Data Selection via Optimal Control for Language Models

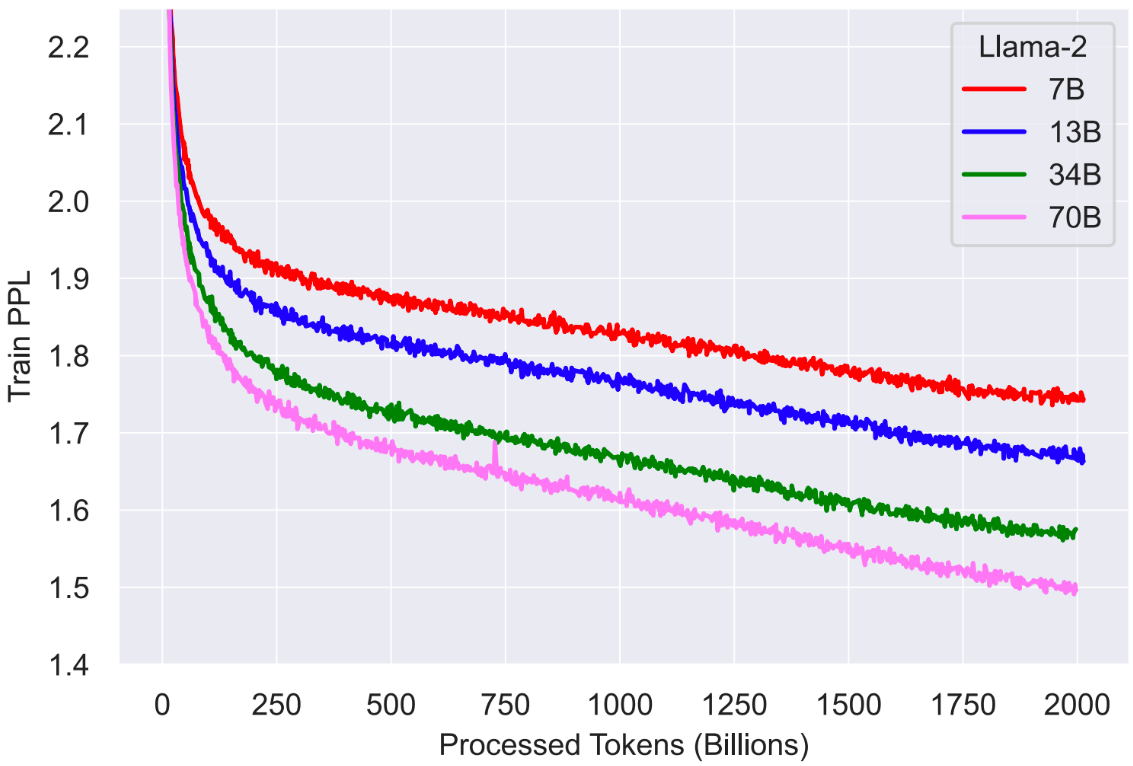
Yuxian Gu[1,2], Li Dong[2], Hongning Wang[1], Yaru Hao[2],
Qingxiu Dong[2], Minlie Huang[1], Furu Wei[2]

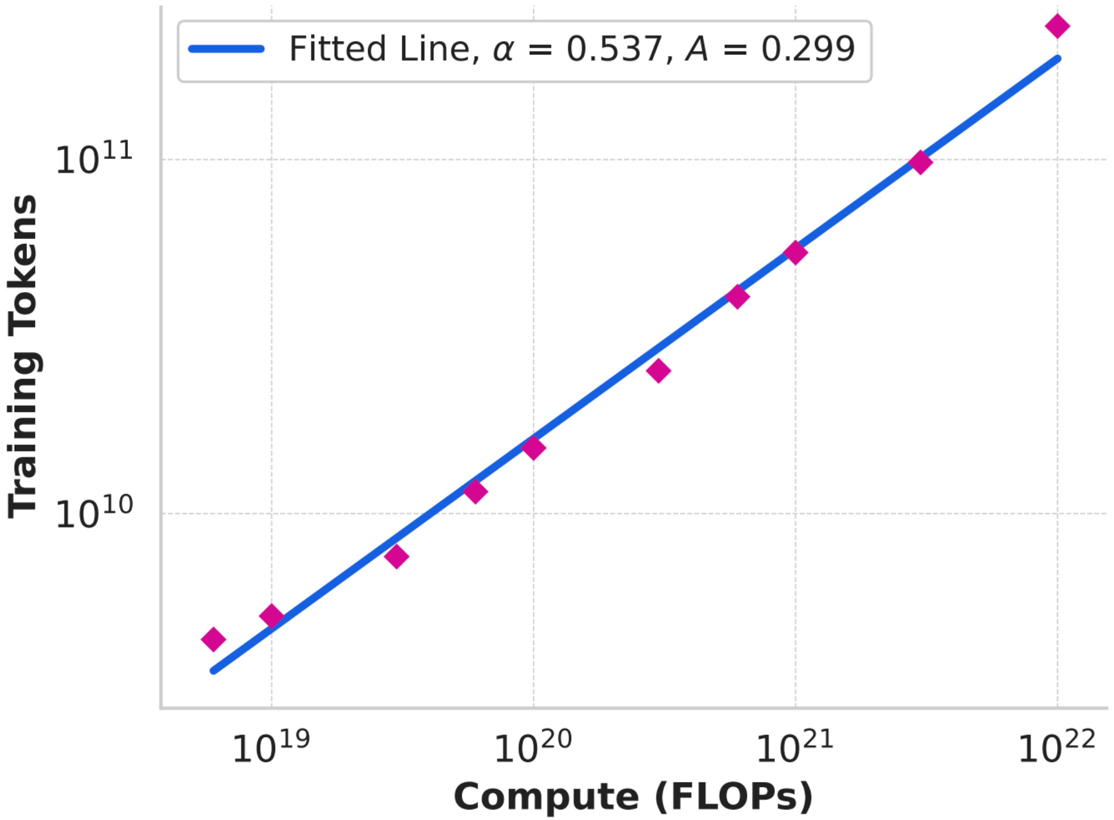
[1]The CoAI Group, Tsinghua University

[2]Microsoft Research

Data challenges for pre-training LMs

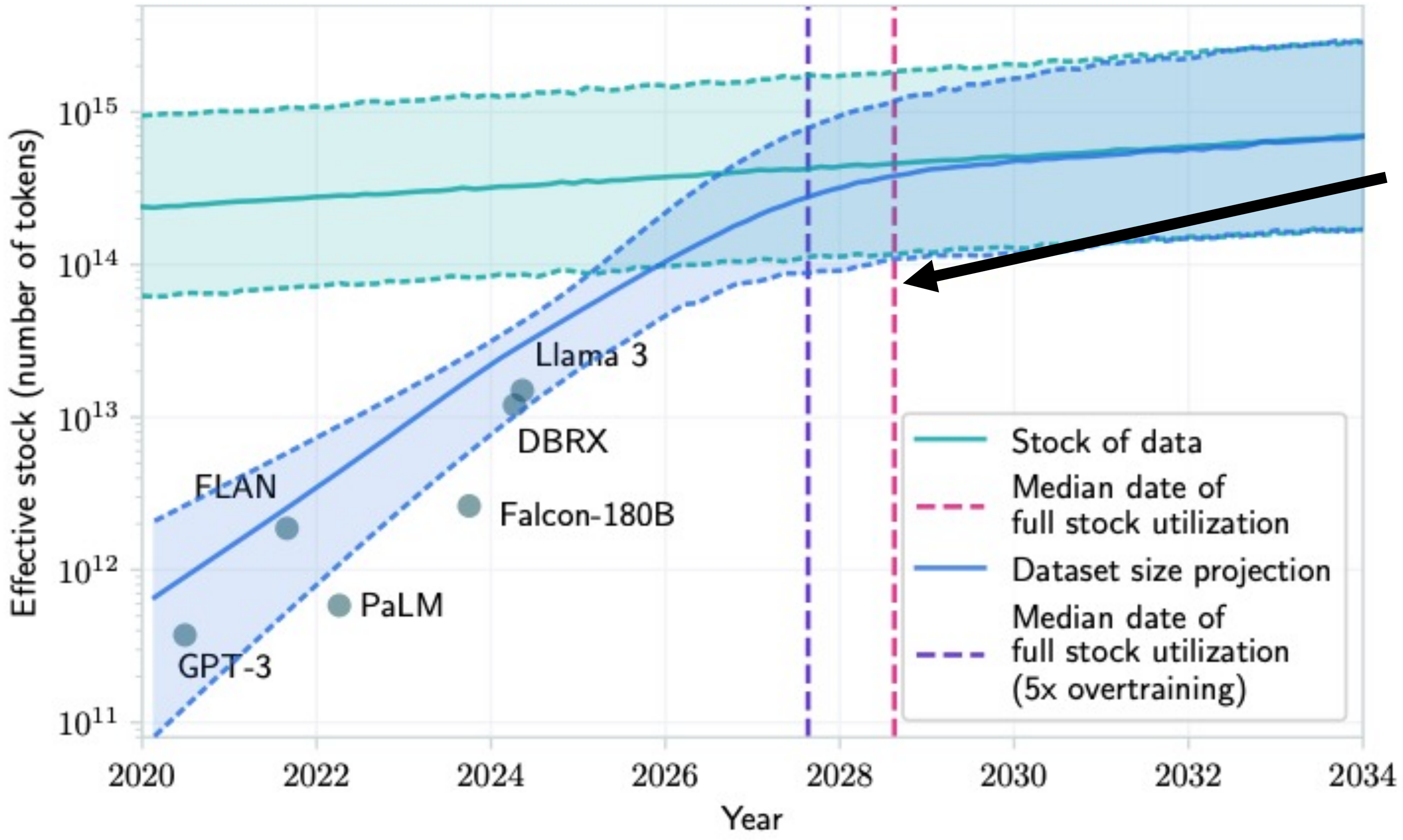
- Large amount of data makes pre-training quite inefficient.





Data challenges for pre-training LMs

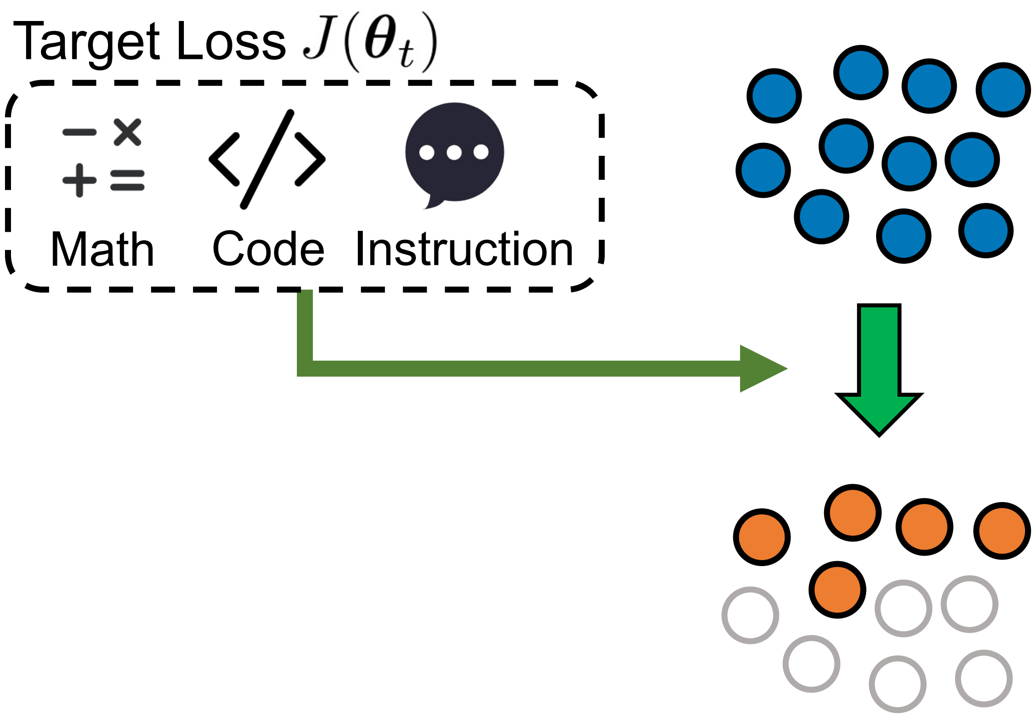
- High-quality pre-training data is running out.

