Towards *"Data-efficient"* Machine Learning Systems

Noveen Sachdeva

@noveens97

UC San Diego

Welcome!

A Few Examples of Successful ML Systems

Generative Media

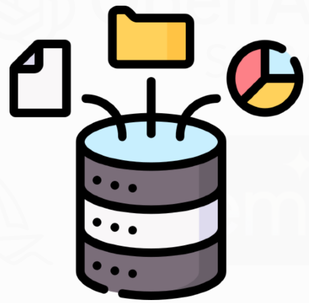
Recommender Systems

Self-Driving Cars

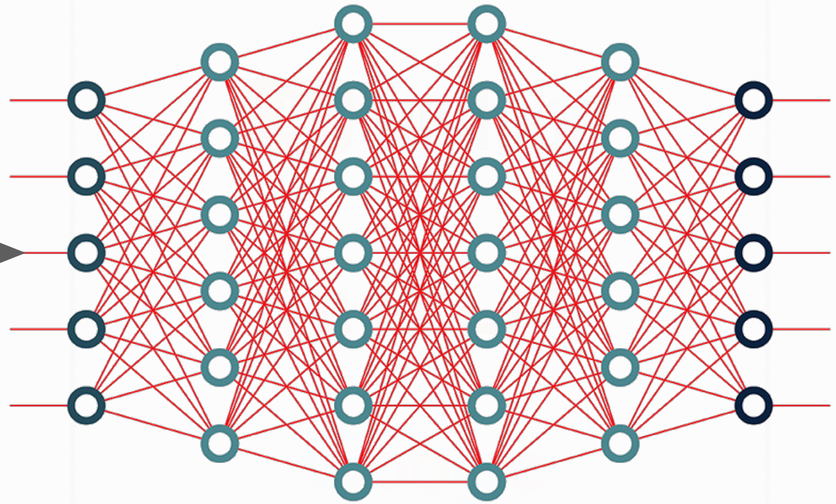
Typical ML Training Recipe

Excluding Many Secret Sauces

Training Data

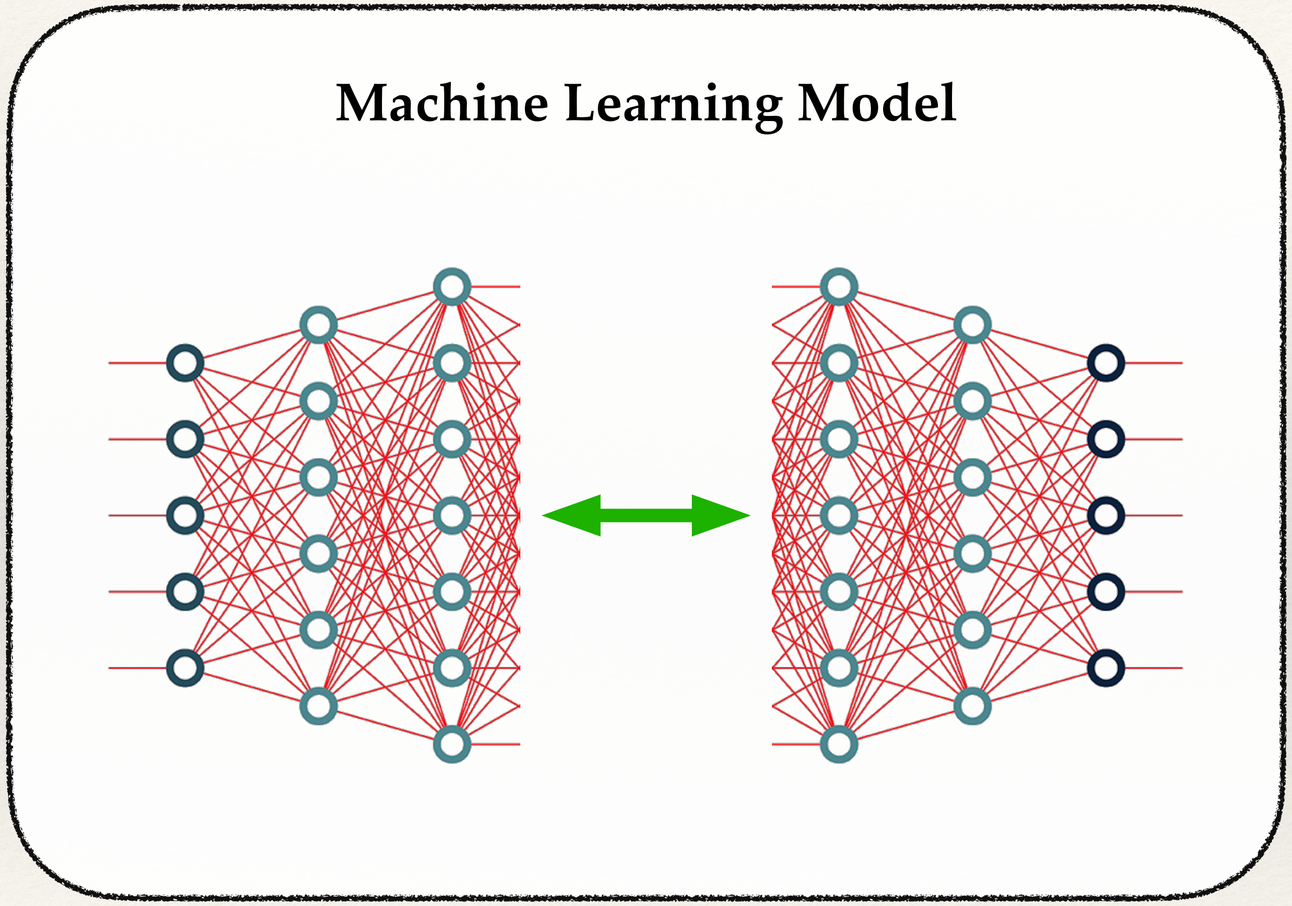


Machine Learning Model



Deploy! 💰

Typical Recipes for Success



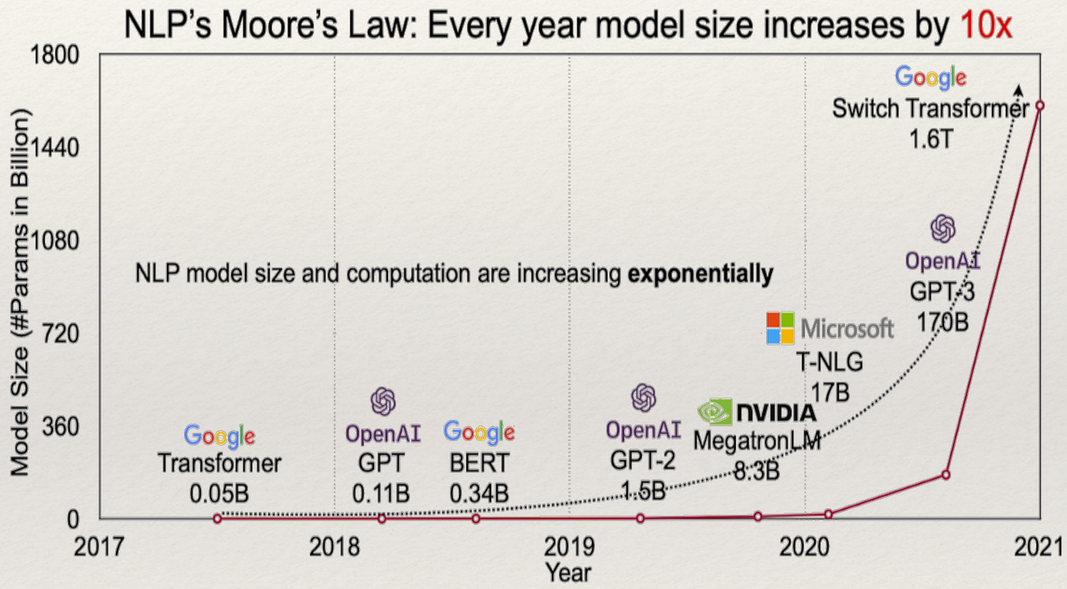
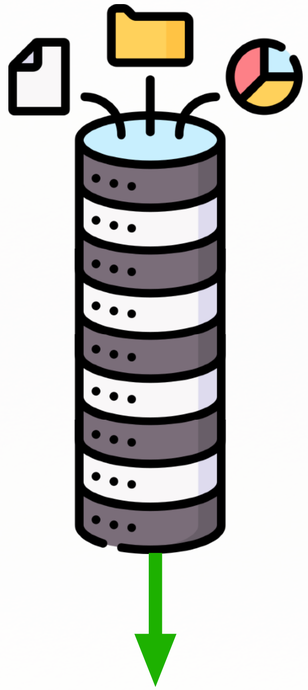


Image credit: https://medium.com/@harishdatalab/unveiling-the-power-of-large-language-models-llms-e235c4eba8a9

Typical Recipes for Success

Training Data



LIFEHACKER

LATEST TECH FOOD ENTERTAINMENT HEALTH MONEY HOME & GARDEN

Home → Tech → AI

AI Companies Are Running Out of Internet

AI is hungry, and there's not enough data to sate its appetite.

PYMNTS

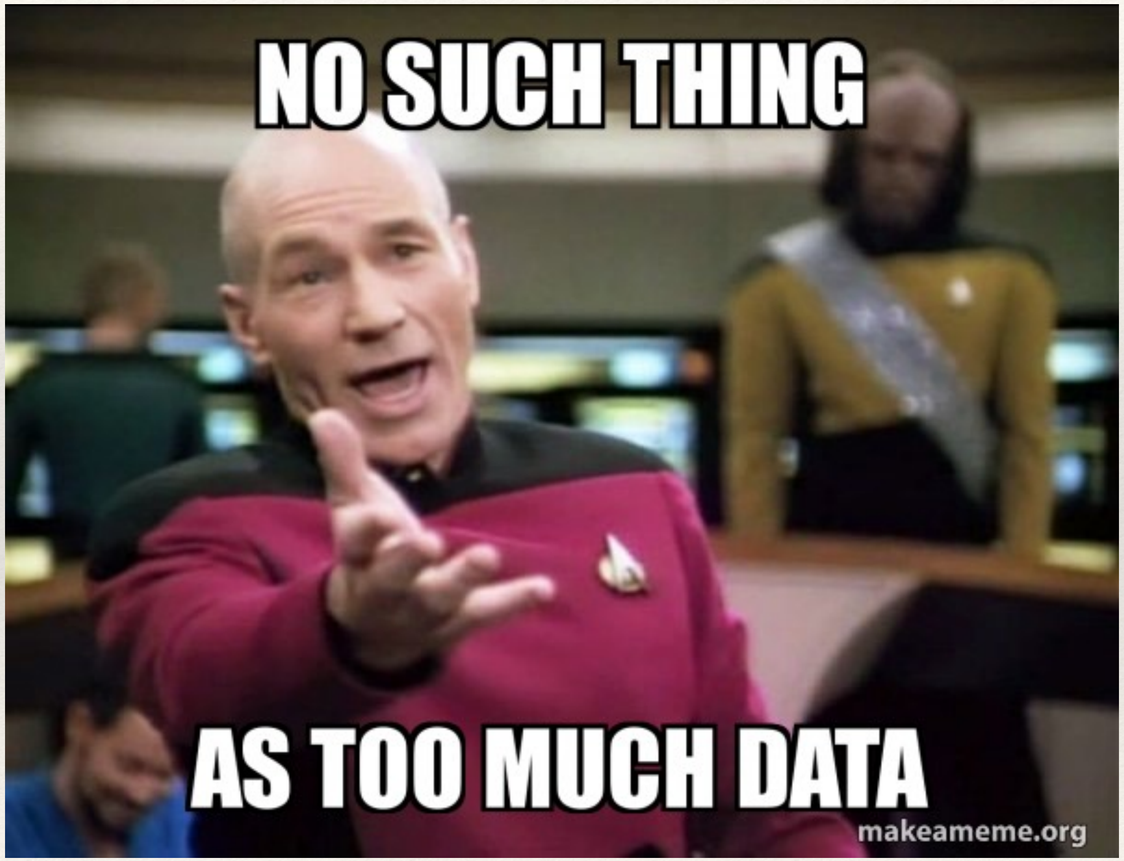
PYMNTS TV | Today | B2B | Retail | Fintech | Digital Transformation | Crypto | EMEA | Tracker® Repo

AI Faces a Data Drought ... for Real

BY **PYMNTS** | APRIL 10, 2024

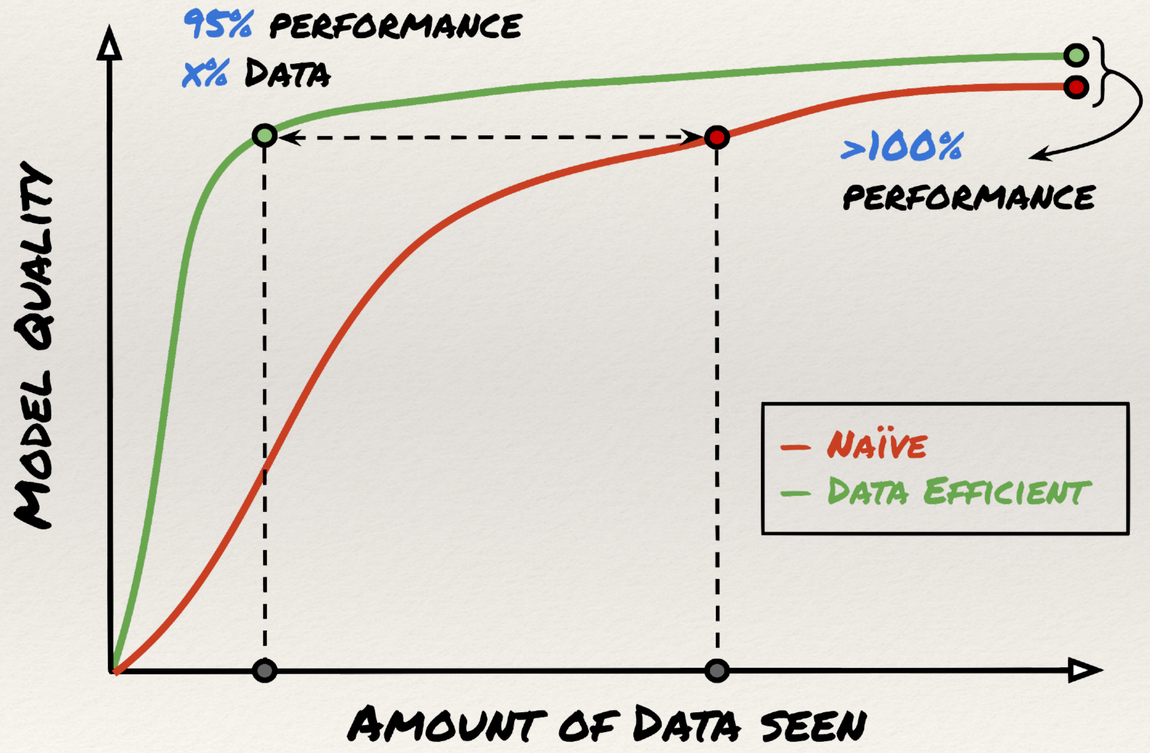
Question: Is **more data** really needed for training **better models**?

