Towards A Platform and Benchmark Suite for Model Training on Dynamic Datasets

Maximilian Böther, Foteini Strati, Viktor Gsteiger, and Ana Klimovic

8th of May 2023

EuroMLSys 2023


Sally's Baking Addiction

We need to update our production models!

How much does retraining really matter?

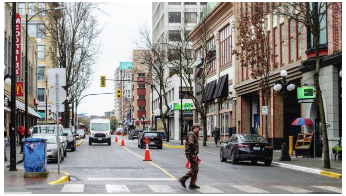
Retraining Method	Purchase Through Rate Increase
No Retraining	0
Weekly Retraining	+2.5%
Daily Retraining	+20.3%

[Online Learning for Recommendations at GrubHub (Egg 2021)]

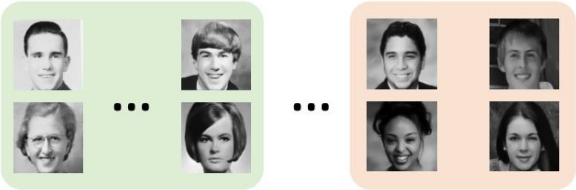
Dynamic Datasets

::= datasets that evolve over time

i.e., data points get added or removed from the set



More data collected



Data *shifts*

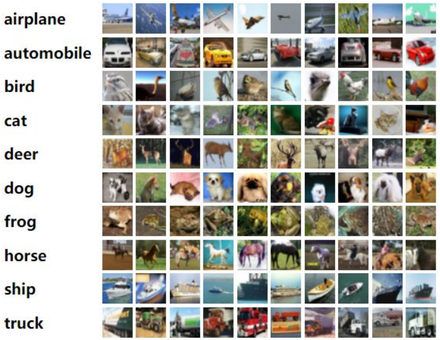
Art. 17 GDPR

Right to erasure (‘right to be forgotten’)

Data *deletion*

What datasets do we use in ML research?

MNIST



CIFAR

ImageNet

Gap between research and practice!

Based on our discussions with industry...

...frequent model retraining or finetuning is common.

But practitioners seem to choose these training hyperparameters (when to train, retrain vs finetune, which data to use, ...) **ad hoc**!

The Costs of Model Retraining

Cost of Retraining

~ Number of Samples

~ Number of Trainings

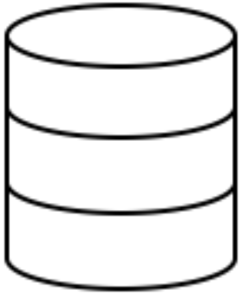
How can we lower the cost of updating production models on dynamic datasets?

Dimensions of Training

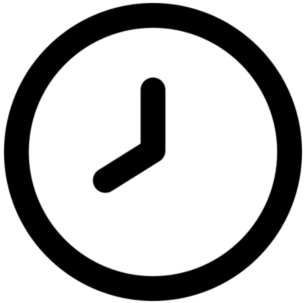
When do we trigger a training?

What data do we train on?

When to trigger training?



Amount-Based



Time-Based

Drift-Based

What data do we train on?

Can we train on less data, but identify *important* data such that we get *similar* model *accuracy* while *reducing compute*?

What data do we train on?

Can we train on less data, but identify ***important*** data such that we get *similar* model *accuracy* while *reducing compute*?

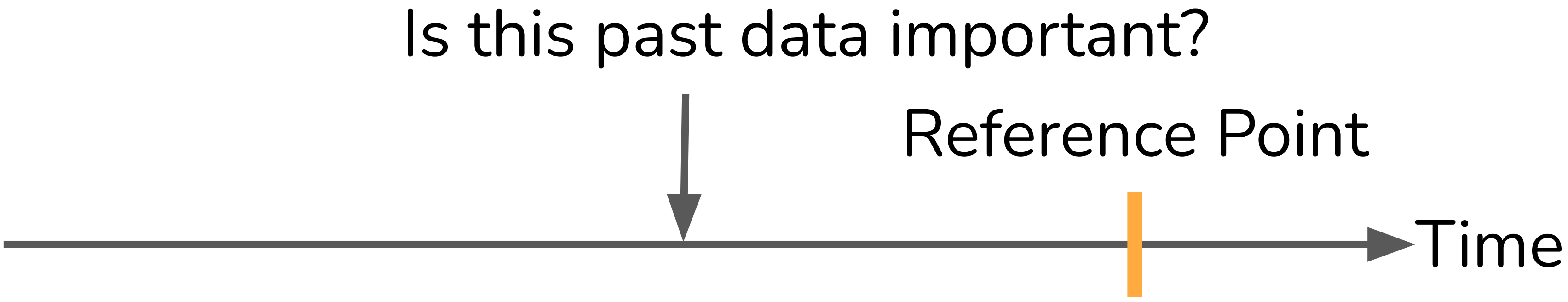
What data do we train on?

What makes data important?