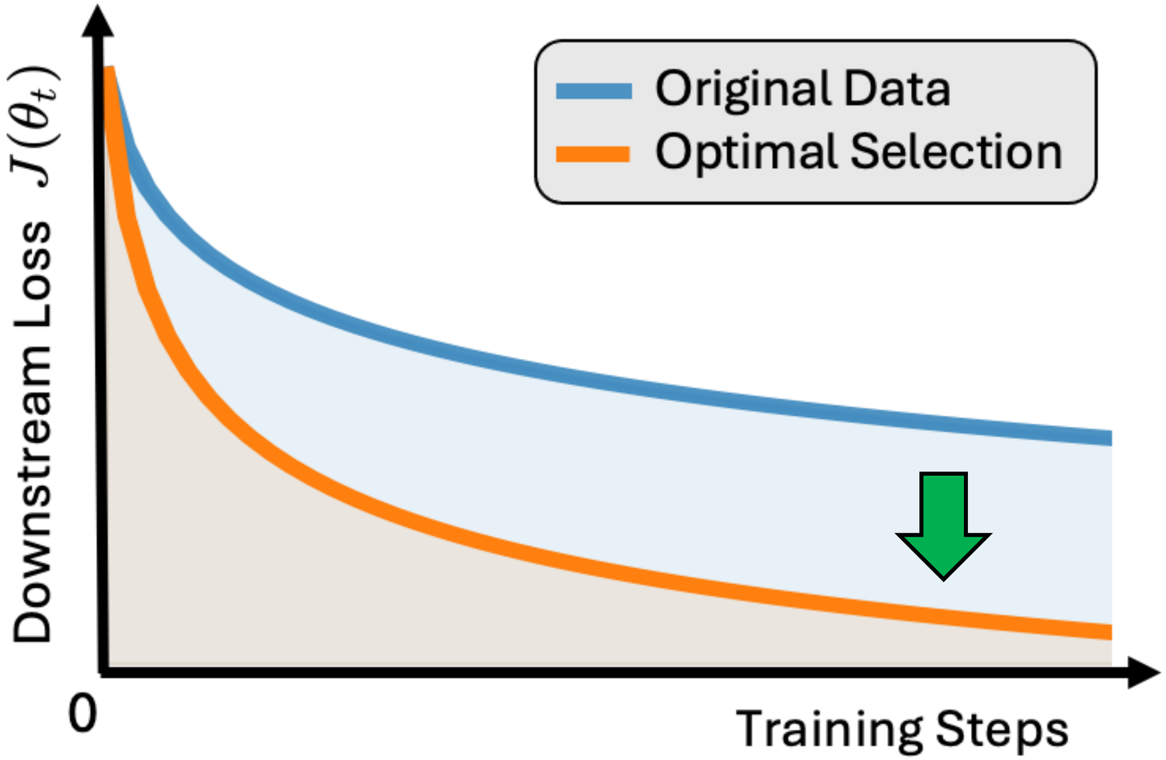


Models consume faster than humans produce.

Possible Solution: Data Selection?

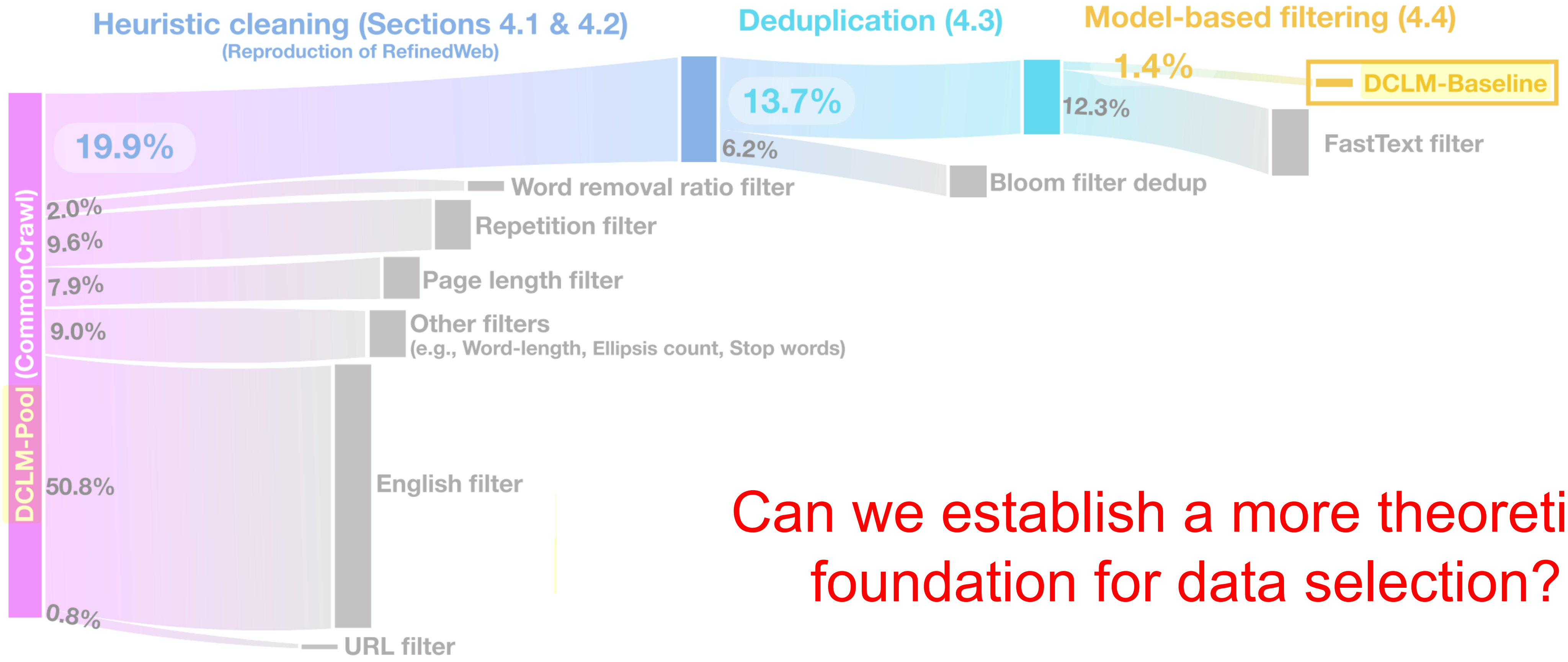
- Select a Pre-Training Corpus Subset for Better Target Performance
  - Target: Math, Code, High-Quality Instruction, etc..





Challenges for Data Selection

- Current data selection/filtering is heuristic-based and tricky task.



Can we establish a more theoretical foundation for data selection?

Overview

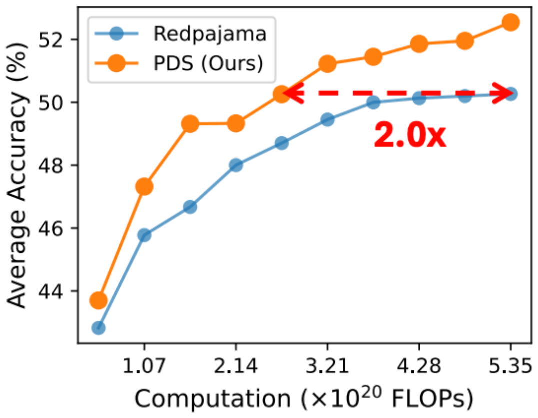
- Data challenges for pre-training LMs
  - Large amount of data makes pre-training quite inefficient.
  - High-quality pre-training data is running out.
  - Data selection/cleaning is a heuristic-based tricky task.

- **PDS**: Data Selection via Optimal Control for Pre-Training

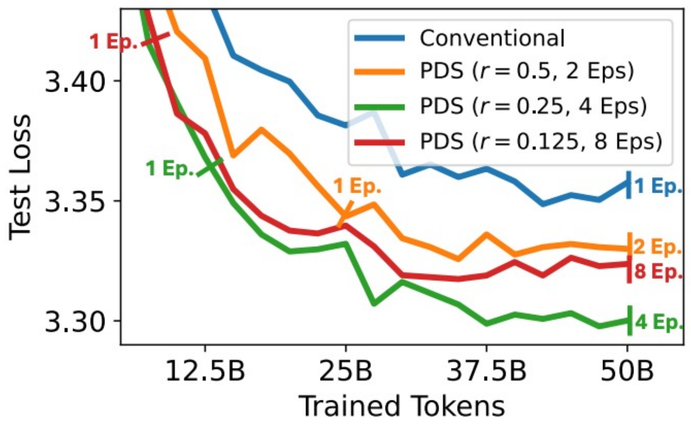
Theorem 2.1: PMP Conditions for Data Selection

$$\boldsymbol{\theta}^*_{t+1} = \boldsymbol{\theta}^*_t - \eta \nabla L(\boldsymbol{\theta}^*_t, \boldsymbol{\gamma}^*), \quad \boldsymbol{\theta}^*_0 = \boldsymbol{\theta}_0,$$
$$\boldsymbol{\lambda}^*_t = \boldsymbol{\lambda}^*_{t+1} + \nabla J(\boldsymbol{\theta}^*_t) - \eta \nabla^2 L(\boldsymbol{\theta}^*_t, \boldsymbol{\gamma}^*)\boldsymbol{\lambda}^*_{t+1},$$
$$\boldsymbol{\gamma}^* = \arg\max_{\boldsymbol{\gamma}} \sum_{n=1}^{|\mathcal{D}|} \gamma_n \left[\sum_{t=0}^{T-1} {\boldsymbol{\lambda}^*_{t+1}}^\top \nabla l(x_n, \boldsymbol{\theta}^*_t) \right]$$

