Rethinking Data Mixing from the Perspective of Large Language Models

Yuanjian Xu[1§], **Tianze Sun**[§*], **Changwei Xu**[2], **XinLong Zhao**[†], **Jianing Hao**[1], **Ran Chen**[2], **Yang Liu**[†], **Ruijie Xu**[2], **Stephen Chen**[2], **Guang Zhang**[1,‡]
[1]Hong Kong University of Science and Technology (Guangzhou) [2]OpenCSG
{yxu085@connect, guangzhang@}hkust-gz.edu.cn

Abstract

Data mixing strategy is essential for large language model (LLM) training. Empirical evidence shows that inappropriate strategies can significantly reduce generalization. Although recent methods have improved empirical performance, several fundamental questions remain open: what constitutes a domain, whether human and model perceptions of domains are aligned, and how domain weighting influences generalization. We address these questions by establishing formal connections between gradient dynamics and domain distributions, offering a theoretical framework that clarifies the role of domains in training dynamics. Building on this analysis, we introduce DoGraph, a reweighting framework that formulates data scheduling as a graph-constrained optimization problem. Empirical results across various model architectures and scales demonstrate that DoGraph consistently delivers competitive performance. Code and data are publicly available at https://github.com/xuyj233/Dograph.

1 Introduction

Training data fundamentally determines the capability of large language models (LLMs) (Xu et al., 2023; Wettig et al., 2024; Albalak et al.). However, domain distributions are imbalanced due to unequal data availability: web-scale corpora are abundant, whereas specialized domains remain scarce (Gao et al., 2021). This raises a key question—can we design a principled sampling strategy to mitigate such imbalance? Exhaustively searching over all possible sampling policies is infeasible, as LLM training is prohibitively expensive. To make progress, we must first answer: what does a "domain" truly mean for a LLM, and are human and model perceptions of domains aligned (Sun et al., 2025)?

Prior data mixing studies have predominantly relied on domain definitions derived from human intuition. Existing approaches can be broadly categorized into two lines of work. The first derives heuristics from small- or medium-scale models and then scales them to LLMs (Liu et al., 2024; Ye et al., 2024; Fan et al., 2023; Xie et al., 2023); however, empirical evidence shows that scaling laws and domain sensitivities observed in small models do not transfer reliably to larger ones (Kang et al., 2024). The second directly performs data reweighting or optimization on LLMs, either at the sample or domain level (Sun et al., 2025; Sow et al., 2025), but often incurs prohibitive computational costs or relies on unrealistic assumptions.

In this work, we argue that the optimization of LLMs continuously reshapes their domain perception, creating a mismatch between human-defined and model-internal representations (Bengio et al., 2013). Figure 1 visualizes this evolution: at initialization, samples from domains such as C4, Wikipedia, Book, and ArXiv form well-separated clusters, reflecting strong domain-specific biases. As training progresses, these clusters gradually merge into an approximately isotropic distribution, indicating that the model internalizes more domain-invariant linguistic structures (Power et al., 2022; Gao et al., 2019).

This evolving misalignment biases existing data mixing methods. To address it, we formally link domain distributions with gradient dynamics, showing how model-defined domains emerge during optimization (Koh and Liang, 2017; Fort et al., 2019). Building on this foundation, we propose *DoGraph*, which formulates domain scheduling as a graph-constrained reweighting problem. DoGraph models the model-perceived domains as graph nodes and learns their weights through optimization. Our main contributions are summarized as follows: 1) We theoretically establish a connection between domain distribution and gradient dynamics, and empirically validate the dynamic correction of domain representations during LLM training. 2) We

* Harbin Institute of Technology † China Mining Group ‡ Corresponding author. § These authors contributed equally.

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 337–349

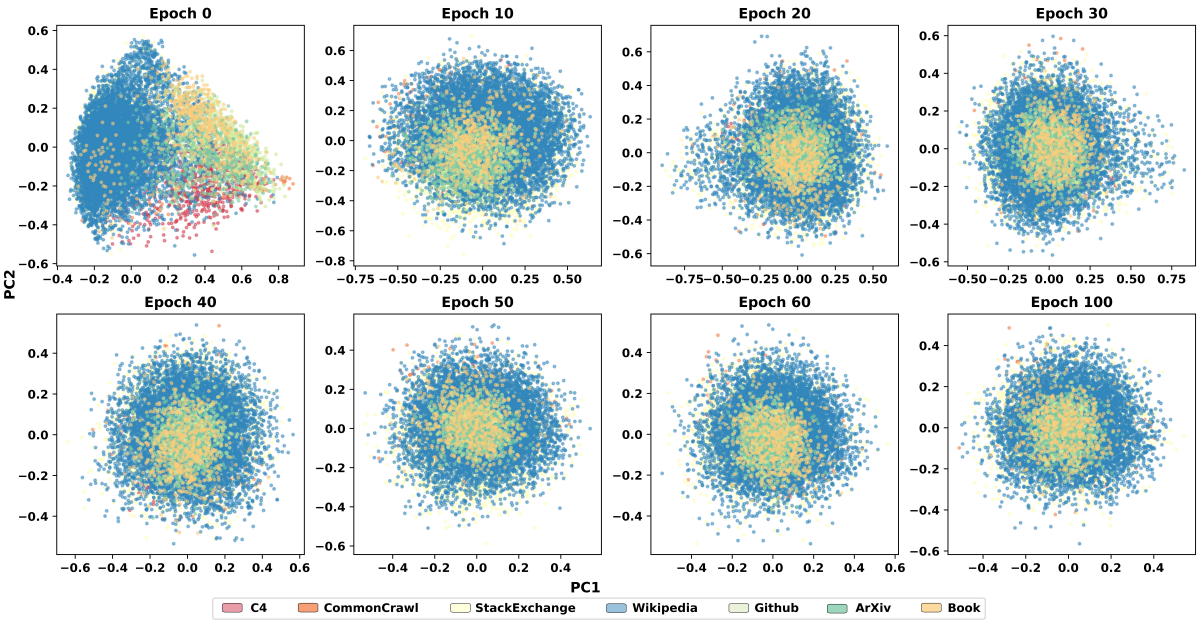

Figure 1: PCA projections of gradient directions at different training epochs. Colors denote data domains (C4, Wikipedia, ArXiv, Book, etc.). Initially, gradients form distinct clusters, showing strong domain bias. Over time, they overlap, indicating that the model homogenizes its domain perception. Experiments use 20% of SlimPajama trained on GPT2-Mini.

propose DoGraph, a graph-constrained reweighting framework that formalizes domain scheduling as an optimization problem. DoGraph is strongly grounded in theoretical principles. 3) We conduct extensive experiments across diverse benchmarks, demonstrating consistent improvements in both performance and domain balance, which validate the competitiveness of our approach.

