IDEAL: Influence-based Data Equilibrium Adaptation for Multi-Capability Language Model Alignment

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have achieved impressive performance through Supervised Fine-tuning (SFT) on diverse instructional datasets. When training on multiple capabilities simultaneously, the optimal data mixture proportions remain underexplored. In this work, we propose **IDEAL**, an Influence-based Data Equilibrium Adaptation framework, which aims to optimize the mixture proportions of distinct SFT datasets based on their task-specific performance. IDEAL employs a machine learning-driven approach based on influence function to iteratively refine the data allocation strategy, prioritizing datasets that enhance target capabilities. Experiments across different capabilities demonstrate that IDEAL significantly outperforms conventional uniform data allocation strategies, achieving strong improvements across diverse tasks.

1 Introduction

Recent advancements in Large Language Models (LLMs) have demonstrated their remarkable ability to master diverse capabilities (Dong et al., 2023; Zhang et al., 2024b; Hu et al., 2023; Mecklenburg et al., 2024) through Supervised Fine-Tuning (SFT) on instruction-aligned datasets (Liu et al., 2023; Lu et al., 2023; Agarwal et al., 2024; Wang et al., 2023). However, a critical challenge persists when harmonizing diverse capabilities during SFT: the optimal mixture proportions of these domains is poorly understood. While heuristic solutions such as manual data reweighting or rule-based curriculum learning (Bengio et al., 2009) exist, they suffer from scalability limitations and suboptimal task balance. Prior attempts to automate data allocation, including pretraining-centric methods (Xie et al., 2024; Ye et al., 2024), fail to address the unique dynamics of SFT—where data-task alignment directly governs cross-domain interference. Consequently, a principled framework for resolving data conflicts in multi-capability SFT remains an open problem.

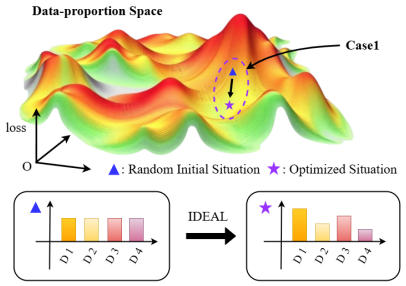


Figure 1: IDEAL adjusts data proportions to optimize model performance, leading to a decrease in loss.

To bridge this gap, we propose **IDEAL** (Influence-based Data Equilibrium Adaptation Learning), a novel framework that dynamically aligns SFT data mixtures with model capabilities. IDEAL employs the influence function (Koh and Liang, 2017)—a second-order optimization tool to optimize the data mix ratios. Unlike previous works use influence function for data sample selection (Xia et al., 2024; Zhang et al., 2024a), we instead employ influence function for dataset capability measurement to optimize the data mix ratios. By iteratively refining dataset proportions based on IDEAL, it prioritizes data subsets that synergistically enhance target capabilities. This model-aware mechanism adapts to the LLM's evolving training dynamics, ensuring equilibrium between data efficiency and task balancing. Crucially, our framework operates without costly hyperparameter sweeps, enabling scalable multi-capability SFT with theoretical guarantees.

Extensive experiments validate IDEAL's effectiveness across diverse capability combinations. On BigBench Hard, GSM8K, HumanEval and IFEval, IDEAL outperforms uniform data blending by 9% on average. Further studies demonstrate the robustness of IDEAL by again improving on other initial seed data scale. These results establish our IDEAL

as a critical lever for training generalist LLMs.

2 Related Works

Data Mixing. Data mixing optimizes training data distributions to enhance multi-task performance. Traditional approaches rely on token ratios (Touvron et al., 2023; Liu et al., 2025) or quality-driven selection (Parmar et al., 2024; Chung et al., 2023; Engstrom et al., 2024; Xia et al., 2024; Kang et al., 2024). Recent learning-based methods for LLMs optimize domain weights via proxy models: DoReMi (Xie et al., 2024) uses distributionally robust optimization on a small proxy model, while DOGE (Fan et al., 2023) extends this to domain-specific re-weighting. Others derive empirical laws from large-scale experiments, such as the mixing proportion law in (Ye et al., 2024). However, these methods require costly global weight searches and often disregard the continuity of data distributions. Our work addresses these gaps by gradient-guided iterative refinement, enabling efficient adaptation.