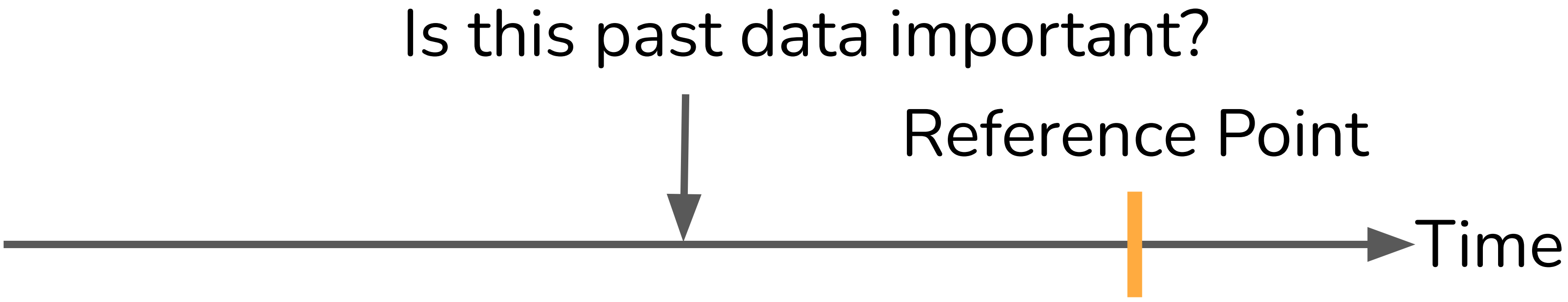
What data do we train on?

Is this past data important?

What data do we train on?

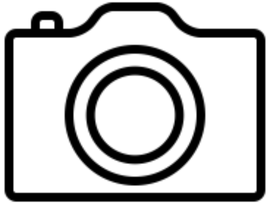


What data do we train on?



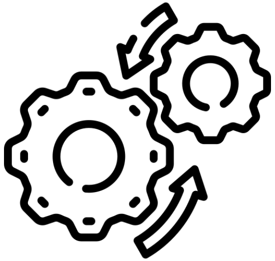
Importance might depend on recency, but not necessarily!

Prior Work: Approaches to Data Importance



Policies for
Static Datasets

- Coresets
- Data Distillation/Valuation
- ….



Policies for
Dynamic Datasets

- Continual Learning

How well do data selection policies work in practical scenarios?

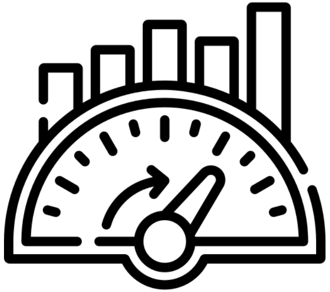
We don’t know.

Systems@ETH Zürich

What do we need in order to find out?

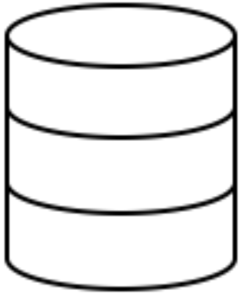
Open Source Platform with:

- pluggable training/selection policies
- dynamic dataset management
- training job orchestration



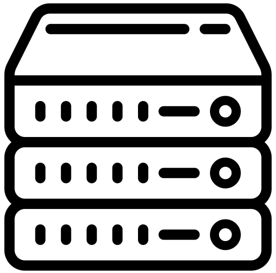
Representative Benchmarking Suite

Why do we need a platform?