Routinely over-heard at big-tech:



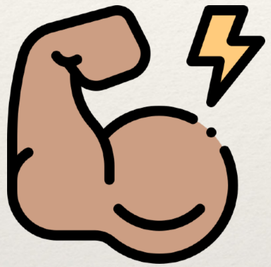
This Dissertation

Data Efficiency



This Dissertation

Why Data Efficiency?



More accurate models

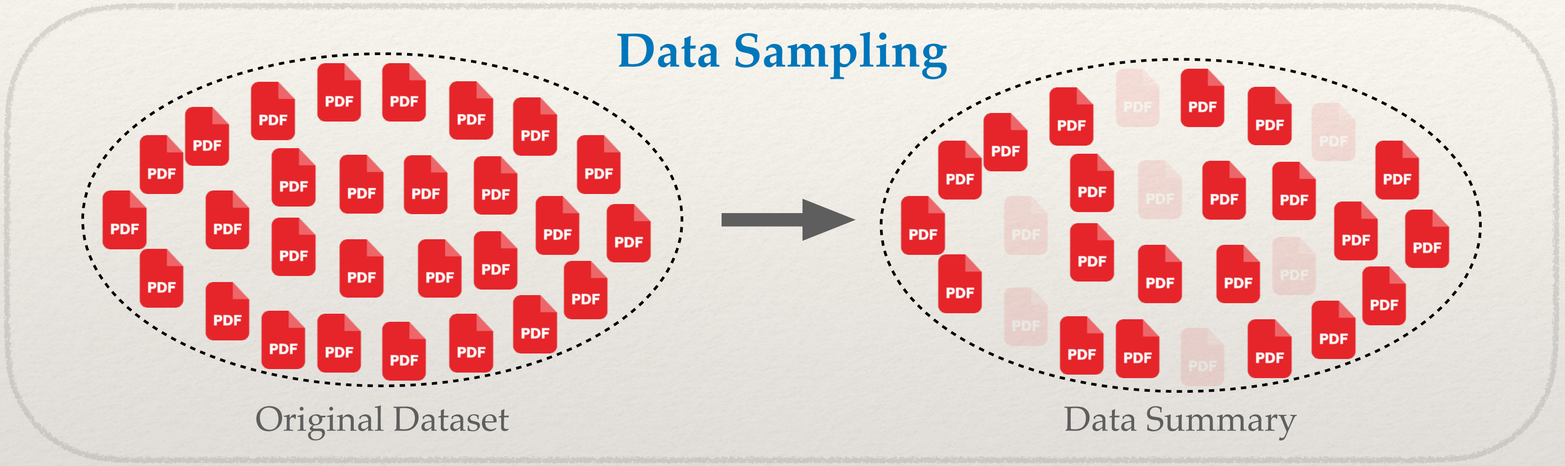
Save money to train

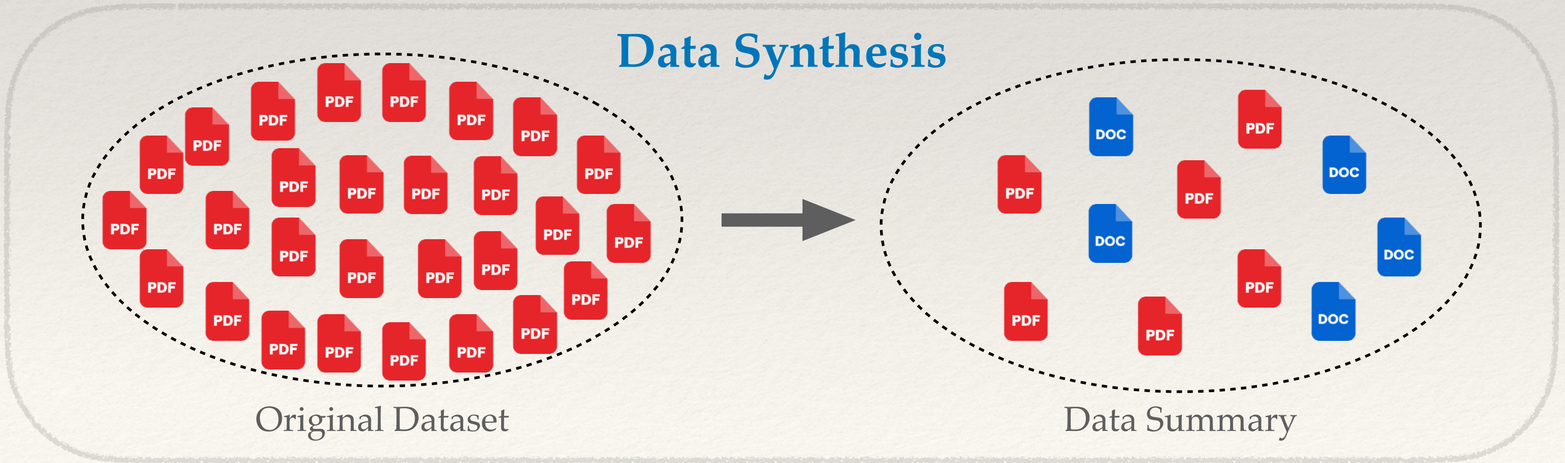
Save time to train

Less CO_2 emissions due to training

This Dissertation

How to be Data-Efficient?





This Dissertation

Outline

Chapter - I: Data Sampling

Part-I: Recommender Systems → Part-II: LLM Pre-Training

Original Dataset → Data Summary

Chapter - II: Data Synthesis

Part-I: Recommender Systems → Part-II: Auto-Regressive Data

Original Dataset → Data Summary

On Sampling Collaborative Filtering Datasets

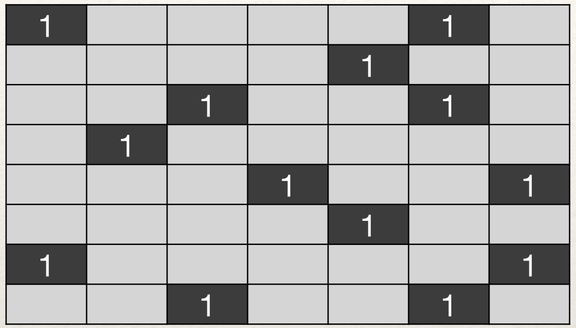
Noveen Sachdeva [1] Carole-Jean Wu [2] Julian McAuley [1]

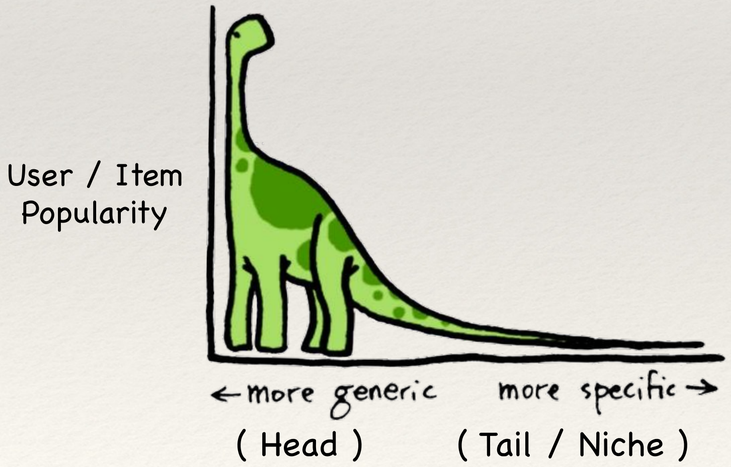
University of California, San Diego [1] Meta AI [2]



Scope

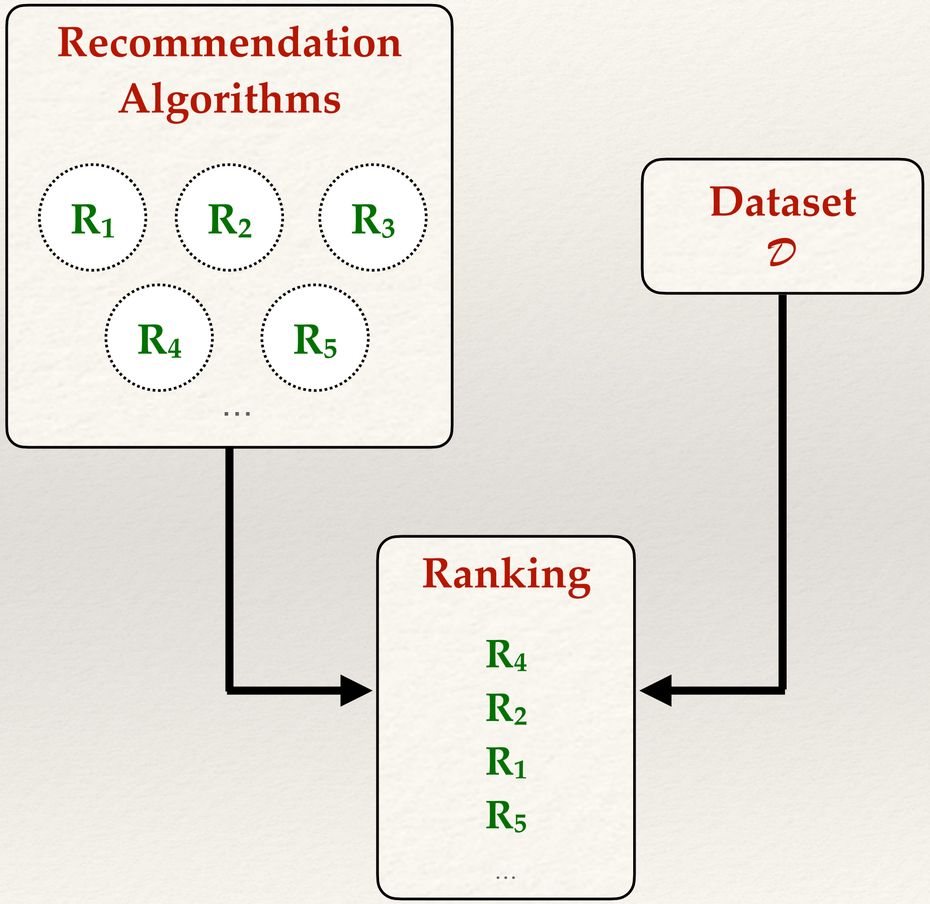
Recommender Systems





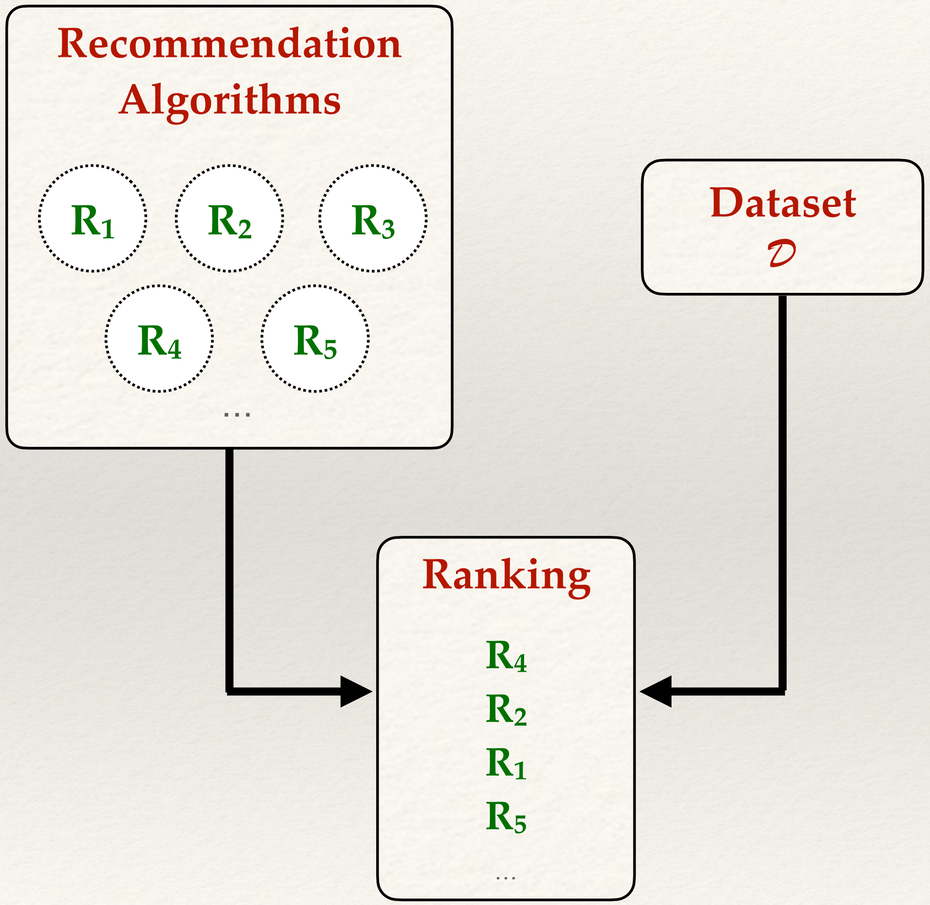
Objective

Infer the Ranking of N-different Recommendation Models



Objective

Naive vs. Data-Efficient



Naive: EXPENSIVE 💰

1. Train all candidate algorithms on the entire dataset
2. Evaluate all algorithms
3. Measure the ranking of all algorithms

