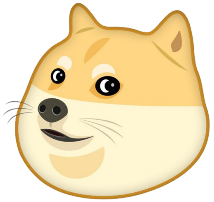
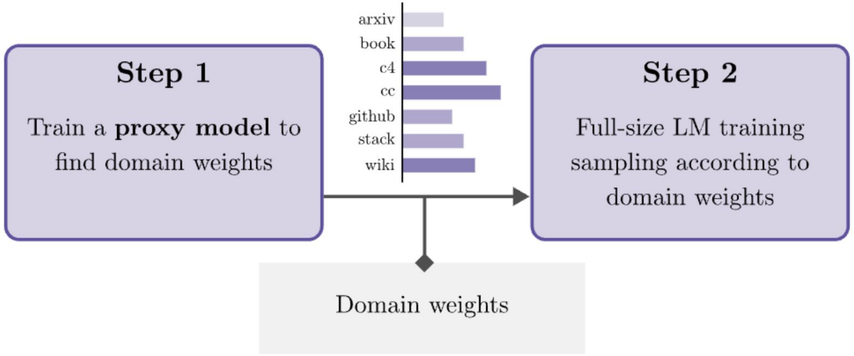
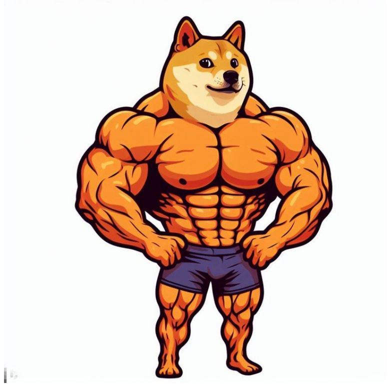
DoGE: Domain Reweighting with Generalization Estimation

Simin Fan, Matteo Pagliardini, Martin Jaggi

ICML 2024







Arxiv: https://arxiv.org/abs/2310.15393

Motivation: Data Selection with Scalability

Redpajama-v1

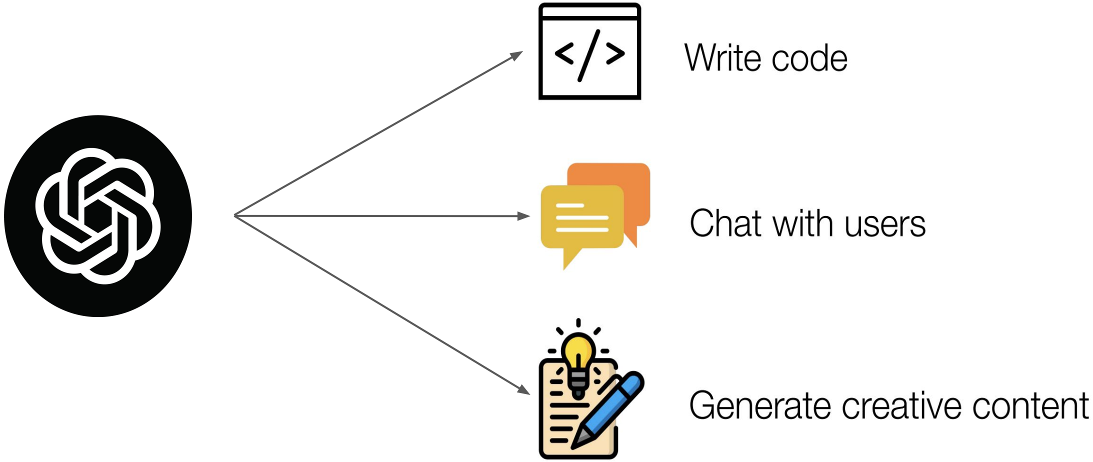
~ 1 Trillion 😕

Redpajama-v2

~ 30 Trillion ‼️

How can we select the best training tokens without large computational overhead?

Motivation: Pretrain for Better Generalization



How to select the most beneficial training tokens for better generalization abilities?