Billions of samples

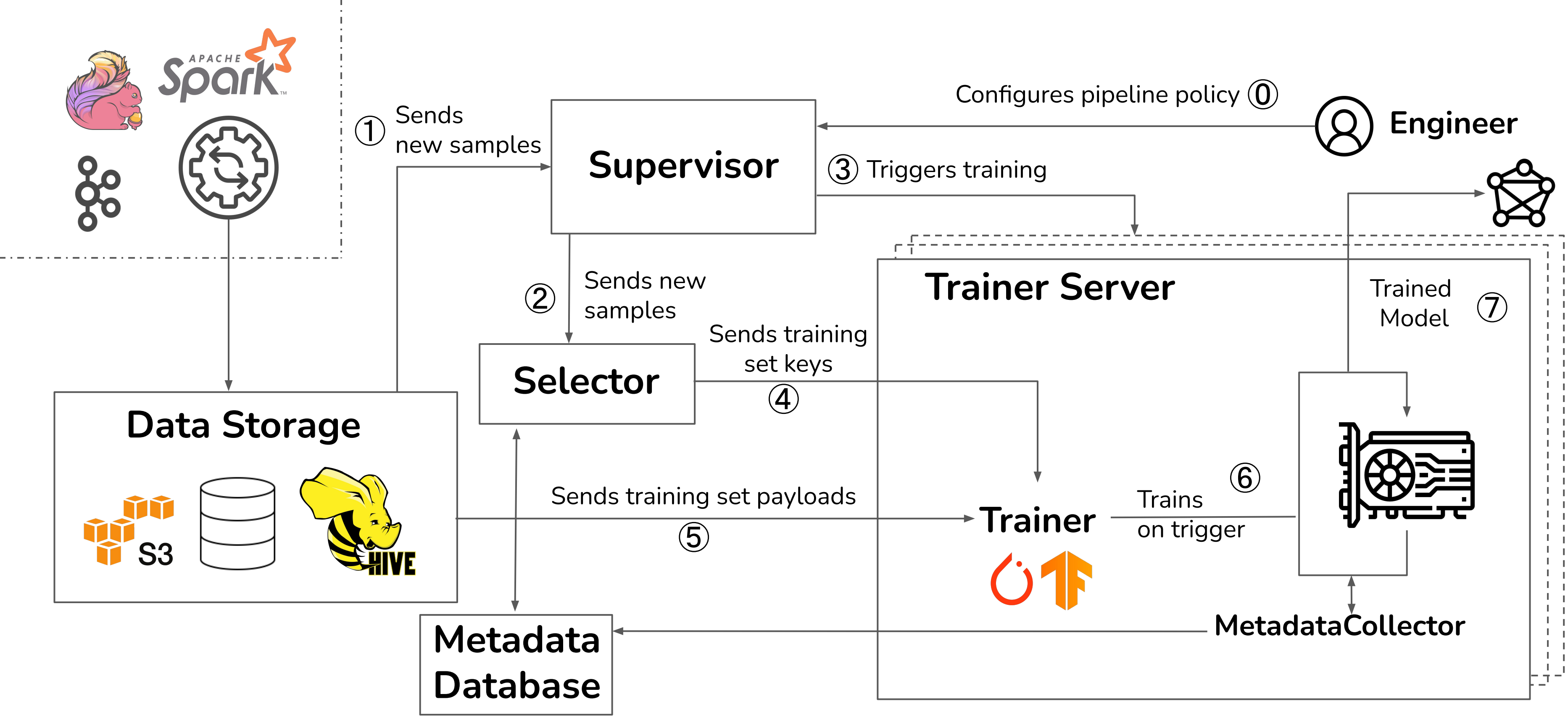
Orchestration is non-trivial

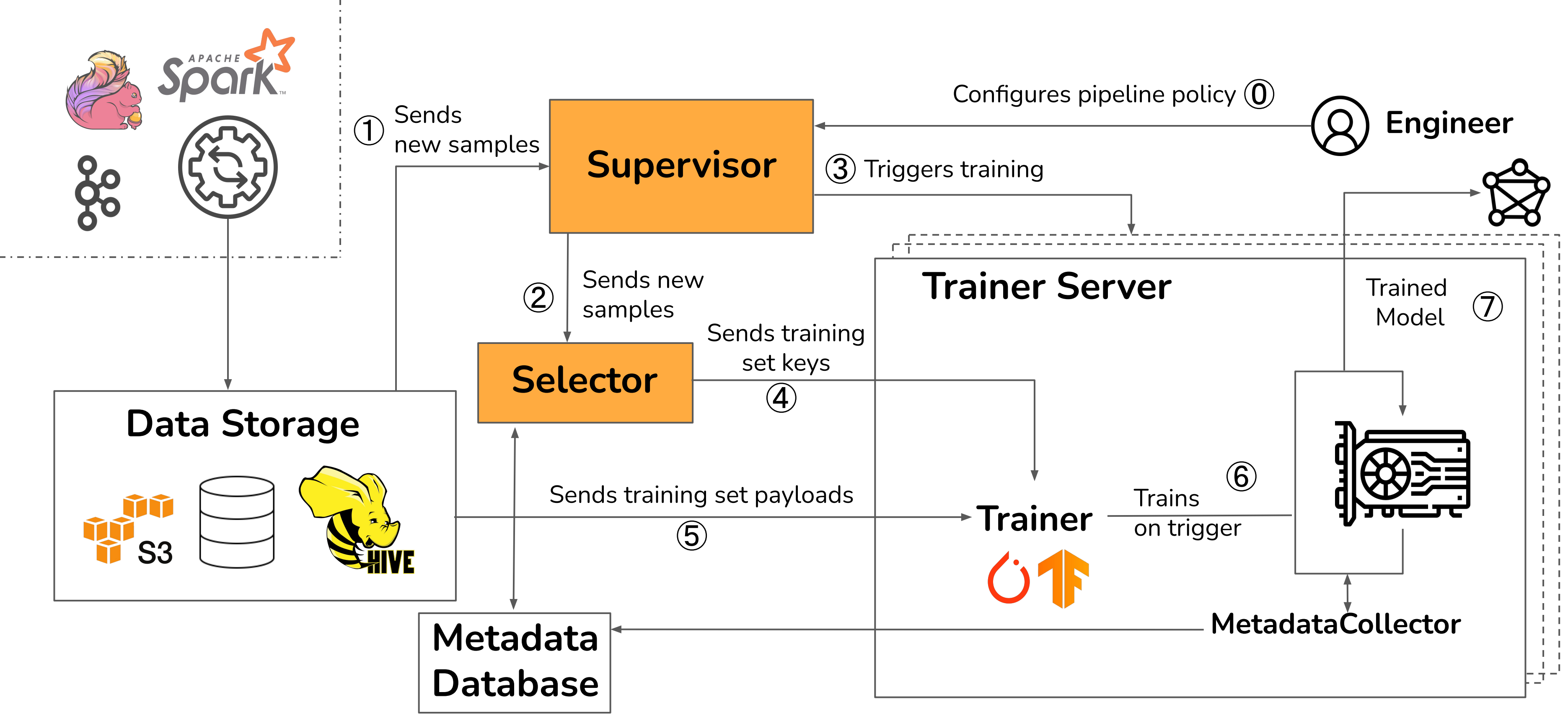


Enable systems optimizations

Systems@ETH Zürich

modyn


APACHE Spark
① Sends new samples
Supervisor
Configures pipeline policy ⓪
Engineer
③ Triggers training
② Sends new samples
Selector
Sends training set keys
④
Trainer Server
Trained Model
⑦
Data Storage
S3
HIVE
Sends training set payloads
⑤
Trainer
Trains on trigger
⑥
MetadataCollector
Metadata Database