Data-Efficient: EFFICIENT 🥳

1. Train all candidate algorithms on a smaller sample of the dataset
2. Evaluate all algorithms
3. Measure the ranking of all algorithms

SVP-CF

Down-sampling Recommendation Data

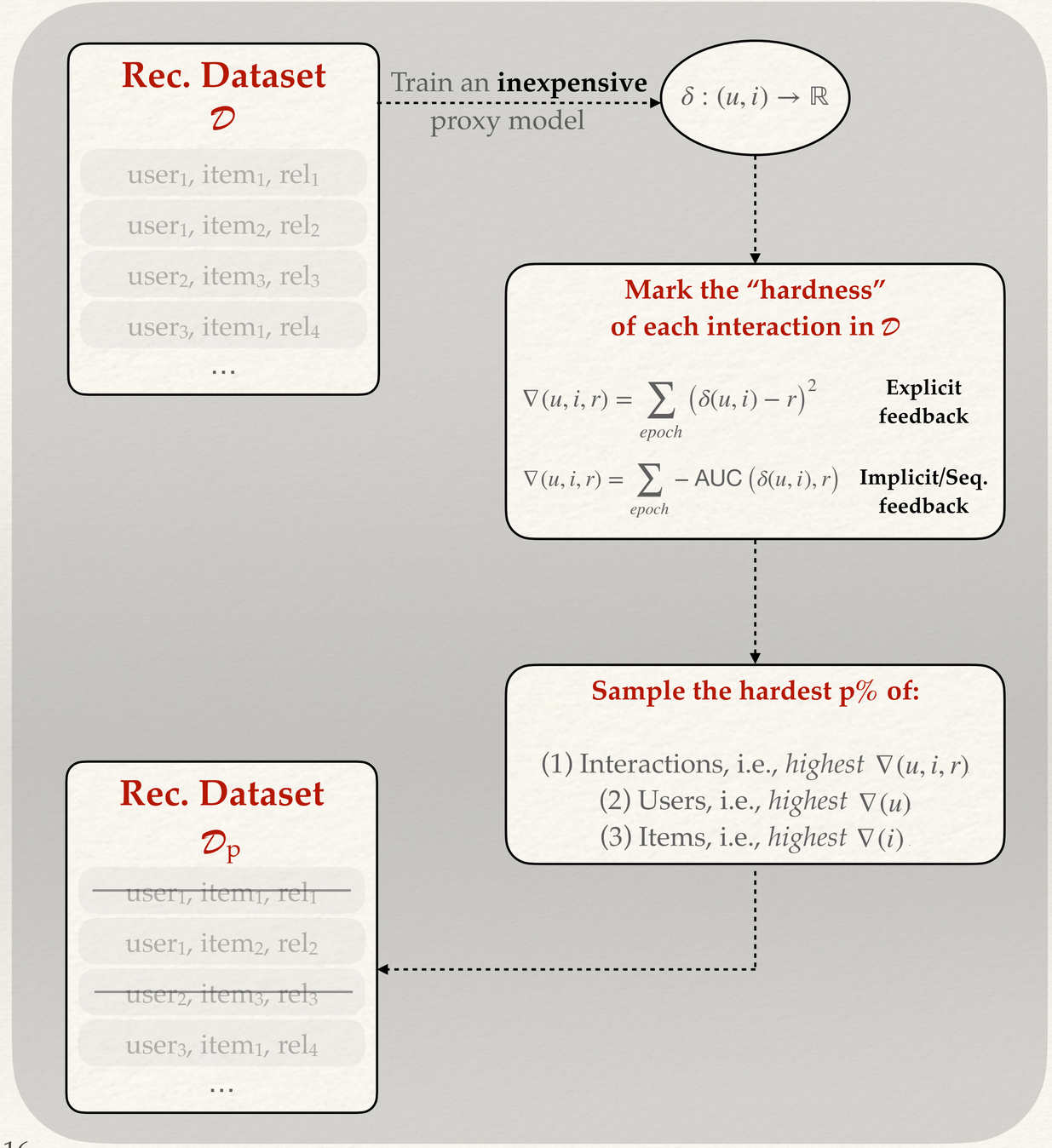
Premise: **Easy** parts of a dataset are most likely **easy** for all recommendation algorithms.
Hence, removing such easy segments of data is unlikely to affect the relative ordering of algorithms.

SVP- CF

Down-sampling Recommendation Data

Robust framework:

- Uses a proxy model to **tag the overall hardness** of each user-item interaction
- Can efficiently **handle various recommendation scenarios**, *e.g.*, explicit, implicit, sequential, etc.
- Can **sample across a variety of data axes**: interactions, users, items, or even combinations of them

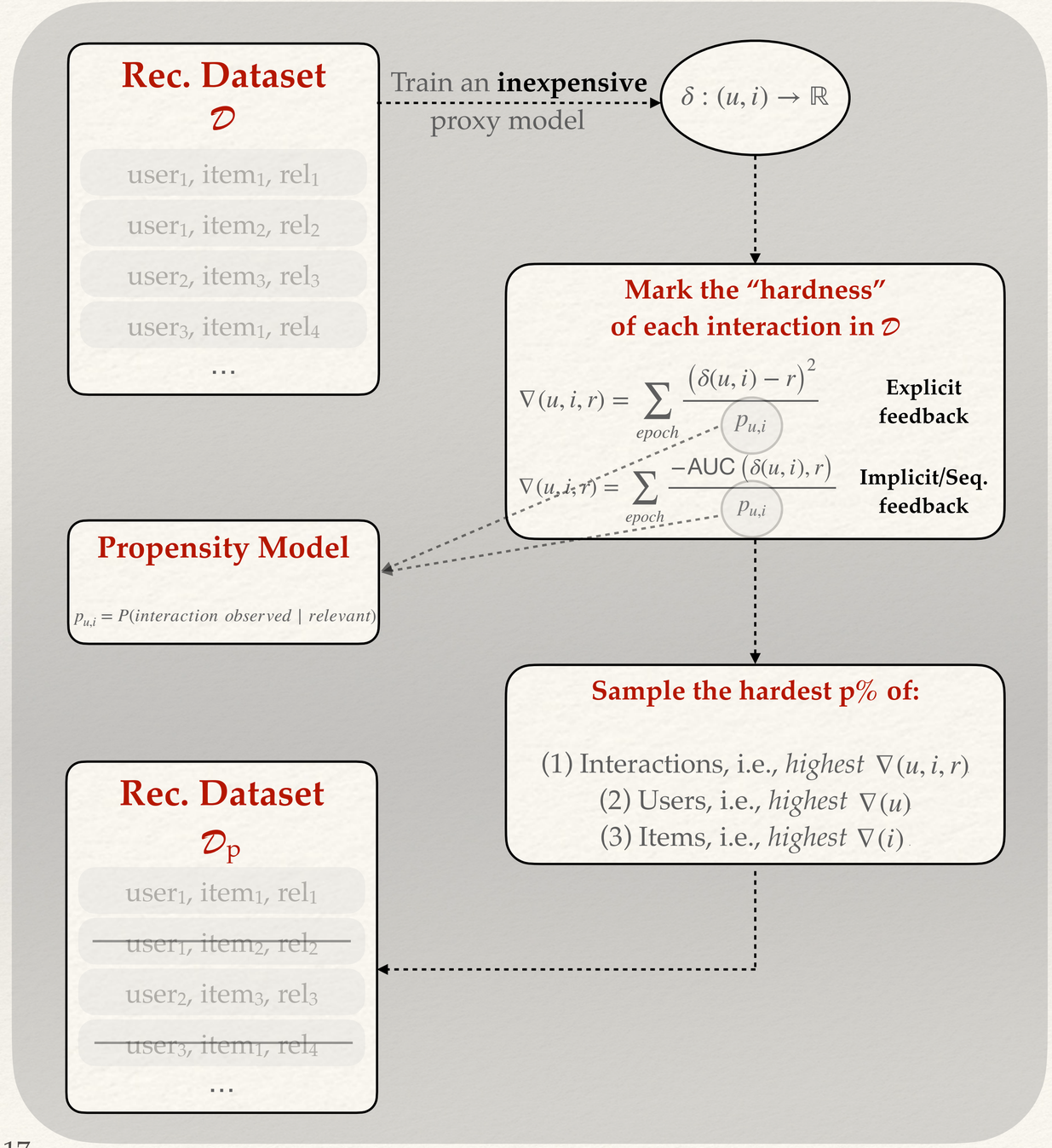


SVP- CF- Prop

Propensity Correction

Due to the large catalog of items, **account for potentially missing data**, especially for long-tail items

- **Re-weigh the hardness scores** using the probability of a user-item interaction going missing (propensity)
- Implicitly handles the long-tail and data sparsity issues in user-item interaction data



DATA-GENIE

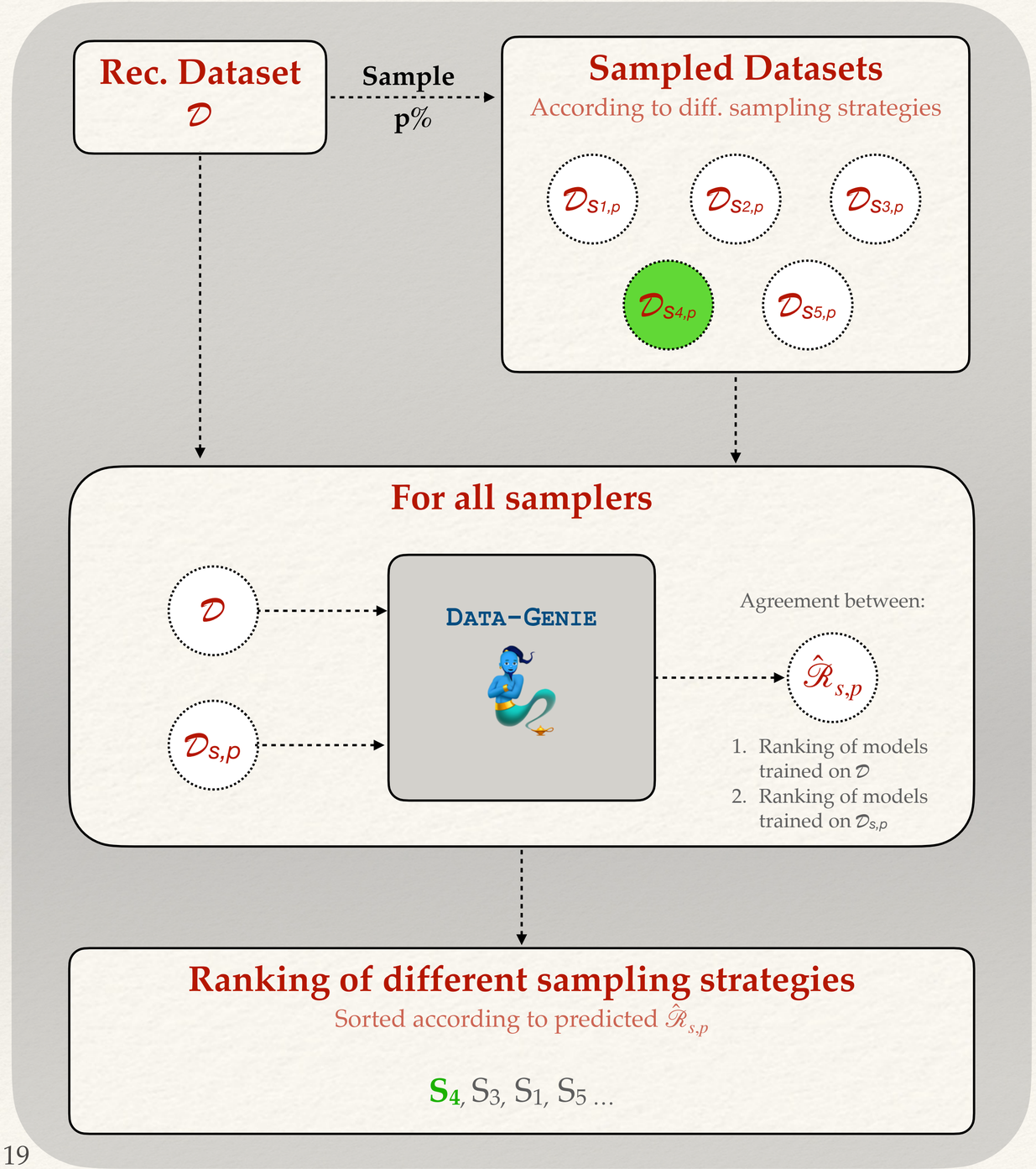
Which sampler is best for my dataset?

Premise: Can we build an oracle-model which given (1) a dataset, (2) list of sampling strategies, and (3) a sampling budget, can **automatically predict** which sampling scheme would be the best?

DATA-GENIE

Which sampler is best for my dataset?