Data Selection for LLM Pretraining

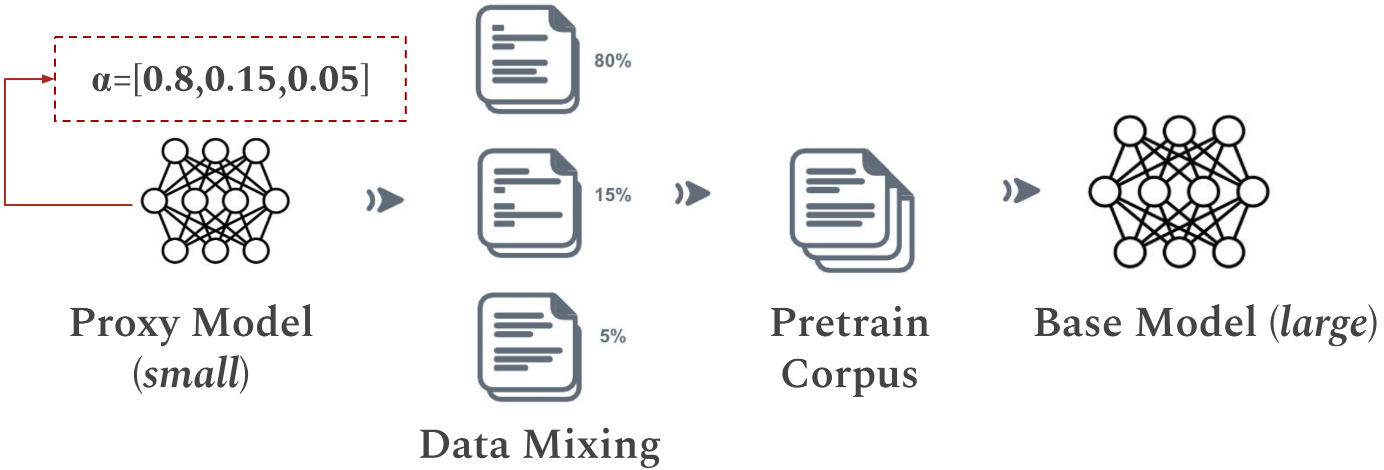
Method		Scalability	In-domain Generalization	Out-of-Domain Generalization
Data-point Assessment	Quality Classifier	❌	❌	❌
	Influence Function	❌	✅	✅
Domain Reweighting	DoReMi [1]	✅	✅	❌
	DoGE [2]	✅	✅	✅

[1] DoReMi: Optimizing Data Mixtures Speeds Up Language Model Pretraining.
[2] DOGE : Domain Reweighting with Generalization Estimation.

Domain Reweighting with Weak-to-Strong Generalization

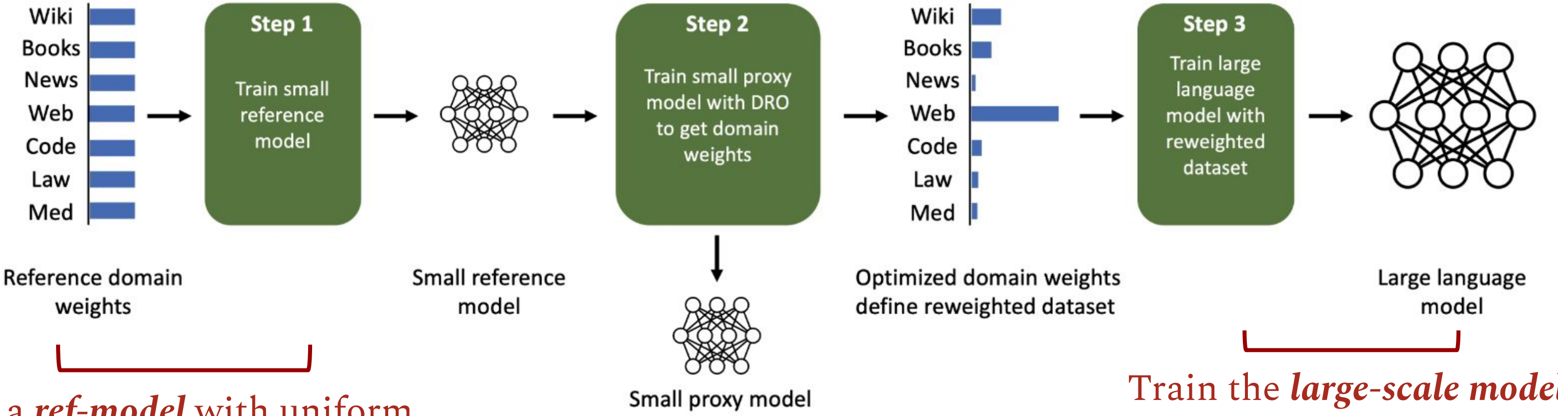
Consider: proxy-model $\boldsymbol{\theta}$; base-model $\boldsymbol{M}$; k training domains $\{\boldsymbol{D1}, \ldots, \boldsymbol{Dk}\}$;

- Optimize domain weights $\alpha \in \Delta^{k}$ according to proxy-model's preference;
- Apply α to train the larger model $\boldsymbol{M}$.