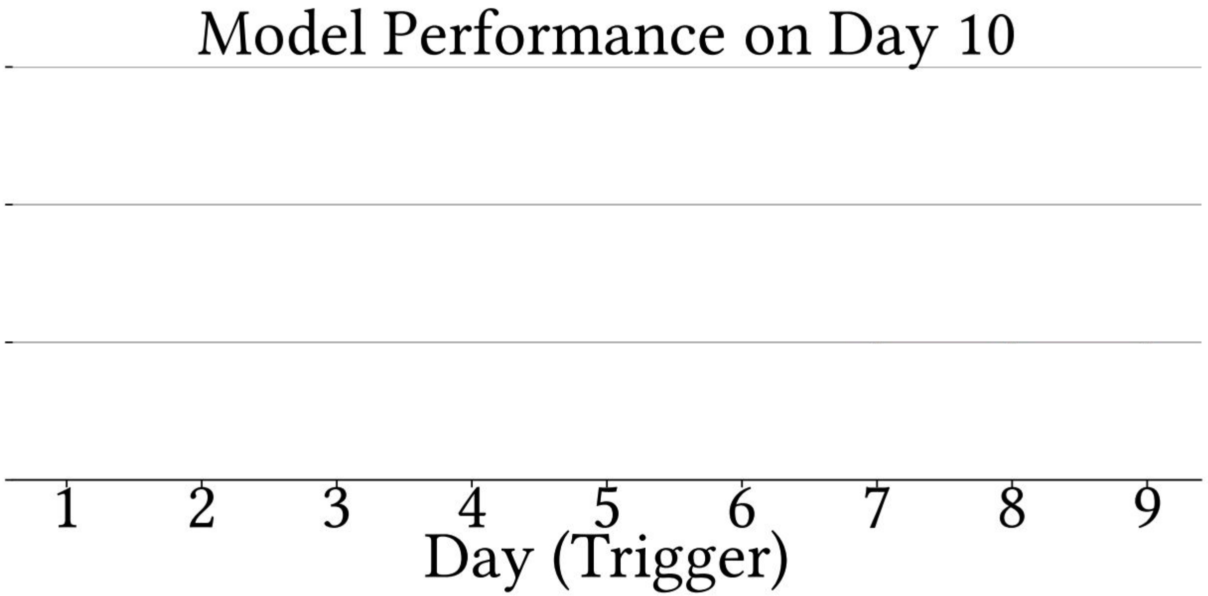


Configures pipeline policy ⓪
Engineer
① Sends new samples
Supervisor
③ Triggers training
② Sends new samples
Selector
Sends training set keys
④
Trainer Server
Trained Model
⑦
Data Storage
S3
HIVE
Sends training set payloads
⑤
Trainer
Trains on trigger
⑥
MetadataCollector
Metadata Database

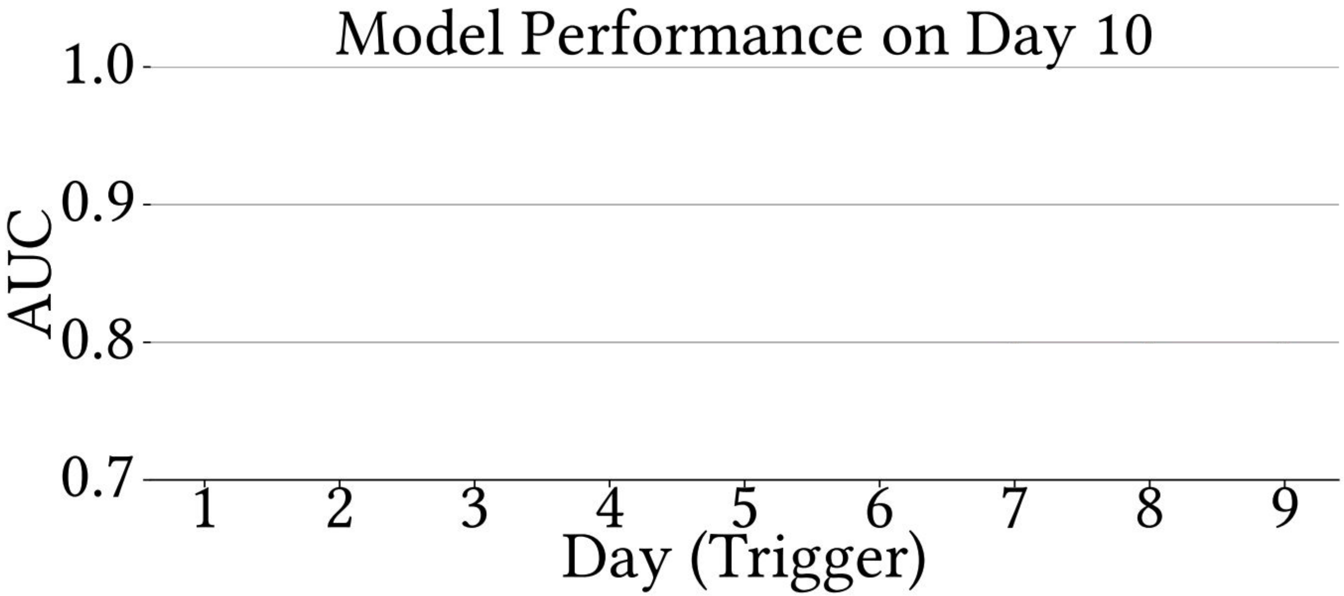
First Results: Criteo Dataset on DLRM

- DLRM Recommendation System Model
- Criteo 1TB Dataset
  - Anonymized categorical and numerical features for ad-click prediction for 24 days
- *Finetune* Setting:
  - Trigger training every day
  - Train on the data from that day