- Dynamically **predict the performance** of a sampling strategy for any given dataset
- Circumvents the time-consuming process of training and benchmarking various recommendation algorithms
- A trained DATA-GENIE model can transfer to **any dataset**, and can predict the utility of **any sampling strategy**



DATA-GENIE

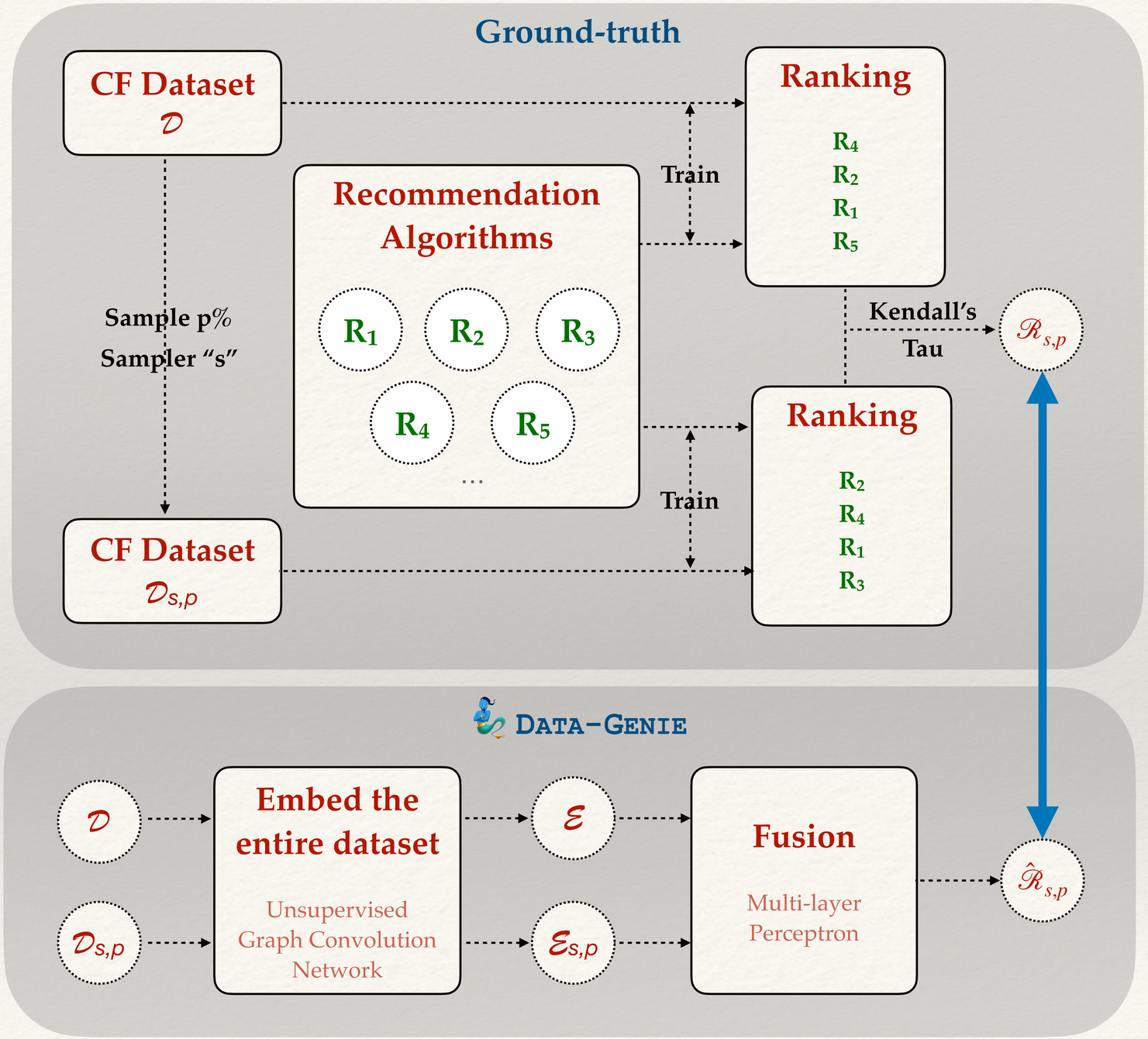
Training Objective

- DATA-GENIE-regression:

$$\arg\min \sum_{\mathcal{D},\, s,\, p} \left(\mathcal{R}_{s,p} - \hat{\mathcal{R}}_{s,p} \right)^2$$

- DATA-GENIE-ranking:

$$\arg\min \sum_{\mathcal{D},\, p} \sum_{\mathcal{R}_{s_i,p} > \mathcal{R}_{s_j,p}} - ln\ \sigma \left(\hat{\mathcal{R}}_{s_i,p} - \hat{\mathcal{R}}_{s_j,p} \right)$$



Experiments

Setup

	Sampling strategy
Interaction sampling	Random
	Stratified
	Temporal
	SVP-CF *w/* MF
	SVP-CF *w/* Bias-only
	SVP-CF-Prop *w/* MF
	SVP-CF-Prop *w/* Bias-only
User sampling	Random
	Head
	SVP-CF *w/* MF
	SVP-CF *w/* Bias-only
	SVP-CF-Prop *w/* MF
	SVP-CF-Prop *w/* Bias-only
Graph	Centrality
	Random-walk
	Forest-fire

Table: Sampling strategies used in our experiments

- 16 different sampling strategies
- 6 collaborative filtering datasets
- Explicit / Implicit / Sequential feedback for each CF-dataset
- 7 recommendation algorithms in our benchmarking suite
- A total of **400*k*** recommendation models trained (~9 months of single-GPU compute time!)

Experiments

Major Results

	Sampling strategy	*Average* Kendall's Tau
Interaction sampling	Random	0.407
	Stratified	0.343
	Temporal	0.405
	SVP-CF *w/* MF	<u>0.484</u>
	SVP-CF *w/* Bias-only	0.468
	SVP-CF-Prop *w/* MF	0.43
	SVP-CF-Prop *w/* Bias-only	0.458
User sampling	Random	0.431
	Head	0.19
	SVP-CF *w/* MF	0.344
	SVP-CF *w/* Bias-only	0.343
	SVP-CF-Prop *w/* MF	0.429
	SVP-CF-Prop *w/* Bias-only	0.445
Graph	Centrality	0.266
	Random-walk	0.396
	Forest-fire	0.382

Table: Average Kendall's Tau of various sampling strategies

- Widely used practice of making dense data subsets (e.g., Head-user, centrality) seem to be the worst ideas of all sampling strategies.
- SVP-CF significantly outperforms other samplers in retaining the ranking of different recommendation algorithms.

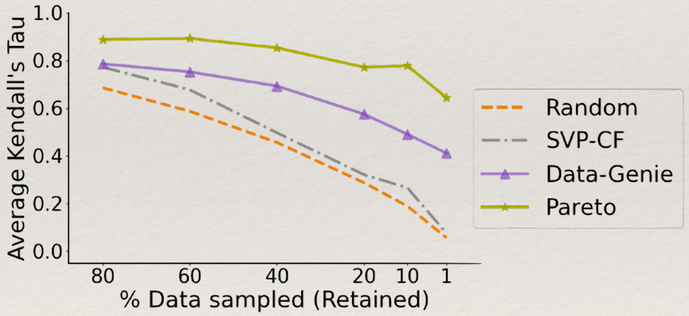


Figure: Does DATA-GENIE improve sampling performance with extreme sampling?

- Using SVP-CF, we can efficiently gauge the ranking of different algorithms with adequate confidence on **40-50%** data sub-samples, leading in an **~2x** time speedup.
- DATA-GENIE enjoys the same level of performance with only **10%** of the original data, equating to **~5.8x** time speedup!

How to Train Data-Efficient LLMs

Noveen Sachdeva [1] Benjamin Coleman [2] Wang-Cheng Kang [2] Jianmo Ni [2]
Lichan Hong [2] Ed H. Chi [2] James Caverlee [2] Julian McAuley [1] Derek Z. Cheng [2]

University of California, San Diego [1] Google DeepMind [2]



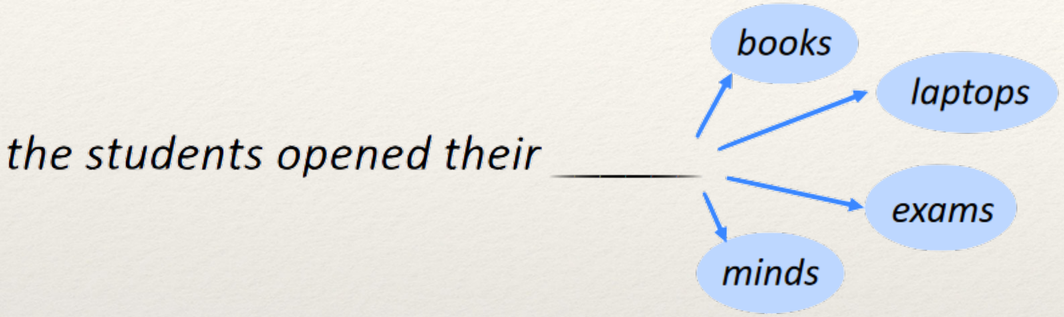






Scope

Language Modeling



Pre-Training

- Very large models
- Very large datasets collected from all of the internet
- Very expensive training procedure
- Evaluation over hundreds of different tasks

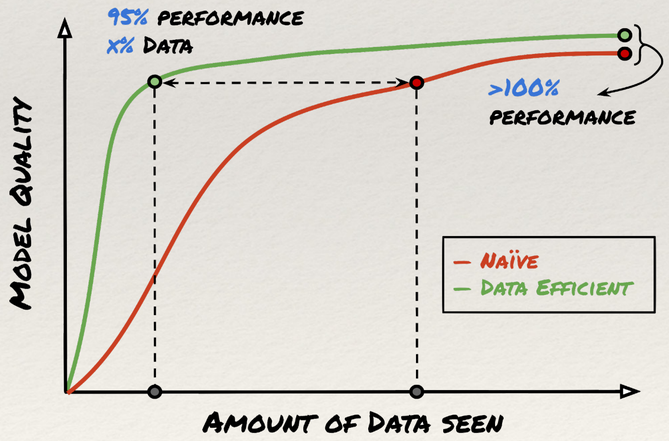
Objective

Perform Accurate Language Modeling

That is, learn better next-token predictors:

- $\delta : [\text{token}_1, \text{token}_2, \ldots, \text{token}_n] \mapsto \mathcal{T}; \ \forall \ \text{token}_i \in \mathcal{T}$

Naive vs. Data-Efficient



Naive:

Train the model on the entire dataset

Data-Efficient:

Train the model on the sampled version of the dataset

Ask-LLM

Sampling High-Quality LLM Pre-Training Data

Premise: Can we prompt an existing LLM to estimate the quality of a pre-training document?

Ask-LLM

Sampling High-Quality LLM Pre-Training Data

Robust framework:

- Leverages the **reasoning capabilities** of modern LLMs rather than common heuristics like perplexity
- We prompt Flan-T5 and Gemma-7B for data quality
- **Explicit control** over what kind of data we prefer

Why P("yes" | prompt) is a good idea:

- **Real-valued** "confidence" score needed to sort millions of documents
- One-shot decoding and **no majority voting needed**

Ask-LLM prompt

###
This is a pretraining .... datapoint.
###

Does the previous paragraph demarcated within ### and ### contain informative signal for pre-training a large-language model? An informative datapoint should be well-formatted, contain some usable knowledge of the world, and strictly NOT have any harmful, racist, sexist, etc. content.

OPTIONS:
- yes
- no

Sampling score = P("yes" | prompt)

Density

Sampling Diverse LLM Pre-Training Data

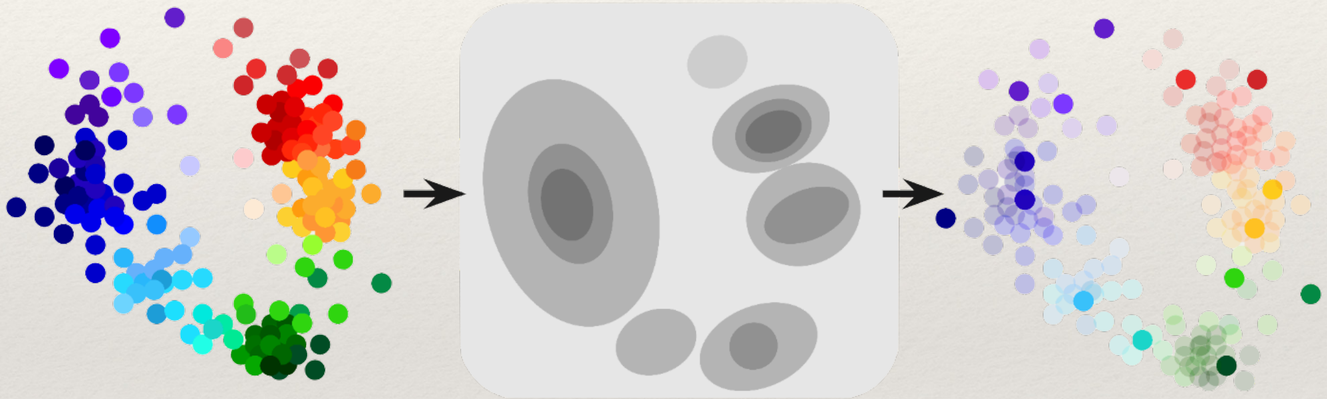
Premise: Can we sample datapoints from diverse topics in the original dataset?

Density

Sampling Diverse LLM Pre-Training Data

Robust framework:

- Estimate data density using hashed sentence-T5 embeddings
- Up-weights the tail components and down-weights the head components
- No need for expensive techniques like clustering, graph-cuts, etc. to localize a notion of coverage



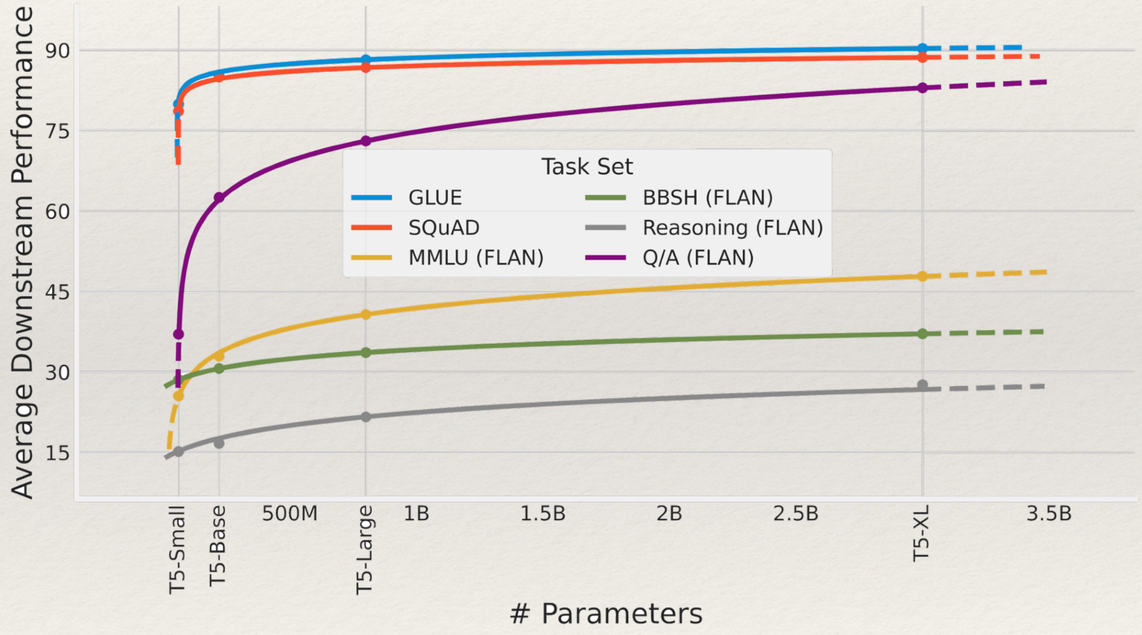
Ask-LLM & Density

Metric: Effective Model Size

- With 100s of metrics, hard to devise a single notion of "quality." Some metrics are hard-to-move whereas some are easy.