Previous Work: DoReMi

Group-DRO with *Excess-Loss*



Train a ***ref-model*** with uniform domain weights. (*well-trained*)

Train the ***large-scale model*** with updated domain weights.

Train another ***proxy-model*** with group-DRO.
→ **update domain weights at each step.**

Previous Work: DoReMi

Group-DRO with *Excess-Loss*

- **Consider**: k domain datasets { ***D1, …, Dk***};

  proxy-model $\boldsymbol{\theta}$; well-trained ***reference-model***;
- **Output**: optimal domain weights $\alpha \in \Delta^k$.

$$\min_{\theta} \max_{\alpha \in \Delta^k} L(\theta, \alpha) = \min_{\theta} \max_{\alpha \in \Delta^k} \sum_{i=1}^{k} \alpha_i \cdot \left[\frac{1}{\sum_{x \in D_i} |x|} \sum_{x \in D_i} \max\{\ell_\theta(x) - \ell_{\text{ref}}(x), 0\} \right]$$

Down-weigh *Redundant* and *Noisy* domains

Redundant: low l_θ, low l_{ref} ⇒ *Low* excess loss (l_θ - l_{ref}) ⇒ ***Down-weighted***;
Noisy: high l_θ, high l_{ref} ⇒ *Low* excess loss (l_θ - l_{ref}) ⇒ ***Down-weighted***;
Learnable: high l_θ, low l_{ref} ⇒ *High* excess loss (l_θ - l_{ref}) ⇒ ***Up-weighted***.

DoGE 🐶 v.s. DoReMi 🎶

DoReMi (*Pitfalls*)

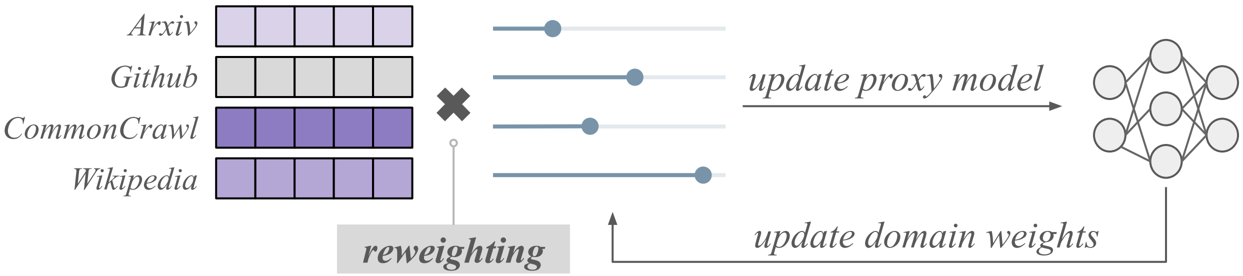
- **Not efficient enough**:
  Two proxy models with two-times memory & three-times computation costs;
- **Lack of Robustness**:
  Highly dependent on the capacity of small auxiliary models;
- **Lack of flexibility:**
  Not capable if the target domain is out of the pretraining corpus.

