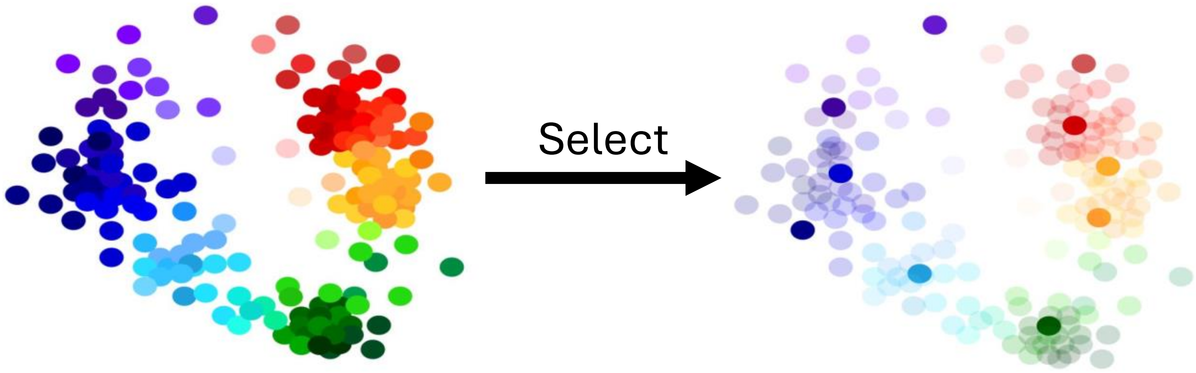
High Quality Data Sources



Massive Scraped Internet Dataset

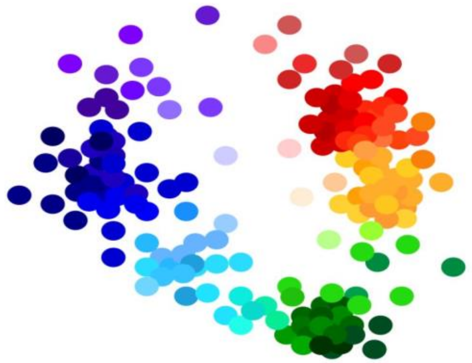
n_{Total}


High Quality Data Sources
Wikipedia
The Free Encyclopedia
Project Gutenberg
arXiv
FineWeb
The finest collection of data the web has to offer
Low Quality
Data Samples
Massive Scraped
Internet Dataset
n_{Total}



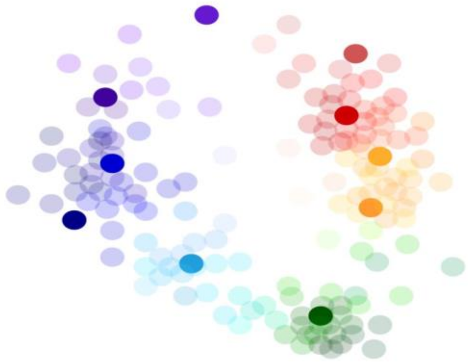
Massive Scraped Internet Dataset
n_{Total}

Selected Dataset
$n_{Total} \gg n_{selected}$



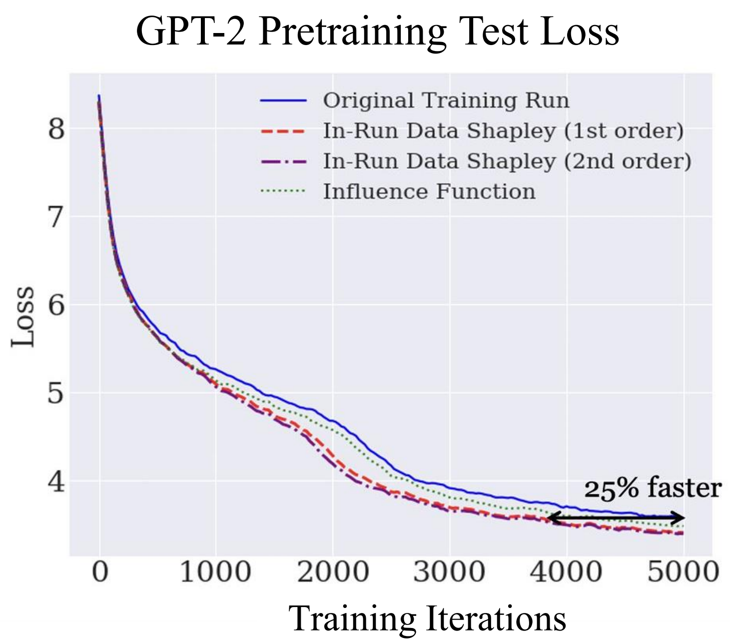
Massive Scraped Internet Dataset
n_{Total}

Select



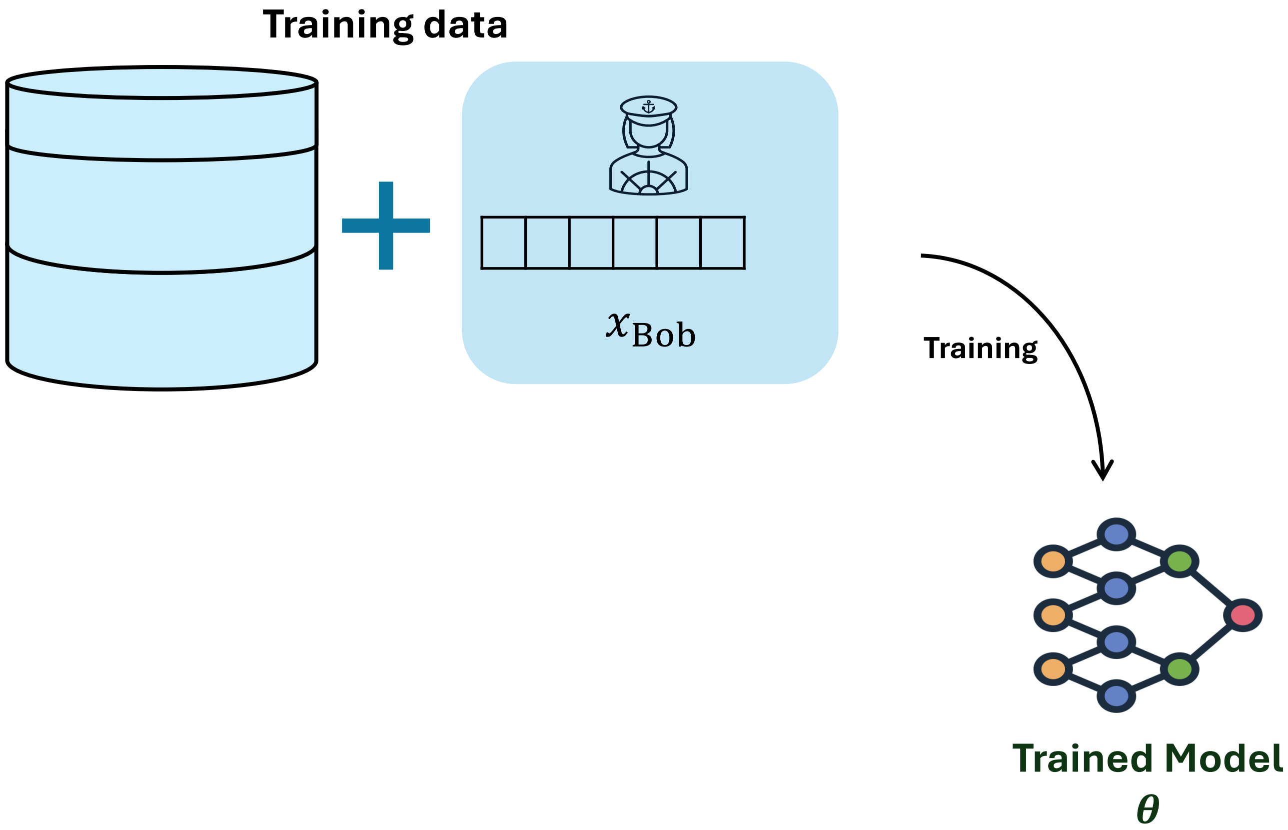
Selected Dataset
$n_{Total} \gg n_{selected}$