2 Methods

In this section, we begin by redefining domains from a learning-theoretic perspective. We then establish their connection to gradients, showing that distributional differences are reflected in gradient geometry in Section 2.1. Finally, we build on this insight to propose the *DoGraph*.

2.1 Rethinking the Definition of Domain

In NLP, the notion of domain has often been unclearly defined, particularly in the training corpora of LLMs, where such boundaries are increasingly blurred. Before developing concrete strategies for domain weighting, it is essential to first clarify what we mean by a domain. We argue that a domain should be defined from the model's perspective, namely as the distribution of inputs it perceives, rather than from a human perspective.

Definition 2.1. *Let $\mathcal{V}$ be a finite vocabulary and $\mathcal{X} = \mathcal{V}^*$ the space of all token sequences. A* domain *is a probability space $(\mathcal{X}, \mathcal{F}, P_X)$, where $\mathcal{F}$ is the σ-algebra on $\mathcal{X}$ and P_X a probability measure. Two domains $\mathcal{D}_1 = (\mathcal{X}, \mathcal{F}, P_1)$ and $\mathcal{D}_2 = (\mathcal{X}, \mathcal{F}, P_2)$ are distinct iff $P_1 \neq P_2$.*

We now formulate the definition of domain as stated in Definition 2.1. Each $x \in \mathcal{X}$ is a finite token sequence from $\mathcal{V}$, with domains distinguished by the regions of $\mathcal{X}$ where their distributions P_X concentrate. In simple cases, such as distinguishing code from natural language, these regions are relatively easy to separate. However, in practice, many domains are much less clear-cut, with boundaries that overlap or gradually shift. Thus, domains differ through probability measures over the same space $\mathcal{V}^*$ rather than through disjoint supports.

Connection between Domain and Gradients A central question is whether domains can be inferred from observable data instead of being imposed a priori, as such assumptions inevitably risk introducing bias. Since each training sample affects learning only via its gradient, the model perceives not raw token frequencies but the geometry of gradient flows. To investigate this, we analyze a simplified self-attention structure in which the Transformer can be linearized, leading to a tractable correspon-

dence between distributions and gradients.

Theorem 2.2. *Under the linearized Transformer setting, for any parameter block $b \in \{V, Q, K, O, W\}$ and two domains P_1, P_2, the difference of expected gradients satisfies $\bar{g}_b(P_1) - \bar{g}_b(P_2) = \int \nabla_{W_b} L(x, y; \theta)\,(P_1 - P_2)(dx, dy)$. Moreover, this difference admits a kernel representation:*

$$\|\bar{g}_b(P_1) - \bar{g}_b(P_2)\|^2 = \mathrm{MMD}^2_{k_b}(P_1, P_2),$$

where $k_b(s, s') = \langle g_b(s), g_b(s') \rangle$ is the gradient-induced kernel.

Theorem 2.2 shows that distributional differences are encoded in the geometry of gradients, implying that domains can be compared through their gradient signatures rather than token-level statistics. From this perspective, a domain is defined by its expected gradient flow, and training can be understood as a continual refinement of the model's perception of domains, with each update adjusting how distributions are represented in gradient space.

2.2 DoGraph

We argue that data weighting should adapt to the model's evolving perception of domains, rather than fixed human-defined boundaries. Building on this idea, we introduce the *DoGraph*, where each domain corresponds to a node in a graph. At every epoch, we collect per-sample gradients and **project them into a low-dimensional subspace via random projection**. Next, we apply K-means clustering in the projected gradient space to obtain model-centric partitions of the training distribution. This partition evolves over training, reflecting the changing geometry of gradients.

Formally, let $g_i \in \mathbb{R}^d$ be the gradient of the i-th sample and $G = [g_1, \ldots, g_n]^\top \in \mathbb{R}^{n \times d}$. We apply a random projection matrix $R \in \mathbb{R}^{d \times k}$ with $R_{pq} \sim \mathcal{N}(0, 1/k)$, yielding $\tilde{g}_i = R^\top g_i, \quad \text{or } \tilde{G} = GR$. By the Johnson–Lindenstrauss lemma,

$$(1-\epsilon)\|g_i - g_j\|_2^2 \le \|\tilde{g}_i - \tilde{g}_j\|_2^2 \le (1+\epsilon)\|g_i - g_j\|_2^2,$$

ensuring that clustering in the projected space preserves the gradient geometry while reducing computational cost and noise. Clustering $\{\tilde{g}_i\}$ into m groups $\{D_1, \ldots, D_m\}$, we compute each domain's mean gradient as $\bar{g}_j = \frac{1}{|D_j|} \sum_{i \in D_j} \tilde{g}_i$. To balance learning, we assign adaptive domain weights $w = (w_1, \ldots, w_m)$ by solving $\min_{w \in \Delta^{m-1}} \mathcal{L}_{\text{opt}}\left(\sum_{j=1}^m w_j \bar{g}_j\right)$, where Δ^{m-1} is the probability simplex.

DoGraph Pipeline At each epoch, per-sample gradients are first extracted and projected into a low-dimensional subspace via random projection, then clustered into domains in the projected space. Domain mean gradients are aggregated through an optimization step that computes the optimal domain weights. The model parameters are updated with the weighted gradient, and the process repeats, allowing both the partition of domains and their relative importance to adapt continuously throughout training. The choice of the optimization objective $\mathcal{L}_{\text{opt}}$ is discussed in the Appendix A.6. In our implementation,$\mathcal{L}_{\text{opt}}$ is instantiated with domain uncertainty weighting. Algorithm 1 summarizes the overall procedure of DoGraph. More implementation details can be found in Appendix A.3.

3 Experiment Results

In this section, we begin by outlining the experimental setup, after which we present the overall performance analysis. We further conduct a perplexity analysis and investigate how model scale influences the observed trends, with detailed results presented in Section 3.3 and Section 3.4. All main results in the paper are reported using the GPT-2 Medium. To isolate the effects of architecture and parameter scale, additional experiments with the LLaMA-1.1B model are deferred to the Appendix A.4. Sensitivity to hyperparameters and the choice of optimizer are analyzed in Appendix A.6 and Appendix A.7.

3.1 Experiments Setup

Our experiments use decoder-only, Transformer-based language models (Vaswani et al., 2017; Radford et al., 2019) at 210M and 300M scales. Models are trained on SlimPajama (Soboleva et al., 2023), spanning seven text domains. We evaluate DoGraph on nine stable benchmarks and compare with representative baselines. Model details, training protocol, and baseline breakdown are in Appendix A.2.