- We devise an "Effective Model Size" metric that is a **scaling-law averaged normalized metric** over all downstream tasks:

" If our ablations (data sampling) lead to *x* performance, what sized LLM should I have trained in the original setting (the full dataset) to achieve the same *x* performance? "



Ask-LLM & Density

Experiments

Setup

- We train T5-Large (800M parameters) for 524B tokens on the C4 dataset

Conclusions

- Up to 44% speedup while training T5-Large
- Training on data sampled by Ask-LLM (Gemma) is equivalent to training a 2x sized model on the entire dataset
- Density sampling recovers full-data performance (flat-line) but Ask-LLM consistently exceeds it

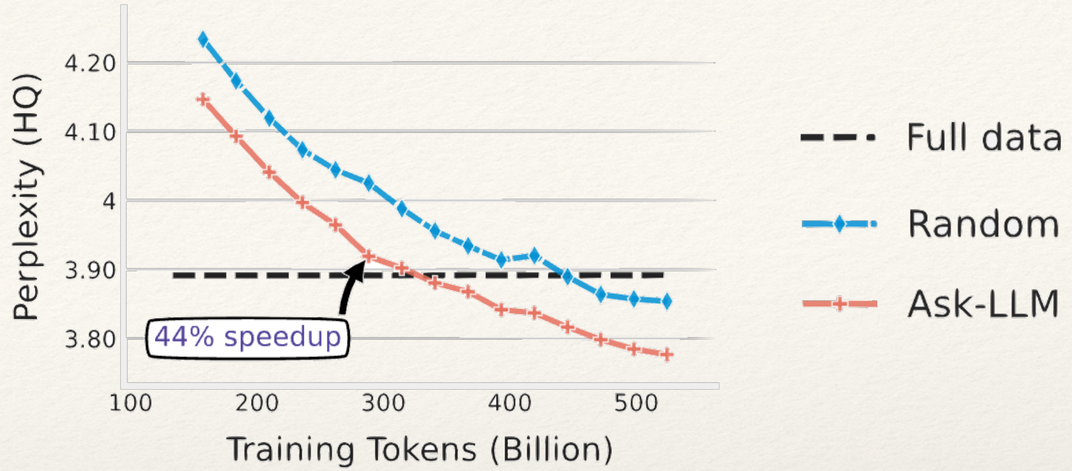


Figure A: Does training on Ask-LLM sampled data converge faster?

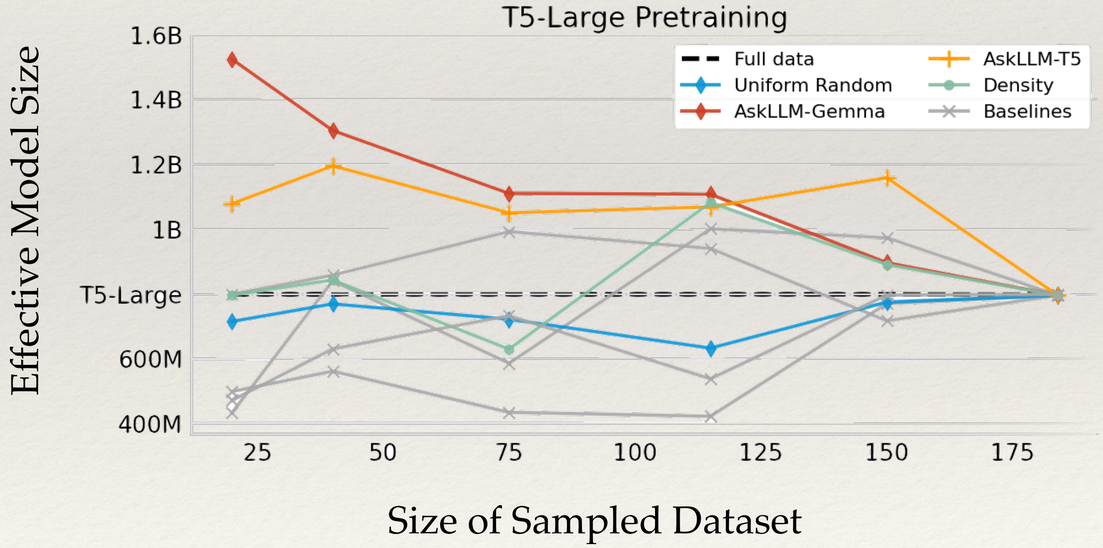


Figure B: Size of sampled data vs. final model quality

This Dissertation

Outline

Chapter - I: Data Sampling

Part-I: Recommender Systems → Part-II: LLM Pre-Training

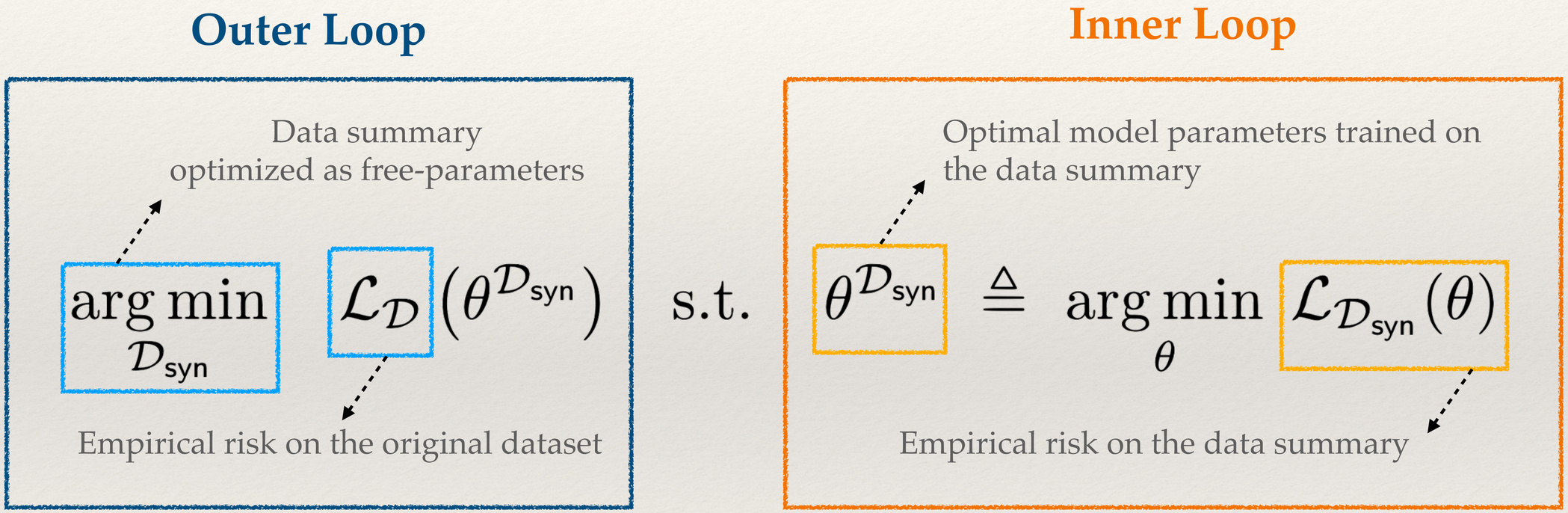
Original Dataset → Data Summary

Chapter - II: Data Synthesis

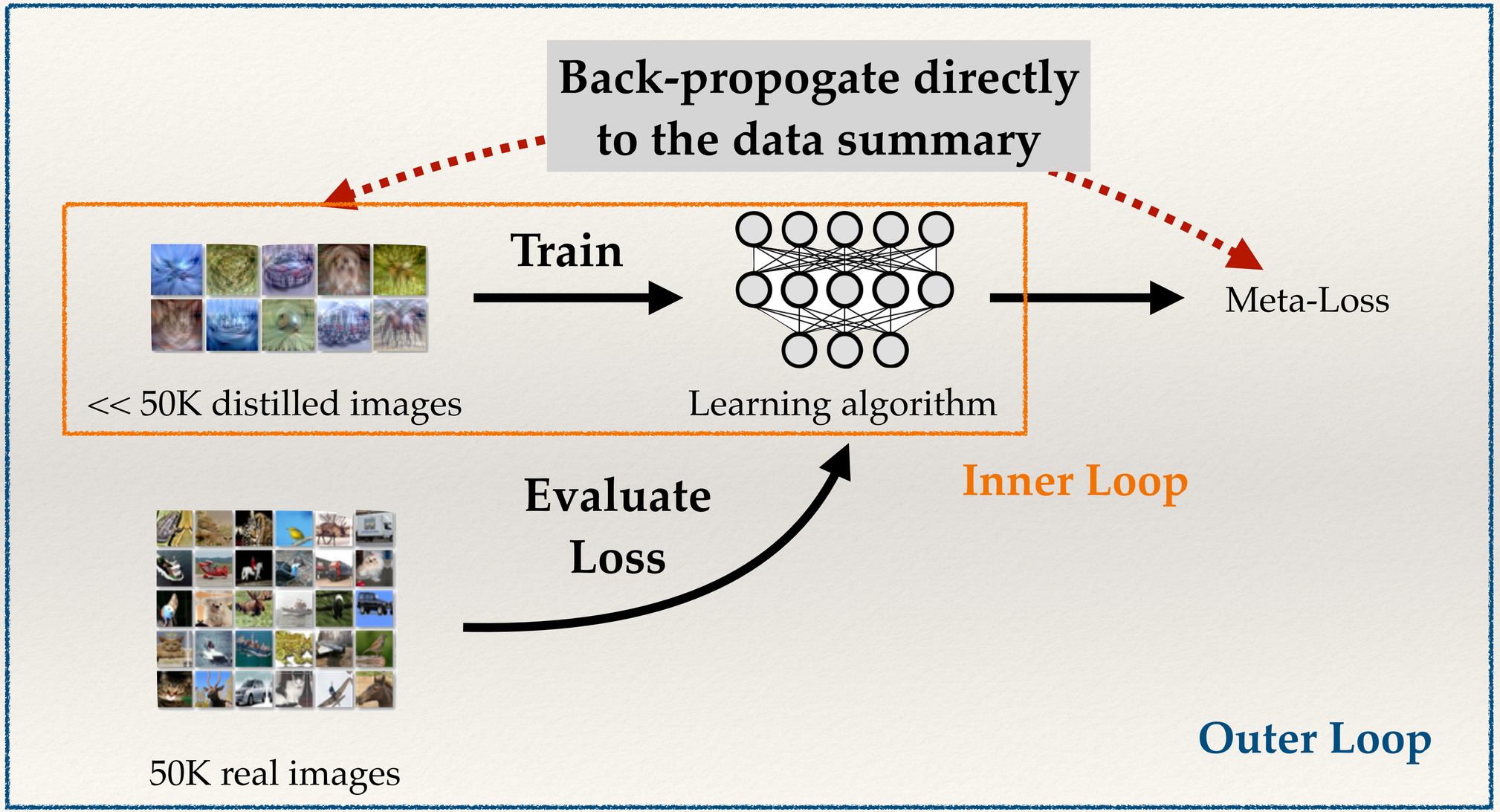
Part-I: Recommender Systems → Part-II: Auto-Regressive Data

Original Dataset → Data Summary

Data Distillation: Automated Data Optimization



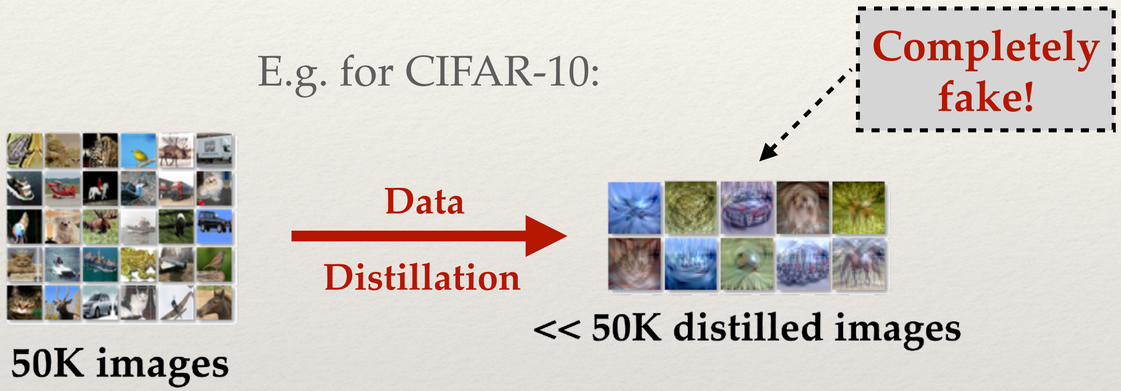
Data Distillation: Automated Data Optimization



Data Distillation: Automated Data Optimization

TL;DR Directly optimize the data summary (stored as free parameters) via meta-learning

E.g. for CIFAR-10:



No Heuristics	Data summary is optimized for training models
Data is optimized for a specific model	Computationally Expensive

Most notably, this framework also requires:

- The distilled data to be "optimizable", e.g., pixel values in an image
- Performing **data distillation for discrete data settings** like user-item interactions, text, graphs, etc. becomes highly non-trivial