Train

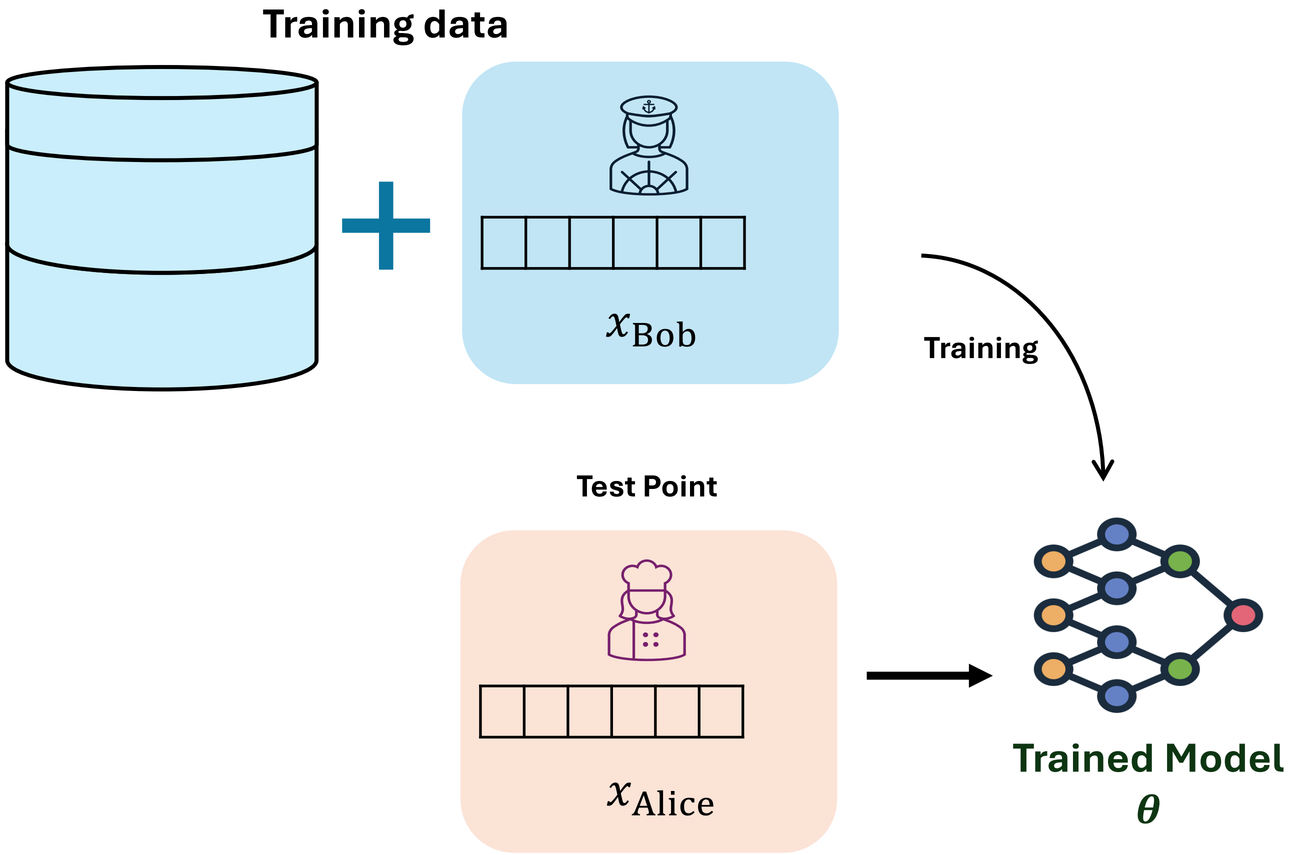


Wang et al (2025), „Data Shapley in One Training Run“, ICLR 2025

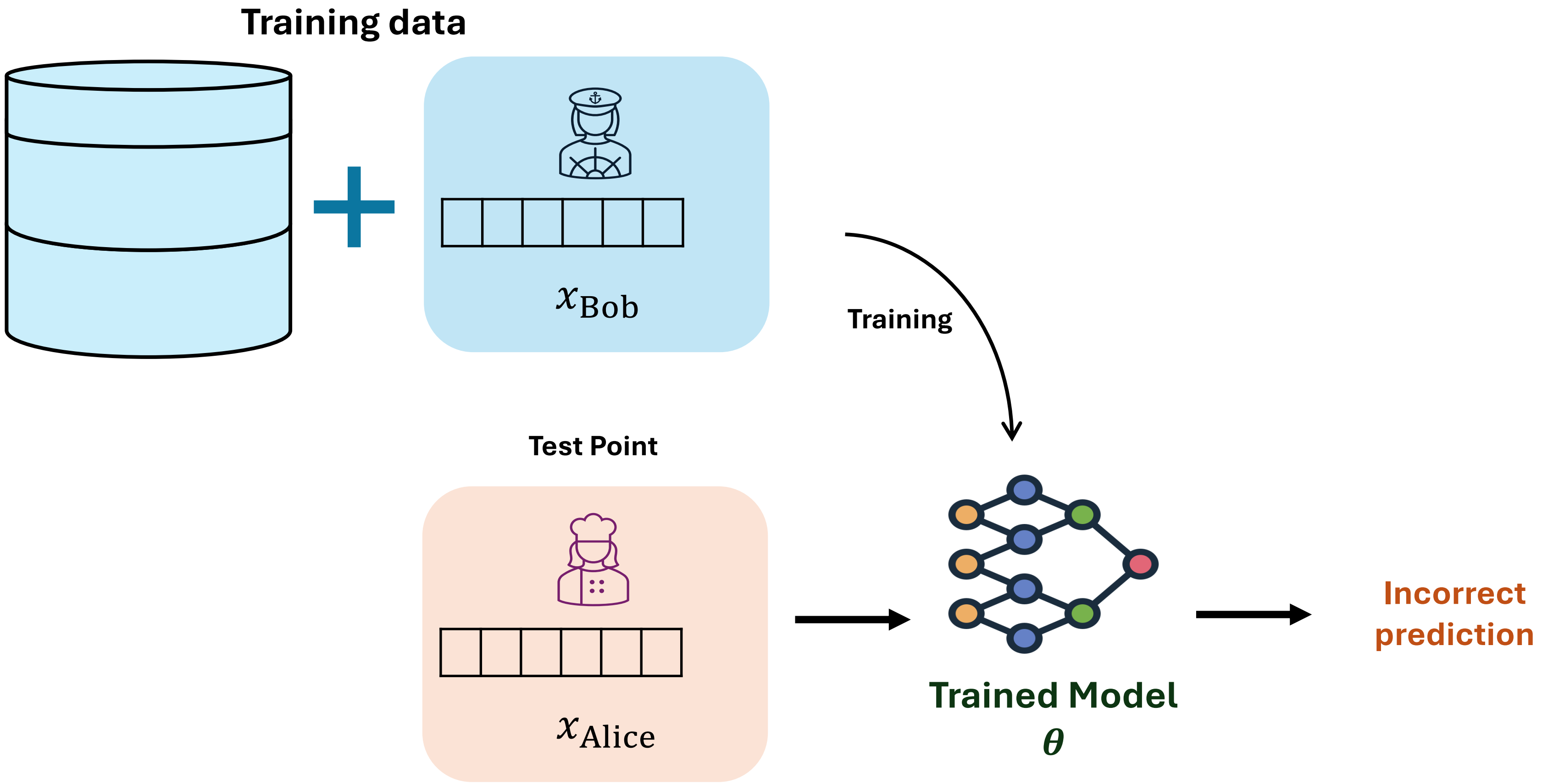
Data Attributions: The Mechanism



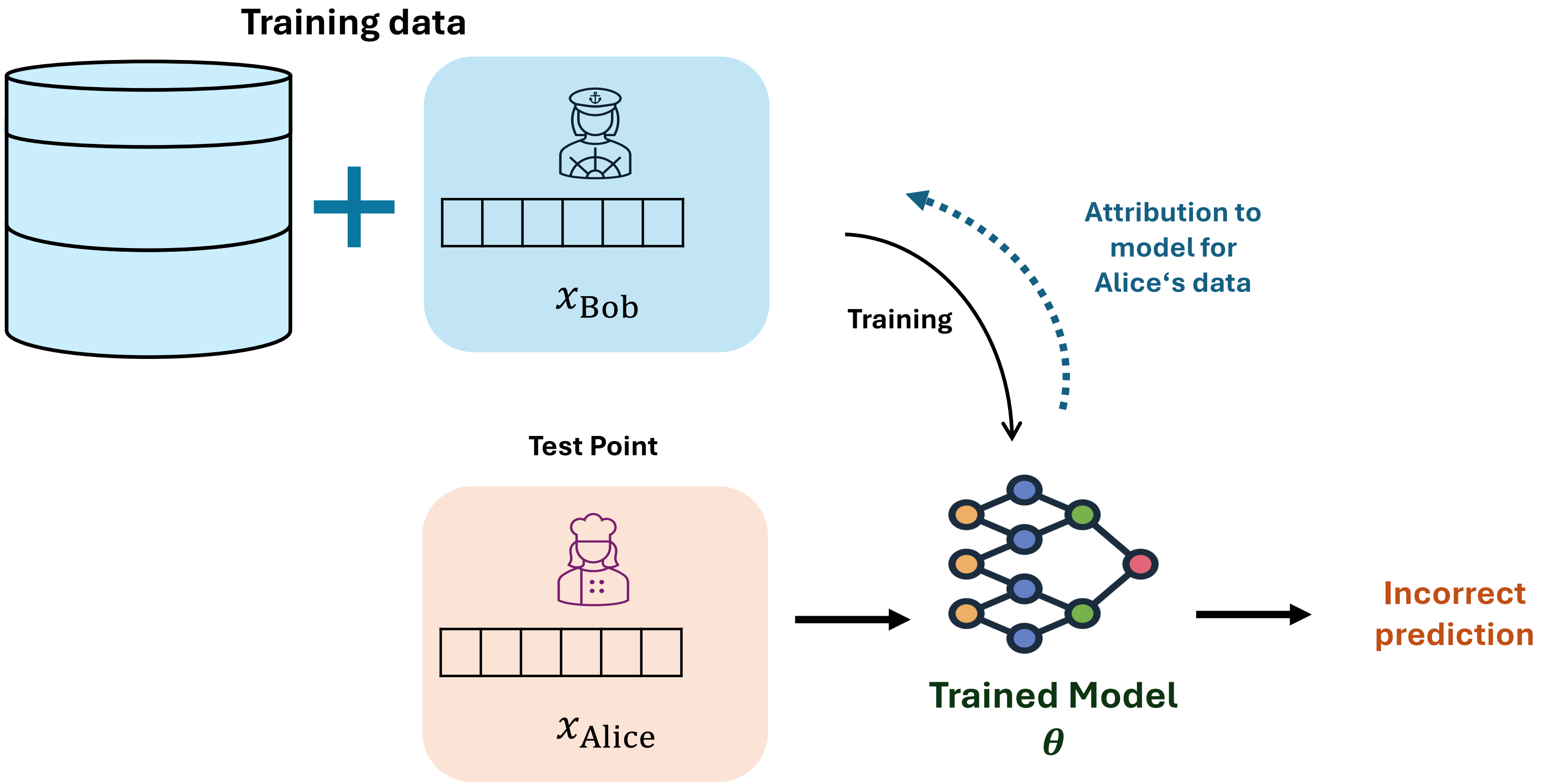
Data Attributions: The Mechanism



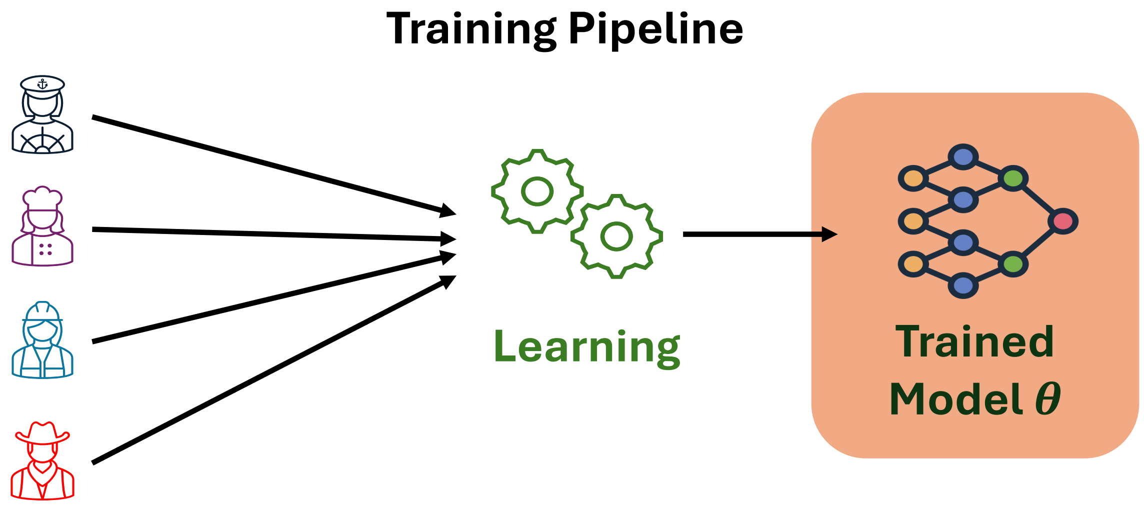
Data Attributions: The Mechanism



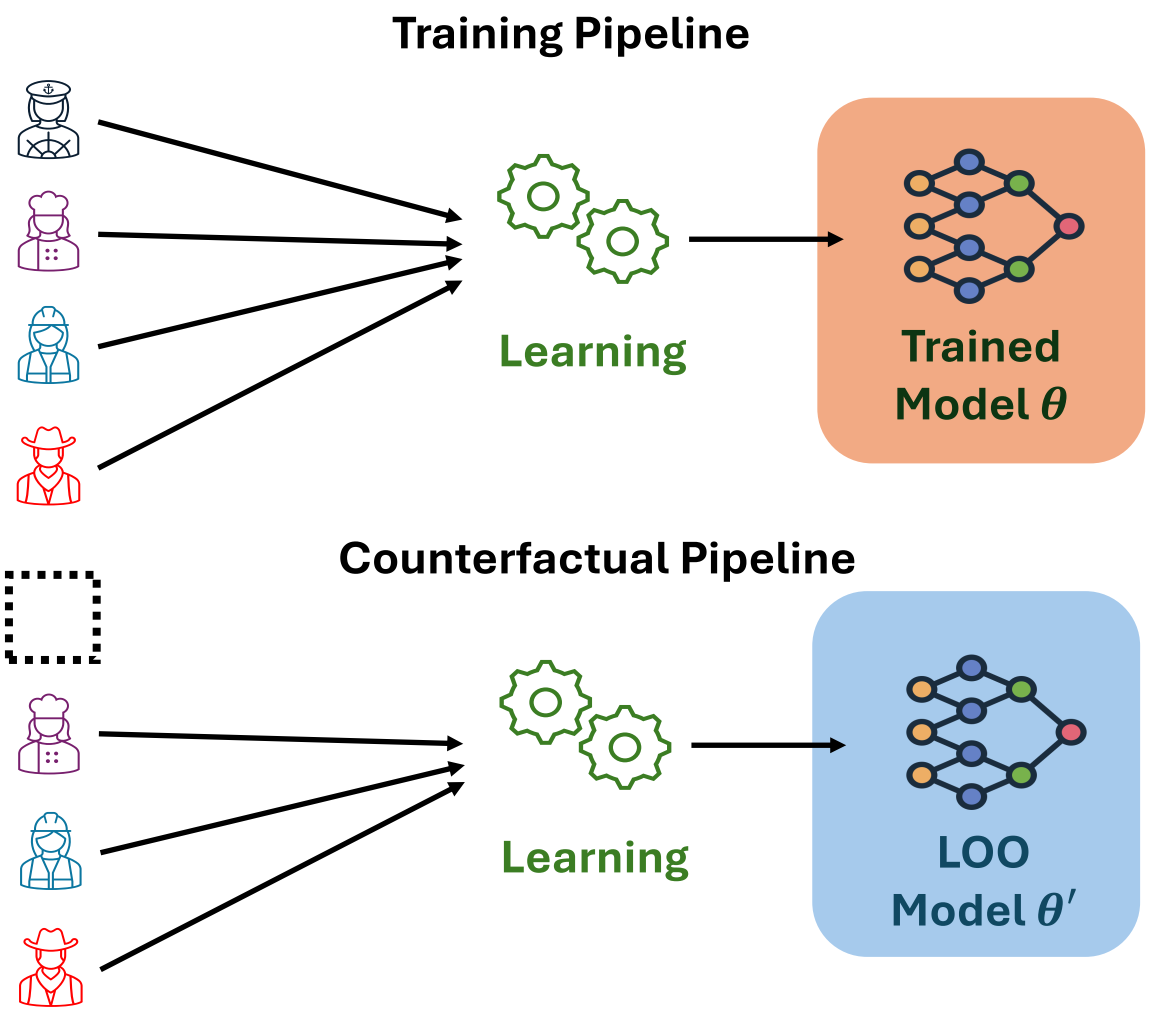
Data Attributions: The Mechanism



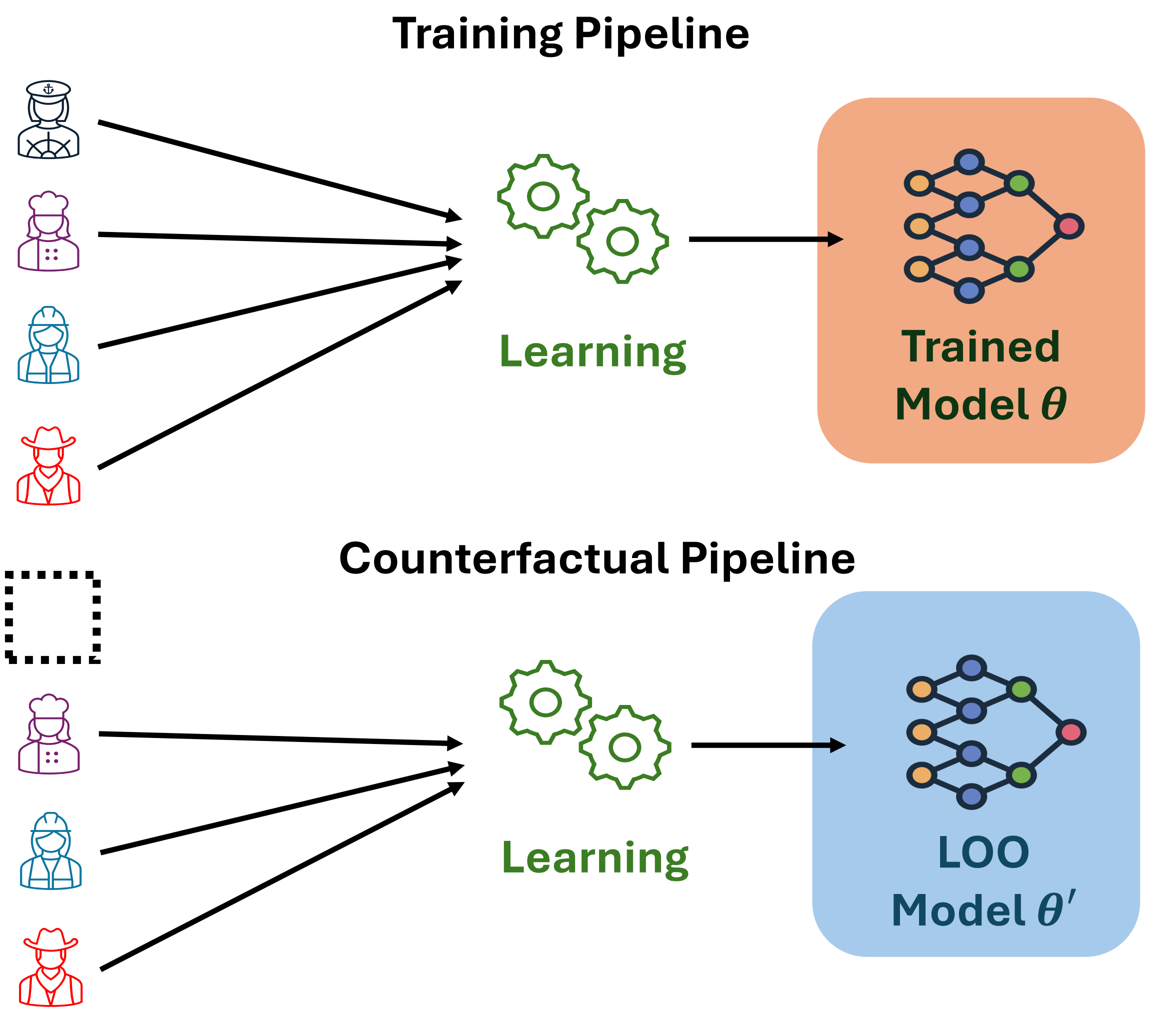
Leave-One-Out Data Attributions Hampel (1974)