Good theoretical foundation



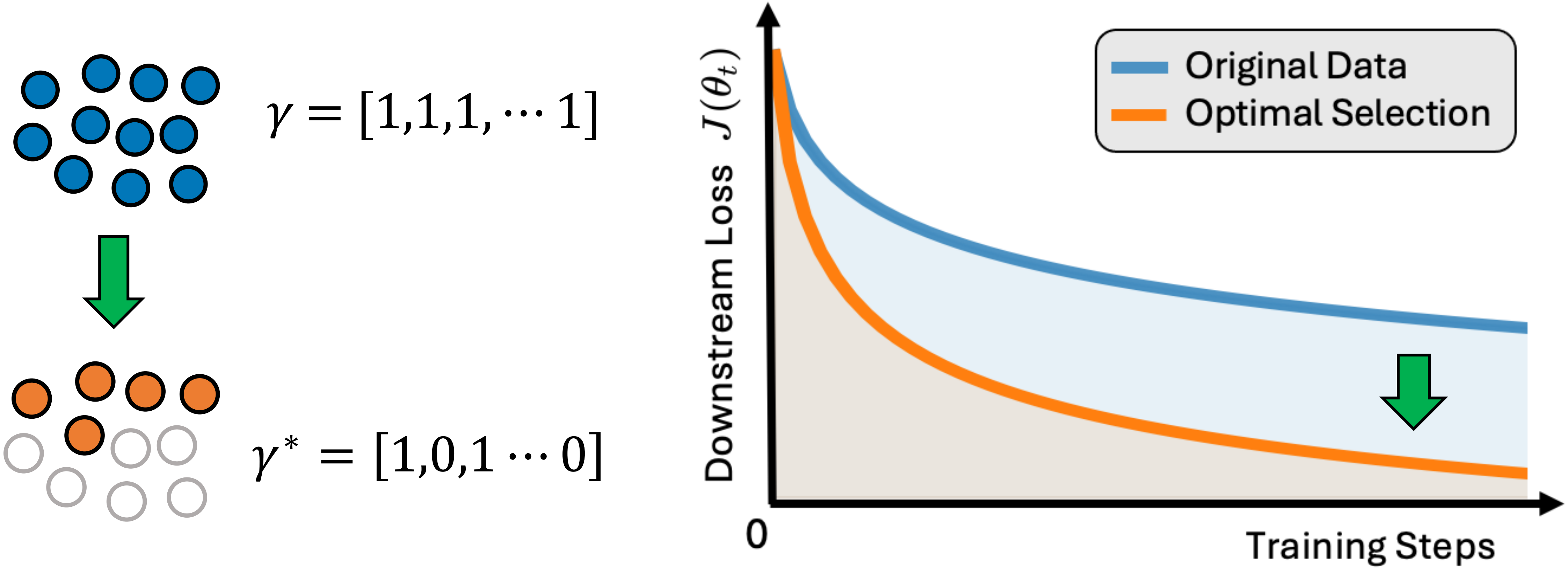
2x acceleration



Improvement on limited data

Formulate Data Selection

- Optimize the data selection strategy for lower downstream loss
  - γ: indicates a sample is selected or not



Training on the Selected Data

- Loss: Treat the $\boldsymbol{\gamma}$ as the weights of the instance losses:

$$L(\boldsymbol{\theta}, \gamma) = \sum_{n=1}^{|\mathcal{D}|} \gamma_n l(x_n, \boldsymbol{\theta})$$

$\gamma_n = 1$: $l(x_n, \theta)$ is selected

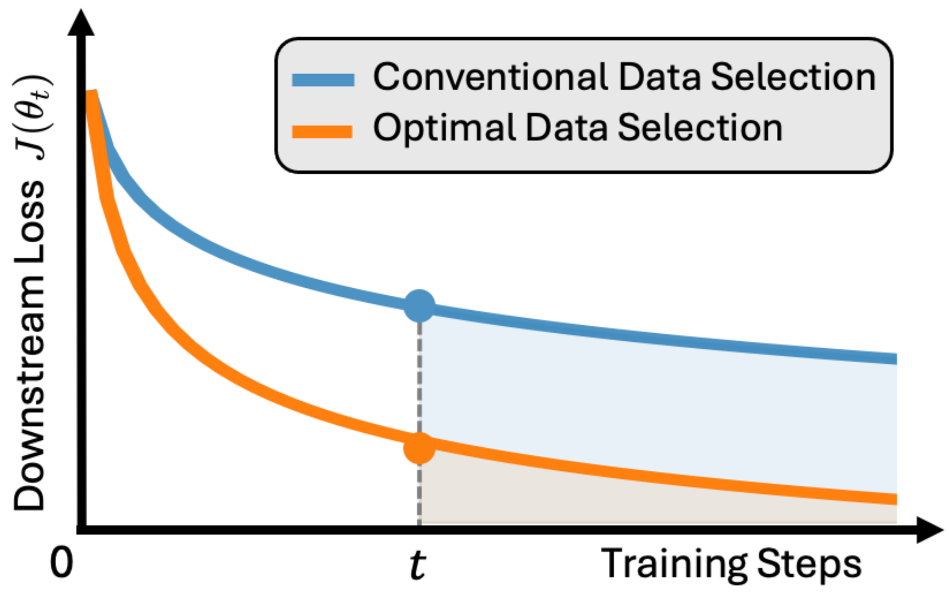
$\gamma_n = 0$: $l(x_n, \theta)$ is ignored

- The model is trained with:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla L(\boldsymbol{\theta}_t, \gamma)$$

Target to Optimize

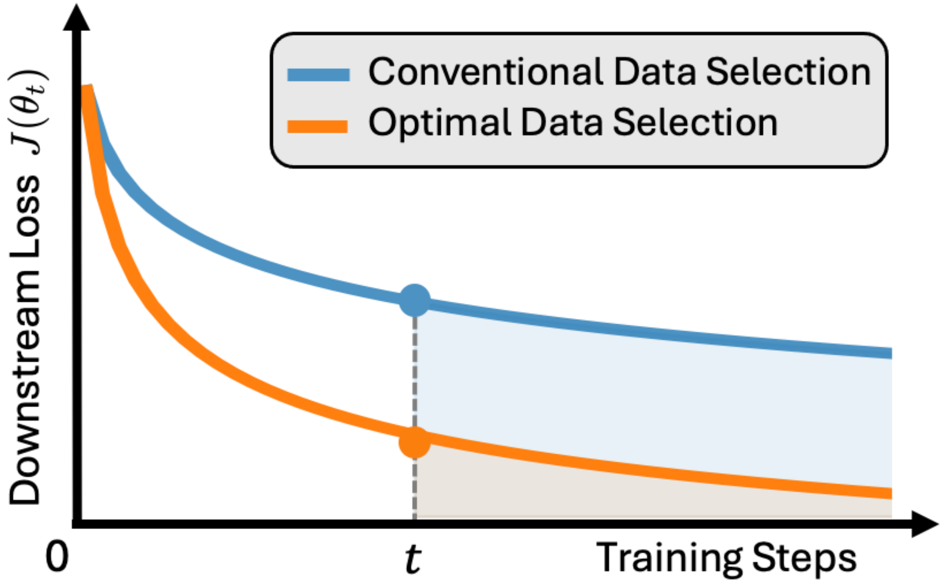
- $J(\boldsymbol{\theta}_t)$: downstream loss to minimize (like on math, code, etc.)
- Minimizing the Aera Under the Loss curve (AUC)
  - AUC is directly related to the Scaling Law constants (see Appendix A in our paper)
  - Optimizing AUC is improving the Scaling Law!



$$\min_{\gamma} \sum_{t=1}^{T} J(\boldsymbol{\theta}_t),$$
$$\text{s.t. } \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla L(\boldsymbol{\theta}_t, \boldsymbol{\gamma})$$

Target to Optimize

- $J(\boldsymbol{\theta}_t)$: downstream loss to minimize (like on math, code, etc.)
- Minimizing the Aera Under the Loss curve (AUC)
  - AUC is directly related to the Scaling Law constants (see Appendix A in our paper)
  - Optimizing AUC is improving the Scaling Law!



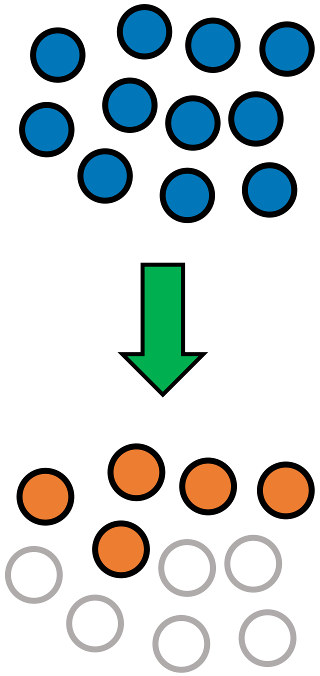
$$\min_{\boldsymbol{\gamma}} \sum_{t=1}^{T} J(\boldsymbol{\theta}_t),$$

$$\text{s.t. } \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla L(\boldsymbol{\theta}_t, \boldsymbol{\gamma})$$

Hard to solve? Optimal Control!

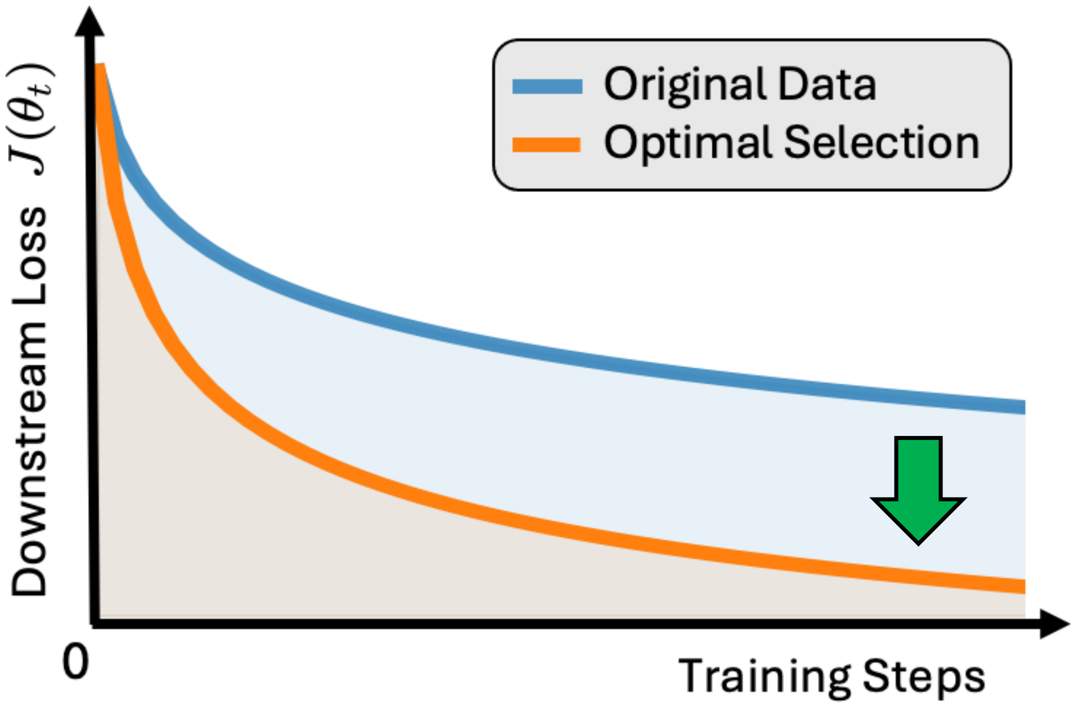
Data Selection as a Control Problem

- Original Problem: Optimize the data selection strategy
  - $\boldsymbol{\gamma}$: indicates a sample is selected or not