Influence Function. The Influence Function (Hampel, 1974) provides interpretable connections between training data and model behavior. Recent work extends it to analyze LLMs: Koh et al. (Koh and Liang, 2017) formalize its role in linking datasets to performance, while gradient-based approximations (Xia et al., 2024; Yu et al., 2024) enable data selection via influence scores despite computational challenges (Grosse et al., 2023). Building on these insights, we propose an efficient influence estimator for SFT, optimizing domain weight allocation by quantifying how training proportions affect multi-task generalization.

3 Methodology

3.1 Problem Formulation

To enhance the capabilities of the base model $\mathcal{M}_0$ within specific domains, we develop corresponding high-quality training datasets $\mathcal{D}_1, \ldots, \mathcal{D}_n$. When integrating these diverse datasets for training, challenges such as data distribution shifts inevitably arise. These shifts can significantly affect the effectiveness of the model training process. To mitigate the data shift, our objective is to determine an optimal mixing ratio for the training datasets.

In a common learning approach, the objective is to minimize the training cost function:

$$\theta^* = \arg\min_\theta \mathcal{L}(\mathcal{D}_t, \theta) = \arg\min_\theta \frac{1}{N} \sum_{z_i \in \mathcal{D}_t} \mathcal{L}(z_i, \theta),$$

where θ is the parameter of the model and $\mathcal{D}_{tr} = [\mathcal{D}_1, \ldots, \mathcal{D}_n]$ represents the whole training dataset. Let N be the total number of training samples $N = |\mathcal{D}_{tr}| = |\mathcal{D}_1| + \cdots + |\mathcal{D}_n| = t_1 + \cdots + t_n$, where $t_1, \ldots, t_n$ represent the numbers of training samples in datasets $\mathcal{D}_1, ..., \mathcal{D}_n$, respectively. In an ideal scenario, an optimal solution θ^* can be found via effective optimization techniques. Inspired by the findings in (Muennighoff et al., 2023), which shows that conducting fewer than 4 training epochs can enhance the model's performance to a degree comparable to using new data, we train for only 1 epoch while simultaneously utilizing downsampling and upsampling techniques to adjust the quantities of different training datasets. As a result, we model the problem in the following manner:

$$\theta^* = \arg\min_\theta \frac{\sum_{z_i \in \mathcal{D}_{tr}} \mathcal{L}(z_i, \theta) + \sum_{j=1}^n \beta_j \sum_{z_i \in \mathcal{D}_j} \mathcal{L}(z_i, \theta)}{N + \sum_{j=1}^n \beta_j t_j}. \tag{1}$$

In this formulation, we use $\beta = (\beta_1, \ldots, \beta_n)$ to control the size of the corresponding training data set. Initially, we utilize the entire training dataset to fine-tune the model $\mathcal{M}_0$ into $\mathcal{M}_1$. Our objective is to understand the impact of β on the model's performance on the validation set $\mathcal{D}_{ref}$. By applying the chain rule, we can determine the impact of a specific $\beta_m \in \{\beta_1, \ldots, \beta_n\}$ on the model's performance on the validation set $\mathcal{D}_{ref}$:

$$\frac{\partial \mathcal{L}(\mathcal{D}_{ref}, \theta^*)}{\partial \beta_m} = \frac{\partial \mathcal{L}(\mathcal{D}_{ref}, \theta^*)}{\partial \theta^*} \frac{\partial \theta^*}{\partial \beta_m}. \tag{2}$$

Lemma 1 *The impact of a specific β_m on the optimal model parameters θ^* trained on the training set $\mathcal{D}_t$ can be explicitly expressed as (3).*

As mentioned before, we initialize the $\beta = (0, \ldots, 0)$, we can get the influence of β_m on the validation set $\mathcal{D}_{ref}$ according to the (2):

$$\frac{\partial \mathcal{L}(\mathcal{D}_{ref}, \theta^*)}{\partial \beta_m} = -\nabla \mathcal{L}(\mathcal{D}_{ref}, \theta^*) \cdot \left[\sum_{z_i \in \mathcal{D}_{tr}} \nabla^2 \mathcal{L}(z_i, \theta^*) \right]^{-1} \nabla \mathcal{L}(\mathcal{D}_m, \theta^*), \tag{4}$$

which is equal to the influence function equation.