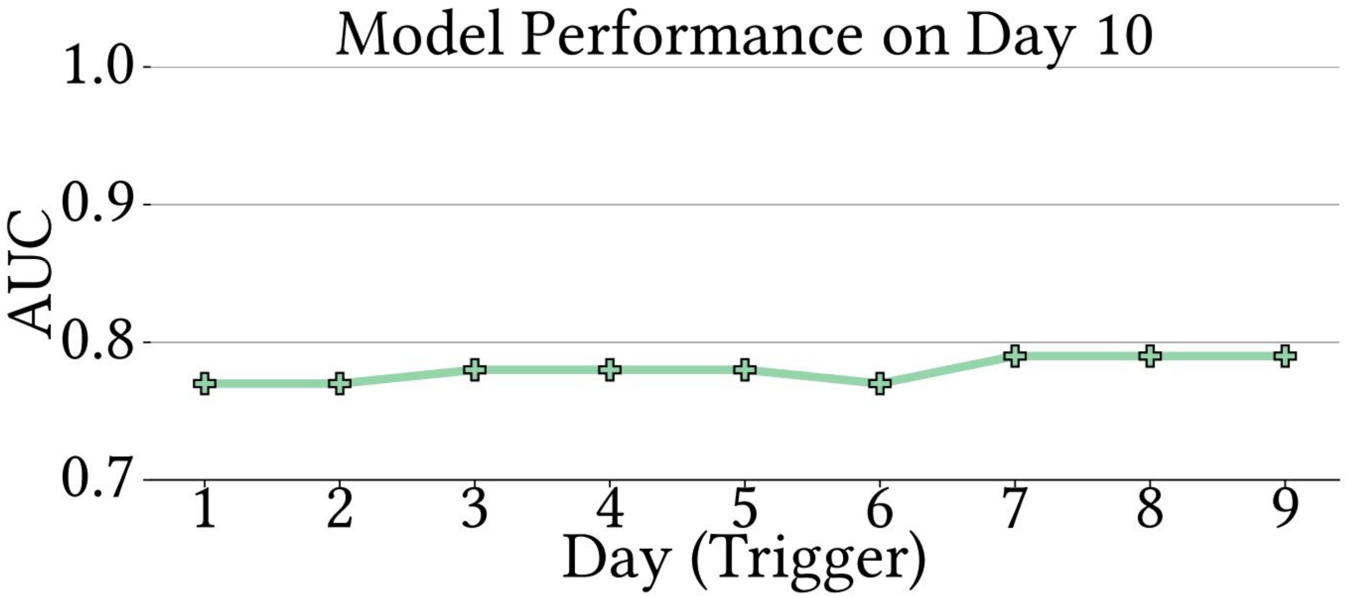
First Results: Criteo Dataset on DLRM



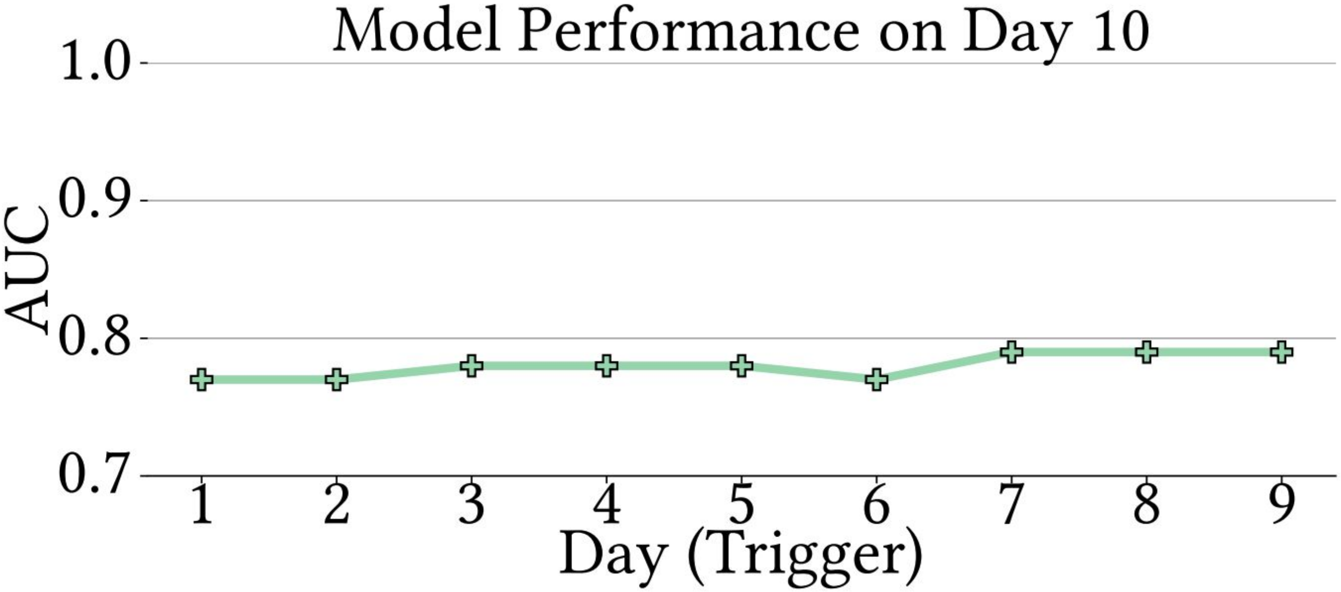
First Results: Criteo Dataset on DLRM



First Results: Criteo Dataset on DLRM

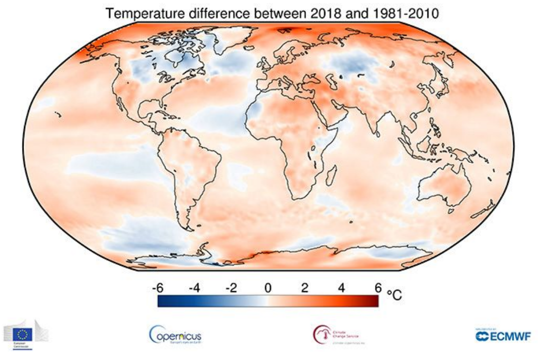


First Results: Criteo Dataset on DLRM



Alibaba: +0.2% AUC => +1% revenue

Towards A Benchmarking Suite



Text Classification

Autonomous Driving

Weather Forecasting