DoGE

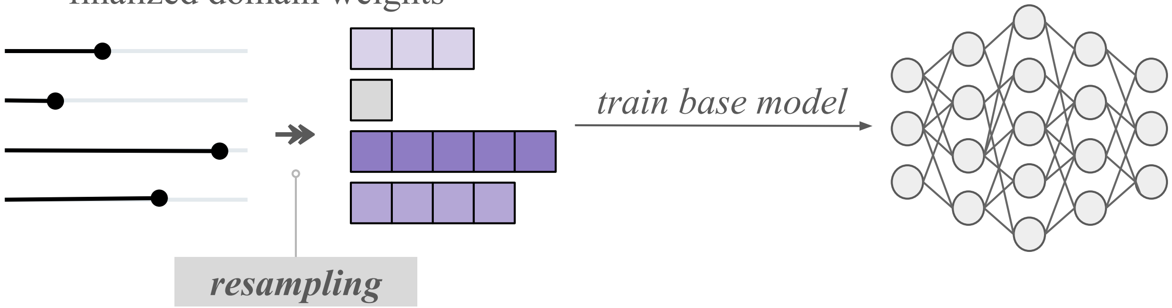
- Formulate domain reweighting as a bilevel-optimization problem.
- Propose an first-order reweighting algorithm with an explicit objective of generalizing to any sets of target;
- Empirical improvement on both
  i) Universal generalization and
  ii) Out-of-domain generalization
  scenarios.

DoGE 🐶: Overview

Step 1 train a **small-scale proxy model** to find optimal domain weights



Step 2 train **large-scale base LLM** with resampled pretrain corpus according to the finalized domain weights



DoGE 🐶: Algorithm

Setup

- **Consider**: proxy-model θ; k training domains $D^{train}=\{D1, \ldots, Dk\}$;

  Set of target domains $D^{tgt}=\{D'1, \ldots, D'n\}$.
- **Output**: optimal domain weights $\alpha \in \Delta^k$.

➔ **Universal Generalization**:

target domains are the entire training domains ($D^{tgt} = D^{train}$)

➔ **Out-of-Domain Generalization**:

target domain is an OoD distribution ($D^{tgt} = D^{ood} \notin D^{train}$) .

DoGE 🐶: Algorithm

Universal Generalization Derivation

➔ **Optimal Domain-weights Search as a Bilevel Problem**

$$\underbrace{\boldsymbol{\alpha} \in \arg\min_{\boldsymbol{\alpha}\in\Delta^k} \sum_{i\in[k]} l_i(\boldsymbol{\theta}^\star(\boldsymbol{\alpha}))} \qquad s.t.\ \underbrace{\boldsymbol{\theta}^\star(\boldsymbol{\alpha}) \in \arg\min_{\boldsymbol{\theta}} \sum_{i\in[k]} \alpha_i l_i(\boldsymbol{\theta})}$$

Outer-loop: update domain weights according to the updated model

Inner-loop: update the proxy model according to current domain weights α

DoGE 🐶: Algorithm

Universal Generalization Derivation

- **Inner-loop: Model Update** at step t :

$$\boldsymbol{\theta}^{(t+1)} \triangleq \boldsymbol{\theta}^{(t)} - \eta^{(t)} \sum_{i\in[k]} \alpha_i^{(t)} \nabla l_i(\boldsymbol{\theta}^{(t)}), \quad \text{with } \boldsymbol{\alpha}^{(t)} \in \Delta^k, \tag{1}$$

- **Outer-loop: We greedily minimize the average loss across all training domains at the next step** *(t+1)*:

$$\boldsymbol{\alpha}_\star^{(t)} = \arg\min_{\boldsymbol{\alpha}\in\Delta^k} \bar{l}(\boldsymbol{\theta}^{(t+1)}) = \arg\min_{\boldsymbol{\alpha}\in\Delta^k} \sum_{i\in[k]} [l_i(\boldsymbol{\theta}^{(t+1)}) - l_i(\boldsymbol{\theta}^{(t)})]$$

First-order Approx. $\approx \arg\min_{\boldsymbol{\alpha}\in\Delta^k} \sum_{i\in[k]} \langle \nabla l_i(\boldsymbol{\theta}^{(t)}), \boldsymbol{\theta}^{(t+1)} - \boldsymbol{\theta}^{(t)} \rangle + o(\|\Delta\boldsymbol{\theta}^{(t)}\|)$

Plug in (1). $= \arg\min_{\boldsymbol{\alpha}\in\Delta^k} \sum_{i\in[k]} \langle \nabla l_i(\boldsymbol{\theta}^{(t)}), -\eta^{(t)} \sum_{j\in[k]} \alpha_j \nabla l_j(\boldsymbol{\theta}^{(t)}) \rangle + o(\|\Delta\boldsymbol{\theta}^{(t)}\|) \quad (2)$