3.2 Efficient Calculation

Evaluating the Gauss-Newton Hessian in the context of (4) presents a formidable challenge. Currently, it is computationally infeasible to directly calculate the inverse of the Hessian matrix for the

$$\frac{\partial \theta^*}{\partial \beta_m} = -\left[\sum_{z_i \in \mathcal{D}_{tr}} \nabla^2 \mathcal{L}(z_i, \theta^*) + \sum_{j=1}^{n} \beta_j \nabla^2 \mathcal{L}(\mathcal{D}_j, \theta^*)\right]^{-1} \nabla \mathcal{L}(\mathcal{D}_m, \theta^*). \tag{3}$$

entire set of parameters. According to K-FAC theory (Martens and Grosse, 2015; Ueno et al., 2020; Zhang et al., 2024a), we use the kronecker product to accelerate the iHVP computation. During the intermediate state of the calculation, we will obtain Λ, which captures the variances of the pseudo-gradient projected onto each eigenvector of the K-FAC approximation. We then identify the 'important' MLP layers by choosing those with lower variances, as these layers exhibit enhanced stability. Reducing the number of calculation layers can significantly alleviate the storage pressure. However, it will also lead to a relatively small magnitude of the final result. To address the above issue, we introduce a dynamic scaling vector γ, which linearly scales the maximum and minimum absolute value in the calculated β to a predefined value range $[m, n], 0 < m < n < 1$. We update the β values as shown in (5):

$$\begin{aligned} \alpha_i =& \frac{\partial \mathcal{L}(\mathcal{D}_{ref}, \theta^*)}{\partial \beta_i}, \beta = -\gamma \odot \alpha, \\ \gamma_i =& \text{sgn}(\alpha_i) \left[m + \frac{(n-m)(\alpha_i - \min(\alpha))}{\max(\alpha) - \min(\alpha)}\right]. \end{aligned} \tag{5}$$

3.3 IDEAL Algorithm

The complete pipeline of our method shown in Algorithm 1. First, we mix all the datasets into $\mathcal{D}_{tr}$. Based on the base model, we train the $\mathcal{M}_1$ model and test it on various benchmarks to identify the weak area $\mathcal{D}_{ref}$. We adjust the ratio of the training set after calculating the β sequence for $\mathcal{D}_{ref}$. Finally, we train the $\mathcal{M}_2$ model on the base model using the newly adjusted training set. If further improvement is desired, the above steps can be iterated to get $\mathcal{M}_3, \ldots, \mathcal{M}_T$ based on the new training set until all the model's capabilities meet the expected standards.

Algorithm 1 IDEAL Algorithm

Require: Initial model $\mathcal{M}_0$, initial training set $\mathcal{D}_{tr} = [\mathcal{D}_1, ..., \mathcal{D}_n]$, maximum iterations T (or stop criteria).

```
for i = 1 to T do
    Train M_0 on D_tr until optimal, resulting in M_i;
    Test the performance of M_i;
    Compute β following (5);
    Update training set: D_tr ← D_tr + Σ_{j=1}^{n} β_j D_j
    if stopping criteria met then
        break
    end if
end for
```

	GSM8K	HumanEval	BBH	IFEval	Total
$\mathcal{D}^1$	10,000	5,374	6,511	2,000	23,885
$\mathcal{D}^2$	4,266	3,768	2,430	4,591	15,055

Table 1: Initial Dataset Statistics.

4 Experiments

4.1 Experiment Setup

Training Setting. We choose the LLama3.1-8B(Grattafiori et al., 2024) as our base model $\mathcal{M}_0$ to adopt full fine-tuning. All models/settings train for 1 epoch. For a fair comparison, each experiment is repeated for 5 runs to report average performance. Other settings can be found in Appendix A.1.