Leave-One-Out Data Attributions Hampel (1974)



Leave-One-Out Data Attributions Hampel (1974)

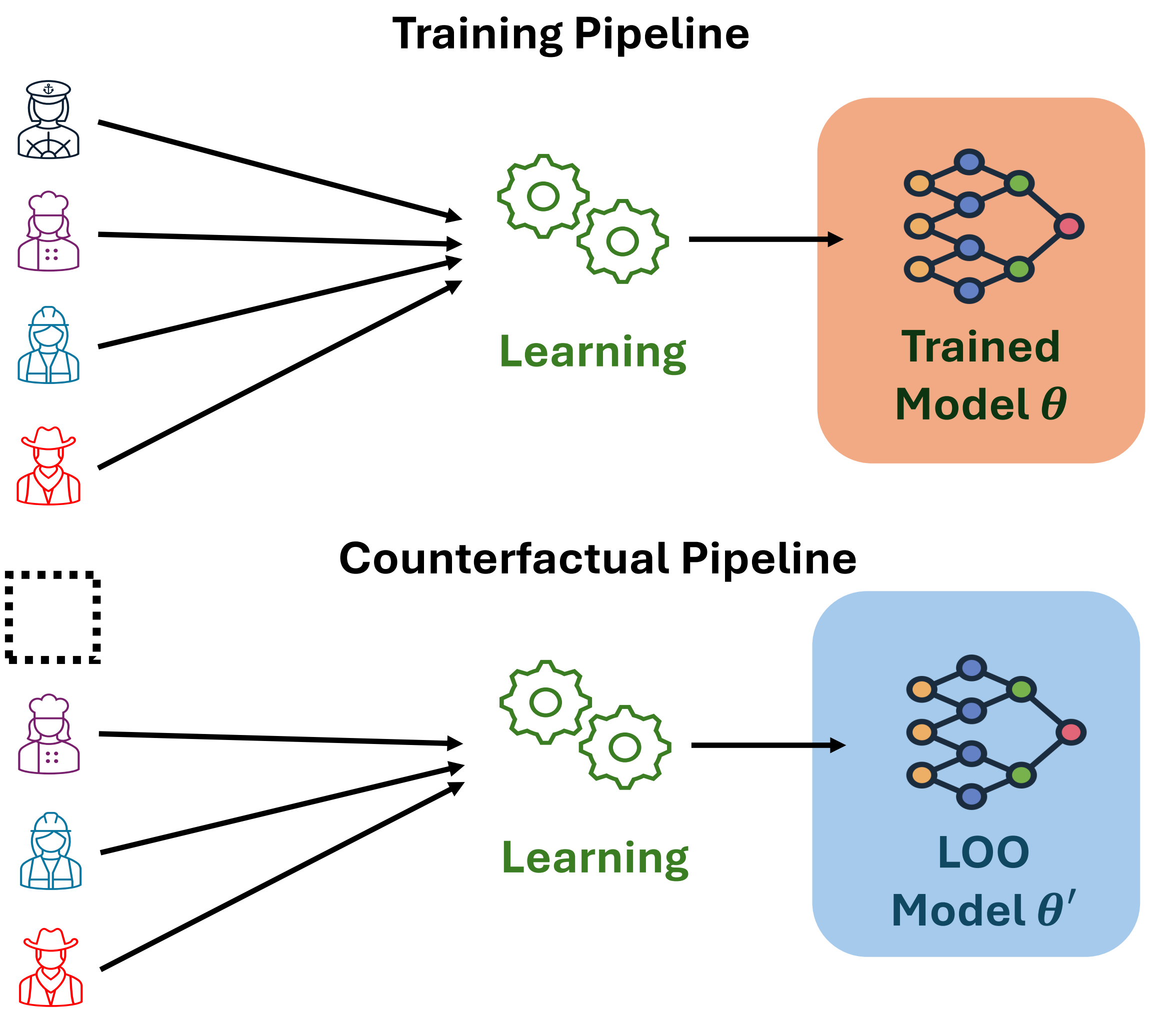


Exact Training Point Attribution

Importance of Bob's Data on test point prediction:

$$a_i = \theta(x_{test}) - \theta'(x_{test})$$

Leave-One-Out Data Attributions Hampel (1974)



Exact Training Point Attribution

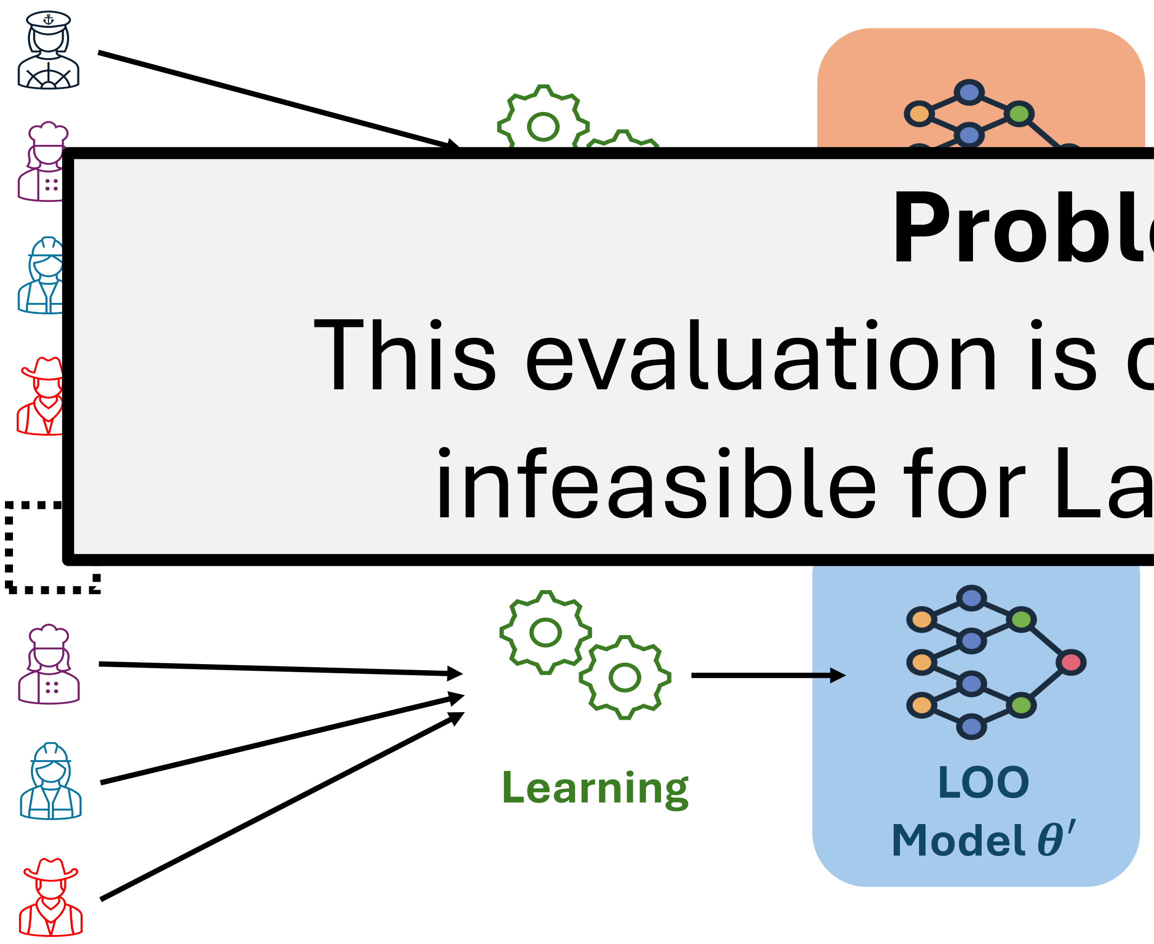
Importance of Bob's Data on test point prediction:

$$a_i = \theta(x_{test}) - \theta'(x_{test})$$

Compute a_i for all i $\in \{1, \ldots, N\}$

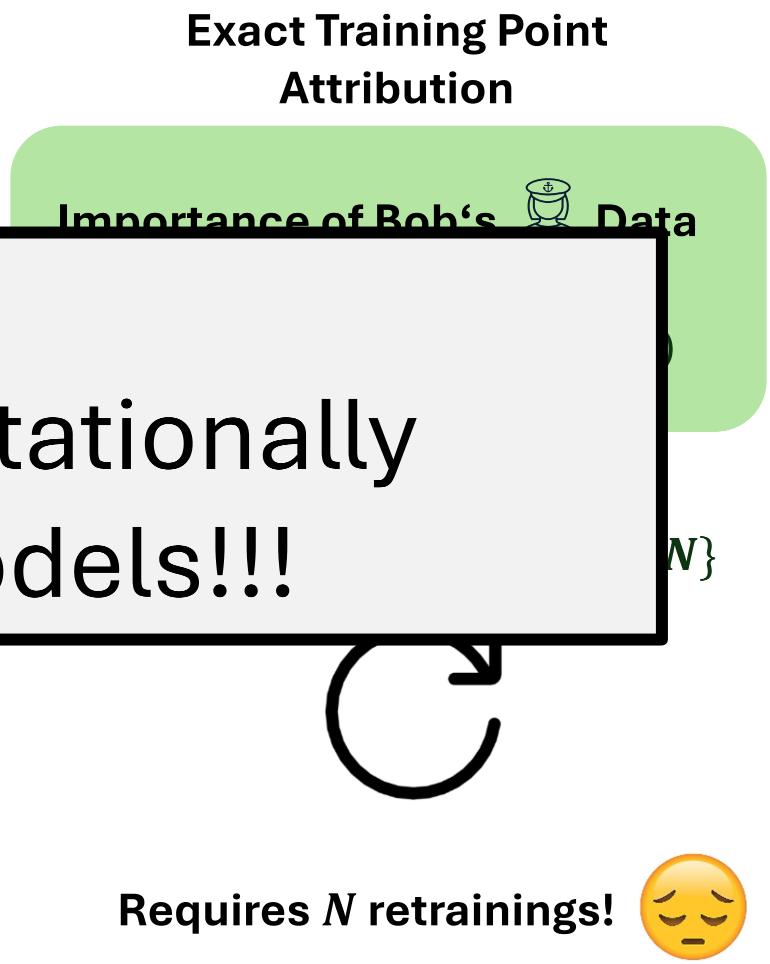
Requires N retrainings! 😔

Leave-One-Out Data Attributions Hampel (1974)



Problem

This evaluation is computationally infeasible for Large Models!!!