3.2 Results in the Pretraining Stage

As shown in Table 1, DoGraph achieves more balanced learning across domains and delivers consistent gains over all baselines. It yields the largest improvements on reasoning-oriented benchmarks,

Method	Commonsense / Reasoning						RC		LM	Avg
	HellaSwag	PiQA	OBQA	COPA	LogiQA	WinoG	SciQ	ARC-E	Lambada	
SlimPajama										
Uniform	26.1	55.5	11.7	58.0	25.7	49.9	49.0	31.4	11.6	35.4
Dynamic Loss-Based	26.6	56.8	13.8	59.0	**29.8**	50.1	53.3	31.7	13.4	37.2
DoReMi	26.4	55.7	12.2	59.0	27.2	49.9	53.3	32.3	12.7	36.5
DOGE	26.2	55.8	11.5	62.0	27.2	50.4	52.8	31.3	11.6	36.5
RegMix	26.1	55.6	13.2	60.0	23.7	50.0	46.6	31.7	14.0	35.7
Data Mixing Law	26.5	54.5	13.0	62.0	24.4	49.1	45.2	32.0	12.0	35.4
DoGraph (Ours)	**27.3**	**56.9**	**14.8**	**63.0**	26.3	**50.8**	**53.5**	**33.9**	**14.5**	**37.9**

Table 1: Downstream benchmark results (**accuracy %**) on SlimPajama (GPT-2 Medium). Tasks grouped into **Commonsense/Reasoning**, **Reading Comprehension**, and **Language Modeling**. Best results highlighted in bold.

highlighting the advantage of its structured weighting mechanism in capturing logical and commonsense dependencies across domains. Moreover, the performance gains on reading comprehension tasks, which require semantic consistency and information integration, demonstrate that DoGraph's adaptive data scheduling enhances semantic alignment and improves overall generalization.

3.3 Perplexity Analysis

Table 2 shows validation perplexity on SlimPajama under various domain-mixing strategies. Uniform sampling performs moderately but fails to balance domain frequencies. Loss-based weighting and prior methods (DoReMi, DOGE) yield unstable gains, overfitting to high-resource domains and degrading on long-tail data. RegMix and Data Mixing Law worsen this trend, with higher perplexity despite larger models. **DoGraph** achieves the best perplexity , reflecting balanced domain integration and strong generalization.

Method	SlimPajama (Val PPL ↓)
Uniform	4.13
DYNAMIC LOSS-BASED	3.10
DoReMi	3.30
DOGE	3.31
RegMix	4.51
Data Mixing Law	4.50
DoGraph (Ours)	**3.09**

Table 2: Pre-training results on SlimPajama. Validation perplexity (PPL) comparison across domain-mixing strategies. Lower values indicate better generalization.

3.4 DoGraph Stability across Model Scales

As shown in Figure 2, validation perplexity decreases with model scale, but the rate of improvement depends on the reweighting strategy. Uniform weighting yields consistently high perplexity, while RegMix offers partial gains that diminish as models grow. DoGraph achieves the lowest perplexity across all scales, validating its ability to dynamically balance domains.

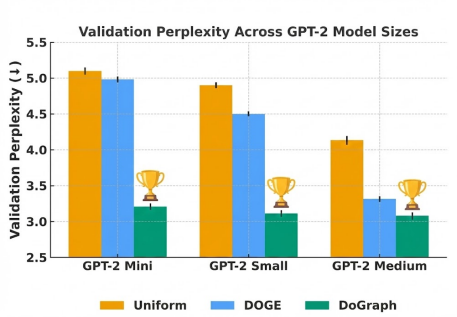

Figure 2: Perplexity across GPT-2 model sizes.

4 Conclusion

We revisited data mixing for LLMs through the lens of gradient dynamics. By characterizing domain differences via gradient geometry, we proposed **DoGraph**, a graph-constrained reweighting framework that adaptively balances domains during training. Experiments across model scales and benchmarks show that DoGraph improves both domain balance and generalization. Our results suggest that domains should be defined by the model's evolving representation rather than human intuition.

5 Limitations

While DoGraph achieves consistent improvements across domains and already reduces computational overhead through randomized gradient projection, its efficiency can still be further optimized. Future work will explore more lightweight aggregation and weighting strategies to enhance scalability in large-scale training.

References

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, and 1 others. A survey on data selection for language models. *Transactions on Machine Learning Research*.

Alexei Baevski and 1 others. 2024. Datacomp: In search of the next generation of multimodal datasets. *arXiv preprint arXiv:2304.14108*.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, and 1 others. 2020. PiQA: Reasoning About Physical Commonsense in Natural Language. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think You Have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge. *arXiv preprint arXiv:1803.05457*.

Simin Fan, Matteo Pagliardini, and Martin Jaggi. 2023. DOGE: Domain reweighting with generalization estimation. In *Second Agent Learning in Open-Endedness Workshop*.

Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. 2019. Deep Ensembles: A Loss Landscape Perspective. *arXiv preprint arXiv:1912.02757*.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. 2019. Representation Degeneration Problem in Training Natural Language Generation Models. *arXiv preprint arXiv:1907.12009*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2021. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, and 5 others. 2023. A framework for few-shot language model evaluation.

Yuxian Gu, Li Dong, Hongning Wang, Yaru Hao, Qingxiu Dong, Furu Wei, and Minlie Huang. 2024. Data Selection via Optimal Control for Language Models. *arXiv preprint arXiv:2410.07064*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, and 1 others. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Feiyang Kang, Yifan Sun, Bingbing Wen, Si Chen, Dawn Song, Rafid Mahmood, and Ruoxi Jia. 2024. AutoScale: Automatic Prediction of Compute-Optimal Data Composition for Training LLMs. *arXiv preprint arXiv:2407.20177*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Pang Wei Koh and Percy Liang. 2017. Understanding Black-box Predictions via Influence Functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1885–1894. PMLR.

Belinda Z. Li and 1 others. 2024. Dolma: An open corpus of three trillion tokens for language model pretraining research. *arXiv preprint arXiv:2402.00159*.

Jian Liu, Leyang Cui, Hanmeng Liu, Dandan Huang, Yile Wang, and Yue Zhang. 2020. LogiQA: A Challenge Dataset for Machine Reading Comprehension with Logical Reasoning. *arXiv preprint arXiv:2007.08124*.

Nan Liu and 1 others. 2024. Regmix: Regularizing data mixtures for language model pretraining. *arXiv preprint arXiv:2407.10671*.