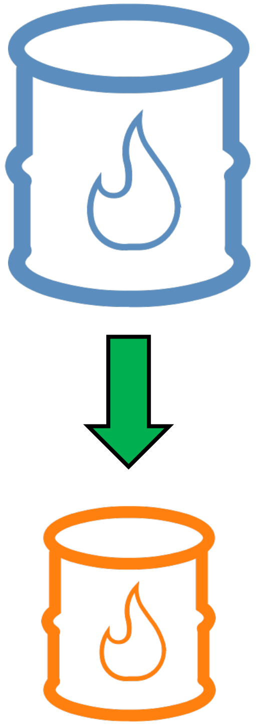
$\gamma = [1,1,1,\cdots 1]$

$\gamma^* = [1,0,1 \cdots 0]$



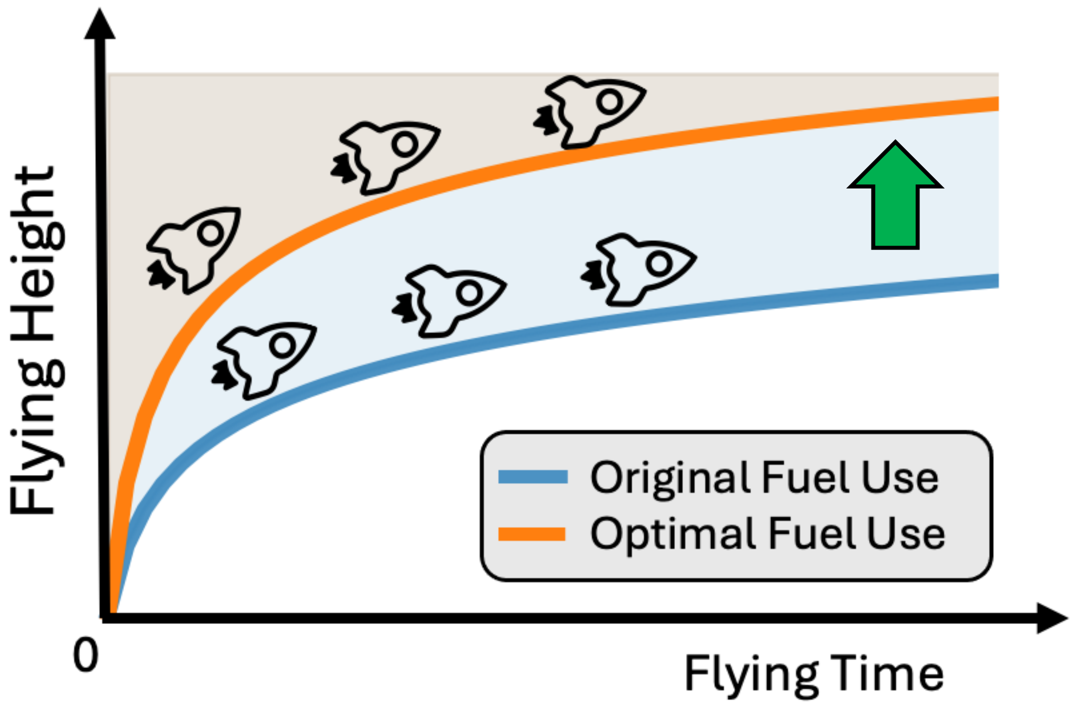
Data Selection as a Control Problem

- Analogy: Optimizing fuel use when flying a rocket
  - Data is the "fuel" in pre-training language models

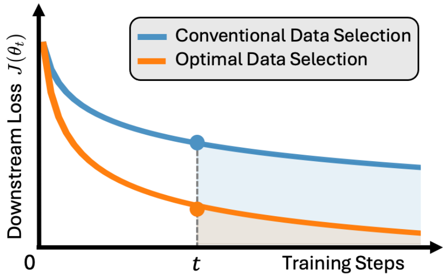
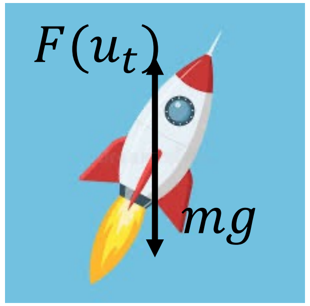


Original Fuel Use

Optimized Fuel Use



Mathematical Equivalence to Optimal Control

	Data Selection for LMs	Fuel Use Optimization
		
Control Variable	Selection Strategy: γ	Fuel Consumption: u_t
Objective	Minimal AUC: $\min_{\gamma} \sum_{t=1}^{T} J(\boldsymbol{\theta}_t),$	Maximize Distance: $\max_{u_t} x = \sum_{t=0}^{T} v_t \, \Delta t$
Constraints	Regularity: $\gamma \in U.$	Constant Total Fuel: $\sum_{t=0}^{T} u_t = U$
Dynamics	Gradient decent: $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla L(\boldsymbol{\theta}_t, \boldsymbol{\gamma})$	Newton's Law: $\frac{v_{t+\Delta t} - v_t}{\Delta t} = -mg + F(u_t)$

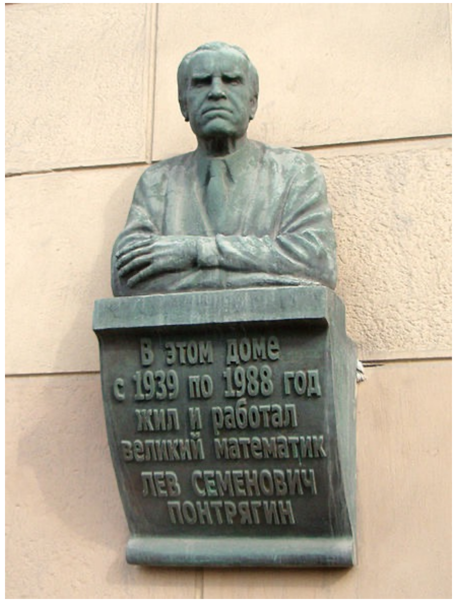
Solving the Problem

- **P**ontryagin's **M**aximum **P**rinciple (PMP)
  - Gives **<u>necessary conditions</u>** for the optimality of the problem

$$\min_{\gamma} \sum_{t=1}^{T} J(\boldsymbol{\theta}_t),$$

$$\text{s.t.} \quad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \nabla L(\boldsymbol{\theta}_t, \gamma)$$

$$L(\boldsymbol{\theta}, \gamma) = \sum_{n=1}^{|\mathcal{D}|} \gamma_n l(x_n, \boldsymbol{\theta})$$



Lev Pontryagin, 1908 - 1988

Conditions for Optimal Data Selection

Theorem 2.1 (PMP Conditions for Data Selection).

$$\begin{cases} \#1 \quad \boldsymbol{\theta}^*_{t+1} = \boldsymbol{\theta}^*_t - \eta \nabla L(\boldsymbol{\theta}^*_t, \boldsymbol{\gamma}^*), \\ \#2 \quad \boldsymbol{\lambda}^*_t = \boldsymbol{\lambda}^*_{t+1} + \nabla J(\boldsymbol{\theta}^*_t) - \eta \nabla^2 L(\boldsymbol{\theta}^*_t, \boldsymbol{\gamma}^*) \boldsymbol{\lambda}^*_{t+1}, \\ \#3 \quad \boldsymbol{\gamma}^* = \arg\max_{\boldsymbol{\gamma}} \sum_{n=1}^{|\mathcal{D}|} \gamma_n \left[\sum_{t=0}^{T-1} {\boldsymbol{\lambda}^*_{t+1}}^\top \nabla l(x_n, \boldsymbol{\theta}^*_t) \right] \end{cases}$$

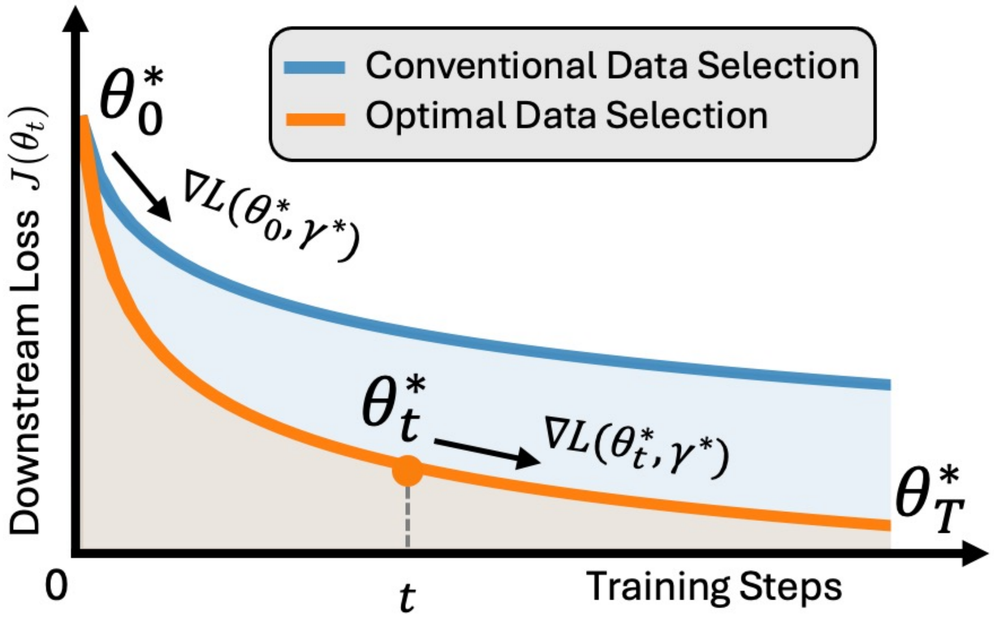
We can get the optimal data score γ^* here!

#1: Learning Condition

$$\theta_{t+1}^{*} = \theta_{t}^{*} - \eta \nabla L(\theta_{t}^{*}, \gamma^{*})$$

Model Parameters   Optimal Data Selection Strategy

- Exactly the parameter updating policy of training LMs
- Constrains the θ_t^* to be reachable with GD *under the optimal data selection*

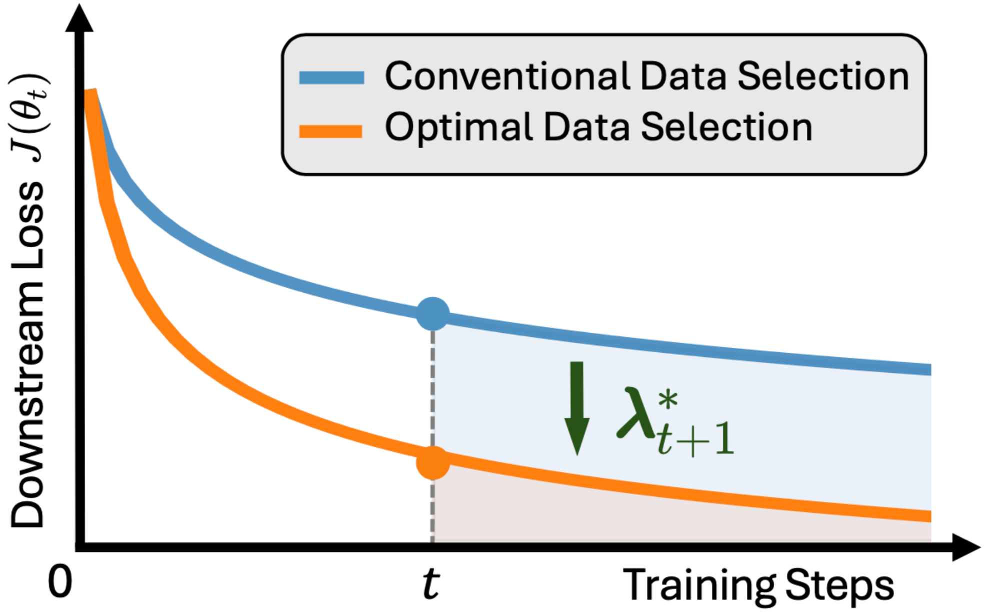


#2: Target Condition

$$\boldsymbol{\lambda}_t^* = \boldsymbol{\lambda}_{t+1}^* + \nabla J(\boldsymbol{\theta}_t^*) - \eta \nabla^2 L(\boldsymbol{\theta}_t^*, \gamma^*) \boldsymbol{\lambda}_{t+1}^*$$