DoGE 🐶: Algorithm

Universal Generalization Derivation

- Define ***Generalization Estimation function*** *on j^{th} domain as*:

$$W_j^{(t)} \triangleq \langle \nabla l_j(\boldsymbol{\theta}^{(t)}), \sum_{i\in[k]} \nabla l_i(\boldsymbol{\theta}^{(t)}) \rangle$$

< Gradient Alignment >

- The outer-problem (2) can be rewritten as:

$$\boldsymbol{\alpha}_\star^{(t)} = \arg\min_{\boldsymbol{\alpha}\in\Delta^k} -\eta^{(t)} \boldsymbol{\alpha}^\top \mathcal{W}^{(t)} + o(\|\Delta\boldsymbol{\theta}^{(t)}\|) \text{, with } \mathcal{W}^{(t)} = [W_1^{(t)}, \ldots, W_k^{(t)}] \in \mathbb{R}^k$$

DoGE 🐶: Algorithm

Universal Generalization Derivation

- Define ***Generalization Estimation function*** *on j^{th} domain as:*

$$W_j^{(t)} \triangleq \langle \nabla l_j(\boldsymbol{\theta}^{(t)}), \sum_{i\in[k]} \nabla l_i(\boldsymbol{\theta}^{(t)}) \rangle$$

< Gradient Alignment >

- The outer-problem (2) can be rewritten as:

$$\boldsymbol{\alpha}_\star^{(t)} = \arg\min_{\boldsymbol{\alpha}\in\Delta^k} -\eta^{(t)}\boldsymbol{\alpha}^\top \mathcal{W}^{(t)} + o(\|\Delta\boldsymbol{\theta}^{(t)}\|) \text{, with } \mathcal{W}^{(t)} = [W_1^{(t)}, \ldots, W_k^{(t)}] \in \mathbb{R}^k$$

- For better stability with stochastic noises, we estimate the high-order error term with Bregman Divergence $D_\Psi(\boldsymbol{\alpha}\|\boldsymbol{\alpha}^{(t-1)})$ with $\Psi(\boldsymbol{\alpha}) = \sum_i \alpha_i \log(\alpha_i)$

$$\boldsymbol{\alpha}^{(t)} = \arg\min_{\boldsymbol{\alpha}\in\Delta^k} -\eta^{(t)}\boldsymbol{\alpha}^\top \mathcal{W}^{(t)} + \boxed{\mu D_\Psi(\boldsymbol{\alpha}\|\boldsymbol{\alpha}^{(t-1)})}$$

DoGE 🐶: Algorithm

Universal Generalization Derivation

➔ Define ***Generalization Estimation function*** *on* j^{th} *domain as:*

$$W_j^{(t)} \triangleq \langle \nabla l_j(\boldsymbol{\theta}^{(t)}), \sum_{i\in[k]} \nabla l_i(\boldsymbol{\theta}^{(t)}) \rangle$$

< Gradient Alignment >

➔ The outer-problem (2) can be rewritten as:

$$\boldsymbol{\alpha}_\star^{(t)} = \arg\min_{\boldsymbol{\alpha}\in\Delta^k} -\eta^{(t)}\boldsymbol{\alpha}^\top \mathcal{W}^{(t)} + o(\|\Delta\boldsymbol{\theta}^{(t)}\|) \text{ , with } \mathcal{W}^{(t)} = [W_1^{(t)}, \ldots, W_k^{(t)}] \in \mathbb{R}^k$$

➔ For better stability with stochastic noises, we estimate the high-order error term with Bregman Divergence $D_\Psi(\boldsymbol{\alpha}\|\boldsymbol{\alpha}^{(t-1)})$ with $\Psi(\boldsymbol{\alpha}) = \sum_i \alpha_i \log(\alpha_i)$

$$\boldsymbol{\alpha}^{(t)} = \arg\min_{\boldsymbol{\alpha}\in\Delta^k} -\eta^{(t)}\boldsymbol{\alpha}^\top \mathcal{W}^{(t)} + \mu D_\Psi(\boldsymbol{\alpha}\|\boldsymbol{\alpha}^{(t-1)}) \dashrightarrow \boxed{\hat{\boldsymbol{\alpha}}^{(t)} = \boldsymbol{\alpha}^{(t-1)} \odot \exp\left(\frac{\eta^{(t)}\mathcal{W}^{(t)}}{\mu}\right)}$$