NEURAL INFORMATION PROCESSING SYSTEMS

Infinite Recommendation Networks: A Data-Centric Approach

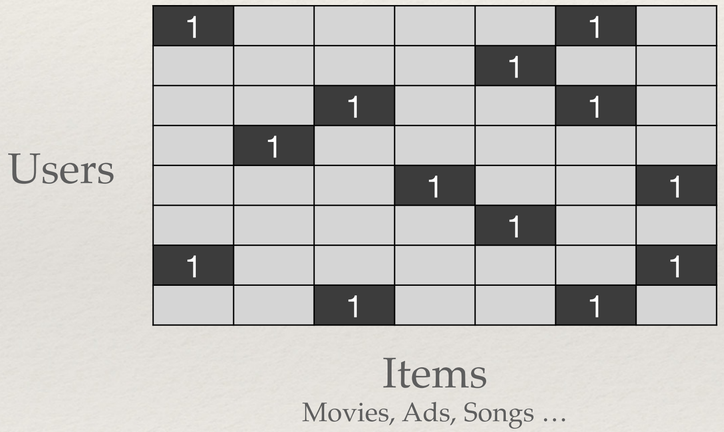
Noveen Sachdeva [1] Mehak Dhaliwal [1] Carole-Jean Wu [2] Julian McAuley [1]

University of California, San Diego [1] Meta AI [2]




Scope

Implicit-feedback Recommender Systems



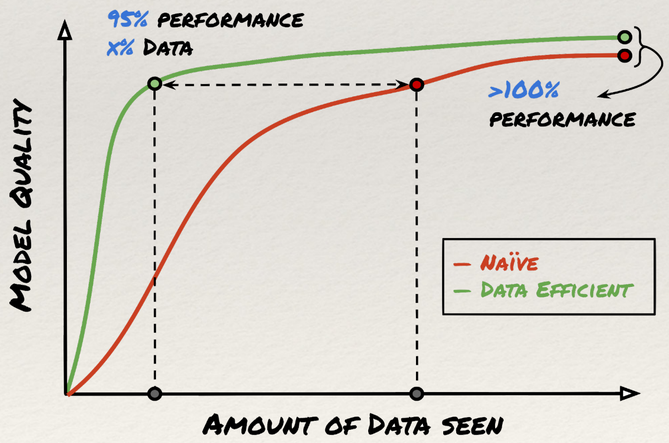
Objective

Perform Accurate Recommendation

That is, learn better relevance predictors:

- $\delta : (\text{user}, \text{item}) \mapsto \mathbb{R};\ \forall\ \text{user} \in \mathcal{U}, \text{item} \in \mathcal{I}$

Naive vs. Data-Efficient



Naive:

Train the recommendation model on the entire dataset

Data-Efficient:

Train the recommendation model on the distilled version of the dataset

∞-AE

A Better Model for Recommendation

Premise: Does stretching the bottleneck layer of an autoencoder till ∞ help in better recommendation?

∞-AE

Primer: Neural Tangent Kernel

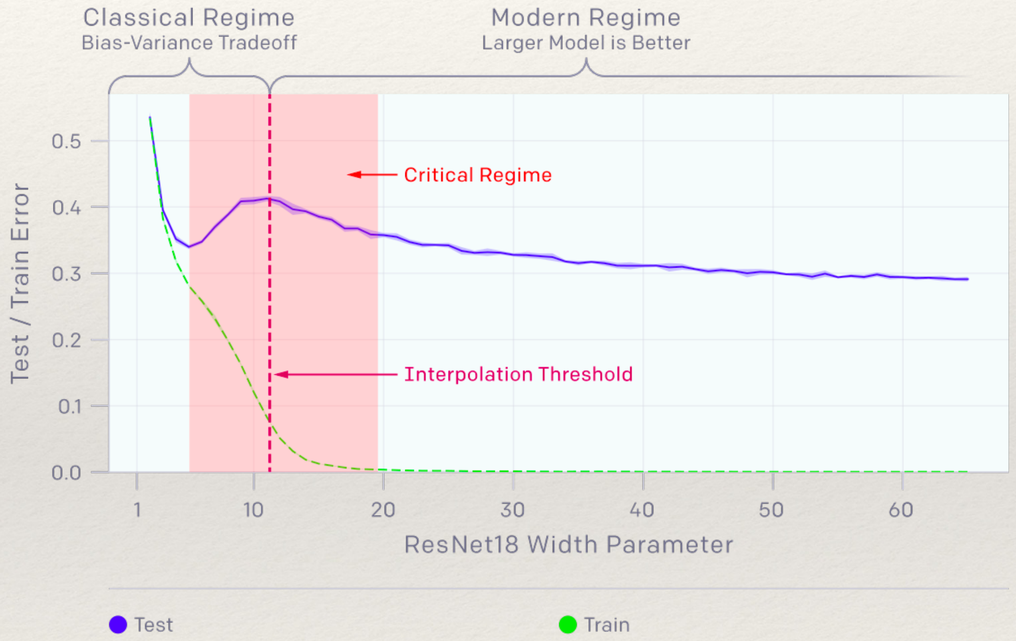
- **Infinite-width Correspondence:** Performing Kernelized Ridge Regression with the Neural Tangent Kernel (NTK) emulates the training of an infinite-width NN for an infinite number of SGD steps.

- For a given neural network architecture $f_\theta : \mathbb{R}^d \mapsto \mathbb{R}$, its corresponding NTK $\mathbb{K} : \mathbb{R}^d \times \mathbb{R}^d \mapsto \mathbb{R}$ is given by:

$$\mathbb{K}(x, x') = \mathop{\mathbb{E}}_{\theta \sim W} \left[\left\langle \frac{\partial f_\theta(x)}{\partial \theta}, \frac{\partial f_\theta(x')}{\partial \theta} \right\rangle \right]$$

- Learning follows a double-descent phenomenon

- Finite-width counterparts empirically outperform NTK for standard image classification tasks



Credit: https://openai.com/blog/deep-double-descent/

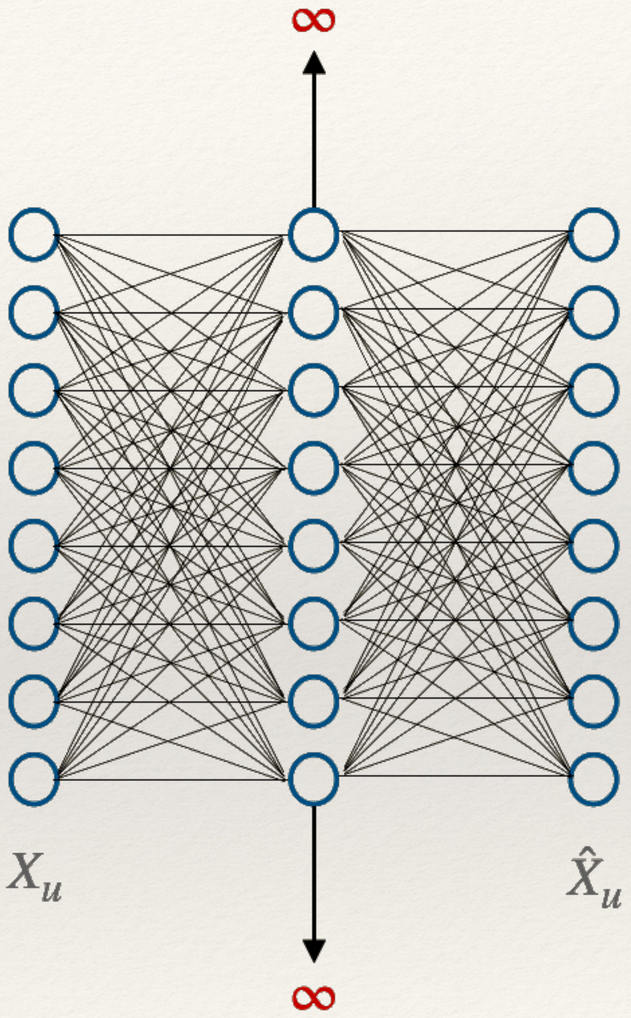
∞-AE

Methodology

- X_u is the bag-of-items representation for user u i.e. all the items that u interacted with, and we aim to reconstruct it along with missing user preferences
- Due to the infinite-width correspondence, ∞-AE optimizes in closed-form:

$$\hat{X} = K \cdot (K + \lambda I)^{-1} \cdot X \quad \text{s.t.} \quad K_{u,v} := \mathbb{K}(X_u, X_v) \quad \forall u, v$$

- The optimization has only a single hyper-parameter λ
- **Time complexity** Training: $\mathcal{O}(U^2 \cdot I + U^{2.376})$ Inference: $\mathcal{O}(U \cdot I)$
- **Memory complexity** Training: $\mathcal{O}(U \cdot I + U^2)$ Inference: $\mathcal{O}(U \cdot I)$



∞-AE

Experiments

Dataset	NeuMF	GCN	MVAE	EASE	∞-AE
Magazine	13.6	22.5	12.1	22.8	**23.0**
ML-1M	25.6	28.8	22.1	29.8	**32.8**
Douban	13.3	16.6	16.1	19.4	**24.9**
Netflix	12.0	—	20.8	26.8	**30.5***

Table: nDCG@10 performance (higher is better) of various recommendation algorithms.
* represents training on 5% random users.

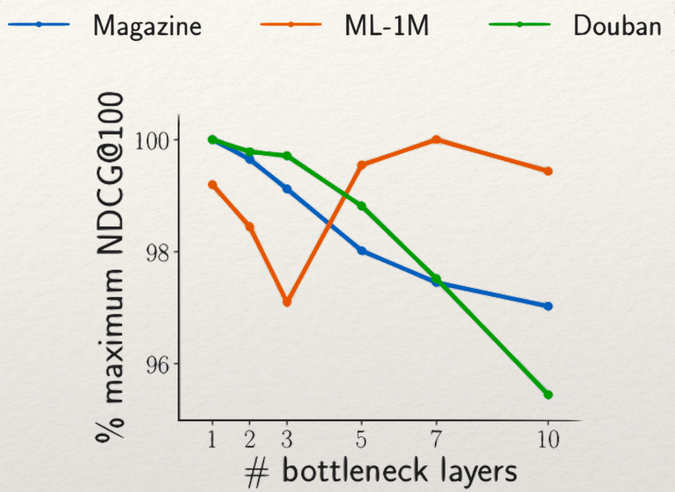


Figure: Performance of ∞-AE with varying depth.

- ∞-AE outperforms various state-of-the-art methods, even when trained on just 5% random users
- 1 layer seems to be enough for optimal recommendation performance: common folk-knowledge
- Even though the model is expensive; it is simplistic, easy to implement (thanks, `JAX`), and the performance is great! But how to scale it up? 🤔

Distill-CF

Data Distillation for Recommendation Data

Key Idea: Use a smooth prior matrix followed by differentiable Gumbel sampling to distill discrete data

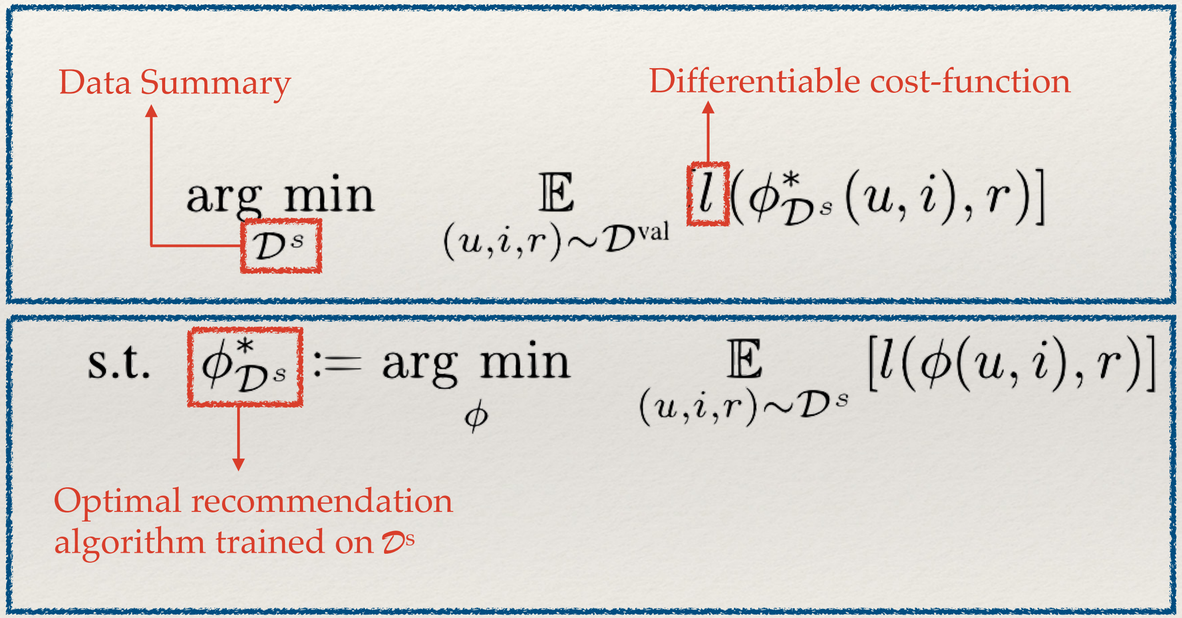
Distill-CF

Overview & Challenges

Unique challenges for distilling recommendation data:

- D^s consists of **discrete** (u, i, r) tuples
- **Semi-structuredness**: some users/items are more popular than others
- D^s is typically extremely **sparse**