Empirical Influence Functions Feldman & Zhang (2020)

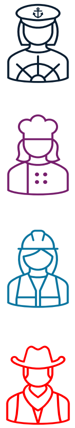
(i.e., Monte Carlo Estimator)

Goal:
Approximate
Training Point Attributions
without training N models

Empirical Influence Functions Feldman & Zhang (2020)

(i.e., Monte Carlo Estimator)

Data Set

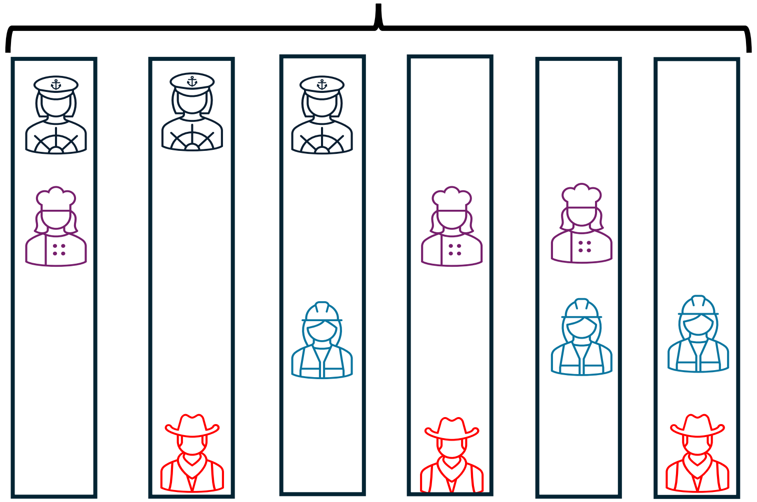


~

Subsample

Data Set

$K \ll N$ Subsets



Goal:

Approximate

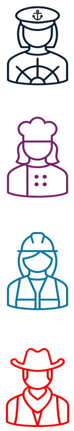
Training Point Attributions

without training N models

Empirical Influence Functions Feldman & Zhang (2020)

(i.e., Monte Carlo Estimator)

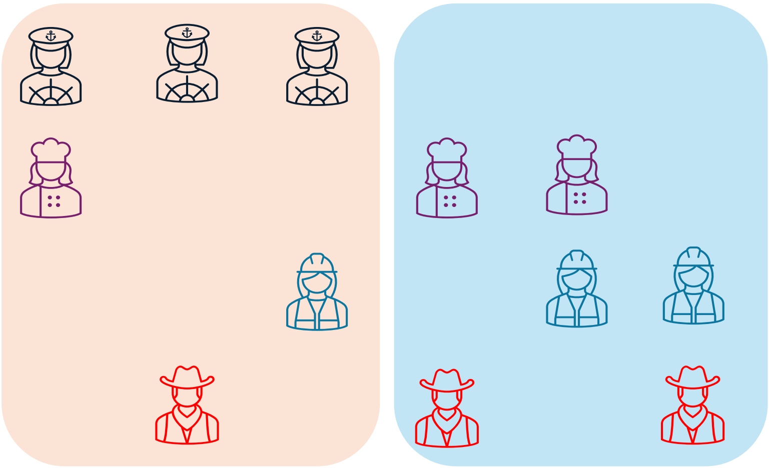
Data Set



~

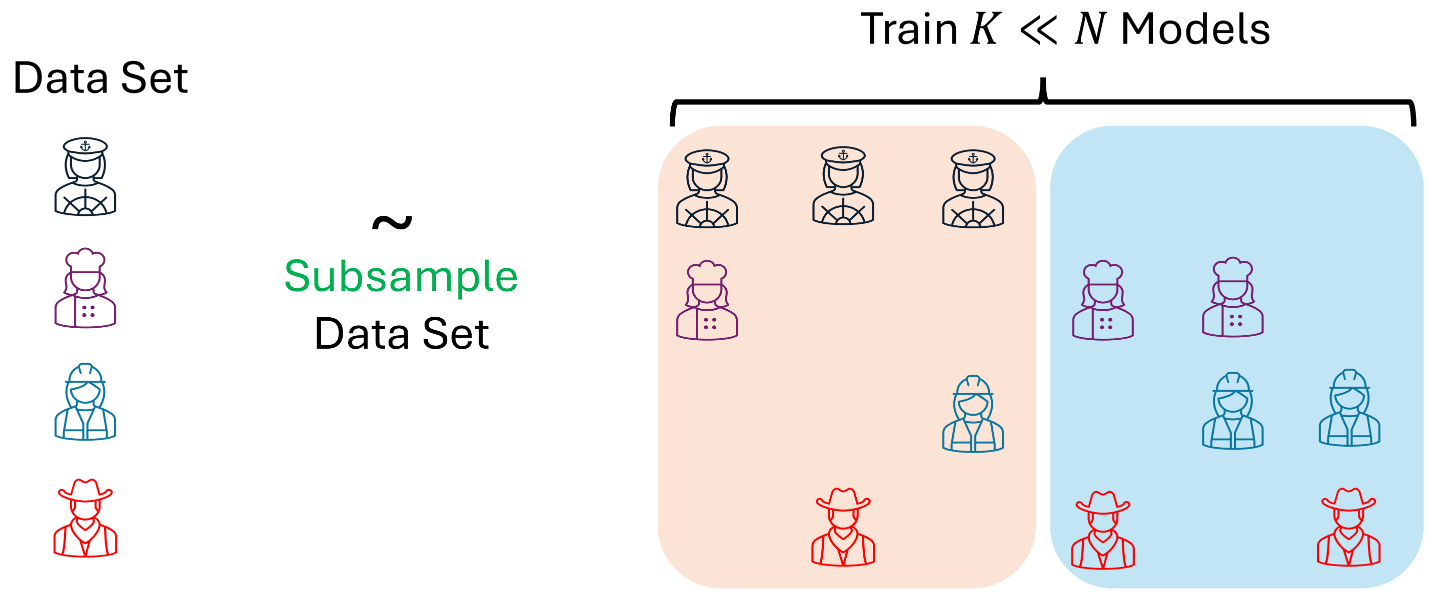
Subsample

Data Set



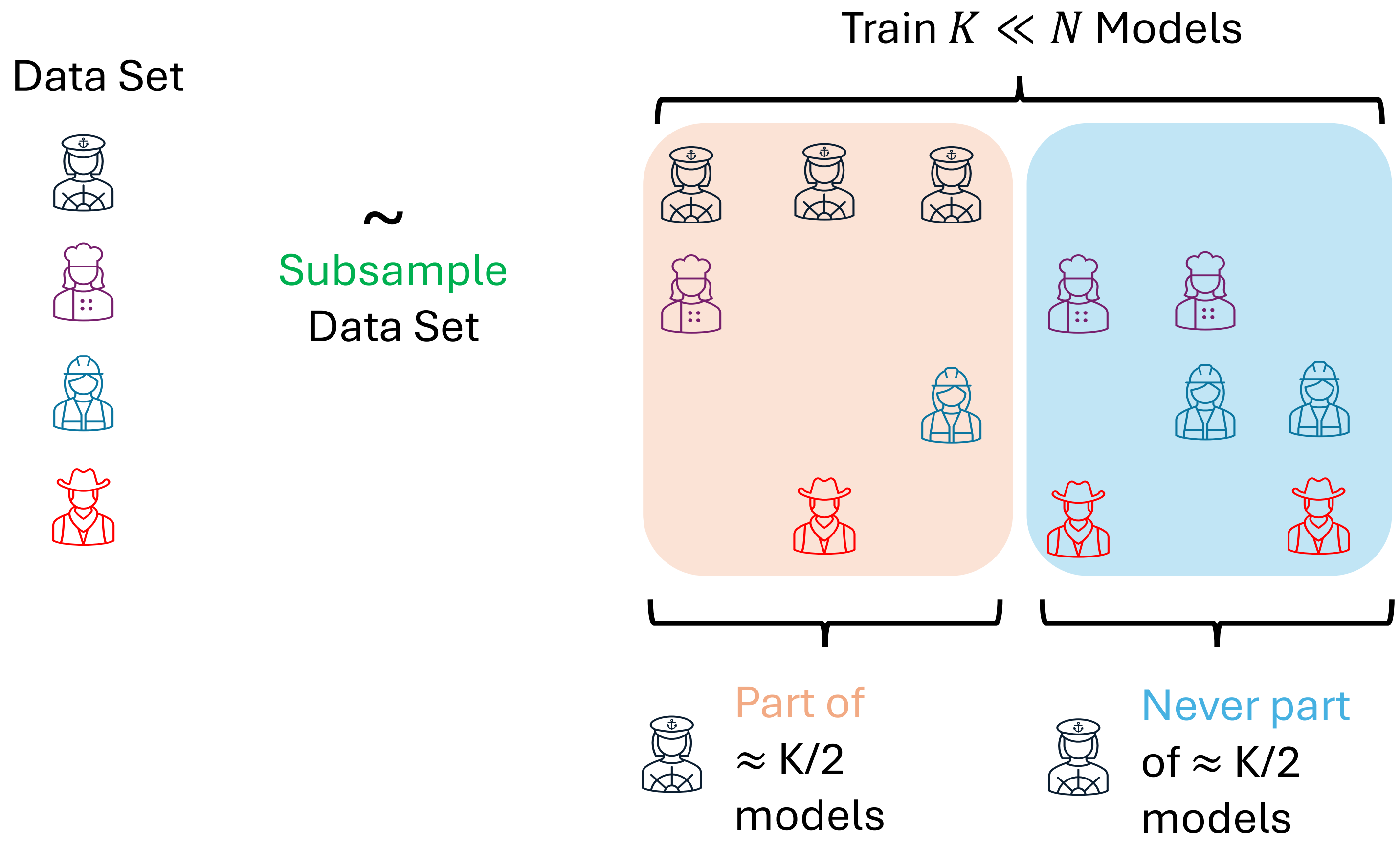
Empirical Influence Functions Feldman & Zhang (2020)

(i.e., Monte Carlo Estimator)



Empirical Influence Functions Feldman & Zhang (2020)

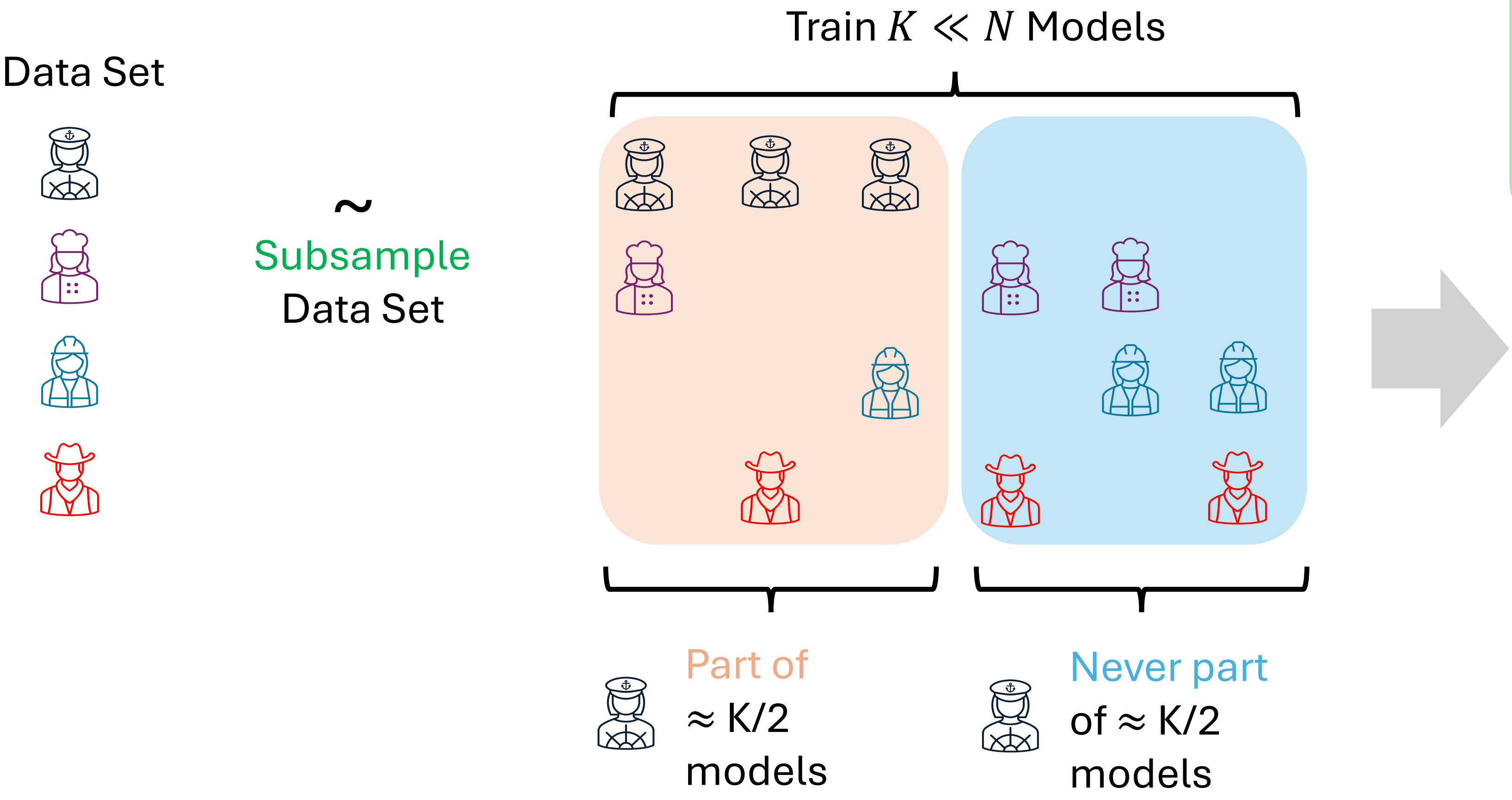
(i.e., Monte Carlo Estimator)



Empirical Influence Functions

Feldman & Zhang (2020)

(i.e., Monte Carlo Estimator)