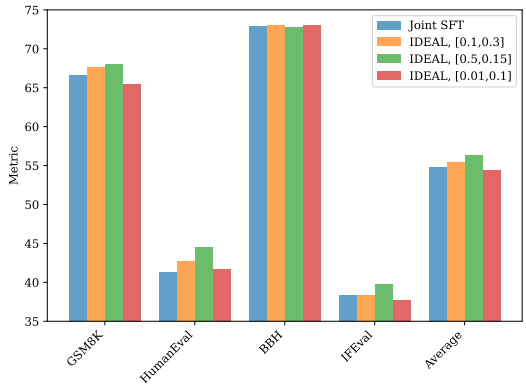
Dataset Preparetion. We select reasoning, mathematics, coding, and instruction-following domains and evaluate on BigBench Hard(BBH)(Suzgun et al., 2022), GSM8K(Cobbe et al., 2021), HumanEval(Chen et al., 2021), and IFEval(Zhou et al., 2023) benchmarks. The detailed dataset information is provided in Appendix A.1. To explore the impact of different state of initial data, we randomly generated two initial training data sets: $\mathcal{D}^1$ and $\mathcal{D}^2$ presented in Table 1.

Baseline. We compare the performance of our IDEAL with other data training strategies as follows. (1) **Specific SFT**, which only uses a specific domain training data for SFT. (2) **Joint SFT**, where the different capability data directly combined. (3) **Random**, we randomly sample different data scales for each capability. (4) **DoReMi** (Xie et al., 2024), which uses the group distributionally robust optimization (Group DRO) steps to generate new domain weights. (5) **DOGE** (Fan et al., 2023), which determines the data proportions between domains by minimizing the discrepancy in backpropagation gradients. For our IDEAL, we set the parameter $[m, n]$ as $[0.05, 0.15]$. In order to accelerate calculation speed, we sample the training set with a sample factor $\sigma = 0.5$.

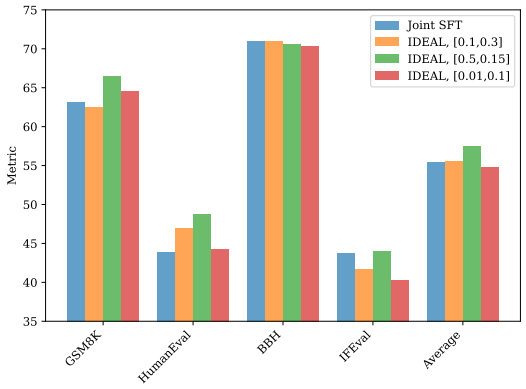
4.2 Main Results

Suboptimial initial data distribution. Joint SFT and random baselines underperform specific SFT across all benchmarks. While random sampling occasionally improves coding tasks (e.g., HumanEval), it fails to generalize to other domains.

Iterative re-weighting methods enhance the



(a) Models' performance after SFT on $\mathcal{D}_1^1$ with different β ranges.



(b) Models' performance after SFT on $\mathcal{D}_1^2$ with different β ranges.

Table 2: Performance comparison of different baselines.

Benchmark		GSM8K		HumanEval		BigBench Hard		IFEval	
Methods	**Dataset**	**Acc**	**Size**	**Pass@1**	**Size**	**Average**	**Size**	**Average**	**Size**
Base	-	56.41	-	27.44	-	62.13	-	12.22	-
Specific SFT	GSM8K	**65.81**	10,000	0.00	0	35.94	0	22.54	0
	HumanEval	48.14	0	37.20	5,374	2.99	0	19.66	0
	BBH	61.87	0	7.32	0	60.19	6,511	**26.70**	0
	IFEval	57.39	0	**46.95**	0	**61.87**	0	22.47	2,000
Joint SFT	$\mathcal{D}^1$	66.62	10,000	41.26	5,374	72.92	6,511	38.36	2,000
Random	-	63.84	3,514	43.90	7,418	75.11	11,420	39.70	3,061
	-	63.23	1,752	40.85	9,349	74.56	12,412	38.21	1,900
DoReMi	$\mathcal{D}^1_{1(\text{DoReMi})}$	65.96	5,000	41.63	8,061	73.44	9,766	34.26	1,057
	$\mathcal{D}^1_{2(\text{DoReMi})}$	64.82	5,323	43.90	1,2091	73.79	4,883	38.16	1,585
DOGE	$\mathcal{D}^1_{1(\text{DOGE})}$	64.82	1,1568	40.02	8,061	74.99	3,255	39.02	1,000
	$\mathcal{D}^1_{2(\text{DOGE})}$	67.10	9,665	42.24	12,091	73.59	1,627	30.53	500
IDEAL	$\mathcal{D}^1_{1(\text{IDEAL})}$	68.01	9,492	44.51	6,180	72.82	6,876	39.78	2,100
	$\mathcal{D}^1_{2(\text{IDEAL})}$	67.55	9,017	50.61	7,107	74.29	7,348	39.03	1,942