Sachin Mehta, Mohammad Hossein Sekhavat, Qingqing Cao, Maxwell Horton, Yanzi Jin, Chenfan Sun, Iman Mirzadeh, Mahyar Najibi, Dmitry Belenko, Peter Zatloukal, and Mohammad Rastegari. 2024. OpenELM: An Efficient Language Model Family with Open Training and Inference Framework. *arXiv preprint arXiv:2404.14619*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering. *arXiv preprint arXiv:1809.02789*.

Denis Paperno, Germán Kruszewski, Angeliki Lazaridou, Quan Ngoc Pham, Raffaella Bernardi, Sandro Pezzelle, Marco Baroni, Gemma Boleda, and Raquel Fernández. 2016. The LAMBADA Dataset: Word Prediction Requiring a Broad Discourse Context. *arXiv preprint arXiv:1606.06031*.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. 2022. Grokking: Generalization Beyond Overfitting on Small Algorithmic Datasets. *arXiv preprint arXiv:2201.02177*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. OpenAI Blog. Version 1, Issue 8.

Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. 2020. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. In *International Conference on Learning Representations (ICLR)*.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. WinoGrande: An Adversarial Winograd Schema Challenge at Scale. *Communications of the ACM*, 64(9):99–106.

Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. 2020. SuperGlue: Learning Feature Matching with Graph Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4938–4947.

Daria Soboleva, Faisal Al-Khateeb, Robert Myers, Jacob R. Steeves, Joel Hestness, and Nolan Dey. 2023. SlimPajama: A 627B Token Cleaned and Deduplicated Version of RedPajama. Dataset available at: https://huggingface.co/datasets/cerebras/SlimPajama-627B.

Debarun Sow, Hannes Woisetschläger, Subhabrata Bulusu, Shuyang Wang, Hans-Arno Jacobsen, and Yuhao Liang. 2025. Dynamic loss-based sample reweighting for improved large language model pretraining. In *The Thirteenth International Conference on Learning Representations*.

Howe Sun and 1 others. 2025. Domain2vec: Vectorizing datasets to find the optimal data mixture without training. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, pages 5998–6008. Curran Associates, Inc.

Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing Multiple Choice Science Questions. *arXiv preprint arXiv:1707.06209*.

Alexander Wettig, Aatmik Gupta, Saumya Malik, and Danqi Chen. 2024. Qurating: Selecting high-quality data for training language models. In *International Conference on Machine Learning*, pages 52915–52971. PMLR.

Sang Michael Xie, Huyen Pham, Xiaowei Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. 2023. Doremi: Optimizing data mixtures speeds up language model pretraining. In *NeurIPS*.

Yuanjian Xu, Qi An, Jiahuan Zhang, Peng Li, and Zaiqing Nie. 2023. Hard sample aware prompt-tuning. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12356–12369.

Jiasheng Ye, Peng Liu, Tianxiang Sun, Yichao Zhou, Jian Zhan, and Xipeng Qiu. 2024. Data mixing laws: Optimizing data mixtures by predicting language modeling performance. *arXiv preprint arXiv:2403.16952*.

Dongkeun Yoon. 2023. SlimPajama-6B. https://huggingface.co/datasets/DKYoon/SlimPajama-6B. Accessed: 2024-09-24.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? *arXiv preprint arXiv:1905.07830*.

A Appendix

Contents

A.1 Connections to Prior Work

We categorize data mixture optimization into two main paradigms: offline and online approaches.

Offline approaches predefine mixture ratios before training. Early scaling-law studies (Kaplan et al., 2020; Hoffmann et al., 2022) established the relationship between model size, data volume, and compute, motivating subsequent work that explicitly models how mixture composition affects performance. Methods such as DoReMi (Xie et al., 2023), RegMix (Liu et al., 2024), and Mixing Laws (Ye et al., 2024) optimize mixture ratios using proxy models or learned predictors, improving efficiency but requiring retraining when datasets change. Other efforts focus on heuristic sample scoring to derive refined data mixtures (Gu et al., 2024), distinct from large-scale corpora that offer fixed domain ratios for benchmarking (Gao et al., 2021; Baevski et al., 2024; Li et al., 2024). Domain2Vec (Sun et al., 2025) further introduces dataset vectorization and distribution alignment, enabling mixture optimization without proxy models.

Online approaches adjust mixtures adaptively during training. Representative methods such as Group-DRO (Sagawa et al., 2020) dynamically reweight domains to improve worst-case generalization under distribution shift. While effective, they rely on explicit domain labels and are costly to scale.

A.2 Experimental Details

Benchmarks. We evaluate our method on nine diverse downstream benchmarks to assess its real-world impact. Guided by prior work (Mehta et al., 2024) and our own observations, we selected these tasks for their performance stability, excluding volatile benchmarks like RTE. The chosen tasks are HellaSwag (Zellers et al., 2019), PiQA (Bisk et al., 2020), OpenBookQA (Mihaylov et al., 2018), Lambada (Paperno et al., 2016), SciQ (Welbl et al., 2017), ARC-Easy (Clark et al., 2018), COPA (Sarlin et al., 2020), LogiQA (Liu et al., 2020), and WinoGrande (Sakaguchi et al., 2021). All evaluations use the `lm-eval-harness` (Gao et al., 2023), and we report normalized accuracy where available, otherwise standard accuracy.

Baselines. To rigorously assess the effectiveness of our proposed method, DoGraph, we benchmark it against a diverse set of reweighting baselines spanning three levels of granularity. We first include the uniform mixing baseline, where all samples contribute equally, as a fundamental reference. We then compare DoGraph with state-of-the-art domain-level reweighting methods, including DoGE (Fan et al., 2023), DoReMi (Xie et al., 2023), Regmix (Liu et al., 2024), and Data Mixing Law (Ye et al., 2024). Finally, to evaluate performance at a finer granularity, we incorporate a representative sample-level reweighting approach, Dynamic Loss-based Sample Reweighting (Sow et al., 2025).

Training Datasets. Our training data strategy is designed to align dataset scale with model capacity. For all GPT-2 models, we utilize the **SlimPajama-6B** dataset (Yoon, 2023), a 6-billion-token corpus comprising seven diverse domains: ArXiv, Books, Common Crawl, C4, GitHub, StackExchange, and Wikipedia. The byte proportion of each source is detailed in Table 6, illustrating the composition of the data mixture used for training. For all LLaMA models, we conduct our experiments using the domains of the Pile dataset (Gao et al., 2021) depicted in Table 7. Due to copyright concerns, we utilize the 17 subsets available on HuggingFace that do not violate copyright issues. These datasets provide a balanced and diverse text distribution suitable for evaluating cross-domain generalization in medium-scale language models.