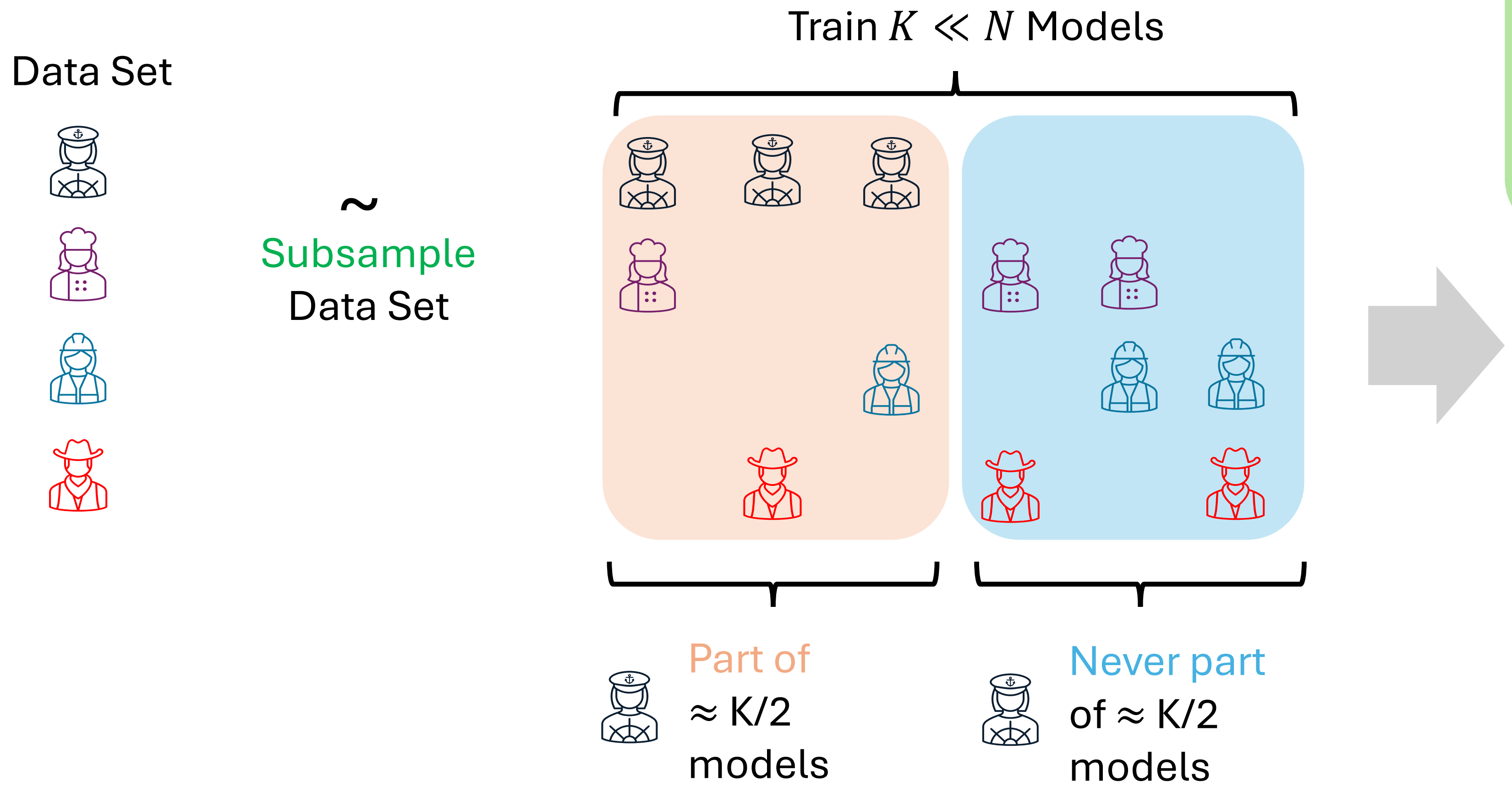
Importance of Bob's Data on test point prediction

$$\widehat{a_1} = \frac{1}{3}\sum_{j=1}^{3}\theta_j(x_{test}) - \frac{1}{3}\sum_{j=4}^{6}\theta'_j(x_{test})$$

Empirical Influence Functions

Feldman & Zhang (2020)

(i.e., Monte Carlo Estimator)



Importance of Bob's Data on test point prediction

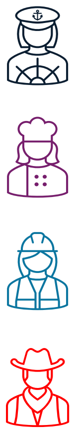
$$\widehat{a_1} = \frac{1}{3}\sum_{j=1}^{3} \theta_j(x_{test}) - \frac{1}{3}\sum_{j=4}^{6} \theta'_j(x_{test})$$

Compute a_i for all i $\in \{1, \dots, N\}$

Empirical Influence Functions Feldman & Zhang (2020)

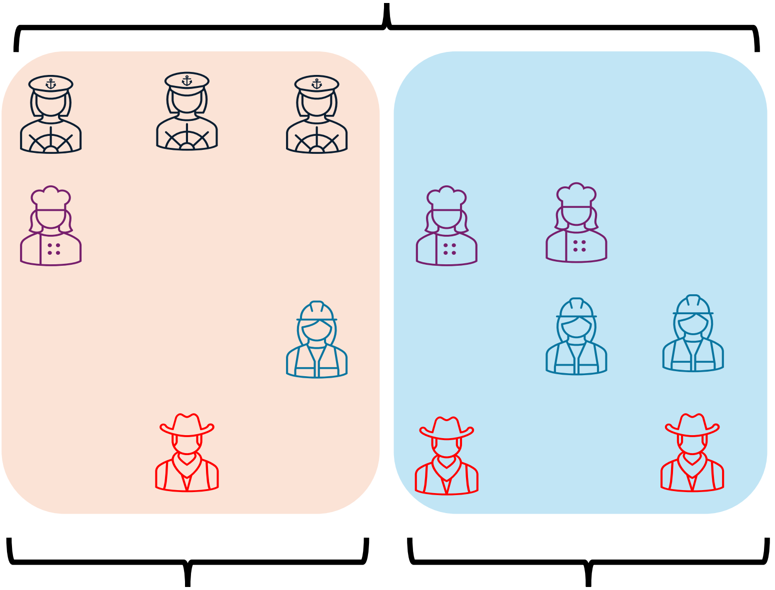
(i.e., Monte Carlo Estimator)

Data Set



$\sim$
Subsample
Data Set

Train $K \ll N$ Models



Part of $\approx$ K/2 models

Never part of $\approx$ K/2 models

Importance of Bob's Data on test point prediction

$$\widehat{a_1} = \frac{1}{3}\sum_{j=1}^{3}\theta_j(x_{test}) - \frac{1}{3}\sum_{j=4}^{6}\theta'_j(x_{test})$$

Compute a_i for all i $\in \{1, \dots, N\}$

Only requires K retrainings!

What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation

Vitaly Feldman * †
Apple

Chiyuan Zhang*
Google Research, Brain Team

Abstract

Deep learning algorithms are well-known to have a propensity for fitting the training data very well and often fit even outliers and mislabeled data points. Such fitting requires memorization of training data labels, a phenomenon that has attracted significant research interest but has not been given a compelling explanation so far. A recent work of Feldman [Fel19] proposes a theoretical explanation for this phenomenon based on a combination of two insights. First, natural image and data distributions are (informally) known to be long-tailed, that is have a significant fraction of rare and atypical examples. Second, in a simple theoretical model such memorization is necessary for achieving close-to-optimal generalization error when the data distribution is long-tailed. However, no direct empirical evidence for this explanation or even an approach for obtaining such evidence were given.

In this work we design experiments to test the key ideas in this theory. The experiments require estimation of the influence of each training example on the accuracy at each test example as well as memorization values of training examples. Estimating these quantities directly is computationally prohibitive but we show that closely-related *subsampled* influence and memorization values can be estimated much more efficiently. Our experiments demonstrate the significant benefits of memorization for generalization on several standard benchmarks. They also provide quantitative and visually compelling evidence for the theory put forth in [Fel19].

What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation

Vitaly Feldman * †
Apple

Chiyuan Zhang*
Google Research, Brain Team

Train 4K ResNet50 models & release attribution scores.

fit
on
rk
nts.
on
ng
ce

on
of the influence of each training example on the accuracy at each test example as well as memorization values of training examples. Estimating these quantities directly is computationally prohibitive but we show that closely-related *subsampled* influence and memorization values can be estimated much more efficiently. Our experiments demonstrate the significant benefits of memorization for generalization on several standard benchmarks. They also provide quantitative and visually compelling evidence for the theory put forth in [Fel19].

What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation

Vitaly Feldman [*] [†]
Apple

Chiyuan Zhang[*]
Google Research, Brain Team

Train 4K ResNet50 models & release attribution scores.

of the influence of each training example on the accuracy at each test example as well as memorization values of training examples. Estimating these quantities directly is computationally prohibitive but we show that closely-related *subsampled* influence and memorization values can be estimated much more efficiently. Our experiments demonstrate the significant benefits of memorization for generalization on several standard benchmarks. They also provide quantitative and visually compelling evidence for the theory put forth in [Fel19].

Datamodels: Predicting Predictions from Training Data

Andrew Ilyas [* 1] Sung Min Park [* 1] Logan Engstrom [* 1] Guillaume Leclerc [1] Aleksander Mądry [1]

Abstract

We present a conceptual framework, *datamodeling*, for analyzing the behavior of a model class in terms of the training data. For any fixed "target" example x, training set S, and learning algorithm, a *datamodel* is a parameterized function $2^S \to \mathbb{R}$ that for any subset of $S' \subset S$—using only information about which examples of S are contained in S'—predicts the outcome of training a model on S' and evaluating on x. Despite the complexity of the underlying process that is being approximated (e.g. end-to-end training and evaluation of deep neural networks), we show that even simple *linear* datamodels successfully predict model outputs. We then demonstrate that datamodels give rise to a variety of applications, such as: accurately predicting the effect of dataset counterfactuals; identifying brittle predictions; finding semantically similar examples; quantifying train-test leakage; and embedding data into a well-behaved and feature-rich *representation space*.

puts a trained model. This learning algorithm need not be deterministic—for example, $\mathcal{A}$ might encode the process of training a neural network from random initialization.

Now, consider a *fixed* target example x and define

$$f_{\mathcal{A}}(x; S) := \text{the outcome of training a model on } S \text{ using } \mathcal{A}\text{, and evaluating it on the input } x, \quad (1)$$

where we leave "outcome" intentionally broad to capture a variety of settings that one might care about. For example, $f_{\mathcal{A}}(x; S)$ may be the cross-entropy loss of a classifier on x, or the error of a regression model on x. The potential stochasticity of $\mathcal{A}$ means $f_{\mathcal{A}}(x; S)$ is a random variable.

Goal. Broadly, we aim to understand how the training examples in S combine through the learning algorithm $\mathcal{A}$ to yield $f_{\mathcal{A}}(x; S)$ (again, for the *specific* example x that we are examining). Towards this goal, we will leverage a classic technique for studying complex black-box functions: *surrogate modeling* (Sacks et al., 1989). In surrogate modeling, one replaces complex functions with inexact but significantly easier-to-analyze approximations, then uses the

What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation

Vitaly Feldman * † — Apple
Chiyuan Zhang* — Google Research, Brain Team

Train 4K ResNet50 models & release attribution scores.

of the influence of each training example on the accuracy at each test example as well as memorization values of training examples. Estimating these quantities directly is computationally prohibitive but we show that closely-related *subsampled* influence and memorization values can be estimated much more efficiently. Our experiments demonstrate the significant benefits of memorization for generalization on several standard benchmarks. They also provide quantitative and visually compelling evidence for the theory put forth in [Fel19].

Datamodels: Predicting Predictions from Training Data

Andrew Ilyas [*1] Sung Min Park [*1] Logan Engstrom [*1] Guillaume Leclerc [1] Aleksander Mądry [1]

Train 1.5M ResNet9 models & release attribution scores.

show that even simple *linear* datamodels successfully predict model outputs. We then demonstrate that datamodels give rise to a variety of applications, such as: accurately predicting the effect of dataset counterfactuals; identifying brittle predictions; finding semantically similar examples; quantifying train-test leakage; and embedding data into a well-behaved and feature-rich *representation space*.

stochasticity of $\mathcal{A}$ means $f_{\mathcal{A}}(x; S)$ is a random variable.

Goal. Broadly, we aim to understand how the training examples in S combine through the learning algorithm $\mathcal{A}$ to yield $f_{\mathcal{A}}(x; S)$ (again, for the *specific* example x that we are examining). Towards this goal, we will leverage a classic technique for studying complex black-box functions: *surrogate modeling* (Sacks et al., 1989). In surrogate modeling, one replaces complex functions with inexact but significantly easier-to-analyze approximations, then uses the

What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation

Vitaly Feldman * † — Apple
Chiyuan Zhang* — Google Research, Brain Team

Train 4K ResNet50 models & release attribution scores.

of the influence of each training example on the accuracy at each test example as well as memorization values of training examples. Estimating these quantities directly is computationally prohibitive but we show that closely-related *subsampled* influence and memorization values can be estimated much more efficiently. Our experiments demonstrate the significant benefits of memorization for generalization on several standard benchmarks. They also provide quantitative and visually compelling evidence for the theory put forth in [Fel19].

Datamodels: Predicting Predictions from Training Data

Andrew Ilyas [*1] Sung Min Park [*1] Logan Engstrom [*1] Guillaume Leclerc [1] Aleksander Mądry [1]

Fundamental Insight

$$f \approx \boldsymbol{a}^\top \boldsymbol{x}$$

where $f: \{0,1\}^N \to \mathbb{R}$

show that even simple *linear* datamodels successfully predict model outputs. We then demonstrate that datamodels give rise to a variety of applications, such as: accurately predicting the effect of dataset counterfactuals; identifying brittle predictions; finding semantically similar examples; quantifying train-test leakage; and embedding data into a well-behaved and feature-rich *representation space*.

stochasticity of $\mathcal{A}$ means $f_\mathcal{A}(x; S)$ is a random variable.