DoGE : Algorithm

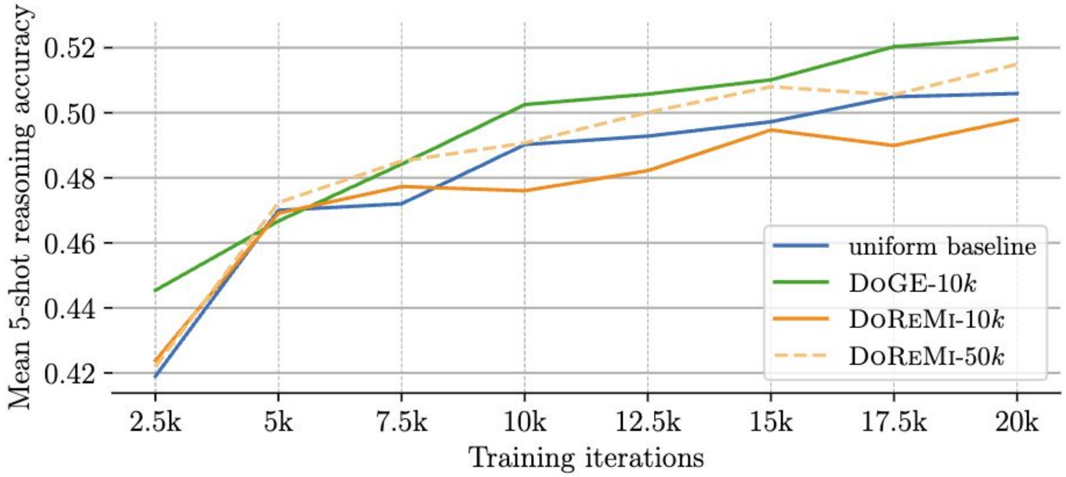
Out-of-Domain Generalization

- For OOD Generalization, we only need to modify the definition of generalization estimation function with the stochastic gradient $\nabla l_{ood}(\boldsymbol{\theta}^{(t)})$ on $\boldsymbol{D}^{ood}$:
  $W_j^{(t)} \triangleq \langle \nabla l_j(\boldsymbol{\theta}^{(t)}), \nabla l_{ood}(\boldsymbol{\theta}^{(t)}) \rangle$, so that we have the generalization estimation scores across all k training domains as: $\mathcal{W}_{ood}^{(t)} = [W_1^{(t)}, \ldots, W_k^{(t)}]$.
- Similarly, we have the domain weight update rule for OOD generalization:

$$\boldsymbol{\alpha}^{(t)} = \frac{\hat{\boldsymbol{\alpha}}^{(t)}}{\sum_{i \in [k]} \hat{\alpha}_i^{(t)}}, \quad \text{with } \hat{\boldsymbol{\alpha}}^{(t)} \leftarrow \boldsymbol{\alpha}^{(t-1)} \odot \exp(\eta^{(t)} \mathcal{W}_{ood}^{(t)} / \mu)$$

Results: Universal Generalization

Average Few-shot Reasoning Accuracy (82M → 684M)



***82M → 684M**: *experiment with 82M proxy-model and 684M base-model.*
DoReMi-50k: *DoReMi trained with 50k steps (5x tokens; ~10x training time).*