Model Architecture. Following prior studies (Liu et al., 2024; Sow et al., 2025), we consider both model architecture and model scale in our evaluation, as summarized in Table 8. Specifically, we evaluate two decoder-only Transformer models based on GPT-2 architecture and two models based on LLaMA architecture, ranging from lightweight to medium scales.

Training Process. Following standardized practices in prior work, we train all models under pro-

Method	Commonsense / Reasoning						RC		LM	Avg
	HellaSwag	PiQA	OBQA	COPA	LogiQA	WinoG	SciQ	ARC-E	Lambada	
The Pile										
Uniform	29.5	58.8	27.3	65.8	23.9	50.5	60.3	40.0	11.7	40.9
Dynamic Loss-Based	29.0	57.7	26.4	64.3	22.8	49.3	60.0	38.9	10.2	39.9
DoReMi	29.4	58.3	27.3	**67.5**	26.4	52.2	61.6	40.6	12.1	41.7
DOGE	29.2	58.5	27.1	64.5	23.2	49.8	60.1	40.0	11.7	40.5
RegMix	29.2	**59.3**	27.3	65.2	25.8	**53.1**	62.8	**41.7**	14.2	42.1
Data Mixing Law	29.2	58.8	26.9	67.2	23.6	50.4	58.6	39.0	11.9	40.6
DoGraph (Ours)	**29.8**	59.2	**27.8**	65.0	**28.3**	51.2	**66.1**	39.2	**15.9**	**42.5**

Table 3: Downstream benchmark results (**accuracy %**) on The Pile (LLaMA-1.1B). Tasks are grouped into **Commonsense/Reasoning**, **Reading Comprehension**, and **Language Modeling**. Our method, DoGraph, achieves consistently better and more balanced results across domains, demonstrating its competitiveness and generalization ability. The best results are highlighted in bold.

Method	Commonsense / Reasoning						RC		LM	Avg
	HellaSwag	PiQA	OBQA	COPA	LogiQA	WinoG	SciQ	ARC-E	Lambada	
The Pile										
Uniform	29.6	58.8	29.4	66.0	25.9	51.1	61.0	39.1	12.6	41.5
Dynamic Loss-Based	29.3	58.1	29.6	66.1	25.0	52.5	62.7	39.9	12.2	41.7
DoReMi	29.6	58.4	29.8	66.0	24.9	51.4	61.1	40.5	12.8	41.6
DOGE	**29.7**	56.9	29.2	64.0	25.5	50.6	61.7	40.6	11.9	41.1
RegMix	29.4	59.5	29.4	66.5	25.1	**53.6**	62.5	**41.2**	12.3	42.2
Data Mixing Law	29.3	58.4	**30.2**	65.9	25.6	51.3	61.3	40.2	12.1	41.6
DoGraph (Ours)	29.6	**60.5**	29.0	**67.0**	**29.7**	51.4	**65.2**	40.2	**15.1**	**43.1**

Table 4: Downstream benchmark results (**accuracy %**) on The Pile (LLaMA-3.2-3B). Tasks are grouped into **Commonsense/Reasoning**, **Reading Comprehension**, and **Language Modeling**. Our method, DoGraph, achieves consistently better and more balanced results across domains, demonstrating its competitiveness and generalization ability. The best results are highlighted in bold.

Method	Commonsense / Reasoning						RC		LM	Avg
	HellaSwag	PiQA	OBQA	COPA	LogiQA	WinoG	SciQ	ARC-E	Lambada	
SlimPajama										
Uniform	26.0	55.4	13.8	57.2	22.8	49.3	32.6	30.6	12.0	33.3
Dynamic Loss-Based	26.2	56.1	13.2	55.3	26.0	49.2	**53.8**	31.8	12.6	36.2
DoReMi	26.1	55.7	12.3	53.5	**26.8**	48.8	52.4	30.9	12.4	35.4
DOGE	26.2	55.0	14.4	**60.5**	23.5	49.0	31.1	30.8	11.4	33.5
RegMix	26.0	54.3	13.3	58.0	24.1	**49.8**	38.7	29.8	12.5	34.1
Data Mixing Law	26.1	56.3	13.4	59.2	24.5	48.9	39.6	30.1	12.4	34.5
DoGraph (Ours)	**26.3**	**57.5**	**14.6**	58.0	26.0	49.7	**53.8**	**32.3**	**12.8**	**36.4**

Table 5: Downstream benchmark results (**accuracy %**) on SlimPajama (GPT-2 Small). Tasks are grouped into **Commonsense/Reasoning**, **Reading Comprehension**, and **Language Modeling**. Our method, DoGraph, achieves consistently better and more balanced results across domains, demonstrating its competitiveness and generalization ability. The best results are highlighted in bold.

tocols summarized in Table 9. Specifically, we adopt a linear warmup cosine schedule with identical weight decay (0.01) and gradient clipping (1.0) across all model scales, while adjusting batch size and training steps according to model capacity. This setup ensures that each model is trained sufficiently to convergence.

A.3 DoGraph Pipeline

Formalized in Algorithm 1, the process begins by projecting high-dimensional per-sample gradients $g_i^{(t)} \in \mathbb{R}^d$ into a lower-dimensional subspace $\tilde{g}_i^{(t)} \in \mathbb{R}^k$ using a random Gaussian matrix $R^{(t)}$, where we set $k = 5000$ for both SlimPajama and

Data Source	Byte Proportion
Common Crawl	54.1%
C4	28.7%
GitHub	4.2%
Books	3.7%
ArXiv	3.4%
Wikipedia	3.1%
StackExchange	2.8%

Table 6: Byte proportion of data sources in the SlimPajama-6B dataset.

The Pile to preserve the gradient manifold's geometric properties per the Johnson-Lindenstrauss Lemma. Subsequently, we identify latent optimization structures by applying K-means clustering to these projected signals, partitioning the mini-batch into $m = 11$ model-centric domains $\{D_j^{(t)}\}$ and computing their respective centroid gradients $\bar{g}_j^{(t)}$. Finally, importance weights $w^{(t)} \in \Delta^{m-1}$ are determined by solving the auxiliary objective $\mathcal{L}_{\text{opt}}$, and the model parameters θ are updated via the weighted aggregate $\sum_{j=1}^{m} w_j^{(t)} \bar{g}_j^{(t)}$, effectively decoupling training dynamics from static, pre-defined domain labels.