Ideal gradient = Target + Learning dynamics

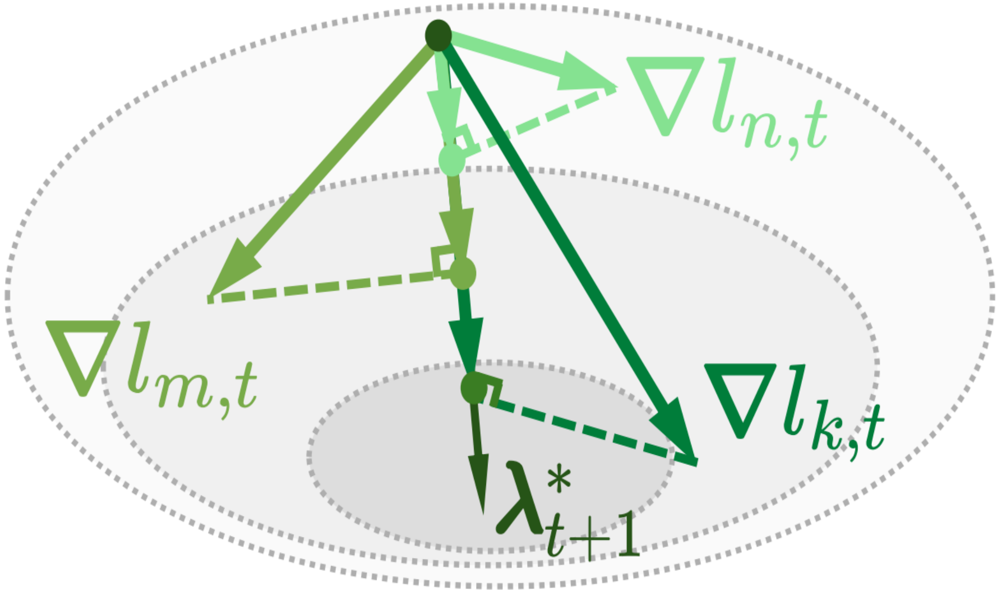
- $\boldsymbol{\lambda}_t^*$: **the ideal gradient** of the high-quality data points.
  - *<u>"**Compass**" for high-quality data.</u>*
- Ideal gradient includes information of Target Loss and Learning Dynamics.



#3: Maximum Condition

$$\boldsymbol{\gamma}^* = \arg\max_{\boldsymbol{\gamma}} \sum_{n=1}^{|\mathcal{D}|} \gamma_n \left[\sum_{t=0}^{T-1} {\boldsymbol{\lambda}_{t+1}^*}^\top \nabla l(x_n, \boldsymbol{\theta}_t^*) \right]$$

Gradient of each sample



$$\sum_t {\lambda_{t+1}^*}^\top \nabla l_{n,t} < \sum_t {\lambda_{t+1}^*}^\top \nabla l_{m,t} < \sum_t {\lambda_{t+1}^*}^\top \nabla l_{k,t}$$

$$\Rightarrow \quad \gamma_n < \gamma_m < \gamma_k$$

Examples with **closer gradients to λ_t** should **have higher γ.**

Summing up

Theorem 2.1 (PMP Conditions for Data Selection).

#1 Learning Condition $\boldsymbol{\theta}_{t+1}^{*} = \boldsymbol{\theta}_{t}^{*} - \eta \nabla L(\boldsymbol{\theta}_{t}^{*}, \boldsymbol{\gamma}^{*})$

#2 Target Condition $\boldsymbol{\lambda}_{t}^{*} = \boldsymbol{\lambda}_{t+1}^{*} + \nabla J(\boldsymbol{\theta}_{t}^{*}) - \eta \nabla^{2} L(\boldsymbol{\theta}_{t}^{*}, \boldsymbol{\gamma}^{*}) \boldsymbol{\lambda}_{t+1}^{*}$

#3 Maximum Condition $\boldsymbol{\gamma}^{*} = \arg\max_{\boldsymbol{\gamma}} \sum_{n=1}^{|\mathcal{D}|} \gamma_n \left[\sum_{t=0}^{T-1} {\boldsymbol{\lambda}_{t+1}^{*}}^{\top} \nabla l(x_n, \boldsymbol{\theta}_{t}^{*}) \right]$

How to Solve (Theoretically)?

Use <u>learning condition</u> to forward compute $\boldsymbol{\theta}_0$ to $\boldsymbol{\theta}_T$

`forward Pass with #1:` $\boldsymbol{\theta}_0 \longrightarrow \boldsymbol{\theta}_1 \longrightarrow \ldots \longrightarrow \boldsymbol{\theta}_T$

#2 Target Condition $\boldsymbol{\lambda}_t^* = \boldsymbol{\lambda}_{t+1}^* + \nabla J(\boldsymbol{\theta}_t^*) - \eta \nabla^2 L(\boldsymbol{\theta}_t^*, \boldsymbol{\gamma}^*) \boldsymbol{\lambda}_{t+1}^*$

#3 Maximum Condition $\boldsymbol{\gamma}^* = \arg\max_{\boldsymbol{\gamma}} \sum_{n=1}^{|\mathcal{D}|} \gamma_n \left[\sum_{t=0}^{T-1} {\boldsymbol{\lambda}_{t+1}^*}^\top \nabla l(x_n, \boldsymbol{\theta}_t^*) \right]$

How to Solve (Theoretically)?

Use <u>Target Condition</u> to reverse compute $\boldsymbol{\lambda}_T$ to $\boldsymbol{\lambda}_0$

```
forward Pass with #1:
```
$$\boldsymbol{\theta}_0 \longrightarrow \boldsymbol{\theta}_1 \longrightarrow \dots \longrightarrow \boldsymbol{\theta}_T$$

```
reverse Pass with #2:
```
$$\boldsymbol{\lambda}_0 \longleftarrow \boldsymbol{\lambda}_1 \longleftarrow \dots \longleftarrow \boldsymbol{\lambda}_T$$

#3 Maximum Condition $\boldsymbol{\gamma}^* = \arg\max_{\boldsymbol{\gamma}} \sum_{n=1}^{|\mathcal{D}|} \gamma_n \left[\sum_{t=0}^{T-1} {\boldsymbol{\lambda}_{t+1}^*}^\top \nabla l(x_n, \boldsymbol{\theta}_t^*) \right]$

How to Solve (Theoretically)?

Use <u>Maximum Condition</u> to solve the final $\boldsymbol{\gamma}$

forward Pass with #1: $\theta_0 \rightarrow \theta_1 \rightarrow \dots \rightarrow \theta_T$

reverse Pass with #2: $\lambda_0 \leftarrow \lambda_1 \leftarrow \dots \leftarrow \lambda_T$

maximum γ with #3: $\lambda_0, \lambda_1, \lambda_T \rightarrow \boldsymbol{\gamma}$

How to Solve (Theoretically)?

- Iteratively Solving $\boldsymbol{\gamma}$ until convergence (Algorithm 1)

while $\boldsymbol{\gamma}$ not converged, do

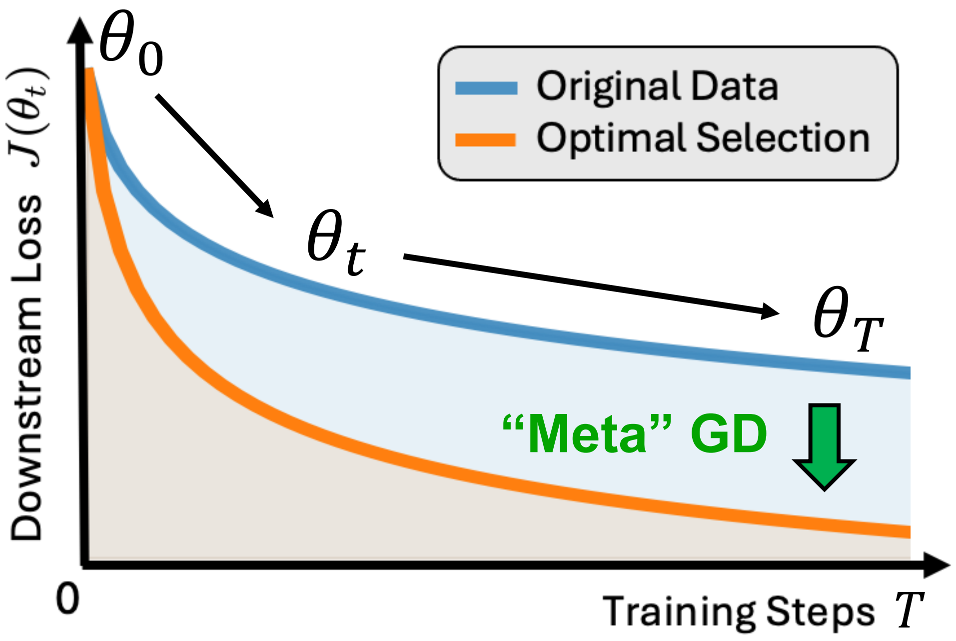
forward Pass with #1: $\boldsymbol{\theta_0} \rightarrow \boldsymbol{\theta_1} \rightarrow \ldots \rightarrow \boldsymbol{\theta_T}$

reverse Pass with #2: $\boldsymbol{\lambda_0} \leftarrow \boldsymbol{\lambda_1} \leftarrow \ldots \leftarrow \boldsymbol{\lambda_T}$

maximum γ with #3: $\boldsymbol{\lambda_0}, \boldsymbol{\lambda_1}, \boldsymbol{\lambda_T} \rightarrow \boldsymbol{\gamma}$

Equivalence to “Meta” Gradient Decent

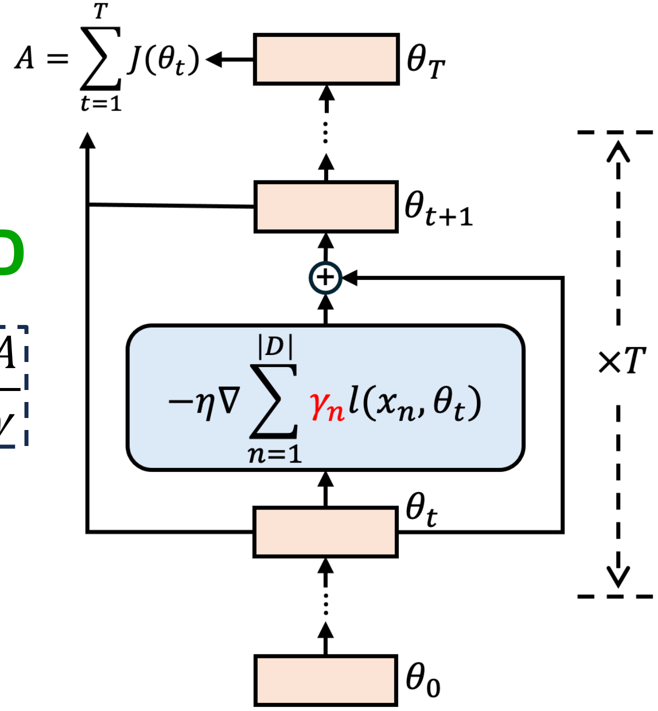
- Optimizing the whole training process with “Meta GD”
- A training process can be viewed as an NN (vertically)