model's specific benchmark score. Methods like DoReMi and DOGE optimize data distributions through multi-step evolution chains (e.g., $\mathcal{D}^1 \rightarrow \mathcal{D}_1^1 \rightarrow \mathcal{D}_2^1$), yet their aggressive distribution shifts cause performance variance across benchmarks despite minor HumanEval gains. As shown in Table 2, both methods have a slight improvement in HumanEval, but there are large variance fluctuations in the model's scores on other benchmarks.

IDEAL achieves optimal balance in 2 iterations. By incrementally refining data ratios, IDEAL surpasses Joint SFT on all metrics and stabilizes performance across benchmarks, notably achieving HumanEval improvements without compromising other tasks—fulfilling efficiency and stability requirements.

5 Sensitivity Study

5.1 Sensitivity to the Selection of γ.

As shown in Lemma 1, the value of β is essentially a small perturbation around 0. The dynamic scaling vector γ plays a vital role in determining the magnitude of the adjustment of β. To explore the optimal range for β, we carry out experiments on three different settings for the range of γ: $[0.01, 0.1]$, $[0.05, 0.15]$, and $[0.1, 0.3]$. Training results on $\mathcal{D}^1$, $\mathcal{D}^2$ are shown in Figure 2a,2b, respectively.

The range of β should be neither too large nor too small. When β is constrained to [0.01, 0.1], limited adjustments yield marginal performance gains due to insufficient data proportion changes. Conversely, a broader range [0.1, 0.3] induces unstable capability fluctuations as drastic data shifts deviate from the original distribution. The optimal range [0.05, 0.15] balances moderate data adjustments with distribution integrity, enabling complementary cross-domain learning while sustaining multi-task stability—achieving the highest average performance through controlled yet impactful proportion updates.

5.2 Dependence on Initial Data Distribution.

Another important aspect to assess the robustness of the IDEAL algorithm is its dependence on the initial data distribution. We conduct experiments using $\mathcal{D}^1$ and $\mathcal{D}^2$ as two distinct initial data distributions. The experimental results, as shown in Figure 2a,2b, indicate that regardless of the initial data distribution, the IDEAL algorithm are able to significantly enhance the model's multi-capabilities. Both settings achieve an obvious improvement in HumanEval benchmark, suggesting that the IDEAL algorithm is robust to different initial data distributions and can adaptively optimize the training dataset proportions to achieve performance gains.

6 Conclusion

We propose an influence-based data equilibrium adaptation, IDEAL, which effectively optimizes dataset proportions for SFT. Our approach offers a scalable solution for multi-capability SFT, ensuring performance enhancement for LLM.

Limitations

Our approach uses approximation approaches due to the large parameter size of LLMs, which can create a gap between theoretical estimates and experimental results. We further analyze it in AppendixA.3. Additionally, the method relies on high-quality training datasets. If dataset quality is unverified, data generation or filtering techniques might be more beneficial for improving model performance.

References

Ishika Agarwal, Krishna Killamsetty, and Lucian et al. Popa. 2024. Delift: Data efficient language model instruction fine tuning. *arXiv preprint arXiv:2411.04425*.

Jacob Austin, Augustus Odena, and Maxwell Nye et al. 2021. Program synthesis with large language models. *Preprint*, arXiv:2108.07732.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Mark Chen, Jerry Tworek, and Heewoo Jun et al. 2021. Evaluating large language models trained on code.

Hyung Won Chung, Noah Constant, and Xavier et al. Garcia. 2023. Unimax: Fairer and more effective language sampling for large-scale multilingual pretraining. *arXiv preprint arXiv:2304.09151*.

Karl Cobbe, Vineet Kosaraju, and Mohammad et al. Bavarian. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

OpenCompass Contributors. 2023. Opencompass: A universal evaluation platform for foundation models. https://github.com/open-compass/opencompass.

Guanting Dong, Hongyi Yuan, and Keming et al. Lu. 2023. How abilities in large language models are affected by supervised fine-tuning data composition. *arXiv preprint arXiv:2310.05492*.