A.4 Scaling to Larger Datasets and Model Sizes

We report results on GPT-2 models from 210M to 300M parameters and a 6B-token SlimPajama subset, as shown in Table 5. DoGraph is scale-free and does not rely on any model-size–specific assumptions. All components, including gradient extraction, random projection, clustering, and domain-level optimization, operate directly on per-step gradients and thus scale linearly with model size. The method does not require proxy models, validation-model fitting, or domain-specific metadata, making it naturally compatible with billion-parameter LLMs. To further prove these, we pretrain LLaMA-1.1B and LLaMA-3B from scratch under the same DoGraph pipeline, as shown in Table 3 and Table 4.

A.5 Clustering Visualization

As shown in Figure 3, while human-defined domains (indicated by colors) become indistinguishable later in training, DoGraph successfully extracts m latent structures from this mixture, proving that model-centric domains are composed of heterogeneous data sources.

Component	Effective Size
Pile-CC	227.12 GiB
PubMed Central	180.55 GiB
Books3	151.44 GiB
OpenWebText2	125.54 GiB
ArXiv	112.42 GiB
Github	95.16 GiB
FreeLaw	76.73 GiB
Stack Exchange	64.39 GiB
USPTO Backgrounds	45.81 GiB
PubMed Abstracts	38.53 GiB
Gutenberg (PG-19)	27.19 GiB
OpenSubtitles	19.47 GiB
Wikipedia (en)	19.13 GiB
DM Mathematics	15.49 GiB
Ubuntu IRC	11.03 GiB
BookCorpus2	9.45 GiB
EuroParl	9.17 GiB
HackerNews	7.80 GiB
YoutubeSubtitles	7.47 GiB
PhilPapers	4.76 GiB
NIH ExPorter	3.79 GiB
Enron Emails	1.76 GiB

Table 7: Overview of the Pile dataset with datasets that are no longer available due to copyright issues marked in gray. Merged into a single column list.

	GPT-2 Small	GPT-2 Medium	LLaMA-1.1B	LLaMA-3.2-3B
Parameters	210M	300M	1.1B	3B
Layers	24	36	22	28
Attention Heads	16	24	32	24
Embedding Dim.	768	768	2048	8192
Hidden Dim.	3072	3072	2048	3072
Max Seq. Length	512	512	2048	131072

Table 8: Model architectures used in our experiments.

A.6 More Analysis about the Choice of Optimization Function

At each training epoch, the DoGraph framework computes domain mean gradients $\{\bar{g}_j\}_{j=1}^{m}$ and determines their adaptive weights $w \in \Delta^{m-1}$ by minimizing an auxiliary objective $\mathcal{L}_{\text{opt}}$. Since m (the number of domains) is typically small, this optimization occurs in a low-dimensional space and can be efficiently solved in closed or iterative form. We discuss several representative objectives and their corresponding solvers below.

Gradient variance minimization. To balance the learning progress across domains, one may minimize the variance of gradient magnitudes while

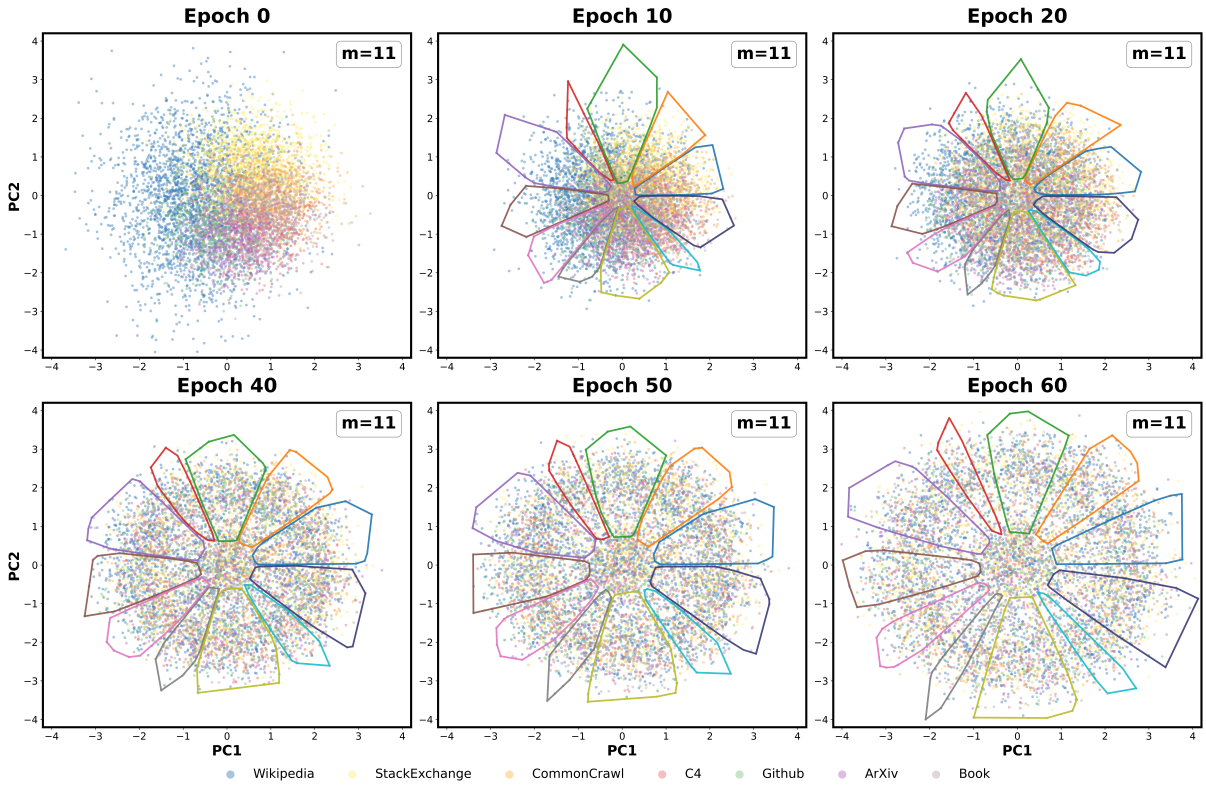

Figure 3: Evolution of per-sample gradients and the emergence of model-centric structures. Points are colored by their original source datasets. Initially, gradients are separated by domain bias. Over time, the model homogenizes its perception of these sources, leading to significant overlap. Despite this mixing, DoGraph identifies m=11 distinct model-centric domains within the gradient space. Experiments use 20% of SlimPajama trained on GPT2-Mini.