Equivalent

“Meta” GD

$$\gamma \leftarrow \gamma - \alpha \frac{\partial A}{\partial \gamma}$$



Solve γ with "Meta" GD

- Iteratively Solving γ until convergence (Algorithm 1)

```
while γ not converged, do
    forward Pass with #1:
    reverse Pass with #2:
    maximum γ with #3:
```

$$\theta_0 \rightarrow \theta_1 \rightarrow \dots \rightarrow \theta_T$$

$$\lambda_0 \leftarrow \lambda_1 \leftarrow \dots \leftarrow \lambda_T$$

$$\lambda_0, \lambda_1, \lambda_T \rightarrow \gamma$$

Solve γ with "Meta" GD

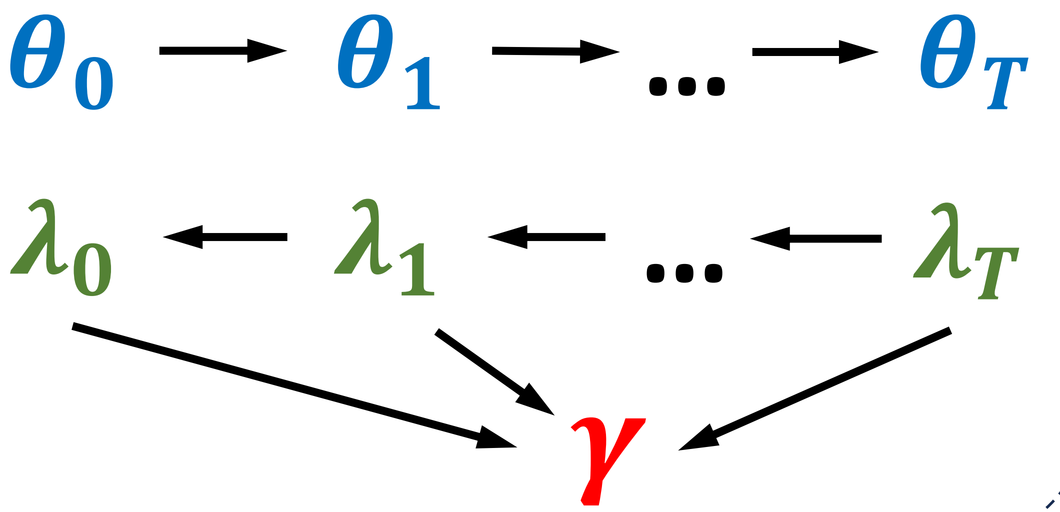
- Iteratively Solving γ until convergence (Algorithm 1)

```
while γ not converged, do
```

Forward pass of "Meta" GD:

Backward pass of "Meta" GD:

Step pass of "Meta" GD:



How to Solve (Theoretically)?

- Iteratively Solving $\boldsymbol{\gamma}$ (Algorithm 1)

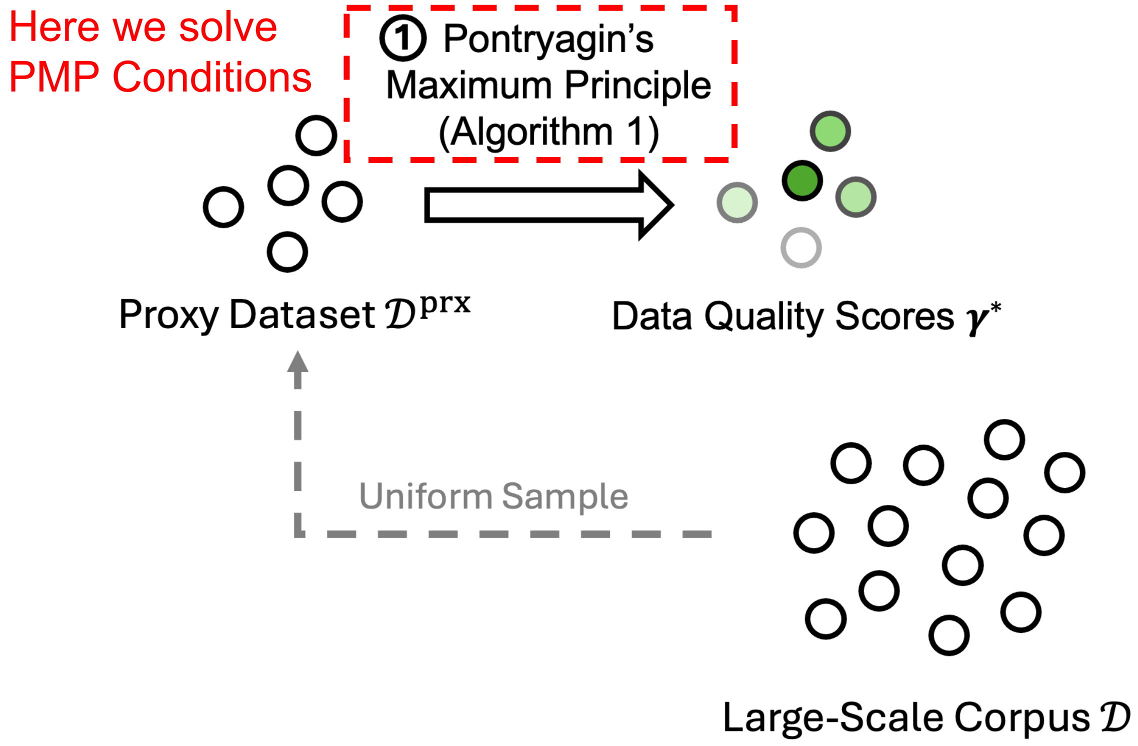
$$\eta \nabla^2 L(\boldsymbol{\theta}_t^*, \gamma^*)\boldsymbol{\lambda}_{t+1}^*$$

```
while γ not converged, do
    forward Pass with #1:  θ_0 → θ_1 → … → θ_T
    reverse Pass with #2:  λ_0 ← λ_1 ← … ← λ_T
    maximum γ with #3:     (λ_0, λ_1, λ_T) → γ
```

Forward and Reverse passes are computationally intensive!

Efficient Implementation: Proxy to Large

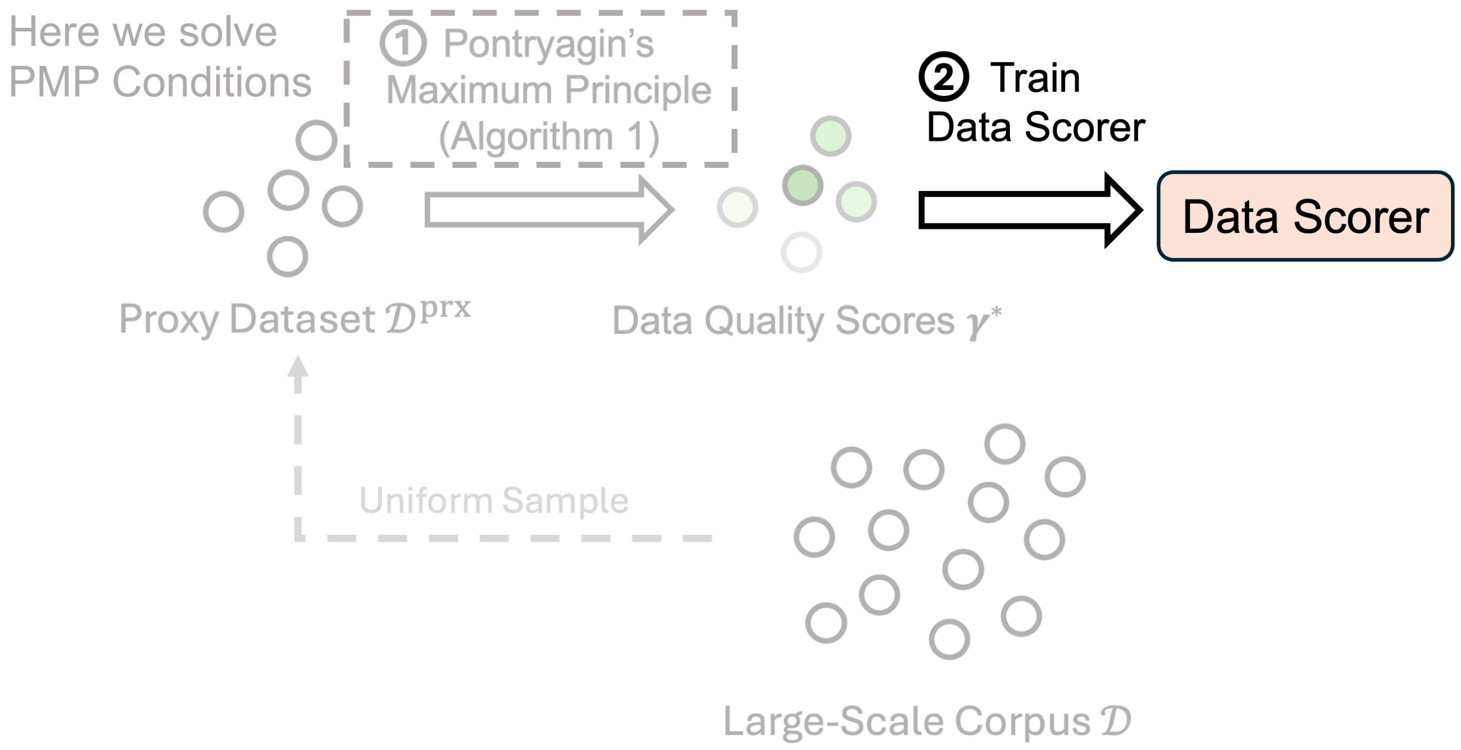
① Solve γ on a small model (e.g., 140M) and data (e.g., 160M tokens)



Efficient Implementation: Proxy to Large

① Solve γ on a small model (e.g., 140M) and data (e.g., 160M tokens)

② Fit γ with a data scorer (e.g., a 140M LM with a regression head)

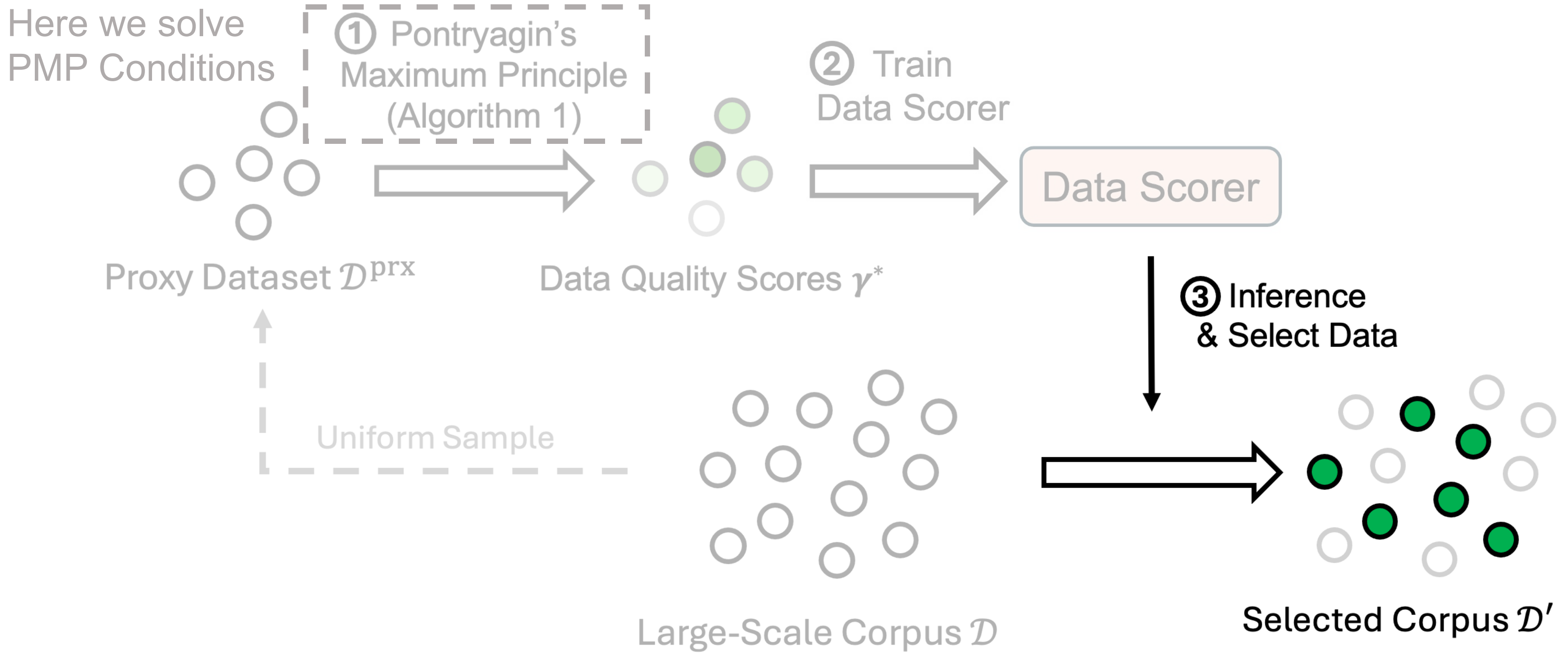


Efficient Implementation: Proxy to Large

① Solve γ on a small model (e.g., 140M) and data (e.g., 160M tokens)