Goal. Broadly, we aim to understand how the training examples in S combine through the learning algorithm $\mathcal{A}$ to yield $f_\mathcal{A}(x; S)$ (again, for the *specific* example x that we are examining). Towards this goal, we will leverage a classic technique for studying complex black-box functions: *surrogate modeling* (Sacks et al., 1989). In surrogate modeling, one replaces complex functions with inexact but significantly easier-to-analyze approximations, then uses the

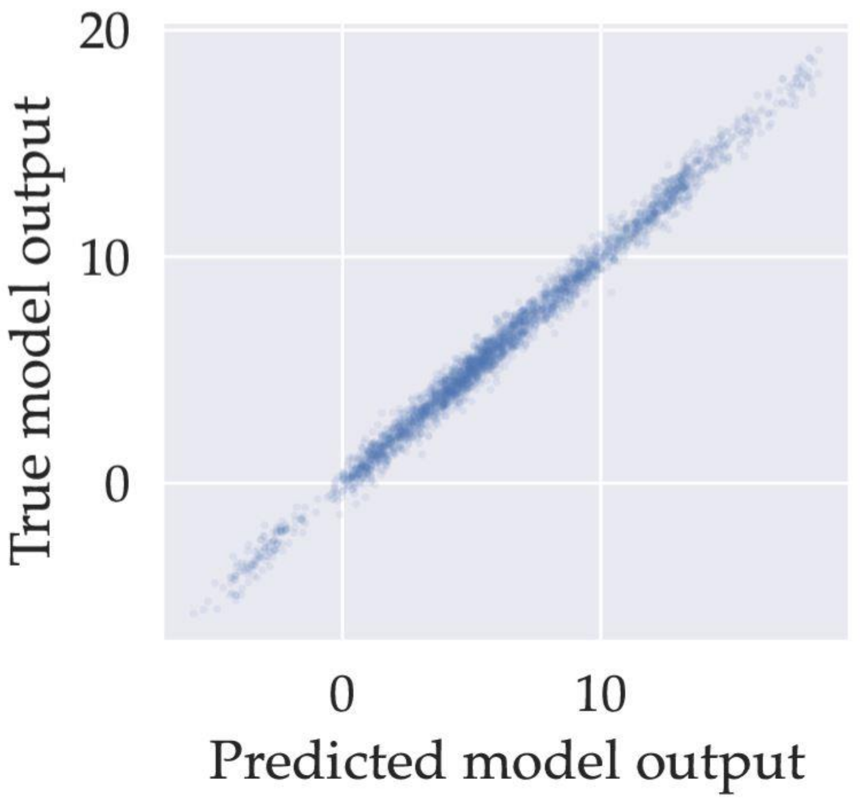
What Neural Networks Memorize and Why:
Discovering the Long Tail via Influence Estimation

Vitaly Feldman * † — Apple
Chiyuan Zhang* — Google Research, Brain Team

Train 4K ResNet50 models & release attribution scores.

of the influence of each training example on the accuracy at each test example as well as memorization values of training examples. Estimating these quantities directly is computationally prohibitive but we show that closely-related *subsampled* influence and memorization values can be estimated much more efficiently. Our experiments demonstrate the significant benefits of memorization for generalization on several standard benchmarks. They also provide quantitative and visually compelling evidence for the theory put forth in [Fel19].

Datamodels: Predicting Predictions from Training Data



What Neural Networks Memorize and Why: Discovering the Long Tail via Influence Estimation

Vitaly Feldman * †
Apple

Chiyuan Zhang*
Google Research, Brain Team

Datamodels: Predicting Predictions from Training Data

Andrew Ilyas * 1 Sung Min Park * 1 Logan Engstrom * 1 Guillaume Leclerc 1 Aleksander Mądry 1

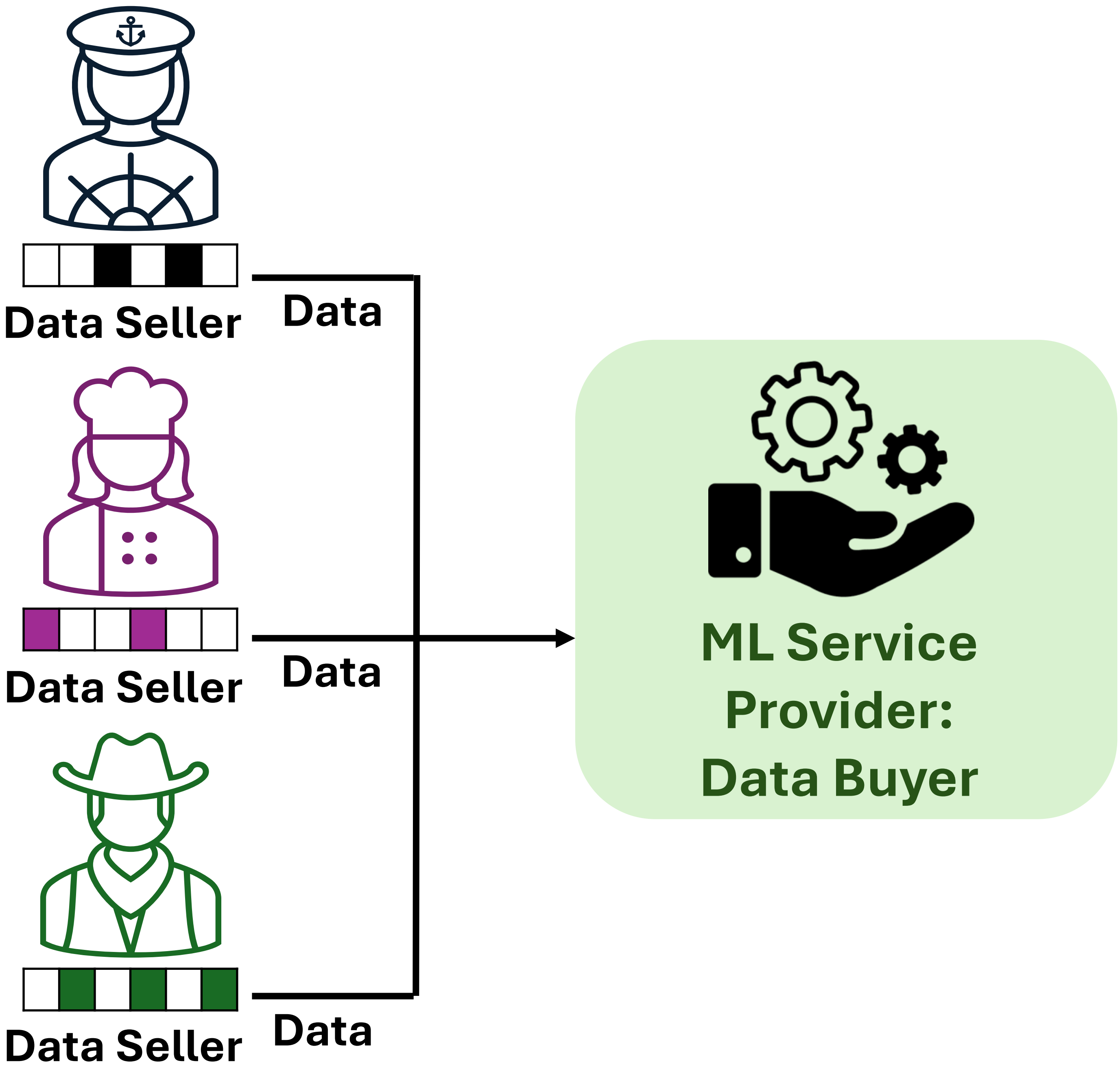
Research Question

Can we effectively verify correctness of released attribution scores using a small number of training runs?

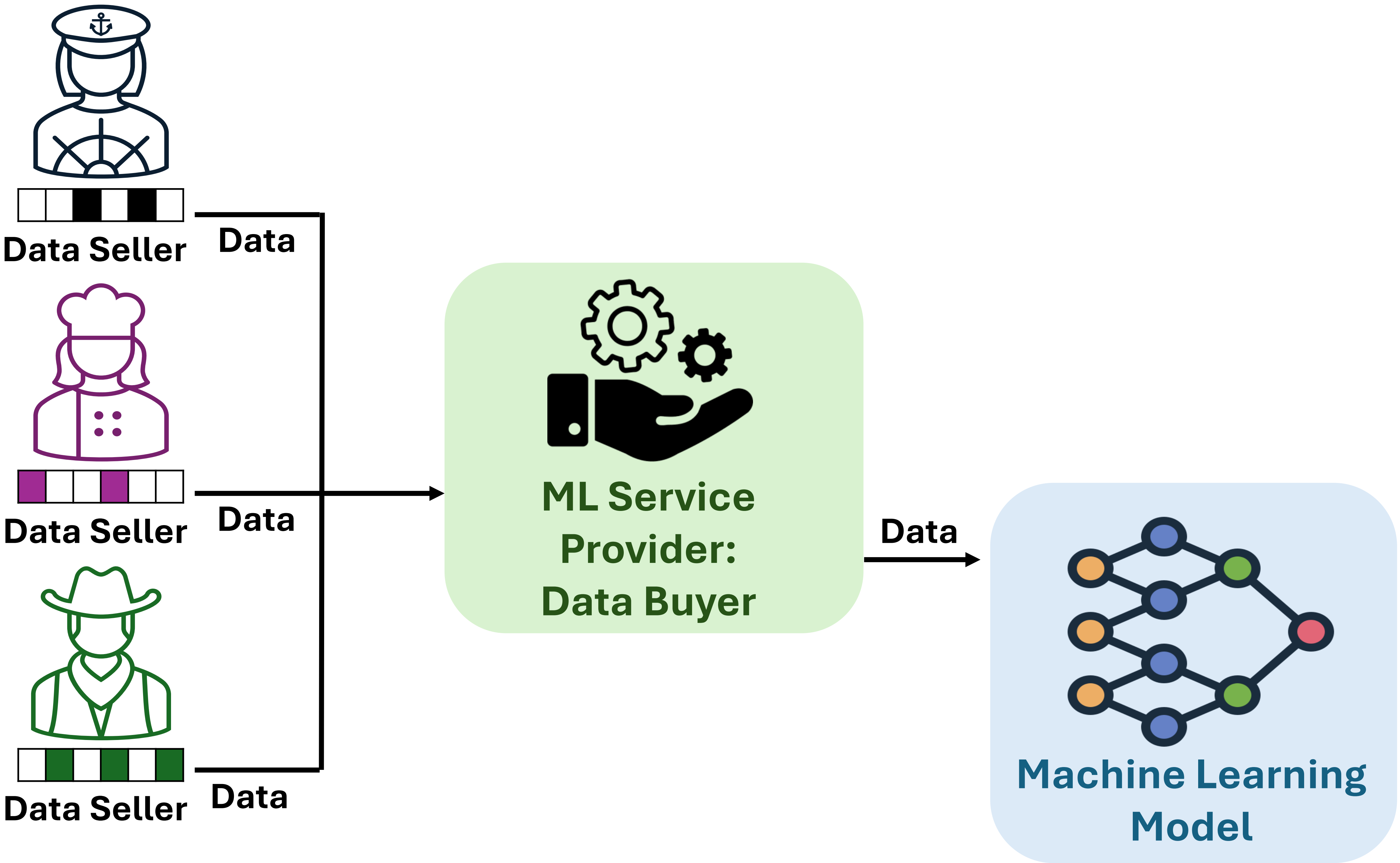
quantitative and visually compelling evidence for the theory put forth in [Fel19].

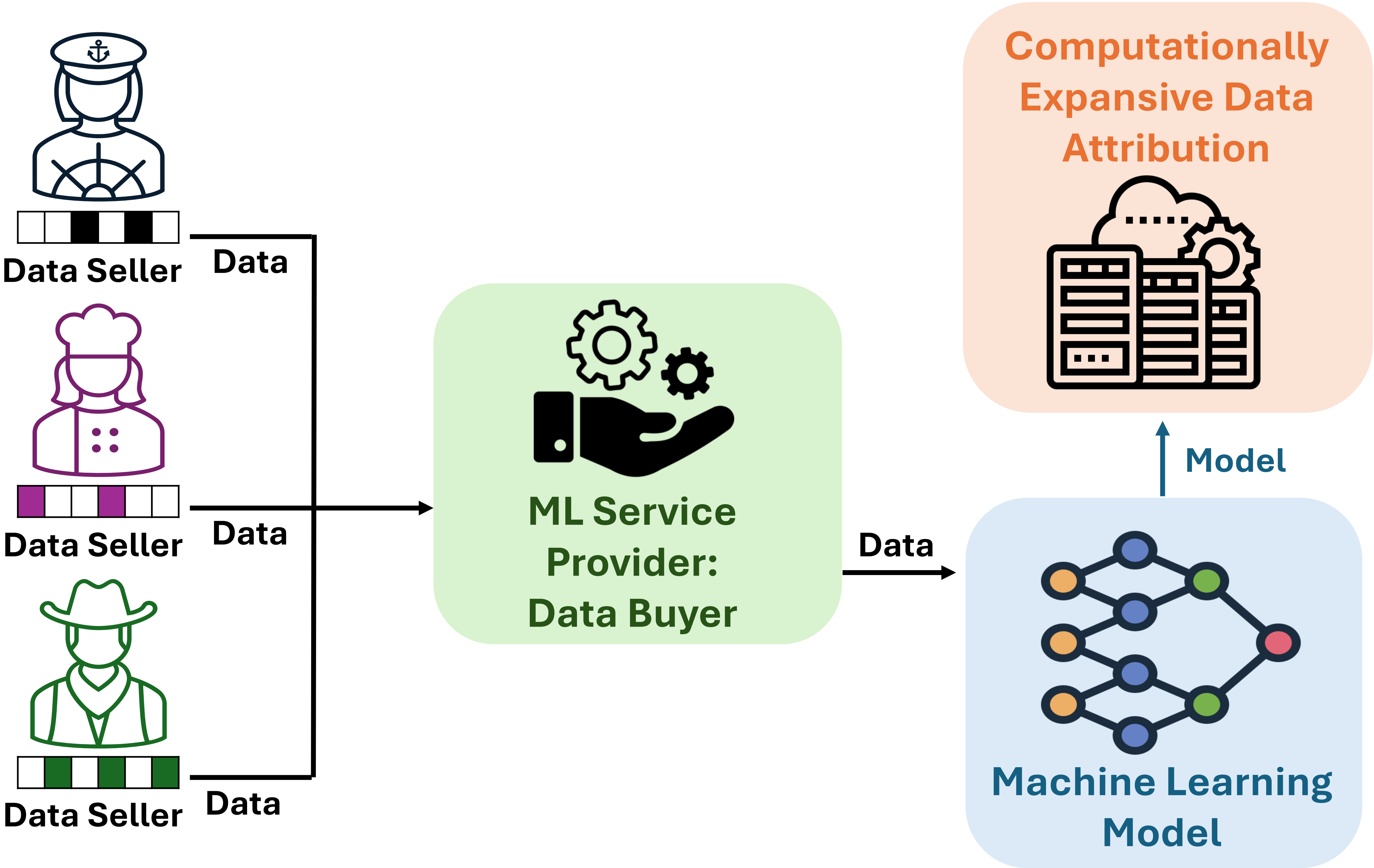
predictions; finding semantically similar examples; quantifying train-test leakage; and embedding data into a well-behaved and feature-rich *representation space*.