Logan Engstrom, Axel Feldmann, and Aleksander Madry. 2024. Dsdm: Model-aware dataset selection with datamodels. *arXiv preprint arXiv:2401.12926*.

Simin Fan, Matteo Pagliardini, and Martin Jaggi. 2023. Doge: Domain reweighting with generalization estimation. *arXiv preprint arXiv:2310.15393*.

Tao Ge, Xin Chan, and Xiaoyang et al. Wang. 2024. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*.

Aaron Grattafiori, Abhimanyu Dubey, and Abhinav Jauhri et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Roger Grosse, Juhan Bae, and Cem et al. Anil. 2023. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*.

Frank R Hampel. 1974. The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393.

Linmei Hu, Zeyi Liu, and Ziwang et al. Zhao. 2023. A survey of knowledge enhanced pre-trained language models. *IEEE Transactions on Knowledge and Data Engineering*.

Feiyang Kang, Hoang Anh Just, and Yifan et al. Sun. 2024. Get more for less: Principled data selection for warming up fine-tuning in llms. *arXiv preprint arXiv:2405.02774*.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR.

Qian Liu, Xiaosen Zheng, and Niklas Muennighoff et al. 2025. Regmix: Data mixture as regression for language model pre-training. *Preprint*, arXiv:2407.01492.

Wei Liu, Weihao Zeng, and Keqing et al. He. 2023. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *arXiv preprint arXiv:2312.15685*.

Keming Lu, Hongyi Yuan, and Zheng et al. Yuan. 2023. # instag: Instruction tagging for analyzing supervised fine-tuning of large language models. In *The Twelfth International Conference on Learning Representations*.

James Martens and Roger Grosse. 2015. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417. PMLR.

Nick Mecklenburg, Yiyou Lin, and Xiaoxiao et al. Li. 2024. Injecting new knowledge into large language models via supervised fine-tuning. *arXiv preprint arXiv:2404.00213*.

Niklas Muennighoff, Alexander Rush, and Boaz et al. Barak. 2023. Scaling data-constrained language models. *Advances in Neural Information Processing Systems*, 36:50358–50376.

OpenAI, Josh Achiam, and Steven Adler et al. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Jupinder Parmar, Shrimai Prabhumoye, and Joseph et al. Jennings. 2024. Data, data everywhere: A guide for pretraining dataset construction. *arXiv preprint arXiv:2407.06380*.

Mirac Suzgun, Nathan Scales, and Nathanael et al. Schärli. 2022. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*.

Hugo Touvron, Thibaut Lavril, and Gautier et al. Izacard. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Yuichiro Ueno, Kazuki Osawa, and Yohei et al. Tsuji. 2020. Rich information is affordable: A systematic performance analysis of second-order optimization using k-fac. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2145–2153.

Yufei Wang, Wanjun Zhong, and Liangyou et al. Li. 2023. Aligning large language models with human: A survey. *arXiv preprint arXiv:2307.12966*.

Mengzhou Xia, Sadhika Malladi, and Suchin et al. Gururangan. 2024. Less: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*.

Sang Michael Xie, Hieu Pham, Xuanyi Dong, Nan Du, Hanxiao Liu, Yifeng Lu, Percy S Liang, Quoc V Le, Tengyu Ma, and Adams Wei Yu. 2024. Doremi: Optimizing data mixtures speeds up language model pretraining. *Advances in Neural Information Processing Systems*, 36.

Can Xu, Qingfeng Sun, and Kai et al. Zheng. 2023. Wizardlm: Empowering large language models to follow complex instructions. *arXiv preprint arXiv:2304.12244*.

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

Zichun Yu, Spandan Das, and Chenyan Xiong. 2024. Mates: Model-aware data selection for efficient pretraining with data influence models. *arXiv preprint arXiv:2406.06046*.

Chi Zhang, Huaping Zhong, and Kuan et al. Zhang. 2024a. Harnessing diversity for important data selection in pretraining large language models. *arXiv preprint arXiv:2409.16986*.

Hengyuan Zhang, Yanru Wu, and Dawei et al. Li. 2024b. Balancing speciality and versatility: a coarse to fine framework for supervised fine-tuning large language model. *arXiv preprint arXiv:2404.10306*.