Outer loop: optimize the data summary for a fixed learning algorithm



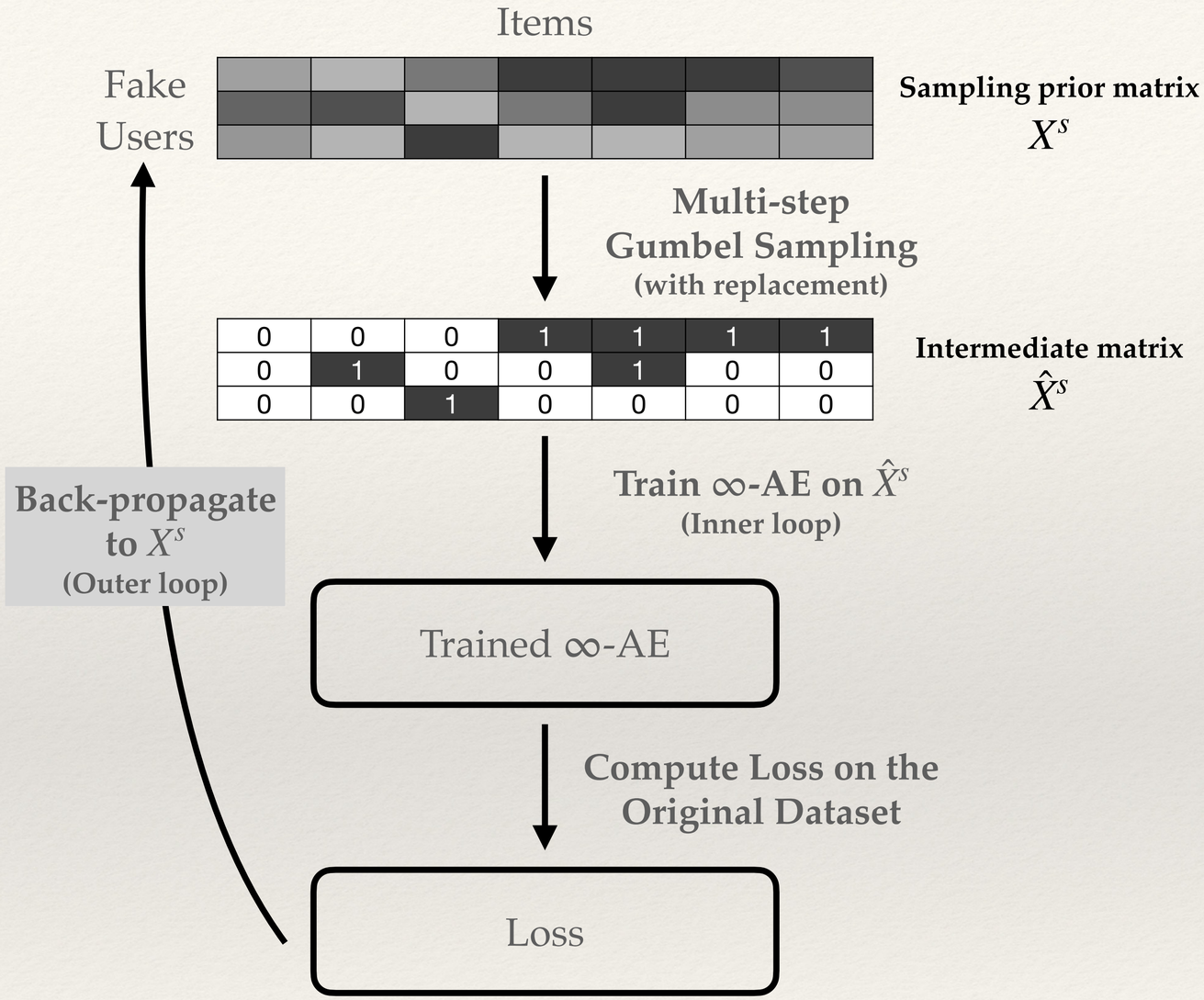
Inner loop: optimize the learning algorithm for a fixed data summary

Distill-CF

Methodology

Robust framework:

- Uses Gumbel sampling on X^s to mitigate the heterogeneity of the problem
- Perform **Gumbel sampling multiple times** for each fake-user to handle dynamic user/item popularity
- **Automatically control sparsity** in $\hat{X}^s$ by controlling the entropy in X^s



Distill-CF

Experiments

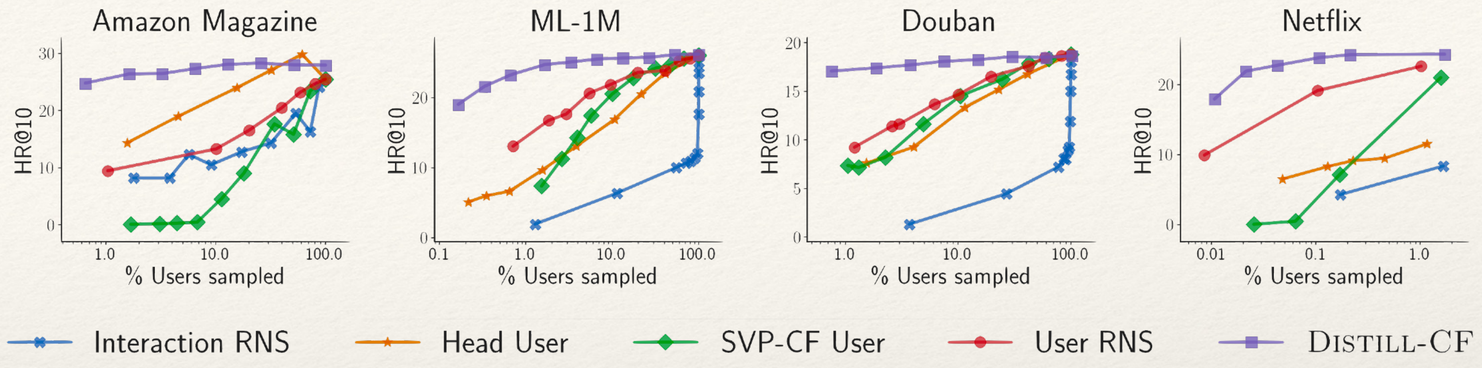


Figure A: Size of data summary vs. trained model quality (Log-scale)

- Using Distill-CF, we can get **96-105%** of full-data performance on as small as **0.1%** data sub-samples, leading to as much as **~1000x** time speedup!
- Distill-CF works well even for the second-best "Baseline" model, even though the data isn't optimized using "Baseline"

Dataset	NeuMF	GCN	MVAE	EASE	∞-AE	∞-AE (Distill-CF)
Magazine	13.6	22.5	12.1	22.8	23.0	**23.8**
ML-1M	25.6	28.8	22.1	29.8	**32.8**	32.5
Douban	13.3	16.6	16.1	19.4	**24.9**	24.2
Netflix	12.0	—	20.8	26.8	**30.5***	**30.5**

Table: nDCG@10 performance of various recommendation algorithms. * represents training on 5% random users. Distill-CF has a user budget of just 500 (0.1% for Netflix).

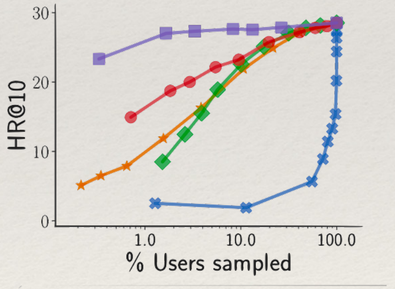


Figure B: Distill-CF + Baseline for the ML-1M dataset.

Farzi Data: Autoregressive Data Distillation

Noveen Sachdeva [1] Zexue He [1] Benjamin Coleman [2]
Wang-Cheng Kang [2] Jianmo Ni [2] Derek Z. Cheng [2] Julian McAuley [1]

University of California, San Diego [1] Google DeepMind [2]

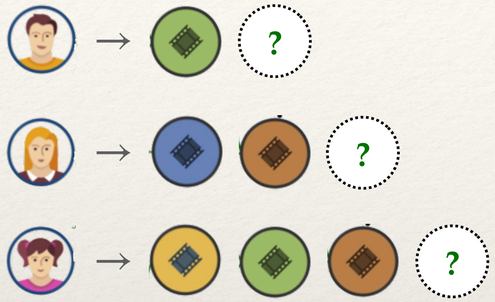




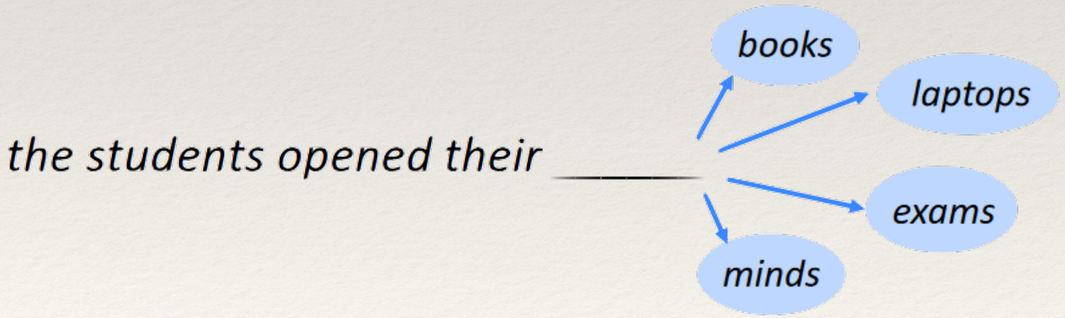


Scope

1. Sequential Recommender Systems



2. Language Modeling



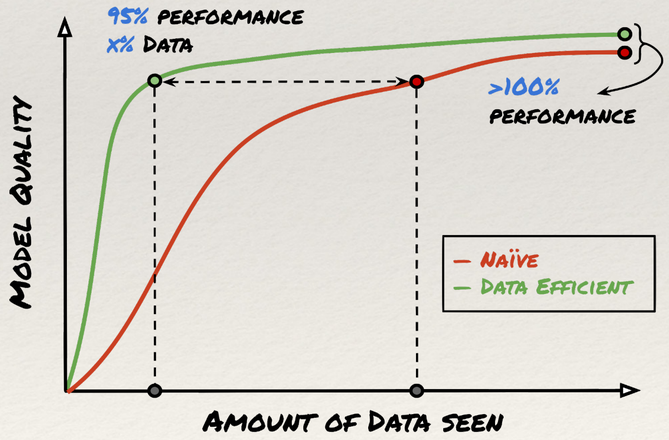
Objective

Perform Accurate Recommendation / LM

That is, learn better next-item / token predictors:

- $\delta : [\text{item}_1, \text{item}_2, \ldots, \text{item}_n] \mapsto \mathcal{I};\ \forall\ \text{item}_i \in \mathcal{I}$
- $\delta : [\text{token}_1, \text{token}_2, \ldots, \text{token}_n] \mapsto \mathcal{T};\ \forall\ \text{token}_i \in \mathcal{T}$

Naive vs. Data-Efficient



Naive:

Train the model on the entire dataset

Data-Efficient:

Train the model on the distilled version of the dataset

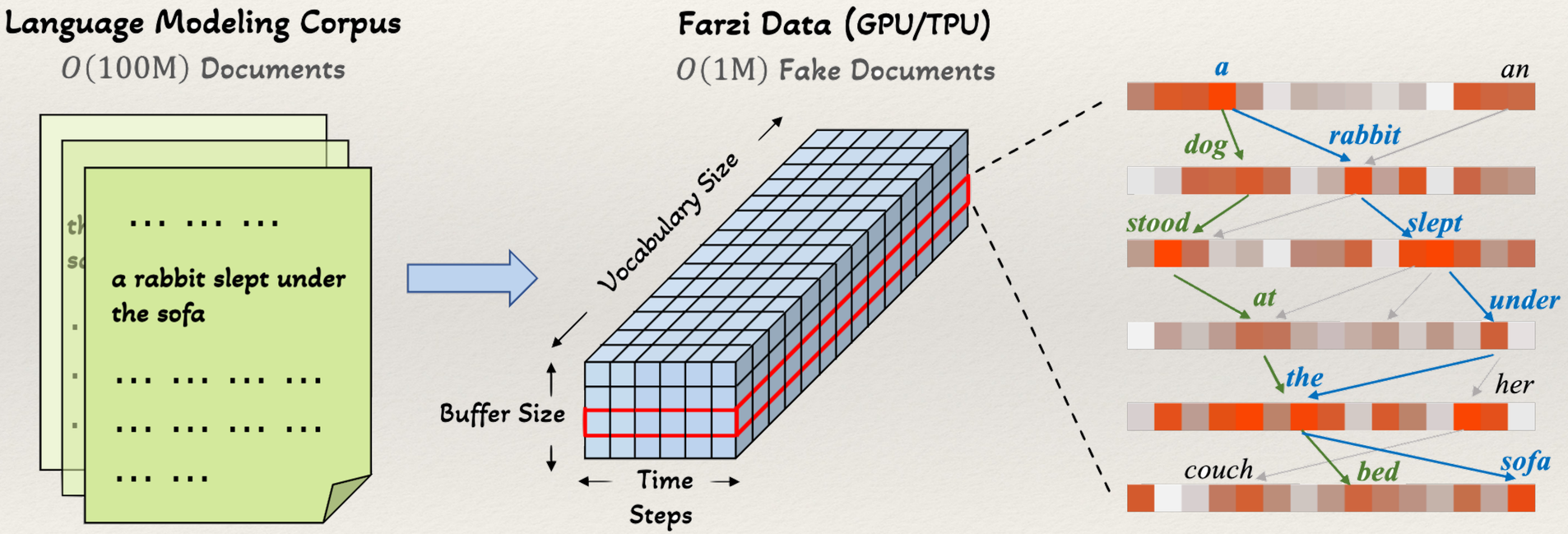
Farzi

Distilling Auto-Regressive Data

Key Idea: Think of a discrete **sequence-of-events** as a **sequence-of-distributions** that can be now distilled via data distillation

Farzi

Intuition



Farzi

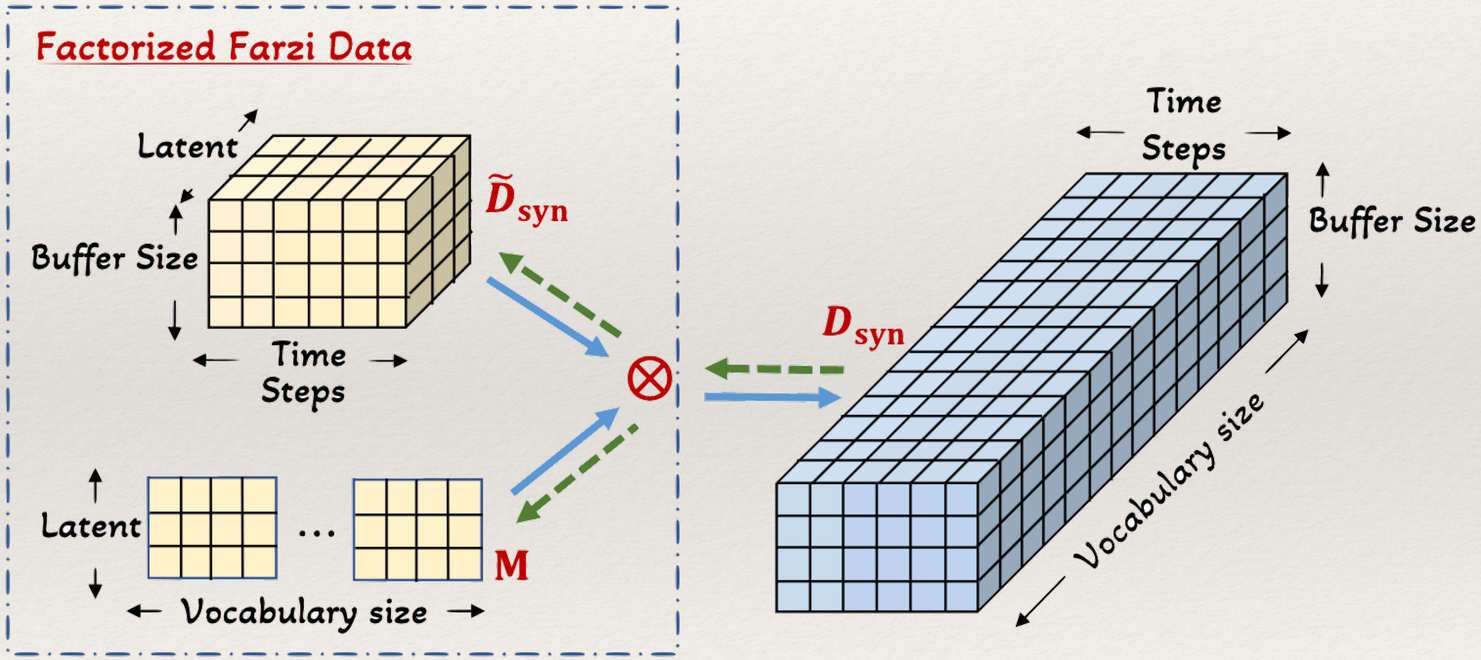
Can we distill this 3d tensor?