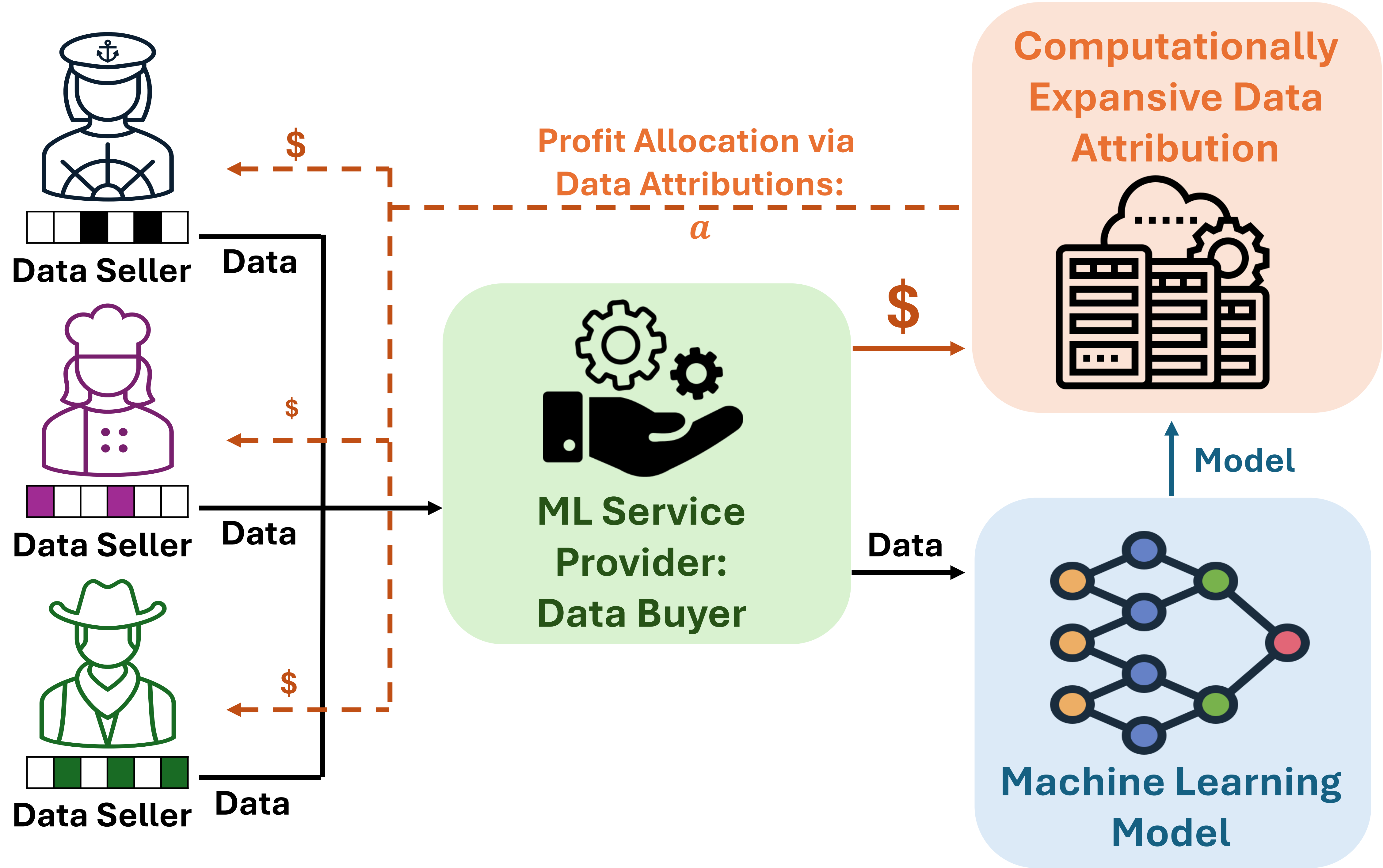
are examining). Towards this goal, we will leverage a classic technique for studying complex black-box functions: *surrogate modeling* (Sacks et al., 1989). In surrogate modeling, one replaces complex functions with inexact but significantly easier-to-analyze approximations, then uses the



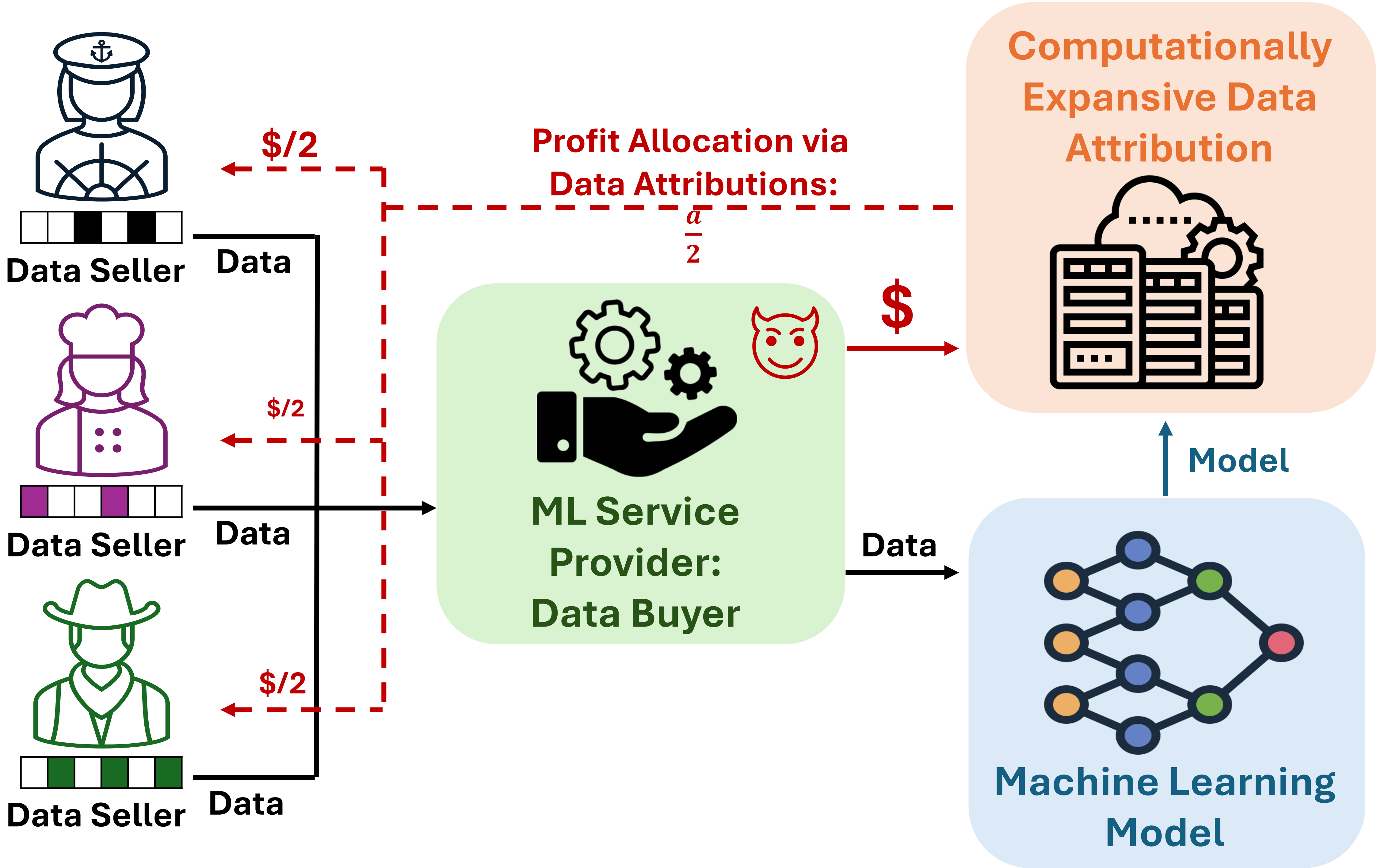
Data Seller
Data
Data Seller
Data
Data Seller
Data
ML Service Provider: Data Buyer


Data Seller
Data
Data Seller
Data
Data Seller
Data
ML Service Provider: Data Buyer
Data
Machine Learning Model


Computationally Expansive Data Attribution
Model
Data Seller
Data
Data Seller
Data
Data Seller
Data
ML Service Provider: Data Buyer
Data
Machine Learning Model



Computationally Expansive Data Attribution
Profit Allocation via Data Attributions:
a
$
$
$
$
Data Seller
Data
Data Seller
Data
Data Seller
Data
ML Service Provider: Data Buyer
Data
Model
Machine Learning Model



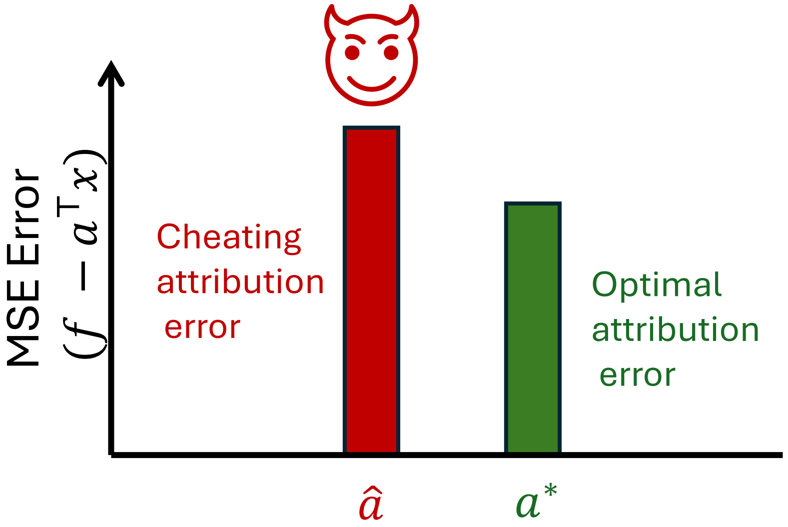
Computationally Expansive Data Attribution
Profit Allocation via Data Attributions: $\frac{a}{2}$
$/2
$/2
$/2
$
Data Seller
Data
Data Seller
Data
Data Seller
Data
ML Service Provider: Data Buyer
Data
Model
Machine Learning Model

Why Naive Verification Fails?

- A naive data seller might just check if attribution $\hat{a}$ has low error
- „Good enough“ isn‘t good enough! Checking if attribution has low error can easily be fooled.
- What if a malicous buyer computes $\hat{a} = \frac{a^*}{2}$. This has low MSE, but is far from optimal!

Why Naive Verification Fails?

- A naive data seller might just check if attribution $\hat{a}$ has low error
- „Good enough“ isn‘t good enough! Checking if attribution has low error can easily be fooled.
- What if a malicous buyer computes $\hat{a} = \frac{a^*}{2}$. This has low MSE, but is far from optimal!