Jeffrey Zhou, Tianjian Lu, and Swaroop Mishra et al. 2023. Instruction-following evaluation for large language models. *Preprint*, arXiv:2311.07911.

A Appendix

A.1 Dataset and Training Information

Reasoning: We selected BigBench Hard(BBH) as the benchmark to evaluate the reasoning capabilities of our model. BBH is a widely recognized benchmark designed to test a model's ability to handle complex and diverse reasoning tasks, making it an ideal choice for assessing the comprehensive reasoning skills of our model. To further enhance the quality of the training dataset, we utilized the official BBH dataset as a foundation and employed GPT-4(OpenAI et al., 2024) to regenerate the corresponding answers. This process allowed us to refine and improve the quality of the dataset, ensuring that the training examples are both accurate and high-quality.

Mathematics: We selected GSM8K as the benchmark to evaluate the mathematical reasoning capabilities of our model. GSM8K is a highly regarded dataset specifically designed to test models on a wide range of math problems, including arithmetic, algebra, and word problems. By covering diverse mathematical scenarios, GSM8K serves as a comprehensive tool for evaluating both the precision and depth of our model's mathematical understanding. We started with the official GSM8K dataset and leveraged GPT-4 to regenerate the corresponding chain-of-thought (CoT) solutions. This approach allowed us to refine the reasoning steps and enhance the clarity and accuracy of the solutions.

Coding: We chose HumanEval as the benchmark to assess the coding capabilities of our model. HumanEval is a well-known dataset specifically designed to evaluate a model's ability to understand, generate, and execute code. It provides a set of programming tasks that require not only syntactic correctness but also semantic understanding, logical reasoning, and problem-solving skills. However, due to the lack of an official training dataset for HumanEval, we constructed a high-quality training set by randomly sampling 5,000 examples from the Tulu-Code dataset(Ge et al., 2024) and combining them with the Mostly Basic Python problems(MBPP)(Austin et al., 2021) training set.

Instruction-following: We selected IFEval as the benchmark to evaluate our model's instruction-following abilities. IFEval is designed to test a model's capacity to understand and execute diverse instructions, making it ideal for assessing alignment with user intent across various scenarios. Due

to the limited size of the official IFEval training set, we enhanced it by sampling additional data from WizardLM Evol-Instruct data(Xu et al., 2023). This combination created a richer and more diverse training set, enabling our model to better generalize and excel in instruction-following tasks.

Training Details: In all fine-tuning training experiments, we set the batch size to 256 and the maximum learning rate as 2×10^{-5} with a cosine decay schedule. We train the base model on the training dataset for 1 epoch on 8 A100 GPUs and evaluate the result by using OpenCompass platform(Contributors, 2023).

Evaluation Metric. For GSM8K, we adopt the 'accuracy' metric. For the HumanEval benchmark, we use the 'pass@1' metric to evaluate the probability that the code generated by the model in a single attempt successfully compiles. For the BBH benchmark, we consider the naive average metric to evaluate the average score of the model across multiple test capabilities in BBH. In IFEval, we adopt four metrics, namely prompt-level-strict-acc(P-s-acc), Inst-level-strict-acc(I-s-acc), prompt-level-loose-acc(P-l-acc), and Inst-level-loose-acc(I-l-acc), to comprehensively evaluate the model's capabilities in detail. The P-s-acc metric assesses the accuracy of the model's responses at the prompt level with strict criteria, while the I-s-acc evaluates the accuracy at the instance level with strict standards. On the other hand, the P-l-acc and I-l-acc use more relaxed criteria for evaluation at the prompt and instance levels respectively. After obtaining these four values, we calculate their average to enable quick comparison among different models or experimental results, which provides a straightforward way to gauge the overall performance of the model in a more comprehensive manner.