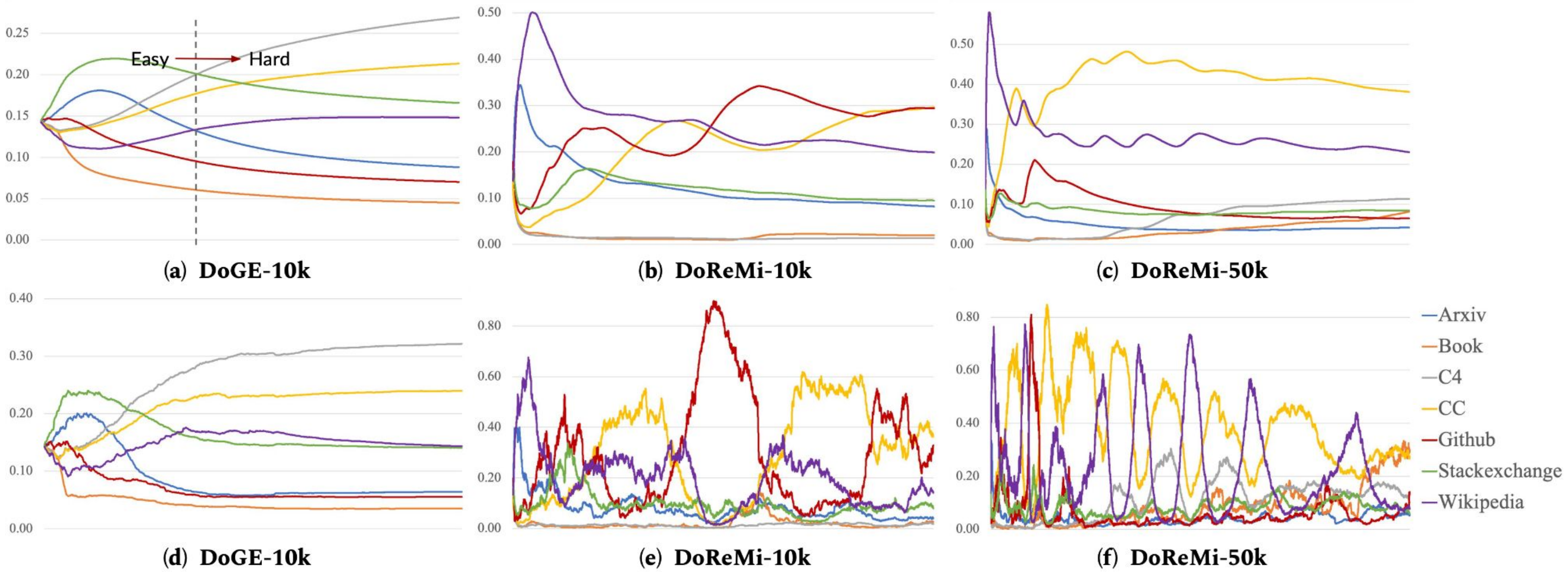
Results: Universal Generalization

Average Perplexity across Domains (82M → 684M)

Domain	Uniform baseline	DoReMi-10k	DoGE-10k	DoReMi-50k
Arxiv	**8.105**	8.698	8.207	9.378
Book	44.990	50.594	**44.574**	42.557
C4	49.066	56.116	**42.558**	41.388
CommonCrawl	45.903	46.459	**40.432**	41.067
Github	3.944	**3.739**	4.107	4.301
Stackexchange	8.628	9.022	**8.332**	9.235
Wikipedia	12.047	**11.380**	11.443	10.519
Average	16.526	17.172	**15.806**	16.124
Worst-case	49.066	56.116	**44.574**	42.557
# domains outperform Baseline	/	2	5	4

Results: Universal Generalization

Domain Weights Evolutions (82M proxy)



Results: OOD Generalization on SlimPajama

Perplexity on Target Domain (82M → 124M)

	Baseline (w/o target)	DoGE	Baseline (w/o target)+fine-tuning	DoGE+fine-tuning	Oracle (with target)
Arxiv	18.92±0.14	**16.70**±0.08	10.47±0.01	**10.20**±0.01	9.78±0.01
Book	82.57±0.05	**63.89**±0.18	65.73±0.06	**56.94**±0.24	66.43±0.19
C4	89.56±0.38	**63.96**±0.11	71.24±0.09	**56.91**±0.17	70.69±0.14
CommonCrawl	81.65±0.47	**57.77**±0.56	65.75±0.01	**51.173**±0.04	67.06±0.15
Github	6.675±0.00	**5.091**±0.03	4.99±0.01	**4.26**±0.01	4.97±0.01
StackExchange	16.941±0.02	**14.77**±0.01	11.24±0.004	**10.98**±0.002	11.26±0.03
Wikipedia	58.04±0.32	**53.87**±0.35	18.38±0.02	**17.71**±0.05	17.61±0.02