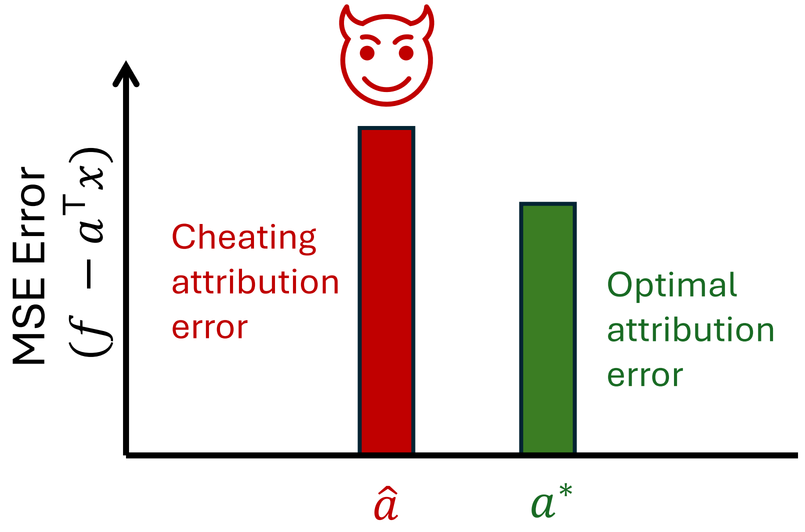


Verifying Near-Optimality

Check if the ML providers‘s answer is close to best possible answer

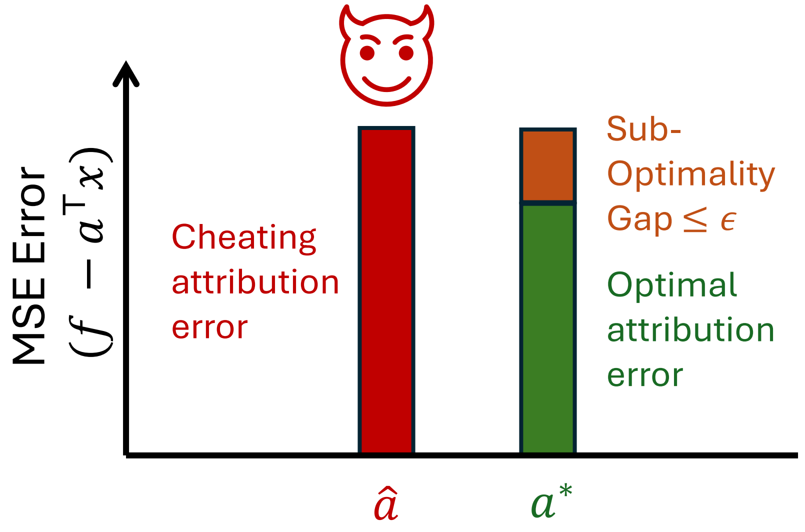
Verifying Near-Optimality

Check if the ML providers‘s answer is close to best possible answer



Verifying Near-Optimality

Check if the ML providers‘s answer is close to best possible answer

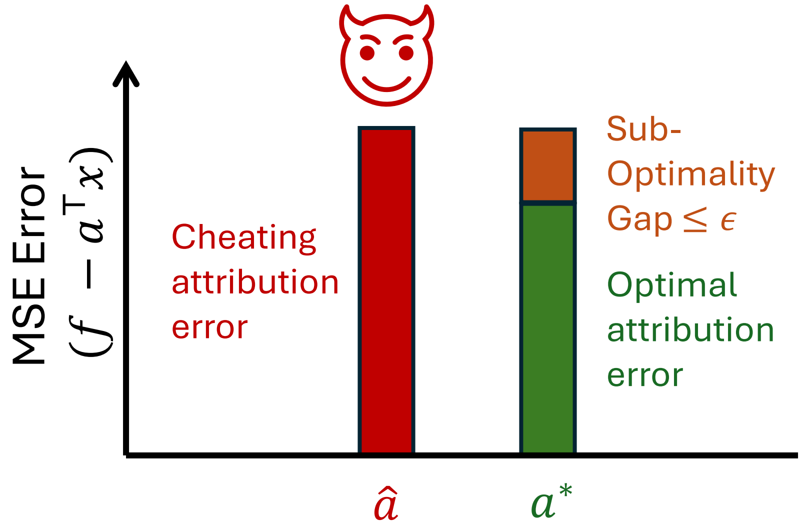


Verifying Near-Optimality

Check if the ML providers‘s answer is close to best possible answer

Sub-Optimality Gap

$$\underbrace{MSE(f, \hat{a}^\top x)}_{\textbf{Task 1}} - \underbrace{MSE\left(f, {a^*}^\top x\right)}_{\textbf{Task 2}} \leq \epsilon$$

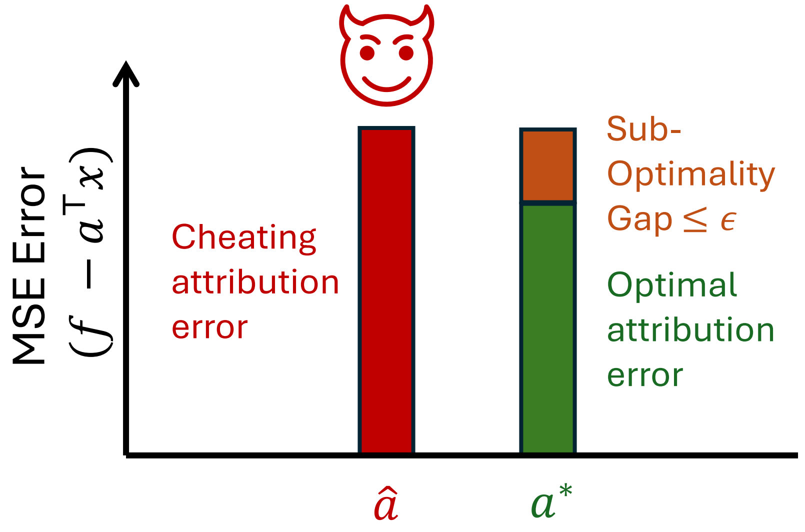


Verifying Near-Optimality

Check if the ML providers's answer is close to best possible answer

Sub-Optimality Gap

$$\underbrace{MSE(f, \hat{a}^\top x)}_{\textbf{Task 1}} - \underbrace{MSE\left(f, {a^*}^\top x\right)}_{\textbf{Task 2}} \leq \epsilon$$



How can the data seller measure this gap without being powerful enough to compute optimal a^*?

The Rules of the Game

Prover (P)

ML Service Provider

Computationally powerful,
provides the attribution scores.

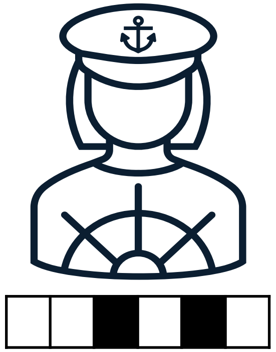
The Rules of the Game

Prover (P)

ML Service Provider

Computationally powerful,
provides the attribution scores.

Verifier (V)



Data Seller

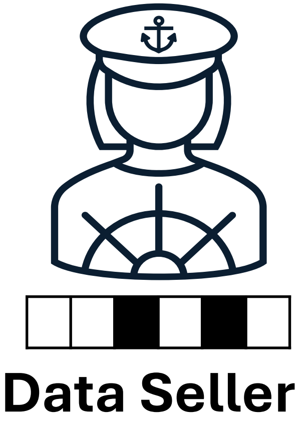
Resource-constrained,
wants to check **P**'s work.

The Rules of the Game

Prover (P)

Computationally powerful, provides the attribution scores.

Verifier (V)



Resource-constrained, wants to check **P**'s work.

Good Protocol

✅ **Completeness:** If everyone is honest, **V**erifier accepts the correct answer.

🛡️ **Soundness:** If the **P**rover cheats, the **V**erifier either detects it and aborts, or still gets a correct answer.

⚡ **Efficiency:** The **V**erifier's work is cheap and, crucially, independent of the dataset size N.

The Non-Interactive Protocol

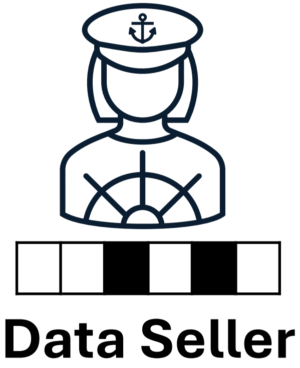
The Verifier can check for near-optimality himself, but it‘s slow

Prover (P)

Computes attribution scores

$\hat{a}$

Verifier (V)



Checks attributions

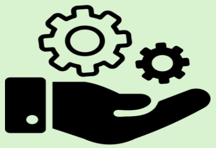
Set error tolerance: ϵ (say $\epsilon = 0.01$)

Task 1: Estimate the error of $\hat{a}$ by training $O(\frac{1}{\epsilon^2})$ models → 10.000 runs

The Non-Interactive Protocol

The Verifier can check for near-optimality himself, but it's slow

Prover (P)

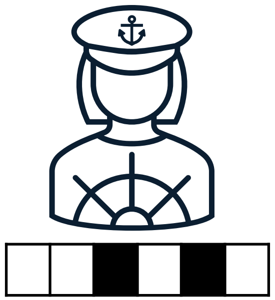


ML Service Provider

Computes attribution scores

$\hat{a}$

Verifier (V)



Data Seller

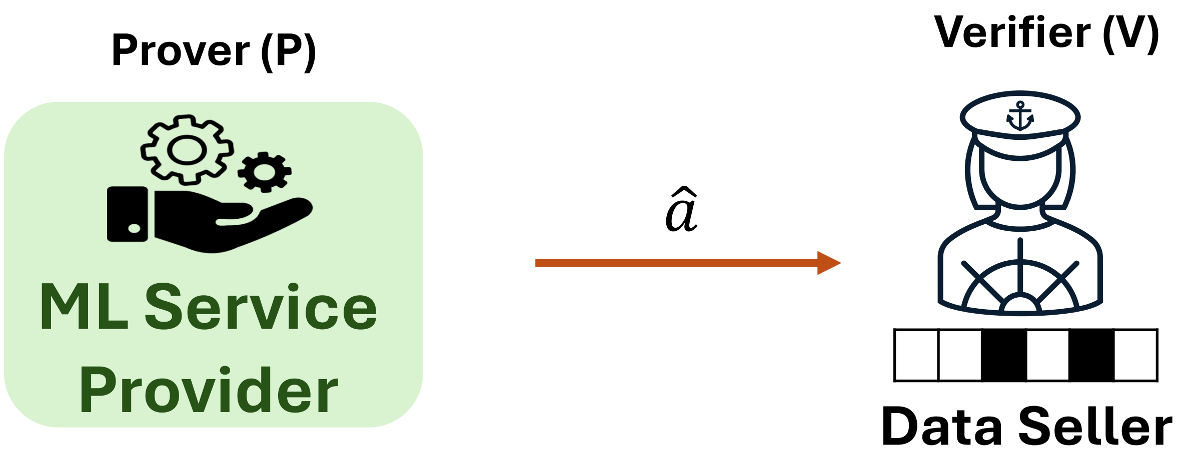
Set error tolerance: ϵ (say $\epsilon = 0.01$)

Task 1: Estimate the error of $\hat{a}$ by training $O(\frac{1}{\epsilon^2})$ models → 10.000 runs

Task 2: Estimate the optimal error by training $O(\frac{1}{\epsilon^3})$ models using a clever „residual estimation" technique by Saunshi et al (2022) → 1.000.000 runs

The Non-Interactive Protocol

The Verifier can check for near-optimality himself, but it‘s slow



Set error tolerance: ϵ (say $\epsilon = 0.01$)

Task 1: Estimate the error of $\hat{a}$ by training $O(\frac{1}{\epsilon^2})$ models → 10.000 runs

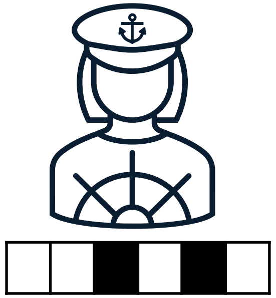
Task 2: Estimate the optimal error by training $O(\frac{1}{\epsilon^3})$ models using a clever „residual estimation“ technique by Saunshi et al (2022) → 1.000.000 runs

The Verifier's total cost is dominated by the expensive task 2:

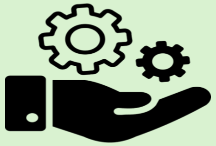
Total # of training runs on the order of $O(\frac{1}{\epsilon^3})$

The Interactive Protocol

Verifier (V)



Data Seller



ML Service Provider

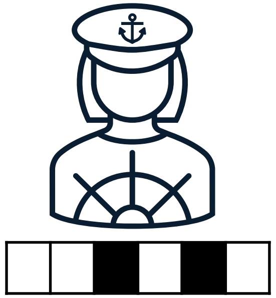
Prover (P)

Set error tolerance: ϵ

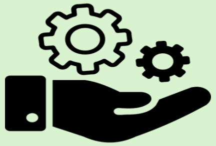
Flag small random subset of size $O(\frac{1}{\epsilon^2})$ as spot checks

The Interactive Protocol

Verifier (V)



Data Seller



ML Service Provider

Prover (P)

Set error tolerance: ϵ

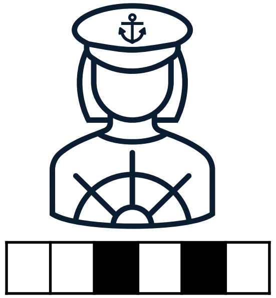
Flag small random subset of size $O(\frac{1}{\epsilon^2})$ as spot checks

Challenge →

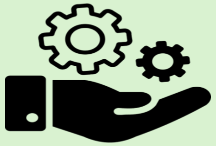
Task 2: Estimate the optimal error by training models on the order of $O(\frac{1}{\epsilon^3})$ using the clever „residual estimation“ technique

The Interactive Protocol

Verifier (V)



Data Seller



ML Service Provider

Prover (P)

Set error tolerance: ϵ

Flag small random subset of size $O(\frac{1}{\epsilon^2})$ as spot checks

Challenge →

Task 2: Estimate the optimal error by training models on the order of $O(\frac{1}{\epsilon^3})$ using the clever „residual estimation“ technique

← **Response**

Do $O(\frac{1}{\epsilon^2})$ spot checks.
If any do not match, **abort**

Task 1: Run consistency check.

Conclusion

- Verifying computationally expensive data attributions is a key challenge for building trust between Data Buyers & Data Sellers
- We need a protocol that is **correct** (completeness & soundness) and **efficient** for the resource-constrained Data Buyer
- We suggested a simple two-message **interactive protocol** with a **spot-checking** mechanism
- Approach **offloads heavy computational burden to the Data Buyer** while maintaining strong guarantees
- Interaction makes things efficient!