② Fit γ with a data scorer (e.g., a 140M LM with a regression head)

③ Infer γ on the whole dataset (e.g., 100B tokens)

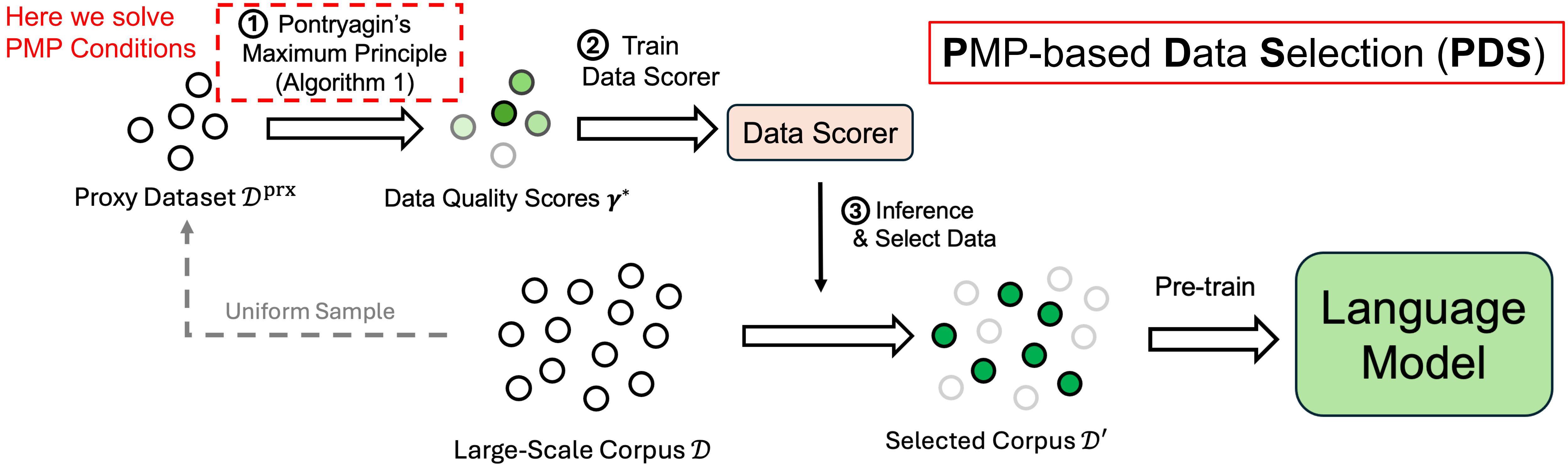


Efficient Implementation: Proxy to Large

① Solve γ on a small model (e.g., 140M) and data (e.g., 160M tokens)

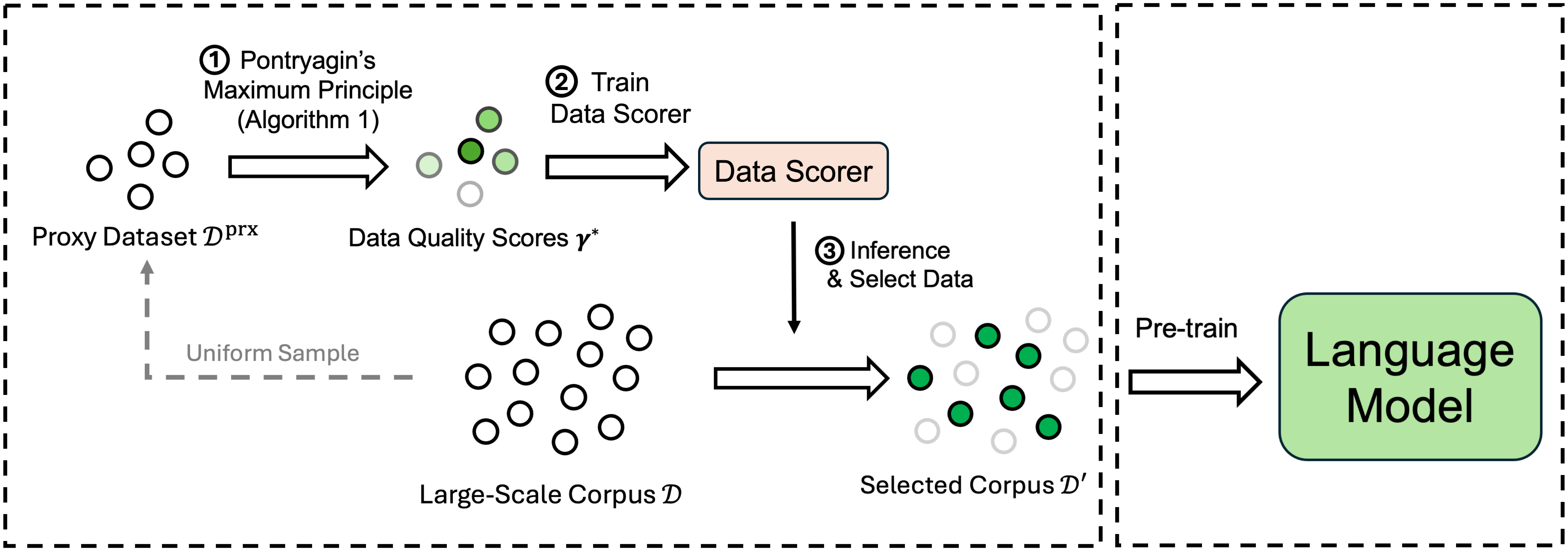
② Fit γ with a data scorer (e.g., a 140M LM with a regression head)

③ Infer γ on the whole dataset (e.g., 100B tokens)



Efficient Implementation: Proxy to Large

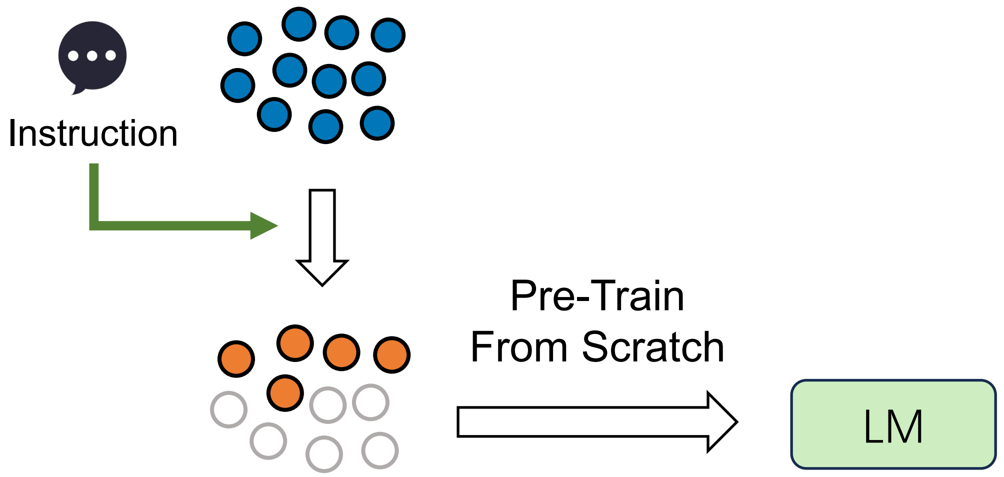
PMP-based Data Selection (PDS) is Efficient



Offline selection (~15h) Pre-Training (~144h)

Experiment Setups

- Training & Evaluation Setups
  - Pre-training LMs from scratch
  - Evaluate zero-shot performance

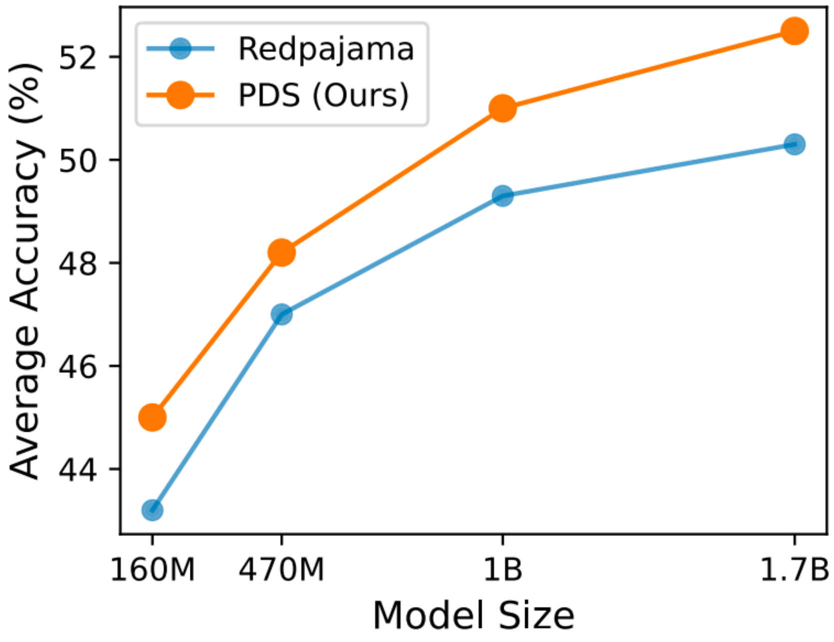


- Data Setups
  - Pre-Training data: CommonCrawl from Redpajama (100B tokens)
  - Downstream loss $J(\theta)$: loss on LIMA (1k high-quality instruction-response pairs)
  - Evaluation: 9 common NLP benchmarks: LAMBADA, Hellaswag, BoolQ, etc.