*****Baseline** (**w/o target**): uniform sampling without target domain.*

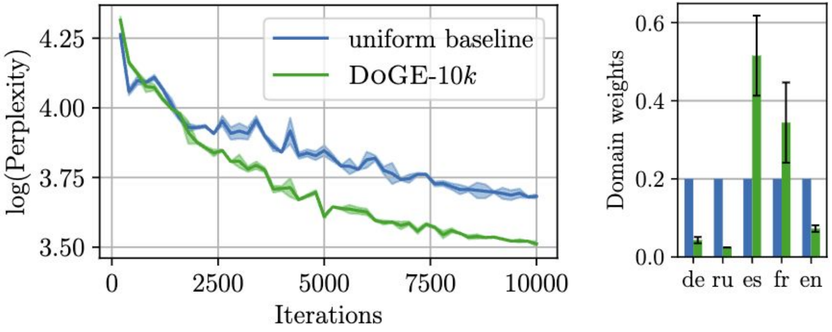
*****Oracle**: uniform sampling with target domain.*

*+**fine-tuning**: finetune on the **small set of validation set** from the target domain, which is used by DoGE to compute domain weights.*

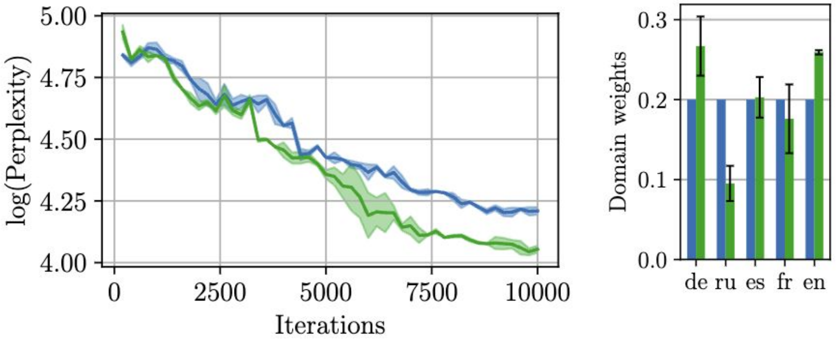
Results: OOD Generalization on Wiki40b

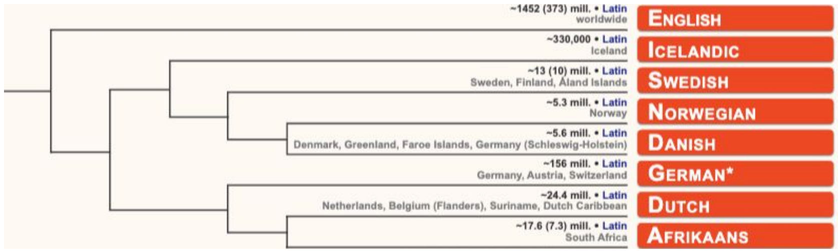
Perplexity on Target Language (82M → 124M)

(a) Catalan



(b) Dutch





Results: Robustness of Domain Weights

Domain Weights from Various Scale of Proxy Model

Domain	DoGE (60M)	DoGE(82M)	DoGE(124M)	DoReMi (60M)	DoReMi (82M)	DoReMi (124M)
Arxiv	0.0997	0.0880	0.0890	0.0781	0.0424	0.0434
Book	0.0467	0.0450	0.0456	0.0830	0.0819	0.0546
C4	0.2455	0.2693	0.2789	0.1343	0.1141	0.1127
CommonCrawl	0.2004	0.2135	0.1968	0.2683	0.3811	0.3781
Github	0.0767	0.0703	0.0714	0.1055	0.0654	0.0753
Stackexchange	0.1968	0.1658	0.1703	0.1157	0.0847	0.0919
Wikipedia	0.1342	0.1482	0.1480	0.2150	0.2307	0.2440
MAE from 82M proxy	1.45%	/	0.48%	3.66%	/	0.91%
Computation Time (hours)[1]	4.5	6.0	10.5	20.5	39.0	51.5

Discussion: Make DoGE stronger? 🪄

- **Granularity of Domains:**

  do more ***fine-grained clustering*** on texts/reweighting on sequence level;

- **Online Reweighting:** get rid of the proxy model

  Challenges: overfitting, computation overheads, etc.

- **Scaling law of *domain preference* [3,4]:**

  How to *smartly predict* larger model's preference from small proxies, instead of *copying*?

[3] RegMix: Data Mixture as Regression for Language Model Pre-training
[4] AutoScale: Automatic Prediction of Compute-optimal Data Composition for Training LLMs.

Thanks for Listening :-)

Olivia Simin Fan | simin.fan@epfl.ch

Sept. 20, 2024