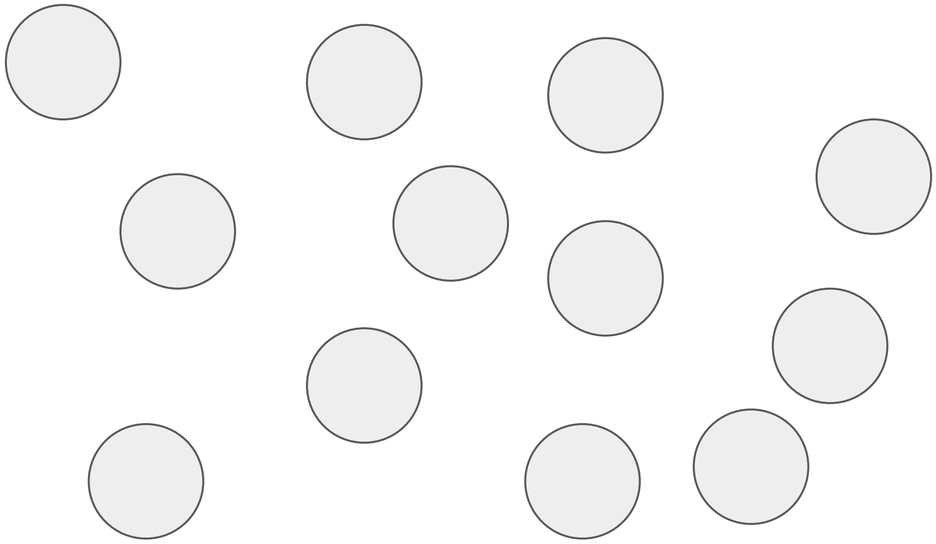
Conclusion

- In practical deployments, models need to adapt to **dynamic data**
- Due to increasingly large models and datasets, **frequent retraining/finetuning is not sustainable**
- *Modyn* is our vision for an open-source platform designed for exploring triggering policies (**when to train**), data selection policies (**what to train on**), and system optimizations