Challenge:

The data summary is 3-dimensional $\Longrightarrow$ computationally intractable

Idea:

Keep a factorized data summary instead!



Farzi

Methodology (Contd.)

Challenge:

No closed-form inner-loop solvers $\Longrightarrow$
How to get meta-gradient?

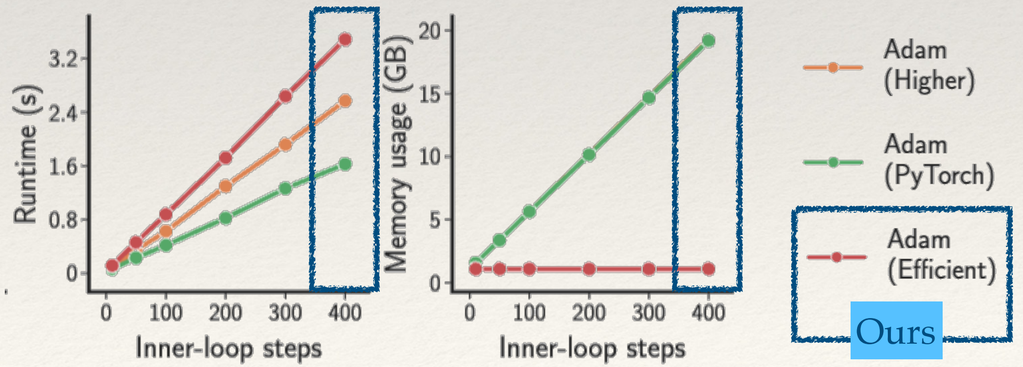
Solution:

Efficient reverse-mode Adam derivation

- Naïve auto-diff memory complexity: $\mathcal{O}(T \cdot \mathcal{G})$
- Reverse-mode Adam memory complexity: $\mathcal{O}(\mathcal{G})$

Algorithm 1 Reverse-mode differentiation of Adam.

Input: $\mathbf{w}_T$, $\mathbf{m}_T$, $\mathbf{v}_T$, γ, α, ϵ, $L(w, x)$, meta-objective $f(w)$
Initialize: $d\mathbf{m} \leftarrow 0$, $d\mathbf{x} \leftarrow 0$, $d\mathbf{w} \leftarrow \nabla_{\mathbf{w}} f(\mathbf{w}_T)$
for $t = T$ **to** 1 **do**
  $\hat{\mathbf{m}}_t \triangleq \mathbf{m}_t / (1 - \beta_1^t)$
  $\hat{\mathbf{v}}_t \triangleq \mathbf{v}_t / (1 - \beta_2^t)$
  $\mathbf{w}_{t-1} = \mathbf{w}_t + \alpha \cdot \hat{\mathbf{m}}_t / (\hat{\mathbf{v}}_t + \epsilon)$
  $\mathbf{g}_t \triangleq \nabla_{\mathbf{w}} L(\mathbf{w}_{t-1}, \mathbf{x})$
  $\mathbf{m}_{t-1} = [\mathbf{m}_t - (1 - \beta_1) \cdot \mathbf{g}_t] / \beta_1$
  $\mathbf{v}_{t-1} = [\mathbf{v}_t - (1 - \beta_2) \cdot \mathbf{g}_t^2] / \beta_2$
  $\epsilon' \triangleq \epsilon \cdot \sqrt{1 - \beta_2^t}$
  $\alpha' \triangleq \alpha \cdot \sqrt{1 - \beta_2^t} \,/\, (1 - \beta_1^t)$
  $\beta' \triangleq (1 - \beta_2) \,/\, (1 - \beta_1)$
  Derivation: $d\mathbf{m} = d\mathbf{m} + \alpha' \cdot \left(\frac{\beta' \cdot \mathbf{m}_t \cdot \mathbf{g}_t}{\sqrt{\mathbf{v}_t} \cdot (\sqrt{\mathbf{v}_t} + \epsilon')^2} - \frac{1}{\sqrt{\mathbf{v}_t} + \epsilon'} \right) \cdot d\mathbf{w}$
  Hessian Vector Products: $d\mathbf{w} = d\mathbf{w} - (1 - \beta_1) \cdot d\mathbf{m} \cdot \nabla_{\mathbf{w}} \nabla_{\mathbf{w}} L(\mathbf{w}_{t-1}, \mathbf{x})$
  Hessian Vector Products: $d\mathbf{x} = d\mathbf{x} - (1 - \beta_1) \cdot d\mathbf{m} \cdot \nabla_{\mathbf{x}} \nabla_{\mathbf{w}} L(\mathbf{w}_{t-1}, \mathbf{x})$
  $d\mathbf{m} = \beta_1 \cdot d\mathbf{m}$
Output: gradient of $f(\mathbf{w}_T)$ *w.r.t.* $\mathbf{w}_0$, $\mathbf{m}_0$, and $\mathbf{x}$



Farzi

Experiments

- Using Farzi, we can get **98-120%** of full-data performance on as small as **0.1%** data sub-samples, leading to as much as **~1000x** time speedup!

- Farzi also improves the performance of models on the tail-portion of users and items — which is of very valuable importance in practice

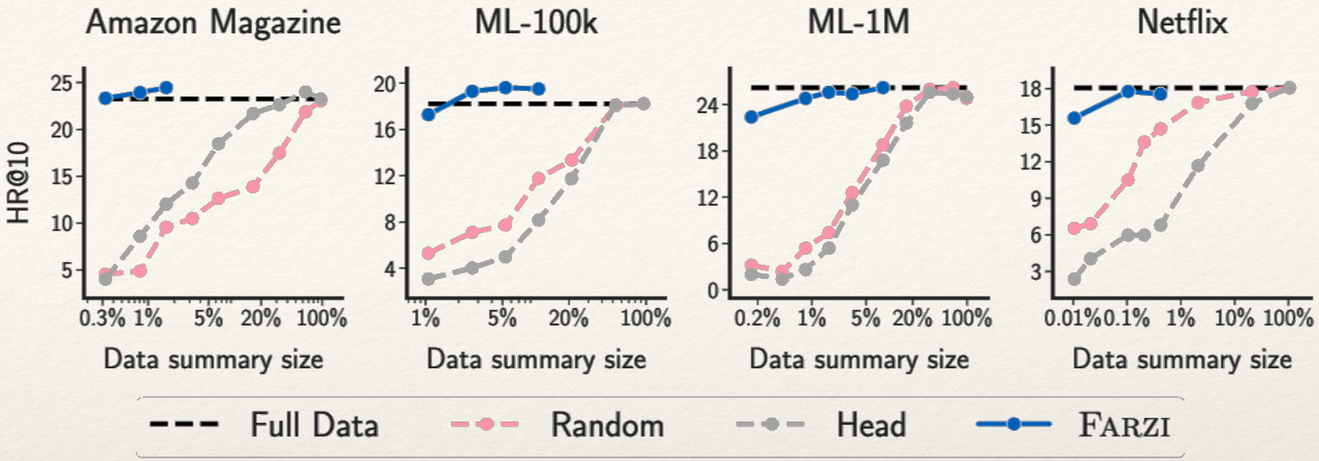


Figure A: Size of data summary vs. trained model quality (Log-scale)

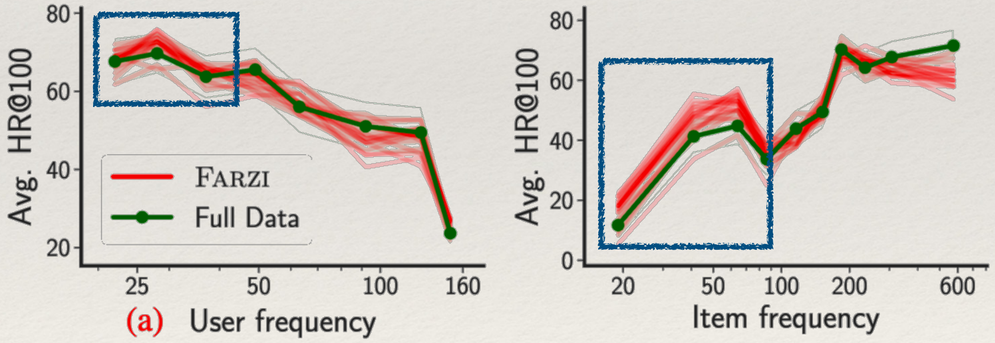


Figure B: Performance of models trained on Farzi Data vs. Full Data on the user/item coldness spectrum.

This Dissertation

Future Roadmap

New Data Modalities

- Language: SFT, RLHF
- Audio
- Video
- …

Data Optimization

- Efficiency: Scalable ways to perform data distillation for bigger models & datasets
- Transferability: Better ways to create universal, drop-in replacement data summaries
- Order-sensitive data optimization techniques

New Applications

- Continual Learning
- Neural Architecture Search
- Hyper-parameter Opt.

Fairness & Privacy

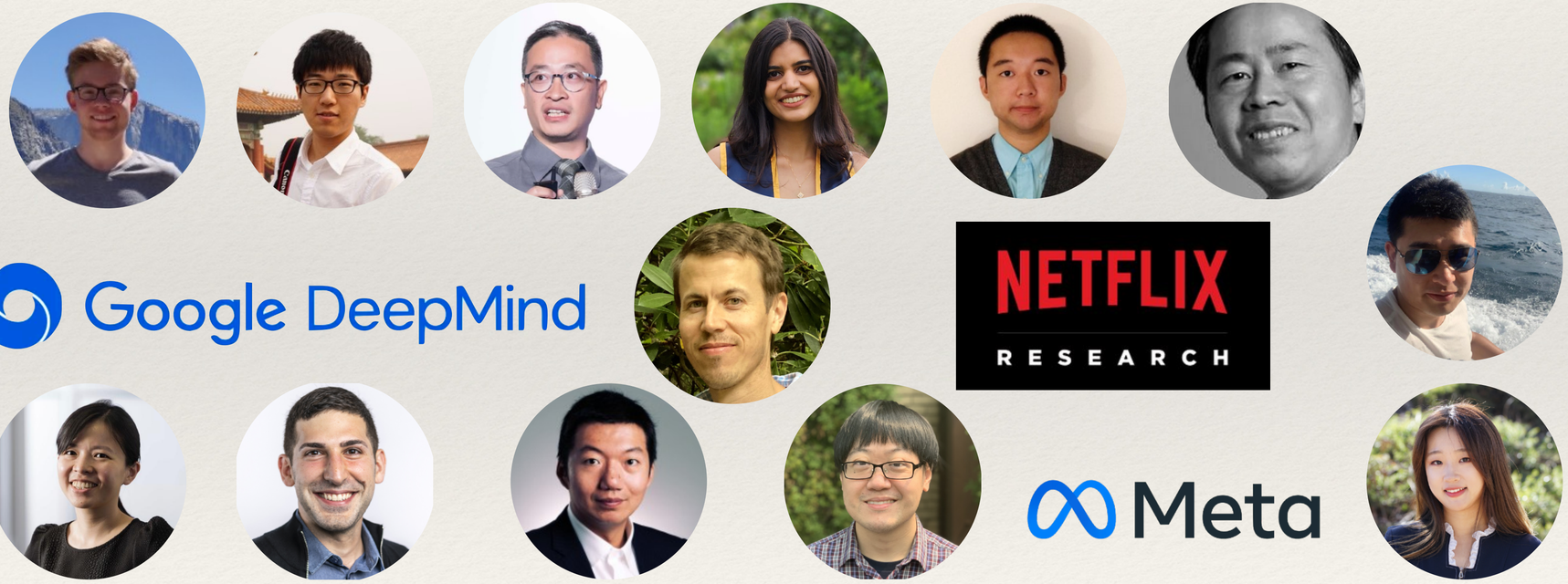
- How to optimize for these constraints while sampling / distillation
- DP: Can we guarantee impossibility of de-anonymization when learning on data summaries?

Gratitude

The McAuley Lab
UC San Diego

Julian McAuley
UC San Diego

"Best Advisor Ever."



My Wonderful Collaborators

Thank you!
Questions?

@noveens97

For papers & code: noveens.com

What we covered:

01 What is Data-Efficiency

02 Data Sampling for RecSys

03 Data Sampling for LLMs

04 Data Distillation for RecSys

05 Data Distillation for Autoregressive Data