	GPT-2 Small	GPT-2 Medium	TinyLLaMA-1.1B	LLaMA-3.2-3B
Minibatch Size	48	48	64	64
Learning Rate ($\times 10^{-3}$)	0.50	0.50	0.50	0.50
Learning Rate End ($\times 10^{-4}$)	1.0	1.0	1.0	1.0
Warmup Steps	500	500	500	500
r	0.4	0.4	0.4	0.4
Training Steps	20,000	20,000	25,000	25,000
Total Documents Seen	960,000	960,000	1280,000	1280,000

Table 9: Training hyperparameters for GPT-2 and LLaMA models in our benchmark evaluations.

maintaining the global descent direction:

$$\mathcal{L}_{\text{opt}}(w) = \text{Var}_j\big[\|w_j \bar{g}_j\|_2\big] + \lambda \Big\| \sum_{j=1}^{m} w_j \bar{g}_j \Big\|_2^2.$$

This convex quadratic problem can be solved by projected gradient descent or quadratic programming with a simplex constraint.

Robust min–max objective. When robustness against hard or under-represented domains is desired, one may adopt a distributionally robust formulation:

$$\mathcal{L}_{\text{opt}}(w) = \tau \log \sum_{j=1}^{m} \exp\big(\|\bar{g}_j\|_2 / \tau\big),$$

which smoothly approximates $\max_j \|\bar{g}_j\|_2$. The optimal weights admit a closed-form softmax solution $w_j \propto \exp(\|\bar{g}_j\|_2/\tau)$.

Gradient alignment regularization. To encourage consistent update directions across domains, we define

$$\mathcal{L}_{\text{opt}}(w) = -\sum_{j=1}^{m} w_j \cos(\bar{g}_j,\ \bar{g}), \quad \bar{g} = \sum_{j=1}^{m} w_j \bar{g}_j.$$

Although non-convex due to the dependence of $\bar{g}$ on w, it can be efficiently solved by a few fixed-point iterations: each step updates w_j in proportion to the cosine similarity between $\bar{g}_j$ and the current aggregate $\bar{g}$.

Domain uncertainty weighting. Alternatively, if each domain exhibits distinct gradient variability, we estimate its intra-domain variance $\sigma_j^2 = \text{Var}_{i \in D_j}\big[\|g_i - \bar{g}_j\|_2^2\big]$ and assign weights inversely proportional to it:

$$\mathcal{L}_{\text{opt}}(w) = \Big\| \sum_j w_j \bar{g}_j \Big\|_2^2 + \beta \sum_j \sigma_j^2 w_j.$$

This convex quadratic form admits a closed-form Newton update.

Algorithm 1: Dograph Pipeline

Input: Training data $\mathcal{D}$, parameters θ, number of clusters m, projection dimension k, epochs T, learning rate η

Output: Trained parameters θ^*, domain weights $\{w^{(t)}\}$

for $t = 1$ **to** T **do**

- Sample random projection matrix $R^{(t)} \in \mathbb{R}^{d\times k}$ with $R^{(t)}_{pq} \sim \mathcal{N}(0, 1/k)$;
- Compute per-sample gradients $g_i^{(t)} = \nabla_\theta L(x_i, y_i; \theta^{(t-1)})$;
- Project gradients: $\tilde{g}_i^{(t)} = R^{(t)\top} g_i^{(t)}$;
- Cluster $\{\tilde{g}_i^{(t)}\}$ into m domains $\{D_1^{(t)}, \ldots, D_m^{(t)}\}$;
- Compute domain mean gradients $\bar{g}_j^{(t)} = \frac{1}{|D_j^{(t)}|} \sum_{i\in D_j^{(t)}} \tilde{g}_i^{(t)}$;
- Optimize weights $w^{(t)} = \arg\min_{w\in\Delta^{m-1}} \mathcal{L}_{\text{opt}}\left(\sum_{j=1}^m w_j \bar{g}_j^{(t)}\right)$;
- Update model parameters: $\theta^{(t)} \leftarrow \theta^{(t-1)} - \eta \sum_{j=1}^m w_j^{(t)} \bar{g}_j^{(t)}$;

return $\theta^{(T)}, \{w^{(t)}\}_{t=1}^T$

Table 10 summarizes the computational complexity of each optimization objective and the corresponding validation perplexity PPL. All variants share the same backbone and differ only in the choice of $\mathcal{L}_{\text{opt}}$. The robust softmax objective achieves the lowest computational cost, while the uncertainty-weighted variant attains the best overall performance.

Table 10: Comparison of optimization objectives in DoGraph.

Method	Complexity	PPL
DoGraph (variance)	$O(m^2)$	3.24
DoGraph (robust softmax)	$O(m)$	3.31
DoGraph (alignment)	$O(m^2)$	3.15
DoGraph (uncertainty)	$O(m^2)$	**3.09**

Among all variants, **DoGraph (uncertainty)** achieves the lowest perplexity, indicating that weighting domains by intra-domain gradient stability provides the most consistent optimization dynamics.

A.7 Impact of Cluster Granularity m.

We investigate the sensitivity of model performance to the number of clusters m. As illustrated in Figure 1, the validation perplexity exhibits a clear U-shaped trend with respect to m. Performance initially improves as m increases from 7 to 11, suggesting that moderately finer-grained, model-centric domains better capture coherent gradient structures and facilitate optimization. However, further increasing m beyond 11 leads to a significant performance degradation. This decline is likely due to over-partitioning, which fragments the gradient space and splits coherent patterns into inconsistent components, thereby weakening signal consistency. Consequently, we select m=11 as our default setting for all subsequent experiments.

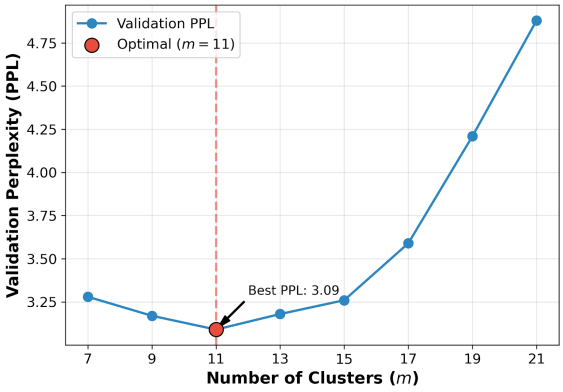

Figure 4: Impact of cluster granularity m on validation perplexity. The U-shaped curve demonstrates that $m = 11$ provides the optimal balance; insufficient granularity fails to resolve gradient structures, while excessive partitioning leads to signal inconsistency.

A.8 Computational Efficiency Analysis

As shown in Figure 5, `dograph` achieves state-of-the-art performance while introducing a modest and practical computational overhead. On a 2× H200 GPU cluster, our method completes pre-training in 20.37 hours, corresponding to a 4.51% increase in runtime compared to `regmix`. This incremental cost falls within the commonly accepted budget for large-scale pre-training.