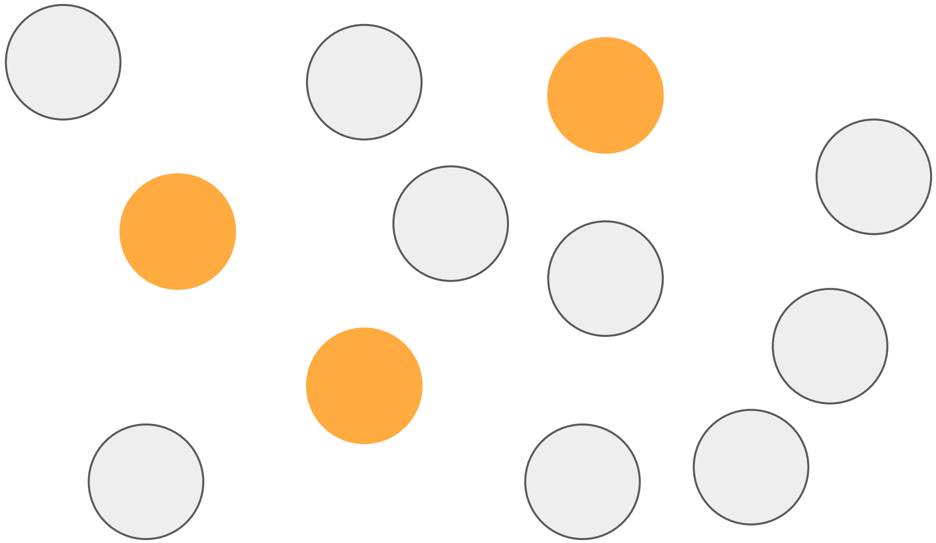
github.com/eth-easl/modyn

BACKUP SLIDES

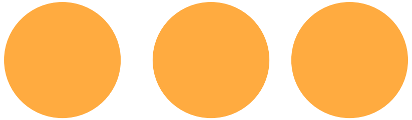
Static Data Selection: RHO-LOSS



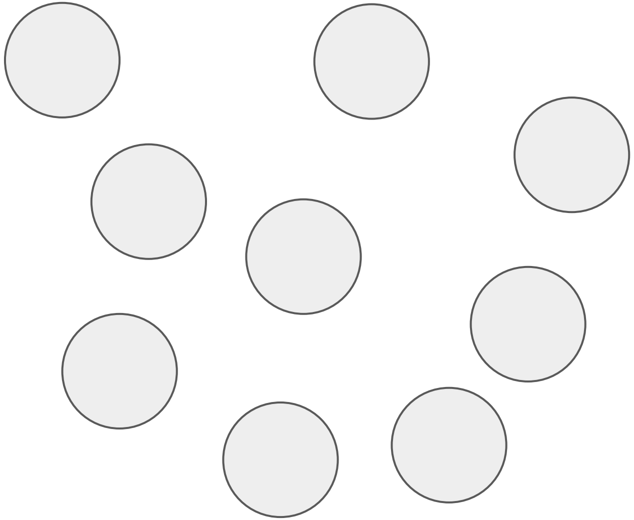
Static Data Selection: RHO-LOSS



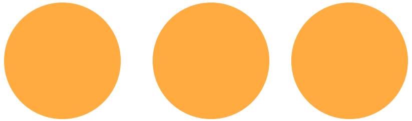
Static Data Selection: RHO-LOSS



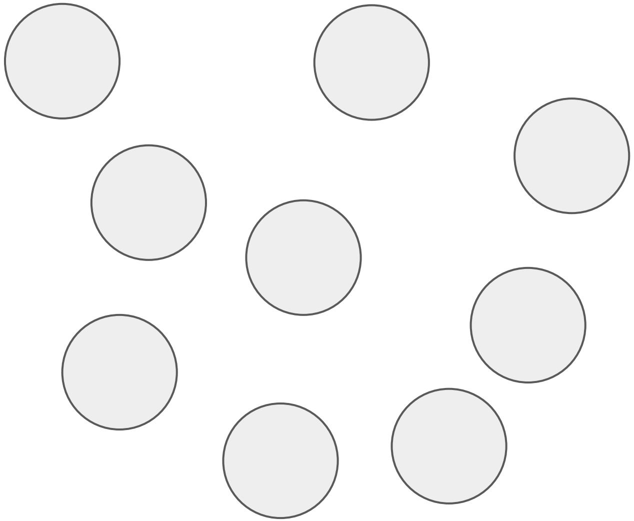
Holdout Set



Static Data Selection: RHO-LOSS



Holdout Set



Find data point that, *if trained on*, minimizes loss on *holdout set*

Static Data Selection: RHO-LOSS

$$\underset{(x,y)\in B_t}{\arg\min} - \log \mathrm{p}(\mathbf{y}^{\mathrm{ho}} \mid \mathbf{x}^{\mathrm{ho}}; \mathcal{D}_{\mathrm{t}} \cup (x, y))$$

Find the data point that minimizes loss of a "holdout set" (think: additional validation set) when trained on

Static Data Selection: RHO-LOSS

Approximation:

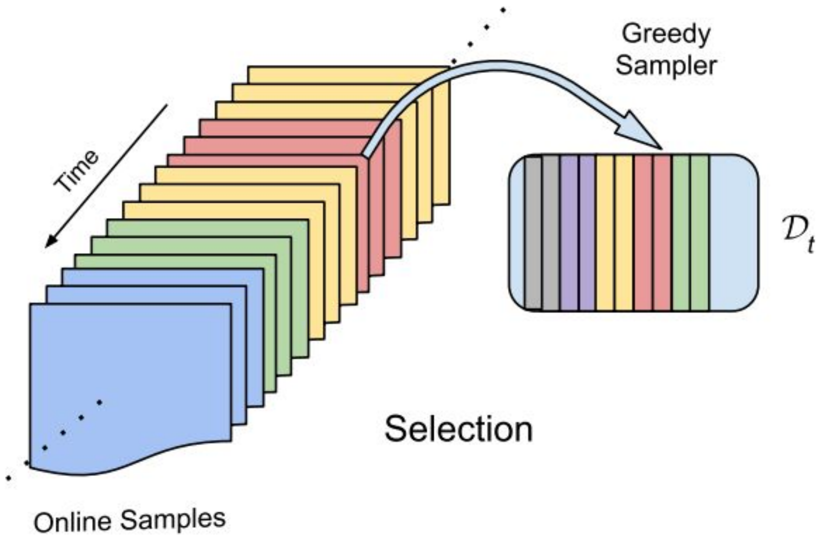
Train proxy model on holdout set, check loss of all points on proxy model

Static Data Selection: RHO-LOSS

$$\underset{(x,y)\in B_t}{\arg\max} \; \overbrace{\underbrace{L[y \mid x; \mathcal{D}_{\mathrm{t}}]}_{\text{training loss}} \; - \; \underbrace{L[y \mid x; \mathcal{D}_{\mathrm{ho}}]}_{\text{irreducible holdout loss (IL)}}}^{\text{reducible holdout loss}}$$

Avoid redundant, noisy, and less-relevant points by calculating IL on proxy model

Dynamic Data Selection: GDumb



Keep a fixed-size, class-balanced buffer

[GDumb: A Simple Approach that Questions Our Progress in Continual Learning]

ML Side: Dynamic Data Selection

How can we adapt static data selection policies to the dynamic setting?