A.2 Proof of Lemma 1

Proof 1 *Since we assume in* (1) *that the model is trained to optimality on the training set $\mathcal{D}_{tr}$, we can obtain the following:*

$$\sum_{z_i \in \mathcal{D}_{tr}} \nabla_\theta \mathcal{L}(z_i, \theta^*) + \sum_{j=1}^{n} \beta_j \nabla_\theta \mathcal{L}(\mathcal{D}_j, \theta^*) = 0. \quad (6)$$

We assume that there is a small perturbation error ϵ in β_m, and we denote $\beta_m \to \beta_m + \epsilon$. Correspondingly, θ^* will also change: $\theta^* \to \theta^* + \Delta\theta$, the (6) *transforms to:*

$$\sum_{z_i \in \mathcal{D}_{tr}} \nabla_\theta \mathcal{L}(z_i, \theta^* + \Delta\theta) + \sum_{j=1}^{n} \beta_j \nabla_\theta \mathcal{L}(\mathcal{D}_j, \theta^* + \Delta\theta) + \epsilon \nabla_\theta \mathcal{L}(\mathcal{D}_m, \theta^* + \Delta\theta) = 0. \quad (7)$$

Combined with the Taylor expansion, we can obtain the following:

$$\sum_{z_i \in \mathcal{D}_{tr}} \nabla^2_\theta \mathcal{L}(z_i, \theta^*)\Delta\theta + \sum_{j=1}^{n} \beta_j \nabla^2_\theta \mathcal{L}(\mathcal{D}_j, \theta^*)\Delta\theta + \epsilon \nabla_\theta \mathcal{L}(\mathcal{D}_m, \theta^*) = 0. \quad (8)$$

By using $\Delta\theta = \frac{\partial\theta^*}{\partial\beta_m}\epsilon$, we can get:

$$\sum_{z_i \in \mathcal{D}_{tr}} \nabla^2 \mathcal{L}(z_i, \theta^*)\frac{\partial\theta^*}{\partial\beta_m} + \sum_{j=1}^{n} \beta_j \nabla^2 \mathcal{L}(\mathcal{D}_j, \theta^*)\frac{\partial\theta^*}{\partial\beta_m} + \nabla \mathcal{L}(\mathcal{D}_m, \theta^*) = 0, \quad (9)$$

which implies (3).

A.3 Estimation Error Analysis

In the process of implementing the IDEAL method, several factors may lead to estimation inaccuracies, which can potentially affect the overall performance and reliability of the proposed approach.

Sub-optimality in Model Training. The IDEAL method assumes that the model is trained to optimality on the training set $\mathcal{D}_{tr}$ as per Equation (3.1). However, in practical scenarios, to prevent overfitting, models are typically not trained to reach the globally optimal parameters. Instead, a balance is struck to obtain sub-optimal parameters that ensure good generalization across different domains. When the model is not trained to its full potential, the gradients and Hessian-related calculations used in our method, such as those in Lemma 1 for calculating $\frac{\partial\theta^*}{\partial\beta_m}$, may not accurately represent the true behavior of the model at its optimal state. This deviation from the ideal training condition can introduce errors in the determination of the optimal mixing ratio β for the training datasets.

Methodological Errors from K-FAC for Hessian Matrix Computation. To enable efficient calculation of the influence function, we rely on the K-FAC theory to decompose the Hessian matrix. As described in Section 3.2, we approximate the Hessian matrix $\mathbf{H}$ by decomposing it into a block-diagonal form according to different MLP layers. While this approximation significantly accelerates the inversion of the second-order gradient matrix, it inevitably introduces methodological errors. The block-diagonal approximation, where

$\mathbf{H}^l \approx \mathbb{E}(x^l x^{l^\top}) \otimes \mathbb{E}(\delta^l \delta^{l^\top})$, simplifies the complex structure of the true Hessian matrix. However, this simplification means that the calculated influence function may deviate from the exact value. When inverting the approximated Hessian matrix to calculate $\frac{\partial \theta^*}{\partial \beta_m}$ in Equation (3), these errors can propagate through the subsequent calculations of $\frac{\partial \mathcal{L}(\mathcal{D}_{ref}, \theta^*)}{\partial \beta_m}$ in Equation (4).

Experimental Errors due to Random Sampling for Accelerated Computation. To expedite the computational process, we resort to random sampling from the training set. Although the law of large numbers assures that the mean of a large number of independent and identically distributed random samples converges to the expected value of the population, there is still a possibility of introducing random biases. In our method, when calculating expectations such as those in the decomposition of the Hessian matrix, the use of sampled data instead of the entire dataset can lead to errors. These random biases potentially result in inaccurate final results.