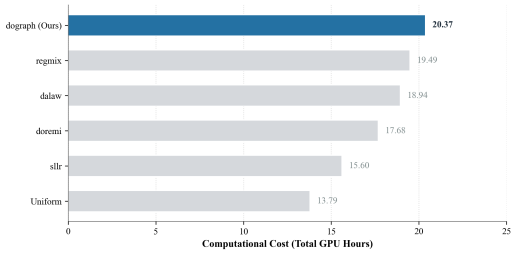

Figure 5: Pre-training GPT-2 Mini on SlimPajama under a 100B-Token Computational Budget. We report the total training time (GPU hours) using 2× NVIDIA H200 GPUs. While our `dograph` method introduces a sophisticated data-driven decision process, the resulting overhead is minimal (only 4.51% over `regmix`), while establishing a new SOTA performance baseline. The marginal increase in budget is well-justified by the superior convergence quality and data selection efficiency.

A.9 Proofs

Assumption A.1 (Linearized Attention Mechanism)**.** *Let $X \in \mathbb{R}^{n\times d}$ denote the sequence representation. We define the projected queries, keys, and values as $Q = XW_Q$, $K = XW_K$, and $V = XW_V$, and the scaled similarity matrix as $S = \frac{1}{\sqrt{d_k}}QK^\top$. The row-wise softmax of S is approximated by its first-order linearization: $P \approx A + \frac{1}{\tau n}CS$, where $A = \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, $C = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$. Consequently, the attention output satisfies $O = PV \approx AV + TQK^\top V$, where $T = \frac{1}{\tau n\sqrt{d_k}}\,C$.*

Assumption A.2 (Linear Output Transformations)**.** *The attention output O is passed through two linear mappings: $H = OW_O$ and $Z = HW$. The model prediction is obtained via $\Pi = \mathrm{softmax}(Z)$, which represents the token-level probability distribution.*

Assumption A.3 (Upstream Gradients and Mismatch Tensor)**.** *Given the ground-truth label matrix Y, the upstream gradients are defined as $G_Z = \Pi - Y$, $G_H = G_Z W^\top$, and $G_O = G_H W_O^\top$. We further define the mismatch tensor $R = (\Pi - Y)M$, where $M = W^\top W_O^\top$.*

Assumption A.4 (Regularity Conditions)**.** *All expectations involved in subsequent derivations are assumed to exist, and the per-sample gradients are square-integrable.*

Per-sample gradients and proof of Theorem 3.2 .

With $dL = \langle G_O, dO\rangle$ and $O \approx AV + TQK^\top V$, the per-sample gradients are

$$\begin{cases} \dfrac{\partial L}{\partial V} = A^\top G_O + KQ^\top T^\top G_O, \\ \dfrac{\partial L}{\partial Q} = (T^\top G_O)\, V^\top K, \\ \dfrac{\partial L}{\partial K} = V\, G_O^\top T\, Q, \\ \dfrac{\partial L}{\partial W} = H^\top G_Z \end{cases}$$

$$\begin{cases} \dfrac{\partial L}{\partial W_V} = X^\top\big(A^\top G_O + KQ^\top T^\top G_O\big), \\ \dfrac{\partial L}{\partial W_Q} = X^\top\big((T^\top G_O)\, V^\top K\big), \\ \dfrac{\partial L}{\partial W_K} = X^\top\big(V\, G_O^\top T\, Q\big), \\ \dfrac{\partial L}{\partial W_O} = O^\top G_H \end{cases}$$

From Assumption A.3, the upstream gradient can be written as $G_O = (\Pi - Y)W^\top W_O^\top = R$. Substituting this into the above expressions shows that all per-sample gradients are *linear* functions of R:

$$\nabla_{W_b} L(x, y; \theta) \;=\; \mathsf{Lin}_b(X, Q, K, V, T)\,[\,R(x, y)\,],$$

where $\mathsf{Lin}_b(\cdot)$ denotes a matrix-valued linear operator determined only by the forward pass variables X, Q, K, V, T. Using the identity $\mathrm{vec}(UGV) = (V^\top \otimes U)\mathrm{vec}(G)$, each matrix gradient can be rewritten in vectorized form as

$$g_b(s) := \mathrm{vec}\big(\nabla_{W_b} L(x, y; \theta)\big) \;=\; \mathcal{L}_b(x)\,\rho(s),$$
$$\rho(s) := \mathrm{vec}\big(R(s)\big).$$

Here, $\mathcal{L}_b(x)$ absorbs all Kronecker factors (e.g., $X^\top$, $T^\top$, K, Q) from the explicit gradient expressions. Hence, for every parameter block b, the sample-wise gradient is a linear transformation of the mismatch vector $\rho(s)$.

Define the expected gradient under a data distribution P as $\bar{g}_b(P) := \mathbb{E}_{s\sim P}[g_b(s)]$. By linearity of expectation (Bochner integral in finite dimensions),

$$\bar{g}_b(P_1) - \bar{g}_b(P_2) \;=\; \int g_b(s)\,(P_1 - P_2)(ds).$$

The inner product between two per-sample gradients naturally defines

$$k_b(s, s') := \langle g_b(s),\, g_b(s')\rangle$$
$$= \rho(s)^\top\, \mathcal{L}_b(x)^\top \mathcal{L}_b(x')\, \rho(s').$$

Because k_b is an inner product in feature space, it is positive semidefinite. Applying Fubini–Tonelli and the bilinearity of the inner product yields

$$
\begin{aligned}
&\left\|\bar{g}_b(P_1) - \bar{g}_b(P_2)\right\|_2^2 \\
&= \Big\langle \int g_b \, d(P_1 - P_2),\ \int g_b \, d(P_1 - P_2) \Big\rangle \\
&= \iint \langle g_b(s), g_b(s') \rangle \, (P_1 - P_2)(ds) \, (P_1 - P_2)(ds') \\
&= \mathbb{E}_{P_1,P_1}[k_b] + \mathbb{E}_{P_2,P_2}[k_b] - 2\,\mathbb{E}_{P_1,P_2}[k_b],
\end{aligned}
$$

which is exactly $\mathrm{MMD}^2_{k_b}(P_1, P_2)$ by definition. Finally, the positive semidefiniteness of k_b follows from

$$
\sum_{i,j} \alpha_i \alpha_j k_b(s_i, s_j) = \Big\| \sum_i \alpha_i g_b(s_i) \Big\|_2^2 \geq 0.
$$

This completes the proof. □