ML Side: Dynamic Data Selection

How can we adapt static data selection policies to the dynamic setting?

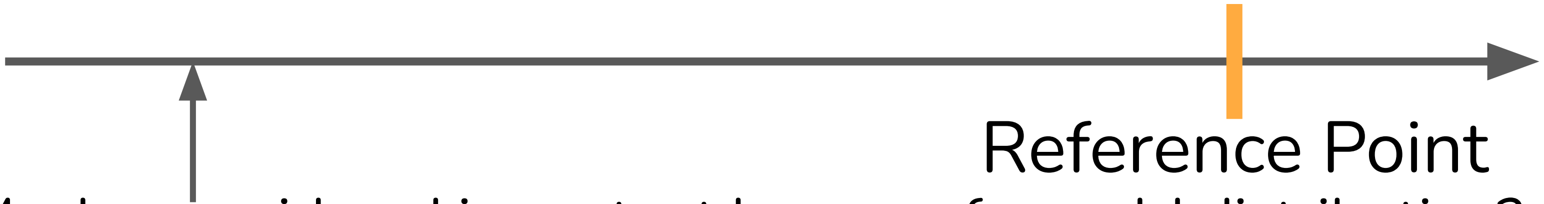
Noise or Distribution Shift?

Reference Point

ML Side: Dynamic Data Selection

How can we adapt static data selection policies to the dynamic setting?

But is this data actually important for learning the *current* distribution?



Maybe considered important because from old distribution?
Maybe considered important because previously deemed important?

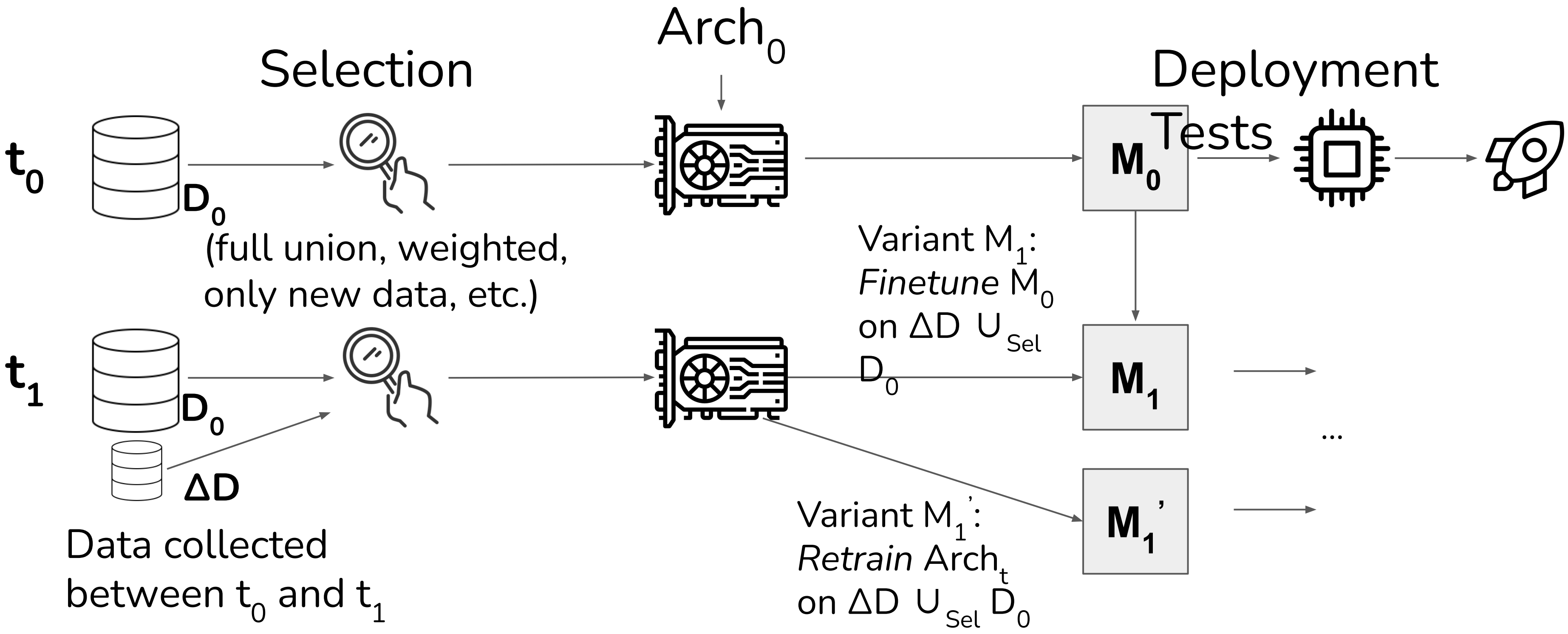
Systems Side: Efficient Implementation

How to efficiently implement selection policies?

How to manage metadata for billions of samples?

How can we efficiently (pre)fetch the training set for training?

Basic Model Updates



Training Policy Design Space

When to update the model?

1. Do we train with a fixed schedule, when a certain number of new data points has arrived or on data shifts?

2. How do we detect data distribution shifts?

How to update the model?

1. Do we retrain from scratch, finetune the existing model, or switch between both?
2. On which old and new data points do we train?
   a. Which metrics do we need for this decision?
   b. How do we collect and store them efficiently?
3. What do we do when old data is deleted?