Performance Improvement

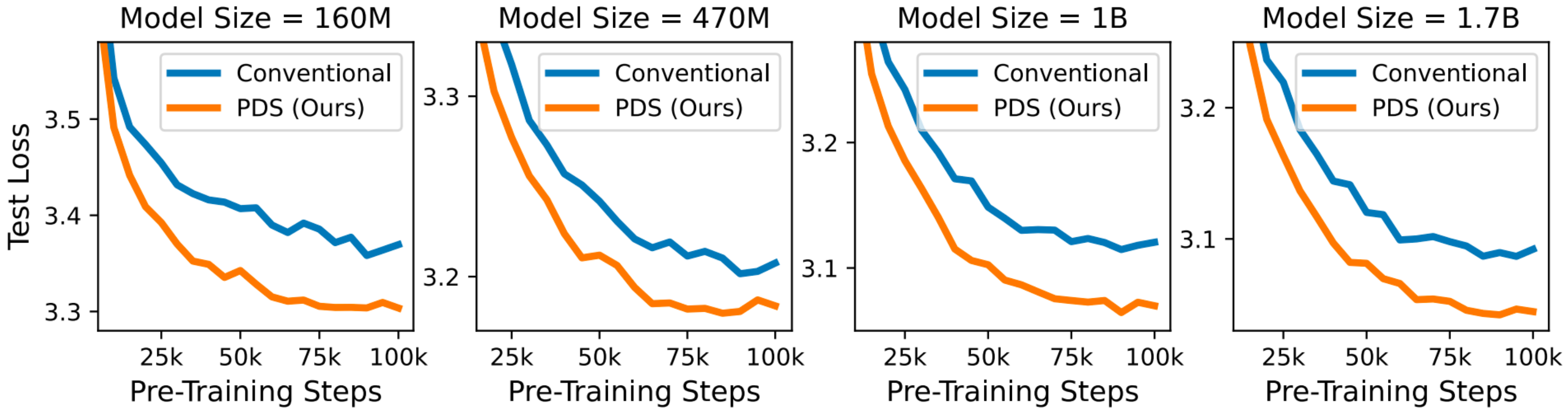
- Select 50B-token corpus from 125B-token corpus.
- Match the total training steps with the baselines (training computation)

	HS	LAMB	Wino.	OBQA	ARC-e	ARC-c	PIQA	SciQ	BoolQ	Avg.
Model Size = 470M										
Conventional	36.7	41.4	52.4	**30.4**	44.8	25.2	61.0	70.6	60.4	47.0
RHO-Loss	36.6	42.4	53.0	29.4	43.7	25.2	60.4	72.8	59.8	47.0
DSIR	36.4	42.6	51.7	29.8	46.0	24.7	61.0	72.0	55.8	46.7
IF-Score	36.6	41.8	**53.4**	29.6	44.7	25.1	60.8	68.8	58.7	46.6
PDS	**37.9**	**44.6**	52.3	29.8	**46.5**	**25.8**	**61.8**	**73.8**	**61.4**	**48.2**
Model Size = 1B										
Conventional	39.9	47.6	52.4	30.6	49.3	26.4	63.1	73.7	60.9	49.3
RHO-Loss	39.8	47.0	53.0	30.8	48.0	26.4	62.9	71.1	**61.0**	48.9
DSIR	40.8	47.8	53.0	31.2	49.8	26.8	62.7	76.6	58.0	49.6
IF-Score	39.4	47.0	52.6	28.6	49.4	26.4	63.5	74.0	60.5	49.0
PDS	**42.1**	**48.8**	**54.0**	**33.4**	**51.3**	**28.0**	**64.1**	**78.5**	58.7	**51.0**



Extrapolation to 400B models on 15T tokens

- Fitting the model loss curves with the Scaling Law



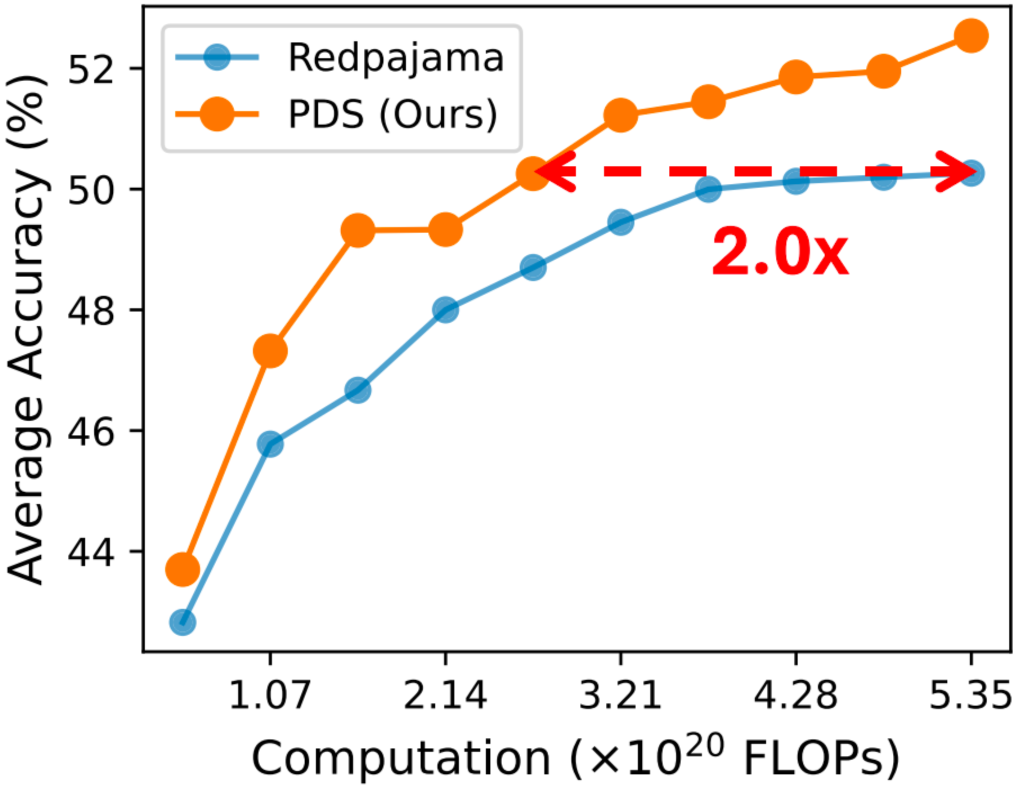
- Extrapolating to 400B models on 10T tokens

$$L(N, D) = E + \frac{A}{N^{\alpha}} + \frac{B}{D^{\beta}}$$

	N	D	Conventional	PDS
GPT-3 [12]	175B	300B	2.882	**2.872**
Llama [72]	6.7B	1.0T	2.942	**2.896**
Llama 2 [73]	70B	2.0T	2.877	**2.855**
Llama 3.1 [21]	405B	15T	2.851	**2.838**

Computation Saving

- 2.0x acceleration on 1.7B models

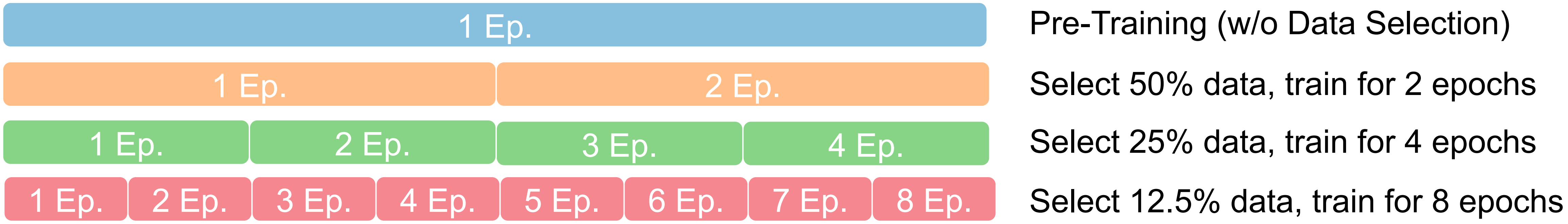


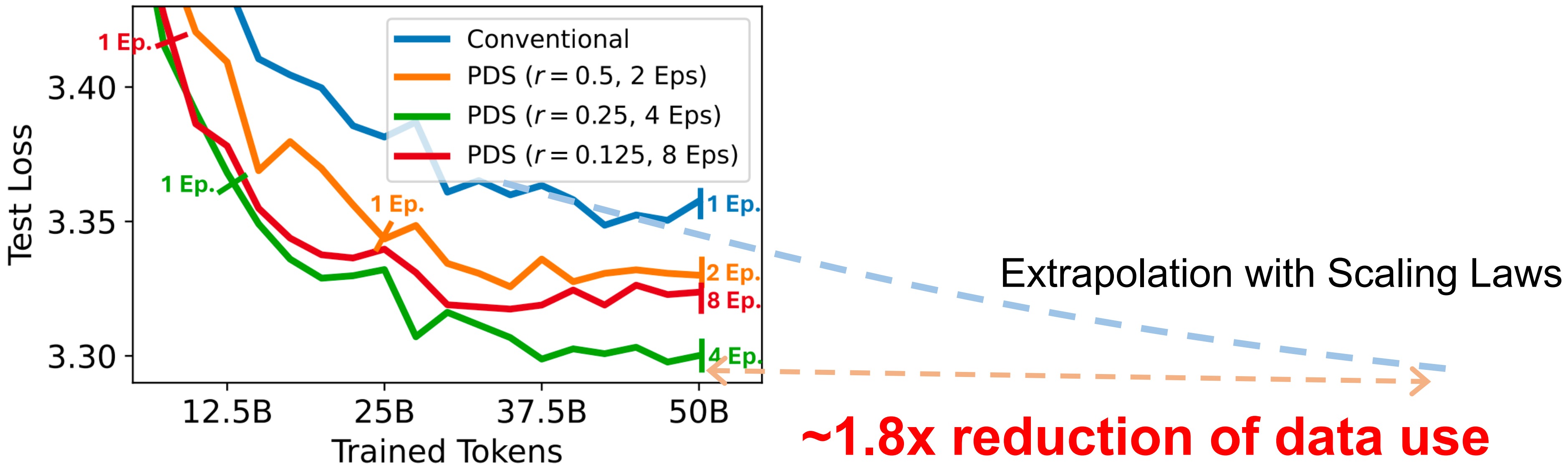
- PDS is efficient and offline
  - Select data once for all models

		FLOPs ($\times 10^{20}$)	Actual Time
PDS	Proxy γ-solver	0.49	15.2 Hours
	Data Scorer	0.063	1.50 Hours
	Data Selection	0.0	10.2 Minutes
Pre-Training		5.1	144 Hours

Data Utilization Improvement

- Performance improvement with limit data (50B tokens)

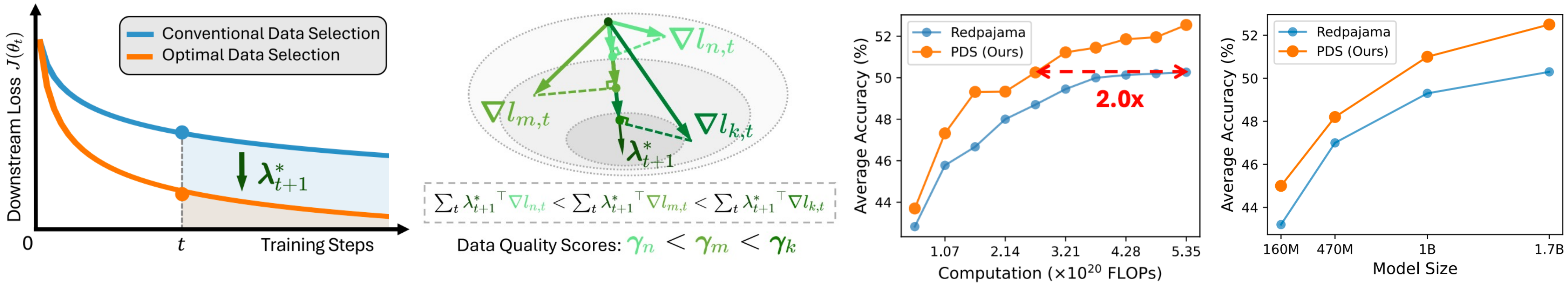




~1.8x reduction of data use

Conclusion

- A novel perspective for Data selection: Optimal Control problem



- Good theoretical guarantees ✅
- Efficient Implementation ✅
- Sound empirical results ✅

A **rigorous, theory-driven alternative** to the ad-hoc practices that currently dominate LM pre-training

Thanks!

Paper: https://arxiv.org/abs/2410.07064

GitHub: https://github.com/microsoft/LMOps/tree/main/data_selection

HuggingFace: https://huggingface.co/Data-Selection

Paper:

Code:

HF: